
Drive Cokoino 4WD Robot Hat to run the 4WD car

Table

1. Preface	2
2. Component List	2
3. Circuit connection	3
3.1 Circuit connection diagram:	4
4. Download Code and Run	4
5. Trouble shooting	12
6. Any questions and suggestions are welcome	12

1. Preface

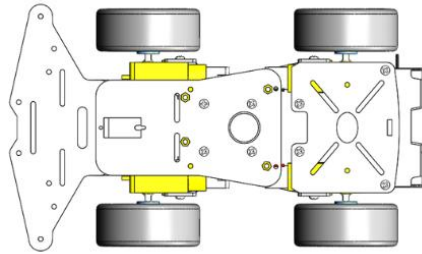
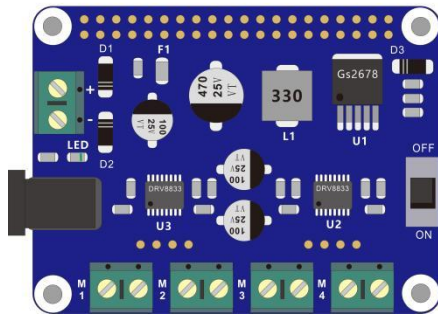
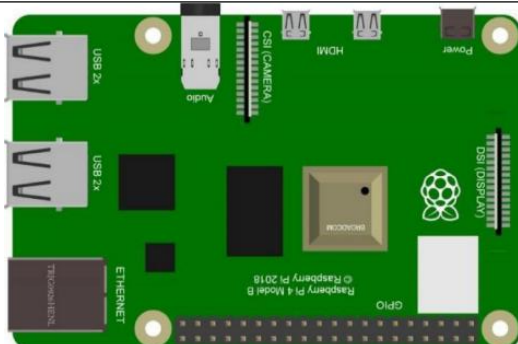
4WD Robot HAT can be paired with Raspberry Pi and 4WD car chassis to assemble a 4WD car. If you already have a small car chassis, that would be really convenient. You can conduct the experiment immediately. If you don't have a car chassis, you can go to Amazon to purchase a car chassis that you like. You can also purchase Cokoino car chassis on Amazon, search for "Cokoino" on Amazon, enter the Cokoino store, and select the car chassis for purchase.

This demo experiment uses the chassis of 4WD Robot HAT, Raspberry Pi 4B, and Cokoino car chassis to assemble a 4WD car, demonstrating the four-wheel drive function of 4WD Robot HAT.

No matter which chassis you use, you can refer to the demo code to run your experiment.

2. Component List

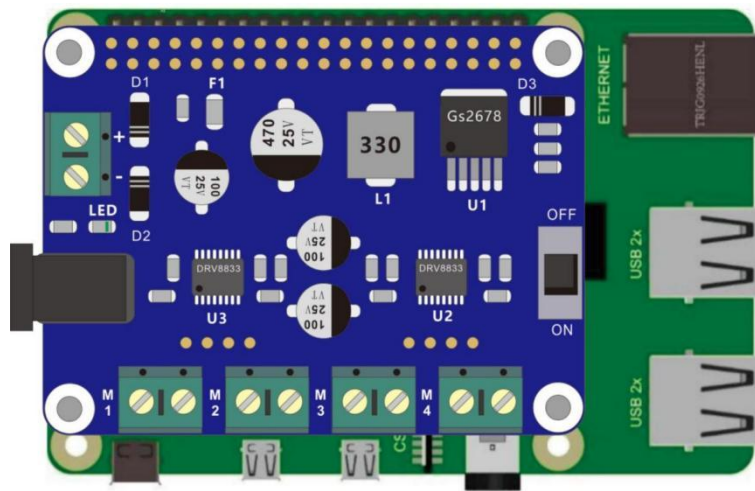
For this Demo experiment, what do you need to prepare like below list

Component/Module	QTY	Picture	Remark
4WD Car Chassis Kit	1		Just for demonstration purposes, currently your product does not include it. If you like it, you can buy it from Cokoino's Amazon store
4WD Robot HAT	1		
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.

