After running the blink_all_gpio program, when using a multimeter to measure the level of the GPIO port, you will find that the GPIO pin will switch between 0 and 3.3v continuously. Using this program we can test whether the GPIO port is working properly.

The method of running blink_all_gpio program is shown below：

orangepi@orangepi3b:~$ **sudo blink_all_gpio** #Remember to add sudo permissions

[sudo] password for orangepi: #You need to enter password here

1) There are a total of **28** GPIO ports in the 40pins of the development board that can be used. The following uses pin 7 — the corresponding GPIO is GPIO4_C3 — — the corresponding wPi serial number is 2—as an example to demonstrate how to set the high and low levels of the GPIO port

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |   PWM15  |     IN | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
```

2) First set the GPIO port as the output mode, the third parameter needs to enter the serial number of the wPi corresponding to the pins

```
root@orangepi:~/wiringOP# gpio mode 2 out
```

3) Then set the GPIO port to output a low level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 0v, it means that the low level is set successfully.

```
root@orangepi:~/wiringOP# gpio write 2 0
```

Using gpio readall, you can see the value of the No. 7 pin (V) to 0

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |   PWM15  |    OUT | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
```

4) Then set the GPIO port to output a high level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 3.3v, it means that the high level is set successfully.

```
root@orangepi:~/wiringOP# gpio write 2 1
```

Using gpio readall, you can see the value of No. 7 pin (V) into 1

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |   PWM15  |    OUT | 1 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
```

5) The setting method of other pins is similar, just modify the serial number of wPi to the corresponding serial number of the pin

3. 16. 2. 40pin GPIO port pull-down resistance setting method

Note that the 4 GPIO pins below Orange Pi 3B are invalid because there are 3.3V on the outside, so setting drop-down is invalid. Other pins can be set normally

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |   PWM15  |     IN | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
 |  118 |   5 | GPIO3_C6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO3_C7 | 6   | 119  |
 |  128 |   7 | GPIO4_A0 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |  130 |   8 |    TXD.7 |     IN | 0 | 15 || 16 | 0 | IN     | RXD.7    | 9   | 131  |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | IN     | GPIO4_A1 | 10  | 129  |
 |  138 |  11 | SPI3_TXD |     IN | 0 | 19 || 20 |   |        | GND      |     |      |
 |  136 |  12 | SPI3_RXD |     IN | 0 | 21 || 22 | 0 | IN     | TXD.9    | 13  | 132  |
 |  139 |  14 | SPI3_CLK |     IN | 0 | 23 || 24 | 0 | IN     | SPI3_CS1 | 15  | 134  |
 |      |     |      GND |        |   | 25 || 26 | 0 | IN     | GPIO3_D6 | 16  | 126  |
 |   32 |  17 |    SDA.3 |     IN | 1 | 27 || 28 | 1 | IN     | SCL.3    | 18  | 33   |
 |  133 |  19 |    RXD.9 |     IN | 0 | 29 || 30 |   |        | GND      |     |      |
 |  124 |  20 | GPIO3_D4 |     IN | 0 | 31 || 32 | 0 | IN     | PWM11    | 21  | 144  |
 |  127 |  22 | GPIO3_D7 |     IN | 0 | 33 || 34 |   |        | GND      |     |      |
 |  120 |  23 | GPIO3_D0 |     IN | 0 | 35 || 36 | 0 | IN     | GPIO3_D5 | 24  | 125  |
 |  123 |  25 | GPIO3_D3 |     IN | 0 | 37 || 38 | 0 | IN     | GPIO3_D2 | 26  | 122  |
 |      |     |      GND |        |   | 39 || 40 | 0 | IN     | GPIO3_D1 | 27  | 121  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
```

1) Below the No. 11 pin — corresponding to GPIO 3_C6-corresponding wPi serial number 5—to demonstrate how to pull-down resistance of the GPIO port

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |    PWM15 |     IN | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
 |  118 |   5 | GPIO3_C6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO3_C7 | 6   | 119  |
 |  128 |   7 | GPIO4_A0 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |  130 |   8 |    TXD.7 |     IN | 0 | 15 || 16 | 0 | IN     | RXD.7    | 9   | 131  |
```

2) First of all, you need to set the GPIO port as the input mode. The third parameter needs to enter the serial number of the wPi corresponding to the pins

```
root@orangepi:~/wiringOP# gpio mode 5 in
```

3) After the setting is set to input mode, execute the following command to set the GPIO port as the pull-up mode

```
root@orangepi:~/wiringOP# gpio mode 5 up
```

4) Then enter the following command to read the level of the GPIO port, if the level is 1, it means that the pull-up mode is set successfully

```
root@orangepi:~/wiringOP# gpio read 5
1
```

5) Then execute the following command to set the GPIO port to pull-down mode

```
root@orangepi:~/wiringOP# gpio mode 5 down
```

6) Then enter the following command to read the level of the GPIO port, if the level is 0, the pull-down mode is set successfully

```
root@orangepi:~/wiringOP# gpio read 5
0
```

3. 16. 3. 40pin SPI Test

1) According to the table below, the spi available for Orange Pi 3B is spi3

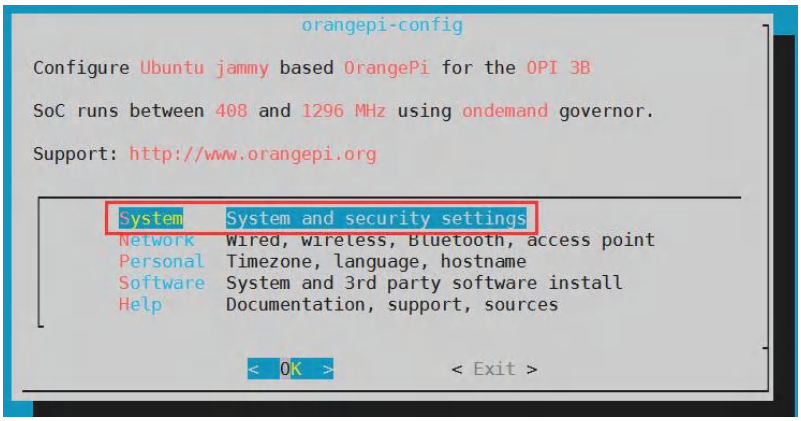
复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	126	GPIO3_D6		
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

2) In the linux system, the SPI in the 40 pin is closed by default, and it needs to be opened manually before it can be used. The detailed steps are as follows:

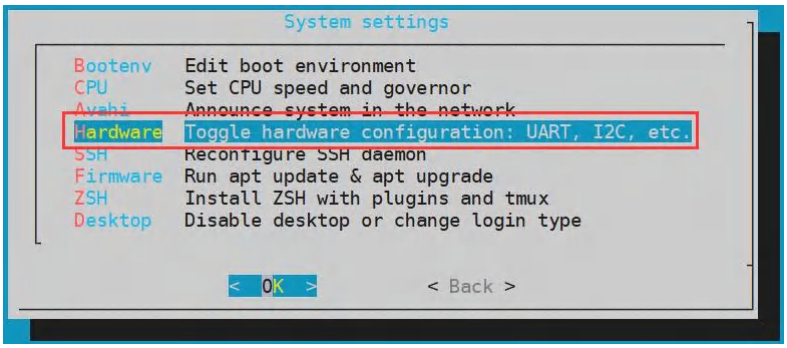
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

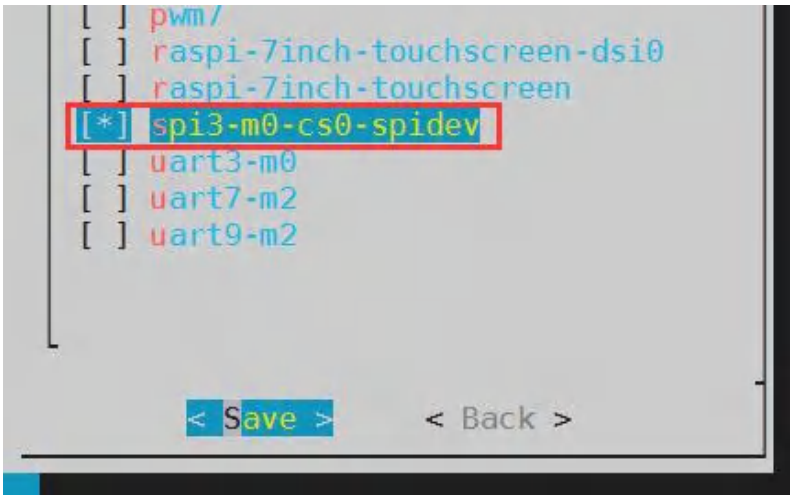
b. Then select **System**



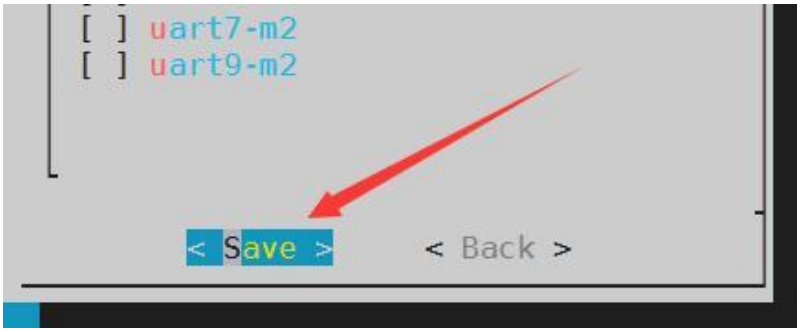
c. Then select **Hardware**



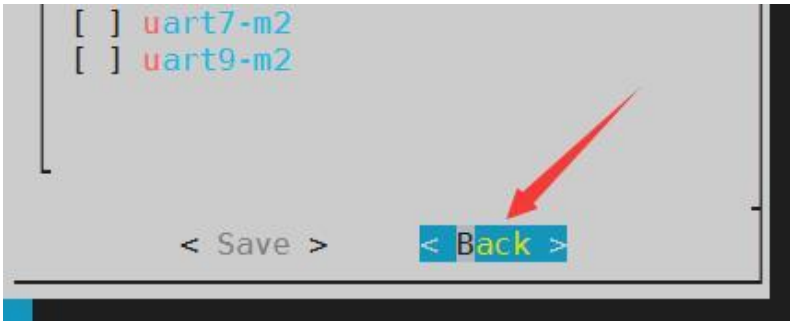
d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the SPI configuration you want to open



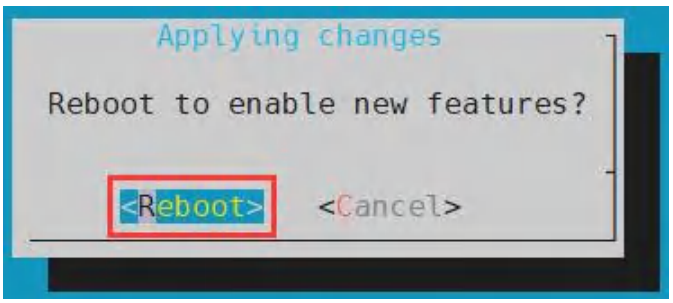
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



3) After restarting, enter the system to check whether there is a **spidev3.0** device node in the Linux system. If it exists, it means that SPI3 has been set up and can be used directly

