

# Analog to Digital Converter Services

These services allow reading from Linux kernel ADC input pins using **libsimpleio**.

## **fd = libsimpleio.adc\_open(chip, channel)**

This service opens an ADC input pin.

The **chip** and **channel** parameters select the ADC input pin.

This service returns a Linux file descriptor number that will be used as a handle for all of the other ADC services.

## **libsimpleio.adc\_close(fd)**

This service closes an ADC input pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimpleio.adc\_open()**.

## **sample = libsimpleio.adc\_read(fd)**

This service reads a single sample from an ADC input pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimpleio.adc\_open()**.

This service returns the sampled data value.

## **ADC Example Program**

```
fd = libsimpleio.adc_open(0, 0)

while true
  print "Sample: "
  print libsimpleio.adc_read(fd);
  delay(1000000)
wend
```

# General Purpose Input/Output Services

These services allow manipulating Linux kernel GPIO pins using **libsimpleio**.

## **fd = libsimplio.gpio\_open(chip, channel, dir, state)**

This service opens a GPIO pin.

The **chip** and **channel** parameters select the GPIO pin. The **dir** parameter selects the data direction (**0**=input, **1**=output). The **state** parameter selects the initial state for an output pin (**0**=off or low, **1**=on or high).

This service returns a Linux file descriptor number that will be used as a handle for all of the other GPIO services.

## **libsimplio.gpio\_close(fd)**

This service closes a GPIO pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimplio.gpio\_open()**.

## **state = libsimplio.gpio\_read(fd)**

This service reads from a GPIO pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimplio.gpio\_open()**.

This service returns the state of the GPIO pin (**0**=off or low, **1**=on or high)

## **libsimplio.gpio\_write(fd, state)**

This services writes to a GPIO pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimplio.gpio\_open()**.

The **state** parameter indicates the value written to the GPIO pin (**0**=off or low, **1**=on or high).

## **GPIO Example Program**

```
fd = libsimplio.gpio_open(0, 26, 1, 0)

while true
  libsimplio.gpio_write(fd, NOT libsimplio.gpio_read(fd))
wend
```

# Pulse Width Modulated Output Services

These services allow controlling Linux kernel PWM output pins using **libsimpleio**.

**fd = libsimpleio.pwm\_open(chip, channel, period, ontime)**

The **chip** and **channel** parameters select the PWM output pin.

The **period** parameter sets the PWM pulse period in nanoseconds.

The **ontime** parameter sets the initial PWM pulse width in nanoseconds.

**libsimpleio.pwm\_close(fd)**

This service closes a PWM output pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimpleio.pwm\_open()**.

**libsimpleio.pwm\_write(fd, ontime)**

This services writes to a PWM output pin.

The **fd** parameter must be a file descriptor number previously returned by **libsimpleio.pwm\_open()**.

The **ontime** sets the PWM pulse width in nanoseconds.

## **PWM Example Program**

```
fd = libsimpleio.pwm_open(0, 0, 10000000, 0)
```

```
while true
  for ontime = 0 to 10000000 step 10000
    libsimpleio.pwm_write(fd, ontime)
    delay(5000)
  next ontime
wend
```