I2C Library

WiringPi includes a library which can make it easier to use the Raspberry Pi's on-board I2C interface.

Not all systems have the I2C development libraries and headers installed, and when you build ***wiringPi***, it detects this at build time. If you are using these helper functions and you get link errors, it means that ***wiringPi*** is not installed with the I2C helper functions. You need to install the I2C development libraries and re-build.

Under Raspbian:

```
sudo apt-get install libi2c-dev
```

then rebuild ***wiringPi***.

Before you can use the I2C, you need to load the kernel modules and you can use the **gpio** utility to load the I2C drivers into the kernel:

```
gpio load i2c
```

If you need a baud rate other than the default 100Kbps, then you can supply this on the command-line:

```
gpio load i2c 400
```

will set the baud rate to 400Kbps – ie. 400,000 bps. (K here is times 1000)

To use the I2C library, you need to:

```
#include <wiringPiI2C.h>
```

in your program. Programs need to be linked with **-lwiringPi** as usual.

You can still use the standard system commands to check the I2C devices, and I recommend you do so – e.g. the **i2cdetect** program. Just remember that on a Rev 1 Raspberry pi it's device 0, and on a Rev. 2 it's device 1. e.g.

```
i2cdetect -y 0 # Rev 1
i2cdetect -y 1 # Rev 2
```

Functions available

- **int wiringPiI2CSetup (int devId) ;**

This initialises the I2C system with your given device identifier. The ID is the I2C number of the device and you can use the **i2cdetect** program to find this out. wiringPiI2CSetup() will work out which revision Raspberry Pi you have and open the appropriate device in /dev.

The return value is the standard Linux filehandle, or -1 if any error – in which case, you can consult *errno* as usual.

E.g. the popular MCP23017 GPIO expander is usually device Id 0x20, so this is the number you would pass into wiringPiI2CSetup().

For all the following functions, if the return value is negative then an error has happened and you should consult *errno*.

- **int wiringPiI2CRead (int fd) ;**

Simple device read. Some devices present data when you read them without having to do any register transactions.

- **int wiringPiI2CWrite (int fd, int data) ;**

Simple device write. Some devices accept data this way without needing to access any internal registers.

- **int wiringPiI2CWriteReg8 (int fd, int reg, int data) ;**
- **int wiringPiI2CWriteReg16 (int fd, int reg, int data) ;**

These write an 8 or 16-bit data value into the device register indicated.

- **int wiringPiI2CReadReg8 (int fd, int reg) ;**
- **int wiringPiI2CReadReg16 (int fd, int reg) ;**

These read an 8 or 16-bit value from the device register indicated.