Welcome

Thank you for choosing Freenove products!

Get Support and Offer Input

Freenove provides free and responsive product and technical support, including but not limited to:

- Product quality issues
- Product use and build issues
- Questions regarding the technology employed in our products for learning and education
- Your input and opinions are always welcome
- We also encourage your ideas and suggestions for new products and product improvements

For any of the above, you may send us an email to:

support@freenove.com

Safety and Precautions

Please follow the following safety precautions when using or storing this product:

- Keep this product out of the reach of children under 6 years old.
- This product should be used only when there is adult supervision present as young children lack necessary judgment regarding safety and the consequences of product misuse.
- This product contains small parts and parts, which are sharp. This product contains electrically conductive parts. Use caution with electrically conductive parts near or around power supplies, batteries and powered (live) circuits.
- When the product is turned ON, activated or tested, some parts will move or rotate. To avoid injuries to hands and fingers, keep them away from any moving parts!
- It is possible that an improperly connected or shorted circuit may cause overheating. Should this happen, immediately disconnect the power supply or remove the batteries and do not touch anything until it cools down! When everything is safe and cool, review the product tutorial to identify the cause.
- Only operate the product in accordance with the instructions and guidelines of this tutorial, otherwise parts may be damaged or you could be injured.
- Store the product in a cool dry place and avoid exposing the product to direct sunlight.
- After use, always turn the power OFF and remove or unplug the batteries before storing.

About Freenove

Freenove provides open source electronic products and services worldwide.

Freenove is committed to assist customers in their education of robotics, programming and electronic circuits so that they may transform their creative ideas into prototypes and new and innovative products. To this end, our services include but are not limited to:

- Educational and Entertaining Project Kits for Robots, Smart Cars and Drones
- Educational Kits to Learn Robotic Software Systems for Arduino, Raspberry Pi and micro: bit
- Electronic Component Assortments, Electronic Modules and Specialized Tools
- **Product Development and Customization Services**

You can find more about Freenove and get our latest news and updates through our website:

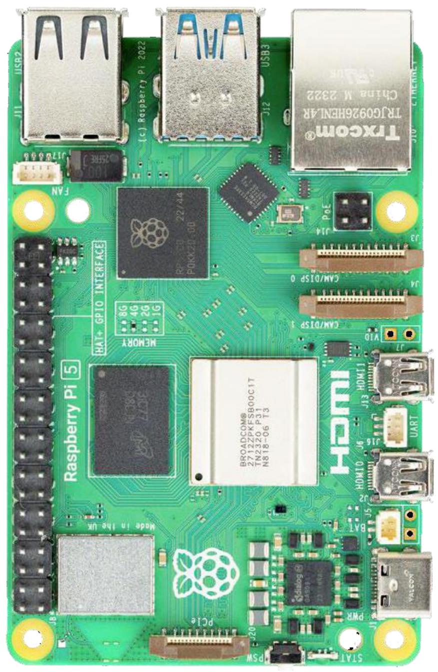
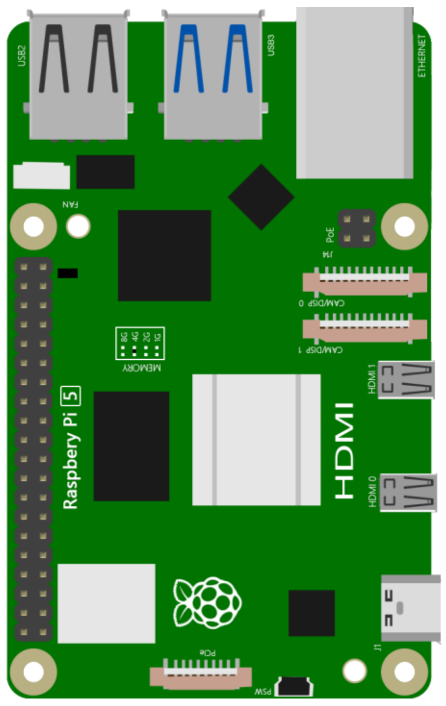
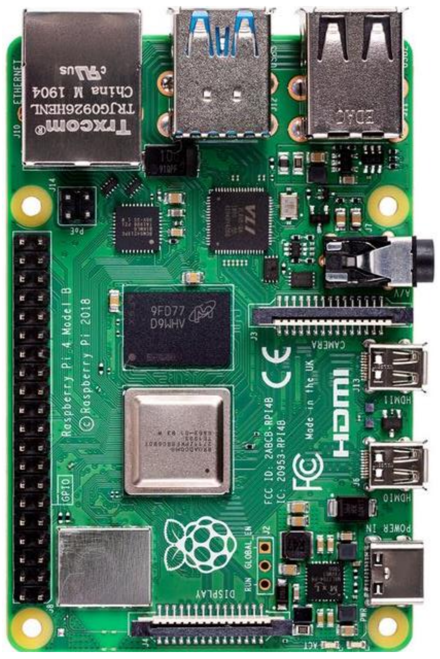
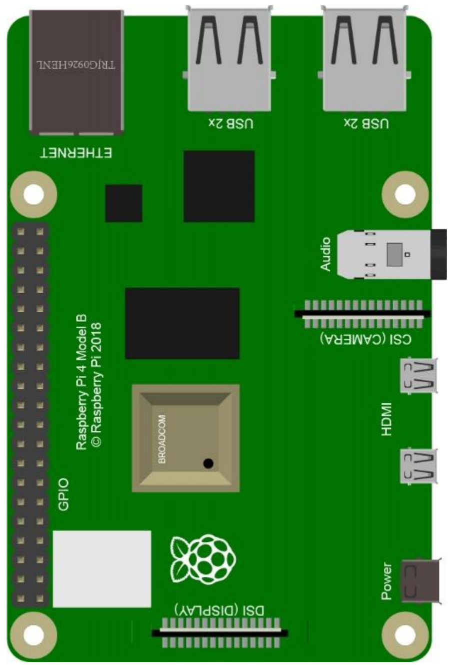
http://www.freenove.com

