libgpiodcxx

Generated by Doxygen 1.9.4

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 5

Namespace Documentation

5.1 gpiod::line Namespace Reference

Namespace containing various type definitions for GPIO lines.

Classes

- class offset

  Wrapper around unsigned int for representing line offsets.

Typedefs

- using **offsets** = ::std::vector< offset >

  Vector of line offsets.
- using **values** = ::std::vector< value >

  Vector of line values.
- using **value_mapping** = ::std::pair< offset, value >

  Represents a mapping of a line offset to line logical state.
- using **value_mappings** = ::std::vector< value_mapping >

  Vector of offset->value mappings. Each mapping is defined as a pair of an unsigned and signed integers.

Enumerations

- enum class value { INACTIVE = 0 , ACTIVE = 1 }

  Logical line states.
- enum class direction { AS_IS = 1 , INPUT , OUTPUT }

  Direction settings.
- enum class edge { NONE = 1 , RISING , FALLING , BOTH }

  Edge detection settings.
- enum class bias {
  AS_IS = 1 , UNKNOWN , DISABLED , PULL_UP ,
  PULL_DOWN }

  Internal bias settings.
- enum class drive { PUSH_PULL = 1 , OPEN_DRAIN , OPEN_SOURCE }

  Drive settings.
- enum class clock { MONOTONIC = 1 , REALTIME , **HTE** }

  Event clock settings.

Functions

- ::std::ostream & operator<< (::std::ostream &out, value val)

  Stream insertion operator for logical line values.
- ::std::ostream & operator<< (::std::ostream &out, direction dir)

  Stream insertion operator for direction values.
- ::std::ostream & operator<< (::std::ostream &out, edge edge)

  Stream insertion operator for edge detection values.
- ::std::ostream & operator<< (::std::ostream &out, bias bias)

  Stream insertion operator for bias values.
- ::std::ostream & operator<< (::std::ostream &out, drive drive)

  Stream insertion operator for drive values.
- ::std::ostream & operator<< (::std::ostream &out, clock clock)

  Stream insertion operator for event clock values.
- ::std::ostream & operator<< (::std::ostream &out, const values &vals)

  Stream insertion operator for the list of output values.
- ::std::ostream & operator<< (::std::ostream &out, const offsets &offs)

  Stream insertion operator for the list of line offsets.
- ::std::ostream & operator<< (::std::ostream &out, const value_mapping &mapping)

  Stream insertion operator for the offset-to-value mapping.
- ::std::ostream & operator<< (::std::ostream &out, const value_mappings &mappings)

  Stream insertion operator for the list of offset-to-value mappings.

5.1.1 Detailed Description

Namespace containing various type definitions for GPIO lines.

5.1.2 Enumeration Type Documentation

5.1.2.1 bias

```
enum class gpiod::line::bias  [strong]
```

Internal bias settings.

Enumerator

AS_IS	Don't change the bias setting when applying line config.
UNKNOWN	The internal bias state is unknown.
DISABLED	The internal bias is disabled.
PULL_UP	The internal pull-up bias is enabled.
PULL_DOWN	The internal pull-down bias is enabled.

5.1.2.2 clock

```
enum class gpiod::line::clock  [strong]
```

Event clock settings.

Enumerator

MONOTONIC	Line uses the monotonic clock for edge event timestamps.
REALTIME	Line uses the realtime clock for edge event timestamps.

5.1.2.3 direction

```
enum class gpiod::line::direction  [strong]
```

Direction settings.

Enumerator

AS_IS	Request the line(s), but don't change current direction.
INPUT	Direction is input - we're reading the state of a GPIO line.
OUTPUT	Direction is output - we're driving the GPIO line.

5.1.2.4 drive

```
enum class gpiod::line::drive  [strong]
```

Drive settings.

Enumerator

PUSH_PULL	Drive setting is push-pull.
OPEN_DRAIN	Line output is open-drain.
OPEN_SOURCE	Line output is open-source.

5.1.2.5 edge

```
enum class gpiod::line::edge  [strong]
```

Edge detection settings.

Enumerator

NONE	Line edge detection is disabled.
RISING	Line detects rising edge events.
FALLING	Line detect falling edge events.
BOTH	Line detects both rising and falling edge events.

5.1.2.6 value

```
enum class gpiod::line::value  [strong]
```

Logical line states.

Enumerator

INACTIVE	Line is inactive.
ACTIVE	Line is active.

5.1.3 Function Documentation

5.1.3.1 operator<<() [1/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            bias bias )
```

Stream insertion operator for bias values.

Parameters

out	Output stream.
bias	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.2 operator<<() [2/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            clock clock )
```

Stream insertion operator for event clock values.

Parameters

out	Output stream.
clock	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.3 operator<<() [3/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            const offsets & offs )
```

Stream insertion operator for the list of line offsets.

Parameters

out	Output stream.
offs	Object to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.4 operator<<() [4/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            const value_mapping & mapping )
```

Stream insertion operator for the offset-to-value mapping.

Parameters

out	Output stream.
mapping	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.5 operator<<() [5/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            const value_mappings & mappings )
```

Stream insertion operator for the list of offset-to-value mappings.

Parameters

out	Output stream.
mappings	Object to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.6 operator<<() [6/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            const values & vals )
```

Stream insertion operator for the list of output values.

Parameters

out	Output stream.
vals	Object to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.7 operator<<() [7/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            direction dir )
```

Stream insertion operator for direction values.

Parameters

out	Output stream.
dir	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.8 operator<<() [8/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            drive drive )
```

Stream insertion operator for drive values.

Parameters

out	Output stream.
drive	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.9 operator<<() [9/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            edge edge )
```

Stream insertion operator for edge detection values.

Parameters

out	Output stream.
edge	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

5.1.3.10 operator<<() [10/10]

```
::std::ostream & gpiod::line::operator<< (
            ::std::ostream & out,
            value val )
```

Stream insertion operator for logical line values.

Parameters

out	Output stream.
val	Value to insert into the output stream in a human-readable form.

Returns

Reference to out.

Chapter 6

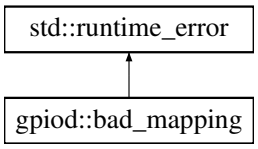
Class Documentation

6.1 gpiod::bad_mapping Class Reference

Exception thrown when the core C library returns an invalid value for any of the line_info properties.

`#include <exception.hpp>`

Inheritance diagram for gpiod::bad_mapping:



Public Member Functions

- bad_mapping (const ::std::string &what)
  Constructor.
- bad_mapping (const bad_mapping &other) noexcept
  Copy constructor.
- bad_mapping (bad_mapping &&other) noexcept
  Move constructor.
- bad_mapping & operator= (const bad_mapping &other) noexcept
  Assignment operator.
- bad_mapping & operator= (bad_mapping &&other) noexcept
  Move assignment operator.

6.1.1 Detailed Description

Exception thrown when the core C library returns an invalid value for any of the line_info properties.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 bad_mapping() [1/3]

```
gpiod::bad_mapping::bad_mapping (
            const ::std::string & what )  [explicit]
```

Constructor.

Parameters

what	Human readable reason for error.

6.1.2.2 bad_mapping() [2/3]

```
gpiod::bad_mapping::bad_mapping (
            const bad_mapping & other )  [noexcept]
```

Copy constructor.

Parameters

other	Object to copy from.

6.1.2.3 bad_mapping() [3/3]

```
gpiod::bad_mapping::bad_mapping (
            bad_mapping && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.1.3 Member Function Documentation

6.1.3.1 operator=() [1/2]

```
bad_mapping & gpiod::bad_mapping::operator= (
            bad_mapping && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.1.3.2 operator=() [2/2]

```
bad_mapping & gpiod::bad_mapping::operator= (
            const bad_mapping & other )  [noexcept]
```

Assignment operator.

Parameters

other	Object to copy from.

Returns

Reference to self.

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

- gpiodcxx/exception.hpp

6.2 gpiod::chip Class Reference

Represents a GPIO chip.

```
#include <chip.hpp>
```

Public Member Functions

- chip (const ::std::filesystem::path &path)

  Instantiates a new chip object by opening the device file indicated by the `path` argument.
- chip (chip &&other) noexcept

  Move constructor.
- chip & **operator=** (const chip &other)=delete
- chip & operator= (chip &&other) noexcept

  Move assignment operator.
- operator bool () const noexcept

  Check if this object is valid.
- void close ()

  Close the GPIO chip device file and free associated resources.
- ::std::filesystem::path path () const

  Get the filesystem path that was used to open this GPIO chip.
- chip_info get_info () const

  Get information about the chip.
- line_info get_line_info (line::offset offset) const

  Retrieve the current snapshot of line information for a single line.
- line_info watch_line_info (line::offset offset) const

  Wrapper around gpiod::chip::get_line_info that retrieves the line info and starts watching the line for changes.
- void unwatch_line_info (line::offset offset) const

  Stop watching the line at given offset for info events.
- int fd () const

  Get the file descriptor associated with this chip.
- bool wait_info_event (const ::std::chrono::nanoseconds &timeout) const

  Wait for line status events on any of the watched lines exposed by this chip.
- info_event read_info_event () const

  Read a single line status change event from this chip.
- int get_line_offset_from_name (const ::std::string &name) const

  Map a GPIO line's name to its offset within the chip.
- request_builder prepare_request ()

  Create a request_builder associated with this chip.

Private Member Functions

- **chip** (const chip &other)

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **request_builder**

6.2.1 Detailed Description

Represents a GPIO chip.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 chip() [1/2]

```
gpiod::chip::chip (
            const ::std::filesystem::path & path )  [explicit]
```

Instantiates a new chip object by opening the device file indicated by the `path` argument.

Parameters

path	Path to the device file to open.

6.2.2.2 chip() [2/2]

```
gpiod::chip::chip (
            chip && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.2.3 Member Function Documentation

6.2.3.1 close()

```
void gpiod::chip::close ( )
```

Close the GPIO chip device file and free associated resources.

Note

The chip object can live after calling this method but any of the chip's mutators will throw a logic_error exception.

6.2.3.2 fd()

```
int gpiod::chip::fd ( ) const
```

Get the file descriptor associated with this chip.

Returns

File descriptor number.

6.2.3.3 get_info()

```
chip_info gpiod::chip::get_info ( ) const
```

Get information about the chip.

Returns

New chip_info object.

6.2.3.4 get_line_info()

```
line_info gpiod::chip::get_line_info (
            line::offset offset ) const
```

Retrieve the current snapshot of line information for a single line.

Parameters

offset	Offset of the line to get the info for.

Returns

New gpiod::line_info object.

6.2.3.5 get_line_offset_from_name()

```
int gpiod::chip::get_line_offset_from_name (
            const ::std::string & name ) const
```

Map a GPIO line's name to its offset within the chip.

Parameters

name	Name of the GPIO line to map.

Returns

Offset of the line within the chip or -1 if the line with given name is not exposed by this chip.

6.2.3.6 operator bool()

```
gpiod::chip::operator bool ( ) const  [explicit], [noexcept]
```

Check if this object is valid.

Returns

True if this object's methods can be used, false otherwise. False usually means the chip was closed. If the user calls any of the methods of this class on an object for which this operator returned false, a logic_error will be thrown.