```
orangepi@orangepi:~$ ls /dev/spidev3.0
/dev/spidev3.0
```

4) Do not short-circuit the mosi and miso pins of SPI3, the output result of running spidev_test is as follows, you can see that the data of TX and RX are inconsistent

```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev3.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.................▒.
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF    | ................................
```

5) Then short-circuit the two pins of mosi (pin 19 in the 40pin interface) and miso (pin 21 in the 40pin interface) of SPI3, and then run the output of spidev_test as follows, you can see the sending and receiving same data


```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev3.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.................▒.
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.................▒.
```

3. 16. 4. 40pin I2C Test

1) From the table below, the I2C available for Orange Pi 3B is I2C2, I2C3, and I2C4 a total of three groups of I2C bus.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	135	GPIO4_A7	SPI3_CS1_M0	
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

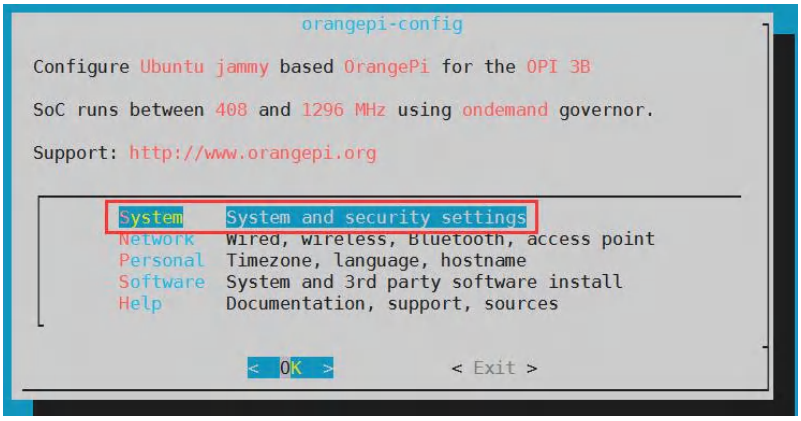
As can be seen from the above table, i2c4_m0 and spi3_m0 are multiplexed pins, and both cannot be opened at the same time, and i2c3_m0 and uart3_m0 are also referenced. Open at the same time, i2c3_m0 and uart3_m0 are also pin multiplexed, and both cannot be opened at the same time

2) In the linux system, the I2C bus in the 40 pin is closed by default, and it needs to be opened manually to use it. The detailed steps are as follows:

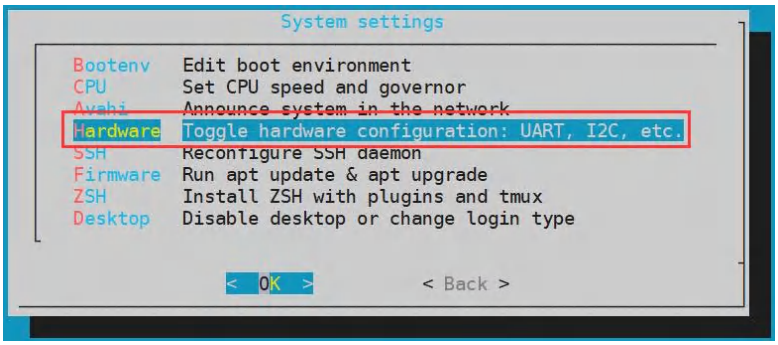
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

b. Then select **System**

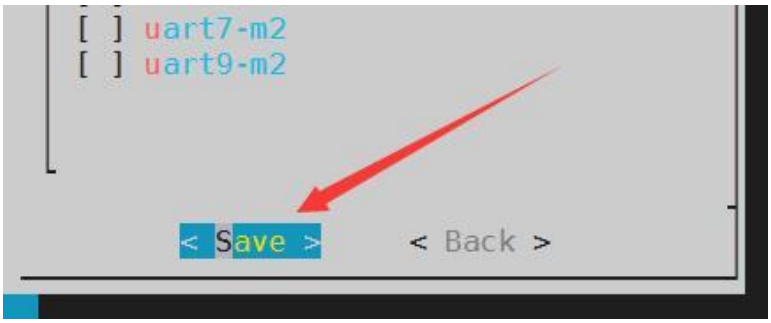


c. Then select **Hardware**

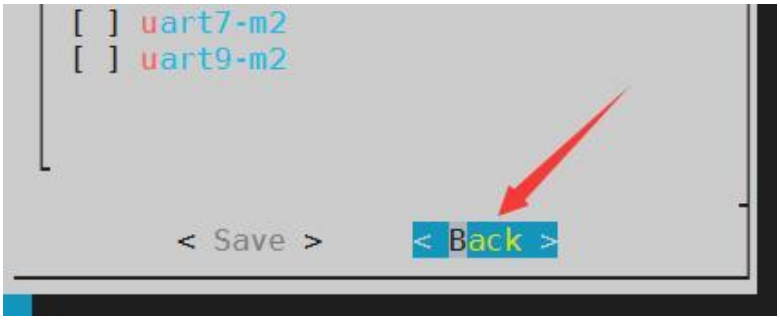


d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the I2C configuration you want to open

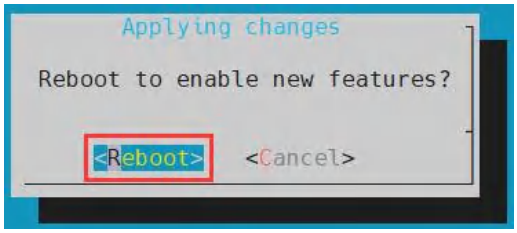
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



3) After starting the Linux system, first confirm that the i2c device node exists under /dev

```
orangepi@orangepi:~# ls /dev/i2c-*
/dev/i2c-0  /dev/i2c-2  /dev/i2c-3  /dev/i2c-4  /dev/i2c-6
```

4) Then connect a i2c device on the i2c pin of the 40Pin connector

	i2c2-m1	i2c3-m0	i2c4-m0
Sda Pin	Corresponding to No. 3 pin	Corresponding to No. 27 pin	Corresponding to No. 19 pin
Sck Pin	Corresponding to No. 5 pin	Corresponding to No. 28 pin	Corresponding to No. 23 pin
Vcc Pin	Corresponding to No. 1 pin	Corresponding to No. 1 pin	Corresponding to No. 1 pin
Gnd Pin	Corresponding to No. 6 pin	Corresponding to No. 6 pin	Corresponding to No. 6 pin

5) Then use the **i2cdetect -y** command, if the address of the connected i2c device can be detected, it means that i2c can be used normally

```
orangepi@orangepi:~$ sudo i2cdetect -y 2      #i2c2 command
orangepi@orangepi:~$ sudo i2cdetect -y 3      #i2c3 command
orangepi@orangepi:~$ sudo i2cdetect -y 4      #i2c4 command
```

```
root@orangepi3b:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- 38 -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

3.16.5. 40pin UART test

1) As can be seen from the table below, the available uarts for Orange Pi 3B are uart2, uart3, uart7 and uart9, a total of four groups of uart buses, of which uart2 is the system’s debugging serial port by default.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	135	GPIO4_A7	SPI3_CS1_M0	
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

As can be seen from the above table, i2c3_m0 and uart3_m0 are pin-multiplexed, and both cannot be turned on at the same time.

