Drive Cokoino Pi Power & 4WD Hat to run the 4WD car

Table

1. Preface

Our final form of this product is a small car chassis with 4 motors and 4 wheels, without motor drive modules, control boards, batteries, and other things. Its highlight lies in its development and scalability. You can choose the motor drive and control board you want to use, install it on the chassis of this car, and make it run, becoming a four-wheel drive car. This will be a challenging and fulfilling task. Wishing you success!
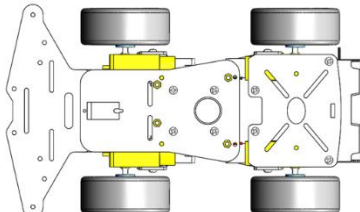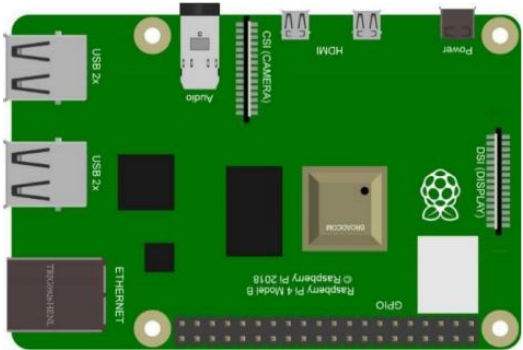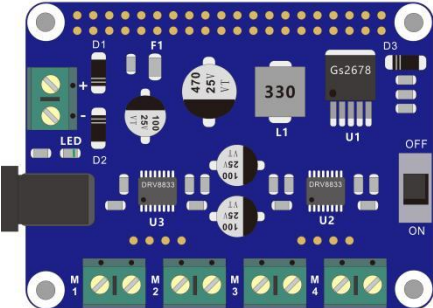
You can also refer to our Demo2, where we assembled Cokoino 4WD Robot HAT , Raspberry Pi 4B, and 2pcs 18650 batteries onto the chassis of the car, creating a 4WD car based on Raspberry Pi. The following tutorial will provide a detailed introduction to Demo2, including component list, component related knowledge, circuit connection, code, and more. If you are interested in demo2, you can refer to the demo1 checklist to prepare relevant materials for experimentation.

If you have any technical issues or suggestions, please provide feedback to us via email:
cokoino@outlook.com

2. Component List

For this Demo experiment, what do you need to prepare like below list

Component/Module	QTY	Picture	Remark
4WD Car Chassis	1		Provided by the 4WD Car Chassis Kit
Raspberry Pi 4B	1		You need to prepare these by yourself.These just as an example,you can DIY what is you want and prepared.
Cokoino 4WD Robot HAT	1		

18650 battery	2		

3. Component related knowledge

3.1Knowledge of the TT Motor

As shown in the figure below, the TT motor consists of a DC motor and related gears, with a yellow outer shell fastened.
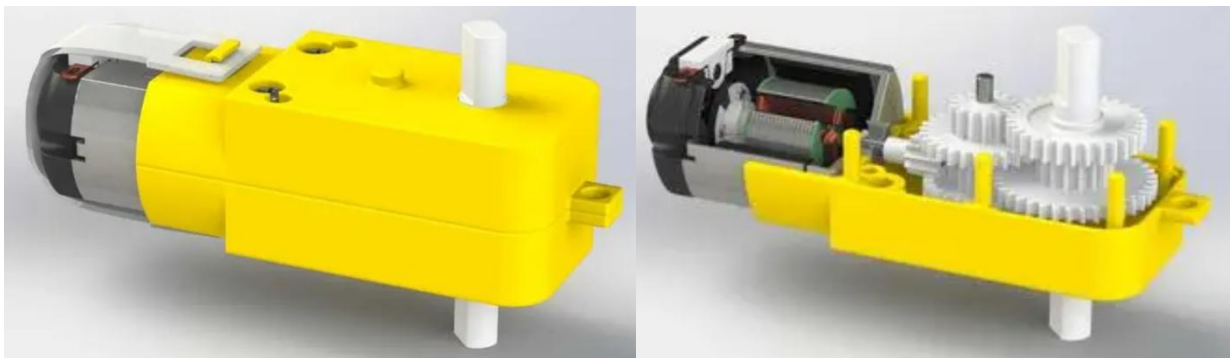
DC Motor

When motor is connected to the power supply, it will rotate in one direction. Reverse the polarity of power supply, the motor will rotate in the opposite direction.

And the speed of motor depends on the voltage between two ends. The larger the voltage, the lager the speed.
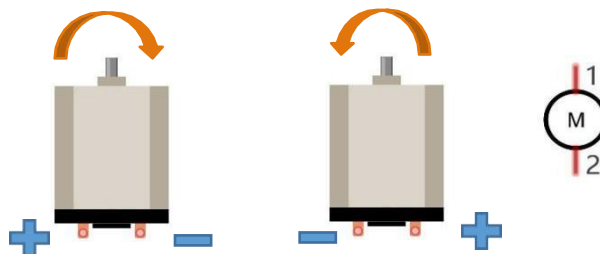
PWM

PWM, Pulse Width Modulation, uses digital pins to send certain frequencies of square waves, that is, the output of high levels and low levels, which alternately last for a while. The total time for each set of high levels and low levels is generally fixed, which is called the period (the reciprocal of the period is frequency). The time of high level outputs are generally called "pulse width", and the duty cycle is the percentage of the ratio of pulse

duration, or pulse width (PW) to the total period (T) of the waveform.

The longer the output of high levels last, the larger the duty cycle and the higher the corresponding voltage in analog signal will be. The following figures show how the analogs signal voltage vary between 0V-5V (high level is 5V) corresponding to the pulse width 0%-100%:
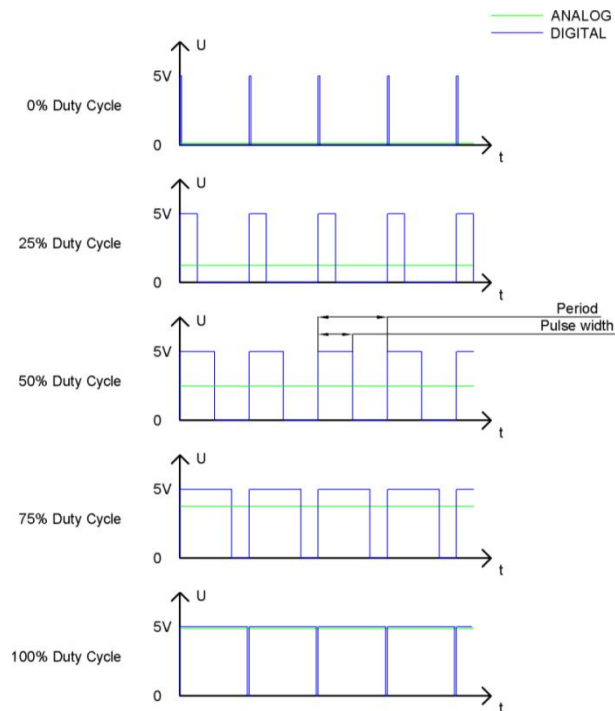
The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on.