6.2.3.7 operator=()

```
chip & gpiod::chip::operator= (
            chip && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.2.3.8 path()

```
::std::filesystem::path gpiod::chip::path ( ) const
```

Get the filesystem path that was used to open this GPIO chip.

Returns

Path to the underlying character device file.

6.2.3.9 prepare_request()

```
request_builder gpiod::chip::prepare_request ( )
```

Create a request_builder associated with this chip.

Returns

New request_builder object.

6.2.3.10 read_info_event()

```
info_event gpiod::chip::read_info_event ( ) const
```

Read a single line status change event from this chip.

Returns

New info_event object.

6.2.3.11 unwatch_line_info()

```
void gpiod::chip::unwatch_line_info (
            line::offset offset ) const
```

Stop watching the line at given offset for info events.

Parameters

offset	Offset of the line to get the info for.

6.2.3.12 wait_info_event()

```
bool gpiod::chip::wait_info_event (
            const ::std::chrono::nanoseconds & timeout ) const
```

Wait for line status events on any of the watched lines exposed by this chip.

Parameters

timeout	Wait time limit in nanoseconds. If set to 0, the function returns immediately. If set to a negative number, the function blocks indefinitely until an event becomes available.

Returns

True if at least one event is ready to be read. False if the wait timed out.

6.2.3.13 watch_line_info()

```
line_info gpiod::chip::watch_line_info (
            line::offset offset ) const
```

Wrapper around gpiod::chip::get_line_info that retrieves the line info and starts watching the line for changes.

Parameters

offset	Offset of the line to get the info for.

Returns

New gpiod::line_info object.

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

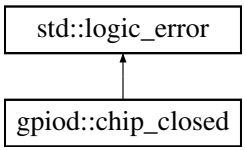
- gpiodcxx/chip.hpp

6.3 gpiod::chip_closed Class Reference

Exception thrown when an already closed chip is used.

`#include <exception.hpp>`

Inheritance diagram for gpiod::chip_closed:



Public Member Functions

- chip_closed (const ::std::string &what)
  Constructor.
- chip_closed (const chip_closed &other) noexcept
  Copy constructor.
- chip_closed (chip_closed &&other) noexcept
  Move constructor.
- chip_closed & operator= (const chip_closed &other) noexcept
  Assignment operator.
- chip_closed & operator= (chip_closed &&other) noexcept
  Move assignment operator.

6.3.1 Detailed Description

Exception thrown when an already closed chip is used.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 chip_closed() [1/3]

```
gpiod::chip_closed::chip_closed (
            const ::std::string & what )  [explicit]
```

Constructor.

Parameters

what	Human readable reason for error.

6.3.2.2 chip_closed() [2/3]

```
gpiod::chip_closed::chip_closed (
            const chip_closed & other )  [noexcept]
```

Copy constructor.

Parameters

other	Object to copy from.

6.3.2.3 chip_closed() [3/3]

```
gpiod::chip_closed::chip_closed (
            chip_closed && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.3.3 Member Function Documentation

6.3.3.1 operator=() [1/2]

```
chip_closed & gpiod::chip_closed::operator= (
            chip_closed && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.3.3.2 operator=() [2/2]

```
chip_closed & gpiod::chip_closed::operator= (
            const chip_closed & other )  [noexcept]
```

Assignment operator.

Parameters

other	Object to copy from.

Returns

Reference to self.

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

- gpiodcxx/exception.hpp

6.4 gpiod::chip_info Class Reference

Represents an immutable snapshot of GPIO chip information.

```
#include <chip-info.hpp>
```

Public Member Functions

- chip_info (const chip_info &other)

  Copy constructor.
- chip_info (chip_info &&other) noexcept

  Move constructor.
- chip_info & operator= (const chip_info &other)

  Assignment operator.
- chip_info & operator= (chip_info &&other) noexcept

  Move assignment operator.
- ::std::string name () const noexcept

  Get the name of this GPIO chip.
- ::std::string label () const noexcept

  Get the label of this GPIO chip.
- ::std::size_t num_lines () const noexcept

  Return the number of lines exposed by this chip.

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **chip**

6.4.1 Detailed Description

Represents an immutable snapshot of GPIO chip information.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 chip_info() [1/2]

```
gpiod::chip_info::chip_info (
            const chip_info & other )
```

Copy constructor.

Parameters

other	Object to copy.

6.4.2.2 chip_info() [2/2]

```
gpiod::chip_info::chip_info (
            chip_info && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.4.3 Member Function Documentation

6.4.3.1 label()

```
::std::string gpiod::chip_info::label ( ) const  [noexcept]
```

Get the label of this GPIO chip.

Returns

String containing the chip name.

6.4.3.2 name()

```
::std::string gpiod::chip_info::name ( ) const  [noexcept]
```

Get the name of this GPIO chip.

Returns

String containing the chip name.

6.4.3.3 num_lines()

```
::std::size_t gpiod::chip_info::num_lines ( ) const  [noexcept]
```

Return the number of lines exposed by this chip.

Returns

Number of lines.

6.4.3.4 operator=() [1/2]

```
chip_info & gpiod::chip_info::operator= (
            chip_info && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.4.3.5 operator=() [2/2]

```
chip_info & gpiod::chip_info::operator= (
            const chip_info & other )
```

Assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

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

- gpiodcxx/chip-info.hpp

6.5 gpiod::edge_event Class Reference

Immutable object containing data about a single edge event.

```
#include <edge-event.hpp>
```

Public Types

- enum class event_type { RISING_EDGE = 1 , FALLING_EDGE }

  Edge event types.

Public Member Functions

- edge_event (const edge_event &other)

  Copy constructor.
- edge_event (edge_event &&other) noexcept

  Move constructor.
- edge_event & operator= (const edge_event &other)

  Copy assignment operator.
- edge_event & operator= (edge_event &&other) noexcept

  Move assignment operator.
- event_type type () const

  Retrieve the event type.
- timestamp timestamp_ns () const noexcept

  Retrieve the event time-stamp.
- line::offset line_offset () const noexcept

  Read the offset of the line on which this event was registered.
- unsigned long global_seqno () const noexcept

  Get the global sequence number of this event.
- unsigned long line_seqno () const noexcept

  Get the event sequence number specific to the concerned line.

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **edge_event_buffer**

6.5.1 Detailed Description

Immutable object containing data about a single edge event.

6.5.2 Member Enumeration Documentation

6.5.2.1 event_type

```
enum class gpiod::edge_event::event_type  [strong]
```

Edge event types.

Enumerator

RISING_EDGE	This is a rising edge event.
FALLING_EDGE	This is falling edge event.

6.5.3 Constructor & Destructor Documentation

6.5.3.1 edge_event() [1/2]

```
gpiod::edge_event::edge_event (
            const edge_event & other )
```

Copy constructor.

Parameters

other	Object to copy.

6.5.3.2 edge_event() [2/2]

```
gpiod::edge_event::edge_event (
            edge_event && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.5.4 Member Function Documentation

6.5.4.1 global_seqno()

```
unsigned long gpiod::edge_event::global_seqno ( ) const  [noexcept]
```

Get the global sequence number of this event.

Returns

Sequence number of the event relative to all lines in the associated line request.

6.5.4.2 line_offset()

```
line::offset gpiod::edge_event::line_offset ( ) const  [noexcept]
```

Read the offset of the line on which this event was registered.

Returns

Line offset.

6.5.4.3 line_seqno()

```
unsigned long gpiod::edge_event::line_seqno ( ) const  [noexcept]
```

Get the event sequence number specific to the concerned line.

Returns

Sequence number of the event relative to this line within the lifetime of the associated line request.

6.5.4.4 operator=() [1/2]

```
edge_event & gpiod::edge_event::operator= (
            const edge_event & other )
```

Copy assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.5.4.5 operator=() [2/2]

```
edge_event & gpiod::edge_event::operator= (
            edge_event && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.5.4.6 timestamp_ns()

```
timestamp gpiod::edge_event::timestamp_ns ( ) const  [noexcept]
```

Retrieve the event time-stamp.

Returns

Time-stamp in nanoseconds as registered by the kernel using the configured edge event clock.

6.5.4.7 type()

```
event_type gpiod::edge_event::type ( ) const
```

Retrieve the event type.

Returns

Event type (rising or falling edge).

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

- gpiodcxx/edge-event.hpp

6.6 gpiod::edge_event_buffer Class Reference

Object into which edge events are read for better performance.

```
#include <edge-event-buffer.hpp>
```

Public Types

- using **const_iterator** = ::std::vector< edge_event >::const_iterator

  Constant iterator for iterating over edge events stored in the buffer.

Public Member Functions

- edge_event_buffer (::std::size_t capacity=64)

  Constructor. Creates a new edge event buffer with given capacity.
- **edge_event_buffer** (const edge_event_buffer &other)=delete
- edge_event_buffer (edge_event_buffer &&other) noexcept

  Move constructor.
- edge_event_buffer & **operator=** (const edge_event_buffer &other)=delete
- edge_event_buffer & operator= (edge_event_buffer &&other) noexcept

  Move assignment operator.
- const edge_event & get_event (unsigned int index) const

  Get the constant reference to the edge event at given index.
- ::std::size_t num_events () const

  Get the number of edge events currently stored in the buffer.
- ::std::size_t capacity () const noexcept

  Maximum capacity of the buffer.
- const_iterator begin () const noexcept

  Get a constant iterator to the first edge event currently stored in the buffer.
- const_iterator end () const noexcept

  Get a constant iterator to the element after the last edge event in the buffer.

Private Attributes

- ::std::unique_ptr< impl > **_m_priv**
- friend **line_request**

6.6.1 Detailed Description

Object into which edge events are read for better performance.

The edge_event_buffer allows reading edge_event objects into an existing buffer which improves the performance by avoiding needless memory allocations.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 edge_event_buffer() [1/2]

```
gpiod::edge_event_buffer::edge_event_buffer (
            ::std::size_t capacity = 64 )  [explicit]
```

Constructor. Creates a new edge event buffer with given capacity.

Parameters

capacity	Capacity of the new buffer.

6.6.2.2 edge_event_buffer() [2/2]

```
gpiod::edge_event_buffer::edge_event_buffer (
            edge_event_buffer && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.6.3 Member Function Documentation

6.6.3.1 begin()

```
const_iterator gpiod::edge_event_buffer::begin ( ) const  [noexcept]
```

Get a constant iterator to the first edge event currently stored in the buffer.

Returns

Constant iterator to the first element.

6.6.3.2 capacity()

```
::std::size_t gpiod::edge_event_buffer::capacity ( ) const  [noexcept]
```

Maximum capacity of the buffer.

Returns

Buffer capacity.

6.6.3.3 end()

```
const_iterator gpiod::edge_event_buffer::end ( ) const  [noexcept]
```

Get a constant iterator to the element after the last edge event in the buffer.

Returns

Constant iterator to the element after the last edge event.

6.6.3.4 get_event()

```
const edge_event & gpiod::edge_event_buffer::get_event (
            unsigned int index ) const
```

Get the constant reference to the edge event at given index.

Parameters

index	Index of the event in the buffer.

Returns

Constant reference to the edge event.

6.6.3.5 num_events()

```
::std::size_t gpiod::edge_event_buffer::num_events ( ) const
```

Get the number of edge events currently stored in the buffer.

Returns

Number of edge events in the buffer.

6.6.3.6 operator=()

```
edge_event_buffer & gpiod::edge_event_buffer::operator= (
            edge_event_buffer && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

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

- gpiodcxx/edge-event-buffer.hpp

6.7 gpiod::info_event Class Reference

Immutable object containing data about a single line info event.