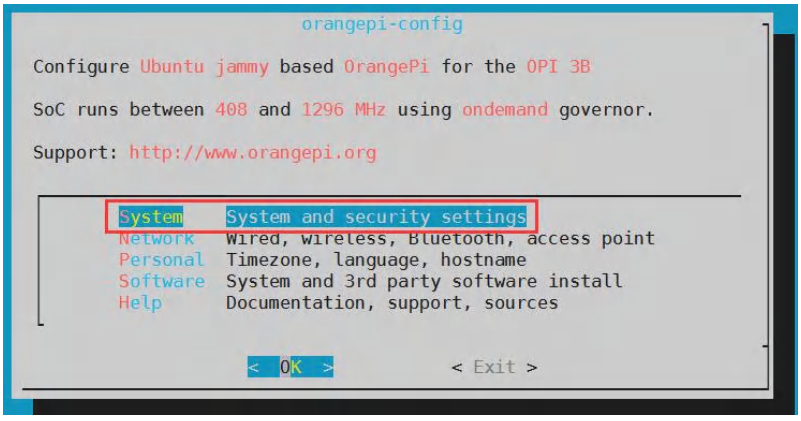
Please note that uart2_m0 is used as the debugging serial port of the system by default. If the configuration of uart2 is turned on, the debugging serial port function will not be available.

2) In the linux system, the UART in the 40 pins is closed by default, and it needs to be opened manually before it can be used. The detailed steps are as follows:

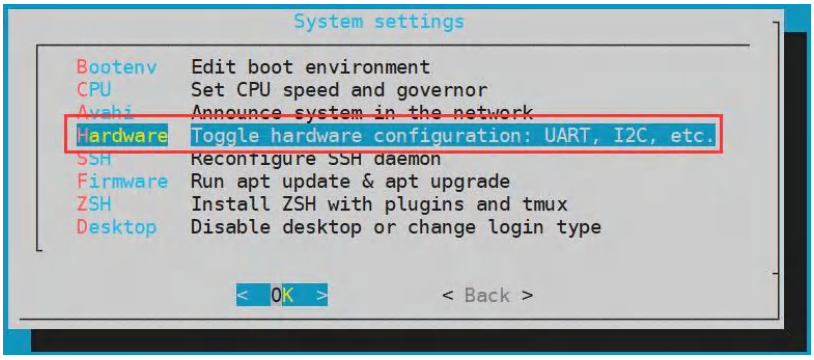
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

b. Then select **System**

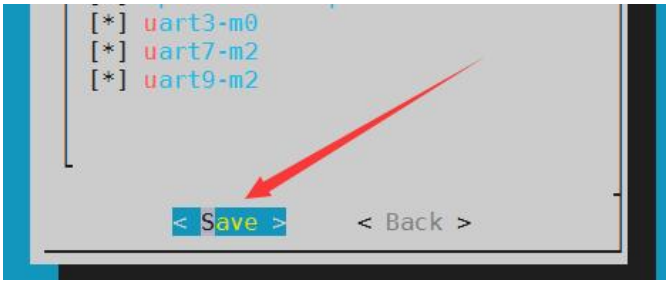


c. Then select **Hardware**

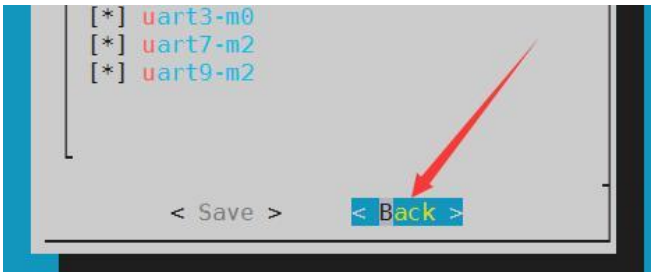


d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the UART configuration you want to open

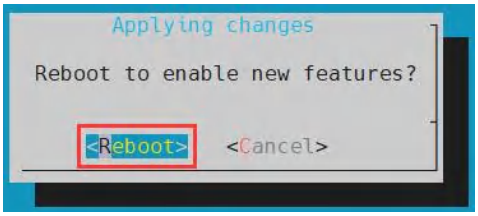
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



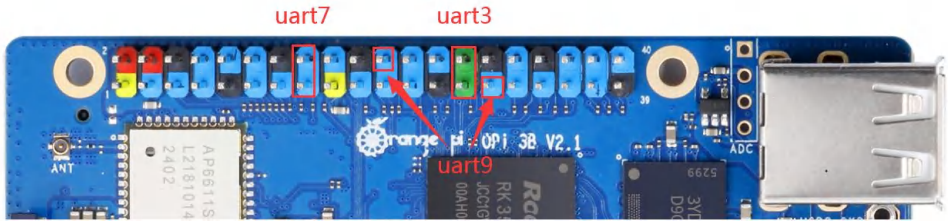
3) After entering the Linux system, first confirm whether there is a device node

corresponding to uart under/dev

```
orangepi@orangepi:~# ls /dev/ttyS*
/dev/ttyS1   /dev/ttyS3   /dev/ttyS7   /dev/ttyS9
```

4) tThen start to test the uart interface, first use the DuPont line to short-circuit the rx and tx of the uart interface to be tested

	uart3	uart7	uart9
Tx pin	Corresponding to the 28 pin	Corresponding to the 16 pin	Corresponding to the 29 pin
Rx Pin	Corresponding to the 27 pin	Corresponding to the 15 pin	Corresponding to the 22 pin



5) Use the **gpio serial** command to test the loopback function of the serial port as shown below, if you can see the following print, it means the serial port communication is normal

a. Test UART3

```
orangepi@orangepi:~$ sudo gpio serial /dev/ttyS3
[sudo] password for orangepi:    #Enter the password here.

Out:    0:   ->    0
Out:    1:   ->    1
Out:    2:   ->    2
Out:    3:   ->    3
Out:    4:   ->    4
Out:    5:   ->    5^C
```

b. Test UART7

```
orangepi@orangepi:~$ sudo gpio serial /dev/ttyS7
[sudo] password for orangepi:    #Enter the password here.

Out:    0:   ->    0
```

```
Out:    1:  ->    1
Out:    2:  ->    2
Out:    3:  ->    3
Out:    4:  ->    4
Out:    5:  ->    5^C
```

c. Test UART9

```
orangepi@orangepi:~$ sudo gpio serial /dev/ttyS9
[sudo] password for orangepi:    #Enter the password here.

Out:    0:  ->    0
Out:    1:  ->    1
Out:    2:  ->    2
Out:    3:  ->    3
Out:    4:  ->    4
Out:    5:  ->    5^C
```

3. 16. 6. How to test PWM using /sys/class/pwm

1) As can be seen from the table below, Orange Pi 3B has two pwm channels available pwm11 and pwm15

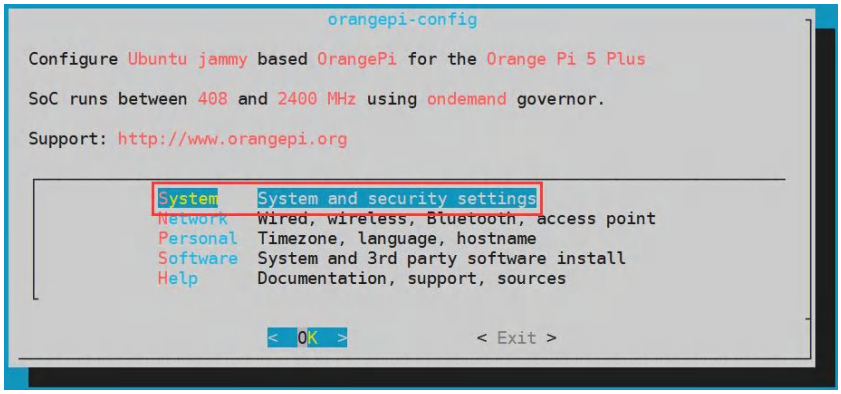
复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	126	GPIO3_D6		
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

2) In the linux system, the PWM in the 40 pin is turned off by default, and it needs to be turned on manually before it can be used. The detailed steps are as follows:

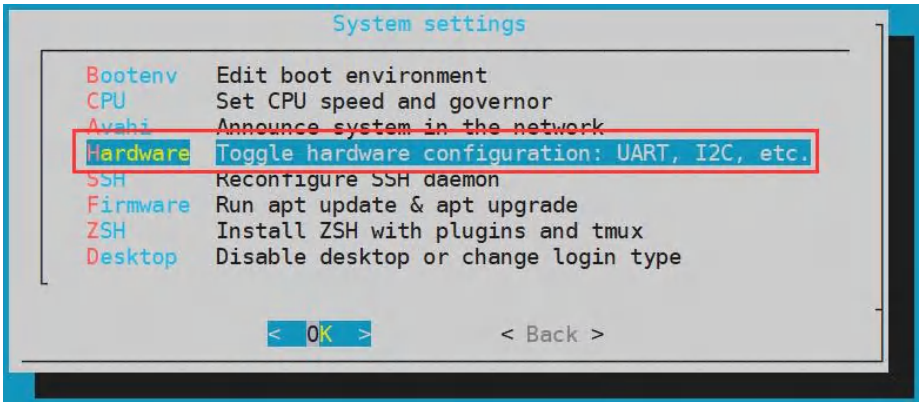
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

b. Then select **System**

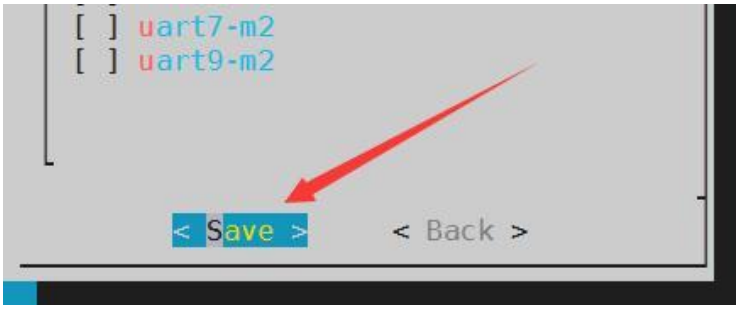


c. Then select **Hardware**

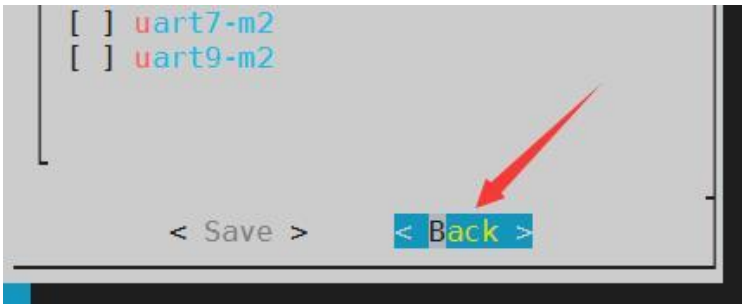


d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the PWM configuration you want to open

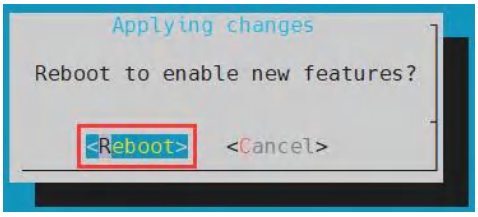
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



3) After opening a pwm, there will be an extra pwmchipX in **/sys/class/pwm/** (X is a specific number), for example, after opening pwm11, check the pwmchipX under **/sys/class/pwm/** one becomes two

```
orangepi@orangepi:~$ ls /sys/class/pwm/
pwmchip0    pwmchip1
```

4) Which pwmchip above corresponds to pwm11, let's check the output of the **ls /sys/class/pwm/ -l** command first, as shown below:

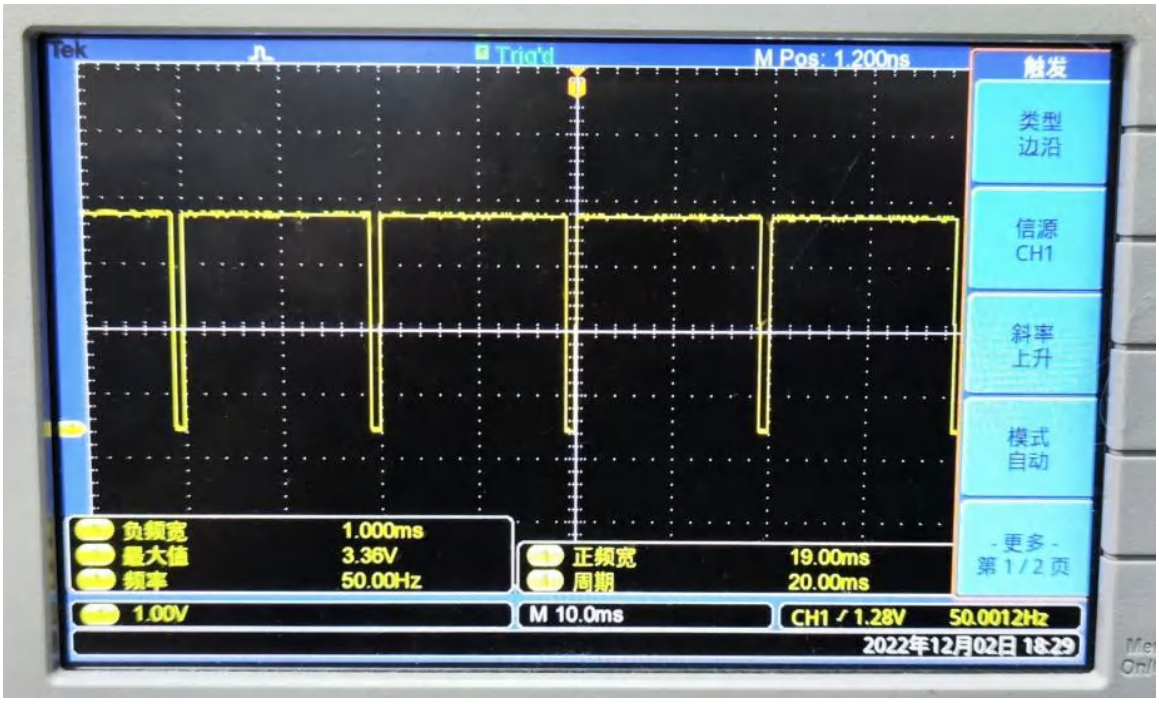
```
orangepi@orangepi3b:~$ ls /sys/class/pwm/ -l
total 0
lrwxrwxrwx 1 root root 0 Jan  1  1970 pwmchip0 -> ../../devices/platform/fe6e0030.pwm/pwm/pwmchip0
lrwxrwxrwx 1 root root 0 Jan  1  1970 pwmchip1 -> ../../devices/platform/fe6f0030.pwm/pwm/pwmchip1
```

5) Then it can be known from the table below that the base address of the pwm11 register is fe6f0030, and then look at the output of the **ls /sys/class/pwm/ -l** command, you can see that pwmchip1 is linked to fe6f0030.pwm, so pwm11 corresponds to pwmchip as pwmchip1

引脚序号	GPIO序号	GPIO	复用功能	复用功能
2		5V		
4		5V		
6		GND		
8	25	GPIO0_D1	UART2_TX_M0	
10	24	GPIO0_D0	UART2_RX_M0	
12	119	GPIO3_C7		
14		GND		
16	131	GPIO4_A3	UART7_RX_M2	
18	129	GPIO4_A1		
20		GND		
22	132	GPIO4_A4	UART9_TX_M2	
24	134	GPIO4_A6	SPI3_CS0_M0	
26	126	GPIO3_D6		
28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
30		GND		
32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
34		GND		
36	125	GPIO3_D5		
38	122	GPIO3_D2		
40	121	GPIO3_D1		

6) Then use the following command to make pwm11 output a 50Hz square wave (please switch to the root user first, and then execute the following command)

```
root@orangepi:~# echo 0 > /sys/class/pwm/pwmchip1/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip1/pwm0/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip1/pwm0/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip1/pwm0/enable
```

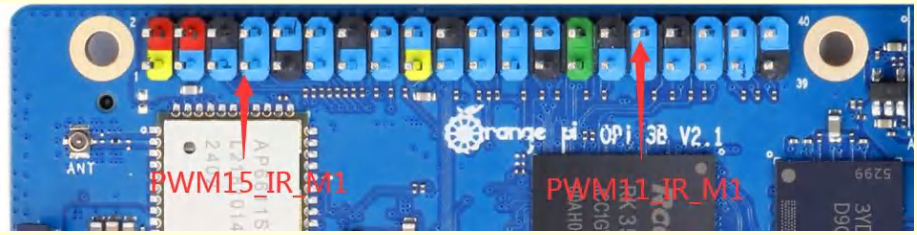


7) Other pwm test methods are similar to the test method of pwm11 demonstrated above.

3. 17. How to use wiringOP hardware PWM

For how to download and install wiringOP, please refer to the instructions in the section How to Install wiringOP.