GND   GND

3.2 Knowledge of Cokoino 4WD Robot HAT

3.2.1 overview

Cokoino 4WD Robot HAT is a Raspberry Pi power and motor driver board suitable for Raspberry Pi 4B/3B/3B+/3A+/2B/1B+/1A+/Pi Zero/Pi Zero W. It is equipped with a PWM voltage management IC GS2678, which can support up to 7-12V input, rated output voltage of 5.1V, and maximum output current of 4A. It can support Raspberry Pi to work without the adapter. Simply connect a 7-12V battery pack to the board, which is more convenient for Raspberry Pi use. At the same time, it is equipped with two Dual-H

Support email:cokoino@outlook.com

Bridge Current Control Motor Driver DRV8833 onboard, which can independently control the speed and steering of four DC motors at the same time. Therefore, it is more recommended for use on 4WD small cars developed for Raspberry Pi.

It has already been listed on Cokoino's Amazon store. If you like it, you can purchase it from Cokoino's Amazon store.

3.2.2 Appearance of the Cokoino 4WD Robot HAT

2D image

Physical pictures

Support email:cokoino@outlook.com
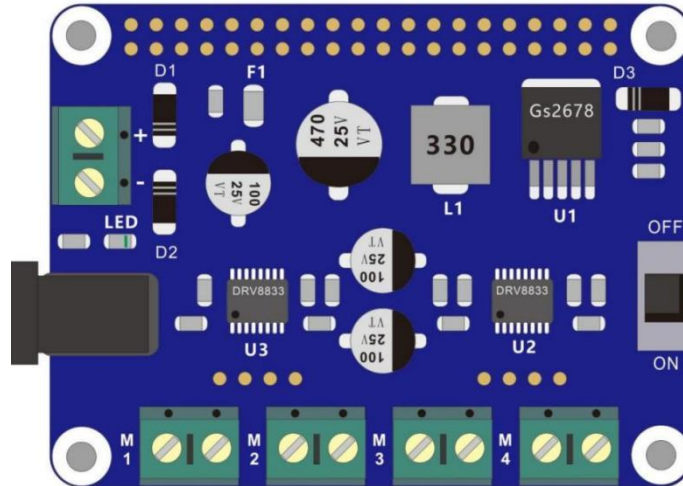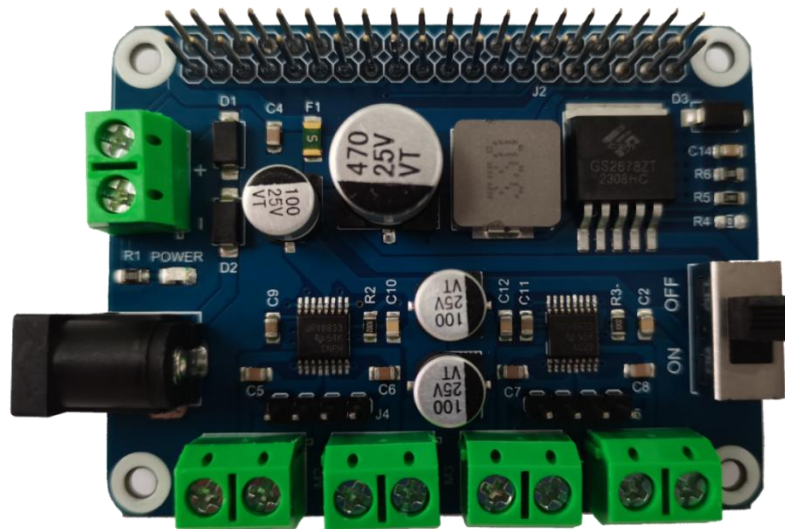
3.2.3 Introduction for the Cokoino 4WD Robot HAT

The Cokoino 4WD Robot HAT is suitable for Raspberry Pi 4B/3B/3B+/3A+/2B/1B+/1A+/Pi Zero/Pi Zero W, using a plastic long needle double row female. The row female corresponds to the insertion of Raspberry Pi, and the long needle corresponds to the signal of each pin of Raspberry Pi.

For the main components and functions of the Cokoino 4WD Robot HAT, please refer to the following table.
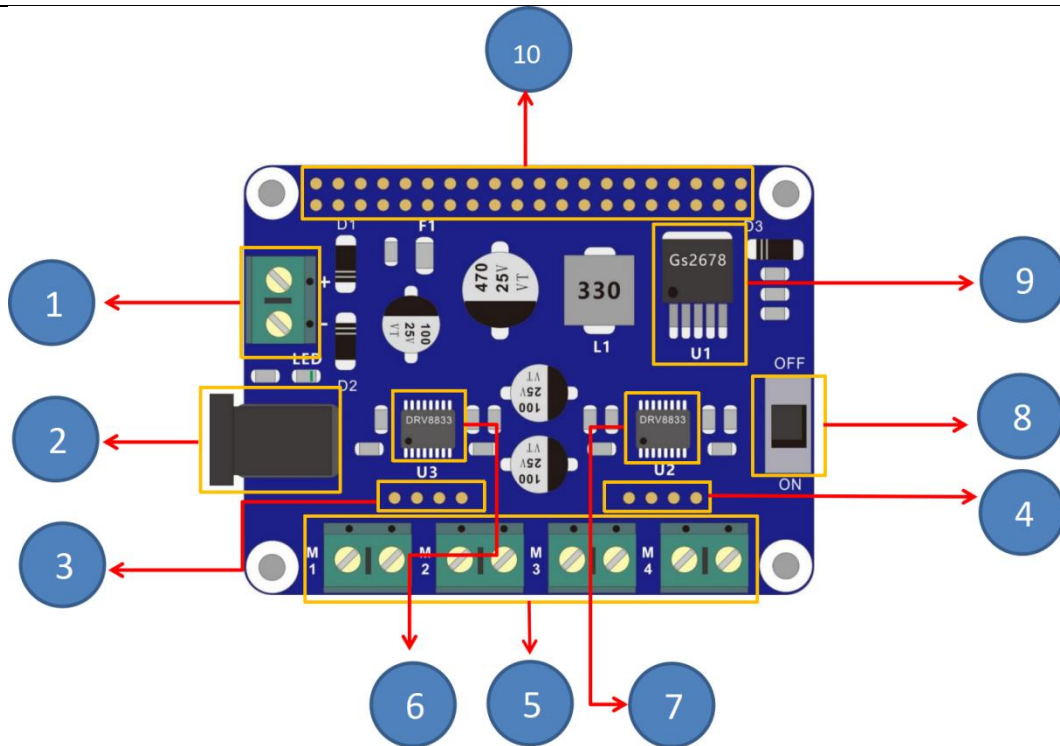
Pin distribution of Pi Power&4WD HAT relative to Raspberry Pi

	Pin 1	Pin 2	
+3V3			+5V
GPIO2 / SDA1			+5V
GPIO3 / SCL1			GND
GPIO4			TXD0 / GPIO 14
GND			RXD0 / GPIO 15
GPIO17			GPIO 18
GPIO27			GND
GPIO22			GPIO 23
+3V3			GPIO 24
GPIO10 / MOSI			GND
GPIO9 / MISO			GPIO 25
GPIO11 / SCLK			CE0# / GPIO8
GND			CE1# / GPIO7
GPIO0 / ID_SD			ID_SC / GPIO1
GPIO5			GND
GPIO6			GPIO12
GPIO13			GND
GPIO19 / MISO			CE2# / GPIO16
GPIO26			MOSI / GPIO20
GND			SCLK / GPIO21
	Pin 39	Pin 40	