18650 battery	2		
---------------	---	--	--

3. Circuit connection

Firstly, refer to the assembly tutorial of the cokoino 4WD car chassis kit to assemble the chassis of the 4WD car. Then, fix the Raspberry Pi 4B onto the M3 * 30mm copper column of the 4WD car chassis with screws. turn off power of the 4WD Robot 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.



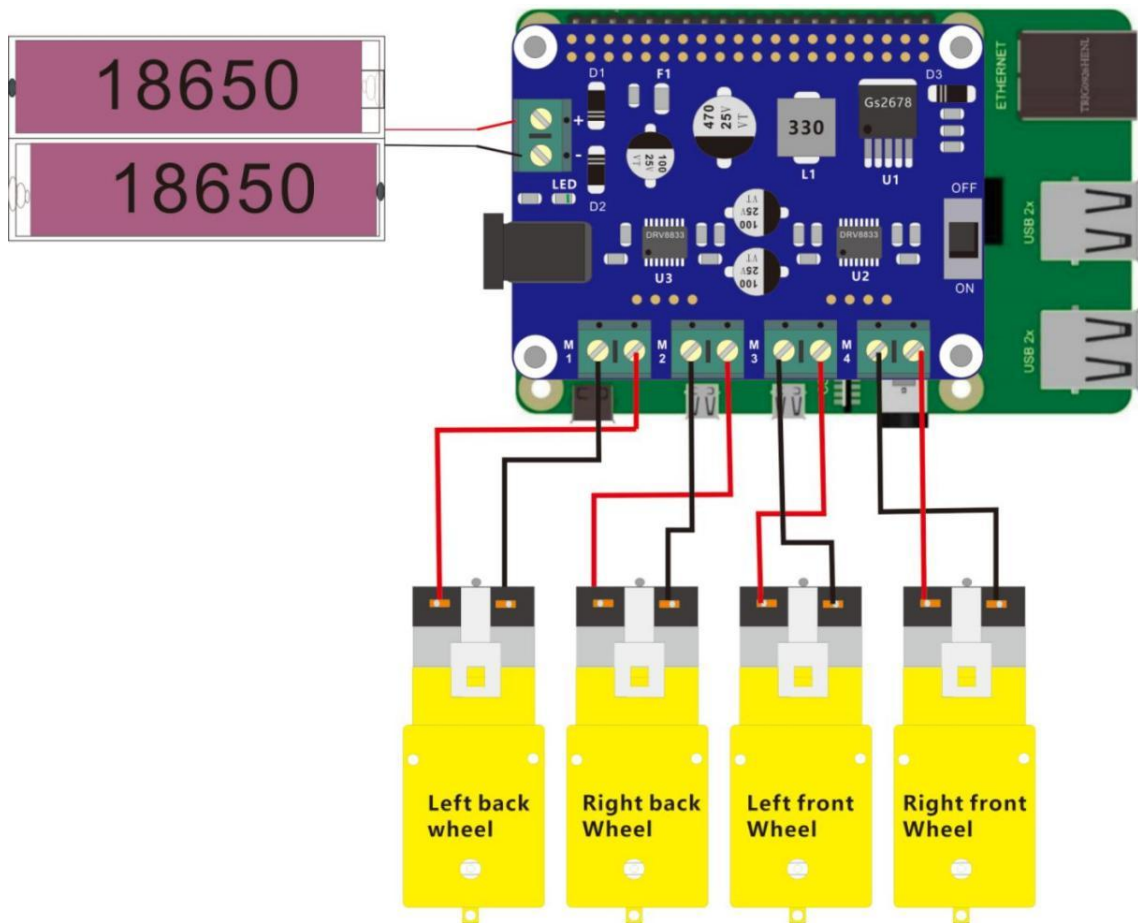
Then connect the red and black wires of the battery box to the 2-pin power terminal on the 4WD Robot HAT, and tighten the terminal with a screwdriver. Finally, connect the red and black wires of the chassis motor of the 4WD car to the motor interface terminals on the 4WD Robot HAT, and tighten the terminals with a screwdriver. Definition: M1 is connected to the motor of the left rear wheel, M2 is connected to the motor of the right rear wheel, M3 is connected to the motor of the left front wheel, and M4 is connected to the motor of the right front wheel.

