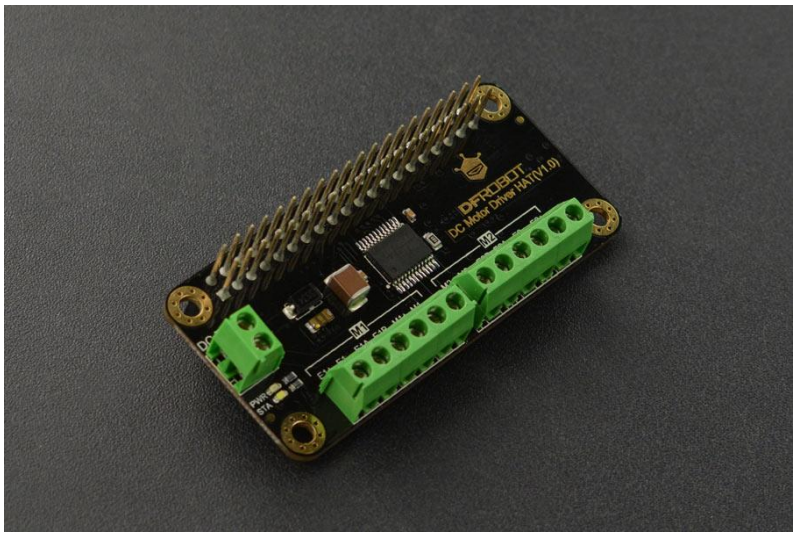
SKU:
DFR0592



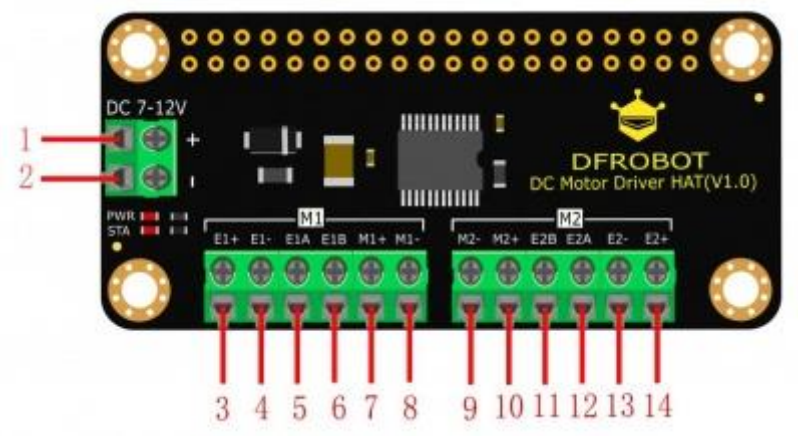
Introduction

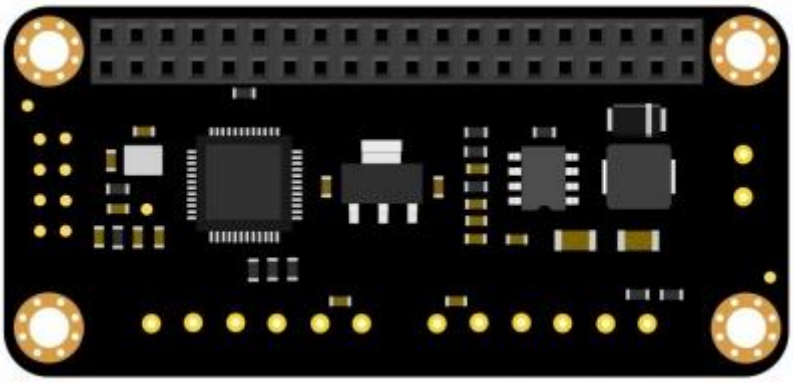
This is a Raspberry Pi DC motor driver board with on-board encoder interface, which can drive 2-way DC motor and DC motor with encoder. It communicates with Raspberry Pi via I2C, easy to configure and drive motors. The DC Motor Driver HAT adopts STM32 micro-processor to analyze the command sent by the upper host, and then convert into motor drive signal after calculation. Besides that, a high-performance TB6612FNG motor drive chip is integrated on the module. The single channel continuous current is 1.2A and the peak current is up to 2A. The motor driver supports 7~12V wide input voltage.

