libgpiod

Generated by Doxygen 1.9.4

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Module Documentation

4.1 Chip info

Functions

- void gpiod_chip_info_free (struct gpiod_chip_info *info)

  Free a chip info object and release all associated resources.
- const char * gpiod_chip_info_get_name (struct gpiod_chip_info *info)

  Get the name of the chip as represented in the kernel.
- const char * gpiod_chip_info_get_label (struct gpiod_chip_info *info)

  Get the label of the chip as represented in the kernel.
- size_t gpiod_chip_info_get_num_lines (struct gpiod_chip_info *info)

  Get the number of lines exposed by the chip.

4.1.1 Detailed Description

Functions for retrieving kernel information about chips.

Line info object contains an immutable snapshot of a chip's status.

The chip info contains all the publicly available information about a chip.

Some accessor methods return pointers. Those pointers refer to internal fields. The lifetimes of those fields are tied to the lifetime of the containing chip info object. Such pointers remain valid until gpiod_chip_info_free is called on the containing chip info object. They must not be freed by the caller.

4.1.2 Function Documentation

4.1.2.1 gpiod_chip_info_free()

```
void gpiod_chip_info_free (
            struct gpiod_chip_info * info )
```

```
#include <gpiod.h>
```

Free a chip info object and release all associated resources.

Parameters

info	GPIO chip info object to free.

4.1.2.2 gpiod_chip_info_get_label()

```
const char * gpiod_chip_info_get_label (
            struct gpiod_chip_info * info )
```

```
#include <gpiod.h>
```

Get the label of the chip as represented in the kernel.

Parameters

info	GPIO chip info object.

Returns

Valid pointer to a human-readable string containing the chip label. The string lifetime is tied to the chip info object so the pointer must not be freed by the caller.

4.1.2.3 gpiod_chip_info_get_name()

```
const char * gpiod_chip_info_get_name (
            struct gpiod_chip_info * info )
```

```
#include <gpiod.h>
```

Get the name of the chip as represented in the kernel.

Parameters

info	GPIO chip info object.

Returns

Valid pointer to a human-readable string containing the chip name. The string lifetime is tied to the chip info object so the pointer must not be freed by the caller.

4.1.2.4 gpiod_chip_info_get_num_lines()

```
size_t gpiod_chip_info_get_num_lines (
            struct gpiod_chip_info * info )
```

```
#include <gpiod.h>
```

Get the number of lines exposed by the chip.

Parameters

info	GPIO chip info object.

Returns

Number of GPIO lines.

4.2 GPIO chips

Functions

- struct gpiod_chip * gpiod_chip_open (const char *path)
  Open a chip by path.
- void gpiod_chip_close (struct gpiod_chip *chip)
  Close the chip and release all associated resources.
- struct gpiod_chip_info * gpiod_chip_get_info (struct gpiod_chip *chip)
  Get information about the chip.
- const char * gpiod_chip_get_path (struct gpiod_chip *chip)
  Get the path used to open the chip.
- struct gpiod_line_info * gpiod_chip_get_line_info (struct gpiod_chip *chip, unsigned int offset)
  Get a snapshot of information about a line.
- struct gpiod_line_info * gpiod_chip_watch_line_info (struct gpiod_chip *chip, unsigned int offset)
  Get a snapshot of the status of a line and start watching it for future changes.
- int gpiod_chip_unwatch_line_info (struct gpiod_chip *chip, unsigned int offset)
  Stop watching a line for status changes.
- int gpiod_chip_get_fd (struct gpiod_chip *chip)
  Get the file descriptor associated with the chip.
- int gpiod_chip_wait_info_event (struct gpiod_chip *chip, int64_t timeout_ns)
  Wait for line status change events on any of the watched lines on the chip.
- struct gpiod_info_event * gpiod_chip_read_info_event (struct gpiod_chip *chip)
  Read a single line status change event from the chip.
- int gpiod_chip_get_line_offset_from_name (struct gpiod_chip *chip, const char *name)
  Map a line's name to its offset within the chip.
- struct gpiod_line_request * gpiod_chip_request_lines (struct gpiod_chip *chip, struct gpiod_request_config *req_cfg, struct gpiod_line_config *line_cfg)
  Request a set of lines for exclusive usage.

4.2.1 Detailed Description

Functions and data structures for GPIO chip operations.

A GPIO chip object is associated with an open file descriptor to the GPIO character device. It exposes basic information about the chip and allows callers to retrieve information about each line, watch lines for state changes and make line requests.

4.2.2 Function Documentation

4.2.2.1 gpiod_chip_close()

```
void gpiod_chip_close (
            struct gpiod_chip * chip )
```

```
#include <gpiod.h>
```

Close the chip and release all associated resources.

Parameters

chip	Chip to close.

4.2.2.2 gpiod_chip_get_fd()

```
int gpiod_chip_get_fd (
            struct gpiod_chip * chip )
```

```
#include <gpiod.h>
```

Get the file descriptor associated with the chip.

Parameters

chip	GPIO chip object.

Returns

File descriptor number for the chip.

This function never fails. The returned file descriptor must not be closed by the caller. Call gpiod_chip_close to close the file descriptor by closing the chip owning it.

4.2.2.3 gpiod_chip_get_info()

```
struct gpiod_chip_info * gpiod_chip_get_info (
            struct gpiod_chip * chip )
```

```
#include <gpiod.h>
```

Get information about the chip.

Parameters

chip	GPIO chip object.

Returns

New GPIO chip info object or NULL if an error occurred. The returned object must be freed by the caller using gpiod_chip_info_free.

4.2.2.4 gpiod_chip_get_line_info()

```
struct gpiod_line_info * gpiod_chip_get_line_info (
            struct gpiod_chip * chip,
            unsigned int offset )
```

```
#include <gpiod.h>
```

Get a snapshot of information about a line.

Parameters

chip	GPIO chip object.
offset	The offset of the GPIO line.

Returns

New GPIO line info object or NULL if an error occurred. The returned object must be freed by the caller using gpiod_line_info_free.

4.2.2.5 gpiod_chip_get_line_offset_from_name()

```
int gpiod_chip_get_line_offset_from_name (
            struct gpiod_chip * chip,
            const char * name )
```

```
#include <gpiod.h>
```

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

Parameters

chip	GPIO chip object.
name	Name of the GPIO line to map.

Returns

Offset of the line within the chip or -1 on error.

Note

If a line with given name is not exposed by the chip, the function sets errno to ENOENT.

4.2.2.6 gpiod_chip_get_path()

```
const char * gpiod_chip_get_path (
            struct gpiod_chip * chip )
```

```
#include <gpiod.h>
```

Get the path used to open the chip.

Parameters

chip	GPIO chip object.

Returns

Path to the file passed as argument to gpiod_chip_open. The returned pointer is valid for the lifetime of the chip object and must not be freed by the caller.

4.2.2.7 gpiod_chip_open()

```
struct gpiod_chip * gpiod_chip_open (
            const char * path )
```

```
#include <gpiod.h>
```

Open a chip by path.

Parameters

path	Path to the gpiochip device file.

Returns

GPIO chip object or NULL if an error occurred. The returned object must be closed by the caller using gpiod_chip_close.

4.2.2.8 gpiod_chip_read_info_event()

```
struct gpiod_info_event * gpiod_chip_read_info_event (
            struct gpiod_chip * chip )
```

```
#include <gpiod.h>
```

Read a single line status change event from the chip.

Parameters

chip	GPIO chip object.

Returns

Newly read watch event object or NULL on error. The event must be freed by the caller using gpiod_info_event_free.

Note

If no events are pending, this function will block.

4.2.2.9 gpiod_chip_request_lines()

```
struct gpiod_line_request * gpiod_chip_request_lines (
            struct gpiod_chip * chip,
            struct gpiod_request_config * req_cfg,
            struct gpiod_line_config * line_cfg )
```

```
#include <gpiod.h>
```

Request a set of lines for exclusive usage.

Parameters

chip	GPIO chip object.
req_cfg	Request config object. Can be NULL for default settings.
line_cfg	Line config object.

Returns

New line request object or NULL if an error occurred. The request must be released by the caller using gpiod_line_request_release.

4.2.2.10 gpiod_chip_unwatch_line_info()

```
int gpiod_chip_unwatch_line_info (
            struct gpiod_chip * chip,
            unsigned int offset )
```

```
#include <gpiod.h>
```

Stop watching a line for status changes.

Parameters

chip	GPIO chip object.
offset	The offset of the line to stop watching.

Returns

0 on success, -1 on failure.

4.2.2.11 gpiod_chip_wait_info_event()

```
int gpiod_chip_wait_info_event (
            struct gpiod_chip * chip,
            int64_t timeout_ns )
```

```
#include <gpiod.h>
```

Wait for line status change events on any of the watched lines on the chip.

Parameters

chip	GPIO chip object.
timeout_ns	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

0 if wait timed out, -1 if an error occurred, 1 if an event is pending.

4.2.2.12 gpiod_chip_watch_line_info()

```
struct gpiod_line_info * gpiod_chip_watch_line_info (
            struct gpiod_chip * chip,
            unsigned int offset )
```

```
#include <gpiod.h>
```

Get a snapshot of the status of a line and start watching it for future changes.

Parameters

chip	GPIO chip object.
offset	The offset of the GPIO line.

Returns

New GPIO line info object or NULL if an error occurred. The returned object must be freed by the caller using gpiod_line_info_free.

Note

Line status does not include the line value. To monitor the line value the line must be requested as an input with edge detection set.

4.3 Line configuration objects

Functions

- struct gpiod_line_config * gpiod_line_config_new (void)

  Create a new line config object.
- void gpiod_line_config_free (struct gpiod_line_config *config)

  Free the line config object and release all associated resources.
- void gpiod_line_config_reset (struct gpiod_line_config *config)

  Reset the line config object.
- int gpiod_line_config_add_line_settings (struct gpiod_line_config *config, const unsigned int *offsets, size_t num_offsets, struct gpiod_line_settings *settings)

  Add line settings for a set of offsets.
- struct gpiod_line_settings * gpiod_line_config_get_line_settings (struct gpiod_line_config *config, unsigned int offset)

  Get line settings for offset.
- int gpiod_line_config_set_output_values (struct gpiod_line_config *config, const enum gpiod_line_value *values, size_t num_values)

  Set output values for a number of lines.
- size_t gpiod_line_config_get_num_configured_offsets (struct gpiod_line_config *config)

  Get the number of configured line offsets.
- size_t gpiod_line_config_get_configured_offsets (struct gpiod_line_config *config, unsigned int *offsets, size_t max_offsets)

  Get configured offsets.

4.3.1 Detailed Description

Functions for manipulating line configuration objects.

The line-config object contains the configuration for lines that can be used in two cases:

- when making a line request
- when reconfiguring a set of already requested lines.

A new line-config object is empty. Using it in a request will lead to an error. In order to a line-config to become useful, it needs to be assigned at least one offset-to-settings mapping by calling gpiod_line_config_add_line_settings.

When calling gpiod_chip_request_lines, the library will request all offsets that were assigned settings in the order that they were assigned. If any of the offsets was duplicated, the last one will take precedence.