```
#include <info-event.hpp>
```

Public Types

- enum class event_type { LINE_REQUESTED = 1 , LINE_RELEASED , LINE_CONFIG_CHANGED }

  Types of info events.

Public Member Functions

- info_event (const info_event &other)

  Copy constructor.
- info_event (info_event &&other) noexcept

  Move constructor.
- info_event & operator= (const info_event &other)

  Copy assignment operator.
- info_event & operator= (info_event &&other) noexcept

  Move assignment operator.
- event_type type () const

  Type of this event.
- ::std::uint64_t timestamp_ns () const noexcept

  Timestamp of the event as returned by the kernel.
- const line_info & get_line_info () const noexcept

  Get the new line information.

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **chip**

6.7.1 Detailed Description

Immutable object containing data about a single line info event.

6.7.2 Member Enumeration Documentation

6.7.2.1 event_type

```
enum class gpiod::info_event::event_type  [strong]
```

Types of info events.

Enumerator

LINE_REQUESTED	Line has been requested.
LINE_RELEASED	Previously requested line has been released.
LINE_CONFIG_CHANGED	Line configuration has changed.

6.7.3 Constructor & Destructor Documentation

6.7.3.1 info_event() [1/2]

```
gpiod::info_event::info_event (
            const info_event & other )
```

Copy constructor.

Parameters

other	Object to copy.

6.7.3.2 info_event() [2/2]

```
gpiod::info_event::info_event (
            info_event && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.7.4 Member Function Documentation

6.7.4.1 get_line_info()

```
const line_info & gpiod::info_event::get_line_info ( ) const  [noexcept]
```

Get the new line information.

Returns

Constant reference to the line info object containing the line data as read at the time of the info event.

6.7.4.2 operator=() [1/2]

```
info_event & gpiod::info_event::operator= (
            const info_event & other )
```

Copy assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.7.4.3 operator=() [2/2]

```
info_event & gpiod::info_event::operator= (
            info_event && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.7.4.4 timestamp_ns()

```
::std::uint64_t gpiod::info_event::timestamp_ns ( ) const  [noexcept]
```

Timestamp of the event as returned by the kernel.

Returns

Timestamp as a 64-bit unsigned integer.

6.7.4.5 type()

```
event_type gpiod::info_event::type ( ) const
```

Type of this event.

Returns

Event type.

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

- gpiodcxx/info-event.hpp

6.8 gpiod::line_config Class Reference

Contains a set of line config options used in line requests and reconfiguration.

`#include <line-config.hpp>`

Public Member Functions

- **line_config** (const line_config &other)=delete
- line_config (line_config &&other) noexcept

  Move constructor.
- line_config & operator= (line_config &&other) noexcept

  Move assignment operator.
- line_config & reset () noexcept

  Reset the line config object.
- line_config & add_line_settings (line::offset offset, const line_settings &settings)

  Add line settings for a single offset.
- line_config & add_line_settings (const line::offsets &offsets, const line_settings &settings)

  Add line settings for a set of offsets.
- line_config & set_output_values (const line::values &values)

  Set output values for a number of lines.
- ::std::map< line::offset, line_settings > get_line_settings () const

  Get a mapping of offsets to line settings stored by this object.

Private Member Functions

- line_config & **operator=** (const line_config &other)

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **line_request**
- friend **request_builder**

6.8.1 Detailed Description

Contains a set of line config options used in line requests and reconfiguration.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 line_config()

```
gpiod::line_config::line_config (
            line_config && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.8.3 Member Function Documentation

6.8.3.1 add_line_settings() [1/2]

```
line_config & gpiod::line_config::add_line_settings (
            const line::offsets & offsets,
            const line_settings & settings )
```

Add line settings for a set of offsets.

Parameters

offsets	Offsets for which to add settings.
settings	Line settings to add.

Returns

Reference to self.

6.8.3.2 add_line_settings() [2/2]

```
line_config & gpiod::line_config::add_line_settings (
            line::offset offset,
            const line_settings & settings )
```

Add line settings for a single offset.

Parameters

offset	Offset for which to add settings.
settings	Line settings to add.

Returns

Reference to self.

6.8.3.3 get_line_settings()

```
::std::map< line::offset, line_settings > gpiod::line_config::get_line_settings ( ) const
```

Get a mapping of offsets to line settings stored by this object.

Returns

Map in which keys represent line offsets and values are the settings corresponding with them.

6.8.3.4 operator=()

```
line_config & gpiod::line_config::operator= (
            line_config && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.8.3.5 reset()

```
line_config & gpiod::line_config::reset ( )  [noexcept]
```

Reset the line config object.

Returns

Reference to self.

6.8.3.6 set_output_values()

```
line_config & gpiod::line_config::set_output_values (
            const line::values & values )
```

Set output values for a number of lines.

Parameters

values	Buffer containing the output values.

Returns

Reference to self.

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

- gpiodcxx/line-config.hpp

6.9 gpiod::line_info Class Reference

Contains an immutable snapshot of the line's state at the time when the object of this class was instantiated.

`#include <line-info.hpp>`

Public Member Functions

- line_info (const line_info &other) noexcept
  Copy constructor.
- line_info (line_info &&other) noexcept
  Move constructor.
- line_info & operator= (const line_info &other) noexcept
  Copy assignment operator.
- line_info & operator= (line_info &&other) noexcept
  Move assignment operator.
- line::offset offset () const noexcept
  Get the hardware offset of the line.
- ::std::string name () const noexcept
  Get the GPIO line name.
- bool used () const noexcept
  Check if the line is currently in use.
- ::std::string consumer () const noexcept
  Read the GPIO line consumer name.
- line::direction direction () const
  Read the GPIO line direction setting.
- line::edge edge_detection () const
  Read the current edge detection setting of this line.
- line::bias bias () const
  Read the GPIO line bias setting.
- line::drive drive () const
  Read the GPIO line drive setting.
- bool active_low () const noexcept
  Check if the signal of this line is inverted.
- bool debounced () const noexcept
  Check if this line is debounced (either by hardware or by the kernel software debouncer).
- ::std::chrono::microseconds debounce_period () const noexcept
  Read the current debounce period in microseconds.
- line::clock event_clock () const
  Read the current event clock setting used for edge event timestamps.

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **chip**
- friend **info_event**

6.9.1 Detailed Description

Contains an immutable snapshot of the line's state at the time when the object of this class was instantiated.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 line_info() [1/2]

```
gpiod::line_info::line_info (
            const line_info & other )  [noexcept]
```

Copy constructor.

Parameters

other	Object to copy.

6.9.2.2 line_info() [2/2]

```
gpiod::line_info::line_info (
            line_info && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.9.3 Member Function Documentation

6.9.3.1 active_low()

```
bool gpiod::line_info::active_low ( ) const  [noexcept]
```

Check if the signal of this line is inverted.

Returns

True if this line is "active-low", false otherwise.

6.9.3.2 bias()

```
line::bias gpiod::line_info::bias ( ) const
```

Read the GPIO line bias setting.

Returns

Returns BIAS_PULL_UP, BIAS_PULL_DOWN, BIAS_DISABLE or BIAS_UNKNOWN.

6.9.3.3 consumer()

```
::std::string gpiod::line_info::consumer ( ) const  [noexcept]
```

Read the GPIO line consumer name.

Returns

Name of the GPIO consumer name as it is represented in the kernel. This routine returns an empty string if the line is not used.

6.9.3.4 debounce_period()

```
::std::chrono::microseconds gpiod::line_info::debounce_period ( ) const  [noexcept]
```

Read the current debounce period in microseconds.

Returns

Current debounce period in microseconds, 0 if the line is not debounced.

6.9.3.5 debounced()

```
bool gpiod::line_info::debounced ( ) const  [noexcept]
```

Check if this line is debounced (either by hardware or by the kernel software debouncer).

Returns

True if the line is debounced, false otherwise.

6.9.3.6 direction()

```
line::direction gpiod::line_info::direction ( ) const
```

Read the GPIO line direction setting.

Returns

Returns DIRECTION_INPUT or DIRECTION_OUTPUT.

6.9.3.7 drive()

```
line::drive gpiod::line_info::drive ( ) const
```

Read the GPIO line drive setting.

Returns

Returns DRIVE_PUSH_PULL, DRIVE_OPEN_DRAIN or DRIVE_OPEN_SOURCE.

6.9.3.8 edge_detection()

```
line::edge gpiod::line_info::edge_detection ( ) const
```

Read the current edge detection setting of this line.

Returns

Returns EDGE_NONE, EDGE_RISING, EDGE_FALLING or EDGE_BOTH.

6.9.3.9 event_clock()

```
line::clock gpiod::line_info::event_clock ( ) const
```

Read the current event clock setting used for edge event timestamps.

Returns

Returns MONOTONIC, REALTIME or HTE.

6.9.3.10 name()

```
::std::string gpiod::line_info::name ( ) const  [noexcept]
```

Get the GPIO line name.

Returns

Name of the GPIO line as it is represented in the kernel. This routine returns an empty string if the line is unnamed.

6.9.3.11 offset()

```
line::offset gpiod::line_info::offset ( ) const  [noexcept]
```

Get the hardware offset of the line.

Returns

Offset of the line within the parent chip.

6.9.3.12 operator=() [1/2]

```
line_info & gpiod::line_info::operator= (
            const line_info & other )  [noexcept]
```

Copy assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.9.3.13 operator=() [2/2]

```
line_info & gpiod::line_info::operator= (
            line_info && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.9.3.14 used()

```
bool gpiod::line_info::used ( ) const  [noexcept]
```

Check if the line is currently in use.

Returns

True if the line is in use, false otherwise.

The user space can't know exactly why a line is busy. It may have been requested by another process or hogged by the kernel. It only matters that the line is used and we can't request it.

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

- gpiodcxx/line-info.hpp

6.10 gpiod::line_request Class Reference

Stores the context of a set of requested GPIO lines.

```
#include <line-request.hpp>
```

Public Member Functions

- **line_request** (const line_request &other)=delete
- line_request (line_request &&other) noexcept

  Move constructor.
- line_request & **operator=** (const line_request &other)=delete
- line_request & operator= (line_request &&other) noexcept

  Move assignment operator.
- operator bool () const noexcept

  Check if this object is valid.
- void release ()

  Release the requested lines and free all associated resources.
- ::std::string chip_name () const

  Get the name of the chip this request was made on.
- ::std::size_t num_lines () const

  Get the number of requested lines.
- line::offsets offsets () const

  Get the list of offsets of requested lines.
- line::value get_value (line::offset offset)

  Get the value of a single requested line.
- line::values get_values (const line::offsets &offsets)

  Get the values of a subset of requested lines.
- line::values get_values ()

  Get the values of all requested lines.
- void get_values (const line::offsets &offsets, line::values &values)

  Get the values of a subset of requested lines into a vector supplied by the caller.
- void get_values (line::values &values)

  Get the values of all requested lines.
- line_request & set_value (line::offset offset, line::value value)

  Set the value of a single requested line.
- line_request & set_values (const line::value_mappings &values)

  Set the values of a subset of requested lines.
- line_request & set_values (const line::offsets &offsets, const line::values &values)

  Set the values of a subset of requested lines.
- line_request & set_values (const line::values &values)

  Set the values of all requested lines.
- line_request & reconfigure_lines (const line_config &config)

  Apply new config options to requested lines.
- int fd () const

  Get the file descriptor number associated with this line request.
- bool wait_edge_events (const ::std::chrono::nanoseconds &timeout) const

  Wait for edge events on any of the lines requested with edge detection enabled.
- ::std::size_t read_edge_events (edge_event_buffer &buffer)

  Read a number of edge events from this request up to the maximum capacity of the buffer.
- ::std::size_t read_edge_events (edge_event_buffer &buffer, ::std::size_t max_events)

  Read a number of edge events from this request.

Private Attributes

- ::std::unique_ptr< impl > **_m_priv**
- friend **request_builder**

6.10.1 Detailed Description

Stores the context of a set of requested GPIO lines.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 line_request()

```
gpiod::line_request::line_request (
            line_request && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.10.3 Member Function Documentation

6.10.3.1 chip_name()

```
::std::string gpiod::line_request::chip_name ( ) const
```

Get the name of the chip this request was made on.

Returns

Name to the GPIO chip.