Support email:cokoino@outlook.com
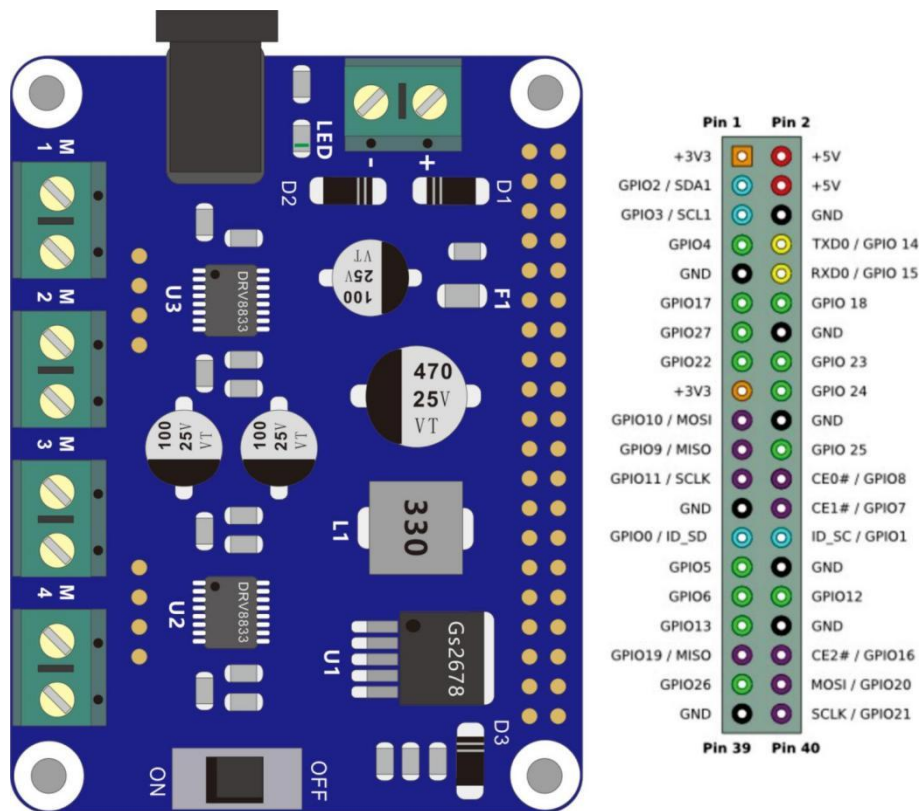
No.	Component	Function
1	Power port(terminal)	Power input 2pin port, pay attention to the positive and negative when connecting the external power supply, power supply range: 7-9V DC
2	Power port(DC port)	Power input DC port, power supply range: 7-9V DC
3	Motor interface (row pin)	2.54 * 4 pin row pin, corresponding to circuit M1, M2 motor interface
4	Motor interface (row pin)	2.54 * 4 pin row pin, corresponding to circuit M3, M4 motor interface
5	Motor interface (terminal)	KF301 two pin terminals, corresponding to circuit M1, M2, M3, M4 motor interface
6	DRV8833	It is motor driver chip that can drive two DC Motors
7	DRV8833	It is motor driver chip that can drive two DC Motors
8	Power Switch	Dial to ON to power on, dial to OFF to power off
9	GS2678	GS2678 is an asynchronous DC buck converter with PWM voltage management IC and pulse width modulation output (PWM). The input voltage range can be from a minimum of 3.6V to a maximum of 32V, the output voltage can be adjusted from 0.8V-23V, and the output current can reach up to 5A
10	RaspberryPi interface	Plastic long needle double row female, the row female is connected to the row pin interface of the Raspberry Pi, and the long needle is used to expand the pins of the Raspberry Pi to the top of the Cokoino 4WD Robot HAT

3.2.4 DRV8833 Specification

Features

- Dual-H-Bridge Current-Control Motor Driver
  - Can Drive Two DC Motors or One Stepper Motor
  - Low MOSFET ON-Resistance: HS + LS 360 mΩ
- Output Current (at V_M = 5 V, 25°C)
  - 1.5-A RMS, 2-A Peak per H-Bridge in PWP and RTY Package Options
  - 500-mA RMS, 2-A Peak per H-Bridge in PW

  Package Option
- Outputs can be in Parallel for
  - 3-A RMS, 4-A Peak (PWP and RTY)
  - 1-A RMS, 4-A Peak (PW)
- Wide Power Supply Voltage Range:

  2.7 to 10.8 V
- PWM Winding Current Regulation and Current

  Limiting
  Enhanced Surface-Mount Packages

Applications

- Battery-Powered Toys
- POS Printers
- Video Security Cameras
- Office Automation Machines
- Gaming Machines
- Robotics

Description

The DRV8833 device provides a dual bridge motor driver solution for toys, printers, and other mechatronic applications.

The device has two H-bridge drivers, and can drive two DC brush motors, a bipolar stepper motor, solenoids, or other inductive loads.

The output driver block of each H-bridge consists of

N-channel power MOSFETs configured as an H-bridge to drive the motor windings. Each H-bridge includes circuitry to regulate or limit the winding current.

Internal shutdown functions with a fault output pin are provided for overcurrent protection, short-circuit protection, undervoltage lockout, and overtemperature. A low-power sleep mode is also

provided.

The DRV8833 is packaged in a 16-pin WQFN package with PowerPAD™ (Eco-friendly: RoHS & no Sb/Br). Thermally

Device Information[1]

PART NUMBER	PACKAGE	BODY SIZE (NOM)
DRV8833	TSSOP (16)	5.00 mm × 4.40 mm
	HTSSOP (16)	5.00 mm × 4.40 mm
	WQFN (16)	4.00 mm × 4.00 mm

(1) For all available packages, see the orderable addendum at the end of the data sheet.

Simplified Schematic

Support email:cokoino@outlook.com
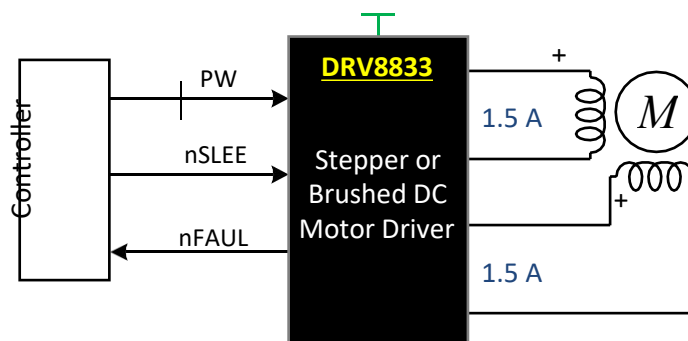
3.3 Knowledge of the Raspberry Pi

GPIO

GPIO: General Purpose Input/Output. Here we will introduce the specific function of the pins on the Raspberry Pi and how you can utilize them in all sorts of ways in your projects. Most RPi Module pins can be used as either an input or output, depending on your program and its functions.

When programming GPIO pins there are 3 different ways to reference them: GPIO Numbering, Physical Numbering and WiringPi GPIO Numbering.