4.3.2 Function Documentation

4.3.2.1 gpiod_line_config_add_line_settings()

```
int gpiod_line_config_add_line_settings (
            struct gpiod_line_config * config,
            const unsigned int * offsets,
            size_t num_offsets,
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Add line settings for a set of offsets.

Parameters

config	Line config object.
offsets	Array of offsets for which to apply the settings.
num_offsets	Number of offsets stored in the offsets array.
settings	Line settings to apply.

Returns

0 on success, -1 on failure.

4.3.2.2 gpiod_line_config_free()

```
void gpiod_line_config_free (
            struct gpiod_line_config * config )
```

```
#include <gpiod.h>
```

Free the line config object and release all associated resources.

Parameters

config	Line config object to free.

4.3.2.3 gpiod_line_config_get_configured_offsets()

```
size_t gpiod_line_config_get_configured_offsets (
            struct gpiod_line_config * config,
            unsigned int * offsets,
            size_t max_offsets )
```

```
#include <gpiod.h>
```

Get configured offsets.

Parameters

config	Line config object.
offsets	Array to store offsets.
max_offsets	Number of offsets that can be stored in the offsets array.

Returns

Number of offsets stored in the offsets array.

If max_offsets is lower than the number of lines actually requested (this value can be retrieved using gpiod_line_config_get_num_configured_offsets), then only up to max_lines offsets will be stored in offsets.

4.3.2.4 gpiod_line_config_get_line_settings()

```
struct gpiod_line_settings * gpiod_line_config_get_line_settings (
            struct gpiod_line_config * config,
            unsigned int offset )
```

```
#include <gpiod.h>
```

Get line settings for offset.

Parameters

config	Line config object.
offset	Offset for which to get line settings.

Returns

New line settings object (must be freed by the caller) or NULL on error.

4.3.2.5 gpiod_line_config_get_num_configured_offsets()

```
size_t gpiod_line_config_get_num_configured_offsets (
            struct gpiod_line_config * config )
```

```
#include <gpiod.h>
```

Get the number of configured line offsets.

Parameters

config	Line config object.

Returns

Number of offsets for which line settings have been added.

4.3.2.6 gpiod_line_config_new()

```
struct gpiod_line_config * gpiod_line_config_new (
            void  )
```

```
#include <gpiod.h>
```

Create a new line config object.

Returns

New line config object or NULL on error. The returned object must be freed by the caller using gpiod_line_config_free.

4.3.2.7 gpiod_line_config_reset()

```
void gpiod_line_config_reset (
            struct gpiod_line_config * config )
```

```
#include <gpiod.h>
```

Reset the line config object.

Parameters

config	Line config object to free.

Resets the entire configuration stored in the object. This is useful if the user wants to reuse the object without reallocating it.

4.3.2.8 gpiod_line_config_set_output_values()

```
int gpiod_line_config_set_output_values (
            struct gpiod_line_config * config,
            const enum gpiod_line_value * values,
            size_t num_values )
```

```
#include <gpiod.h>
```

Set output values for a number of lines.

Parameters

config	Line config object.
values	Buffer containing the output values.
num_values	Number of values in the buffer.

Returns

0 on success, -1 on error.

This is a helper that allows users to set multiple (potentially different) output values at once while using the same line settings object. Instead of modifying the output value in the settings object and calling gpiod_line_config_add_line_settings multiple times, we can specify the settings, add them for a set of offsets and then call this function to set the output values.

Values set by this function override whatever values were specified in the regular line settings.

Each value must be associated with the line identified by the corresponding entry in the offset array filled by gpiod_line_request_get_requested_offsets.

4.4 Line definitions

Enumerations

- enum gpiod_line_value { GPIOD_LINE_VALUE_ERROR = -1 , GPIOD_LINE_VALUE_INACTIVE = 0 , GPIOD_LINE_VALUE_ACTIVE = 1 }

  Logical line state.
- enum gpiod_line_direction { GPIOD_LINE_DIRECTION_AS_IS = 1 , GPIOD_LINE_DIRECTION_INPUT , GPIOD_LINE_DIRECTION_OUTPUT }

  Direction settings.

- enum gpiod_line_edge { GPIOD_LINE_EDGE_NONE = 1 , GPIOD_LINE_EDGE_RISING , GPIOD_LINE_EDGE_FALLING , GPIOD_LINE_EDGE_BOTH }

  Edge detection settings.
- enum gpiod_line_bias { GPIOD_LINE_BIAS_AS_IS = 1 , GPIOD_LINE_BIAS_UNKNOWN , GPIOD_LINE_BIAS_DISABLED , GPIOD_LINE_BIAS_PULL_UP , GPIOD_LINE_BIAS_PULL_DOWN }

  Internal bias settings.
- enum gpiod_line_drive { GPIOD_LINE_DRIVE_PUSH_PULL = 1 , GPIOD_LINE_DRIVE_OPEN_DRAIN , GPIOD_LINE_DRIVE_OPEN_SOURCE }

  Drive settings.
- enum gpiod_line_clock { GPIOD_LINE_CLOCK_MONOTONIC = 1 , GPIOD_LINE_CLOCK_REALTIME , GPIOD_LINE_CLOCK_HTE }

  Clock settings.

4.4.1 Detailed Description

These defines are used across the API.

4.4.2 Enumeration Type Documentation

4.4.2.1 gpiod_line_bias

`enum gpiod_line_bias`

`#include <gpiod.h>`

Internal bias settings.

Enumerator

GPIOD_LINE_BIAS_AS_IS	Don't change the bias setting when applying line config.
GPIOD_LINE_BIAS_UNKNOWN	The internal bias state is unknown.
GPIOD_LINE_BIAS_DISABLED	The internal bias is disabled.
GPIOD_LINE_BIAS_PULL_UP	The internal pull-up bias is enabled.
GPIOD_LINE_BIAS_PULL_DOWN	The internal pull-down bias is enabled.

4.4.2.2 gpiod_line_clock

`enum gpiod_line_clock`

`#include <gpiod.h>`

Clock settings.

Enumerator

GPIOD_LINE_CLOCK_MONOTONIC	Line uses the monotonic clock for edge event timestamps.
GPIOD_LINE_CLOCK_REALTIME	Line uses the realtime clock for edge event timestamps.
GPIOD_LINE_CLOCK_HTE	Line uses the hardware timestamp engine for event timestamps.

4.4.2.3 gpiod_line_direction

```
enum gpiod_line_direction
```

```
#include <gpiod.h>
```

Direction settings.

Enumerator

GPIOD_LINE_DIRECTION_AS_IS	Request the line(s), but don't change direction.
GPIOD_LINE_DIRECTION_INPUT	Direction is input - for reading the value of an externally driven GPIO line.
GPIOD_LINE_DIRECTION_OUTPUT	Direction is output - for driving the GPIO line.

4.4.2.4 gpiod_line_drive

```
enum gpiod_line_drive
```

```
#include <gpiod.h>
```

Drive settings.

Enumerator

GPIOD_LINE_DRIVE_PUSH_PULL	Drive setting is push-pull.
GPIOD_LINE_DRIVE_OPEN_DRAIN	Line output is open-drain.
GPIOD_LINE_DRIVE_OPEN_SOURCE	Line output is open-source.

4.4.2.5 gpiod_line_edge

```
enum gpiod_line_edge
```

```
#include <gpiod.h>
```

Edge detection settings.

Enumerator

GPIOD_LINE_EDGE_NONE	Line edge detection is disabled.
GPIOD_LINE_EDGE_RISING	Line detects rising edge events.
GPIOD_LINE_EDGE_FALLING	Line detects falling edge events.
GPIOD_LINE_EDGE_BOTH	Line detects both rising and falling edge events.

4.4.2.6 gpiod_line_value

```
enum gpiod_line_value
```

```
#include <gpiod.h>
```

Logical line state.

Enumerator

GPIOD_LINE_VALUE_ERROR	Returned to indicate an error when reading the value.
GPIOD_LINE_VALUE_INACTIVE	Line is logically inactive.
GPIOD_LINE_VALUE_ACTIVE	Line is logically active.

4.5 Line edge events handling

Enumerations

- enum gpiod_edge_event_type { GPIOD_EDGE_EVENT_RISING_EDGE = 1 , GPIOD_EDGE_EVENT_FALLING_EDGE }

  Event types.

Functions

- void gpiod_edge_event_free (struct gpiod_edge_event *event)

  Free the edge event object.
- struct gpiod_edge_event * gpiod_edge_event_copy (struct gpiod_edge_event *event)

  Copy the edge event object.
- enum gpiod_edge_event_type gpiod_edge_event_get_event_type (struct gpiod_edge_event *event)

  Get the event type.
- uint64_t gpiod_edge_event_get_timestamp_ns (struct gpiod_edge_event *event)

  Get the timestamp of the event.
- unsigned int gpiod_edge_event_get_line_offset (struct gpiod_edge_event *event)

  Get the offset of the line which triggered the event.
- unsigned long gpiod_edge_event_get_global_seqno (struct gpiod_edge_event *event)

  Get the global sequence number of the event.
- unsigned long gpiod_edge_event_get_line_seqno (struct gpiod_edge_event *event)

Get the event sequence number specific to the line.

- struct gpiod_edge_event_buffer * gpiod_edge_event_buffer_new (size_t capacity)

  Create a new edge event buffer.
- size_t gpiod_edge_event_buffer_get_capacity (struct gpiod_edge_event_buffer *buffer)

  Get the capacity (the max number of events that can be stored) of the event buffer.
- void gpiod_edge_event_buffer_free (struct gpiod_edge_event_buffer *buffer)

  Free the edge event buffer and release all associated resources.
- struct gpiod_edge_event * gpiod_edge_event_buffer_get_event (struct gpiod_edge_event_buffer *buffer, unsigned long index)

  Get an event stored in the buffer.
- size_t gpiod_edge_event_buffer_get_num_events (struct gpiod_edge_event_buffer *buffer)

  Get the number of events a buffer has stored.

4.5.1 Detailed Description

Functions and data types for handling edge events.

An edge event object contains information about a single line edge event. It contains the event type, timestamp and the offset of the line on which the event occurred as well as two sequence numbers (global for all lines in the associated request and local for this line only).

Edge events are stored into an edge-event buffer object to improve performance and to limit the number of memory allocations when a large number of events are being read.

4.5.2 Enumeration Type Documentation

4.5.2.1 gpiod_edge_event_type

```
enum gpiod_edge_event_type
```

```
#include <gpiod.h>
```

Event types.

Enumerator

GPIOD_EDGE_EVENT_RISING_EDGE	Rising edge event.
GPIOD_EDGE_EVENT_FALLING_EDGE	Falling edge event.

4.5.3 Function Documentation

4.5.3.1 gpiod_edge_event_buffer_free()

```
void gpiod_edge_event_buffer_free (
            struct gpiod_edge_event_buffer * buffer )
```

```
#include <gpiod.h>
```

Free the edge event buffer and release all associated resources.

Parameters

buffer	Edge event buffer to free.

4.5.3.2 gpiod_edge_event_buffer_get_capacity()

```
size_t gpiod_edge_event_buffer_get_capacity (
            struct gpiod_edge_event_buffer * buffer )
```