6.10.3.2 fd()

```
int gpiod::line_request::fd ( ) const
```

Get the file descriptor number associated with this line request.

Returns

File descriptor number.

6.10.3.3 get_value()

```
line::value gpiod::line_request::get_value (
            line::offset offset )
```

Get the value of a single requested line.

Parameters

offset	Offset of the line to read within the chip.

Returns

Current line value.

6.10.3.4 get_values() [1/4]

```
line::values gpiod::line_request::get_values ( )
```

Get the values of all requested lines.

Returns

List of read values.

6.10.3.5 get_values() [2/4]

```
line::values gpiod::line_request::get_values (
            const line::offsets & offsets )
```

Get the values of a subset of requested lines.

Parameters

offsets	Vector of line offsets

Returns

Vector of lines values with indexes of values corresponding to those of the offsets.

6.10.3.6 get_values() [3/4]

```
void gpiod::line_request::get_values (
            const line::offsets & offsets,
            line::values & values )
```

Get the values of a subset of requested lines into a vector supplied by the caller.

Parameters

offsets	Vector of line offsets.
values	Vector for storing the values. Its size must be at least that of the offsets vector. The indexes of read values will correspond with those in the offsets vector.

6.10.3.7 get_values() [4/4]

```
void gpiod::line_request::get_values (
            line::values & values )
```

Get the values of all requested lines.

Parameters

values	Array in which the values will be stored. Must hold at least the number of elements returned by line_request::num_lines.

6.10.3.8 num_lines()

```
::std::size_t gpiod::line_request::num_lines ( ) const
```

Get the number of requested lines.

Returns

Number of lines in this request.

6.10.3.9 offsets()

```
line::offsets gpiod::line_request::offsets ( ) const
```

Get the list of offsets of requested lines.

Returns

List of hardware offsets of the lines in this request.

6.10.3.10 operator bool()

```
gpiod::line_request::operator bool ( ) const  [explicit], [noexcept]
```

Check if this object is valid.

Returns

True if this object's methods can be used, false otherwise. False usually means the request was released. If the user calls any of the methods of this class on an object for which this operator returned false, a logic_error will be thrown.

6.10.3.11 operator=()

```
line_request & gpiod::line_request::operator= (
            line_request && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.10.3.12 read_edge_events() [1/2]

```
::std::size_t gpiod::line_request::read_edge_events (
            edge_event_buffer & buffer )
```

Read a number of edge events from this request up to the maximum capacity of the buffer.

Parameters

buffer	Edge event buffer to read events into.

Returns

Number of events read.

6.10.3.13 read_edge_events() [2/2]

```
::std::size_t gpiod::line_request::read_edge_events (
            edge_event_buffer & buffer,
            ::std::size_t max_events )
```

Read a number of edge events from this request.

Parameters

buffer	Edge event buffer to read events into.
max_events	Maximum number of events to read. Limited by the capacity of the buffer.

Returns

Number of events read.

6.10.3.14 reconfigure_lines()

```
line_request & gpiod::line_request::reconfigure_lines (
            const line_config & config )
```

Apply new config options to requested lines.

Parameters

config	New configuration.

Returns

Reference to self.

6.10.3.15 release()

```
void gpiod::line_request::release ( )
```

Release the requested lines and free all associated resources.

Note

The object can still be used after this method is called but using any of the mutators will result in throwing a logic_error exception.

6.10.3.16 set_value()

```
line_request & gpiod::line_request::set_value (
            line::offset offset,
            line::value value )
```

Set the value of a single requested line.

Parameters

offset	Offset of the line to set within the chip.
value	New line value.

Returns

Reference to self.

6.10.3.17 set_values() [1/3]

```
line_request & gpiod::line_request::set_values (
            const line::offsets & offsets,
            const line::values & values )
```

Set the values of a subset of requested lines.

Parameters

offsets	Vector containing the offsets of lines to set.
values	Vector containing new values with indexes corresponding with those in the offsets vector.

Returns

Reference to self.

6.10.3.18 set_values() [2/3]

```
line_request & gpiod::line_request::set_values (
            const line::value_mappings & values )
```

Set the values of a subset of requested lines.

Parameters

values	Vector containing a set of offset->value mappings.

Returns

Reference to self.

6.10.3.19 set_values() [3/3]

```
line_request & gpiod::line_request::set_values (
            const line::values & values )
```

Set the values of all requested lines.

Parameters

values	Array of new line values. The size must be equal to the value returned by line_request::num_lines.

Returns

Reference to self.

6.10.3.20 wait_edge_events()

```
bool gpiod::line_request::wait_edge_events (
            const ::std::chrono::nanoseconds & timeout ) const
```

Wait for edge events on any of the lines requested with edge detection enabled.

Parameters

timeout	Wait time limit in nanoseconds. If set to 0, the function returns immediately. If set to a negative number, the function blocks indefinitely until an event becomes available.

Returns

True if at least one event is ready to be read. False if the wait timed out.

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

- gpiodcxx/line-request.hpp

6.11 gpiod::line_settings Class Reference

Stores GPIO line settings.

```
#include <line-settings.hpp>
```

Public Member Functions

- **line_settings** ()

  Initializes the line_settings object with default values.
- line_settings (const line_settings &other)

  Copy constructor.
- line_settings (line_settings &&other) noexcept

  Move constructor.
- line_settings & operator= (const line_settings &other)

  Copy assignment operator.
- line_settings & operator= (line_settings &&other)

  Move assignment operator.
- line_settings & reset () noexcept

  Reset the line settings to default values.
- line_settings & set_direction (line::direction direction)

  Set direction.
- line::direction direction () const

  Get direction.
- line_settings & set_edge_detection (line::edge edge)

  Set edge detection.
- line::edge edge_detection () const

  Get edge detection.
- line_settings & set_bias (line::bias bias)

  Set bias setting.
- line::bias bias () const

  Get bias setting.
- line_settings & set_drive (line::drive drive)

  Set drive setting.
- line::drive drive () const

  Get drive setting.
- line_settings & set_active_low (bool active_low)

  Set the active-low setting.
- bool active_low () const noexcept

  Get the active-low setting.
- line_settings & set_debounce_period (const ::std::chrono::microseconds &period)

  Set debounce period.
- ::std::chrono::microseconds debounce_period () const noexcept

  Get debounce period.
- line_settings & set_event_clock (line::clock event_clock)

  Set the event clock to use for edge event timestamps.
- line::clock event_clock () const

  Get the event clock used for edge event timestamps.
- line_settings & set_output_value (line::value value)

  Set the output value.
- line::value output_value () const

  Get the output value.

Private Attributes

- ::std::unique_ptr< impl > **_m_priv**
- friend **line_config**

6.11.1 Detailed Description

Stores GPIO line settings.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 line_settings() [1/2]

```
gpiod::line_settings::line_settings (
            const line_settings & other )
```

Copy constructor.

Parameters

other	Object to copy.

6.11.2.2 line_settings() [2/2]

```
gpiod::line_settings::line_settings (
            line_settings && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.11.3 Member Function Documentation

6.11.3.1 active_low()

```
bool gpiod::line_settings::active_low ( ) const  [noexcept]
```

Get the active-low setting.

Returns

Current active-low setting.

6.11.3.2 bias()

```
line::bias gpiod::line_settings::bias ( ) const
```

Get bias setting.

Returns

Current bias.

6.11.3.3 debounce_period()

```
::std::chrono::microseconds gpiod::line_settings::debounce_period ( ) const  [noexcept]
```

Get debounce period.

Returns

Current debounce period.

6.11.3.4 direction()

```
line::direction gpiod::line_settings::direction ( ) const
```

Get direction.

Returns

Current direction setting.

6.11.3.5 drive()

```
line::drive gpiod::line_settings::drive ( ) const
```

Get drive setting.

Returns

Current drive.

6.11.3.6 edge_detection()

```
line::edge gpiod::line_settings::edge_detection ( ) const
```

Get edge detection.

Returns

Current edge detection setting.

6.11.3.7 event_clock()

```
line::clock gpiod::line_settings::event_clock ( ) const
```

Get the event clock used for edge event timestamps.

Returns

Current event clock type.

6.11.3.8 operator=() [1/2]

```
line_settings & gpiod::line_settings::operator= (
            const line_settings & other )
```

Copy assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.11.3.9 operator=() [2/2]

```
line_settings & gpiod::line_settings::operator= (
            line_settings && other )
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.11.3.10 output_value()

```
line::value gpiod::line_settings::output_value ( ) const
```

Get the output value.

Returns

Current output value.

6.11.3.11 reset()

```
line_settings & gpiod::line_settings::reset ( )  [noexcept]
```

Reset the line settings to default values.

Returns

Reference to self.

6.11.3.12 set_active_low()

```
line_settings & gpiod::line_settings::set_active_low (
            bool active_low )
```

Set the active-low setting.

Parameters

active_low	New active-low setting.

Returns

Reference to self.

6.11.3.13 set_bias()

```
line_settings & gpiod::line_settings::set_bias (
            line::bias bias )
```

Set bias setting.

Parameters

bias	New bias.

Returns

Reference to self.

6.11.3.14 set_debounce_period()

```
line_settings & gpiod::line_settings::set_debounce_period (
            const ::std::chrono::microseconds & period )
```

Set debounce period.

Parameters

period	New debounce period in microseconds.

Returns

Reference to self.

6.11.3.15 set_direction()

```
line_settings & gpiod::line_settings::set_direction (
            line::direction direction )
```

Set direction.

Parameters

direction	New direction.

Returns

Reference to self.

6.11.3.16 set_drive()

```
line_settings & gpiod::line_settings::set_drive (
            line::drive drive )
```

Set drive setting.

Parameters

drive	New drive.

Returns

Reference to self.

6.11.3.17 set_edge_detection()

```
line_settings & gpiod::line_settings::set_edge_detection (
            line::edge edge )
```

Set edge detection.

Parameters

edge	New edge detection setting.

Returns

Reference to self.

6.11.3.18 set_event_clock()

```
line_settings & gpiod::line_settings::set_event_clock (
            line::clock event_clock )
```

Set the event clock to use for edge event timestamps.

Parameters

event_clock	Clock to use.

Returns

Reference to self.

6.11.3.19 set_output_value()

```
line_settings & gpiod::line_settings::set_output_value (
            line::value value )
```

Set the output value.

Parameters

value	New output value.

Returns

Reference to self.

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

- gpiodcxx/line-settings.hpp

6.12 gpiod::line::offset Class Reference

Wrapper around unsigned int for representing line offsets.

`#include <line.hpp>`

Public Member Functions

- offset (unsigned int off=0)
  Constructor with implicit conversion from unsigned int.
- offset (const offset &other)=default
  Copy constructor.
- offset (offset &&other)=default
  Move constructor.
- offset & operator= (const offset &other)=default
  Assignment operator.
- offset & operator= (offset &&other) noexcept=default
  Move assignment operator.
- **operator unsigned int** () const noexcept
  Conversion operator to `unsigned int`.

Private Attributes

- unsigned int **_m_offset**

6.12.1 Detailed Description

Wrapper around unsigned int for representing line offsets.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 offset() [1/3]

```
gpiod::line::offset::offset (
            unsigned int off = 0 )  [inline]
```

Constructor with implicit conversion from unsigned int.

Parameters

off	Line offset.

6.12.2.2 offset() [2/3]

```
gpiod::line::offset::offset (
            const offset & other )  [default]
```

Copy constructor.

Parameters

other	Object to copy.

6.12.2.3 offset() [3/3]

```
gpiod::line::offset::offset (
            offset && other )  [default]
```

Move constructor.

Parameters

other	Object to move.

6.12.3 Member Function Documentation

6.12.3.1 operator=() [1/2]

```
offset & gpiod::line::offset::operator= (
            const offset & other )  [default]
```

Assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.12.3.2 operator=() [2/2]

```
offset & gpiod::line::offset::operator= (
            offset && other )  [default], [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

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

- gpiodcxx/line.hpp

6.13 gpiod::request_builder Class Reference

Intermediate object storing the configuration for a line request.