BCM GPIO Numbering

The Raspberry Pi CPU uses Broadcom (BCM) processing chips BCM2835, BCM2836 or BCM2837. GPIO pin numbers are assigned by the processing chip manufacturer and are how the computer recognizes each pin. The pin numbers themselves do not make sense or have meaning as they are only a form of identification. Since their numeric values and physical locations have no specific order, there is no way to remember them so you will need to have a printed reference or a reference board that fits over the pins.
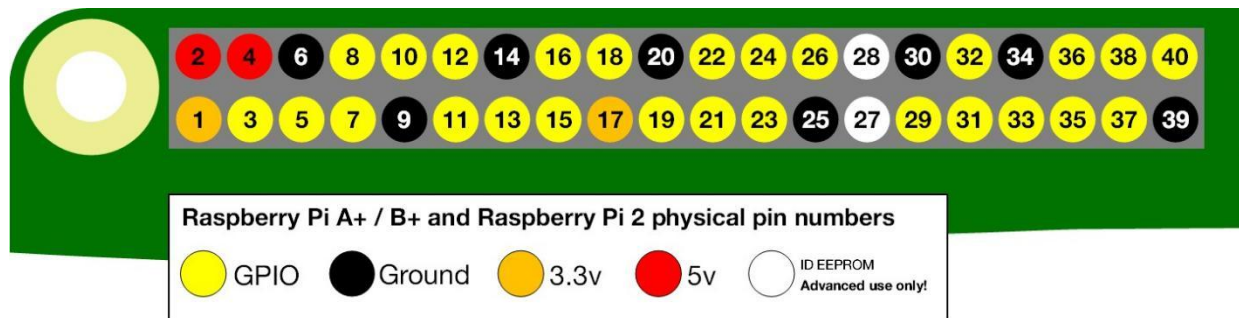
Each pin's functional assignment is defined in the image below:

Support : cokoino@outlook.com
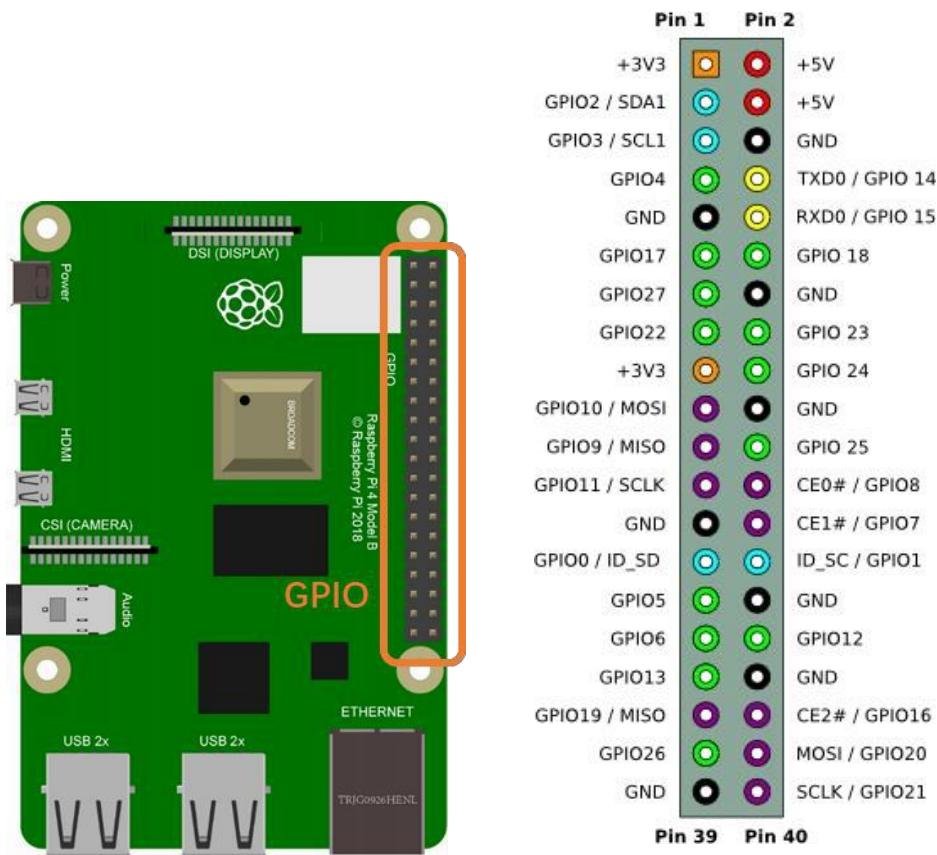
For more details about pin definition of GPIO, please refer to http://pinout.xyz/

PHYSICAL Numbering

Another way to refer to the pins is by simply counting across and down from pin 1 at the top left (nearest to the SD card). This is 'Physical Numbering', as shown below:

Raspberry Pi A+ / B+ and Raspberry Pi 2 physical pin numbers

⬤ GPIO ⬤ Ground ⬤ 3.3v ⬤ 5v ◯ ID EEPROM Advanced use only!

WiringPi GPIO Numbering

Different from the previous two types of GPIO serial numbers, RPi GPIO serial number of the WiringPi are numbered according to the BCM chip use in RPi.

wiringPi Pin	BCM GPIO	Name	Header	Name	BCM GPIO	wiringPi Pin		For A+, B+, 2B, 3B, 3B+, 4B, Zero
—	—	3.3v	1 | 2	5v	—	—		
8	R1:0/R2:2	SDA	3 | 4	5v	—	—		
9	R1:1/R2:3	SCL	5 | 6	0v	—	—		
7	4	GPIO7	7 | 8	TxD	14	15		
—	—	0v	9 | 10	RxD	15	16	For Pi B	
0	17	GPIO0	11 | 12	GPIO1	18	1		
2	R1:21/R2:27	GPIO2	13 | 14	0v	—	—		
3	22	GPIO3	15 | 16	GPIO4	23	4		
—	—	3.3v	17 | 18	GPIO5	24	5		
12	10	MOSI	19 | 20	0v	—	—		
13	9	MISO	21 | 22	GPIO6	25	6		
14	11	SCLK	23 | 24	CE0	8	10		
—	—	0v	25 | 26	CE1	7	11		
30	0	SDA.0	27 | 28	SCL.0	1	31		
21	5	GPIO.21	29 | 30	0V				
22	6	GPIO.22	31 | 32	GPIO.26	12	26		
23	13	GPIO.23	33 | 34	0V				
24	19	GPIO.24	35 | 36	GPIO.27	16	27		
25	26	GPIO.25	37 | 38	GPIO.28	20	28		
		0V	39 | 40	GPIO.29	21	29		
wiringPi Pin	BCM GPIO	Name	Header	Name	BCM GPIO	wiringPi Pin		

(For more details, please refer to https://projects.drogon.net/raspberry-pi/wiringpi/pins/)

Support : cokoino@outlook.com

You can also use the following command to view their correlation.

`gpio readall`

```
+-----+-----+---------+------+---+---Pi 4B--+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode |  Name   | wPi | BCM |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
|     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
|   2 |   8 |   SDA.1 | ALT0 | 1 |  3 || 4  |   |      | 5v      |     |     |
|   3 |   9 |   SCL.1 | ALT0 | 1 |  5 || 6  |   |      | 0v      |     |     |
|   4 |   7 |  GPIO. 7 |  IN | 1 |  7 || 8  | 0 |  IN  | TxD     | 15  | 14  |
|     |     |      0v |      |   |  9 || 10 | 1 |  IN  | RxD     | 16  | 15  |
|  17 |   0 |  GPIO. 0 |  IN | 0 | 11 || 12 | 0 |  IN  | GPIO. 1 | 1   | 18  |
|  27 |   2 |  GPIO. 2 |  IN | 0 | 13 || 14 |   |      | 0v      |     |     |
|  22 |   3 |  GPIO. 3 |  IN | 0 | 15 || 16 | 0 |  IN  | GPIO. 4 | 4   | 23  |
|     |     |    3.3v |      |   | 17 || 18 | 0 |  IN  | GPIO. 5 | 5   | 24  |
|  10 |  12 |    MOSI |  IN | 0 | 19 || 20 |   |      | 0v      |     |     |
|   9 |  13 |    MISO |  IN | 0 | 21 || 22 | 0 |  IN  | GPIO. 6 | 6   | 25  |
|  11 |  14 |    SCLK |  IN | 0 | 23 || 24 | 1 |  IN  | CE0     | 10  | 8   |
|     |     |      0v |      |   | 25 || 26 | 1 |  IN  | CE1     | 11  | 7   |
|   0 |  30 |   SDA.0 |  IN | 1 | 27 || 28 | 1 |  IN  | SCL.0   | 31  | 1   |
|   5 |  21 |  GPIO.21 |  IN | 1 | 29 || 30 |   |      | 0v      |     |     |
|   6 |  22 |  GPIO.22 |  IN | 1 | 31 || 32 | 0 |  IN  | GPIO.26 | 26  | 12  |
|  13 |  23 |  GPIO.23 |  IN | 0 | 33 || 34 |   |      | 0v      |     |     |
|  19 |  24 |  GPIO.24 |  IN | 0 | 35 || 36 | 0 |  IN  | GPIO.27 | 27  | 16  |
|  26 |  25 |  GPIO.25 |  IN | 0 | 37 || 38 | 0 |  IN  | GPIO.28 | 28  | 20  |
|     |     |      0v |      |   | 39 || 40 | 0 |  IN  | GPIO.29 | 29  | 21  |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode |  Name   | wPi | BCM |
+-----+-----+---------+------+---+---Pi 4B--+---+------+---------+-----+-----+
```