```
#include <gpiod.h>
```

Get the capacity (the max number of events that can be stored) of the event buffer.

Parameters

buffer	Edge event buffer.

Returns

The capacity of the buffer.

4.5.3.3 gpiod_edge_event_buffer_get_event()

```
struct gpiod_edge_event * gpiod_edge_event_buffer_get_event (
            struct gpiod_edge_event_buffer * buffer,
            unsigned long index )
```

```
#include <gpiod.h>
```

Get an event stored in the buffer.

Parameters

buffer	Edge event buffer.
index	Index of the event in the buffer.

Returns

Pointer to an event stored in the buffer. The lifetime of the event is tied to the buffer object. Users must not free the event returned by this function.

Warning

Thread-safety: Since events are tied to the buffer instance, different threads may not operate on the buffer and any associated events at the same time. Events can be copied using gpiod_edge_event_copy in order to create a standalone objects - which each may safely be used from a different thread concurrently.

4.5.3.4 gpiod_edge_event_buffer_get_num_events()

```
size_t gpiod_edge_event_buffer_get_num_events (
            struct gpiod_edge_event_buffer * buffer )
```

```
#include <gpiod.h>
```

Get the number of events a buffer has stored.

Parameters

buffer	Edge event buffer.

Returns

Number of events stored in the buffer.

4.5.3.5 gpiod_edge_event_buffer_new()

```
struct gpiod_edge_event_buffer * gpiod_edge_event_buffer_new (
            size_t capacity )
```

```
#include <gpiod.h>
```

Create a new edge event buffer.

Parameters

capacity	Number of events the buffer can store (min = 1, max = 1024).

Returns

New edge event buffer or NULL on error.

Note

If capacity equals 0, it will be set to a default value of 64. If capacity is larger than 1024, it will be limited to 1024.

The user space buffer is independent of the kernel buffer (gpiod_request_config_set_event_buffer_size). As the user space buffer is filled from the kernel buffer, there is no benefit making the user space buffer larger than the kernel buffer. The default kernel buffer size for each request is (16 * num_lines).

4.5.3.6 gpiod_edge_event_copy()

```
struct gpiod_edge_event * gpiod_edge_event_copy (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Copy the edge event object.

Parameters

event	Edge event to copy.

Returns

Copy of the edge event or NULL on error. The returned object must be freed by the caller using gpiod_edge_event_free.

4.5.3.7 gpiod_edge_event_free()

```
void gpiod_edge_event_free (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Free the edge event object.

Parameters

event	Edge event object to free.

4.5.3.8 gpiod_edge_event_get_event_type()

```
enum gpiod_edge_event_type gpiod_edge_event_get_event_type (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Get the event type.

Parameters

event	GPIO edge event.

Returns

The event type (GPIOD_EDGE_EVENT_RISING_EDGE or GPIOD_EDGE_EVENT_FALLING_EDGE).

4.5.3.9 gpiod_edge_event_get_global_seqno()

```
unsigned long gpiod_edge_event_get_global_seqno (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Get the global sequence number of the event.

Parameters

event	GPIO edge event.

Returns

Sequence number of the event in the series of events for all lines in the associated line request.

4.5.3.10 gpiod_edge_event_get_line_offset()

```
unsigned int gpiod_edge_event_get_line_offset (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Get the offset of the line which triggered the event.

Parameters

event	GPIO edge event.

Returns

Line offset.

4.5.3.11 gpiod_edge_event_get_line_seqno()

```
unsigned long gpiod_edge_event_get_line_seqno (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Get the event sequence number specific to the line.

Parameters

event	GPIO edge event.

Returns

Sequence number of the event in the series of events only for this line within the lifetime of the associated line request.

4.5.3.12 gpiod_edge_event_get_timestamp_ns()

```
uint64_t gpiod_edge_event_get_timestamp_ns (
            struct gpiod_edge_event * event )
```

```
#include <gpiod.h>
```

Get the timestamp of the event.

Parameters

event	GPIO edge event.

Returns

Timestamp in nanoseconds.

Note

The source clock for the timestamp depends on the event_clock setting for the line.

4.6 Line info

Functions

- void gpiod_line_info_free (struct gpiod_line_info *info)

  Free a line info object and release all associated resources.
- struct gpiod_line_info * gpiod_line_info_copy (struct gpiod_line_info *info)

  Copy a line info object.
- unsigned int gpiod_line_info_get_offset (struct gpiod_line_info *info)

  Get the offset of the line.
- const char * gpiod_line_info_get_name (struct gpiod_line_info *info)

  Get the name of the line.
- bool gpiod_line_info_is_used (struct gpiod_line_info *info)

  Check if the line is in use.
- const char * gpiod_line_info_get_consumer (struct gpiod_line_info *info)

  Get the name of the consumer of the line.
- enum gpiod_line_direction gpiod_line_info_get_direction (struct gpiod_line_info *info)

  Get the direction setting of the line.
- enum gpiod_line_edge gpiod_line_info_get_edge_detection (struct gpiod_line_info *info)

  Get the edge detection setting of the line.
- enum gpiod_line_bias gpiod_line_info_get_bias (struct gpiod_line_info *info)

  Get the bias setting of the line.
- enum gpiod_line_drive gpiod_line_info_get_drive (struct gpiod_line_info *info)

  Get the drive setting of the line.
- bool gpiod_line_info_is_active_low (struct gpiod_line_info *info)

  Check if the logical value of the line is inverted compared to the physical.
- bool gpiod_line_info_is_debounced (struct gpiod_line_info *info)

  Check if the line is debounced (either by hardware or by the kernel software debouncer).
- unsigned long gpiod_line_info_get_debounce_period_us (struct gpiod_line_info *info)

  Get the debounce period of the line, in microseconds.
- enum gpiod_line_clock gpiod_line_info_get_event_clock (struct gpiod_line_info *info)

  Get the event clock setting used for edge event timestamps for the line.

4.6.1 Detailed Description

Functions for retrieving kernel information about both requested and free lines.

Line info object contains an immutable snapshot of a line's status.

The line info contains all the publicly available information about a line, which does not include the line value. The line must be requested to access the line value.

Some accessor methods return pointers. Those pointers refer to internal fields. The lifetimes of those fields are tied to the lifetime of the containing line info object. Such pointers remain valid until gpiod_line_info_free is called on the containing line info object. They must not be freed by the caller.

4.6.2 Function Documentation

4.6.2.1 gpiod_line_info_copy()

```
struct gpiod_line_info * gpiod_line_info_copy (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Copy a line info object.

Parameters

info	Line info to copy.

Returns

Copy of the line info or NULL on error. The returned object must be freed by the caller using :gpiod_line_↩ info_free.

4.6.2.2 gpiod_line_info_free()

```
void gpiod_line_info_free (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Free a line info object and release all associated resources.

Parameters

info	GPIO line info object to free.

4.6.2.3 gpiod_line_info_get_bias()

```
enum gpiod_line_bias gpiod_line_info_get_bias (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the bias setting of the line.

Parameters

info	GPIO line object.

Returns

Returns GPIOD_LINE_BIAS_PULL_UP, GPIOD_LINE_BIAS_PULL_DOWN, GPIOD_LINE_BIAS_DISABLED or GPIOD_LINE_BIAS_UNKNOWN.

4.6.2.4 gpiod_line_info_get_consumer()

```
const char * gpiod_line_info_get_consumer (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the name of the consumer of the line.

Parameters

info	GPIO line info object.

Returns

Name of the GPIO consumer as it is represented in the kernel. This function returns a valid pointer to a null-terminated string or NULL if the consumer name is not set. The string lifetime is tied to the line info object so the pointer must not be freed.

4.6.2.5 gpiod_line_info_get_debounce_period_us()

```
unsigned long gpiod_line_info_get_debounce_period_us (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the debounce period of the line, in microseconds.

Parameters

info	GPIO line info object.

Returns

Debounce period in microseconds. 0 if the line is not debounced.

4.6.2.6 gpiod_line_info_get_direction()

```
enum gpiod_line_direction gpiod_line_info_get_direction (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the direction setting of the line.

Parameters

info	GPIO line info object.

Returns

Returns GPIOD_LINE_DIRECTION_INPUT or GPIOD_LINE_DIRECTION_OUTPUT.

4.6.2.7 gpiod_line_info_get_drive()

```
enum gpiod_line_drive gpiod_line_info_get_drive (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the drive setting of the line.

Parameters

info	GPIO line info object.

Returns

Returns GPIOD_LINE_DRIVE_PUSH_PULL, GPIOD_LINE_DRIVE_OPEN_DRAIN or GPIOD_LINE_DRIVE_OPEN_SOURCE

4.6.2.8 gpiod_line_info_get_edge_detection()

```
enum gpiod_line_edge gpiod_line_info_get_edge_detection (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the edge detection setting of the line.

Parameters

info	GPIO line info object.

Returns

Returns GPIOD_LINE_EDGE_NONE, GPIOD_LINE_EDGE_RISING, GPIOD_LINE_EDGE_FALLING or GPIOD_LINE_EDGE_BOTH.

4.6.2.9 gpiod_line_info_get_event_clock()

```
enum gpiod_line_clock gpiod_line_info_get_event_clock (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the event clock setting used for edge event timestamps for the line.

Parameters

info	GPIO line info object.

Returns

Returns GPIOD_LINE_CLOCK_MONOTONIC, GPIOD_LINE_CLOCK_HTE or GPIOD_LINE_CLOCK_REALTIME.

4.6.2.10 gpiod_line_info_get_name()

```
const char * gpiod_line_info_get_name (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the name of the line.

Parameters

info	GPIO line info object.

Returns

Name of the GPIO line as it is represented in the kernel. This function returns a valid pointer to a null-terminated string or NULL if the line is unnamed. The string lifetime is tied to the line info object so the pointer must not be freed.

4.6.2.11 gpiod_line_info_get_offset()

```
unsigned int gpiod_line_info_get_offset (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Get the offset of the line.

Parameters

info	GPIO line info object.

Returns

Offset of the line within the parent chip.

The offset uniquely identifies the line on the chip. The combination of the chip and offset uniquely identifies the line within the system.

4.6.2.12 gpiod_line_info_is_active_low()

```
bool gpiod_line_info_is_active_low (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Check if the logical value of the line is inverted compared to the physical.

Parameters

info	GPIO line object.

Returns

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

4.6.2.13 gpiod_line_info_is_debounced()

```
bool gpiod_line_info_is_debounced (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

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

Parameters

info	GPIO line info object.

Returns

True if the line is debounced, false otherwise.

4.6.2.14 gpiod_line_info_is_used()

```
bool gpiod_line_info_is_used (
            struct gpiod_line_info * info )
```

```
#include <gpiod.h>
```

Check if the line is in use.

Parameters

info	GPIO line object.

Returns

True if the line is in use, false otherwise.

The exact reason a line is busy cannot be determined from user space. It may have been requested by another process or hogged by the kernel. It only matters that the line is used and can't be requested until released by the existing consumer.

