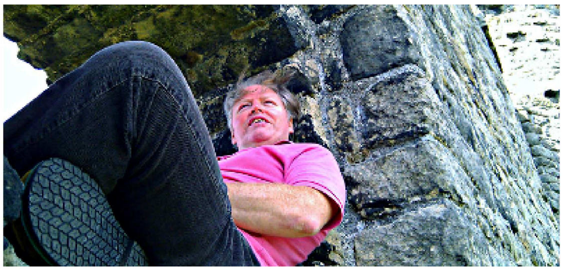
Gordons Projects

Projects, Fun and Games from Gordon @ Drogon



Download and Install

WiringPi is now maintained under GIT for ease of change tracking, however there is a *Plan B* if you're unable to use GIT for whatever reasons (usually your firewall will be blocking you, so do check that first!)

If you do not have GIT installed, then under any of the Debian releases (e.g. Raspbian), you can install it with:

```
sudo apt-get install git-core
```

If you get any errors here, make sure your Pi is up to date with the latest versions of Raspbian:

```
sudo apt-get update
sudo apt-get upgrade
```

To obtain WiringPi using GIT:

```
git clone git://git.drogon.net/wiringPi
```

If you have already used the clone operation for the first time, then

```
cd wiringPi
git pull origin
```

Will fetch an updated version then you can re-run the build script below.

To build/install there is a new simplified script:

```
cd wiringPi
./build
```

The new build script will compile and install it all for you – it does use the sudo command at one point, so you may wish to inspect the script before running it.

Plan B

Click on this URL: (it should open in a new page)

https://git.drogon.net/?p=wiringPi;a=summary

Then look for the link marked **snapshot** at the right-hand side. You want to click on the top one.

This will download a tar.gz file with a name *like* wiringPi-98bcb20.tar.gz. Note that the numbers and letters after **wiringPi** (98bcb20 in this case) will probably be different – they're a unique identifier for each release.

You then need to do this to install:

```
tar xfz wiringPi-98bcb20.tar.gz
cd wiringPi-98bcb20
./build
```

Remmeber the actual filename will be different – you will have to check the name and adjust accordingly.

Test wiringPi's installation

run the gpio command to check the installation:

```
gpio -v
gpio readall
```

That should give you some confidence that it's working OK.

WiringPi is released under the GNU Lesser Public License version 3.

Comments

Download and Install — 321 Comments

Tobi
on May 23, 2012 at 10:40 am said:

Hi..
The PWM-Test (test2) with a LED between Pin12 and Pin3 (GND) produces weird noises on the sound output jack. Can I solve this somehow ?

Gordon
on May 23, 2012 at 11:02 am said:

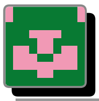
No, sorry!

The Pi only has 2 PWM output pins and they're normally used for the audio output on the 3.5mm jack socket. If you re-program the GPIO pin for PWM, it then uses one of the 2 PWM channels that goes to the audio output – so when you use it to drive an LED or motor, etc. it will make some sort of noise on the audio…

-Gordon

Carter
on September 26, 2013 at 12:00 am said:

Ya, this is an unfortunate bit about the PI, BUT, I handled off loading PWM to a I2C board to work around this limitation. Pardon the outright ad, and there are several out there, but Adafruit's http://www.adafruit.com/products/815
worked for me.



Jacques
on June 18, 2012 at 12:05 pm said:

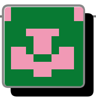
Thank you for your work done with this project. I can now run test2 successfully from command line only. I have a rough idea how to write and compile programs with text editors. (I've used Geany as my editor.)

I am teaching my 13 year old daughter with Geany on the Raspi to write basic C++ programs. (I never used Linux for C/C++ before yesterday.) How do I include these libraries into g++ so that I can use Geany to write and test the code? It will be great if we can "do stuff" via GPIO as well. I don't want to drag out the Arduino as well.

