In this tutorial I will show you how to connect some motors to your Raspberry Pi. Doing so will allow your Raspberry Pi to interact in the real world, making it possible to build a robot, turn on a fan on a hot day or even drop a treat for your cat or dog while your away.

**Objective**

What we plan to do is safely connect one or two motors to the Raspberry Pi with as few components as possible. Once we have the electronics put together on the breadboard, I will show you how to control them easily using Python to first make the motor spin, and then add some control to change the motor direction so we can go backwards.

This guide will require a careful eye to catch any mistakes, and a bit of courage, especially if you are new to the GPIO connectors. I would like to stress I am not responsible for any damage caused to your Raspberry Pi and/or components.
Q. IMPORTANT: Do not connect a motor, no matter how small directly to the Raspberry Pi, it will damage your Raspberry Pi.

The main processor can only supply enough power to light a LED, roughly 20mA. A motor will want at least 400mA of current to start turning.

Requirements

To get a motor running, you will need:

- A Raspberry Pi with SD card preinstalled with Raspbian
- A Breadboard to connect everything on
- An L293 or SN755410 motor driver chip (I will refer both as L293D in this tutorial)
- Jumper cables to connect everything up (Male to male and female to male)
- One or two DC motors rated for 6v
- 4x AA batteries and holder

GPIO pins

If you haven’t looked closely at your Raspberry Pi before, now might be the best time to have a good look. There are 26 pins grouped in two rows of 13, and these collectively are called the General Purpose Input Output header or GPIO for short. These are a mix of four power pins, five ground pins and 17 data pins.
Figure 1. The Layout of the GPIOs with the pin numbering. Pin 1 is the top left labelled 3V3

Some of these data pins have extra functions such as an i2c bus, SPI bus and UART serial connectors, all of which can connect to other hardware to allow the Raspberry Pi to talk to items such as an Arduino, an Analogue to Digital Convertor (ADC) or add-on boards such as a PiGlow or PiFace.

TIP: When working with the GPIO pins, always do this while the Pi is unplugged, as any accident by connecting (or shorting) 2 pins together can cause damage to the Raspberry Pi.

Assembling the Circuit

Adding Power and Ground

It is important to do this while the power to the Raspberry Pi is off, or disconnected, as you want to avoid shorting any connectors by mistake.

The first thing you need to do is connect up the power and ground wires. As with most electronics projects, everything that connects together will require a common ground. This is shown with the black wires.
The ground on the Raspberry Pi is physical **pin 6**. Referring to Figure one this is worked out by starting at the top left with **pin 3V3**, counting left to right so **5V** is **pin 2**, **GPIO 2** (labelled 2) is **pin 3** and so on.

Reading pin numbers on Integrated Circuit (IC) chips is easily done by having the notch or dot to the left then starting from bottom left gives us **pin 1**.

**Adding the Data Wires**

Now add three wires from the **GPIO** pins to the **L293D**.
- GPIO 25–Pin 22 > L293D–Pin 1
- GPIO 24–Pin 18 > L293D–Pin 2
- GPIO 23–Pin 16 > L293D–Pin 7

Figure 4. Add the three GPIO wires to control the motor

Add the motor:

- Motor–wire 1 > L293D–pin 3
- Motor–wire 2 > L293D–pin 6
It is extremely important that you double-check every connection before adding the batteries. Only when you are happy that everything is in place, connect the battery wires to the power rails of the breadboard.

**Add a Second Motor (Optional)**

One of the great features of the **L293D** is it that it can handle two motors independently and each motor can run at different speeds or directions. Using this one IC makes it possible to create a two-wheeled robot capable of turning, going forwards and going backwards easily.

Adding a second motor involves just three additional wires and another motor.

- GPIO 11–Pin 23 > L293D–Pin 9
- GPIO 9–Pin 21 > L293D–Pin 10
- GPIO 10–Pin 19 > L293D–Pin 15
Powering On

It is important to check and double-check any wiring before adding any power source to your project as some of the wiring can get a bit fiddly it is easy to miss a connection and send 5V in to the 3.3V of the Raspberry Pi.