Specification

- Main Controller: STM32
- Operating Voltage: 7~12V
- Logical Voltage: 5V
- Motor Driver Chip: TB6612FNG
- Communication Interface: I2C
- Default IIC Address: 0 x10
- Continuous Drive Current: 2A
- Working Mode：2-way DC Motor/ 2-way DC Motor with Encoder
- 2 Indicators
    - Power Indicator
    - Communication Status Indicator
- Dimension: 65×30mm/2.56×1.18”

Board Overview

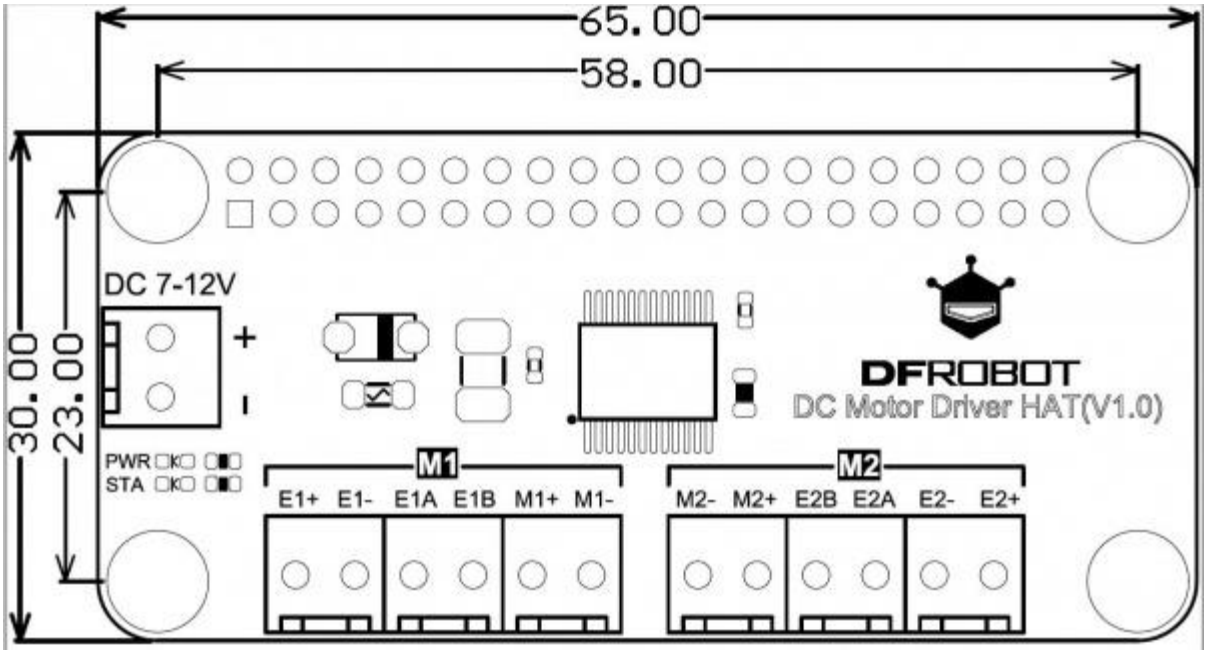




Num	Name	Description
1	+	External Power Positive (7~12V)
2	-	External Power Negative
3	E1+	Encoder E1 Positive
4	E1-	Encoder E1 Negative
5	E1A	Encoder E1 A Phase Output
6	E1B	Encoder E1 B Phase Output

Num	Name	Description
7	M1+	Motor M1 Positive
8	M1-	Motor M1 Negative
9	M2-	Motor M2 Negative
10	M2+	Motor M2 Positive
11	E2B	Encoder E2 B Phase Output
12	E2A	Encoder E2 A Phase Output
13	E2-	Encoder E2 Negative
14	E2+	Encoder E2 Positive

Dimension Diagram



Function Configuration

- Control Mode: I2C command
- Driving Signal: PWM 100Hz~12750Hz
- Encoder Motor Function(Only DC Motor with Encoder)
    - Disable/enable Encoder
    - Reduction Ratio Configuration
    - Get Revolving Speed of the Encoder Motor
- Basic Function
    - Set Driving Signal Frequency
    - Set Duty Ratio
    - Adjust Speed
        - PWM Speed Adjustment (Keep the frequency, change the duty ratio. The larger the duty ratio, the larger the rotating speed)
        - Adjust the Driving Signal Frequency

Tutorial (Raspberry Pi)