Regards

Gordon
on June 18, 2012 at 10:46 pm said:

I really don't know much about Geany and its environment. All my work is done using a text editor like Vi (or vim), and Makefiles…
I'll have a look at it if I have some time though.
-Gordon



Jacques
on June 19, 2012 at 11:09 am said:

Thank you for both replies to my questions. I still have a lot to learn, at least I can write with Geany, its just the compiling bit that's a bummer. I think my question is more general in any case. Basically, how to get extra libraries and headers into C++ with linux. So far I did not had much luck figuring it out.
Regards and thank you once a again for helping me make my own LED pattern go blink blink blink!

Gordon
on June 20, 2012 at 3:54 pm said:

What you can do is simply copy the source-code in-line with your own program. Just add it all on at the top of the program. It's not elegant, but it should work!

I've not had time to look at Geany yet though.

-Gordon

arlukas
on August 21, 2012 at 8:48 pm said:

Hello Jaques
Like you I use Geany to program, compile and debug C programs and also I had the same problem with wiring library. Here is my solution.
Go to ‘Build’ menu and then ‘Set Build Commands’. In the second option, ‘Build’, you have to write the following:

```
gcc -o “%e” -I/usr/local/include “%f” -L/usr/local/lib -lwiringPi
```

I assume that you have followed the standar installation instructions.

Hope this helps.

Regards.

Ralph McArdell
on June 27, 2012 at 2:53 pm said:

Hello Gordon,

I have just downloaded and perused your WiringPi code.

Looks very interesting. Thanks for making it available as it shows many useful specifics for using the Raspberry Pi’s GPIO.

One thing that I noticed was that in the definition of the pinMode function in wiringPi.c the static flag pwmRunning seems only to be defined and initialised to FALSE and tested and never set to anything else. Should it possibly be set to TRUE somewhere, maybe at the end of the if (!pwmRunning) block? Or have I missed the point?

Regards

Ralph

Gordon
on June 27, 2012 at 3:36 pm said:

Looks like you’ve not missed the point, but I’ve missed a line! There should be a pwmRunning = TRUE in that block. However it’s not really doing any harm, just a minor optimisation and I’ll sort it out then next time I do an update!

Thanks!

-Gordon

David
on July 5, 2012 at 2:20 pm said:

Could this code be used on my Raspberri Pi? I know it runs on an Arduino to control leds on a reef tank. Thanks

```
byte MoonPhase()
{
```

```
int m,d,y;
int yy,mm;
long K1,K2,K3,J,V;
byte PWMvalue;
m = month();
d = day();
y = year();
yy = y-((12-m)/10);
mm = m+9;
if (mm>=12) mm -= 12;
K1 = 365.25*(yy+4712);
K2 = 30.6*mm+.5;
K3 = int(int((yy/100)+49)*.75)-38;
J = K1+K2+d+59-K3;
V = (J-2451550.1)/0.29530588853;
V -= int(V/100)*100;
V = abs(V-50);
PWMvalue = 4*abs(50-V); // 5.12=100% 4=~80%
//pinMode(lowATOPin,OUTPUT);
return (PWMvalue*100)/255;
}
```

It returns 0-100.
You can comment the kast line and return PWMvalue for 0-255.

Gordon
on July 5, 2012 at 2:26 pm said:

Essentially yes.

You may have to tweak a few things though – e.g. there might not be a “byte” type (and even on the arduino, you probably want to #include <stdint.h> and use uint8_t instead of ‘byte’…)
The PWM value is 0 to 1023, so might need a little scaling.
ints on the Pi are signed 32-bit values, as are longs, so do check ranges and so on.

Not sure where you’re getting the day(), month() and year() functions from either, but I’ll assume you have something else to provide them… If not, and you’re not sure how to do this stuff under unix then lookup the manual page for the ctime() function.

-Gordon

David
on July 7, 2012 at 1:23 am said:

I tried to cc the code and got this error.
It also did not like the stdint.h

```
lunar.c:1:1: error: unknown type name ‘unit8_t’
lunar.c: In function ‘MoonPhase’:
lunar.c:7:1: error: unknown type name ‘unit8_t’
lunar.c:16:6: error: expected expression before ‘int’
lunar.c:19:6: error: expected expression before ‘int’
```

Gordon
on July 7, 2012 at 8:14 am said:

uint8_t is defined in stdint.h – it’s surprising if your system doesn’t have stdint.h though.

```
#include <stdint.h>
```

?