4. Circuit Connection

First, turn off power of the Pi Power & 4WD HAT,Then insert it onto the Raspberry Pi, paying attention to the assembly direction.Taking Raspberry Pi 4B as an example, assemble as shown in the following figure.


Support : cokoino@outlook.com
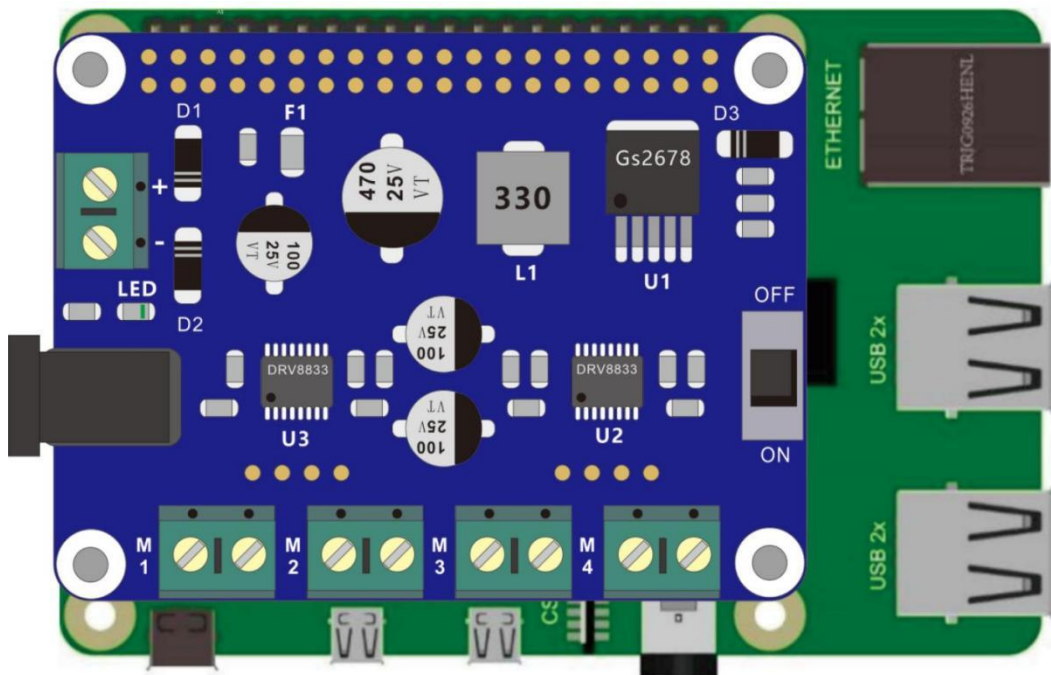
Then build the circuit according to the circuit connection diagrams. After the circuit is built and verified correct, turn on the power of Pi Power & 4WD HAT.

CAUTION: Avoid any possible short circuits (especially connecting 5V or GND, 3.3V and GND)!

WARNING: A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your RPi!

4.1 Circuit connection diagram:
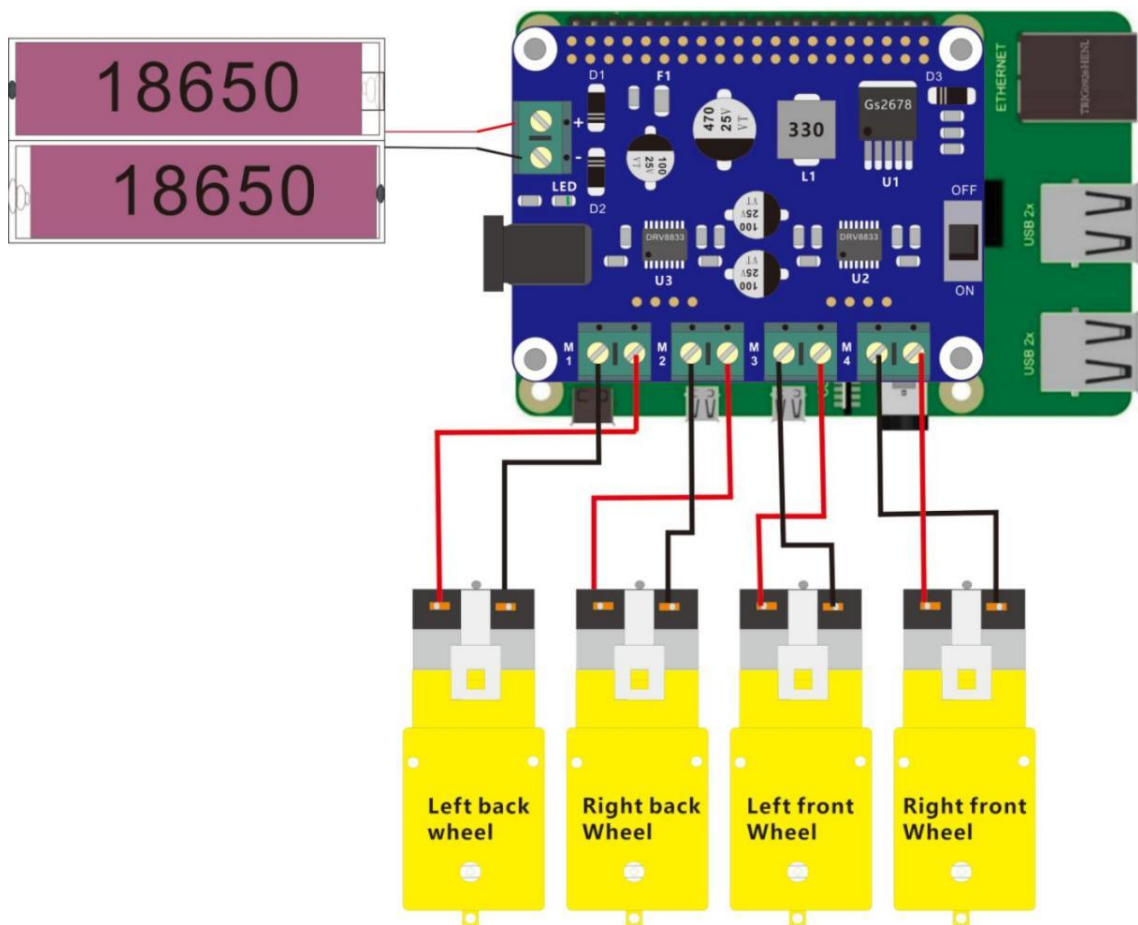
5. Download Code and Run

We recommend use Python code to achieve the function.