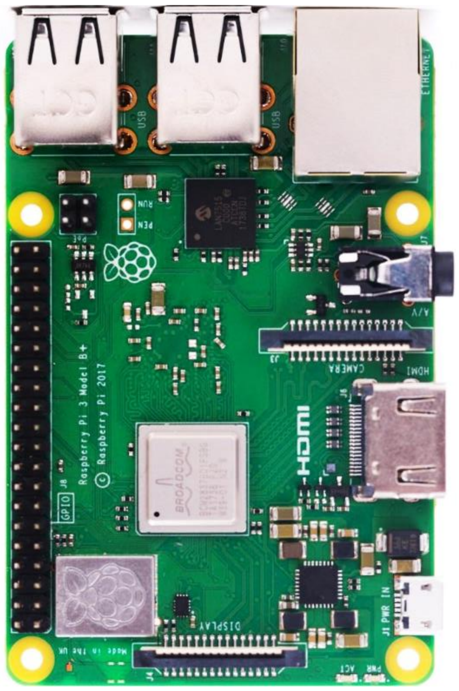
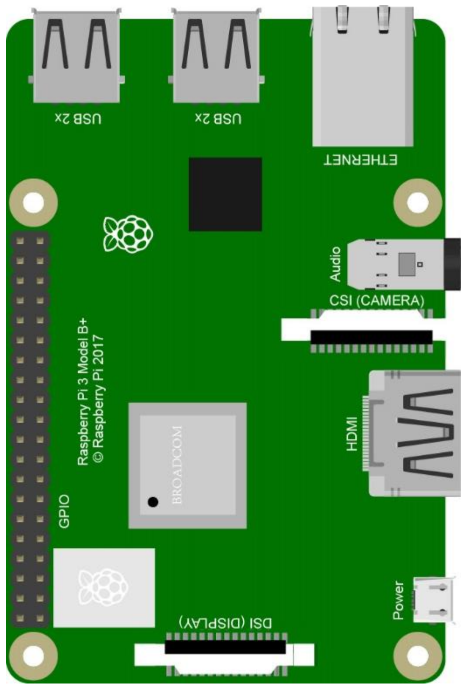
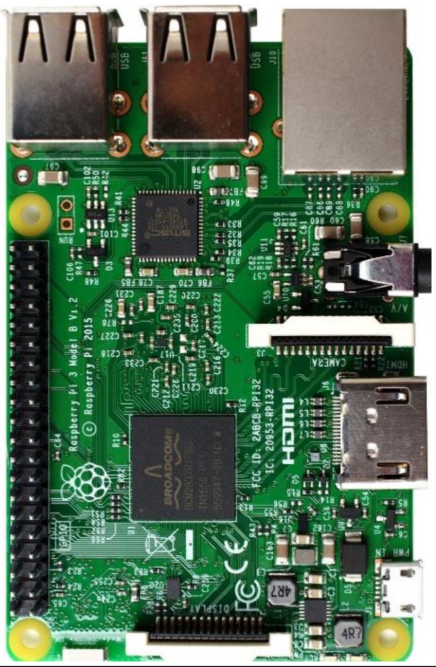
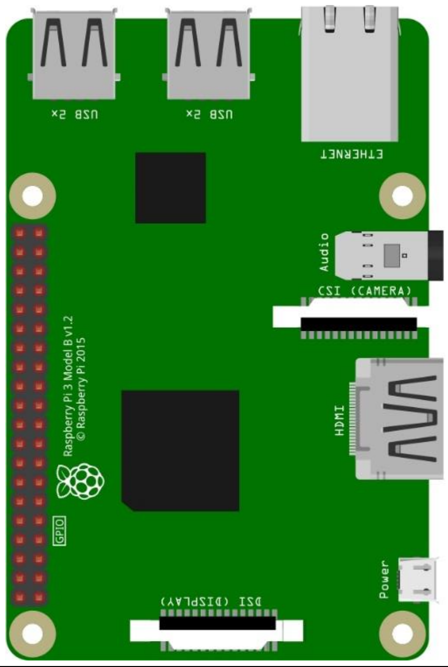
Copyright

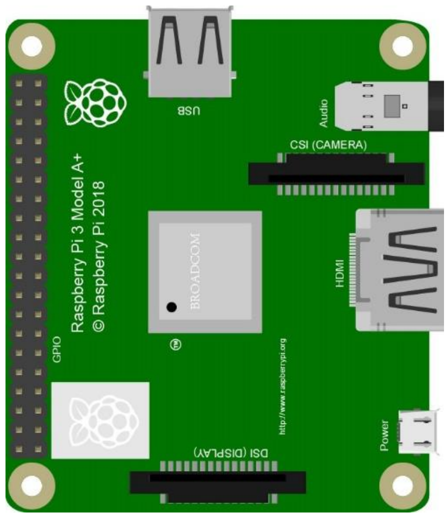
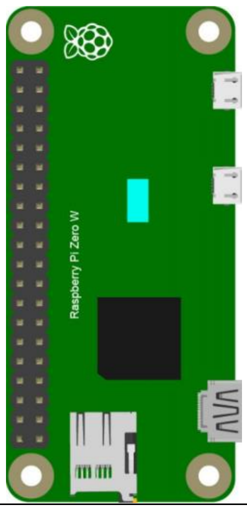
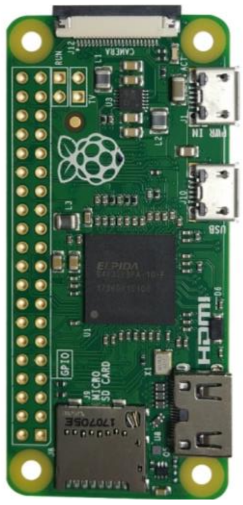
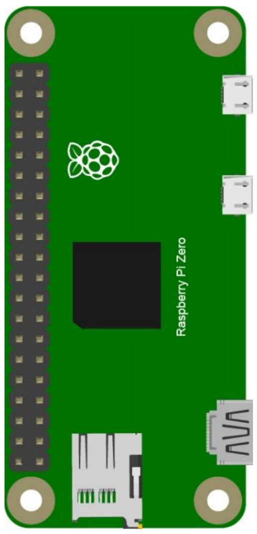
Contents