-Gordon

HBrydon
on January 9, 2013 at 8:27 pm said:

This is a response to a really old thread but I think the coding error is because you typed in ‘unit8_t’ and it should be ‘uint8_t’.

Gordon
on January 9, 2013 at 8:35 pm said:

Ah… I’m a shade dyslexic and often don’t see things like that myself…

-Gordon

James
on February 22, 2013 at 6:32 am said:

Sometimes I’m dylsexic too!

Simon
on July 7, 2012 at 12:36 pm said:

Gordon – have recently downloaded this update with added serial commands. I’m very new to all this and wondered if you could help? I’ve written some midi code for the Arduino (which all works fine)based on the Arduino midi example:

```
void setup() {
// Set MIDI baud rate:
Serial.begin(31250);
}

void loop() {
// play notes from F#-0 (0x1E) to F#-5 (0x5A):
for (int note = 0x1E; note < 0x5A; note ++) {
//Note on channel 1 (0x90), some note value (note), middle velocity (0x45):
noteOn(0x90, note, 0x45);
delay(100);
//Note on channel 1 (0x90), some note value (note), silent velocity (0x00):
noteOn(0x90, note, 0x00);
delay(100);
}
}

// plays a MIDI note. Doesn't check to see that
// cmd is greater than 127, or that data values are less than 127:
void noteOn(int cmd, int pitch, int velocity) {
Serial.write(cmd);
Serial.write(pitch);
Serial.write(velocity);
}
```

I've tried to adapt this using your serial library but I just can't work it out – if you, or someone else who reads this, wouldn't mind could you adapt the above code so it will run on raspberry pi – once I have that I think I can work on it from there.

Thanks for your help

Gordon
on July 7, 2012 at 12:51 pm said:

ok. A very quick translation:

Serial.begin (baud) becomes fd = serialOpen ("/dev/ttyAMA0", 31250) ;

However sadly this will fail as that's not a recognised baud rate in the Linux world.

So if you can change the baud rates then it'll work, otherwise I don't know where to start, but a quick google suggests this may help you: http://joost.damad.be/2009/05/how-to-set-serial-port-at-midi-speed-in.html

in your noteOn() function:

Serial.write (val) becomes serialPutchar (fd, val) ;

And that's it at it's simplest. the 'fd' value (and int) is the file descriptor (everything is a file in unix land). You just need to make that avalable to any code that wants to use the serial port. (maybe a global for you). The /dev/ttyAMA0 is the on-board serial port (3.3v, remember). However it could be a USB serial port – e.g. /dev/ttyUSB0 for example…

-Gordon

wally
on August 23, 2012 at 7:12 pm said:

Hello Gordon,

"However it could be a USB serial port – e.g. /dev/ttyUSB0 for example…"

With device "/dev/ttyAMA0″ and a MAX3232 it works nice. Your second suggestion sending serial data to USB sounds interesting. Anyway, i do not have a /dev/ttyUSB0" in my /dev/ folder. May you explain how to do this ?

thanks wally

Gordon
on August 23, 2012 at 7:20 pm said:

You'll only see a /dev/ttyUSB0 if you plug in a USB serial device.
The Pi's on-board serial port is /dev/ttyAMA0.

In Linux all serial ports work more or less the same – they just have different names – /dev/ttyAMA0 for the on-board on one a Pi, /dev/ttyS0 for the on-board one on a typical PC, /dev/ttyUSB0 for usb serial devices (ftdi), and /dev/ttyACM0 for some other modem devices (sometimes 3G modems)

-Gordon

wally
on August 23, 2012 at 8:02 pm said:

"You'll only see a /dev/ttyUSB0 if you plug in a USB serial device."

sure, i'm donkey 🙂

thank you

Gordon
on August 23, 2012 at 8:05 pm said:

Nah, but it’s easy to get confused if you’re new to all this!