`#include <request-builder.hpp>`

Public Member Functions

- **request_builder** (const request_builder &other)=delete
- request_builder (request_builder &&other) noexcept

  Move constructor.
- request_builder & **operator=** (const request_builder &other)=delete
- request_builder & operator= (request_builder &&other) noexcept

  Move assignment operator.
- request_builder & set_request_config (request_config &req_cfg)

  Set the request config for the request.
- const request_config & get_request_config () const noexcept

  Get the current request config.
- request_builder & set_consumer (const ::std::string &consumer) noexcept

  Set consumer in the request config stored by this object.
- request_builder & set_event_buffer_size (::std::size_t event_buffer_size) noexcept

  Set the event buffer size in the request config stored by this object.
- request_builder & set_line_config (line_config &line_cfg)

  Set the line config for this request.
- const line_config & get_line_config () const noexcept

  Get the current line config.
- request_builder & add_line_settings (line::offset offset, const line_settings &settings)

  Add line settings to the line config stored by this object for a single offset.
- request_builder & add_line_settings (const line::offsets &offsets, const line_settings &settings)

  Add line settings to the line config stored by this object for a set of offsets.
- request_builder & set_output_values (const line::values &values)

  Set output values for a number of lines in the line config stored by this object.
- line_request do_request ()

  Make the line request.

Private Member Functions

- **request_builder** (chip &chip)
- friend::std::ostream & **operator<<** (::std::ostream &out, const request_builder &builder)

Private Attributes

- ::std::unique_ptr< impl > **_m_priv**
- friend **chip**

6.13.1 Detailed Description

Intermediate object storing the configuration for a line request.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 request_builder()

```
gpiod::request_builder::request_builder (
            request_builder && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to be moved.

6.13.3 Member Function Documentation

6.13.3.1 add_line_settings() [1/2]

```
request_builder & gpiod::request_builder::add_line_settings (
            const line::offsets & offsets,
            const line_settings & settings )
```

Add line settings to the line config stored by this object for a set of offsets.

Parameters

offsets	Offsets for which to add settings.
settings	Settings to add.

Returns

Reference to self.

6.13.3.2 add_line_settings() [2/2]

```
request_builder & gpiod::request_builder::add_line_settings (
            line::offset offset,
            const line_settings & settings )
```

Add line settings to the line config stored by this object for a single offset.

Parameters

offset	Offset for which to add settings.
settings	Line settings to use.

Returns

Reference to self.

6.13.3.3 do_request()

```
line_request gpiod::request_builder::do_request ( )
```

Make the line request.

Returns

New line_request object.

6.13.3.4 get_line_config()

```
const line_config & gpiod::request_builder::get_line_config ( ) const  [noexcept]
```

Get the current line config.

Returns

Const reference to the current line config stored by this object.

6.13.3.5 get_request_config()

```
const request_config & gpiod::request_builder::get_request_config ( ) const  [noexcept]
```

Get the current request config.

Returns

Const reference to the current request config stored by this object.

6.13.3.6 operator=()

```
request_builder & gpiod::request_builder::operator= (
            request_builder && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to be moved.

Returns

Reference to self.

6.13.3.7 set_consumer()

```
request_builder & gpiod::request_builder::set_consumer (
            const ::std::string & consumer )  [noexcept]
```

Set consumer in the request config stored by this object.

Parameters

consumer	New consumer string.

Returns

Reference to self.

6.13.3.8 set_event_buffer_size()

```
request_builder & gpiod::request_builder::set_event_buffer_size (
            ::std::size_t event_buffer_size )  [noexcept]
```

Set the event buffer size in the request config stored by this object.

Parameters

event_buffer_size	New event buffer size.

Returns

Reference to self.

6.13.3.9 set_line_config()

```
request_builder & gpiod::request_builder::set_line_config (
            line_config & line_cfg )
```

Set the line config for this request.

Parameters

line_cfg	Line config to use.

Returns

Reference to self.

6.13.3.10 set_output_values()

```
request_builder & gpiod::request_builder::set_output_values (
            const line::values & values )
```

Set output values for a number of lines in the line config stored by this object.

Parameters

values	Buffer containing the output values.

Returns

Reference to self.

6.13.3.11 set_request_config()

```
request_builder & gpiod::request_builder::set_request_config (
            request_config & req_cfg )
```

Set the request config for the request.

Parameters

req_cfg	Request config to use.

Returns

Reference to self.

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

- gpiodcxx/request-builder.hpp

6.14 gpiod::request_config Class Reference

Stores a set of options passed to the kernel when making a line request.

`#include <request-config.hpp>`

Public Member Functions

- **request_config** ()
  Constructor.
- **request_config** (const request_config &other)=delete
- request_config (request_config &&other) noexcept
  Move constructor.
- request_config & operator= (request_config &&other) noexcept
  Move assignment operator.
- request_config & set_consumer (const ::std::string &consumer) noexcept
  Set the consumer name.
- ::std::string consumer () const noexcept
  Get the consumer name.
- request_config & set_event_buffer_size (::std::size_t event_buffer_size) noexcept
  Set the size of the kernel event buffer.
- ::std::size_t event_buffer_size () const noexcept
  Get the edge event buffer size from this request config.

Private Member Functions

- request_config & **operator=** (const request_config &other)

Private Attributes

- ::std::shared_ptr< impl > **_m_priv**
- friend **request_builder**

6.14.1 Detailed Description

Stores a set of options passed to the kernel when making a line request.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 request_config()

```
gpiod::request_config::request_config (
            request_config && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.14.3 Member Function Documentation

6.14.3.1 consumer()

```
::std::string gpiod::request_config::consumer ( ) const  [noexcept]
```

Get the consumer name.

Returns

Currently configured consumer name. May be an empty string.

6.14.3.2 event_buffer_size()

```
::std::size_t gpiod::request_config::event_buffer_size ( ) const  [noexcept]
```

Get the edge event buffer size from this request config.

Returns

Current edge event buffer size setting.

6.14.3.3 operator=()

```
request_config & gpiod::request_config::operator= (
            request_config && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.14.3.4 set_consumer()

```
request_config & gpiod::request_config::set_consumer (
            const ::std::string & consumer )  [noexcept]
```

Set the consumer name.

Parameters

consumer	New consumer name.

Returns

Reference to self.

6.14.3.5 set_event_buffer_size()

```
request_config & gpiod::request_config::set_event_buffer_size (
            ::std::size_t event_buffer_size )  [noexcept]
```

Set the size of the kernel event buffer.

Parameters

event_buffer_size	New event buffer size.

Returns

Reference to self.

Note

The kernel may adjust the value if it's too high. If set to 0, the default value will be used.

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

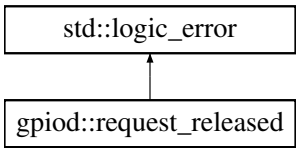
- gpiodcxx/request-config.hpp

6.15 gpiod::request_released Class Reference

Exception thrown when an already released line request is used.

`#include <exception.hpp>`

Inheritance diagram for gpiod::request_released:



Public Member Functions

- request_released (const ::std::string &what)
  Constructor.
- request_released (const request_released &other) noexcept
  Copy constructor.
- request_released (request_released &&other) noexcept
  Move constructor.
- request_released & operator= (const request_released &other) noexcept
  Assignment operator.
- request_released & operator= (request_released &&other) noexcept
  Move assignment operator.

6.15.1 Detailed Description

Exception thrown when an already released line request is used.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 request_released() [1/3]

```
gpiod::request_released::request_released (
            const ::std::string & what )  [explicit]
```

Constructor.

Parameters

what	Human readable reason for error.

6.15.2.2 request_released() [2/3]

```
gpiod::request_released::request_released (
            const request_released & other )  [noexcept]
```

Copy constructor.

Parameters

other	Object to copy from.

6.15.2.3 request_released() [3/3]

```
gpiod::request_released::request_released (
            request_released && other )  [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.15.3 Member Function Documentation

6.15.3.1 operator=() [1/2]

```
request_released & gpiod::request_released::operator= (
            const request_released & other )  [noexcept]
```

Assignment operator.

Parameters

other	Object to copy from.

Returns

Reference to self.

6.15.3.2 operator=() [2/2]

```
request_released & gpiod::request_released::operator= (
            request_released && other )  [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

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

- gpiodcxx/exception.hpp

6.16 gpiod::timestamp Class Reference

Stores the edge and info event timestamps as returned by the kernel and allows to convert them to std::chrono↩ ::time_point.

`#include <timestamp.hpp>`

Public Types

- using **time_point_monotonic** = ::std::chrono::time_point<::std::chrono::steady_clock >

  Monotonic time_point.
- using **time_point_realtime** = ::std::chrono::time_point<::std::chrono::system_clock, ::std::chrono↩ ::nanoseconds >

  Real-time time_point.

Public Member Functions

- timestamp (::std::uint64_t ns)

  Constructor with implicit conversion from `uint64_t`.
- timestamp (const timestamp &other) noexcept=default

  Copy constructor.
- timestamp (timestamp &&other) noexcept=default

  Move constructor.
- timestamp & operator= (const timestamp &other) noexcept=default

  Assignment operator.
- timestamp & operator= (timestamp &&other) noexcept=default

  Move assignment operator.
- **operator::std::uint64_t** () noexcept

  Conversion operator to `std::uint64_t`.
- ::std::uint64_t ns () const noexcept

  Get the timestamp in nanoseconds.
- time_point_monotonic to_time_point_monotonic () const

  Convert the timestamp to a monotonic time_point.
- time_point_realtime to_time_point_realtime () const

  Convert the timestamp to a real-time time_point.

Private Attributes

- ::std::uint64_t **_m_ns**

6.16.1 Detailed Description

Stores the edge and info event timestamps as returned by the kernel and allows to convert them to std::chrono↩ ::time_point.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 timestamp() [1/3]

```
gpiod::timestamp::timestamp (
            ::std::uint64_t ns )  [inline]
```

Constructor with implicit conversion from `uint64_t`.

Parameters

ns	Timestamp in nanoseconds.

6.16.2.2 timestamp() [2/3]

```
gpiod::timestamp::timestamp (
            const timestamp & other )  [default], [noexcept]
```

Copy constructor.

Parameters

other	Object to copy.

6.16.2.3 timestamp() [3/3]

```
gpiod::timestamp::timestamp (
            timestamp && other )  [default], [noexcept]
```

Move constructor.

Parameters

other	Object to move.

6.16.3 Member Function Documentation

6.16.3.1 ns()

```
::std::uint64_t gpiod::timestamp::ns ( ) const  [inline], [noexcept]
```

Get the timestamp in nanoseconds.

Returns

Timestamp in nanoseconds.

6.16.3.2 operator=() [1/2]

```
timestamp & gpiod::timestamp::operator= (
            const timestamp & other )  [default], [noexcept]
```

Assignment operator.

Parameters

other	Object to copy.

Returns

Reference to self.

6.16.3.3 operator=() [2/2]

```
timestamp & gpiod::timestamp::operator= (
            timestamp && other )  [default], [noexcept]
```

Move assignment operator.

Parameters

other	Object to move.

Returns

Reference to self.

6.16.3.4 to_time_point_monotonic()

```
time_point_monotonic gpiod::timestamp::to_time_point_monotonic ( ) const  [inline]
```

Convert the timestamp to a monotonic time_point.

Returns

time_point associated with the steady clock.