4.7 Line request operations

Functions

- void gpiod_line_request_release (struct gpiod_line_request *request)

  Release the requested lines and free all associated resources.
- const char * gpiod_line_request_get_chip_name (struct gpiod_line_request *request)

  Get the name of the chip this request was made on.
- size_t gpiod_line_request_get_num_requested_lines (struct gpiod_line_request *request)

  Get the number of lines in the request.
- size_t gpiod_line_request_get_requested_offsets (struct gpiod_line_request *request, unsigned int *offsets, size_t max_offsets)

  Get the offsets of the lines in the request.
- enum gpiod_line_value gpiod_line_request_get_value (struct gpiod_line_request *request, unsigned int offset)

  Get the value of a single requested line.
- int gpiod_line_request_get_values_subset (struct gpiod_line_request *request, size_t num_values, const unsigned int *offsets, enum gpiod_line_value *values)

  Get the values of a subset of requested lines.
- int gpiod_line_request_get_values (struct gpiod_line_request *request, enum gpiod_line_value *values)

  Get the values of all requested lines.

- int gpiod_line_request_set_value (struct gpiod_line_request *request, unsigned int offset, enum gpiod_line_value value)

  Set the value of a single requested line.
- int gpiod_line_request_set_values_subset (struct gpiod_line_request *request, size_t num_values, const unsigned int *offsets, const enum gpiod_line_value *values)

  Set the values of a subset of requested lines.
- int gpiod_line_request_set_values (struct gpiod_line_request *request, const enum gpiod_line_value *values)

  Set the values of all lines associated with a request.
- int gpiod_line_request_reconfigure_lines (struct gpiod_line_request *request, struct gpiod_line_config *config)

  Update the configuration of lines associated with a line request.
- int gpiod_line_request_get_fd (struct gpiod_line_request *request)

  Get the file descriptor associated with a line request.
- int gpiod_line_request_wait_edge_events (struct gpiod_line_request *request, int64_t timeout_ns)

  Wait for edge events on any of the requested lines.
- int gpiod_line_request_read_edge_events (struct gpiod_line_request *request, struct gpiod_edge_event_buffer *buffer, size_t max_events)

  Read a number of edge events from a line request.

4.7.1 Detailed Description

Functions allowing interactions with requested lines.

4.7.2 Function Documentation

4.7.2.1 gpiod_line_request_get_chip_name()

```
const char * gpiod_line_request_get_chip_name (
            struct gpiod_line_request * request )
```

```
#include <gpiod.h>
```

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

Parameters

request	Line request object.

Returns

Name the GPIO chip device. The returned pointer is valid for the lifetime of the request object and must not be freed by the caller.

4.7.2.2 gpiod_line_request_get_fd()

```
int gpiod_line_request_get_fd (
            struct gpiod_line_request * request )
```

```
#include <gpiod.h>
```

Get the file descriptor associated with a line request.

Parameters

request	GPIO line request.

Returns

The file descriptor associated with the request. This function never fails. The returned file descriptor must not be closed by the caller. Call gpiod_line_request_release to close the file.

4.7.2.3 gpiod_line_request_get_num_requested_lines()

```
size_t gpiod_line_request_get_num_requested_lines (
            struct gpiod_line_request * request )
```

```
#include <gpiod.h>
```

Get the number of lines in the request.

Parameters

request	Line request object.

Returns

Number of requested lines.

4.7.2.4 gpiod_line_request_get_requested_offsets()

```
size_t gpiod_line_request_get_requested_offsets (
            struct gpiod_line_request * request,
            unsigned int * offsets,
            size_t max_offsets )
```

```
#include <gpiod.h>
```

Get the offsets of the lines in the request.

Parameters

request	Line request object.
offsets	Array to store offsets.
max_offsets	Number of offsets that can be stored in the offsets array.

Returns

Number of offsets stored in the offsets array.

If max_offsets is lower than the number of lines actually requested (this value can be retrieved using gpiod_line_request_get_num_requested_lines), then only up to max_lines offsets will be stored in offsets.

4.7.2.5 gpiod_line_request_get_value()

```
enum gpiod_line_value gpiod_line_request_get_value (
            struct gpiod_line_request * request,
            unsigned int offset )
```

```
#include <gpiod.h>
```

Get the value of a single requested line.

Parameters

request	Line request object.
offset	The offset of the line of which the value should be read.

Returns

Returns 1 or 0 on success and -1 on error.

4.7.2.6 gpiod_line_request_get_values()

```
int gpiod_line_request_get_values (
            struct gpiod_line_request * request,
            enum gpiod_line_value * values )
```

```
#include <gpiod.h>
```

Get the values of all requested lines.

Parameters

request	GPIO line request.
values	Array in which the values will be stored. Must be sized to hold the number of lines filled by gpiod_line_request_get_num_requested_lines. Each value is associated with the line identified by the corresponding entry in the offset array filled by gpiod_line_request_get_requested_offsets.

Returns

0 on success, -1 on failure.

4.7.2.7 gpiod_line_request_get_values_subset()

```
int gpiod_line_request_get_values_subset (
            struct gpiod_line_request * request,
            size_t num_values,
            const unsigned int * offsets,
            enum gpiod_line_value * values )
```

```
#include <gpiod.h>
```

Get the values of a subset of requested lines.

Parameters

request	GPIO line request.
num_values	Number of lines for which to read values.
offsets	Array of offsets identifying the subset of requested lines from which to read values.
values	Array in which the values will be stored. Must be sized to hold `num_values` entries. Each value is associated with the line identified by the corresponding entry in `offsets`.

Returns

0 on success, -1 on failure.

4.7.2.8 gpiod_line_request_read_edge_events()

```
int gpiod_line_request_read_edge_events (
            struct gpiod_line_request * request,
            struct gpiod_edge_event_buffer * buffer,
            size_t max_events )
```

```
#include <gpiod.h>
```

Read a number of edge events from a line request.

Parameters

request	GPIO line request.
buffer	Edge event buffer, sized to hold at least `max_events`.
max_events	Maximum number of events to read.

Returns

On success returns the number of events read from the file descriptor, on failure return -1.

Note

This function will block if no event was queued for the line request.

Any exising events in the buffer are overwritten. This is not an append operation.

4.7.2.9 gpiod_line_request_reconfigure_lines()

```
int gpiod_line_request_reconfigure_lines (
            struct gpiod_line_request * request,
            struct gpiod_line_config * config )
```

```
#include <gpiod.h>
```

Update the configuration of lines associated with a line request.

Parameters

request	GPIO line request.
config	New line config to apply.

Returns

0 on success, -1 on failure.

Note

The new line configuration completely replaces the old.

Any requested lines without overrides are configured to the requested defaults.

Any configured overrides for lines that have not been requested are silently ignored.

4.7.2.10 gpiod_line_request_release()

```
void gpiod_line_request_release (
            struct gpiod_line_request * request )
```

```
#include <gpiod.h>
```

Release the requested lines and free all associated resources.

Parameters

request	Line request object to release.

4.7.2.11 gpiod_line_request_set_value()

```
int gpiod_line_request_set_value (
            struct gpiod_line_request * request,
            unsigned int offset,
            enum gpiod_line_value value )
```

```
#include <gpiod.h>
```

Set the value of a single requested line.

Parameters

request	Line request object.
offset	The offset of the line for which the value should be set.
value	Value to set.

Returns

0 on success, -1 on failure.

4.7.2.12 gpiod_line_request_set_values()

```
int gpiod_line_request_set_values (
            struct gpiod_line_request * request,
            const enum gpiod_line_value * values )
```

```
#include <gpiod.h>
```

Set the values of all lines associated with a request.

Parameters

request	GPIO line request.
values	Array containing the values to set. Must be sized to contain the number of lines filled by gpiod_line_request_get_num_requested_lines. Each value is associated with the line identified by the corresponding entry in the offset array filled by gpiod_line_request_get_requested_offsets.

Returns

0 on success, -1 on failure.

4.7.2.13 gpiod_line_request_set_values_subset()

```
int gpiod_line_request_set_values_subset (
            struct gpiod_line_request * request,
            size_t num_values,
            const unsigned int * offsets,
            const enum gpiod_line_value * values )
```

```
#include <gpiod.h>
```

Set the values of a subset of requested lines.

Parameters

request	GPIO line request.
num_values	Number of lines for which to set values.
offsets	Array of offsets, containing the number of entries specified by `num_values`, identifying the requested lines for which to set values.
values	Array of values to set, containing the number of entries specified by `num_values`. Each value is associated with the line identified by the corresponding entry in `offsets`.

Returns

0 on success, -1 on failure.

4.7.2.14 gpiod_line_request_wait_edge_events()

```
int gpiod_line_request_wait_edge_events (
            struct gpiod_line_request * request,
            int64_t timeout_ns )
```

```
#include <gpiod.h>
```

Wait for edge events on any of the requested lines.

Parameters

request	GPIO line request.
timeout_ns	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

0 if wait timed out, -1 if an error occurred, 1 if an event is pending.

Lines must have edge detection set for edge events to be emitted. By default edge detection is disabled.

4.8 Line settings objects

Functions

- struct gpiod_line_settings * gpiod_line_settings_new (void)

  Create a new line settings object.
- void gpiod_line_settings_free (struct gpiod_line_settings *settings)

  Free the line settings object and release all associated resources.
- void gpiod_line_settings_reset (struct gpiod_line_settings *settings)

  Reset the line settings object to its default values.
- struct gpiod_line_settings * gpiod_line_settings_copy (struct gpiod_line_settings *settings)

  Copy the line settings object.
- int gpiod_line_settings_set_direction (struct gpiod_line_settings *settings, enum gpiod_line_direction direction)

  Set direction.
- enum gpiod_line_direction gpiod_line_settings_get_direction (struct gpiod_line_settings *settings)

  Get direction.
- int gpiod_line_settings_set_edge_detection (struct gpiod_line_settings *settings, enum gpiod_line_edge edge)

  Set edge detection.
- enum gpiod_line_edge gpiod_line_settings_get_edge_detection (struct gpiod_line_settings *settings)

  Get edge detection.
- int gpiod_line_settings_set_bias (struct gpiod_line_settings *settings, enum gpiod_line_bias bias)

  Set bias.
- enum gpiod_line_bias gpiod_line_settings_get_bias (struct gpiod_line_settings *settings)

  Get bias.
- int gpiod_line_settings_set_drive (struct gpiod_line_settings *settings, enum gpiod_line_drive drive)

  Set drive.
- enum gpiod_line_drive gpiod_line_settings_get_drive (struct gpiod_line_settings *settings)

  Get drive.
- void gpiod_line_settings_set_active_low (struct gpiod_line_settings *settings, bool active_low)

  Set active-low setting.
- bool gpiod_line_settings_get_active_low (struct gpiod_line_settings *settings)

  Get active-low setting.
- void gpiod_line_settings_set_debounce_period_us (struct gpiod_line_settings *settings, unsigned long period)

  Set debounce period.
- unsigned long gpiod_line_settings_get_debounce_period_us (struct gpiod_line_settings *settings)

  Get debounce period.
- int gpiod_line_settings_set_event_clock (struct gpiod_line_settings *settings, enum gpiod_line_clock event↩_clock)

  Set event clock.
- enum gpiod_line_clock gpiod_line_settings_get_event_clock (struct gpiod_line_settings *settings)

  Get event clock setting.
- int gpiod_line_settings_set_output_value (struct gpiod_line_settings *settings, enum gpiod_line_value value)

  Set the output value.
- enum gpiod_line_value gpiod_line_settings_get_output_value (struct gpiod_line_settings *settings)

  Get the output value.