Then build the circuit according to the circuit connection diagrams. After the circuit is built and verified correct, turn on the power of 4WD Robot 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!

3.1 Circuit connection diagram:



4. Download Code and Run

We recommend use Python code to achieve the function.

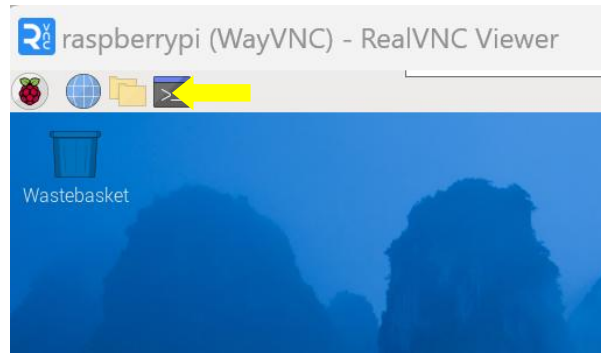
Please visit our GitHub resources at (<https://github.com/cokoino>) to download the latest available project code. We provide **Python** language code for this project .

This is the method for obtaining the code:

```
cd
git clone --depth 1 https://github.com/cokoino/CKK0018
```

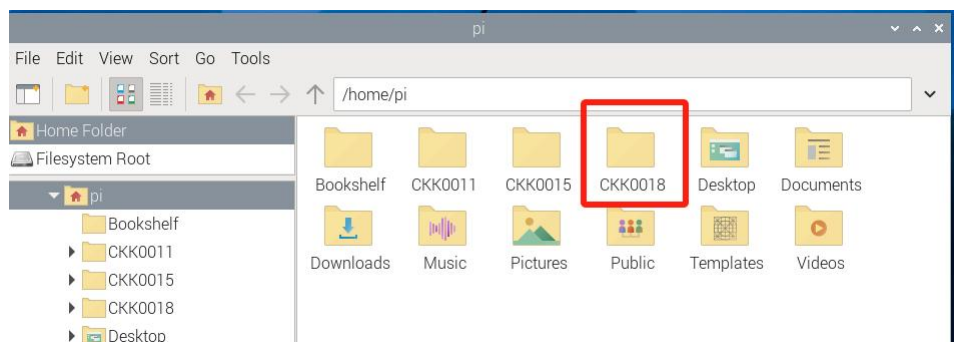
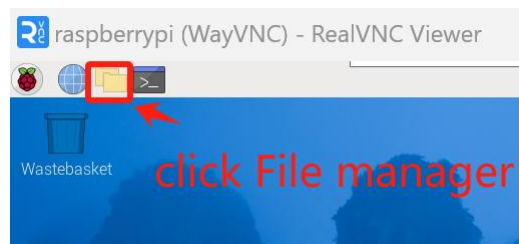
In the pi directory of the RPi terminal, enter the following command.

(There is no need for a password. If you get some errors, please check your commands.)



After the download is completed, a new folder "CKK0018" is generated, which contains all of the tutorials and required code.