Always check your wiring and then check it again!

If you haven’t set up an SD card for your Pi before, it will be worth understanding how to create one by reading the How to Flash an SD Card for Raspberry Pi tutorial, first.

With a freshly created Raspbian SD card in place connect the Raspberry Pi as usual and boot up.

Add the batteries, ensuring that you pay attention to the correct rails (the long strips along the top and bottom, if you have any), as you only want the black wire to connect to the ground, and the red wire to positive or source of the chip only.
The next job is to tell the Raspberry Pi that a motor, or two, has been connected. To do this I am using a language called Python. It comes installed on Raspbian which is a bonus. If you are using another Operating System such as Arch or PiDora, double-check if RPi.GPIO is available.

Testing

In order to get the motors to work double-click LXTerminal on your desktop to bring up a terminal window. This is where you will be writing Python code using a program called Nano. Nano is a text editor, similar to Notepad or TextEdit but for the command prompt, I will teach your some commands as we go along if you are new to it.

To turn the motor on for two seconds use the following code:

```python
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
Motor1A = 16
Motor1B = 18
Motor1E = 22
GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)
print "Turning motor on"
GPIO.output(Motor1A,GPIO.HIGH)
GPIO.output(Motor1B,GPIO.LOW)
GPIO.output(Motor1E,GPIO.HIGH)
sleep(2)
print "Stopping motor"
GPIO.output(Motor1E,GPIO.LOW)
GPIO.cleanup()
```

The first two lines tell Python what is needed in the program.

The first line will want to access a module called RPi.GPIO. This module handles all the hard work involved around turning the GPIO pins on and off on the Raspberry Pi.
The second line brings in `sleep` from the module `time` to make it possible to pause the script giving it time to perform a certain action, in this case leaving a motor on for a few seconds.

The function `setmode` tells `RPi.GPIO` to use the board numbering on the Raspberry Pi. The numbers 16, 18 and 22 we will use to tell Python they are the pins associated with the motors.

When using the L293D you can give it a direction, by turning one side on to turn in one direction, called `pin A` and vice versa is `pin B`. To turn the motor on use a pin called `Enable`, labelled `E` in the test script—this is `pin 22`. I’ll cover this a bit more later.

Finally, tell the Raspberry Pi these are all outputs which is done with `GPIO.OUT`.

With the script set up, the Raspberry Pi ready to turn the motors. It will turn on some pins, wait two seconds then turn them off again, shown in the remainder of the script.

Save and exit by pressing `CTRL-X`, along the bottom a message asks you to confirm the changes. Press `Y` and `Enter` to confirm. Now you are back at the command prompt to run the script and see the motor spin to life.

```
sudo python motor.py
```

If the motor didn’t turn, double check your wiring or batteries. Debugging and finding out why something doesn’t work can be annoying, but is a useful step in learning something new!
Now Go Backwards

It's brilliant to have a motor spin, but even better to make it turn backwards, so I'll show you how to do that.

Nothing needs to be done to the wiring, this is purely Python now. This is achieved by creating a new script, calling it `motorback.py`. To create the script in Nano, enter the command:

```
nano motorback.py
```

Enter the following code:

```python
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BOARD)
Motor1A = 16
Motor1B = 18
Motor1E = 22

GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)

print "Going forwards"
GPIO.output(Motor1A,GPIO.HIGH)
GPIO.output(Motor1B,GPIO.LOW)
GPIO.output(Motor1E,GPIO.HIGH)
sleep(2)

print "Going backwards"
GPIO.output(Motor1A,GPIO.LOW)
GPIO.output(Motor1B,GPIO.HIGH)
GPIO.output(Motor1E,GPIO.HIGH)
sleep(2)

print "Now stop"
GPIO.output(Motor1E,GPIO.LOW)
```

CTRL-X then Y followed by Enter to save.

The script is very similar to the previous one, but if you notice for backwards we made Motor1A low and Motor1B high.
High and low are programming names for on and off.

To stop the motor you'll turn off, low, Motor1E.

Enable is the switch to turn the motor on and off, regardless of what A and B are doing.

If you are finding this confusing, look at a Truth Table to see what is happening.