4.8.1 Detailed Description

Functions for manipulating line settings objects.

Line settings object contains a set of line properties that can be used when requesting lines or reconfiguring an existing request.

Mutators in general can only fail if the new property value is invalid. The return values can be safely ignored - the object remains valid even after a mutator fails and simply uses the sane default appropriate for given property.

4.8.2 Function Documentation

4.8.2.1 gpiod_line_settings_copy()

```
struct gpiod_line_settings * gpiod_line_settings_copy (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Copy the line settings object.

Parameters

settings	Line settings object to copy.

Returns

New line settings object that must be freed using gpiod_line_settings_free or NULL on failure.

4.8.2.2 gpiod_line_settings_free()

```
void gpiod_line_settings_free (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Free the line settings object and release all associated resources.

Parameters

settings	Line settings object.

4.8.2.3 gpiod_line_settings_get_active_low()

```
bool gpiod_line_settings_get_active_low (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get active-low setting.

Parameters

settings	Line settings object.

Returns

True if active-low is enabled, false otherwise.

4.8.2.4 gpiod_line_settings_get_bias()

```
enum gpiod_line_bias gpiod_line_settings_get_bias (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get bias.

Parameters

settings	Line settings object.

Returns

Current bias setting.

4.8.2.5 gpiod_line_settings_get_debounce_period_us()

```
unsigned long gpiod_line_settings_get_debounce_period_us (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get debounce period.

Parameters

settings	Line settings object.

Returns

Current debounce period in microseconds.

4.8.2.6 gpiod_line_settings_get_direction()

```
enum gpiod_line_direction gpiod_line_settings_get_direction (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get direction.

Parameters

settings	Line settings object.

Returns

Current direction.

4.8.2.7 gpiod_line_settings_get_drive()

```
enum gpiod_line_drive gpiod_line_settings_get_drive (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get drive.

Parameters

settings	Line settings object.

Returns

Current drive setting.

4.8.2.8 gpiod_line_settings_get_edge_detection()

```
enum gpiod_line_edge gpiod_line_settings_get_edge_detection (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get edge detection.

Parameters

settings	Line settings object.

Returns

Current edge detection setting.

4.8.2.9 gpiod_line_settings_get_event_clock()

```
enum gpiod_line_clock gpiod_line_settings_get_event_clock (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get event clock setting.

Parameters

settings	Line settings object.

Returns

Current event clock setting.

4.8.2.10 gpiod_line_settings_get_output_value()

```
enum gpiod_line_value gpiod_line_settings_get_output_value (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Get the output value.

Parameters

settings	Line settings object.

Returns

Current output value.

4.8.2.11 gpiod_line_settings_new()

```
struct gpiod_line_settings * gpiod_line_settings_new (
            void  )
```

```
#include <gpiod.h>
```

Create a new line settings object.

Returns

New line settings object or NULL on error. The returned object must be freed by the caller using gpiod_line_settings_free.

4.8.2.12 gpiod_line_settings_reset()

```
void gpiod_line_settings_reset (
            struct gpiod_line_settings * settings )
```

```
#include <gpiod.h>
```

Reset the line settings object to its default values.

Parameters

settings	Line settings object.

4.8.2.13 gpiod_line_settings_set_active_low()

```
void gpiod_line_settings_set_active_low (
            struct gpiod_line_settings * settings,
            bool active_low )
```

```
#include <gpiod.h>
```

Set active-low setting.

Parameters

settings	Line settings object.
active_low	New active-low setting.

4.8.2.14 gpiod_line_settings_set_bias()

```
int gpiod_line_settings_set_bias (
            struct gpiod_line_settings * settings,
            enum gpiod_line_bias bias )
```

```
#include <gpiod.h>
```

Set bias.

Parameters

settings	Line settings object.
bias	New bias.

Returns

0 on success, -1 on failure.

4.8.2.15 gpiod_line_settings_set_debounce_period_us()

```
void gpiod_line_settings_set_debounce_period_us (
            struct gpiod_line_settings * settings,
            unsigned long period )
```

```
#include <gpiod.h>
```

Set debounce period.

Parameters

settings	Line settings object.
period	New debounce period in microseconds.

4.8.2.16 gpiod_line_settings_set_direction()

```
int gpiod_line_settings_set_direction (
            struct gpiod_line_settings * settings,
            enum gpiod_line_direction direction )
```

```
#include <gpiod.h>
```

Set direction.

Parameters

settings	Line settings object.
direction	New direction.

Returns

0 on success, -1 on error.

4.8.2.17 gpiod_line_settings_set_drive()

```
int gpiod_line_settings_set_drive (
            struct gpiod_line_settings * settings,
            enum gpiod_line_drive drive )
```

```
#include <gpiod.h>
```

Set drive.

Parameters

settings	Line settings object.
drive	New drive setting.

Returns

0 on success, -1 on failure.

4.8.2.18 gpiod_line_settings_set_edge_detection()

```
int gpiod_line_settings_set_edge_detection (
            struct gpiod_line_settings * settings,
            enum gpiod_line_edge edge )
```

```
#include <gpiod.h>
```

Set edge detection.

Parameters

settings	Line settings object.
edge	New edge detection setting.

Returns

0 on success, -1 on failure.

4.8.2.19 gpiod_line_settings_set_event_clock()

```
int gpiod_line_settings_set_event_clock (
            struct gpiod_line_settings * settings,
            enum gpiod_line_clock event_clock )
```

```
#include <gpiod.h>
```

Set event clock.

Parameters

settings	Line settings object.
event_clock	New event clock.

Returns

0 on success, -1 on failure.

4.8.2.20 gpiod_line_settings_set_output_value()

```
int gpiod_line_settings_set_output_value (
            struct gpiod_line_settings * settings,
            enum gpiod_line_value value )
```

```
#include <gpiod.h>
```

Set the output value.

Parameters

settings	Line settings object.
value	New output value.

Returns

0 on success, -1 on failure.

4.9 Line status watch events

Enumerations

- enum gpiod_info_event_type { GPIOD_INFO_EVENT_LINE_REQUESTED = 1 , GPIOD_INFO_EVENT_LINE_RELEASED , GPIOD_INFO_EVENT_LINE_CONFIG_CHANGED }

  Line status change event types.

Functions

- void gpiod_info_event_free (struct gpiod_info_event *event)

  Free the info event object and release all associated resources.
- enum gpiod_info_event_type gpiod_info_event_get_event_type (struct gpiod_info_event *event)

  Get the event type of the status change event.
- uint64_t gpiod_info_event_get_timestamp_ns (struct gpiod_info_event *event)

  Get the timestamp of the event.
- struct gpiod_line_info * gpiod_info_event_get_line_info (struct gpiod_info_event *event)

  Get the snapshot of line-info associated with the event.

4.9.1 Detailed Description

Accessors for the info event objects allowing to monitor changes in GPIO line status.

Callers are notified about changes in a line's status due to GPIO uAPI calls. Each info event contains information about the event itself (timestamp, type) as well as a snapshot of line's status in the form of a line-info object.

4.9.2 Enumeration Type Documentation

4.9.2.1 gpiod_info_event_type

```
enum gpiod_info_event_type
```

```
#include <gpiod.h>
```

Line status change event types.

Enumerator

GPIOD_INFO_EVENT_LINE_REQUESTED	Line has been requested.
GPIOD_INFO_EVENT_LINE_RELEASED	Previously requested line has been released.
GPIOD_INFO_EVENT_LINE_CONFIG_CHANGED	Line configuration has changed.

4.9.3 Function Documentation

4.9.3.1 gpiod_info_event_free()

```
void gpiod_info_event_free (
            struct gpiod_info_event * event )
```

```
#include <gpiod.h>
```

Free the info event object and release all associated resources.

Parameters

event	Info event to free.

4.9.3.2 gpiod_info_event_get_event_type()

```
enum gpiod_info_event_type gpiod_info_event_get_event_type (
            struct gpiod_info_event * event )
```

```
#include <gpiod.h>
```

Get the event type of the status change event.

Parameters

event	Line status watch event.

Returns

One of GPIOD_INFO_EVENT_LINE_REQUESTED, GPIOD_INFO_EVENT_LINE_RELEASED or GPIOD_INFO_EVENT_LINE

4.9.3.3 gpiod_info_event_get_line_info()

```
struct gpiod_line_info * gpiod_info_event_get_line_info (
            struct gpiod_info_event * event )
```

```
#include <gpiod.h>
```

Get the snapshot of line-info associated with the event.

Parameters

event	Line info event object.

Returns

Returns a pointer to the line-info object associated with the event. The object lifetime is tied to the event object, so the pointer must be not be freed by the caller.

Warning

Thread-safety: Since the line-info object is tied to the event, different threads may not operate on the event and line-info at the same time. The line-info can be copied using gpiod_line_info_copy in order to create a standalone object - which then may safely be used from a different thread concurrently.

4.9.3.4 gpiod_info_event_get_timestamp_ns()

```
uint64_t gpiod_info_event_get_timestamp_ns (
            struct gpiod_info_event * event )
```

```
#include <gpiod.h>
```

Get the timestamp of the event.

Parameters

event	Line status watch event.

Returns

Timestamp in nanoseconds, read from the monotonic clock.

4.10 Request configuration objects

Functions

- struct gpiod_request_config * gpiod_request_config_new (void)
  Create a new request config object.
- void gpiod_request_config_free (struct gpiod_request_config *config)
  Free the request config object and release all associated resources.
- void gpiod_request_config_set_consumer (struct gpiod_request_config *config, const char *consumer)
  Set the consumer name for the request.
- const char * gpiod_request_config_get_consumer (struct gpiod_request_config *config)
  Get the consumer name configured in the request config.
- void gpiod_request_config_set_event_buffer_size (struct gpiod_request_config *config, size_t event_buffer↩_size)
  Set the size of the kernel event buffer for the request.
- size_t gpiod_request_config_get_event_buffer_size (struct gpiod_request_config *config)
  Get the edge event buffer size for the request config.

4.10.1 Detailed Description

Functions for manipulating request configuration objects.

Request config objects are used to pass a set of options to the kernel at the time of the line request. The mutators don't return error values. If the values are invalid, in general they are silently adjusted to acceptable ranges.

4.10.2 Function Documentation

4.10.2.1 gpiod_request_config_free()

```
void gpiod_request_config_free (
            struct gpiod_request_config * config )
```

```
#include <gpiod.h>
```

Free the request config object and release all associated resources.

Parameters

config	Line config object.

4.10.2.2 gpiod_request_config_get_consumer()

```
const char * gpiod_request_config_get_consumer (
            struct gpiod_request_config * config )
```

```
#include <gpiod.h>
```

Get the consumer name configured in the request config.

Parameters

config	Request config object.

Returns

Consumer name stored in the request config.

4.10.2.3 gpiod_request_config_get_event_buffer_size()

```
size_t gpiod_request_config_get_event_buffer_size (
            struct gpiod_request_config * config )
```

```
#include <gpiod.h>
```

Get the edge event buffer size for the request config.

Parameters

config	Request config object.