Raspberry Pi

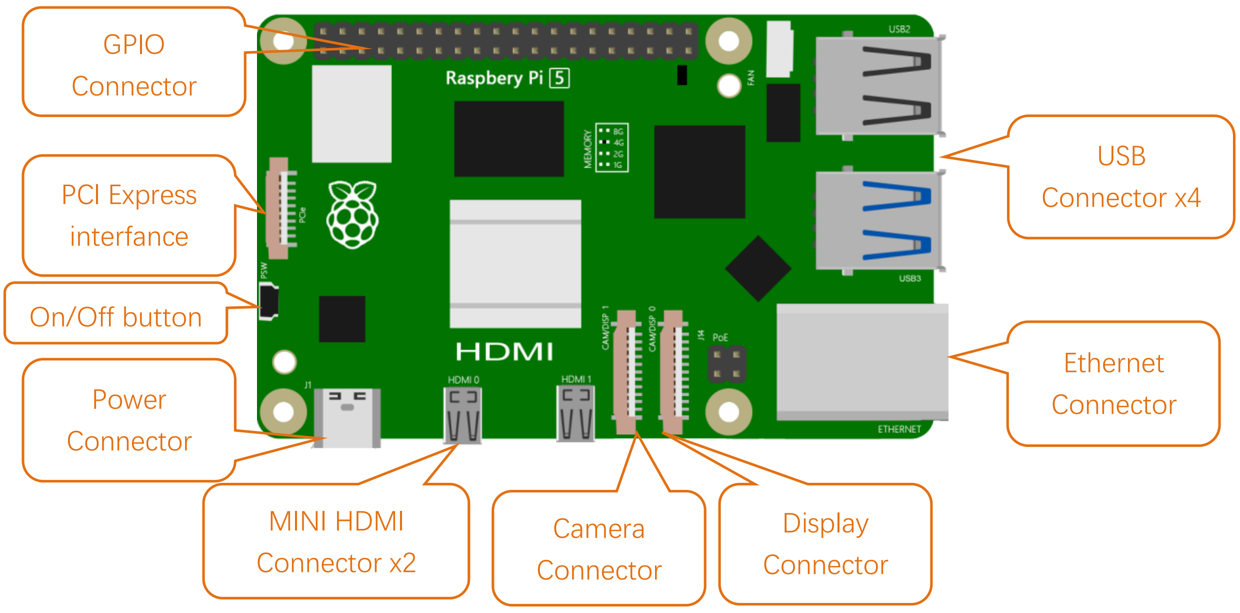
Below are the Raspberry Pi pictures and model pictures supported by this product.

Practicality picture of Raspberry Pi 5:	Model diagram of Raspberry Pi 5:
	
Practicality picture of Raspberry Pi 4 Model B:	Model diagram of Raspberry Pi 4 Model B:
	

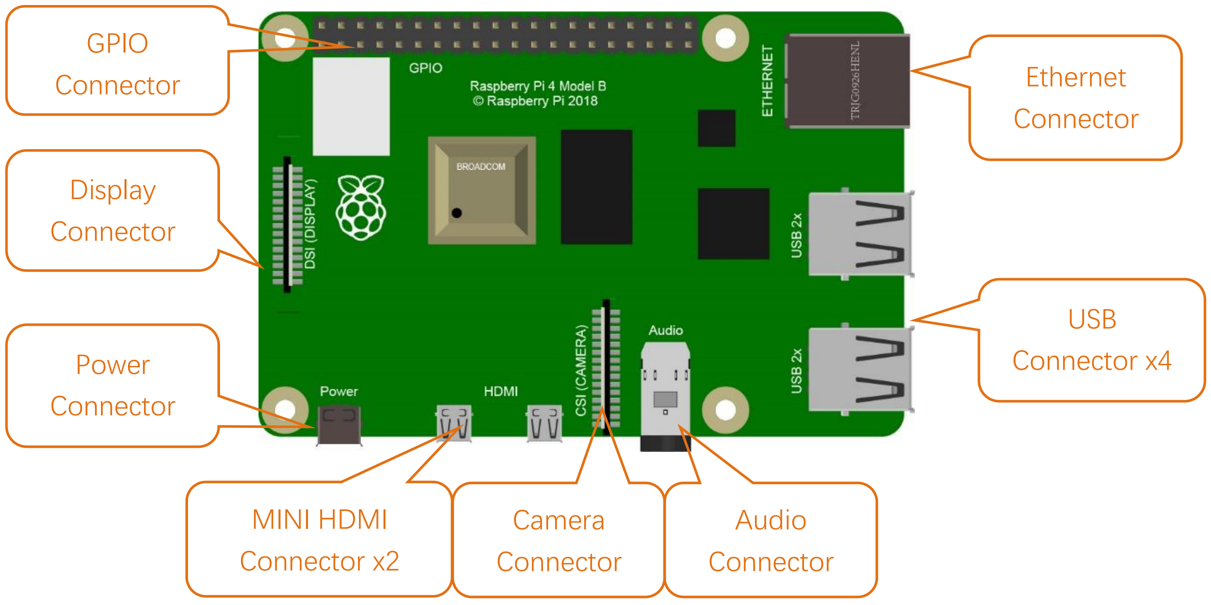
Practicality picture of Raspberry Pi 3 Model B+:	Model diagram of Raspberry Pi 3 Model B+:
	 USB 2x USB 2x ETHERNET Audio CSI (CAMERA) HDMI GPIO Raspberry Pi 3 Model B+ © Raspberry Pi 2017 BROADCOM DSI (DISPLAY) Power
Practicality picture of Raspberry Pi 3 Model B:	Model diagram of Raspberry Pi 3 Model B:
	 USB 2x USB 2x ETHERNET Audio CSI (CAMERA) HDMI GPIO Raspberry Pi 3 Model B v1.2 © Raspberry Pi 2015 DSI (DISPLAY) Power
Practicality picture of Raspberry Pi 3 Model A+:	Model diagram of Raspberry Pi 3 Model A+:

	 USB Audio CSI (CAMERA) Raspberry Pi 3 Model A+ © Raspberry Pi 2018 GPIO HDMI BROADCOM http://www.raspberrypi.org Power DSI (DISPLAY)
Practicality picture of Raspberry Pi Zero W: 	Model diagram of Raspberry Pi Zero W: 
Practicality picture of Raspberry Pi Zero： 	Model diagram of Raspberry Pi Zero： 