6.16.3.5 to_time_point_realtime()

```
time_point_realtime gpiod::timestamp::to_time_point_realtime ( ) const  [inline]
```

Convert the timestamp to a real-time time_point.

Returns

time_point associated with the system clock.

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

- gpiodcxx/timestamp.hpp

Chapter 7

File Documentation

7.1 gpiod.hpp File Reference

```
#include "gpiodcxx/chip.hpp"
#include "gpiodcxx/chip-info.hpp"
#include "gpiodcxx/edge-event.hpp"
#include "gpiodcxx/edge-event-buffer.hpp"
#include "gpiodcxx/exception.hpp"
#include "gpiodcxx/info-event.hpp"
#include "gpiodcxx/line.hpp"
#include "gpiodcxx/line-config.hpp"
#include "gpiodcxx/line-info.hpp"
#include "gpiodcxx/line-request.hpp"
#include "gpiodcxx/line-settings.hpp"
#include "gpiodcxx/request-builder.hpp"
#include "gpiodcxx/request-config.hpp"
```

7.2 gpiod.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_GPIOD_CXX_HPP__
9 #define __LIBGPIOD_GPIOD_CXX_HPP__
10
15 /*
16 * We don't make this symbol private because it needs to be accessible by
17 * the declarations in exception.hpp in order to expose the symbols of classes
18 * inheriting from standard exceptions.
19 */
20 #define GPIOD_CXX_API __attribute__((visibility("default")))
21
26 #define __LIBGPIOD_GPIOD_CXX_INSIDE__
27 #include "gpiodcxx/chip.hpp"
28 #include "gpiodcxx/chip-info.hpp"
29 #include "gpiodcxx/edge-event.hpp"
30 #include "gpiodcxx/edge-event-buffer.hpp"
31 #include "gpiodcxx/exception.hpp"
32 #include "gpiodcxx/info-event.hpp"
33 #include "gpiodcxx/line.hpp"
34 #include "gpiodcxx/line-config.hpp"
35 #include "gpiodcxx/line-info.hpp"
36 #include "gpiodcxx/line-request.hpp"
37 #include "gpiodcxx/line-settings.hpp"
38 #include "gpiodcxx/request-builder.hpp"
39 #include "gpiodcxx/request-config.hpp"
40 #undef __LIBGPIOD_GPIOD_CXX_INSIDE__
41
42 #endif /* __LIBGPIOD_GPIOD_CXX_HPP__ */
```

7.3 gpiodcxx/chip-info.hpp File Reference

```
#include <memory>
#include <ostream>
```

Classes

- class gpiod::chip_info

  Represents an immutable snapshot of GPIO chip information.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const chip_info &chip)

  Stream insertion operator for GPIO chip objects.

7.3.1 Function Documentation

7.3.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const chip_info & chip )
```

Stream insertion operator for GPIO chip objects.

Parameters

out	Output stream to write to.
chip	GPIO chip to insert into the output stream.

Returns

Reference to out.

7.4 chip-info.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_CHIP_INFO_HPP__
9 #define __LIBGPIOD_CXX_CHIP_INFO_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
```

```
13 #endif
14
15 #include <memory>
16 #include <ostream>
17
18 namespace gpiod {
19
20 class chip;
21
25 class chip_info final
26 {
27 public:
28
33     chip_info(const chip_info& other);
34
39     chip_info(chip_info&& other) noexcept;
40
41     ~chip_info();
42
48     chip_info& operator=(const chip_info& other);
49
55     chip_info& operator=(chip_info&& other) noexcept;
56
61     ::std::string name() const noexcept;
62
67     ::std::string label() const noexcept;
68
73     ::std::size_t num_lines() const noexcept;
74
75 private:
76
77     chip_info();
78
79     struct impl;
80
81     ::std::shared_ptr<impl> _m_priv;
82
83     friend chip;
84 };
85
92 ::std::ostream& operator<<(::std::ostream& out, const chip_info& chip);
93
94 } /* namespace gpiod */
95
96 #endif /* __LIBGPIOD_CXX_CHIP_INFO_HPP__ */
```

7.5 gpiodcxx/chip.hpp File Reference

```
#include <chrono>
#include <cstddef>
#include <iostream>
#include <filesystem>
#include <memory>
#include "line.hpp"
```

Classes

- class gpiod::chip

  Represents a GPIO chip.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const chip &chip)

  Stream insertion operator for GPIO chip objects.

7.5.1 Function Documentation

7.5.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const chip & chip )
```

Stream insertion operator for GPIO chip objects.

Parameters

out	Output stream to write to.
chip	GPIO chip to insert into the output stream.

Returns

Reference to out.

7.6 chip.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_CHIP_HPP__
9 #define __LIBGPIOD_CXX_CHIP_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <chrono>
16 #include <cstddef>
17 #include <iostream>
18 #include <filesystem>
19 #include <memory>
20
21 #include "line.hpp"
22
23 namespace gpiod {
24
25 class chip_info;
26 class info_event;
27 class line_config;
28 class line_info;
29 class line_request;
30 class request_builder;
31 class request_config;
32
36 class chip final
37 {
38 public:
39
45     explicit chip(const ::std::filesystem::path& path);
46
51     chip(chip&& other) noexcept;
52
53     ~chip();
54
55     chip& operator=(const chip& other) = delete;
56
62     chip& operator=(chip&& other) noexcept;
```

```
63
71     explicit operator bool() const noexcept;
72
78     void close();
79
84     ::std::filesystem::path path() const;
85
90     chip_info get_info() const;
91
98     line_info get_line_info(line::offset offset) const;
99
106     line_info watch_line_info(line::offset offset) const;
107
112     void unwatch_line_info(line::offset offset) const;
113
118     int fd() const;
119
130     bool wait_info_event(const ::std::chrono::nanoseconds& timeout) const;
131
136     info_event read_info_event() const;
137
144     int get_line_offset_from_name(const ::std::string& name) const;
145
150     request_builder prepare_request();
151
152 private:
153
154     struct impl;
155
156     ::std::shared_ptr<impl> _m_priv;
157
158     chip(const chip& other);
159
160     friend request_builder;
161 };
162
169 ::std::ostream& operator<<(::std::ostream& out, const chip& chip);
170
171 } /* namespace gpiod */
172
173 #endif /* __LIBGPIOD_CXX_CHIP_HPP__ */
```

7.7 gpiodcxx/edge-event-buffer.hpp File Reference

```
#include <cstddef>
#include <iostream>
#include <memory>
#include <vector>
```

Classes

- class gpiod::edge_event_buffer

  Object into which edge events are read for better performance.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const edge_event_buffer &buf)

  Stream insertion operator for GPIO edge event buffer objects.

7.7.1 Function Documentation

7.7.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const edge_event_buffer & buf )
```

Stream insertion operator for GPIO edge event buffer objects.

Parameters

out	Output stream to write to.
buf	GPIO edge event buffer object to insert into the output stream.

Returns

Reference to out.

7.8 edge-event-buffer.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_EDGE_EVENT_BUFFER_HPP__
9 #define __LIBGPIOD_CXX_EDGE_EVENT_BUFFER_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <cstddef>
16 #include <iostream>
17 #include <memory>
18 #include <vector>
19
20 namespace gpiod {
21
22 class edge_event;
23 class line_request;
24
32 class edge_event_buffer final
33 {
34 public:
35
40     using const_iterator = ::std::vector<edge_event>::const_iterator;
41
47     explicit edge_event_buffer(::std::size_t capacity = 64);
48
49     edge_event_buffer(const edge_event_buffer& other) = delete;
50
55     edge_event_buffer(edge_event_buffer&& other) noexcept;
56
57     ~edge_event_buffer();
58
59     edge_event_buffer& operator=(const edge_event_buffer& other) = delete;
60
66     edge_event_buffer& operator=(edge_event_buffer&& other) noexcept;
67
73     const edge_event& get_event(unsigned int index) const;
74
79     ::std::size_t num_events() const;
80
85     ::std::size_t capacity() const noexcept;
86
92     const_iterator begin() const noexcept;
93
99     const_iterator end() const noexcept;
100
101 private:
102
```

```
103     struct impl;
104
105     ::std::unique_ptr<impl> _m_priv;
106
107     friend line_request;
108 };
109
116 ::std::ostream& operator«(::std::ostream& out, const edge_event_buffer& buf);
117
118 } /* namespace gpiod */
119
120 #endif /* __LIBGPIOD_CXX_EDGE_EVENT_BUFFER_HPP__ */
```

7.9 gpiodcxx/edge-event.hpp File Reference

```
#include <cstdint>
#include <iostream>
#include <memory>
#include "timestamp.hpp"
```

Classes

- class gpiod::edge_event

  Immutable object containing data about a single edge event.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const edge_event &event)

  Stream insertion operator for edge events.

7.9.1 Function Documentation

7.9.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const edge_event & event )
```

Stream insertion operator for edge events.

Parameters

out	Output stream to write to.
event	Edge event to insert into the output stream.

Returns

Reference to out.

7.10 edge-event.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_EDGE_EVENT_HPP__
9 #define __LIBGPIOD_CXX_EDGE_EVENT_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <cstdint>
16 #include <iostream>
17 #include <memory>
18
19 #include "timestamp.hpp"
20
21 namespace gpiod {
22
23 class edge_event_buffer;
24
28 class edge_event final
29 {
30 public:
31
35     enum class event_type
36     {
37         RISING_EDGE = 1,
39         FALLING_EDGE,
41     };
42
47     edge_event(const edge_event& other);
48
53     edge_event(edge_event&& other) noexcept;
54
55     ~edge_event();
56
62     edge_event& operator=(const edge_event& other);
63
69     edge_event& operator=(edge_event&& other) noexcept;
70
75     event_type type() const;
76
82     timestamp timestamp_ns() const noexcept;
83
89     line::offset line_offset() const noexcept;
90
96     unsigned long global_seqno() const noexcept;
97
103     unsigned long line_seqno() const noexcept;
104
105 private:
106
107     edge_event();
108
109     struct impl;
110     struct impl_managed;
111     struct impl_external;
112
113     ::std::shared_ptr<impl> _m_priv;
114
115     friend edge_event_buffer;
116 };
117
124 ::std::ostream& operator<<(::std::ostream& out, const edge_event& event);
125
126 } /* namespace gpiod */
127
128 #endif /* __LIBGPIOD_CXX_EDGE_EVENT_HPP__ */
```

7.11 gpiodcxx/exception.hpp File Reference

```
#include <stdexcept>
#include <string>
```

Classes

- class gpiod::chip_closed

  Exception thrown when an already closed chip is used.
- class gpiod::request_released

  Exception thrown when an already released line request is used.
- class gpiod::bad_mapping

  Exception thrown when the core C library returns an invalid value for any of the line_info properties.

7.12 exception.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_EXCEPTION_HPP__
9 #define __LIBGPIOD_CXX_EXCEPTION_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <stdexcept>
16 #include <string>
17
18 namespace gpiod {
19
23 class GPIOD_CXX_API chip_closed final :  public ::std::logic_error
24 {
25 public:
26
31     explicit chip_closed(const ::std::string& what);
32
37     chip_closed(const chip_closed& other) noexcept;
38
43     chip_closed(chip_closed&& other) noexcept;
44
50     chip_closed& operator=(const chip_closed& other) noexcept;
51
57     chip_closed& operator=(chip_closed&& other) noexcept;
58
59     ~chip_closed();
60 };
61
65 class GPIOD_CXX_API request_released final :  public ::std::logic_error
66 {
67 public:
68
73     explicit request_released(const ::std::string& what);
74
79     request_released(const request_released& other) noexcept;
80
85     request_released(request_released&& other) noexcept;
86
92     request_released& operator=(const request_released& other) noexcept;
93
99     request_released& operator=(request_released&& other) noexcept;
100
101     ~request_released();
102 };
103
108 class GPIOD_CXX_API bad_mapping final :  public ::std::runtime_error
109 {
110 public:
```

```
111
116     explicit bad_mapping(const ::std::string& what);
117
122     bad_mapping(const bad_mapping& other) noexcept;
123
128     bad_mapping(bad_mapping&& other) noexcept;
129
135     bad_mapping& operator=(const bad_mapping& other) noexcept;
136
142     bad_mapping& operator=(bad_mapping&& other) noexcept;
143
144     ~bad_mapping();
145 };
146
147 } /* namespace gpiod */
148
149 #endif /* __LIBGPIOD_CXX_EXCEPTION_HPP__ */
```