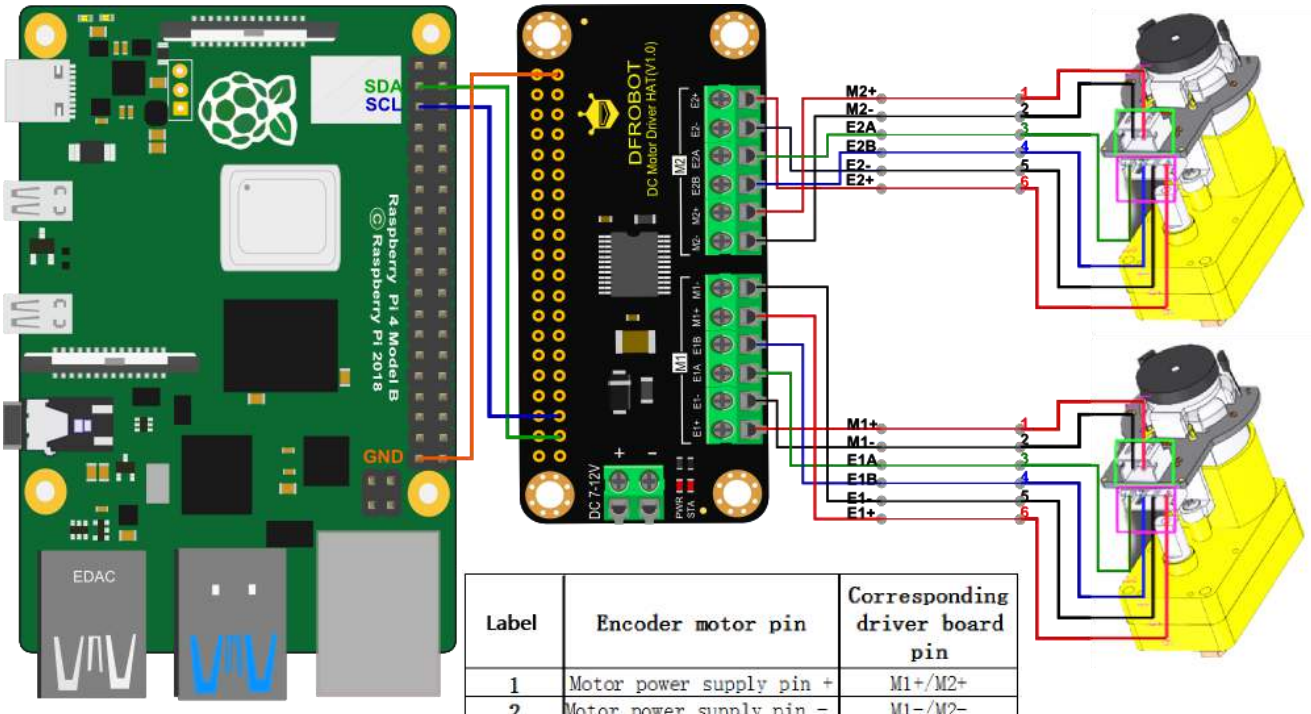
Requirements

- **Hardware**
    - Raspberry Pi x 1
    - DC Motor Driver HAT x 1
    - Metal DC Geared Motor w/Encoder x 2
    - DC Motors x 2

⚠ **NOTE**: the operating voltage should be within 7~12V, and the motor's current should not be more than 2A.

- **Software**
    - DFRobot_RasbperryPi_Motor_Library

Connection Diagram



Label	Encoder motor pin	Corresponding driver board pin
1	Motor power supply pin +	M1+/M2+
2	Motor power supply pin -	M1-/M2-
3	Encoder A phase output	E1A/E2A
4	Encoder B phase output	E1B/E2B
5	Encoder supply GND	E1-/E2-
6	Encoder supply +	E1+/E2+