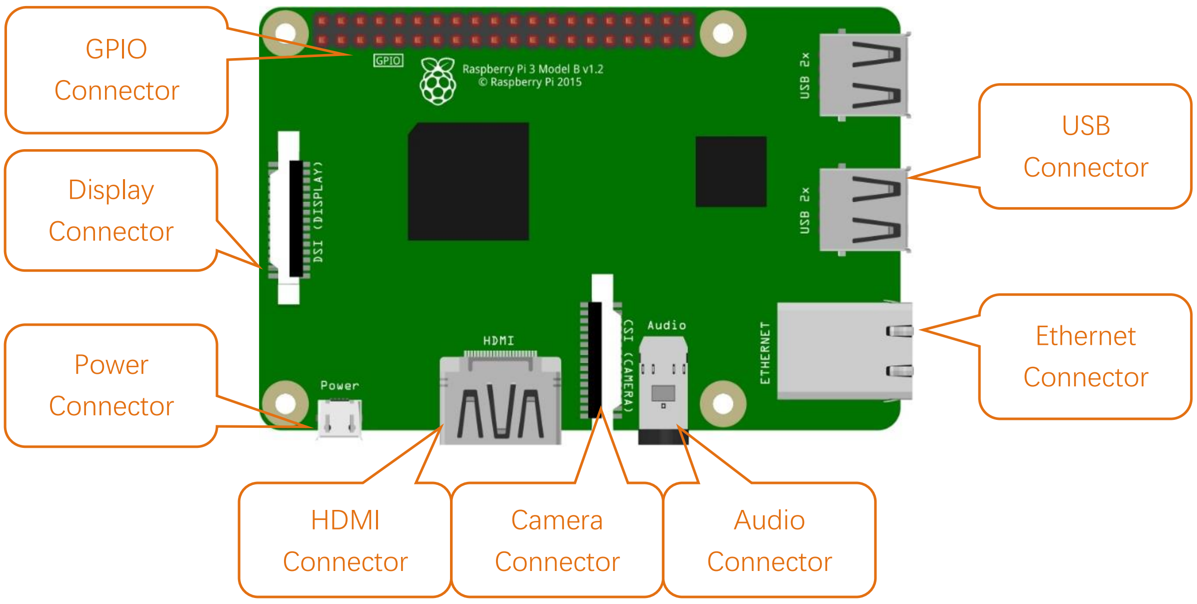
Hardware interface diagram of RPi 5 is shown below:



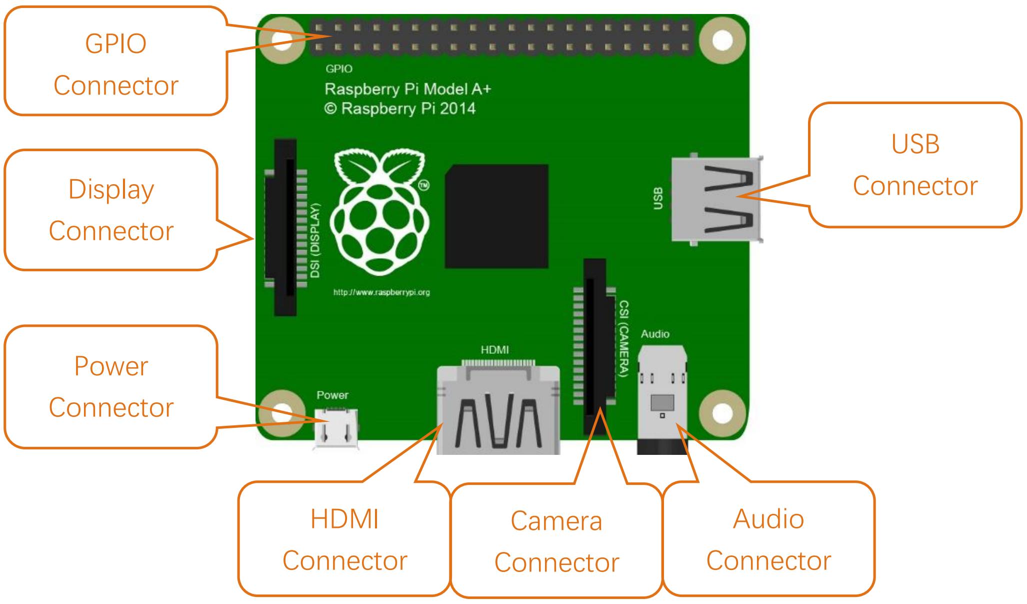
Hardware interface diagram of RPi 4B is shown below:



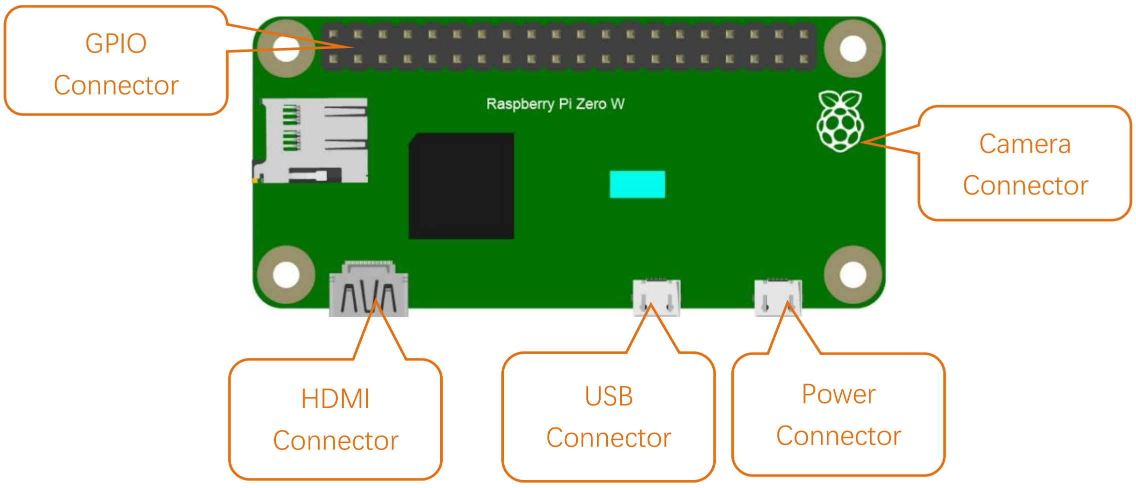
Hardware interface diagram of RPi 3B+/3B/2B/1B+ are shown below:



Hardware interface diagram of RPi 3A+/A+ is shown below:

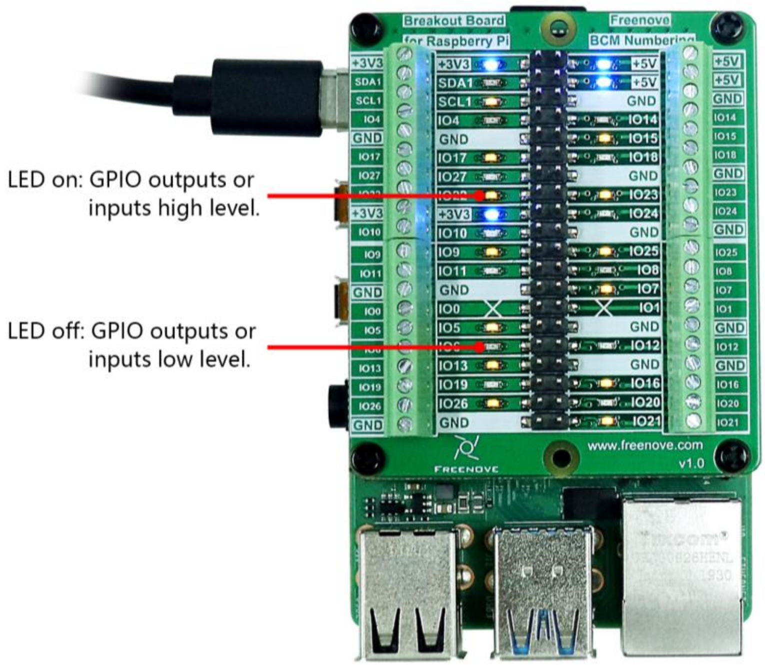


Hardware interface diagram of RPi Zero/Zero W is shown below:

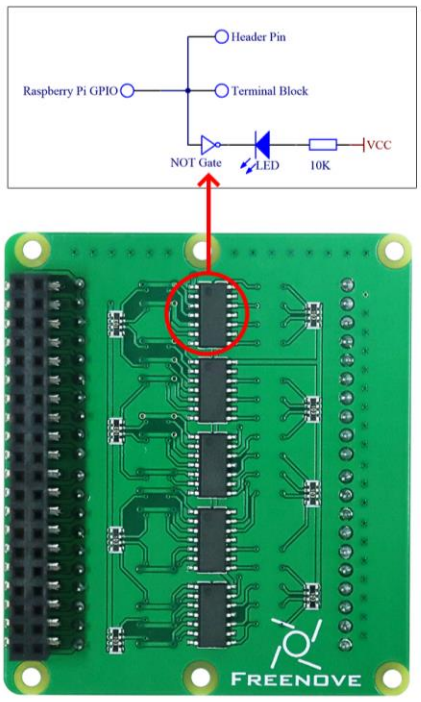


Breakout Board

Led Indicator

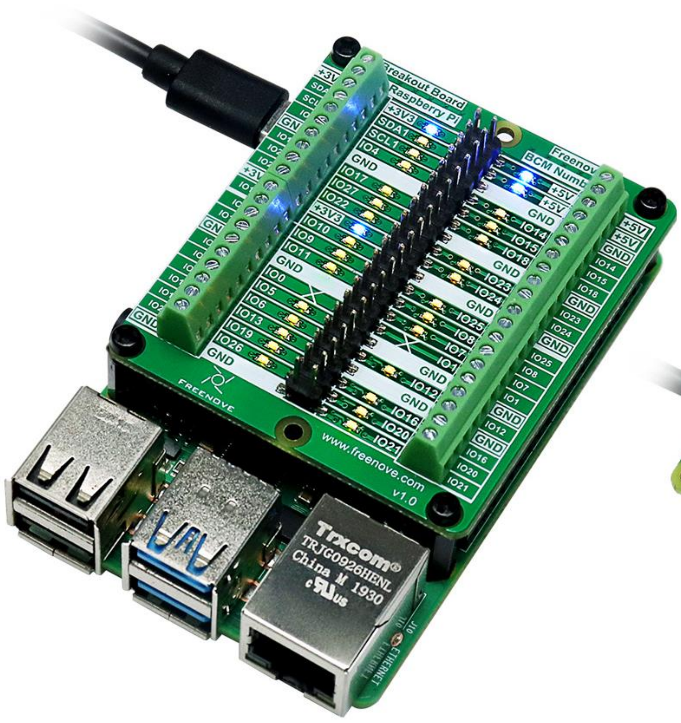

LED on: GPIO outputs or inputs high level.
LED off: GPIO outputs or inputs low level.
Breakout Board for Raspberry Pi
Freenove BCM Numbering
www.freenove.com v1.0
FREENOVE

The GPIO will not be affected by the status LED.

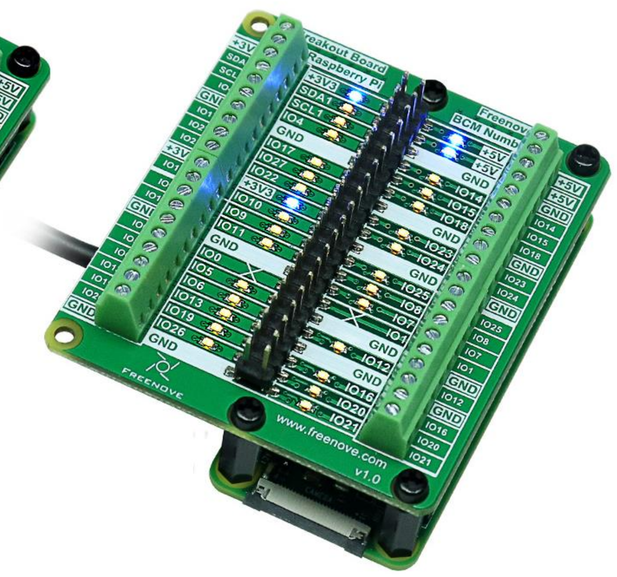


The status LED is driven by the NOT gate chip instead of the GPIO.