The development board can use a total of 2 PWM channels, PWM15_IR_M1 and PWM11_IR_M1. The locations of their pins are as shown in the figure below:



3. 17. 1. How to set PWM using wiringOP's gpio command

3. 17. 1. 1. Set the corresponding pin to PWM mode

1) The corresponding relationship between PWM pin and wPi serial number is as shown in the following table:

PWM Pin	wPi serial number
PWM15_IR_M1	2
PWM11_IR_M1	21

2) The command to set the pin to PWM mode is as follows, taking PWM15_IR_M1 as an example. The third parameter needs to enter the serial number of the wPi corresponding to the PWM15_IR_M1 pin.

```
orangepi@orangepi:~$ gpio mode 2 pwm
```

3) After the pin is set to PWM mode, it will output a square wave with a frequency of 200Hz and a duty cycle of 50% by default. At this time, we use an oscilloscope to measure the corresponding PWM pin and you can see the following waveform.

3.17.1.2. Method of adjusting PWM duty cycle

1) The calculation formula of PWM duty cycle is as follows. We can adjust the PWM duty cycle by setting the values of CCR and ARR.

PWM duty cycle = CCR / ARR

In:

The value range of CCR is 0~65535, and the default value is 500.

The value range of ARR is 0~65535, and the default value is 1000.

It should be noted that our CCR value needs to be smaller than the ARR value because the duty cycle cannot be greater than 1.

When setting CCR > ARR, the following error message will be prompted:

gpio: CCR should be less than or equal to ARR (XXX)

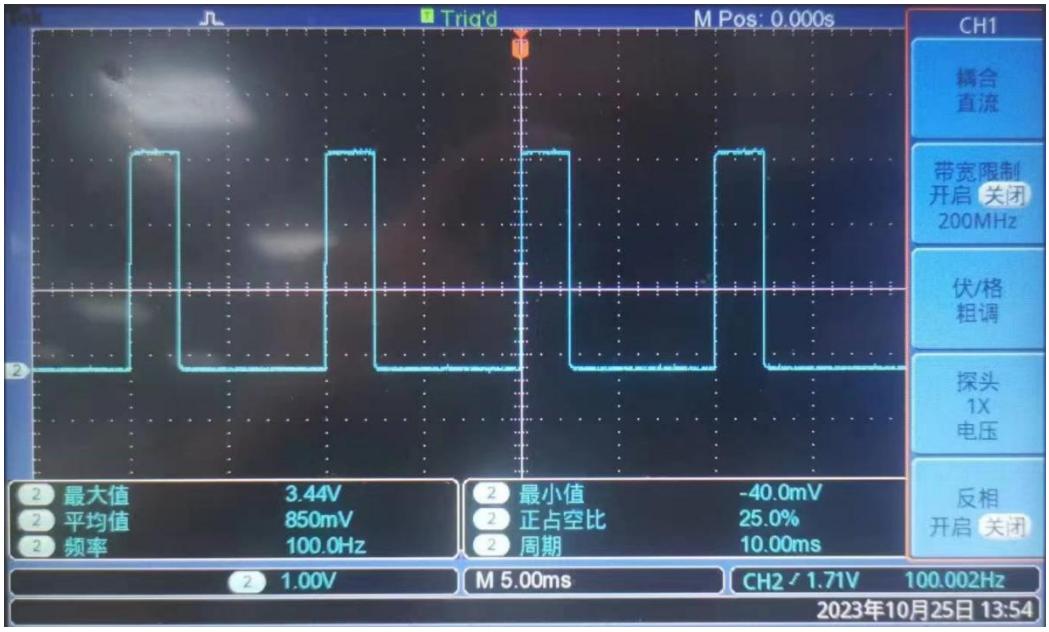
When setting ARR < CCR, the following error message will be prompted:

gpio: ARR should be greater than or equal to CCR (XXX)

2) We can use the following command to set the ARR of the PWM15_IR_M1 pin to 2000

```
orangepi@orangepi:~$ gpio pwmr 2 2000
```

3) After running the above command, you can observe through the oscilloscope that the PWM duty cycle changes from the default 50% (500/1000) to 25% (500/2000)



4) We can use the following command to set the CCR of the PWM15_IR_M1 pin to 1000

```
orangepi@orangepi:~$ gpio pwm 2 1000
```

5) After running the above command, you can observe through the oscilloscope that the PWM duty cycle changes from 25% (500/2000) to 50% (1000/2000)

3.17.1.3. Method of adjusting PWM frequency

The calculation formula of PWM frequency is as follows:

PWM frequency = clock source frequency / (frequency division factor * ARR)

In：

The default value of the clock source frequency is 24000000Hz.

The value range of the frequency division coefficient is 2 ~ 512, and the default value is 120.

The value range of ARR is 0 ~ 65535, and the default value is 1000.

The default value of PWM frequency is 24000000 / (120 * 1000) = 200Hz.

It should be noted that if the frequency division coefficient is set to an odd number, the actual frequency division coefficient is the set value minus one.

3.17.1.3.1. Method to adjust PWM frequency by setting frequency division coefficient

1) We can use the following command to set the frequency division coefficient of the PWM15_IR_M1 pin to 4

```
orangepi@orangepi:~$ gpio pwmc 2 4
```

2) According to the above formula, the calculated value of PWM frequency is 6000Hz. It can be observed through the oscilloscope that the measured value of PWM frequency is 6010Hz, and the error can be ignored.

3. 17. 1. 3. 2. Method of directly setting the PWM frequency

1) We can use the **gpio pwmTone** command to set the frequency of the PWM pin. For example, use the following command to set the PWM frequency of the PWM15_IR_M1 pin to 500Hz.

```
orangepi@orangepi:~$ gpio pwmTone 2 500
```

When setting the PWM frequency, you need to ensure:

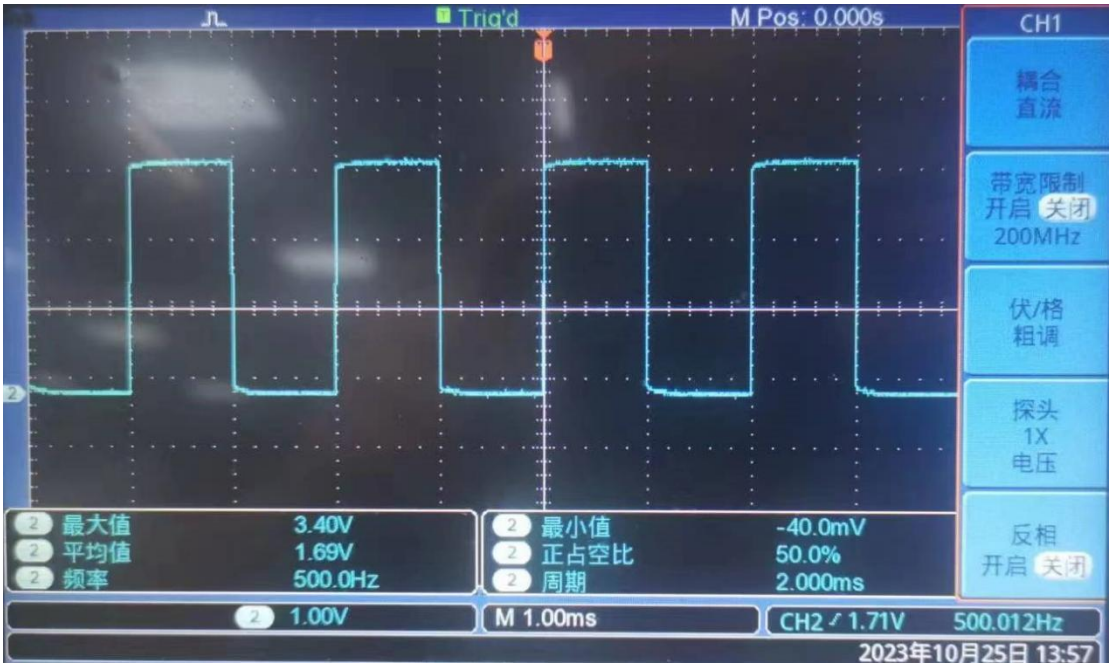
The set frequency value is < 24000000 / (frequency division factor * 2).

For example, the default frequency division coefficient is 120. If the frequency division coefficient is not modified, the set frequency value should be less than 100000Hz.

If the setting value is too large, the following error will appear:

gpio: The PWM frequency you set is too high to be possible

2) Then you can observe through the oscilloscope that the PWM frequency becomes 500Hz



3. 17. 2. How to use the PWM test program

1) In the example directory of wiringOP, there is a program named pwm.c. This program demonstrates how to operate PWM using the PWM-related API in wiringOP.

```
orangepi@orangepi:~$ cd wiringOP/examples/
orangepi@orangepi:~/wiringOP/examples$ ls pwm.c
pwm.c
```

2) The command to compile **pwm.c** into an executable program is as follows

```
orangepi@orangepi:~/wiringOP/examples$ gcc -o pwm pwm.c -lwiringPi
```

3) Then you can execute the PWM test program. When executing the PWM test program, you need to specify the PWM pin. For example, you can use the following command to test the PWM15_IR_M1 pin:

```
orangepi@orangepi:~/wiringOP/examples$ sudo ./pwm 2
```

4) After the pwm program is executed, the following contents will be tested in sequence:

a. Adjust the PWM duty cycle by setting ARR
b. Adjust the PWM duty cycle by setting CCR
c. Adjust the PWM frequency by setting the frequency division coefficient
d. Directly set the PWM frequency

5) After each test is completed, the current pwm waveform will be maintained for 5 seconds. After all test contents are completed, a new round of testing will be restarted.