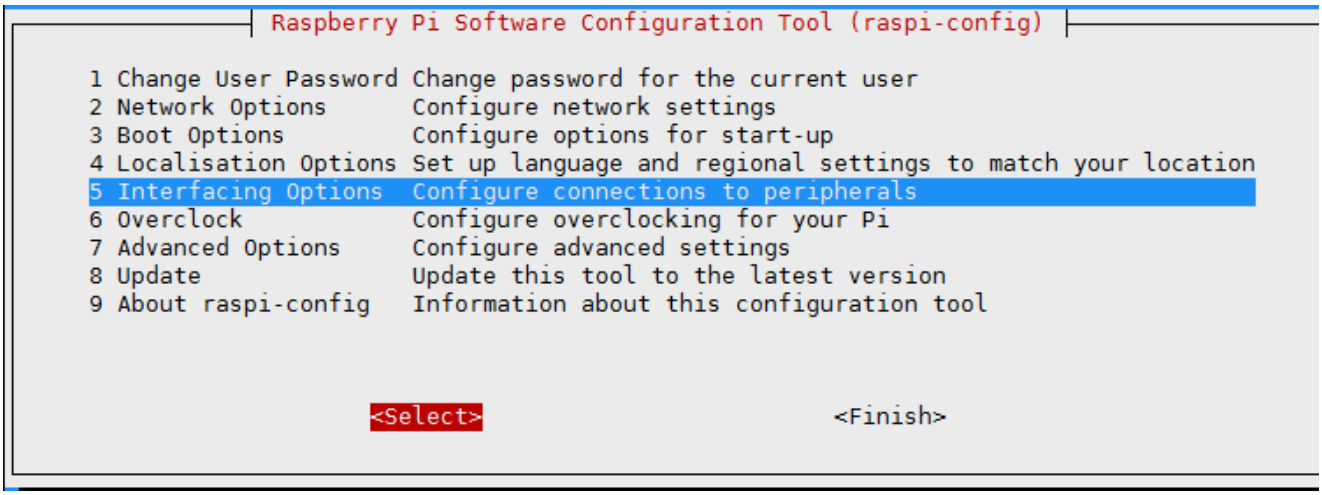
Operations

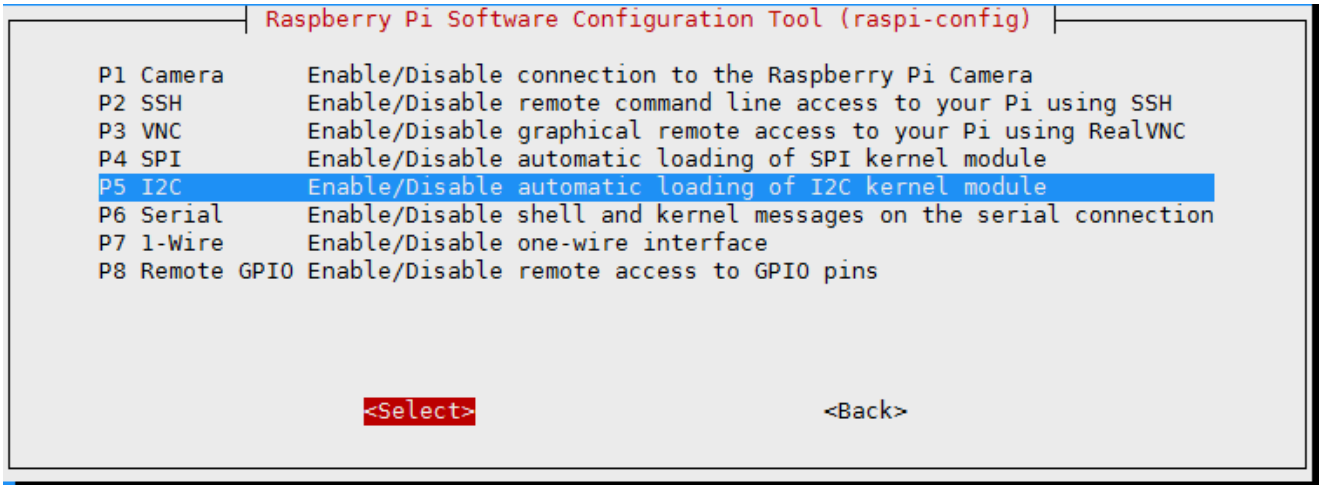
Step1: Plug the DC Motor Driver HAT in a Raspberry Pi's main-board, connect driver and motor, power up.