<table>
<thead>
<tr>
<th>Enable</th>
<th>A</th>
<th>B</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>Not spinning - Enable is off</td>
</tr>
<tr>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>Not spinning - Enable is off</td>
</tr>
<tr>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Not spinning - Both inputs are off</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Turning clockwise*</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Turning counter-clockwise*</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Not spinning - Both inputs are on</td>
</tr>
</tbody>
</table>

* Direction is down to the motor

There are only two states which allow the motor to turn, when Enable is on or high, and either A or B must be high, but not both.

For two motors look at the following script. All that is different is a couple more lines to set up the second motor.

```python
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
Motor1A = 16
Motor1B = 18
Motor1E = 22
Motor2A = 23
Motor2B = 21
Motor2E = 19
GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)
GPIO.setup(Motor2A,GPIO.OUT)
GPIO.setup(Motor2B,GPIO.OUT)
GPIO.setup(Motor2E,GPIO.OUT)
print "Going forwards"
GPIO.output(Motor1A,GPIO.HIGH)
```
Conclusion

In this tutorial I have shown you the basics of connecting motors to your Raspberry Pi. It may take a deep breath and can-do attitude if you are new to connecting anything to your brand new Pi, but you will soon find that once you start playing with the GPIO pins that it is hard to stop.

This tutorial opens the doors to making anything like robots with blinking LED lights and ultrasonic sensors in order to sense its environment.

Find a chassis to mount everything on use a USB mobile phone charger battery to make your Raspberry Pi fully mobile.
Tyler Fraser • a year ago

Your enable is wrong for motor 2 you have:

Motor2A = 23
Motor2B = 21
Motor2E = 19

It should be
Motor2A = 19
Motor2B = 21
Motor2E = 23

This solves the only one motor going reverse when both should be.

Refer to this image:

![L293D diagram](image)

see more

Ewan Gallacher • Tyler Fraser • 4 months ago

Thanks

Nir Geller • a year ago

Soo even if we connect to batteries we still need to connect the raspberry pi to another power source? And can we achieve the same connections without a breadboard? Because want to build like a little car with 2 motors and a raspberry pi, but I don't think I'll be able to put a bread board on the car...

Sorry for the stupid questions I am really new to this

Nir Geller • Tyler Fraser • a year ago

Yes, as long as you run the raspberry and the motors on 5 Volts.

Dev Synopass • 5 months ago

Thanks a lot for the great tutorial, I followed it and successfully build my first Pi car with four wheels. Thanks!

Qureeb Hameed • 9 months ago

RPi.GPIO.SetupException: No access to /dev/mem.

Mike Skovgaard • Qureeb Hameed • 9 months ago

GPIO access is a privileged operation. So you need to run your script with...

GPIO access is a privileged operation. So you need to run your script with `sudo`. I.e.: `sudo python motorback.py`

batmobil ➤ Mike Skovgaard · 7 months ago
This is the kind of critical info you really need to add in a basic tutorial.

Arturs Lataks · a year ago
Can I use 3V rated FK-180SH-3240 motors?

Diogo Caneco · 2 months ago
I can't make it work.

Naughty Guy · 3 months ago
Hello Guys. I want to control speed of a DC motor using PWM can you please help me with the python code

ali · 4 months ago
Thanks for the instruction, i made my own robot then working on voice recognition on it... ;)

Efias Cabrera · 6 months ago
Serve the L293B model ?. I have not the model L293D

Calliah · 6 months ago
Question - if the four ground pins (4, 5, 12, 13) on the chip are all connected (at least based on my multimeter), can I just hook only one of the four pins to the common ground rail that will be used from the pi and power pack for the motors? Is it safe to do this?

Prathamesh · 8 months ago
Thank you for this useful information. I am working on a project named Robotic Arm control using Raspberry pi. For that I am using Apache2 web server. I want to control 5 motors through Web server so my question is, Can I Connect pin 16 and 18 of Raspberry Pi to pin 2 & 7 and at 10 & 15 both? or to All three IC inputs?