Asssembly



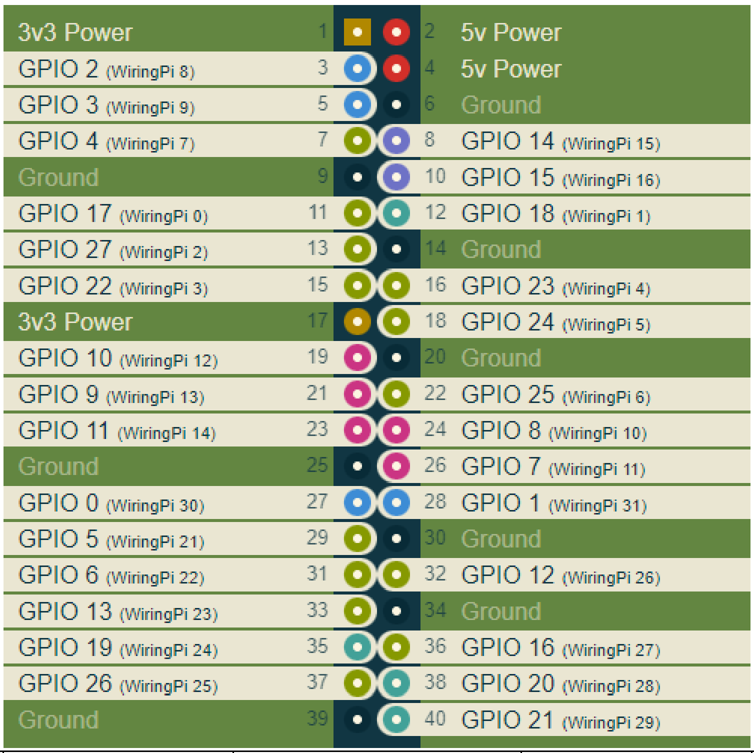
Raspberry Pi 4B



Raspberry Pi Zero

GPIO

GPIO Numbering Relationship



3v3 Power	1	2	5v Power
GPIO 2 (WiringPi 8)	3	4	5v Power
GPIO 3 (WiringPi 9)	5	6	Ground
GPIO 4 (WiringPi 7)	7	8	GPIO 14 (WiringPi 15)
Ground	9	10	GPIO 15 (WiringPi 16)
GPIO 17 (WiringPi 0)	11	12	GPIO 18 (WiringPi 1)
GPIO 27 (WiringPi 2)	13	14	Ground
GPIO 22 (WiringPi 3)	15	16	GPIO 23 (WiringPi 4)
3v3 Power	17	18	GPIO 24 (WiringPi 5)
GPIO 10 (WiringPi 12)	19	20	Ground
GPIO 9 (WiringPi 13)	21	22	GPIO 25 (WiringPi 6)
GPIO 11 (WiringPi 14)	23	24	GPIO 8 (WiringPi 10)
Ground	25	26	GPIO 7 (WiringPi 11)
GPIO 0 (WiringPi 30)	27	28	GPIO 1 (WiringPi 31)
GPIO 5 (WiringPi 21)	29	30	Ground
GPIO 6 (WiringPi 22)	31	32	GPIO 12 (WiringPi 26)
GPIO 13 (WiringPi 23)	33	34	Ground
GPIO 19 (WiringPi 24)	35	36	GPIO 16 (WiringPi 27)
GPIO 26 (WiringPi 25)	37	38	GPIO 20 (WiringPi 28)
Ground	39	40	GPIO 21 (WiringPi 29)

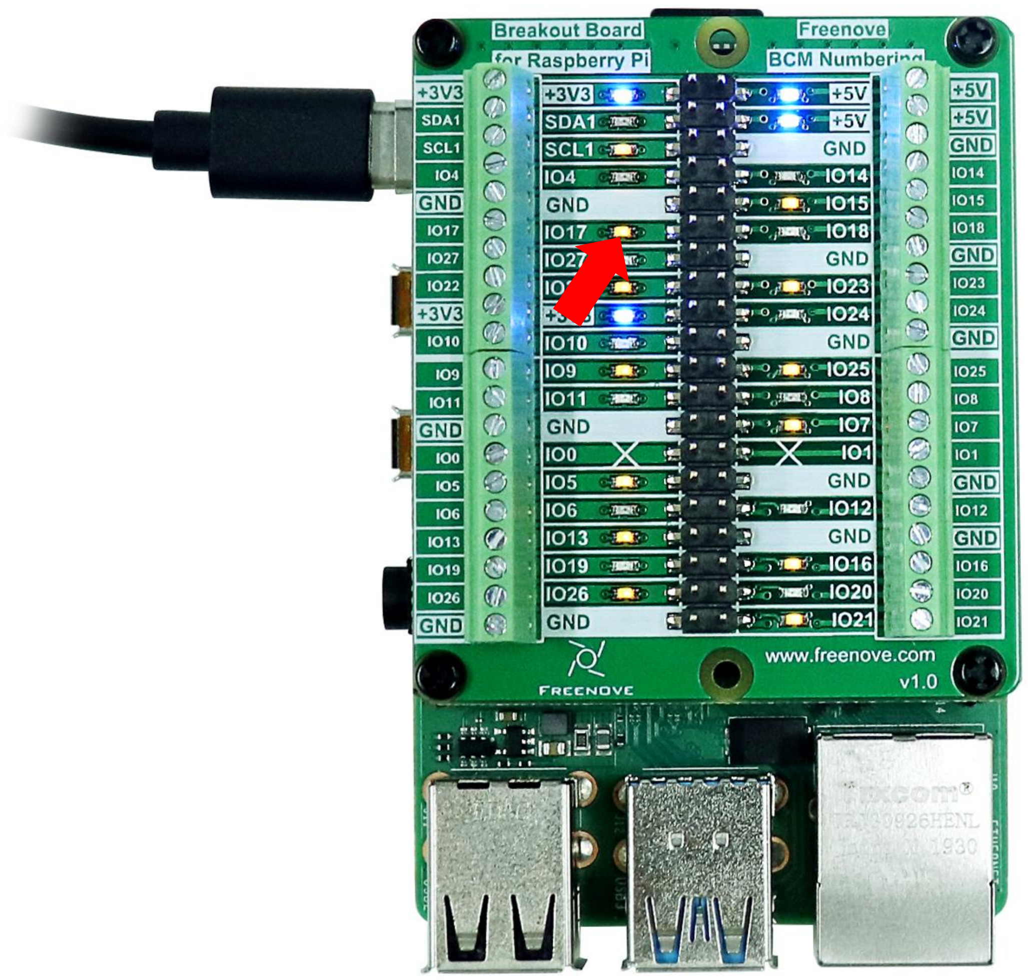
WingPi	BCM	Physical		BCM	WingPi
3.3V	**3.3V**	1	2	5V	**5V**
8	SDA1	3	4	**5V**	**5V**
9	SCL1	5	6	**GND**	**GND**
7	GPIO4	7	8	GPIO14/TXD0	15
GND	**GND**	9	10	GPIO15/RXD0	16
0	GPIO17	11	12	GPIO18	1
2	GPIO27	13	14	**GND**	**GND**
3	GPIO22	15	16	GPIO23	4
3.3V	**3.3V**	17	18	GPIO24	5
12	GPIO10/MOSI)	19	20	**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

For more details about pin definition of GPIO, please refer to http://pinout.xyz/

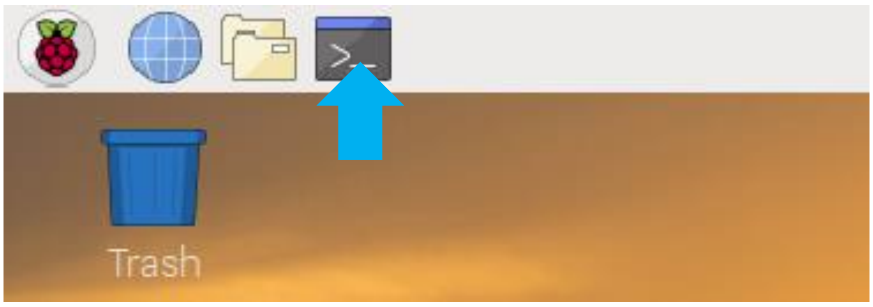
Project Example

LED Blink

We will make the LED indicator of GPIO 17 blink.

