

**NAME**

**gpio\_open**, **gpio\_close** - library to handle GPIO pins

**LIBRARY**

General-Purpose Input Output (GPIO) library (libgpio, -lgpio)

**SYNOPSIS**

```
#include <sys/types.h>
```

```
#include <libgpio.h>
```

```
gpio_handle_t
```

```
gpio_open(unsigned int unit);
```

```
gpio_handle_t
```

```
gpio_open_device(const char *device);
```

```
void
```

```
gpio_close(gpio_handle_t handle);
```

```
int
```

```
gpio_pin_list(gpio_handle_t handle, gpio_config_t **pcfgs);
```

```
int
```

```
gpio_pin_config(gpio_handle_t handle, gpio_config_t *cfg);
```

```
int
```

```
gpio_pin_set_name(gpio_handle_t handle, gpio_pin_t pin, char *name);
```

```
int
```

```
gpio_pin_set_flags(gpio_handle_t handle, gpio_config_t *cfg);
```

```
gpio_value_t
```

```
gpio_pin_get(gpio_handle_t handle, gpio_pin_t pin);
```

```
int
```

```
gpio_pin_set(gpio_handle_t handle, gpio_pin_t pin, gpio_value_t value);
```

```
int
```

```
gpio_pin_toggle(gpio_handle_t handle, gpio_pin_t pin);
```

*int*  
**gpio\_pin\_low**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_high**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_input**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_output**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_opendrain**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_pushpull**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_tristate**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_pullup**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_pulldown**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_invin**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_invout**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

*int*  
**gpio\_pin\_pulsate**(*gpio\_handle\_t handle, gpio\_pin\_t pin*);

## DESCRIPTION

The **libgpio** library provides an interface to configure GPIO pins. The library operates with a *gpio\_handle\_t* opaque type which can be created with **gpio\_open()** or **gpio\_open\_device()**. When no more GPIO operations are needed, this handle can be destroyed with **gpio\_close()**.

To get a list of all available pins, one can call **gpio\_pin\_list()**. This function takes a pointer to a *gpio\_config\_t* which is dynamically allocated. This pointer should be freed with `free(3)` when it is no longer necessary.

The function **gpio\_pin\_config()** retrieves the current configuration of a pin. The pin number should be passed in via the *g\_pin* variable which is part of the *gpio\_config\_t* structure.

The function **gpio\_pin\_set\_name()** sets the name used to describe a pin.

The function **gpio\_pin\_set\_flags()** configures a pin with the flags passed in by the *gpio\_config\_t* structure. The pin number should also be passed in through the *g\_pin* variable. All other structure members will be ignored by this function. The list of flags can be found in */usr/include/sys/gpio.h*.

To get or set the state of a GPIO pin, the functions **gpio\_pin\_get()** and **gpio\_pin\_set()** are available, respectively. To toggle the state, use **gpio\_pin\_toggle()**.

The functions **gpio\_pin\_low()** and **gpio\_pin\_high()** are wrappers around **gpio\_pin\_set()**.

The functions **gpio\_pin\_input()**, **gpio\_pin\_output()**, **gpio\_pin\_opendrain()**, **gpio\_pin\_pushpull()**, **gpio\_pin\_tristate()**, **gpio\_pin\_pullup()**, **gpio\_pin\_pulldown()**, **gpio\_pin\_invin()**, **gpio\_pin\_invout()** and **gpio\_pin\_pulsate()** are wrappers around **gpio\_pin\_set\_flags()**.

## EXAMPLES

The following example shows how to configure pin 16 as output and then drive it high:

```
#include <sys/types.h>
#include <err.h>
#include <libgpio.h>

gpio_handle_t handle;

handle = gpio_open(0);
if (handle == GPIO_INVALID_HANDLE)
    err(1, "gpio_open failed");
gpio_pin_output(handle, 16);
gpio_pin_high(handle, 16);
gpio_close(handle);
```

The following example shows how to get a configuration of a pin:

```
gpio_config_t cfg;
```

```
cfg.g_pin = 32;  
gpio_pin_config(handle, &cfg);
```

The structure will contain the name of the pin and its flags.

#### SEE ALSO

gpiobus(4), gpioctl(8)

#### HISTORY

The **libgpio** library first appeared in FreeBSD 11.0.

#### AUTHORS

The **libgpio** library was implemented by Rui Paulo <[rpaulo@FreeBSD.org](mailto:rpaulo@FreeBSD.org)>.