Anika Rahman · 8 months ago
This stuff is great! It's just what I was looking for. Is it possible to put the Pi to sleep for a few hours instead of just a few seconds? I am making a pill dispenser for my project and plan to set up the Pi to sleep for hours and to dispense the medicines at set times. I would be grateful if someone could point me in the right direction please. Thanks in advance!
Alexmrb · 9 months ago
I know you are getting asked lots of questions, but would it be possible to control two DC motors with an L293D controller board built for the Arduino Mega? Just wondering because it would be pretty cool if I could snag a controller on eBay for 2 bucks and play around with some motors if that’s the case.

Reply · Share ·

JOASH Meachak · 10 months ago
Can I Use a different breadboard??

Reply · Share ·

#letsFootball · a year ago
Hi, This information was worth reading and I was able to do as much as you intended in this article. But I have a question.
Can we control the DC motor speed with this setup? Like increase/decrease the rpm.
Any information on this will be helpful

Thanks in advance.
~Arpith

Reply · Share ·

Cody Cryderman → #letsFootball · a year ago
You would use Pulse Width Modulation to control speed or PWM. Google it

Reply · Share ·

Geek Projex → Cody Cryderman · 9 months ago
the pain is... Pi only has one PWM source...gpio 18.

I currently have a tracked vehicle (2 wheel drive) on two small electric motors (hobby style/toy) Im using pwm the real limitation is that you cannot stop only one of the motors, in my circuit both enable from pin 18 (pwm0) so either both on or both off, but independent direction.
The speed is also the same (only one pwm source). I can only see one way around this limitation which would be to use 2 more pins to bias diodes between pin 18’s connections to each of the IC’s enable pins, giving us software control over the enabling of either. Then we could stop one and run the other at any speed any direction, BUT if both are going, they will always be at the same speed.

I think.

Trying this is my next step if both on or both off doesn’t work out for maneuvering. Please correct my understanding of pwm as you see fit, as my own is fairly minimal....I think...

Reply · Share ·

disqus_hzfCaFC4sQ → Geek Projex · 5 months ago
Why dont you just write a code to imitate pwm? that is what I have succesfully done in the past
Geek Projex • Geek Projex • 9 months ago
I just looked at what the pins do again in different combos and now feel dumb for thinking I couldn't stop them. Back to the pin code... I think I was wrong. pwm 4eva! yay! I think...

Mattpx • a year ago
Hello, I am Matt.
Thank you very much it was very useful! I am trying to add a push switch to the circuit following this example [http://razpisampler.oreilly.com/](http://razpisampler.oreilly.com/) but I wonder where should I place the switch with this breadboard.

Thank you for your help!

see more

John Green • a year ago
Is it possible to use motors with more v?

Alex • a year ago
Hi, thanks for the tutorial. One problem I have is that no matter what I do the second motor will not go in reverse. I have tried wiring it 4 times now! Any help appreciated.

Alex

Khuong • a year ago
Hi, I'm also doing this project and have had problem with the Pi fuse blown TWICE. I have set up as shown above for the 1 motor case with a battery supply of 7.2V .8A for the motor. I made sure that the ground on the pi and the battery are connected as instructed. The motor runs as expected but suddenly shutdown and can't boot up. I have looked around and confirm that the fuse was blown (the voltage drop was >0.3 rather that the standard .1V), I have tried on two different pis and the same thing happens. Please let me know if you have any advice on this issue. It will be much appreciated. Thanks. Please comment or email me at khuongnguyensac@gmail.com.
Edward C. Charbonnet • a year ago

Thanks for such a useful article. All worked as described. Do you know a way to enhance this to control the motor speed? I understand that would involve some control over pulse width.

Carlos Luna • Edward C. Charbonnet • a year ago

Hi Edward.

I thought I might go ahead and answer your question if you’re still interested.

I found a way to control the speed using PWM via the RPi.GPIO library Jason uses in his python examples.

Using the last code he has on this page (I can’t figure out how to add code to this response), add the following lines but replace the "<tab>" with actual tabs, otherwise Python will be angry.

