Raspberry Pi GPIO-Part 2: Adafruit DC Motor HAT for Raspberry Pi

Overview

The **DC+Stepper Motor HAT from Adafruit** is a perfect add-on for any motor project as it can drive up to 4 DC or 2 Stepper motors with full PWM speed control. However, the Raspberry Pi does not have a lot of PWM pins, we use a fully-dedicated PWM driver chip onboard to both control motor direction and speed. This chip handles all the motor and speed controls over I2C. Only two GPIO pins (SDA & SCL) are required to drive the multiple motors, and since it is I2C you can also connect any other I2C devices or HATs to the same pins.

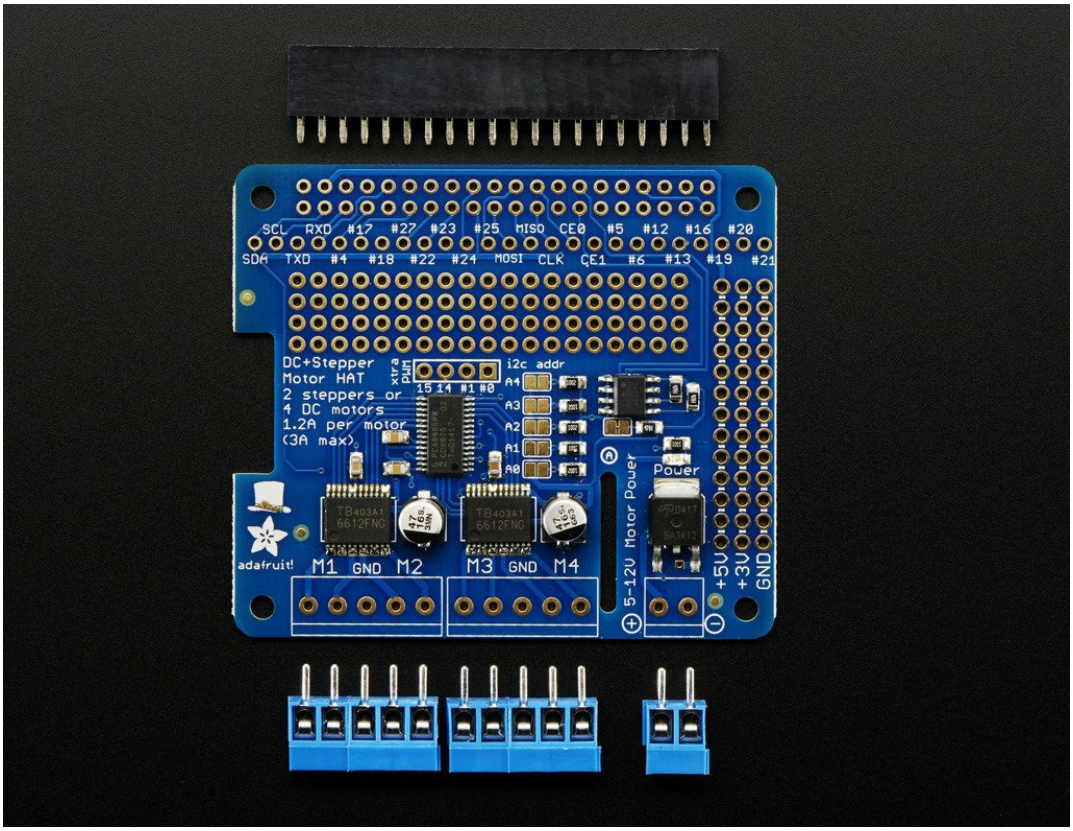
Note: I2C is a very commonly used standard designed to allow one chip to talk to another. So, since the Raspberry Pi can talk I2C we can connect it to a variety of I2C capable chips and modules.

Features:

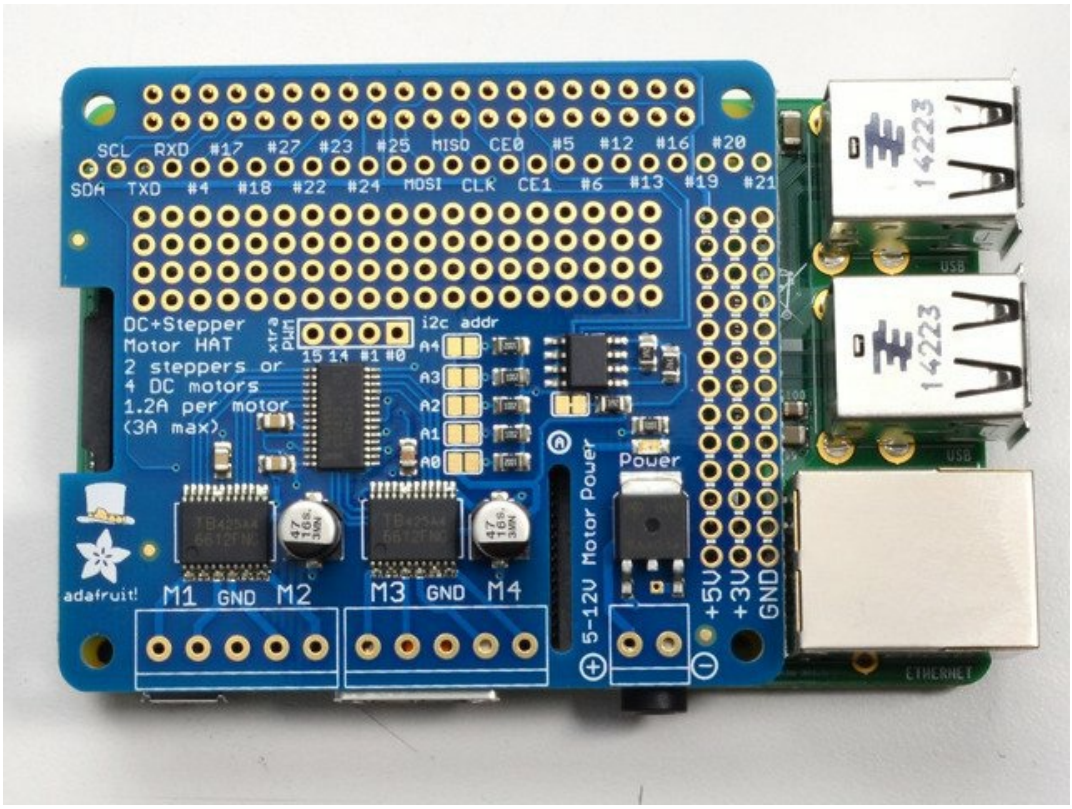
- 4 H-Bridges: TB6612 MOSFET chipset provides 1.2A per bridge (3A brief peak) with thermal shutdown protection, internal kickback protection diodes. Can run motors on 4.5VDC to 13.5VDC.
- **Up to 4 bi-directional DC motors** with individual 8-bit speed selection (so, about 0.5% resolution).
- **Up to 2 stepper motors** (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping.
- Big terminal block connectors to easily hook up wires (18-26AWG) and power.
- Polarity protected 2-pin terminal block and jumper to connect external 5-12VDC power.
- Install the easy-to-use Python library.

Assembly

The Motor HAT comes with an assembled and tested HAT, terminal blocks, and 2x20 plain header. Some soldering is required to assemble the headers on. Here we leave a link with a step-by-step guide of how to solder the headers and a video to show you tips on soldering.



Once the motor HAT is assembled, we place it on top so that the short pins of the 2x20 header line up with the pads on the HAT.



Powering Motors

Note the HAT does not power the Raspberry Pi, and we strongly recommend having two separate power supplies - one for the RPi and one for the motors, as motors can put a lot of noise onto a power supply and it could cause stability problems.