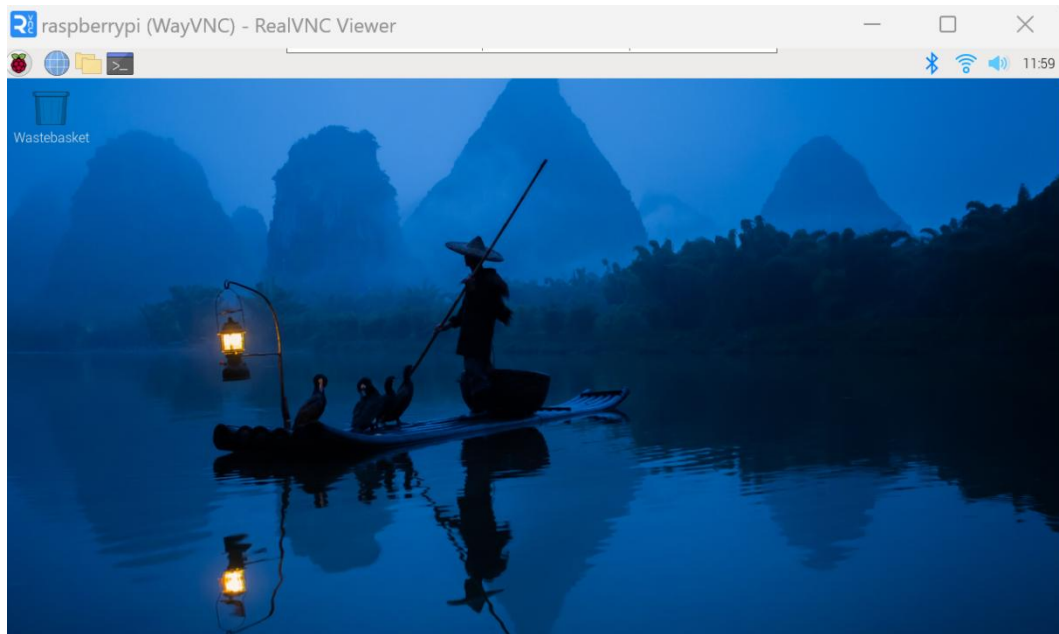
Click File manager, you will find the folder "CKK0018"



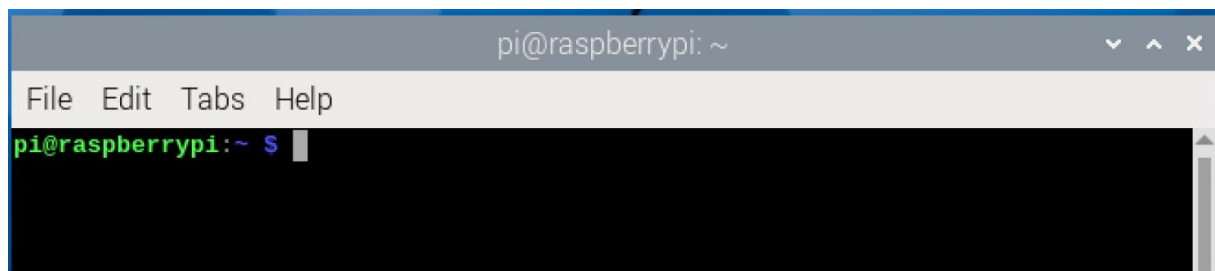
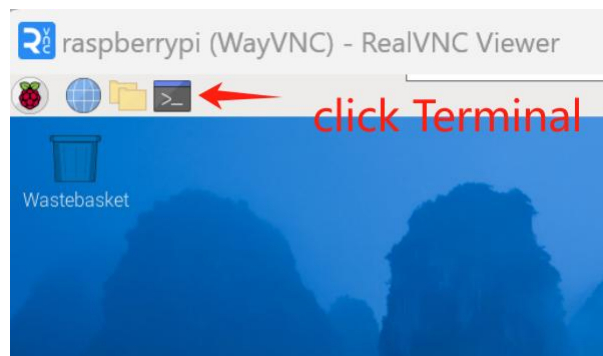
If you have no experience with Python, we suggest that you refer to this website for basic information and knowledge.

<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 Configuring Raspberry Pi System").



Click “Terminal”, following interface appears.