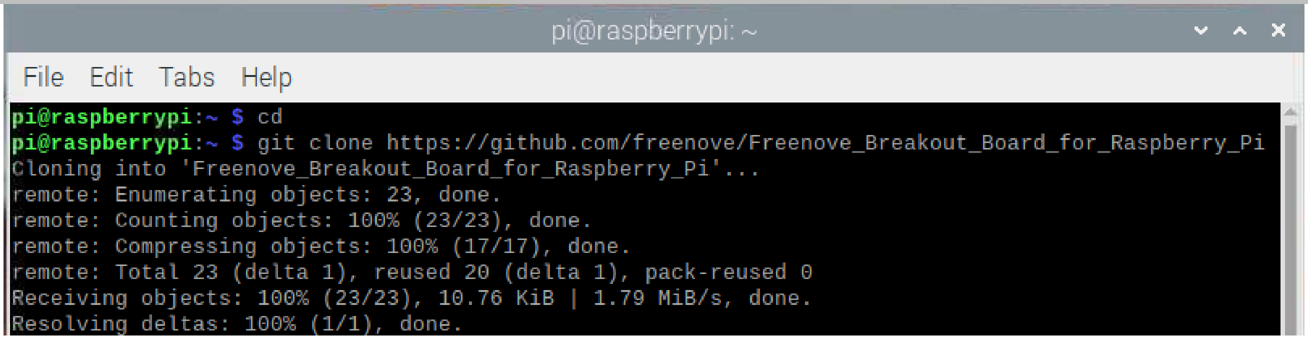


Geting the Code



Run following commands in terminal:

```
cd
git clone https://github.com/freenove/Freenove_Breakout_Board_for_Raspberry_Pi
```



```
pi@raspberrypi:~ $ cd
pi@raspberrypi:~ $ git clone https://github.com/freenove/Freenove_Breakout_Board_for_Raspberry_Pi
Cloning into 'Freenove_Breakout_Board_for_Raspberry_Pi'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 23 (delta 1), reused 20 (delta 1), pack-reused 0
Receiving objects: 100% (23/23), 10.76 KiB | 1.79 MiB/s, done.
Resolving deltas: 100% (1/1), done.
```

Run the Code

C Code 01.1.1 Blink

Raspberry Pi 5 currently does not support the cotrol of GPIOs with the C code we provide.
We will update to make it compatible with Raspberry Pi 5 in the near future. Please wait with patience.

If you have any concerns, please contact us via: support@freenove.com

1. If you did not update wiring pi, please execute following commands **one by one**.

```
sudo apt-get update
git clone https://github.com/WiringPi/WiringPi
cd WiringPi
./build
```

2. Use cd command to enter 01.1.1_Blink directory of C code.

```
cd ~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/C_Code/01.1.1_Blink
```

3. Use the following command to compile the code “Blink.c” and generate executable file “Blink”.

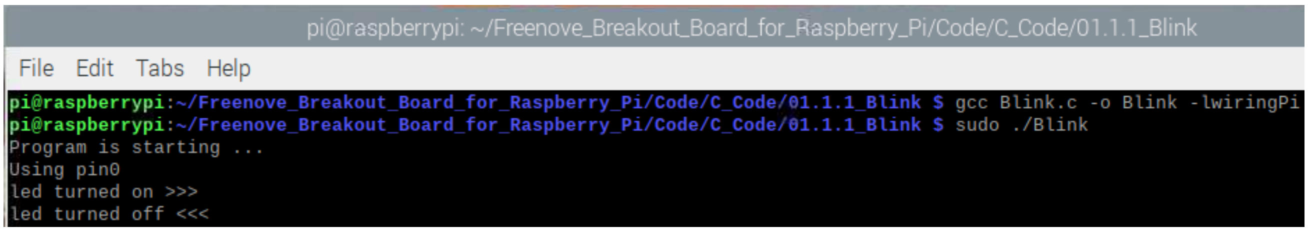
“l” of “lwiringPi” is low case of “L”.

```
gcc Blink.c -o Blink -lwiringPi
```

4. Then run the generated file “blink”.

```
sudo ./Blink
```

Now your LED should start blinking!


pi@raspberrypi: ~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/C_Code/01.1.1_Blink
File Edit Tabs Help

```
pi@raspberrypi:~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/C_Code/01.1.1_Blink $ gcc Blink.c -o Blink -lwiringPi
pi@raspberrypi:~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/C_Code/01.1.1_Blink $ sudo ./Blink
Program is starting ...
Using pin0
led turned on >>>
led turned off <<<
```

You can press “**Ctrl+C**” to end the program. The following is the program code:

```
#include <wiringPi.h>
#include <stdio.h>

#define  ledPin    0  //define the led pin number

void main(void)
{
    printf("Program is starting ... \n");

    wiringPiSetup();  //Initialize wiringPi.

    pinMode(ledPin, OUTPUT);//Set the pin mode
    printf("Using pin%d\n",%ledPin);    //Output information on terminal
    while(1){
        digitalWrite(ledPin, HIGH);  //Make GPIO output HIGH level
        printf("led turned on >>>\n");      //Output information on terminal
        delay(1000);                        //Wait for 1 second
        digitalWrite(ledPin, LOW);  //Make GPIO output LOW level
        printf("led turned off <<<\n");         //Output information on terminal
        delay(1000);                        //Wait for 1 second
    }
}
```

In the code above, the configuration function for GPIO is shown below as:

void pinMode(int pin, int mode);
This sets the mode of a pin to either INPUT, OUTPUT, PWM_OUTPUT or GPIO_CLOCK. Note that only wiringPi pin 1 (BCM_GPIO 18) supports PWM output and only wiringPi pin 7 (BCM_GPIO 4) supports CLOCK output modes. This function has no effect when in Sys mode. If you need to change the pin mode, then you can do it with the gpio program in a script before you start your program
void digitalWrite (int pin, int value);
Writes the value HIGH or LOW (1 or 0) to the given pin, which must have been previously set as an output.