Voltage requirements

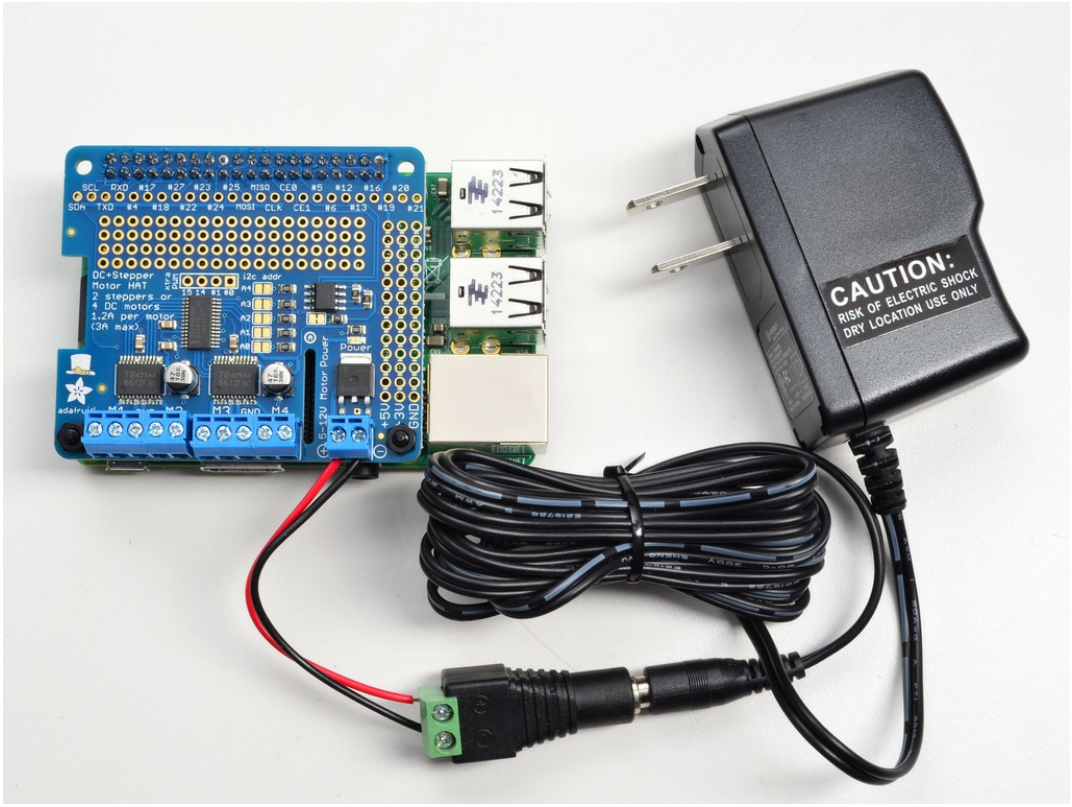
The motor controllers on this HAT are designed to run from **5V to 12V**. Therefore, the first important thing is to verify the voltage specifications for the motor. Some small hobby motors are only intended to run at 1.5V (**MOST 1.5-3V MOTORS WILL NOT WORK or will be damaged by 5V power**), but its just as common to have 6-12V motors.

Current requirements

The motor driver chips that come with the kit are designed to provide up to **1.2 A per motor**, with 3A peak current. Note that once you head towards 2A you will probably want to put a heat-sink on the motor driver, otherwise you will get thermal failure, possibly burning out the chip.

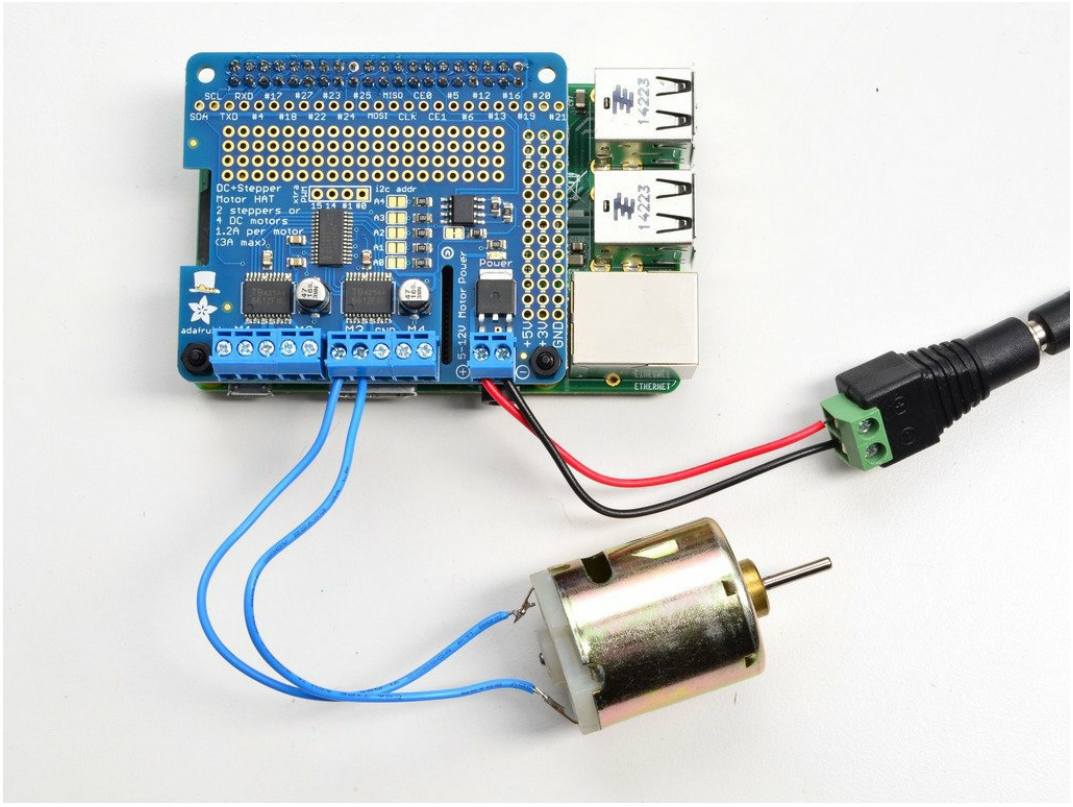
Am important thing you can not run motors off of a 9V battery so don't waste your time/batteries!