7.13 info-event.hpp

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_INFO_EVENT_HPP__
9 #define __LIBGPIOD_CXX_INFO_EVENT_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <cstdint>
16 #include <iostream>
17 #include <memory>
18
19 #include "timestamp.hpp"
20
21 namespace gpiod {
22
23 class chip;
24 class line_info;
25
29 class info_event final
30 {
31 public:
32
36     enum class event_type
37     {
38         LINE_REQUESTED = 1,
40         LINE_RELEASED,
42         LINE_CONFIG_CHANGED,
44     };
45
50     info_event(const info_event& other);
51
56     info_event(info_event&& other) noexcept;
57
58     ~info_event();
59
65     info_event& operator=(const info_event& other);
66
72     info_event& operator=(info_event&& other) noexcept;
73
78     event_type type() const;
79
84     ::std::uint64_t timestamp_ns() const noexcept;
85
91     const line_info& get_line_info() const noexcept;
92
93 private:
94
95     info_event();
96
97     struct impl;
98
99     ::std::shared_ptr<impl> _m_priv;
100
101     friend chip;
102 };
103
110 ::std::ostream& operator<<(::std::ostream& out, const info_event& event);
111
112 } /* namespace gpiod */
113
114 #endif /* __LIBGPIOD_CXX_INFO_EVENT_HPP__ */
```

7.14 gpiodcxx/line-config.hpp File Reference

```
#include <map>
#include <memory>
```

Classes

- class gpiod::line_config

  Contains a set of line config options used in line requests and reconfiguration.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const line_config &config)

  Stream insertion operator for GPIO line config objects.

7.14.1 Function Documentation

7.14.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const line_config & config )
```

Stream insertion operator for GPIO line config objects.

Parameters

out	Output stream to write to.
config	Line config object to insert into the output stream.

Returns

Reference to out.

7.15 line-config.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_LINE_CONFIG_HPP__
9 #define __LIBGPIOD_CXX_LINE_CONFIG_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
```

```
13 #endif
14
15 #include <map>
16 #include <memory>
17
18 namespace gpiod {
19
20 class chip;
21 class line_request;
22 class line_settings;
23
28 class line_config final
29 {
30 public:
31
32
33     line_config();
34
35     line_config(const line_config& other) = delete;
36
41     line_config(line_config&& other) noexcept;
42
43     ~line_config();
44
50     line_config& operator=(line_config&& other) noexcept;
51
56     line_config& reset() noexcept;
57
64     line_config& add_line_settings(line::offset offset, const line_settings& settings);
65
72     line_config& add_line_settings(const line::offsets& offsets, const line_settings& settings);
73
79     line_config& set_output_values(const line::values& values);
80
87     ::std::map<line::offset, line_settings> get_line_settings() const;
88
89 private:
90
91     struct impl;
92
93     ::std::shared_ptr<impl> _m_priv;
94
95     line_config& operator=(const line_config& other);
96
97     friend line_request;
98     friend request_builder;
99 };
100
107 ::std::ostream& operator<<(::std::ostream& out, const line_config& config);
108
109 } /* namespace gpiod */
110
111 #endif /* __LIBGPIOD_CXX_LINE_CONFIG_HPP__ */
```

7.16 gpiodcxx/line-info.hpp File Reference

```
#include <chrono>
#include <iostream>
#include <memory>
#include <string>
```

Classes

- class gpiod::line_info

  Contains an immutable snapshot of the line's state at the time when the object of this class was instantiated.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const line_info &info)

  Stream insertion operator for GPIO line info objects.

7.16.1 Function Documentation

7.16.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const line_info & info )
```

Stream insertion operator for GPIO line info objects.

Parameters

out	Output stream to write to.
info	GPIO line info object to insert into the output stream.

Returns

Reference to out.

7.17 line-info.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_LINE_INFO_HPP__
9 #define __LIBGPIOD_CXX_LINE_INFO_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <chrono>
16 #include <iostream>
17 #include <memory>
18 #include <string>
19
20 namespace gpiod {
21
22 class chip;
23 class info_event;
24
29 class line_info final
30 {
31 public:
32
37     line_info(const line_info& other) noexcept;
38
43     line_info(line_info&& other) noexcept;
44
45     ~line_info();
46
52     line_info& operator=(const line_info& other) noexcept;
53
59     line_info& operator=(line_info&& other) noexcept;
60
65     line::offset offset() const noexcept;
66
72     ::std::string name() const noexcept;
73
82     bool used() const noexcept;
83
90     ::std::string consumer() const noexcept;
```

```
91
96      line::direction direction() const;
97
102      line::edge edge_detection() const;
103
109      line::bias bias() const;
110
116      line::drive drive() const;
117
122      bool active_low() const noexcept;
123
129      bool debounced() const noexcept;
130
136      ::std::chrono::microseconds debounce_period() const noexcept;
137
143      line::clock event_clock() const;
144
145 private:
146
147      line_info();
148
149      struct impl;
150
151      ::std::shared_ptr<impl> _m_priv;
152
153      friend chip;
154      friend info_event;
155 };
156
163 ::std::ostream& operator<<(::std::ostream& out, const line_info& info);
164
165 } /* namespace gpiod */
166
167 #endif /* __LIBGPIOD_CXX_LINE_INFO_HPP__ */
```

7.18 gpiodcxx/line-request.hpp File Reference

```
#include <chrono>
#include <cstddef>
#include <iostream>
#include <memory>
#include "misc.hpp"
```

Classes

- class gpiod::line_request

  Stores the context of a set of requested GPIO lines.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const line_request &request)

  Stream insertion operator for line requests.

7.18.1 Function Documentation

7.18.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const line_request & request )
```

Stream insertion operator for line requests.

Parameters

out	Output stream to write to.
request	Line request object to insert into the output stream.

Returns

Reference to out.

7.19 line-request.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_LINE_REQUEST_HPP__
9 #define __LIBGPIOD_CXX_LINE_REQUEST_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <chrono>
16 #include <cstddef>
17 #include <iostream>
18 #include <memory>
19
20 #include "misc.hpp"
21
22 namespace gpiod {
23
24 class chip;
25 class edge_event;
26 class edge_event_buffer;
27 class line_config;
28
32 class line_request final
33 {
34 public:
35
36     line_request(const line_request& other) = delete;
37
42     line_request(line_request&& other) noexcept;
43
44     ~line_request();
45
46     line_request& operator=(const line_request& other) = delete;
47
53     line_request& operator=(line_request&& other) noexcept;
54
63     explicit operator bool() const noexcept;
64
71     void release();
72
77     ::std::string chip_name() const;
78
83     ::std::size_t num_lines() const;
84
89     line::offsets offsets() const;
90
96     line::value get_value(line::offset offset);
97
104     line::values get_values(const line::offsets& offsets);
105
110     line::values get_values();
111
121     void get_values(const line::offsets& offsets, line::values& values);
122
129     void get_values(line::values& values);
130
137     line_request& set_value(line::offset offset, line::value value);
138
144     line_request& set_values(const line::value_mappings& values);
145
153     line_request& set_values(const line::offsets& offsets, const line::values& values);
154
```

```
161     line_request& set_values(const line::values& values);
162
168     line_request& reconfigure_lines(const line_config& config);
169
175     int fd() const;
176
187     bool wait_edge_events(const ::std::chrono::nanoseconds& timeout) const;
188
195     ::std::size_t read_edge_events(edge_event_buffer& buffer);
196
204     ::std::size_t read_edge_events(edge_event_buffer& buffer, ::std::size_t max_events);
205
206 private:
207
208     line_request();
209
210     struct impl;
211
212     ::std::unique_ptr<impl> _m_priv;
213
214     friend request_builder;
215 };
216
223 ::std::ostream& operator<<(::std::ostream& out, const line_request& request);
224
225 } /* namespace gpiod */
226
227 #endif /* __LIBGPIOD_CXX_LINE_REQUEST_HPP__ */
```

7.20 line-settings.hpp

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_LINE_SETTINGS_HPP__
9 #define __LIBGPIOD_CXX_LINE_SETTINGS_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <chrono>
16 #include <memory>
17
18 #include "line.hpp"
19
20 namespace gpiod {
21
22 class line_config;
23
27 class line_settings final
28 {
29 public:
30
34     line_settings();
35
40     line_settings(const line_settings& other);
41
46     line_settings(line_settings&& other) noexcept;
47
48     ~line_settings();
49
55     line_settings& operator=(const line_settings& other);
56
62     line_settings& operator=(line_settings&& other);
63
68     line_settings& reset() noexcept;
69
75     line_settings& set_direction(line::direction direction);
76
81     line::direction direction() const;
82
88     line_settings& set_edge_detection(line::edge edge);
89
94     line::edge edge_detection() const;
95
101     line_settings& set_bias(line::bias bias);
102
107     line::bias bias() const;
108
114     line_settings& set_drive(line::drive drive);
115
120     line::drive drive() const;
```

```
121
127     line_settings& set_active_low(bool active_low);
128
133     bool active_low() const noexcept;
134
140     line_settings& set_debounce_period(const ::std::chrono::microseconds& period);
141
146     ::std::chrono::microseconds debounce_period() const noexcept;
147
153     line_settings& set_event_clock(line::clock event_clock);
154
159     line::clock event_clock() const;
160
166     line_settings& set_output_value(line::value value);
167
172     line::value output_value() const;
173
174 private:
175
176     struct impl;
177
178     ::std::unique_ptr<impl> _m_priv;
179
180     friend line_config;
181 };
182
189 ::std::ostream& operator<<(::std::ostream& out, const line_settings& settings);
190
191 } /* namespace gpiod */
192
193 #endif /* __LIBGPIOD_CXX_LINE_SETTINGS_HPP__ */
```

7.21 gpiodcxx/line.hpp File Reference

```
#include <ostream>
#include <utility>
#include <vector>
```

Classes

- class gpiod::line::offset

  Wrapper around unsigned int for representing line offsets.

Namespaces

- namespace gpiod::line

  Namespace containing various type definitions for GPIO lines.

Typedefs

- using **gpiod::line::offsets** = ::std::vector< offset >

  Vector of line offsets.
- using **gpiod::line::values** = ::std::vector< value >

  Vector of line values.
- using **gpiod::line::value_mapping** = ::std::pair< offset, value >

  Represents a mapping of a line offset to line logical state.
- using **gpiod::line::value_mappings** = ::std::vector< value_mapping >

  Vector of offset->value mappings. Each mapping is defined as a pair of an unsigned and signed integers.