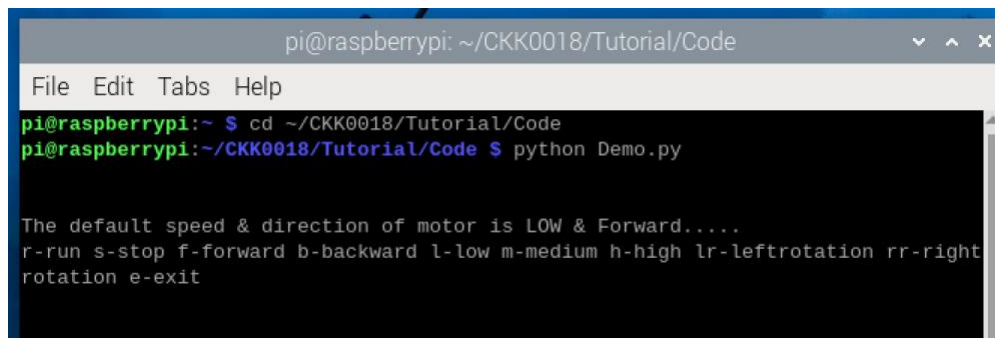
Use cd command to enter Demo1 directory of Python code.

```
cd ~/CKK0018/Tutorial/Code
```

Use python command to execute python code Demo1.py.

```
python Demo.py
```

Then the interface displays “ The default speed & direction of motor is LOW & Forward.....r-run s-stop f-forward b-backward l-low m-medium h-high lr-leftrotation rr-rightrotation e-exit”

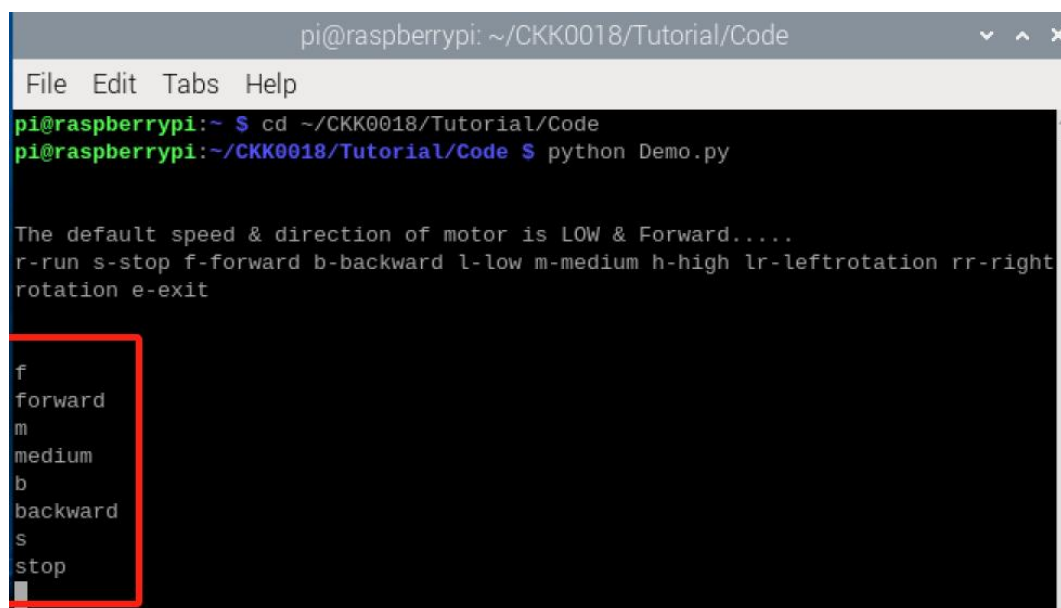


```
pi@raspberrypi: ~/CKK0018/Tutorial/Code
File Edit Tabs Help
pi@raspberrypi:~ $ cd ~/CKK0018/Tutorial/Code
pi@raspberrypi:~/CKK0018/Tutorial/Code $ python Demo.py

The default speed & direction of motor is LOW & Forward.....
r-run s-stop f-forward b-backward l-low m-medium h-high lr-leftrotation rr-right
rotation 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 with the low speed; After entering "m"and press enter, the motors rotate speed change to medium;After entering "b"and press enter,the car starts to move backward;After entering "s"and press enter,the motors stop rotating.

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



```
pi@raspberrypi: ~/CKK0018/Tutorial/Code
File Edit Tabs Help
pi@raspberrypi:~ $ cd ~/CKK0018/Tutorial/Code
pi@raspberrypi:~/CKK0018/Tutorial/Code $ python Demo.py