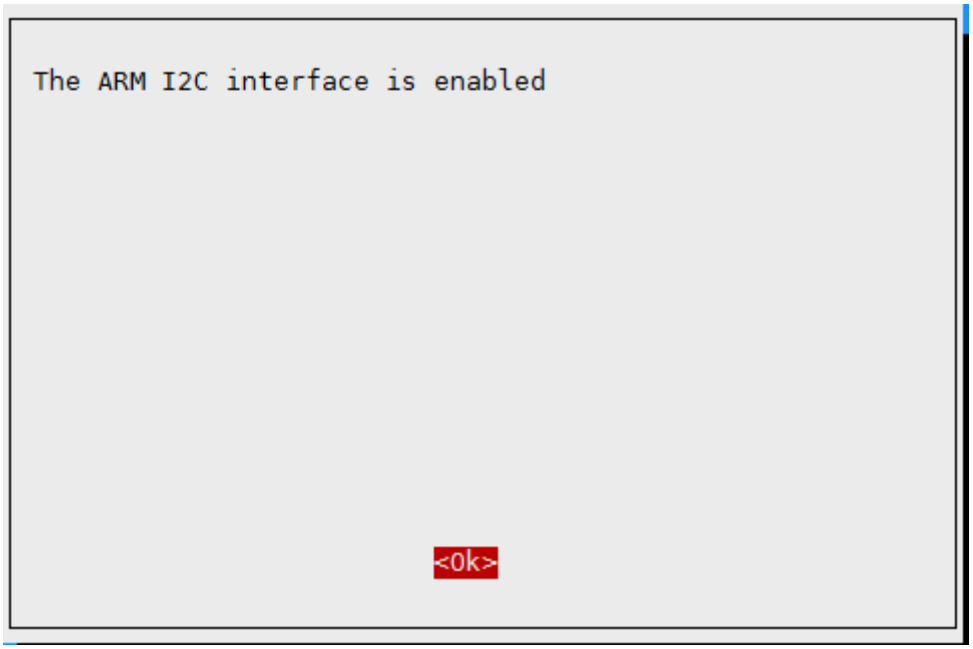
Step2: Detect if the I2C interface is enabled. Input the command **i2cdetec-y 1**, if it is disabled, the following interface will appear:

```
pi@raspberrypi:~ $ i2cdetect -y 1
Error: Could not open file `/dev/i2c-1' or `/dev/i2c/1': No such file or directory
```

Step3: Enable I2C interface. (Skip this step if I2C is already enabled.) Input the command **sudo raspi-config** to enter the configuration interface. Shown as below:


Raspberry Pi Software Configuration Tool (raspi-config)
1 Change User Password Change password for the current user
2 Network Options Configure network settings
3 Boot Options Configure options for start-up
4 Localisation Options Set up language and regional settings to match your location
5 Interfacing Options Configure connections to peripherals
6 Overclock Configure overclocking for your Pi
7 Advanced Options Configure advanced settings
8 Update Update this tool to the latest version
9 About raspi-config Information about this configuration tool
<Select>
<Finish>


Raspberry Pi Software Configuration Tool (raspi-config)
P1 Camera Enable/Disable connection to the Raspberry Pi Camera
P2 SSH Enable/Disable remote command line access to your Pi using SSH
P3 VNC Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI Enable/Disable automatic loading of SPI kernel module
P5 I2C Enable/Disable automatic loading of I2C kernel module
P6 Serial Enable/Disable shell and kernel messages on the serial connection
P7 1-Wire Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins
<Select>
<Back>


The ARM I2C interface is enabled
<Ok>

When finished the configuration, input **sudo reboot** to restart Raspberry Pi.

Step4: Detect I2C address. Input **i2cdetect -y 1** to detect the device I2C address, shown as below:

```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: 10 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Step5: Download the DFRobot_RaspberryPi_Motor Library. Enter **sudo git clone https://github.com/DFRobot/DFRobot_RaspberryPi_Motor**. Input **ls** to check commands.

```
pi@raspberrypi:~ $ git clone  https://github.com/DFRobot/DFRobot_RaspberryPi_Motor
Cloning into 'DFRobot_RaspberryPi_Motor'...
remote: Enumerating objects: 44, done.
remote: Counting objects: 100% (44/44), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 44 (delta 18), reused 39 (delta 13), pack-reused 0
Unpacking objects: 100% (44/44), done.
pi@raspberrypi:~ $ ls
Desktop                    DFRobot_RaspberryPi_Motor-master  Documents  Pictures  Public
DFRobot_RaspberryPi_Motor  DFRobot_VL53L1X                   Downloads  program   Videos
```

Example

Enter the python catalogue in DFRobot_RaspberryPi_Motor library.