Returns

Edge event buffer size setting from the request config.

4.10.2.4 gpiod_request_config_new()

```
struct gpiod_request_config * gpiod_request_config_new (
            void  )
```

```
#include <gpiod.h>
```

Create a new request config object.

Returns

New request config object or NULL on error. The returned object must be freed by the caller using gpiod_request_config_free.

4.10.2.5 gpiod_request_config_set_consumer()

```
void gpiod_request_config_set_consumer (
            struct gpiod_request_config * config,
            const char * consumer )
```

```
#include <gpiod.h>
```

Set the consumer name for the request.

Parameters

config	Request config object.
consumer	Consumer name.

Note

If the consumer string is too long, it will be truncated to the max accepted length.

4.10.2.6 gpiod_request_config_set_event_buffer_size()

```
void gpiod_request_config_set_event_buffer_size (
            struct gpiod_request_config * config,
            size_t event_buffer_size )
```

```
#include <gpiod.h>
```

Set the size of the kernel event buffer for the request.

Parameters

config	Request config object.
event_buffer_size	New event buffer size.

Note

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

The kernel buffer is distinct from and independent of the user space buffer (gpiod_edge_event_buffer_new).

4.11 Stuff that didn't fit anywhere else

Functions

- bool gpiod_is_gpiochip_device (const char *path)

  Check if the file pointed to by path is a GPIO chip character device.
- const char * gpiod_api_version (void)

  Get the API version of the library as a human-readable string.

4.11.1 Detailed Description

Various libgpiod-related functions.

4.11.2 Function Documentation

4.11.2.1 gpiod_api_version()

```
const char * gpiod_api_version (
            void  )
```

```
#include <gpiod.h>
```

Get the API version of the library as a human-readable string.

Returns

A valid pointer to a human-readable string containing the library version. The pointer is valid for the lifetime of the program and must not be freed by the caller.

4.11.2.2 gpiod_is_gpiochip_device()

```
bool gpiod_is_gpiochip_device (
            const char * path )
```

```
#include <gpiod.h>
```

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 either a GPIO chip character device or a symbolic link to one.

Chapter 5

Class Documentation

5.1 gpiod_chip Struct Reference

`#include <gpiod.h>`

5.1.1 Detailed Description

Refer to GPIO chips for functions that operate on gpiod_chip.

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

- gpiod.h

5.2 gpiod_chip_info Struct Reference

`#include <gpiod.h>`

5.2.1 Detailed Description

Refer to Chip info for functions that operate on gpiod_chip_info.

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

- gpiod.h

5.3 gpiod_edge_event Struct Reference

`#include <gpiod.h>`

5.3.1 Detailed Description

Refer to Line edge events handling for functions that operate on gpiod_edge_event.

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

- gpiod.h

5.4 gpiod_edge_event_buffer Struct Reference

`#include <gpiod.h>`

5.4.1 Detailed Description

Refer to Line edge events handling for functions that operate on gpiod_edge_event_buffer.

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

- gpiod.h

5.5 gpiod_info_event Struct Reference

`#include <gpiod.h>`

5.5.1 Detailed Description

Refer to Line status watch events for functions that operate on gpiod_info_event.

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

- gpiod.h

5.6 gpiod_line_config Struct Reference

`#include <gpiod.h>`

5.6.1 Detailed Description

Refer to Line configuration objects for functions that operate on gpiod_line_config.

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

- gpiod.h

5.7 gpiod_line_info Struct Reference

`#include <gpiod.h>`

5.7.1 Detailed Description

Refer to Line info for functions that operate on gpiod_line_info.

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

- gpiod.h

5.8 gpiod_line_request Struct Reference

`#include <gpiod.h>`

5.8.1 Detailed Description

Refer to Line request operations for functions that operate on gpiod_line_request.

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

- gpiod.h

5.9 gpiod_line_settings Struct Reference

`#include <gpiod.h>`

5.9.1 Detailed Description

Refer to Line settings objects for functions that operate on gpiod_line_settings.

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

- gpiod.h

5.10 gpiod_request_config Struct Reference

`#include <gpiod.h>`

5.10.1 Detailed Description

Refer to Request configuration objects for functions that operate on gpiod_request_config.

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

- gpiod.h

Chapter 6

File Documentation

6.1 gpiod.h File Reference

```
#include <stdbool.h>
#include <stddef.h>
#include <stdint.h>
```

Enumerations

- enum gpiod_line_value { GPIOD_LINE_VALUE_ERROR = -1 , GPIOD_LINE_VALUE_INACTIVE = 0 , GPIOD_LINE_VALUE_ACTIVE = 1 }

  Logical line state.
- enum gpiod_line_direction { GPIOD_LINE_DIRECTION_AS_IS = 1 , GPIOD_LINE_DIRECTION_INPUT , GPIOD_LINE_DIRECTION_OUTPUT }

  Direction settings.
- enum gpiod_line_edge { GPIOD_LINE_EDGE_NONE = 1 , GPIOD_LINE_EDGE_RISING , GPIOD_LINE_EDGE_FALLING , GPIOD_LINE_EDGE_BOTH }

  Edge detection settings.
- enum gpiod_line_bias { GPIOD_LINE_BIAS_AS_IS = 1 , GPIOD_LINE_BIAS_UNKNOWN , GPIOD_LINE_BIAS_DISABLED , GPIOD_LINE_BIAS_PULL_UP , GPIOD_LINE_BIAS_PULL_DOWN }

  Internal bias settings.
- enum gpiod_line_drive { GPIOD_LINE_DRIVE_PUSH_PULL = 1 , GPIOD_LINE_DRIVE_OPEN_DRAIN , GPIOD_LINE_DRIVE_OPEN_SOURCE }

  Drive settings.
- enum gpiod_line_clock { GPIOD_LINE_CLOCK_MONOTONIC = 1 , GPIOD_LINE_CLOCK_REALTIME , GPIOD_LINE_CLOCK_HTE }

  Clock settings.
- enum gpiod_info_event_type { GPIOD_INFO_EVENT_LINE_REQUESTED = 1 , GPIOD_INFO_EVENT_LINE_RELEASED , GPIOD_INFO_EVENT_LINE_CONFIG_CHANGED }

  Line status change event types.
- enum gpiod_edge_event_type { GPIOD_EDGE_EVENT_RISING_EDGE = 1 , GPIOD_EDGE_EVENT_FALLING_EDGE }

  Event types.

Functions

- struct gpiod_chip * gpiod_chip_open (const char *path)

  Open a chip by path.
- void gpiod_chip_close (struct gpiod_chip *chip)

  Close the chip and release all associated resources.
- struct gpiod_chip_info * gpiod_chip_get_info (struct gpiod_chip *chip)

  Get information about the chip.
- const char * gpiod_chip_get_path (struct gpiod_chip *chip)

  Get the path used to open the chip.
- struct gpiod_line_info * gpiod_chip_get_line_info (struct gpiod_chip *chip, unsigned int offset)

  Get a snapshot of information about a line.
- struct gpiod_line_info * gpiod_chip_watch_line_info (struct gpiod_chip *chip, unsigned int offset)

  Get a snapshot of the status of a line and start watching it for future changes.
- int gpiod_chip_unwatch_line_info (struct gpiod_chip *chip, unsigned int offset)

  Stop watching a line for status changes.
- int gpiod_chip_get_fd (struct gpiod_chip *chip)

  Get the file descriptor associated with the chip.
- int gpiod_chip_wait_info_event (struct gpiod_chip *chip, int64_t timeout_ns)

  Wait for line status change events on any of the watched lines on the chip.
- struct gpiod_info_event * gpiod_chip_read_info_event (struct gpiod_chip *chip)

  Read a single line status change event from the chip.
- int gpiod_chip_get_line_offset_from_name (struct gpiod_chip *chip, const char *name)

  Map a line's name to its offset within the chip.
- struct gpiod_line_request * gpiod_chip_request_lines (struct gpiod_chip *chip, struct gpiod_request_config *req_cfg, struct gpiod_line_config *line_cfg)

  Request a set of lines for exclusive usage.
- void gpiod_chip_info_free (struct gpiod_chip_info *info)

  Free a chip info object and release all associated resources.
- const char * gpiod_chip_info_get_name (struct gpiod_chip_info *info)

  Get the name of the chip as represented in the kernel.
- const char * gpiod_chip_info_get_label (struct gpiod_chip_info *info)

  Get the label of the chip as represented in the kernel.
- size_t gpiod_chip_info_get_num_lines (struct gpiod_chip_info *info)

  Get the number of lines exposed by the chip.
- void gpiod_line_info_free (struct gpiod_line_info *info)

  Free a line info object and release all associated resources.
- struct gpiod_line_info * gpiod_line_info_copy (struct gpiod_line_info *info)

  Copy a line info object.
- unsigned int gpiod_line_info_get_offset (struct gpiod_line_info *info)

  Get the offset of the line.
- const char * gpiod_line_info_get_name (struct gpiod_line_info *info)

  Get the name of the line.
- bool gpiod_line_info_is_used (struct gpiod_line_info *info)

  Check if the line is in use.
- const char * gpiod_line_info_get_consumer (struct gpiod_line_info *info)

  Get the name of the consumer of the line.
- enum gpiod_line_direction gpiod_line_info_get_direction (struct gpiod_line_info *info)

  Get the direction setting of the line.
- enum gpiod_line_edge gpiod_line_info_get_edge_detection (struct gpiod_line_info *info)

  Get the edge detection setting of the line.

- enum gpiod_line_bias gpiod_line_info_get_bias (struct gpiod_line_info *info)

  Get the bias setting of the line.
- enum gpiod_line_drive gpiod_line_info_get_drive (struct gpiod_line_info *info)

  Get the drive setting of the line.
- bool gpiod_line_info_is_active_low (struct gpiod_line_info *info)

  Check if the logical value of the line is inverted compared to the physical.
- bool gpiod_line_info_is_debounced (struct gpiod_line_info *info)

  Check if the line is debounced (either by hardware or by the kernel software debouncer).
- unsigned long gpiod_line_info_get_debounce_period_us (struct gpiod_line_info *info)

  Get the debounce period of the line, in microseconds.
- enum gpiod_line_clock gpiod_line_info_get_event_clock (struct gpiod_line_info *info)

  Get the event clock setting used for edge event timestamps for the line.
- void gpiod_info_event_free (struct gpiod_info_event *event)

  Free the info event object and release all associated resources.
- enum gpiod_info_event_type gpiod_info_event_get_event_type (struct gpiod_info_event *event)

  Get the event type of the status change event.
- uint64_t gpiod_info_event_get_timestamp_ns (struct gpiod_info_event *event)

  Get the timestamp of the event.
- struct gpiod_line_info * gpiod_info_event_get_line_info (struct gpiod_info_event *event)

  Get the snapshot of line-info associated with the event.
- struct gpiod_line_settings * gpiod_line_settings_new (void)

  Create a new line settings object.
- void gpiod_line_settings_free (struct gpiod_line_settings *settings)

  Free the line settings object and release all associated resources.
- void gpiod_line_settings_reset (struct gpiod_line_settings *settings)

  Reset the line settings object to its default values.
- struct gpiod_line_settings * gpiod_line_settings_copy (struct gpiod_line_settings *settings)

  Copy the line settings object.
- int gpiod_line_settings_set_direction (struct gpiod_line_settings *settings, enum gpiod_line_direction direction)

  Set direction.
- enum gpiod_line_direction gpiod_line_settings_get_direction (struct gpiod_line_settings *settings)

  Get direction.
- int gpiod_line_settings_set_edge_detection (struct gpiod_line_settings *settings, enum gpiod_line_edge edge)

  Set edge detection.
- enum gpiod_line_edge gpiod_line_settings_get_edge_detection (struct gpiod_line_settings *settings)

  Get edge detection.
- int gpiod_line_settings_set_bias (struct gpiod_line_settings *settings, enum gpiod_line_bias bias)

  Set bias.
- enum gpiod_line_bias gpiod_line_settings_get_bias (struct gpiod_line_settings *settings)

  Get bias.
- int gpiod_line_settings_set_drive (struct gpiod_line_settings *settings, enum gpiod_line_drive drive)

  Set drive.
- enum gpiod_line_drive gpiod_line_settings_get_drive (struct gpiod_line_settings *settings)

  Get drive.
- void gpiod_line_settings_set_active_low (struct gpiod_line_settings *settings, bool active_low)

  Set active-low setting.
- bool gpiod_line_settings_get_active_low (struct gpiod_line_settings *settings)

  Get active-low setting.