6) The detailed execution process of the PWM test program is as follows:

a. Adjust the PWM duty cycle by setting ARR: You can observe through the oscilloscope that the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 50% to 25%, maintains it for 5 seconds, and then PWM The waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 25% to 50% and remains for 5 seconds.

b. Adjust the PWM duty cycle by setting CCR: You can observe through the oscilloscope that the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 50% to 100%, maintains it for 5 seconds, and then PWM The waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 100% to 50% and remains for 5 seconds.

c. Adjust the PWM frequency by setting the frequency division coefficient: It can be observed through the oscilloscope that the PWM frequency first changes to 12000Hz, and then the PWM waveform changes every 0.5 seconds. After changing 9 times, the PWM frequency changes to 1200Hz and remains for 5 seconds. Then the PWM waveform changes every 0.5 seconds. After changing 9 times, the PWM frequency changes to 12000Hz and remains for 5 seconds.

d. Directly set the PWM frequency: It can be observed through the oscilloscope that the PWM frequency first changes to 2000Hz, and then the PWM frequency increases by 2000Hz every two seconds. After changing 9 times, the PWM frequency changes to 20000Hz and remains for 5 seconds.

3. 18. How to install and use wiringOP-Python

wiringOP-Python is the Python language version of wiringOP, which is used to operate the hardware resources of the development board, such as GPIO, I2C, SPI and UART, in the Python program.

In addition, please note that all the following commands are operated under the root user.

3. 18. 1. wiringOP-Python installation method

1) First install the dependency package

```
root@orangepi:~# sudo apt-get update
root@orangepi:~# sudo apt-get -y install git swig python3-dev python3-setuptools
```

2) Then use the following command to download the source code of wiringOP-Python

Note that the following git clone--recursive command will automatically download the source code of wiringOP, because wiringOP-Python depends on wiringOP. Please make sure that the download process does not report an error due to network problems。

If there is a problem with the download code from GitHub, you can use the wiringOP-Python source code that comes with the Linux image directly, and the storage location is:/usr/src/wiringOP-Python

```
root@orangepi:~# git clone --recursive https://github.com/orangepi-xunlong/wiringOP-Python -b next
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# git submodule update --init --remote
```

3) Then use the following command to compile wiringOP-Python and install it into the Linux system of the development board

```
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# python3 generate-bindings.py > bindings.i
root@orangepi:~/wiringOP-Python# sudo python3 setup.py install
```

4) Then enter the following command. If there is a help information output, it means that Wiringop-Python is successfully installed. Press the **q** key to exit the interface of the help information

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi;  help(wiringpi)"
Help on module wiringpi:

NAME
    wiringpi
```

```
DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.
```

5) Test whether the wiringOP-Python is installed successfully under the Python command line is shown below：

a. First use the python3 command to enter the command line mode of python3

```
root@orangepi:~# python3
```

b. Then import the python module of wiringpi

```
>>> import wiringpi;
```

c. Enter the following command to view the help information of wiringOP-Python, and press the **q** key to exit the interface of the help information

```
>>> help(wiringpi)
Help on module wiringpi:

NAME
    wiringpi

DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.

CLASSES
    builtins.object
        GPIO
        I2C
        Serial
        nes
```

```
    class GPIO(builtins.object)
     |  GPIO(pinmode=0)
     |

>>>
```

3. 18. 2. 40pin GPIO port test

wiringOP-Python is the same as wiringOP, you can also determine which GPIO pin to operate by specifying the wPi number, because there is no command to check the wPi number in wiringOP-Python, so you can only check the board wPi number and physical Correspondence between pins.

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |    PWM15 |     IN | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
 |  118 |   5 | GPIO3_C6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO3_C7 | 6   | 119  |
 |  128 |   7 | GPIO4_A0 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |  130 |   8 |    TXD.7 |     IN | 0 | 15 || 16 | 0 | IN     | RXD.7    | 9   | 131  |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | IN     | GPIO4_A1 | 10  | 129  |
 |  138 |  11 | SPI3_TXD |     IN | 0 | 19 || 20 |   |        | GND      |     |      |
 |  136 |  12 | SPI3_RXD |     IN | 0 | 21 || 22 | 0 | IN     | TXD.9    | 13  | 132  |
 |  139 |  14 | SPI3_CLK |     IN | 0 | 23 || 24 | 0 | IN     | SPI3_CS1 | 15  | 134  |
 |      |     |      GND |        |   | 25 || 26 | 0 | IN     | GPIO3_D6 | 16  | 126  |
 |   32 |  17 |    SDA.3 |     IN | 1 | 27 || 28 | 1 | IN     | SCL.3    | 18  | 33   |
 |  133 |  19 |    RXD.9 |     IN | 0 | 29 || 30 |   |        | GND      |     |      |
 |  124 |  20 | GPIO3_D4 |     IN | 0 | 31 || 32 | 0 | IN     | PWM11    | 21  | 144  |
 |  127 |  22 | GPIO3_D7 |     IN | 0 | 33 || 34 |   |        | GND      |     |      |
 |  120 |  23 | GPIO3_D0 |     IN | 0 | 35 || 36 | 0 | IN     | GPIO3_D5 | 24  | 125  |
 |  123 |  25 | GPIO3_D3 |     IN | 0 | 37 || 38 | 0 | IN     | GPIO3_D2 | 26  | 122  |
 |      |     |      GND |        |   | 39 || 40 | 0 | IN     | GPIO3_D1 | 27  | 121  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
```

1) The following takes pin 7 — the corresponding GPIO is GPIO4_C3 — the corresponding wPi number is 2—as an example to demonstrate how to set the high and low levels of the GPIO port

```
root@orangepi3b:~# gpio readall
 +------+-----+----------+--------+---+   PI3B   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  140 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |  141 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |  147 |   2 |    PWM15 |     IN | 0 |  7 || 8  | 1 | ALT1   | RXD.2    | 3   | 25   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT1   | TXD.2    | 4   | 24   |
```

2) The steps to test directly with the command are as follows:

a. First set the GPIO port to output mode, where the first parameter of the **pinMode** function is the serial number of the wPi corresponding to the pin, and the second parameter is the GPIO mode

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
from wiringpi import GPIO; wiringpi.wiringPiSetup() ; \
wiringpi.pinMode(2, GPIO.OUTPUT) ;  "
```

b. Then set the GPIO port to output low level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 0v, it means that the low level is set successfully.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.LOW)"
```

c. Then set the GPIO port to output a high level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 3.3v, it means that the high level is set successfully.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.HIGH)"
```

3) The steps to test in the command line of python3 are as follows:

a. First use the python3 command to enter the command line mode of python3

```
root@orangepi:~# python3
```

b. Then import the python module of wiringpi

```
>>> import wiringpi
>>> from wiringpi import GPIO
```

c. Then set the GPIO port as the output mode. The first parameter of the **pinMode** function is the serial number of the wPi corresponding to the pin, and the second parameter is the GPIO mode.

```
>>> wiringpi.wiringPiSetup()
0
>>> wiringpi.pinMode(2, GPIO.OUTPUT)
```

d. Then set the GPIO port to output low level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 0v, it means that the

low level is set successfully.

```
>>> wiringpi.digitalWrite(2, GPIO.LOW)
```

e. Then set the GPIO port to output a high level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 3.3v, it means that the high level is set successfully.