Enumerations

- enum class gpiod::line::value { gpiod::line::INACTIVE = 0 , gpiod::line::ACTIVE = 1 }

  Logical line states.
- enum class gpiod::line::direction { gpiod::line::AS_IS = 1 , gpiod::line::INPUT , gpiod::line::OUTPUT }

  Direction settings.
- enum class gpiod::line::edge { gpiod::line::NONE = 1 , gpiod::line::RISING , gpiod::line::FALLING , gpiod::line::BOTH }

  Edge detection settings.
- enum class gpiod::line::bias { gpiod::line::AS_IS = 1 , gpiod::line::UNKNOWN , gpiod::line::DISABLED , gpiod::line::PULL_UP , gpiod::line::PULL_DOWN }

  Internal bias settings.
- enum class gpiod::line::drive { gpiod::line::PUSH_PULL = 1 , gpiod::line::OPEN_DRAIN , gpiod::line::OPEN_SOURCE }

  Drive settings.
- enum class gpiod::line::clock { gpiod::line::MONOTONIC = 1 , gpiod::line::REALTIME , **HTE** }

  Event clock settings.

Functions

- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, value val)

  Stream insertion operator for logical line values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, direction dir)

  Stream insertion operator for direction values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, edge edge)

  Stream insertion operator for edge detection values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, bias bias)

  Stream insertion operator for bias values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, drive drive)

  Stream insertion operator for drive values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, clock clock)

  Stream insertion operator for event clock values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, const values &vals)

  Stream insertion operator for the list of output values.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, const offsets &offs)

  Stream insertion operator for the list of line offsets.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, const value_mapping &mapping)

  Stream insertion operator for the offset-to-value mapping.
- ::std::ostream & gpiod::line::operator<< (::std::ostream &out, const value_mappings &mappings)

  Stream insertion operator for the list of offset-to-value mappings.

7.22 line.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_LINE_HPP__
9 #define __LIBGPIOD_CXX_LINE_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <ostream>
16 #include <utility>
17 #include <vector>
18
19 namespace gpiod {
20
24 namespace line {
25
29 class offset
30 {
31 public:
36     offset(unsigned int off = 0) :  _m_offset(off) {   }
37
42     offset(const offset& other) = default;
43
48     offset(offset&& other) = default;
49
50     ~offset() = default;
51
57     offset& operator=(const offset& other) = default;
58
64     offset& operator=(offset&& other) noexcept = default;
65
69     operator unsigned int() const noexcept
70     {
71         return this->_m_offset;
72     }
73
74 private:
75     unsigned int _m_offset;
76 };
77
81 enum class value
82 {
83     INACTIVE = 0,
85     ACTIVE = 1,
87 };
88
92 enum class direction
93 {
94     AS_IS = 1,
96     INPUT,
98     OUTPUT,
100 };
101
105 enum class edge
106 {
107     NONE = 1,
109     RISING,
111     FALLING,
113     BOTH,
115 };
116
120 enum class bias
121 {
122     AS_IS = 1,
124     UNKNOWN,
126     DISABLED,
128     PULL_UP,
130     PULL_DOWN,
132 };
133
137 enum class drive
138 {
139     PUSH_PULL = 1,
141     OPEN_DRAIN,
143     OPEN_SOURCE,
145 };
146
150 enum class clock
151 {
152     MONOTONIC = 1,
```

```
154     REALTIME,
156     HTE,
157     /*« Line uses the hardware timestamp engine for event timestamps.  */
158 };
159
163 using offsets = ::std::vector<offset>;
164
168 using values = ::std::vector<value>;
169
173 using value_mapping = ::std::pair<offset, value>;
174
179 using value_mappings = ::std::vector<value_mapping>;
180
187 ::std::ostream& operator«(::std::ostream& out, value val);
188
195 ::std::ostream& operator«(::std::ostream& out, direction dir);
196
203 ::std::ostream& operator«(::std::ostream& out, edge edge);
204
211 ::std::ostream& operator«(::std::ostream& out, bias bias);
212
219 ::std::ostream& operator«(::std::ostream& out, drive drive);
220
227 ::std::ostream& operator«(::std::ostream& out, clock clock);
228
235 ::std::ostream& operator«(::std::ostream& out, const values& vals);
236
243 ::std::ostream& operator«(::std::ostream& out, const offsets& offs);
244
252 ::std::ostream& operator«(::std::ostream& out, const value_mapping& mapping);
253
261 ::std::ostream& operator«(::std::ostream& out, const value_mappings& mappings);
262
263 } /* namespace line */
264
265 } /* namespace gpiod */
266
267 #endif /* __LIBGPIOD_CXX_LINE_HPP__ */
```

7.23 gpiodcxx/misc.hpp File Reference

```
#include <string>
```

Functions

- bool gpiod::is_gpiochip_device (const ::std::filesystem::path &path)

  Check if the file pointed to by path is a GPIO chip character device.
- const ::std::string & gpiod::api_version ()

  Get the human readable version string for libgpiod API.

7.23.1 Function Documentation

7.23.1.1 api_version()

```
const ::std::string & gpiod::api_version ( )
```

Get the human readable version string for libgpiod API.

Returns

String containing the library version.

7.23.1.2 is_gpiochip_device()

```
bool gpiod::is_gpiochip_device (
            const ::std::filesystem::path & path )
```

Check if the file pointed to by path is a GPIO chip character device.

Parameters

path	Path to check.

Returns

True if the file exists and is a GPIO chip character device or a symbolic link to it.

7.24 misc.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_MISC_HPP__
9 #define __LIBGPIOD_CXX_MISC_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <string>
16
17 namespace gpiod {
18
25 bool is_gpiochip_device(const ::std::filesystem::path& path);
26
31 const ::std::string& api_version();
32
33 } /* namespace gpiod */
34
35 #endif /* __LIBGPIOD_CXX_MISC_HPP__ */
```

7.25 gpiodcxx/request-builder.hpp File Reference

```
#include <memory>
#include <ostream>
```

Classes

- class gpiod::request_builder

  Intermediate object storing the configuration for a line request.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const request_builder &builder)

  Stream insertion operator for GPIO request builder objects.

7.25.1 Function Documentation

7.25.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const request_builder & builder )
```

Stream insertion operator for GPIO request builder objects.

Parameters

out	Output stream to write to.
builder	Request builder object to insert into the output stream.

Returns

Reference to out.

7.26 request-builder.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_REQUEST_BUILDER_HPP__
9 #define __LIBGPIOD_CXX_REQUEST_BUILDER_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <memory>
16 #include <ostream>
17
18 namespace gpiod {
19
20 class chip;
21 class line_config;
22 class line_request;
23 class request_config;
24
28 class request_builder final
29 {
30 public:
31
32     request_builder(const request_builder& other) = delete;
33
38     request_builder(request_builder&& other) noexcept;
39
40     ~request_builder();
41
42     request_builder& operator=(const request_builder& other) = delete;
43
49     request_builder& operator=(request_builder&& other) noexcept;
50
56     request_builder& set_request_config(request_config& req_cfg);
57
63     const request_config& get_request_config() const noexcept;
64
70     request_builder& set_consumer(const ::std::string& consumer) noexcept;
71
78     request_builder& set_event_buffer_size(::std::size_t event_buffer_size) noexcept;
```

```
79
85      request_builder& set_line_config(line_config &line_cfg);
86
92      const line_config& get_line_config() const noexcept;
93
101      request_builder& add_line_settings(line::offset offset, const line_settings& settings);
102
110      request_builder& add_line_settings(const line::offsets& offsets, const line_settings& settings);
111
118      request_builder& set_output_values(const line::values& values);
119
124      line_request do_request();
125
126 private:
127
128      struct impl;
129
130      request_builder(chip& chip);
131
132      ::std::unique_ptr<impl> _m_priv;
133
134      friend chip;
135      friend ::std::ostream& operator<<(::std::ostream& out, const request_builder& builder);
136 };
137
144 ::std::ostream& operator<<(::std::ostream& out, const request_builder& builder);
145
146 } /* namespace gpiod */
147
148 #endif /* __LIBGPIOD_CXX_REQUEST_BUILDER_HPP__ */
```

7.27 gpiodcxx/request-config.hpp File Reference

```
#include <cstddef>
#include <iostream>
#include <memory>
#include <string>
#include "line.hpp"
```

Classes

- class gpiod::request_config

  Stores a set of options passed to the kernel when making a line request.

Functions

- ::std::ostream & gpiod::operator<< (::std::ostream &out, const request_config &config)

  Stream insertion operator for request_config objects.

7.27.1 Function Documentation

7.27.1.1 operator<<()

```
::std::ostream & gpiod::operator<< (
            ::std::ostream & out,
            const request_config & config )
```

Stream insertion operator for request_config objects.

Parameters

out	Output stream to write to.
config	request_config to insert into the output stream.

Returns

Reference to out.

7.28 request-config.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2021-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_REQUEST_CONFIG_HPP__
9 #define __LIBGPIOD_CXX_REQUEST_CONFIG_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <cstddef>
16 #include <iostream>
17 #include <memory>
18 #include <string>
19
20 #include "line.hpp"
21
22 namespace gpiod {
23
24 class chip;
25
30 class request_config final
31 {
32 public:
33
37     request_config();
38
39     request_config(const request_config& other) = delete;
40
45     request_config(request_config&& other) noexcept;
46
47     ~request_config();
48
54     request_config& operator=(request_config&& other) noexcept;
55
61     request_config& set_consumer(const ::std::string& consumer) noexcept;
62
67     ::std::string consumer() const noexcept;
68
76     request_config& set_event_buffer_size(::std::size_t event_buffer_size) noexcept;
77
82     ::std::size_t event_buffer_size() const noexcept;
83
84 private:
85
86     struct impl;
87
88     ::std::shared_ptr<impl> _m_priv;
89
90     request_config& operator=(const request_config& other);
91
92     friend request_builder;
93 };
94
101 ::std::ostream& operator<<(::std::ostream& out, const request_config& config);
102
103 } /* namespace gpiod */
104
105 #endif /* __LIBGPIOD_CXX_REQUEST_CONFIG_HPP__ */
```

7.29 gpiodcxx/timestamp.hpp File Reference

```
#include <chrono>
#include <cstdint>
```

Classes

- class gpiod::timestamp

  Stores the edge and info event timestamps as returned by the kernel and allows to convert them to std::chrono↩ ::time_point.

7.30 timestamp.hpp

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_CXX_TIMESTAMP_HPP__
9 #define __LIBGPIOD_CXX_TIMESTAMP_HPP__
10
11 #if !defined(__LIBGPIOD_GPIOD_CXX_INSIDE__)
12 #error "Only gpiod.hpp can be included directly."
13 #endif
14
15 #include <chrono>
16 #include <cstdint>
17
18 namespace gpiod {
19
24 class timestamp final
25 {
26 public:
27
31     using time_point_monotonic = ::std::chrono::time_point<::std::chrono::steady_clock>;
32
36     using time_point_realtime = ::std::chrono::time_point<::std::chrono::system_clock,
37                                 ::std::chrono::nanoseconds>;
38
43     timestamp(::std::uint64_t ns) :  _m_ns(ns) { }
44
49     timestamp(const timestamp& other) noexcept = default;
50
55     timestamp(timestamp&& other) noexcept = default;
56
62     timestamp& operator=(const timestamp& other) noexcept = default;
63
69     timestamp& operator=(timestamp&& other) noexcept = default;
70
71     ~timestamp() = default;
72
76     operator ::std::uint64_t() noexcept
77     {
78         return this->ns();
79     }
80
85     ::std::uint64_t ns() const noexcept
86     {
87         return this->_m_ns;
88     }
89
94     time_point_monotonic to_time_point_monotonic()const
95 {
96         return time_point_monotonic(::std::chrono::nanoseconds(this->ns()));
97     }
98
103     time_point_realtime to_time_point_realtime()const
104 {
105          return time_point_realtime(::std::chrono::nanoseconds(this->ns()));
106     }
107
108 private:
109     ::std::uint64_t _m_ns;
```

```
110 };
111
112 } /* namespace gpiod */
113
114 #endif /* __LIBGPIOD_CXX_TIMESTAMP_HPP__ */
```

Index