The default speed & direction of motor is LOW & Forward.....
r-run s-stop f-forward b-backward l-low m-medium h-high lr-leftrotation rr-right
rotation e-exit

f
forward
m
medium
b
backward
s
stop
```

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. #define the pin for drv8833#1
7. NSLEEP1 = 12 #Enabling signal pin for drv8833


```

8.  AN11 = 17
9.  AN12 = 27
10. BN11 = 22
11. BN12 = 23
12. #define the pin for drv8833#2
13. NSLEEP2 = 13 #Enabling signal pin for drv8833
14. AN21 = 24
15. AN22 = 25
16. BN21 = 26
17. BN22 = 16
18. temp1=1
19.
20. GPIO.setmode(GPIO.BCM)
21. #Define pin as output signal
22. GPIO.setup(NSLEEP1,GPIO.OUT)
23. GPIO.setup(NSLEEP2,GPIO.OUT)
24. GPIO.setup(AN11,GPIO.OUT)
25. GPIO.setup(AN12,GPIO.OUT)
26. GPIO.setup(BN11,GPIO.OUT)
27. GPIO.setup(BN12,GPIO.OUT)
28. GPIO.setup(AN21,GPIO.OUT)
29. GPIO.setup(AN22,GPIO.OUT)
30. GPIO.setup(BN21,GPIO.OUT)
31. GPIO.setup(BN22,GPIO.OUT)
32. #Initialize the motor drive signal so that the motor is in a stopped state
33. GPIO.output(AN11,GPIO.LOW)
34. GPIO.output(AN12,GPIO.LOW)
35. GPIO.output(BN11,GPIO.LOW)
36. GPIO.output(BN12,GPIO.LOW)
37. GPIO.output(AN21,GPIO.LOW)
38. GPIO.output(AN22,GPIO.LOW)
39. GPIO.output(BN21,GPIO.LOW)
40. GPIO.output(BN22,GPIO.LOW)
41. p1=GPIO.PWM(NSLEEP1,1000)#Define p1 as a pulse signal of 1000 Hz
42. p2=GPIO.PWM(NSLEEP2,1000)#Define p2 as a pulse signal of 1000 Hz
43. p1.start(30)#P1 defaults to a duty cycle of 30%
44. p2.start(30)#P2 defaults to a duty cycle of 30%
45.
46. print("\n")
47. print("The default speed & direction of motor is LOW & Forward.....")
48. print("r-run s-stop f-forward b-backward l-low m-medium h-high lr-leftrotation rr-rightrotation e-exit")
49. print("\n")
50.
51. while(1):
52.
53.     x=input()
54.
55.     if x=='r':
56.         print("run")
57.         if(temp1==1):
58.             GPIO.output(AN11,GPIO.HIGH)
59.             GPIO.output(AN12,GPIO.LOW)

```



```

60.     GPIO.output(BN11,GPIO.HIGH)
61.     GPIO.output(BN12,GPIO.LOW)
62.     GPIO.output(AN21,GPIO.HIGH)
63.     GPIO.output(AN22,GPIO.LOW)
64.     GPIO.output(BN21,GPIO.HIGH)
65.     GPIO.output(BN22,GPIO.LOW)
66.     print("backward")
67.     x='z'
68.     else:
69.         GPIO.output(AN11,GPIO.LOW)
70.         GPIO.output(AN12,GPIO.HIGH)
71.         GPIO.output(BN11,GPIO.LOW)
72.         GPIO.output(BN12,GPIO.HIGH)
73.         GPIO.output(AN21,GPIO.LOW)
74.         GPIO.output(AN22,GPIO.HIGH)
75.         GPIO.output(BN21,GPIO.LOW)
76.         GPIO.output(BN22,GPIO.HIGH)
77.         print("forward")
78.         x='z'
79.
80.
81.     elif x=='s':
82.         print("stop")
83.         GPIO.output(AN11,GPIO.LOW)
84.         GPIO.output(AN12,GPIO.LOW)
85.         GPIO.output(BN11,GPIO.LOW)
86.         GPIO.output(BN12,GPIO.LOW)
87.         GPIO.output(AN21,GPIO.LOW)
88.         GPIO.output(AN22,GPIO.LOW)
89.         GPIO.output(BN21,GPIO.LOW)
90.         GPIO.output(BN22,GPIO.LOW)
91.         x='z'
92.
93.     elif x=='f':
94.         print("forward")
95.         GPIO.output(AN11,GPIO.LOW)
96.         GPIO.output(AN12,GPIO.HIGH)
97.         GPIO.output(BN11,GPIO.LOW)
98.         GPIO.output(BN12,GPIO.HIGH)
99.         GPIO.output(AN21,GPIO.LOW)
100.        GPIO.output(AN22,GPIO.HIGH)
101.        GPIO.output(BN21,GPIO.LOW)
102.        GPIO.output(BN22,GPIO.HIGH)
103.        temp1=0
104.        x='z'
105.
106.    elif x=='b':
107.        print("backward")
108.        GPIO.output(AN11,GPIO.HIGH)
109.        GPIO.output(AN12,GPIO.LOW)
110.        GPIO.output(BN11,GPIO.HIGH)
111.        GPIO.output(BN12,GPIO.LOW)

```

