Wiring Pi

GPIO Interface library for the Raspberry Pi



Software Tone Library

WiringPi includes a software-driven sound handler capable of outputting a simple tone/square wave signal on any of the Raspberry Pi's GPIO pins.

There are some limitations... To maintain a low CPU usage, the minimum pulse width is 100μ S. That gives a maximum frequency of 1/0.0002 = 5000Hz.

Also note that while the routines run themselves at a higher and real-time priority, Linux can still affect the accuracy of the generated tone.

However, within these limitations, simple tones on a high impedance speaker or piezo sounder is possible.

To use:

```
#include <wiringPi.h>
#include <softTone.h>
```

When compiling your program you must include the pthread library as well as the wiringPi library:

```
cc -o myprog myprog.c -lwiringPi -lpthread
```

You must initialise **wiringPi** with one of wiringPiSetup(), wiringPiSetupGpio() or wiringPiSetupPhys() functions. wiringPiSetupSys() is not fast enough, so you must run your programs with sudo.

Some expansion modules may also be fast enough to handle software PWM – it has been tested with the MCP23S17 GPIO expander on the PiFace for example.

The following two functions are available:

int softToneCreate (int pin);

This creates a software controlled tone pin. You can use any GPIO pin and the pin numbering will be that of the *wiringPiSetup*() function you used.

The return value is 0 for success. Anything else and you should check the global *ermo* variable to see what went wrong.

void softToneWrite (int pin, int freq);

This updates the tone frequency value on the given pin. The tone will be played until you set the frequency to 0.

Notes

- Each pin activated in softTone mode uses approximately 0.5% of the CPU.
- You need to keep your program running to maintain the sound output!