Please visit our GitHub resources at (**https://github.com/cokoino**) to download the latest available project code. We provide **Python** language code for this project .

This is the method for obtaining the code:

```
cd
git clone --depth 1 https://github.com/cokoino/CKK0011
```
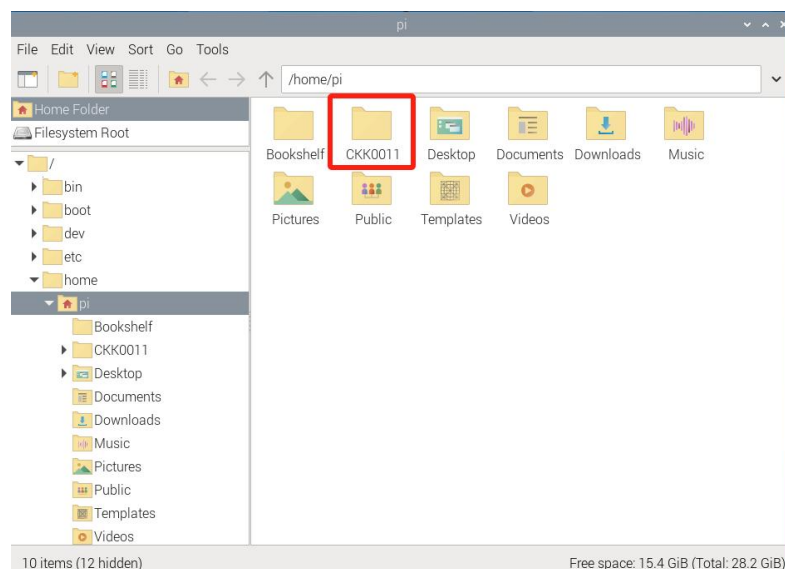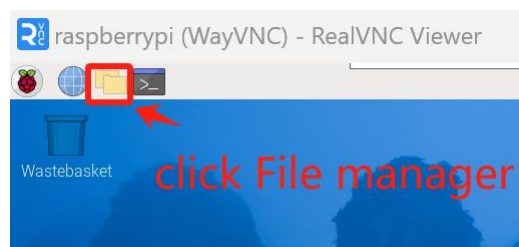
In the pi directory of the RPi terminal, enter the following command.

(There is no need for a password. If you get some errors, please check your commands.)
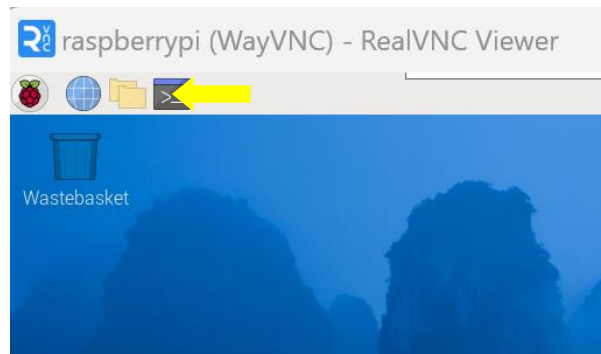
After the download is completed, a new folder "CKK0011" is generated, which contains all of the tutorials and required code.

Click File manager,you will find the folder "CKK0011"

If you have no experience with Python, we suggest that you refer to this website for basic information and knowledge.
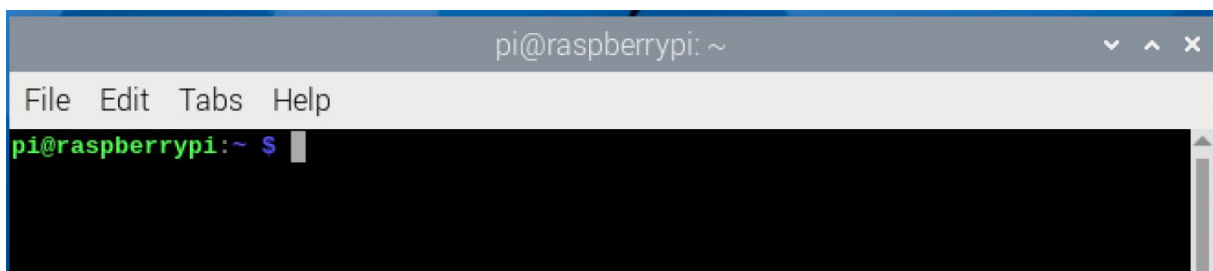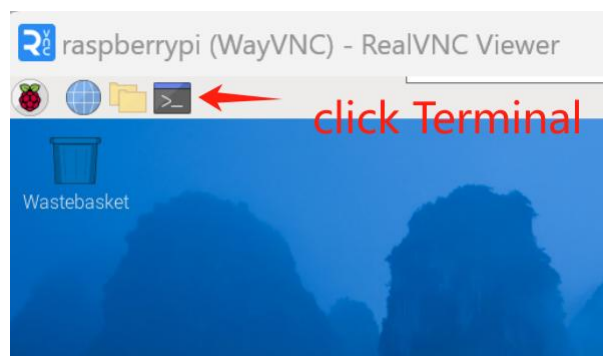
https://python.swaroopch.com/basics.html

Then,log in to Raspberry Pi by using VNC Viewer and operated proper setting.(For those who have no impression or are unclear about the operation, please refer to the PDF document "2 Installing and

Support : cokoino@outlook.com

Configuring Raspberry Pi System").
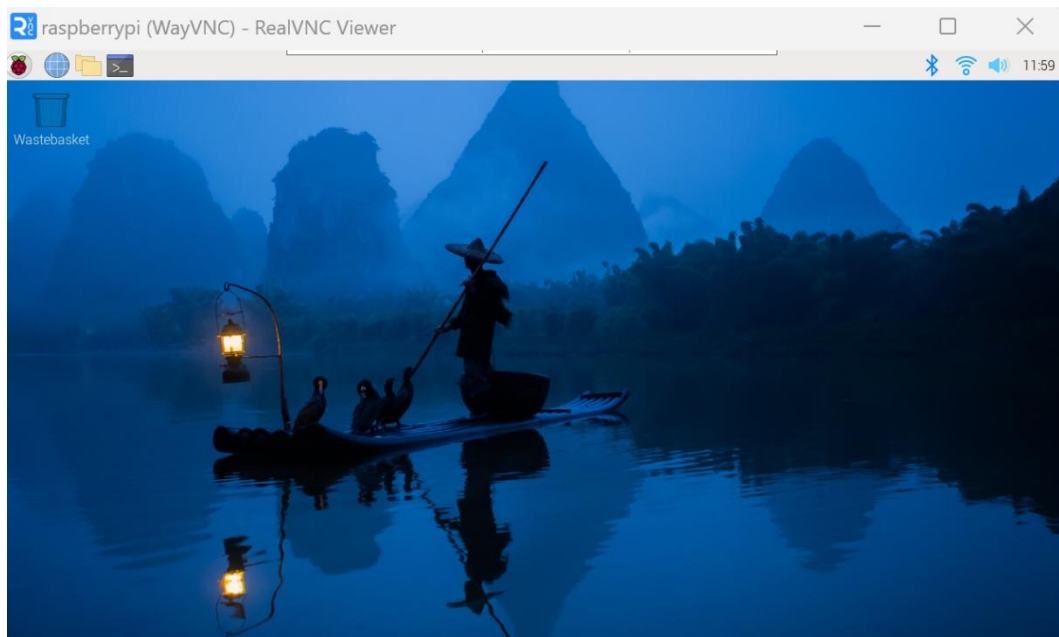
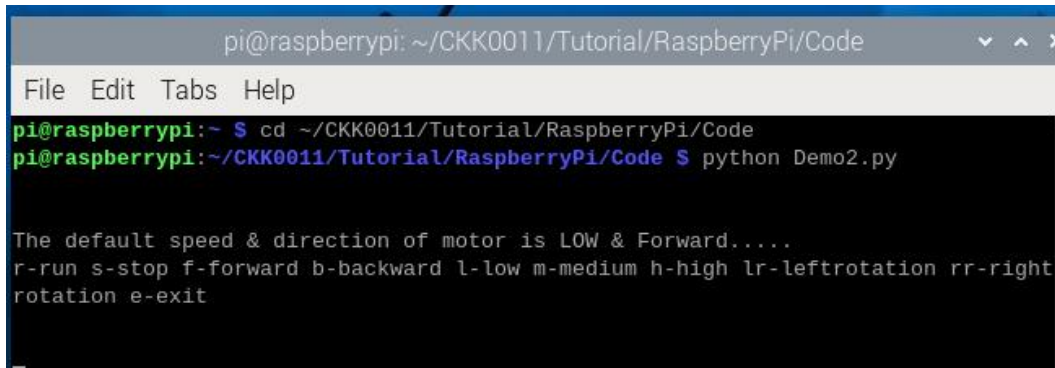Click "Terminal", following interface appears.