-Gordon

j gearing
on July 13, 2012 at 10:29 am said:

I seem to be stuck already. Am I ok doing this in lx terminal or should i be elsewhere, the files are in the examples folder

```
pi@raspberrypi:/tmp/wiringPi/examples$ sudo ./test1
sudo: ./test1: command not found
pi@raspberrypi:/tmp/wiringPi/examples$ sudo ./test2
sudo: ./test2: command not found
```

thanks

JG

Gordon
on July 13, 2012 at 2:56 pm said:

lxterm is fine.
did you do the sequence of:
cd wiringPi/wiringPi
make
sudo make install
cd ../ gpio
make
sudo make install
— and finally:
cd ../examples
make <--- did you type make? sudo ./test1 I'm wondering if you missed out the 'make' command in the examples directory. However, unless you connect up some LEDs to the Pi, you're not going to see very much happening - they are really intended to be used as examples to base your own GPIO programs on. -Gordon

j gearing
on July 13, 2012 at 1:18 pm said:

sorry. missed out a line
thanks

JG

Gordon
on July 13, 2012 at 2:57 pm said:

Ah, guesed you missed the final ‘make’ them. Do look at the programs though and if you want more examples, there are a few more on my website – e.g. lookup the TuxX program and the ladders game …

-Gordon

Duncan
on July 20, 2012 at 12:57 pm said:

Have just tried to install following your guidelines but ran into a problem:

```
pi@raspberrypi /tmp $ tar xfz wiringPi.tgz
pi@raspberrypi /tmp $ cd wiringPi/wiringPi
pi@raspberrypi /tmp/wiringPi/wiringPi $ make
[CC] wiringPi.c
[CC] serial.c
[CC] wiringShift.c
[CC] lcd.c
[CC] piHiPri.c
[CC] piThread.c
[AR] wiringPi.o serial.o wiringShift.o lcd.o piHiPri.o piThread.o
make: avr-ar: Command not found
make: *** [libwiringPi.a] Error 127
pi@raspberrypi /tmp/wiringPi/wiringPi $
```

Got round the problem by using a Makefile from a download yesterday. The fix was:

```
$(TARGET): $(OBJ)
@echo [AR] $(OBJ)
@ar rcs $(TARGET) $(OBJ)
@size $(TARGET)
```

Gordon
on July 20, 2012 at 2:20 pm said:

Aargh)-:
bother.
That’s what I get for working on some ATmega and Pi projects at the same time. I’ve put the wrong thing in the wrong makefile – (and it happens I have the arduino IDE installed on my Pi’s so I never noticed it).
Thanks for that. I’ll fix it right away.

-Gordon

Stew
on July 20, 2012 at 1:31 pm said:

I get an error when trying to ‘make’:
http://i45.photobucket.com/albums/f87/Stew2000/RaspberryPi/DSCF0383-2.jpg

Gordon
on July 20, 2012 at 2:20 pm said:

Yes. I’ve goofed on the Makefile. Will fix it right away.
-Gordon

Quentin
on July 20, 2012 at 5:59 pm said:

Hi Gordon and thank you for your library. Is it possible to stop the program after I started it?

Gordon
on July 20, 2012 at 6:02 pm said:

Type Control-C in the terminal window you started the program in.
(That’s the usual unix/linux way!)
-Gordon

Mike
on July 26, 2012 at 4:33 am said:

I just wanted to let you know that I have prepared and inserted this into the Arch Linux ARM repos.

Gordon
on July 26, 2012 at 7:54 am said:

Thanks!

What I’m aiming to do is to move it all to GIT, but that requires learning GIT first – I just need to sit down for an afternoon with no interruptions and get on with it..

-Gordon

Dan Riches
on July 29, 2012 at 3:24 pm said:

Hi Gordon

I’ve downloaded and run the install for wiringPi and this completed all ok. Trouble is I can’t seem to compile my own test app, what I did is this:

```
#include
#include
#include
#include
int main()
{
if(wiringPiSetup() == -1)
exit(1);
else
printf(“wiringPi library ok!”);

return(0);
}
```