Wiring Pi

GPIO Interface library for the Raspberry Pi

Timing

While Linux provides a multitude of system calls and functions to providing various timing and sleeping functions, sometimes it can be quite confusing, especially if you are new to Linux, so the ones presented here mimic those available on the Arduino platform, making porting code and writing new code somewhat easier.

Note: Even if you are not using any of the input/output functions you still need to call one of the ***wiringPi*** setup functions – just use **wiringPiSetupSys()** if you don't need root access in your program and remember to #include <wiringPi.h>

- **unsigned int millis (void)**

  This returns a number representing the number of milliseconds since your program called one of the ***wiringPiSetup*** functions. It returns an unsigned 32-bit number which wraps after 49 days.

- **unsigned int micros (void)**

  This returns a number representing the number of microseconds since your program called one of the ***wiringPiSetup*** functions. It returns an unsigned 32-bit number which wraps after approximately 71 minutes.

- **void delay (unsigned int howLong)**

  This causes program execution to pause for at least ***howLong*** milliseconds. Due to the multi-tasking nature of Linux it could be longer. Note that the maximum delay is an unsigned 32-bit integer or approximately 49 days.

- **void delayMicroseconds (unsigned int howLong)**

  This causes program execution to pause for at least ***howLong*** microseconds. Due to the multi-tasking nature of Linux it could be longer. Note that the maximum delay is an unsigned 32-bit integer microseconds or approximately 71 minutes.

  Delays under 100 microseconds are timed using a hard-coded loop continually polling the system time, Delays over 100 microseconds are done using the system nanosleep() function – You may need to consider the implications of very short delays on the overall performance of the system, especially if using threads.