```

112.     GPIO.output(AN21,GPIO.HIGH)
113.     GPIO.output(AN22,GPIO.LOW)
114.     GPIO.output(BN21,GPIO.HIGH)
115.     GPIO.output(BN22,GPIO.LOW)
116.     temp1=1
117.     x='z'
118.
119.     elif x=='lr':
120.         print("leftrotation")
121.         GPIO.output(AN11,GPIO.HIGH)
122.         GPIO.output(AN12,GPIO.LOW)
123.         GPIO.output(BN11,GPIO.LOW)
124.         GPIO.output(BN12,GPIO.HIGH)
125.         GPIO.output(AN21,GPIO.HIGH)
126.         GPIO.output(AN22,GPIO.LOW)
127.         GPIO.output(BN21,GPIO.LOW)
128.         GPIO.output(BN22,GPIO.HIGH)
129.         temp1=0
130.         x='z'
131.
132.     elif x=='rr':
133.         print("rightrotation")
134.         GPIO.output(AN11,GPIO.LOW)
135.         GPIO.output(AN12,GPIO.HIGH)
136.         GPIO.output(BN11,GPIO.HIGH)
137.         GPIO.output(BN12,GPIO.LOW)
138.         GPIO.output(AN21,GPIO.LOW)
139.         GPIO.output(AN22,GPIO.HIGH)
140.         GPIO.output(BN21,GPIO.HIGH)
141.         GPIO.output(BN22,GPIO.LOW)
142.         temp1=0
143.         x='z'
144.
145.     elif x=='l':
146.         print("low")
147.         p1.ChangeDutyCycle(30)#Set the P1 pulse signal duty cycle to 30%
148.         p2.ChangeDutyCycle(30)#Set the P2 pulse signal duty cycle to 30%
149.         x='z'
150.     elif x=='m':
151.         print("medium")
152.         p1.ChangeDutyCycle(60)#Set the P1 pulse signal duty cycle to 60%
153.         p2.ChangeDutyCycle(60)#Set the P2 pulse signal duty cycle to 60%
154.         x='z'
155.     elif x=='h':
156.         print("high")
157.         p1.ChangeDutyCycle(90)#Set the P1 pulse signal duty cycle to 90%
158.         p2.ChangeDutyCycle(90)#Set the P2 pulse signal duty cycle to 90%
159.         x='z'
160.     elif x=='e':
161.         GPIO.cleanup()
162.         print("GPIO Clean up")
163.         break

```

```

164.
165.     else:
166.         print("<<< wrong data >>>")
167.         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/>

“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

5. 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.

Question3: If the car does not move after sending the command, but the buzzing sound of the motor can be heard.

Answer: Because the default low speed set by the program is 30% duty cycle, it may not support the rotational power of your motor. You can change the 30 in the program to 40 or higher to adapt to your motor.

6. 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