- void gpiod_line_settings_set_debounce_period_us (struct gpiod_line_settings *settings, unsigned long period)

  Set debounce period.
- unsigned long gpiod_line_settings_get_debounce_period_us (struct gpiod_line_settings *settings)

  Get debounce period.
- int gpiod_line_settings_set_event_clock (struct gpiod_line_settings *settings, enum gpiod_line_clock event_clock)

  Set event clock.
- enum gpiod_line_clock gpiod_line_settings_get_event_clock (struct gpiod_line_settings *settings)

  Get event clock setting.
- int gpiod_line_settings_set_output_value (struct gpiod_line_settings *settings, enum gpiod_line_value value)

  Set the output value.
- enum gpiod_line_value gpiod_line_settings_get_output_value (struct gpiod_line_settings *settings)

  Get the output value.
- struct gpiod_line_config * gpiod_line_config_new (void)

  Create a new line config object.
- void gpiod_line_config_free (struct gpiod_line_config *config)

  Free the line config object and release all associated resources.
- void gpiod_line_config_reset (struct gpiod_line_config *config)

  Reset the line config object.
- int gpiod_line_config_add_line_settings (struct gpiod_line_config *config, const unsigned int *offsets, size_t num_offsets, struct gpiod_line_settings *settings)

  Add line settings for a set of offsets.
- struct gpiod_line_settings * gpiod_line_config_get_line_settings (struct gpiod_line_config *config, unsigned int offset)

  Get line settings for offset.
- int gpiod_line_config_set_output_values (struct gpiod_line_config *config, const enum gpiod_line_value *values, size_t num_values)

  Set output values for a number of lines.
- size_t gpiod_line_config_get_num_configured_offsets (struct gpiod_line_config *config)

  Get the number of configured line offsets.
- size_t gpiod_line_config_get_configured_offsets (struct gpiod_line_config *config, unsigned int *offsets, size_t max_offsets)

  Get configured offsets.
- struct gpiod_request_config * gpiod_request_config_new (void)

  Create a new request config object.
- void gpiod_request_config_free (struct gpiod_request_config *config)

  Free the request config object and release all associated resources.
- void gpiod_request_config_set_consumer (struct gpiod_request_config *config, const char *consumer)

  Set the consumer name for the request.
- const char * gpiod_request_config_get_consumer (struct gpiod_request_config *config)

  Get the consumer name configured in the request config.
- void gpiod_request_config_set_event_buffer_size (struct gpiod_request_config *config, size_t event_buffer_size)

  Set the size of the kernel event buffer for the request.
- size_t gpiod_request_config_get_event_buffer_size (struct gpiod_request_config *config)

  Get the edge event buffer size for the request config.
- void gpiod_line_request_release (struct gpiod_line_request *request)

  Release the requested lines and free all associated resources.
- const char * gpiod_line_request_get_chip_name (struct gpiod_line_request *request)

  Get the name of the chip this request was made on.
- size_t gpiod_line_request_get_num_requested_lines (struct gpiod_line_request *request)

Get the number of lines in the request.

- size_t gpiod_line_request_get_requested_offsets (struct gpiod_line_request *request, unsigned int *offsets, size_t max_offsets)

  Get the offsets of the lines in the request.
- enum gpiod_line_value gpiod_line_request_get_value (struct gpiod_line_request *request, unsigned int offset)

  Get the value of a single requested line.
- int gpiod_line_request_get_values_subset (struct gpiod_line_request *request, size_t num_values, const unsigned int *offsets, enum gpiod_line_value *values)

  Get the values of a subset of requested lines.
- int gpiod_line_request_get_values (struct gpiod_line_request *request, enum gpiod_line_value *values)

  Get the values of all requested lines.
- int gpiod_line_request_set_value (struct gpiod_line_request *request, unsigned int offset, enum gpiod_line_value value)

  Set the value of a single requested line.
- int gpiod_line_request_set_values_subset (struct gpiod_line_request *request, size_t num_values, const unsigned int *offsets, const enum gpiod_line_value *values)

  Set the values of a subset of requested lines.
- int gpiod_line_request_set_values (struct gpiod_line_request *request, const enum gpiod_line_value *values)

  Set the values of all lines associated with a request.
- int gpiod_line_request_reconfigure_lines (struct gpiod_line_request *request, struct gpiod_line_config *config)

  Update the configuration of lines associated with a line request.
- int gpiod_line_request_get_fd (struct gpiod_line_request *request)

  Get the file descriptor associated with a line request.
- int gpiod_line_request_wait_edge_events (struct gpiod_line_request *request, int64_t timeout_ns)

  Wait for edge events on any of the requested lines.
- int gpiod_line_request_read_edge_events (struct gpiod_line_request *request, struct gpiod_edge_event_buffer *buffer, size_t max_events)

  Read a number of edge events from a line request.
- void gpiod_edge_event_free (struct gpiod_edge_event *event)

  Free the edge event object.
- struct gpiod_edge_event * gpiod_edge_event_copy (struct gpiod_edge_event *event)

  Copy the edge event object.
- enum gpiod_edge_event_type gpiod_edge_event_get_event_type (struct gpiod_edge_event *event)

  Get the event type.
- uint64_t gpiod_edge_event_get_timestamp_ns (struct gpiod_edge_event *event)

  Get the timestamp of the event.
- unsigned int gpiod_edge_event_get_line_offset (struct gpiod_edge_event *event)

  Get the offset of the line which triggered the event.
- unsigned long gpiod_edge_event_get_global_seqno (struct gpiod_edge_event *event)

  Get the global sequence number of the event.
- unsigned long gpiod_edge_event_get_line_seqno (struct gpiod_edge_event *event)

  Get the event sequence number specific to the line.
- struct gpiod_edge_event_buffer * gpiod_edge_event_buffer_new (size_t capacity)

  Create a new edge event buffer.
- size_t gpiod_edge_event_buffer_get_capacity (struct gpiod_edge_event_buffer *buffer)

  Get the capacity (the max number of events that can be stored) of the event buffer.
- void gpiod_edge_event_buffer_free (struct gpiod_edge_event_buffer *buffer)

  Free the edge event buffer and release all associated resources.

- struct gpiod_edge_event * gpiod_edge_event_buffer_get_event (struct gpiod_edge_event_buffer *buffer, unsigned long index)

  Get an event stored in the buffer.
- size_t gpiod_edge_event_buffer_get_num_events (struct gpiod_edge_event_buffer *buffer)

  Get the number of events a buffer has stored.
- bool gpiod_is_gpiochip_device (const char *path)

  Check if the file pointed to by path is a GPIO chip character device.
- const char * gpiod_api_version (void)

  Get the API version of the library as a human-readable string.