Therefore, you can use a 9V 1A, 12V 1A, or 12V 5A DC regulated switching power adapter. In case you want to make it portable, you can use a big Lead Acid or multiple-AA NiMH battery pack of 4 to 8 batteries to vary the voltage from about 6V to 12V as your motors require.



Connecting DC Motors

To connect a motor, simply solder two wires to the terminals and then connect them to either the **M1, M2, M3**, or **M4**. If your motor is running 'backwards' from the way you like, just swap the wires in the terminal block. For this demo, please connect it to **M3**.



Installing Software

We can download the Python library to control DC and stepper motors. Before you start, we need to install the **python smbus library** as well as 'git'. For the latter, execute the following command:

```
$ sudo apt-get install python-smbus git
```

Now, we download the code as:

```
$ cd code
$ git clone https://github.com/adafruit/Adafruit-Motor-HAT-Python-Library.git
$ cd Adafruit-Motor-HAT-Python-Library
$ sudo python setup.py install
```

Now you can get started with testing to watch your motor spin back and forth. First access to:

```
$ cd Adafruit-Motor-HAT-Python/examples
$ nano DCTest.py
```

Here you will see the code which shows you everything the MotorHAT library can do and how to do it.

DC motor control

1. Start with importing at least these libraries:

```
#!/usr/bin/python
from Adafruit_MotorHAT import Adafruit_MotorHAT, Adafruit_DCMotor
```

```
import time
import atexit
```

2. The MotorHAT library contains a few different classes, one is the **MotorHAT class** itself which is the main PWM controller. You always need to create an object, and set the address (or frequency). By default the address is 0x60. We can change this address, but for now we are not going to do it.

```
# create a default object, no changes to I2C address or frequency
mh = Adafruit_MotorHAT(addr=0x60)
```

3. The PWM driver is 'free running' - that means that even if the python code or Pi linux kernel crashes, the PWM driver will still continue to work. But it means that the motors **DO NOT STOP** when the python code quits For that reason, we strongly recommend this 'at exit' code when using DC motors, it will do its best to shut down all the motors.

```
# recommended for auto-disabling motors on shutdown!
def turnOffMotors():
        mh.getMotor(1).run(Adafruit_MotorHAT.RELEASE)
        mh.getMotor(2).run(Adafruit_MotorHAT.RELEASE)
        mh.getMotor(3).run(Adafruit_MotorHAT.RELEASE)
        mh.getMotor(4).run(Adafruit_MotorHAT.RELEASE)

atexit.register(turnOffMotors)
```

Creating the DC motor object

4. Now that you have the motor HAT object, note that each HAT can control up to 4 motors. That means you can have multiple HATs running. To create the actual DC motor object, you can request it from the MotorHAT object you created above with `getMotor(num)` with a value between 1 and 4, for the terminal number that the motor is attached to

```
# In this case is M3
myMotor = mh.getMotor(3)
```

DC motors are simple beasts, you can basically only set the speed and direction.

Setting DC Motor Speed

5. To set the speed, call `setSpeed(speed)` where speed varies from 0 (off) to 255 (Maximum). This is the PWM duty cycle of the motor.

```
# set the speed to start, from 0 (off) to 255 (max speed)
myMotor.setSpeed(150)
```

Setting DC Motor Direction

6. To set the direction, we use the funciton `run(direction)` where direction is a constant from one of the following:
7.

- `Adafruit_MotorHAT.FORWARD` - DC motor spins forward.
- `Adafruit_MotorHAT.BACKWARD` - DC motor spins backward.

- `Adafruit_MotorHAT.RELEASE` - DC motor is 'off', not spinning but will also not hold its place.

```
while (True):
        print "Forward! "
        myMotor.run(Adafruit_MotorHAT.FORWARD)

        print "\tSpeed up..."
  # This will loop from 0-254
        for i in range(255):
                myMotor.setSpeed(i)
    # It will stop 10 ms
                time.sleep(0.01)

        print "\tSlow down..."
  # This will loop from 244-0
        for i in reversed(range(255)):
                myMotor.setSpeed(i)
                time.sleep(0.01)

        print "Backward! "
        myMotor.run(Adafruit_MotorHAT.BACKWARD)

        print "\tSpeed up..."
        for i in range(255):
                myMotor.setSpeed(i)
                time.sleep(0.01)

        print "\tSlow down..."
        for i in reversed(range(255)):
                myMotor.setSpeed(i)
                time.sleep(0.01)

        print "Release"
        myMotor.run(Adafruit_MotorHAT.RELEASE)
        time.sleep(1.0)
```

Reference[1]