Wiring Pi

GPIO Interface library for the Raspberry Pi

Setup

There are four ways to initialise ***wiringPi***.

- **int wiringPiSetup (void) ;**
- **int wiringPiSetupGpio (void) ;**
- **int wiringPiSetupPhys (void) ;**
- **int wiringPiSetupSys (void) ;**

One of the ***setup*** functions **must** be called at the start of your program or your program will fail to work correctly. You may experience symptoms from it simply not working to segfaults and timing issues.

> **Note**: ***wiringPi*** version 1 returned an error code if these functions failed for whatever reason. Version 2 returns always returns zero. After discussions and inspection of many programs written by users of ***wiringPi*** and observing that many people don't bother checking the return code, I took the stance that should one of the ***wiringPi*** setup functions fail, then it would be considered a fatal program fault and the program execution will be terminated at that point with an error message printed on the terminal.
>
> - *If you want to restore the v1 behaviour, then you need to set the environment variable:* ***WIRINGPI_CODES*** *(to any value, it just needs to exist)*

The differences between the setup functions are as follows:

- **wiringPiSetup (void) ;**

  This initialises ***wiringPi*** and assumes that the calling program is going to be using the ***wiringPi*** pin numbering scheme. This is a simplified numbering scheme which provides a mapping from virtual pin numbers 0 through 16 to the real underlying Broadcom GPIO pin numbers. See the pins page for a table which maps the ***wiringPi*** pin number to the Broadcom GPIO pin number to the physical location on the edge connector.

  This function needs to be called with root privileges.

- **wiringPiSetupGpio (void) ;**

  This is identical to above, however it allows the calling programs to use the Broadcom GPIO pin numbers directly with no re-mapping.

  As above, this function needs to be called with root privileges, and note that some pins are different from revision 1 to revision 2 boards.

- **wiringPiSetupPhys (void) ;**

  Identical to above, however it allows the calling programs to use the physical pin numbers *on the P1 connector only*.

  As above, this function needs to be called with root priviliges.

- **wiringPiSetupSys (void) ;**

  This initialises ***wiringPi*** but uses the */sys/class/gpio* interface rather than accessing the hardware directly. This can be called as a non-root user provided the GPIO pins have been exported before-hand using the **gpio** program. Pin numbering in this mode is the native Broadcom GPIO numbers – the same as wiringPiSetupGpio() above, so be aware of the differences between Rev 1 and Rev 2 boards.

  Note: In this mode you can only use the pins which have been exported via the /sys/class/gpio interface before you run your program. You can do this in a separate shell-script, or by using the system() function from inside your program to call the **gpio** program.

  Also note that some functions have no effect when using this mode as they're not currently possible to action unless called with root privileges. (although you can use system() to call **gpio** to set/change modes if needed)