6.2 gpiod.h

Go to the documentation of this file.

```
1 /* SPDX-License-Identifier:  LGPL-2.1-or-later */
2 /* SPDX-FileCopyrightText:  2017-2022 Bartosz Golaszewski <brgl@bgdev.pl> */
3
8 #ifndef __LIBGPIOD_GPIOD_H__
9 #define __LIBGPIOD_GPIOD_H__
10
11 #include <stdbool.h>
12 #include <stddef.h>
13 #include <stdint.h>
14
15 #ifdef __cplusplus
16 extern "C" {
17 #endif
18
27 struct gpiod_chip;
28
37 struct gpiod_chip_info;
38
47 struct gpiod_line_info;
48
58 struct gpiod_line_settings;
59
68 struct gpiod_line_config;
69
79 struct gpiod_request_config;
80
90 struct gpiod_line_request;
91
100 struct gpiod_info_event;
101
110 struct gpiod_edge_event;
111
121 struct gpiod_edge_event_buffer;
122
141 struct gpiod_chip *gpiod_chip_open(const char *path);
142
147 void gpiod_chip_close(struct gpiod_chip *chip);
148
155 struct gpiod_chip_info *gpiod_chip_get_info(struct gpiod_chip *chip);
156
164 const char *gpiod_chip_get_path(struct gpiod_chip *chip);
165
173 struct gpiod_line_info *gpiod_chip_get_line_info(struct gpiod_chip *chip,
174                           unsigned int offset);
175
186 struct gpiod_line_info *gpiod_chip_watch_line_info(struct gpiod_chip *chip,
187                             unsigned int offset);
188
195 int gpiod_chip_unwatch_line_info(struct gpiod_chip *chip, unsigned int offset);
196
206 int gpiod_chip_get_fd(struct gpiod_chip *chip);
207
219 int gpiod_chip_wait_info_event(struct gpiod_chip *chip, int64_t timeout_ns);
220
228 struct gpiod_info_event *gpiod_chip_read_info_event(struct gpiod_chip *chip);
229
238 int gpiod_chip_get_line_offset_from_name(struct gpiod_chip *chip,
239                         const char *name);
240
249 struct gpiod_line_request *
250 gpiod_chip_request_lines(struct gpiod_chip *chip,
251              struct gpiod_request_config *req_cfg,
```

```
252             struct gpiod_line_config *line_cfg);
253
278 void gpiod_chip_info_free(struct gpiod_chip_info *info);
279
287 const char *gpiod_chip_info_get_name(struct gpiod_chip_info *info);
288
296 const char *gpiod_chip_info_get_label(struct gpiod_chip_info *info);
297
303 size_t gpiod_chip_info_get_num_lines(struct gpiod_chip_info *info);
304
317 enum gpiod_line_value {
318     GPIOD_LINE_VALUE_ERROR = -1,
320     GPIOD_LINE_VALUE_INACTIVE = 0,
322     GPIOD_LINE_VALUE_ACTIVE = 1,
324 };
325
329 enum gpiod_line_direction {
330     GPIOD_LINE_DIRECTION_AS_IS = 1,
332     GPIOD_LINE_DIRECTION_INPUT,
335     GPIOD_LINE_DIRECTION_OUTPUT,
337 };
338
342 enum gpiod_line_edge {
343     GPIOD_LINE_EDGE_NONE = 1,
345     GPIOD_LINE_EDGE_RISING,
347     GPIOD_LINE_EDGE_FALLING,
349     GPIOD_LINE_EDGE_BOTH,
351 };
352
356 enum gpiod_line_bias {
357     GPIOD_LINE_BIAS_AS_IS = 1,
359     GPIOD_LINE_BIAS_UNKNOWN,
361     GPIOD_LINE_BIAS_DISABLED,
363     GPIOD_LINE_BIAS_PULL_UP,
365     GPIOD_LINE_BIAS_PULL_DOWN,
367 };
368
372 enum gpiod_line_drive {
373     GPIOD_LINE_DRIVE_PUSH_PULL = 1,
375     GPIOD_LINE_DRIVE_OPEN_DRAIN,
377     GPIOD_LINE_DRIVE_OPEN_SOURCE,
379 };
380
384 enum gpiod_line_clock {
385     GPIOD_LINE_CLOCK_MONOTONIC = 1,
387     GPIOD_LINE_CLOCK_REALTIME,
389     GPIOD_LINE_CLOCK_HTE,
391 };
392
419 void gpiod_line_info_free(struct gpiod_line_info *info);
420
427 struct gpiod_line_info *gpiod_line_info_copy(struct gpiod_line_info *info);
428
437 unsigned int gpiod_line_info_get_offset(struct gpiod_line_info *info);
438
447 const char *gpiod_line_info_get_name(struct gpiod_line_info *info);
448
459 bool gpiod_line_info_is_used(struct gpiod_line_info *info);
460
470 const char *gpiod_line_info_get_consumer(struct gpiod_line_info *info);
471
478 enum gpiod_line_direction
479 gpiod_line_info_get_direction(struct gpiod_line_info *info);
480
487 enum gpiod_line_edge
488 gpiod_line_info_get_edge_detection(struct gpiod_line_info *info);
489
496 enum gpiod_line_bias
497 gpiod_line_info_get_bias(struct gpiod_line_info *info);
498
505 enum gpiod_line_drive
506 gpiod_line_info_get_drive(struct gpiod_line_info *info);
507
514 bool gpiod_line_info_is_active_low(struct gpiod_line_info *info);
515
522 bool gpiod_line_info_is_debounced(struct gpiod_line_info *info);
523
530 unsigned long
531 gpiod_line_info_get_debounce_period_us(struct gpiod_line_info *info);
532
540 enum gpiod_line_clock
541 gpiod_line_info_get_event_clock(struct gpiod_line_info *info);
542
561 enum gpiod_info_event_type {
562     GPIOD_INFO_EVENT_LINE_REQUESTED = 1,
564     GPIOD_INFO_EVENT_LINE_RELEASED,
566     GPIOD_INFO_EVENT_LINE_CONFIG_CHANGED,
```

```
568 };
569
574 void gpiod_info_event_free(struct gpiod_info_event *event);
575
583 enum gpiod_info_event_type
584 gpiod_info_event_get_event_type(struct gpiod_info_event *event);
585
591 uint64_t gpiod_info_event_get_timestamp_ns(struct gpiod_info_event *event);
592
606 struct gpiod_line_info *
607 gpiod_info_event_get_line_info(struct gpiod_info_event *event);
608
631 struct gpiod_line_settings *gpiod_line_settings_new(void);
632
637 void gpiod_line_settings_free(struct gpiod_line_settings *settings);
638
643 void gpiod_line_settings_reset(struct gpiod_line_settings *settings);
644
651 struct gpiod_line_settings *
652 gpiod_line_settings_copy(struct gpiod_line_settings *settings);
653
660 int gpiod_line_settings_set_direction(struct gpiod_line_settings *settings,
661                       enum gpiod_line_direction direction);
662
668 enum gpiod_line_direction
669 gpiod_line_settings_get_direction(struct gpiod_line_settings *settings);
670
677 int gpiod_line_settings_set_edge_detection(struct gpiod_line_settings *settings,
678                        enum gpiod_line_edge edge);
679
685 enum gpiod_line_edge
686 gpiod_line_settings_get_edge_detection(struct gpiod_line_settings *settings);
687
694 int gpiod_line_settings_set_bias(struct gpiod_line_settings *settings,
695                  enum gpiod_line_bias bias);
696
702 enum gpiod_line_bias
703 gpiod_line_settings_get_bias(struct gpiod_line_settings *settings);
704
711 int gpiod_line_settings_set_drive(struct gpiod_line_settings *settings,
712                   enum gpiod_line_drive drive);
713
719 enum gpiod_line_drive
720 gpiod_line_settings_get_drive(struct gpiod_line_settings *settings);
721
727 void gpiod_line_settings_set_active_low(struct gpiod_line_settings *settings,
728                     bool active_low);
729
735 bool gpiod_line_settings_get_active_low(struct gpiod_line_settings *settings);
736
742 void
743 gpiod_line_settings_set_debounce_period_us(struct gpiod_line_settings *settings,
744                        unsigned long period);
745
751 unsigned long
752 gpiod_line_settings_get_debounce_period_us(
753         struct gpiod_line_settings *settings);
754
761 int gpiod_line_settings_set_event_clock(struct gpiod_line_settings *settings,
762                     enum gpiod_line_clock event_clock);
763
769 enum gpiod_line_clock
770 gpiod_line_settings_get_event_clock(struct gpiod_line_settings *settings);
771
778 int gpiod_line_settings_set_output_value(struct gpiod_line_settings *settings,
779                      enum gpiod_line_value value);
780
786 enum gpiod_line_value
787 gpiod_line_settings_get_output_value(struct gpiod_line_settings *settings);
788
817 struct gpiod_line_config *gpiod_line_config_new(void);
818
823 void gpiod_line_config_free(struct gpiod_line_config *config);
824
832 void gpiod_line_config_reset(struct gpiod_line_config *config);
833
842 int gpiod_line_config_add_line_settings(struct gpiod_line_config *config,
843                     const unsigned int *offsets,
844                     size_t num_offsets,
845                     struct gpiod_line_settings *settings);
846
854 struct gpiod_line_settings *
855 gpiod_line_config_get_line_settings(struct gpiod_line_config *config,
856                     unsigned int offset);
857
879 int gpiod_line_config_set_output_values(struct gpiod_line_config *config,
880                     const enum gpiod_line_value *values,
```

```
881                     size_t num_values);
882
888 size_t
889 gpiod_line_config_get_num_configured_offsets(struct gpiod_line_config *config);
890
902 size_t
903 gpiod_line_config_get_configured_offsets(struct gpiod_line_config *config,
904                       unsigned int *offsets,
905                       size_t max_offsets);
906
926 struct gpiod_request_config *gpiod_request_config_new(void);
927
932 void gpiod_request_config_free(struct gpiod_request_config *config);
933
941 void gpiod_request_config_set_consumer(struct gpiod_request_config *config,
942                         const char *consumer);
943
949 const char *
950 gpiod_request_config_get_consumer(struct gpiod_request_config *config);
951
961 void
962 gpiod_request_config_set_event_buffer_size(struct gpiod_request_config *config,
963                         size_t event_buffer_size);
964
970 size_t
971 gpiod_request_config_get_event_buffer_size(struct gpiod_request_config *config);
972
986 void gpiod_line_request_release(struct gpiod_line_request *request);
987
994 const char *
995 gpiod_line_request_get_chip_name(struct gpiod_line_request *request);
996
1002 size_t
1003 gpiod_line_request_get_num_requested_lines(struct gpiod_line_request *request);
1004
1016 size_t
1017 gpiod_line_request_get_requested_offsets(struct gpiod_line_request *request,
1018                       unsigned int *offsets,
1019                       size_t max_offsets);
1020
1027 enum gpiod_line_value
1028 gpiod_line_request_get_value(struct gpiod_line_request *request,
1029                   unsigned int offset);
1030
1042 int gpiod_line_request_get_values_subset(struct gpiod_line_request *request,
1043                       size_t num_values,
1044                       const unsigned int *offsets,
1045                       enum gpiod_line_value *values);
1046
1058 int gpiod_line_request_get_values(struct gpiod_line_request *request,
1059                    enum gpiod_line_value *values);
1060
1068 int gpiod_line_request_set_value(struct gpiod_line_request *request,
1069                   unsigned int offset,
1070                   enum gpiod_line_value value);
1071
1084 int gpiod_line_request_set_values_subset(struct gpiod_line_request *request,
1085                       size_t num_values,
1086                       const unsigned int *offsets,
1087                       const enum gpiod_line_value *values);
1088
1100 int gpiod_line_request_set_values(struct gpiod_line_request *request,
1101                    const enum gpiod_line_value *values);
1102
1114 int gpiod_line_request_reconfigure_lines(struct gpiod_line_request *request,
1115                       struct gpiod_line_config *config);
1116
1125 int gpiod_line_request_get_fd(struct gpiod_line_request *request);
1126
1140 int gpiod_line_request_wait_edge_events(struct gpiod_line_request *request,
1141                       int64_t timeout_ns);
1142
1154 int gpiod_line_request_read_edge_events(struct gpiod_line_request *request,
1155                       struct gpiod_edge_event_buffer *buffer,
1156                       size_t max_events);
1157
1179 enum gpiod_edge_event_type {
1180     GPIOD_EDGE_EVENT_RISING_EDGE = 1,
1182     GPIOD_EDGE_EVENT_FALLING_EDGE,
1184 };
1185
1190 void gpiod_edge_event_free(struct gpiod_edge_event *event);
1191
1198 struct gpiod_edge_event *gpiod_edge_event_copy(struct gpiod_edge_event *event);
1199
1206 enum gpiod_edge_event_type
1207 gpiod_edge_event_get_event_type(struct gpiod_edge_event *event);
```

```
1208
1216 uint64_t gpiod_edge_event_get_timestamp_ns(struct gpiod_edge_event *event);
1217
1223 unsigned int gpiod_edge_event_get_line_offset(struct gpiod_edge_event *event);
1224
1231 unsigned long gpiod_edge_event_get_global_seqno(struct gpiod_edge_event *event);
1232
1239 unsigned long gpiod_edge_event_get_line_seqno(struct gpiod_edge_event *event);
1240
1253 struct gpiod_edge_event_buffer *
1254 gpiod_edge_event_buffer_new(size_t capacity);
1255
1262 size_t
1263 gpiod_edge_event_buffer_get_capacity(struct gpiod_edge_event_buffer *buffer);
1264
1269 void gpiod_edge_event_buffer_free(struct gpiod_edge_event_buffer *buffer);
1270
1285 struct gpiod_edge_event *
1286 gpiod_edge_event_buffer_get_event(struct gpiod_edge_event_buffer *buffer,
1287                   unsigned long index);
1288
1294 size_t
1295 gpiod_edge_event_buffer_get_num_events(struct gpiod_edge_event_buffer *buffer);
1296
1312 bool gpiod_is_gpiochip_device(const char *path);
1313
1320 const char *gpiod_api_version(void);
1321
1326 #ifdef __cplusplus
1327 } /* extern "C" */
1328 #endif
1329
1330 #endif /* __LIBGPIOD_GPIOD_H__ */
```

Index