Use cd command to enter Demo1 directory of Python code.

```
cd  ~/CKK0011/Tutorial/RaspberryPi/Code
```

Use python command to execute python code Demo2.py.

```
python Demo2.py
```

Then the interface displays "The default direction of motor is Forward......r-run s-stop f-forward b-backward lr-leftrotation rr-rightrotation e-exit"
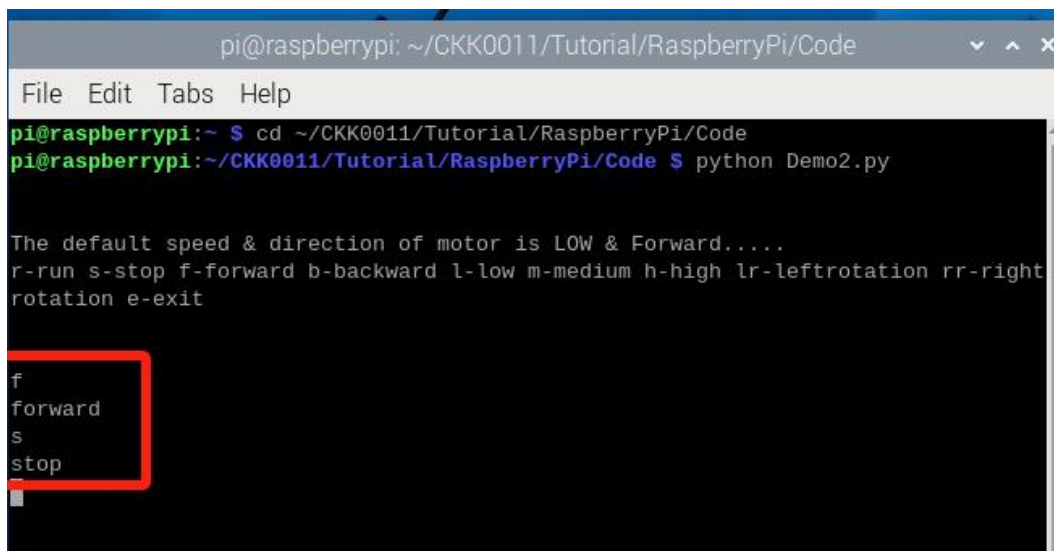
Finally, Enter the command according to the prompt information and press enter to observe whether the car will perform the corresponding action. For example, after entering "f" and press enter, the car starts to move forward; After entering "s"and press enter, the car stops moving.

At the same time, the remote desktop interface has corresponding information display.

You can enter the "e" command or press "Ctrl+C" to end the program.

The following is the program code:

```
1.    # Python Script
2.    # https://www.electronicshub.org/raspberry-pi-l298n-interface-tutorial-control-dc-motor-l298n-raspberry-pi/
3.
4.    import RPi.GPIO as GPIO
5.    from time import sleep
6.
7.    NSLEEP1 = 12
8.    AN11 = 17
9.    AN12 = 27
10.   BN11 = 22
11.   BN12 = 23
12.   NSLEEP2 = 13
13.   AN21 = 24
```