```
cd /DFRobot_RaspberryPi_Motor/raspberry
```

How to drive a DC motor with encoder?

Use the variable M1,M2, ALL for id parameter to represent motor M1, motor M2, M1 and M2.

- Step1. Enable the Encoder:

```
•  set_encoder_enable(self, id)
•   eg：
•     board = Board(1,0x10)
•     set_encoder_enable ([board.M1]) #Enable encoder motor M1
•     set_encoder_enable ([board.M2]) #Enable encoder motor M2
```

- ```
  set_encoder_enable（[board.M1,board.M2]）#Enable encoder Motor M1 and M2
  ```

```
set_encoder_enable（board.ALL）#Enable encoder motor M1 and M2
```

- Step2. Set the motor reduction ratio(43:1 in demo). Related with the reduction of the motor.

```
set_encoder_reduction_ratio(self, id, reduction_ratio)
```

- Step3. Set the frequency of PWM signal (1000Hz in demo)

```
set_moter_pwm_frequency(self, frequency)
```

- Step4. Set the rotating direction and speed of motor M1, M2, M1 and M2. (Duty ratio: 0~100)

```
motor_movement(self, id, orientation, speed)
```

- Step5. Get the speed of the encoder motor

```
get_encoder_speed(self, id)
```

⚠ **NOTE**: Different motors have different reduction ratio. Please revise the reduction ratio of the motor before running the codes in the demo.

Click to check more function configuration and usage description.

Run the example DC_Motor_Demo.py. Print the duty ratio and rotating speed of the encoder motor.

```
python DC_Motor_Demo.py
```

- Program Function: Motor driver generates a signal with 1KHz frequency and a changing duty ratio within 5%~95% to make the encoder motor M1 to rotate clockwise with speed first increasing then decreasing; M2 rotate anti-clockwise with speed first increasing then decreasing, and stop. At the same time, print the rotating speed on the serial port. The whole program is repeatedly executed.
```

```
pi@raspberrypi:~/DFRobot_RaspberryPi_Motor/raspberry $ ls
DC_Motor_Demo.py  DFRobot_RaspberryPi_DC_Motor.py
pi@raspberrypi:~/DFRobot_RaspberryPi_Motor/raspberry $ python DC_Motor_Demo.py
Board list conform:
['0x10']
board begin success
duty: 5, M1 encoder speed: -1 rpm, M2 encoder speed 1 rpm
duty: 15, M1 encoder speed: -18 rpm, M2 encoder speed 19 rpm
duty: 25, M1 encoder speed: -36 rpm, M2 encoder speed 36 rpm
duty: 35, M1 encoder speed: -54 rpm, M2 encoder speed 55 rpm
duty: 45, M1 encoder speed: -73 rpm, M2 encoder speed 298 rpm
duty: 55, M1 encoder speed: -90 rpm, M2 encoder speed -357 rpm
duty: 65, M1 encoder speed: 0 rpm, M2 encoder speed 409 rpm
duty: 75, M1 encoder speed: -126 rpm, M2 encoder speed 124 rpm
duty: 85, M1 encoder speed: -144 rpm, M2 encoder speed 512 rpm
duty: 95, M1 encoder speed: 0 rpm, M2 encoder speed 522 rpm
duty: 85, M1 encoder speed: -148 rpm, M2 encoder speed 288 rpm
duty: 75, M1 encoder speed: -131 rpm, M2 encoder speed 651 rpm
duty: 65, M1 encoder speed: 0 rpm, M2 encoder speed 112 rpm
duty: 55, M1 encoder speed: -98 rpm, M2 encoder speed 247 rpm
duty: 45, M1 encoder speed: -80 rpm, M2 encoder speed 154 rpm
duty: 35, M1 encoder speed: -62 rpm, M2 encoder speed -88 rpm
duty: 25, M1 encoder speed: -43 rpm, M2 encoder speed 42 rpm
duty: 15, M1 encoder speed: -23 rpm, M2 encoder speed 23 rpm
stop all motor
board status: everything ok
```

How to drive a DC motor?

Connect the DC motor to the interface of M1+, M1- or M2+, M2-. Disable the function of encoder as the above demo, set frequency, duty ratio, rotating direction.

⚠ **NOTE**: For motors without encoder, we cannot get its speed, but only adjust the speed through PWM.

Compatibility Test

MCU	Pass	Fail
Raspberyy Pi 4B	√	
Raspberyy Pi 3B	√	
Raspberyy Pi 3B+	√	
Raspberyy Pi Zero W	√	

MCU	Pass	Fail
Raspberyy Pi 2B+	√	

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**

More Documents

Schematic Diagram