At line 13, add:

```
# we set the PWM controller to Motor1A for forward movement with 250 Hz frequency
pwm1A = GPIO.PWM( Motor1A, 250 )
# we do the same for Motor1B (for reverse movement)
pwm1B = GPIO.PWM( Motor1B, 250 )
```

At line 21, add:

```
# we start both pwm controllers to zero duty cycle (off)
```

Edward C. Charbonnet • Carlos Luna • a year ago

Carlos - yes I am still interested and thanks so much.

Taking the last of Jason code examples on that page and making the changes you recommend sounds promising but. The chip used in this project is the L293D. Does it have PWM (pulse width modulation) capabilities simply using the commands you provide. I would have thought that slightly different hardware (IC chip) would be required.

Carlos Luna • Edward C. Charbonnet • a year ago

Hi Edward.

The L293D does not generate PWM pulses. Its main purpose is to provide voltage to the motors and to reverse the current to them. Adafruit has a very good lesson about this on their site: https://learn.adafruit.com/ada...

So it is up to your micro controller to generate this pulse. Adafruit have created their own Linux distro, called Occidentalis, to tackle the problem of the Pi only having one PWM pin. They’ve created kernel modules to provide low-level, more reliable pulses. With that you can use any of the I/O pins on RPi Motor. But you won’t find this in one of the many
as a PWM pin. But, as someone said in one of the many articles I’ve read about this, if you want reliable, uninterruptable PWM pulses, use an Arduino.

Regards,
Carlos

Seth Williams • a year ago
Hi Jason, This is an excellent guide on controlling DC motors through Python. Thanks a lot for sharing. I’ll be looking forward to more such tutorials from you. You can find out more in our Python course curriculum here http://www.fireboxtraining.com....

Bohao • a year ago
I followed this tutorial. However, my DC motor doesn’t run and L293D chip is getting very hot. I couldn’t figure out what is going on.

Bohao • a year ago
I used LED light instead of DC motor for testing, everything is working. My DC motor is RS-360SH. Does anyone know why it doesn’t work?

Nahuel Mata • a year ago
Hey i have to do a hydraulic bridge for a science project and im having trouble with the programing im using two dc motors and the raspberry pi with the l293d and i was wondering how to control it, i mean where is al this in terms of UI. Also if i script it how do i assign it to the keys and make the motor rotate only when i press the keys?

Nicolaos Constantinou • a year ago
I have the same problem both going forward but only one go back.... The best solution and it seems to work is to have to L293D chip......Only this way (in my opinion)

Jason Barnett • Nicolaos Constantinou • a year ago
Which version Raspberry Pi do you have? I have noticed I forgot about the Rev1/Rev2 issue with pins 21 and 27 being used - So if you have a Revision 2 (with the mounting holes) you will need to swap 21 with 27 in the Python code.

Tanner Wendland • 2 years ago
No matter what I do the second motor won’t go in reverse, but the first one does.

Austin Roberts • Tanner Wendland • 2 months ago
I realize this is probably way too late for you, but it might benefit someone.
I realize this is probably way too late for you, but it might benefit someone else reading this tutorial: I was using a Pi2, and used a different set of pins than the tutorial suggests. At one point I had the pin variables set incorrectly for Motor1R, which meant it was always low. If you have a multimeter, check the output of the pins and make sure you’re turning on the right ones before turning your attention to the driver chip.

Jason Barnett · Tanner Wendland · 2 years ago

Silly as it sounds, have you checked the wiring? If that is fine it could be a faulty driver chip, I have had quite a few fail. It took about 2 hours on one to come to the conclusion when 2 motors went forward and only 1 went back. Unfortunately they are very susceptible to ESD (Electrostatic Damage)

Piotr · 2 years ago

What this tutorial needs is one extra chapter about how to make a library/class for operating the motors and not just writing GPIO.output all the way.

And to mention that there are small PCBs with motor controllers, some even robot oriented that power Pi from their power (battery) supply :)

Jason Barnett · Piotr · 2 years ago

I plan to expand on this further, making it easier to program. In this tutorial I wanted to get to the basics, and breakdown what is happening instead of hiding it away.

Jonny · 10 months ago

Can i use 6 batteries to make it faster