Support : cokoino@outlook.com

```python
14.     AN22 = 25
15.     BN21 = 26
16.     BN22 = 16
17.     temp1=1
18.
19.     GPIO.setmode(GPIO.BCM)
20.     GPIO.setup(NSLEEP1,GPIO.OUT)
21.     GPIO.setup(NSLEEP2,GPIO.OUT)
22.     GPIO.setup(AN11,GPIO.OUT)
23.     GPIO.setup(AN12,GPIO.OUT)
24.     GPIO.setup(BN11,GPIO.OUT)
25.     GPIO.setup(BN12,GPIO.OUT)
26.     GPIO.setup(AN21,GPIO.OUT)
27.     GPIO.setup(AN22,GPIO.OUT)
28.     GPIO.setup(BN21,GPIO.OUT)
29.     GPIO.setup(BN22,GPIO.OUT)
30.     GPIO.output(AN11,GPIO.LOW)
31.     GPIO.output(AN12,GPIO.LOW)
32.     GPIO.output(BN11,GPIO.LOW)
33.     GPIO.output(BN12,GPIO.LOW)
34.     GPIO.output(AN21,GPIO.LOW)
35.     GPIO.output(AN22,GPIO.LOW)
36.     GPIO.output(BN21,GPIO.LOW)
37.     GPIO.output(BN22,GPIO.LOW)
38.     p1=GPIO.PWM(NSLEEP1,1000)
39.     p2=GPIO.PWM(NSLEEP2,1000)
40.     p1.start(25)
41.     p2.start(25)
42.
43.     print("\n")
44.     print("The default speed & direction of motor is LOW & Forward.....")
45.     print("r-run s-stop f-forward b-backward l-low m-medium h-high lr-leftrotation rr-rightrotation e-exit")
46.     print("\n")
47.
48.     while(1):
49.
50.         x=input()
51.
52.         if x=='r':
53.             print("run")
54.             if(temp1==1):
55.                 GPIO.output(AN11,GPIO.HIGH)
56.                 GPIO.output(AN12,GPIO.LOW)
57.                 GPIO.output(BN11,GPIO.HIGH)
58.                 GPIO.output(BN12,GPIO.LOW)
59.                 GPIO.output(AN21,GPIO.HIGH)
60.                 GPIO.output(AN22,GPIO.LOW)
61.                 GPIO.output(BN21,GPIO.HIGH)
62.                 GPIO.output(BN22,GPIO.LOW)
63.                 print("backward")
64.                 x='z'
65.             else:
66.                 GPIO.output(AN11,GPIO.LOW)
67.                 GPIO.output(AN12,GPIO.HIGH)
68.                 GPIO.output(BN11,GPIO.LOW)
69.                 GPIO.output(BN12,GPIO.HIGH)
70.                 GPIO.output(AN21,GPIO.LOW)
71.                 GPIO.output(AN22,GPIO.HIGH)
```

```python
72.            GPIO.output(BN21,GPIO.LOW)
73.            GPIO.output(BN22,GPIO.HIGH)
74.            print("forward")
75.            x='z'
76.
77.
78.        elif x=='s':
79.            print("stop")
80.            GPIO.output(AN11,GPIO.LOW)
81.            GPIO.output(AN12,GPIO.LOW)
82.            GPIO.output(BN11,GPIO.LOW)
83.            GPIO.output(BN12,GPIO.LOW)
84.            GPIO.output(AN21,GPIO.LOW)
85.            GPIO.output(AN22,GPIO.LOW)
86.            GPIO.output(BN21,GPIO.LOW)
87.            GPIO.output(BN22,GPIO.LOW)
88.            x='z'
89.
90.        elif x=='f':
91.            print("forward")
92.            GPIO.output(AN11,GPIO.LOW)
93.            GPIO.output(AN12,GPIO.HIGH)
94.            GPIO.output(BN11,GPIO.LOW)
95.            GPIO.output(BN12,GPIO.HIGH)
96.            GPIO.output(AN21,GPIO.LOW)
97.            GPIO.output(AN22,GPIO.HIGH)
98.            GPIO.output(BN21,GPIO.LOW)
99.            GPIO.output(BN22,GPIO.HIGH)
100.           temp1=0
101.           x='z'
102.
103.       elif x=='b':
104.           print("backward")
105.           GPIO.output(AN11,GPIO.HIGH)
106.           GPIO.output(AN12,GPIO.LOW)
107.           GPIO.output(BN11,GPIO.HIGH)
108.           GPIO.output(BN12,GPIO.LOW)
109.           GPIO.output(AN21,GPIO.HIGH)
110.           GPIO.output(AN22,GPIO.LOW)
111.           GPIO.output(BN21,GPIO.HIGH)
112.           GPIO.output(BN22,GPIO.LOW)
113.           temp1=1
114.           x='z'
115.
116.       elif x=='lr':
117.           print("leftrotation")
118.           GPIO.output(AN11,GPIO.HIGH)
119.           GPIO.output(AN12,GPIO.LOW)
120.           GPIO.output(BN11,GPIO.LOW)
121.           GPIO.output(BN12,GPIO.HIGH)
122.           GPIO.output(AN21,GPIO.HIGH)
123.           GPIO.output(AN22,GPIO.LOW)
124.           GPIO.output(BN21,GPIO.LOW)
125.           GPIO.output(BN22,GPIO.HIGH)
126.           temp1=0
127.           x='z'
128.
129.       elif x=='rr':
```

```python
130.        print("rightrotation")
131.        GPIO.output(AN11,GPIO.LOW)
132.        GPIO.output(AN12,GPIO.HIGH)
133.        GPIO.output(BN11,GPIO.HIGH)
134.        GPIO.output(BN12,GPIO.LOW)
135.        GPIO.output(AN21,GPIO.LOW)
136.        GPIO.output(AN22,GPIO.HIGH)
137.        GPIO.output(BN21,GPIO.HIGH)
138.        GPIO.output(BN22,GPIO.LOW)
139.        temp1=0
140.        x='z'
141.
142.    elif x=='l':
143.        print("low")
144.        p1.ChangeDutyCycle(25)
145.        p2.ChangeDutyCycle(25)
146.        x='z'
147.    elif x=='m':
148.        print("medium")
149.        p1.ChangeDutyCycle(50)
150.        p2.ChangeDutyCycle(50)
151.        x='z'
152.    elif x=='h':
153.        print("high")
154.        p1.ChangeDutyCycle(75)
155.        p2.ChangeDutyCycle(75)
156.        x='z'
157.    elif x=='e':
158.        GPIO.cleanup()
159.        print("GPIO Clean up")
160.        break
161.
162.    else:
163.        print("<<<  wrong data  >>>")
164.        print("please enter the defined data to continue.....")
```

About RPi.GPIO：

RPi.GPIO
This is a Python module to control the GPIO on a Raspberry Pi. It includes basic output function and input function of GPIO, and functions used to generate PWM.
GPIO.setmode(mode)
Sets the mode for pin serial number of GPIO. mode=GPIO.BOARD, which represents the GPIO pin serial number based on physical location of RPi. mode=GPIO.BCM, which represents the pin serial number based on CPU of BCM chip.
GPIO.setup(pin,mode)
Sets pin to input mode or output mode, "pin" for the GPIO pin, "mode" for INPUT or OUTPUT.
GPIO.output(pin,mode)
Sets pin to output mode, "pin" for the GPIO pin, "mode" for HIGH (high level) or LOW (low level).

For more functions related to RPi.GPIO, please refer to:

https://sourceforge.net/p/raspberry-gpio-python/wiki/Examples/

Support : cokoino@outlook.com

"import time" time is a module of python.

https://docs.python.org/2/library/time.html?highlight=time%20time#module-time

In subfunction setup(), GPIO.setmode (GPIO.BOARD) is used to set the serial number for GPIO based on physical location of the pin.

GPIO Numbering Relationship

WingPi	BCM(Extension)	Physical		BCM(Extension)	WingPi
3.3V	3.3V	1	2	5V	5V
8	GPIO2/SDA1	3	4	5V	5V
9	GPIO3/SCL1	5	6	GND	GND
7	GPIO4	7	8	GPIO14/TXD0	15
GND	GND	9	10	GPIO15/RXD0	16
0	GPIO17	11	12	GPIO18	1
2	GPIO27	13	14	GND	GND
3	GPIO22	15	16	GPIO23	4
3.3V	3.3V	17	18	GPIO24	5
12	GPIO10/MOSI)	19	20	GND	GND
13	GPIO9/MOIS	21	22	GPIO25	6
14	GPIO11/SCLK	23	24	GPIO8/CE0	10
GND	GND	25	26	GPIO7/CE1	11
30	GPIO0/SDA0	27	28	GPIO1/SCL0	31
21	GPIO5	29	30	GND	GND
22	GPIO6	31	32	GPIO12	26
23	GPIO13	33	34	GND	GND
24	GPIO19	35	36	GPIO16	27
25	GPIO26	37	38	GPIO20	28
GND	GND	39	40	GPIO21	29

6. Trouble shooting

Question 1: If the car does not move according to the code instructions, such as the command being forward, but the actual action of the car is backward or other actions.

Answer: Please check if the motors on the four wheels of the car are connected to the corresponding positions of the Cokoino 4WD Robot HAT. If not, the car will not perform the corresponding actions according to the instructions.

Question2: Using RealVNC Viewer as a remote desktop connection does not connect to Raspberry Pi, and Raspberry Pi does not display when connected to the monitor.

Answer: Please check the battery level that supplies power to the Cokoino 4WD Robot HAT. When the battery voltage is below 6.8V, the Raspberry Pi cannot be powered on. Please fully charge the battery before use.

Support : cokoino@outlook.com

7. Any questions and suggestions are welcome

THANK YOU for participating in this Demo2 experience!

If you find any errors, omissions or you have suggestions and/or questions about this lesson, please feel free to contact us:

cokoino@outlook.com

We will make every effort to make changes and correct errors as soon as feasibly possible and publish a revised version.

If you want to learn more about Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our Amazon Store by search for "**LK COKOINO**" on Amazon. We will continue to launch fun, cost-effective, innovative and exciting products.

Thank you again for choosing Cokoino products.

LK COKOINO

Support : cokoino@outlook.com