For more related wiringpi functions, please refer to http://wiringpi.com/reference/

GPIO connected to ledPin in the circuit is GPIO17 and GPIO17 is defined as 0 in the wiringPi numbering. So ledPin should be defined as 0 pin. You can refer to the corresponding table in Chapter 0.

```
#define ledPin  0 //define the led pin number
```

GPIO Numbering Relationship

WingPi	BCM(Extension)	Physical		BCM(Extension)	WingPi
3.3V	**3.3V**	1	2	5V	**5V**
8	SDA1	3	4	**5V**	**5V**
9	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

In the main function main(), initialize wiringPi first.

```
    wiringPiSetup();  //Initialize wiringPi.
```

After the wiringPi is initialized successfully, you can set the ledPin to output mode and then enter the while loop, which is an endless loop (a while loop). That is, the program will always be executed in this cycle, unless it is ended because of external factors. In this loop, use digitalWrite (ledPin, HIGH) to make ledPin output high level, then LED turns ON. After a period of time delay, use digitalWrite(ledPin, LOW) to make ledPin output low level, then LED turns OFF, which is followed by a delay. Repeat the loop, then LED will start blinking.

```
    pinMode(ledPin, OUTPUT);//Set the pin mode
    printf("Using pin%d\n",%ledPin);   //Output information on terminal
    while(1){
        digitalWrite(ledPin, HIGH);  //Make GPIO output HIGH level
        printf("led turned on >>>\n");     //Output information on terminal
        delay(1000);                       //Wait for 1 second
        digitalWrite(ledPin, LOW);  //Make GPIO output LOW level
        printf("led turned off <<<\n");        //Output information on terminal
        delay(1000);                       //Wait for 1 second
    }
```

Python Code 01.1.1 Blink

Now, we will use Python language to make a LED blink.

If you have any concerns, please contact us via: support@freenove.com

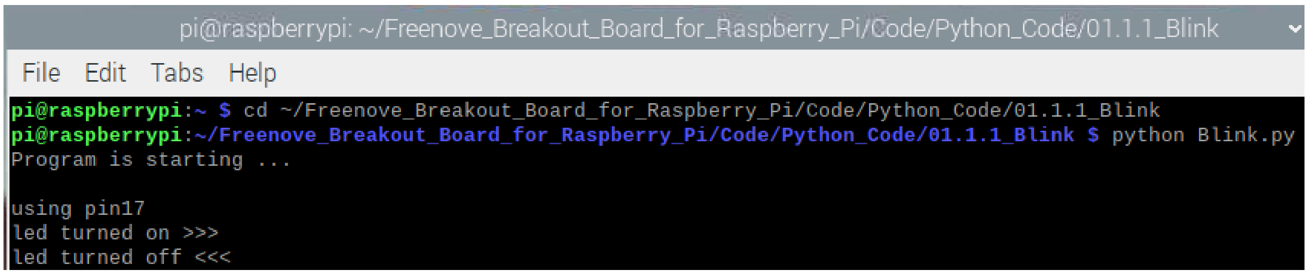
1. Use cd command to enter 01.1.1_Blink directory of Python code.

```
cd ~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/Python_Code/01.1.1_Blink
```

2. Use python command to execute python code blink.py.

```
python Blink.py
```

The LED starts blinking.



pi@raspberrypi: ~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/Python_Code/01.1.1_Blink
File Edit Tabs Help

```
pi@raspberrypi:~ $ cd ~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/Python_Code/01.1.1_Blink
pi@raspberrypi:~/Freenove_Breakout_Board_for_Raspberry_Pi/Code/Python_Code/01.1.1_Blink $ python Blink.py
Program is starting ...

using pin17
led turned on >>>
led turned off <<<
```

You can press "**Ctrl+C**" to end the program. The following is the program code:

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

ledPin = 17    # define ledPin

def setup():
    GPIO.setmode(GPIO.BCM)       # use BCM GPIO Numbering
    GPIO.setup(ledPin, GPIO.OUT)   # set the ledPin to OUTPUT mode
    GPIO.output(ledPin, GPIO.LOW)  # make ledPin output LOW level
    print ('using pin%d'%ledPin)

def loop():
    while True:
        GPIO.output(ledPin, GPIO.HIGH)  # make ledPin output HIGH level to turn on led
        print ('led turned on >>>')        # print information on terminal
        time.sleep(1)                  # Wait for 1 second
        GPIO.output(ledPin, GPIO.LOW)   # make ledPin output LOW level to turn off led
        print ('led turned off <<<')
        time.sleep(1)                  # Wait for 1 second

def destroy():
    GPIO.cleanup()                      # Release all GPIO

if __name__ == '__main__':     # Program entrance
    print ('Program is starting ... \n')
    setup()
    try:
        loop()
    except KeyboardInterrupt:   # Press ctrl-c to end the program.
        destroy()
```

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. GPIO17 uses pin 11 of the board, so define ledPin as 11 and set ledPin to output mode (output low level).

```
ledPin = 17     # define ledPin

def setup():
    GPIO.setmode(GPIO.BCM)        # use BCM GPIO Numbering
    GPIO.setup(ledPin, GPIO.OUT)   # set the ledPin to OUTPUT mode
    GPIO.output(ledPin, GPIO.LOW)  # make ledPin output LOW level
    print ('using pin%d'%ledPin)
```

GPIO Numbering Relationship

WingPi	BCM(Extension)	Physical		BCM(Extension)	WingPi
3.3V	**3.3V**	1	2	5V	**5V**
8	SDA1	3	4	**5V**	**5V**
9	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

In loop(), there is a while loop, which is an endless loop (a while loop). That is, the program will always be executed in this loop, unless it is ended because of external factors. In this loop, set ledPin output high level, then the LED turns ON. After a period of time delay, set ledPin output low level, then the LED turns OFF, which is followed by a delay. Repeat the loop, then LED will start blinking.

```
def loop():
    while True:
        GPIO.output(ledPin, GPIO.HIGH)  # make ledPin output HIGH level to turn on led
        print ('led turned on >>>')        # print information on terminal
        time.sleep(1)                   # Wait for 1 second
        GPIO.output(ledPin, GPIO.LOW)   # make ledPin output LOW level to turn off led
        print ('led turned off <<<')
        time.sleep(1)                   # Wait for 1 second
```

Finally, when the program is terminated, subfunction (a function within the file) will be executed, the LED will be turned off and then the IO port will be released. If you close the program Terminal directly, the program will also be terminated but the finish() function will not be executed. Therefore, the GPIO resources will not be released which may cause a warning message to appear the next time you use GPIO. Therefore, do not get into the habit of closing Terminal directly.

```
def finish():
    GPIO.cleanup()                     # Release all GPIO
```

What's next?

Thanks for your reading.

This book is all over here. If you find any mistakes, missions or you have other ideas and questions about contents of this book or the kit and ect., please feel free to contact us, and we will check and correct it as soon as possible.

After completing the contents in this book, you can try to reform this smart car, such as purchasing and installing other Freenove electronic modules, or improving the code to achieve different functions. We will also try our best to add more new functions and update the code on our github (https://github.com/freenove).

If you want to learn more about Arduino, Raspberry Pi, smart cars, robots and orther interesting products in science and technology, please continue to focus on our website. We will continue to launch cost-effective, innovative and exciting products.

www.freenove.com

https://www.amazon.com/freenove

Thank you again for choosing Freenove products.