NAME

libevent −− Munts Technologies Simple I/O Library for Linux: Event Notification Module

SYNOPSIS

#include <libevent.h>

void EVENT_init(int ****error*);**
void EVENT_close(int ****error*);**
void EVENT_register_fd(int *fd***, int** *events***, int ****error***);**
void EVENT_unregister_fd(int *fd***, int ****error***);**
void EVENT_wait(int * *fd***, int ****event***, int** *timeoutms***, int ****error***);**

Link with **-lsimpleio**

DESCRIPTION

All functions return **0** in **error* upon success or an **errno** value in **error* upon failure.

EVENT_init() must be called before any of the other functions.

EVENT_close() may be called to release any internal resources previously acquired by **EVENT_init()**.

EVENT_register_fd() may be called to register **epoll(7)** event notifications for the given file descriptor *fd*. Event codes such as **EPOLLIN** (input ready) are defined in the **/usr/include/sys/epoll.h** header file.

EVENT_unregister_fd() may be called to unregister event notifications for the given file descriptor *fd*.

EVENT_wait() may be called to wait until an event notification occurs. The *timeoutms* parameter indicates the time in milliseconds to wait for a notfication. If a notification occurs before the timeout expires, **error* is set to **0** and **fd* and **event* are set to the next available file descriptor and event code. If no notification occurs before the timeout expires, **error* is set to **EAGAIN** and **fd* and **event* are invalid.

SEE ALSO

libsimpleio(2), **libgpio**(2), **libhidraw**(2), **libi2c**(2), **libserial**(2),
libspi(2)

AUTHOR

Philip Munts, President, Munts AM Corp dba Munts Technologies

NAME

libgpio −− Munts Technologies Simple I/O Library for Linux: GPIO Module

SYNOPSIS

#include <libgpio.h>

void GPIO_configure(int *pin***, int** *direction***, int** *state***, int** *edge***, int** *polarity***, int ****error***);**
void GPIO_open(char ****name*, int *** *fd***, int ****error***);**
void GPIO_close(int *fd***, int ****error***);**
void GPIO_read(int *fd***, int ****state***, int ****error***);**
void GPIO_write(int *fd***, int** *state***, int ****error***);**

Link with **-lsimpleio**

DESCRIPTION

All functions return either **0** (upon success) or an **errno** value (upon failure) in **error*.

GPIO_configure() may be called to configure a single GPIO pin. The *pin* parameter selects the GPIO pin (as numbered by the Linux kernel) to be configured. The *direction* parameter may be **GPIO_DIRECTION_INPUT** or **GPIO_DIRECTION_OUTPUT**. For input pins, the *state* parameter must be **0**. For output pins, the *state* parameter may be **0** or **1** to set the initial state. For input pins, the *edge* parameter may be **GPIO_EDGE_NONE**, **GPIO_EDGE_RISING**, **GPIO_EDGE_FALLING**, or **GPIO_EDGE_BOTH**. For output pins, the *edge* parameter must be **GPIO_EDGE_NONE.** The *polarity* parameter may be **GPIO_ACTIVELOW** or **GPIO_ACTIVEHIGH**.

The **udev** rules included in the **libsimpleio** package will create a symbolic link from **/dev/gpioxx** to **/sys/class/gpio/gpioxx/value** when a GPIO pin is configured.

GPIO_open() may be called to open a GPIO pin device. Either **/sys/class/gpio/gpioxx/value** or **/dev/gpioxx** may be passed in the *name* parameter. Upon success, a file descriptor for the GPIO pin is returned in **fd*.

GPIO_close() may be called to close a GPIO pin device.

GPIO_read() may be called to get the current state of a GPIO pin. Upon success, the current state (**0** or **1**) of the GPIO pin will be returned in **state*.

GPIO_write() may be called to change a GPIO pin output state. The new state is passed in the *state* parameter.

SEE ALSO

libsimpleio(2), **libevent**(2), **libhidraw**(2), **libi2c**(2), **libserial**(2), **libspi**(2)

AUTHOR

Philip Munts, President, Munts AM Corp dba Munts Technologies

NAME

libsimpleio −− Munts Technologies Simple I/O Library for Linux

DESCRIPTION

libsimpleio is an attempt to encapsulate (as much as possible) the ugliness of Linux I/O device access. It provides services for the following types of I/O devices:

* GPIO (General Purpose Input/Output) Pins
* Raw HID (Human Interface Device) Devices
* I2C (Inter-Integrated Circuit) Bus Devices
* Serial Ports
* SPI (Serial Peripheral Interface) Bus Devices

Although **libsimpleio** was originally intended for Linux microcomputers such as the Raspberry Pi, it can also be useful on larger desktop Linux systems (particularly the raw HID and serial port services).

SEE ALSO

libevent(2), **libgpio**(2), **libhidraw**(2), **libi2c**(2), **libserial**(2),
libspi(2)

AUTHOR

Philip Munts, President, Munts AM Corp dba Munts Technologies