```
>>> wiringpi.digitalWrite(2, GPIO.HIGH)
```

4) The method of wiringOP-Python setting GPIO high and low levels in python code can refer to the **blink.py** test program in the examples below. The **blink.py** test program will set the voltage of all GPIO ports in the 40 pins of the development board to change continuously.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# ls blink.py
blink.py
root@orangepi:~/wiringOP-Python/examples# python3 blink.py
```

3.18.3. 40pin SPI test

1) According to the table below, the spi available for Orange Pi 3B is spi3

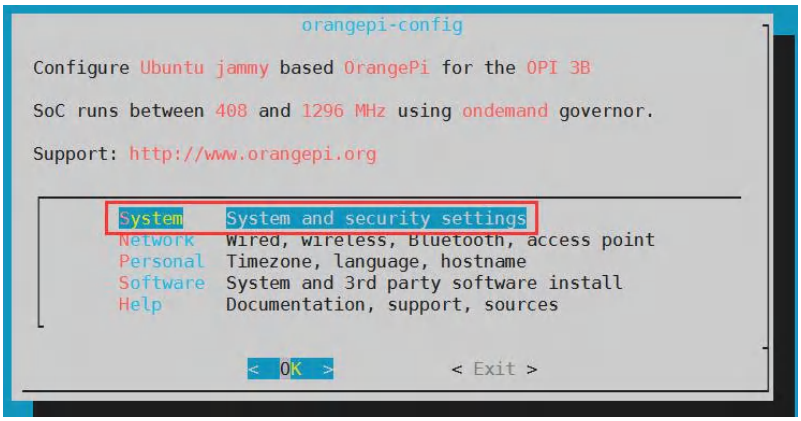
复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	126	GPIO3_D6		
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

2) In the linux system, the SPI in the 40 pin is closed by default, and it needs to be opened manually before it can be used. The detailed steps are as follows:

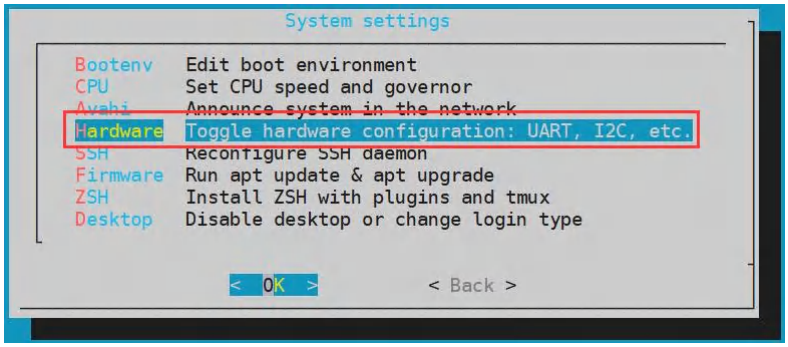
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

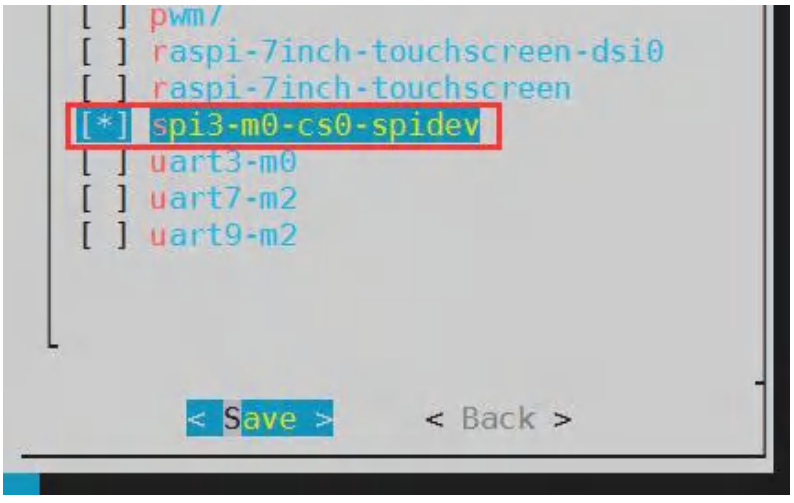
b. Then select **System**



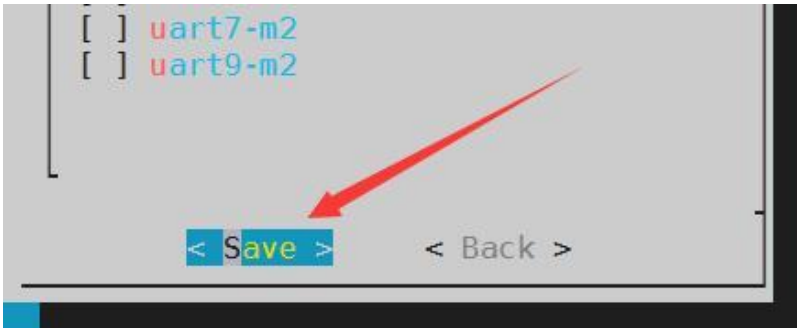
c. Then select **Hardware**



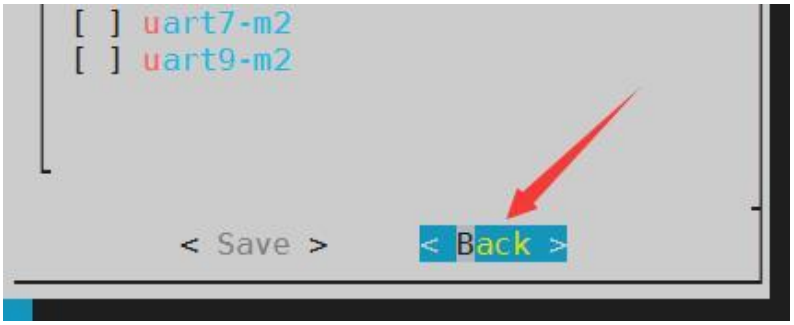
d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the SPI configuration you want to open



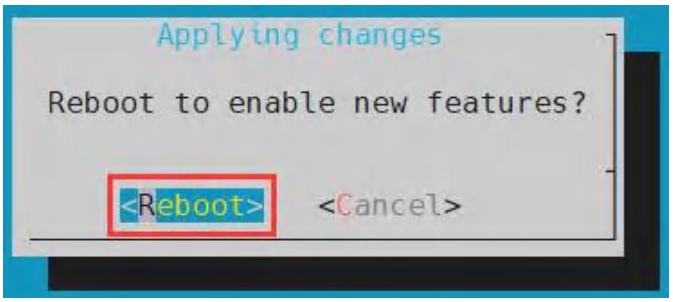
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



3) First check whether there is a **spidev3.0** device node in the Linux system. If it exists, it means that SPI3 has been set up and can be used directly

```
orangepi@orangepi:~$ ls /dev/spidev3.0
/dev/spidev3.0
```

4) Then you can use the **spidev_test.py** program in the examples to test the loopback function of the SPI. The **spidev_test.py** program needs to specify the following two parameters:

a. **--channel:** Specify the channel number of SPI
b. **--port:** Specify the port number of the SPI

5) Do not short-circuit the mosi and miso pins of SPI3, the output of running

spidev_test.py is as follows, you can see that the data of TX and RX are inconsistent

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py \
--channel 3 --port 0
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev3.0
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    |......@.....|
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF    |................|
```

6) Then use the Dupont wire to short-circuit the two pins of txd (pin 19 in the 40pin interface) and rxd (pin 21 in the 40pin interface) of SPI3 and then run the output of spidev_test.py as follows, you can see The data sent and received are the same, indicating that the SPI3 loopback test is normal

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py \
--channel 3 --port 0
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev3.0
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    |......@.....|
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    |......@.....|
```

3. 18. 4. 4.40pin I2C test

1) As can be seen from the table below, the available i2c for Orange Pi 3B is i2c2, i2c3 and i2c4, a total of three sets of i2c buses

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	126	GPIO3_D6		
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

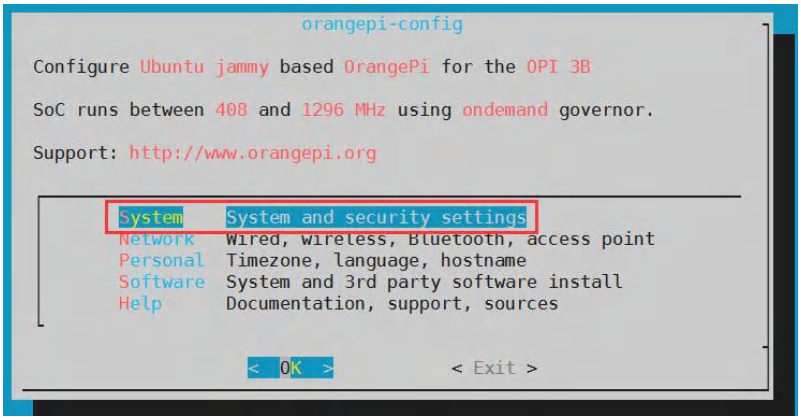
As can be seen from the above table, i2c4_m0 and spi3_m0 are multiplexed pins, and both cannot be opened at the same time. i2c3_m0 and uart3_m0 are also multiplexed pins, and both cannot be opened at the same time

2) In the linux system, the I2C bus in the 40 pin is closed by default, and it needs to be opened manually to use it. The detailed steps are as follows:

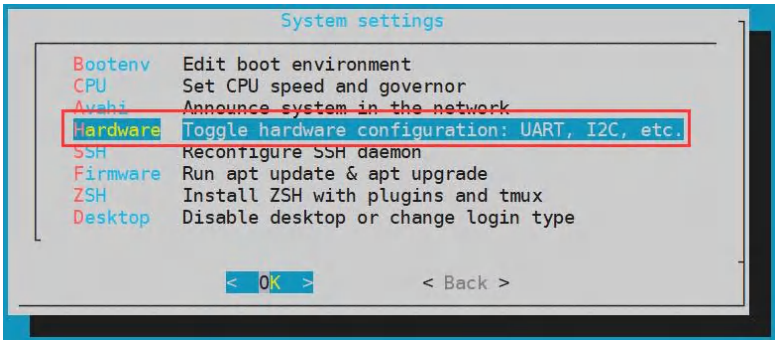
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

b. Then select **System**

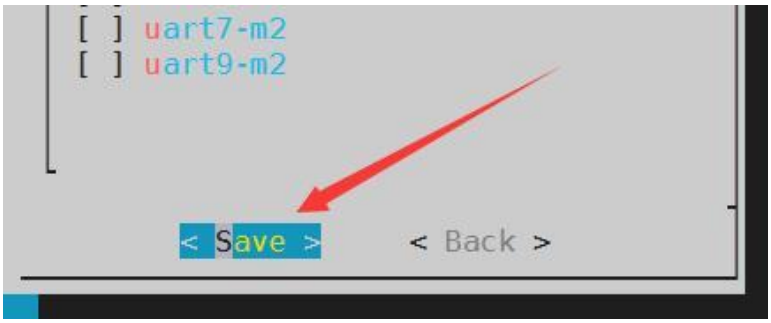


c. Then select **Hardware**

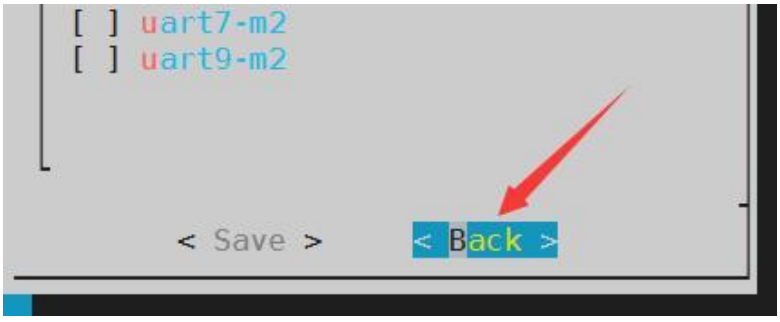


d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the I2C configuration you want to open

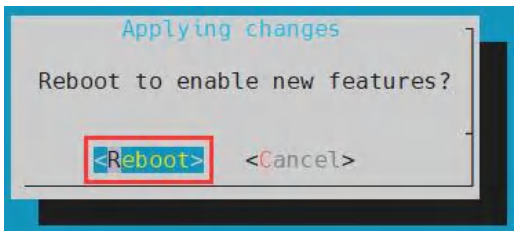
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect

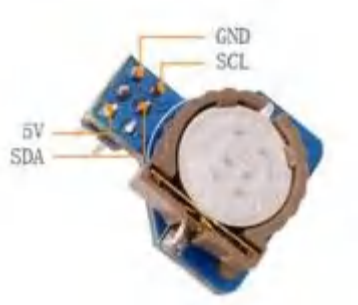


3) After starting the Linux system, first confirm that there is an i2c device node under /dev

```
orangepi@orangepi:~# ls /dev/i2c-*
/dev/i2c-0  /dev/i2c-2  /dev/i2c-3  /dev/i2c-4  /dev/i2c-6
```

4) Then connect a I2C device on the I2C pin of the 40PIN connector. Here is an example of the DS1307 RTC module.

	i2c2-m1	i2c3-m0	i2c4-m0
Sda Pin	Corresponding to No. 3 pin	Corresponding to No. 27 pin	Corresponding to No. 19 pin
Sck Pin	Corresponding to No. 5 pin	Corresponding to No. 28 pin	Corresponding to No. 23 pin
Vcc Pin	Corresponding to No. 1 pin	Corresponding to No. 1 pin	Corresponding to No. 1 pin
Gnd Pin	Corresponding to No. 6 pin	Corresponding to No. 6 pin	Corresponding to No. 6 pin



5) Then use the **i2cdetect -y** command If the address of the connected I2C device can be detected, it means that the I2C can be used normally

```
orangepi@orangepi:~$ sudo i2cdetect -y 2      #i2c2 command
orangepi@orangepi:~$ sudo i2cdetect -y 3      #i2c3 command
orangepi@orangepi:~$ sudo i2cdetect -y 4      #i2c4 command
```

```
root@orangepi3b:~# i2cdetect -y 4
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

6) Then you can run the **ds1307.py** test program in the **examples** to read the RTC time

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 ds1307.py --device \
"/dev/i2c-4"
```

```
Thu 2023-01-05  14:57:55
Thu 2023-01-05  14:57:56
Thu 2023-01-05  14:57:57
^C
exit
```

3.18.5. 40pin's UART test

1) As can be seen from the table below, the available uarts for Orange Pi 3B are uart2, uart3, uart7 and uart9, a total of four groups of uart buses, of which uart2 is the system's debugging serial port by default.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	I2C2_SDA_M1	GPIO4_B4	140	3	4		5V		
	I2C2_SCL_M1	GPIO4_B5	141	5	6		GND		
	PWM15_IR_M1(fe700030)	GPIO4_C3	147	7	8	25	GPIO0_D1	UART2_TX_M0	
		GND		9	10	24	GPIO0_D0	UART2_RX_M0	
		GPIO3_C6	118	11	12	119	GPIO3_C7		
		GPIO4_A0	128	13	14		GND		
	UART7_TX_M2	GPIO4_A2	130	15	16	131	GPIO4_A3	UART7_RX_M2	
		3.3V		17	18	129	GPIO4_A1		
I2C4_SDA_M0	SPI3_MOSI_M0	GPIO4_B2	138	19	20		GND		
	SPI3_MISO_M0	GPIO4_B0	136	21	22	132	GPIO4_A4	UART9_TX_M2	
I2C4_SCL_M0	SPI3_CLK_M0	GPIO4_B3	139	23	24	134	GPIO4_A6	SPI3_CS0_M0	
		GND		25	26	126	GPIO3_D6		
UART3_RX_M0	I2C3_SDA_M0	GPIO1_A0	32	27	28	33	GPIO1_A1	I2C3_SCL_M0	UART3_TX_M0
	UART9_RX_M2	GPIO4_A5	133	29	30		GND		
		GPIO3_D4	124	31	32	144	GPIO4_C0	PWM11_IR_M1(fe6f0030)	
		GPIO3_D7	127	33	34		GND		
		GPIO3_D0	120	35	36	125	GPIO3_D5		
		GPIO3_D3	123	37	38	122	GPIO3_D2		
		GND		39	40	121	GPIO3_D1		

As can be seen from the above table, i2c3_m0 and uart3_m0 are pin-multiplexed, and both cannot be turned on at the same time.

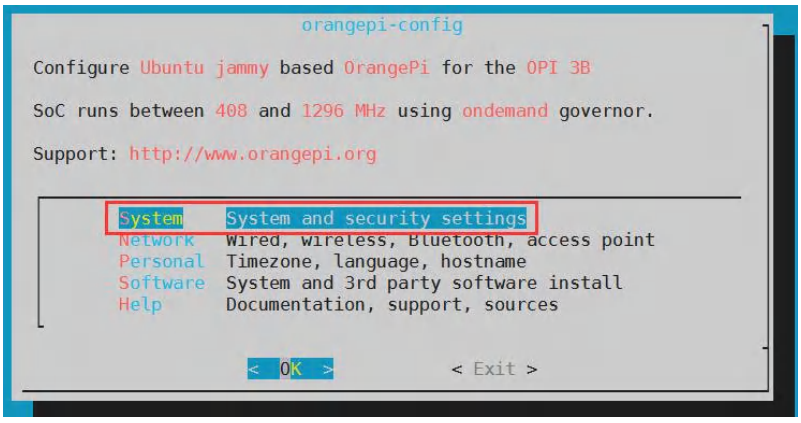
Please note that uart2_m0 is used as the debugging serial port of the system by default. If the configuration of uart2 is turned on, the debugging serial port function will not be available.

2) In the linux system, the UART in the 40 pins is closed by default, and it needs to be opened manually before it can be used. The detailed steps are as follows:

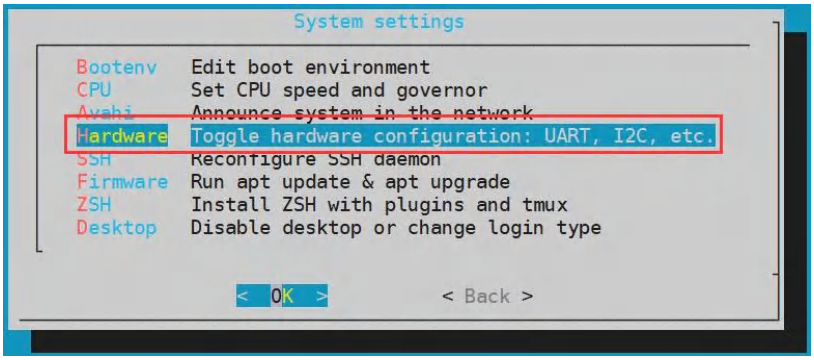
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

b. Then select **System**

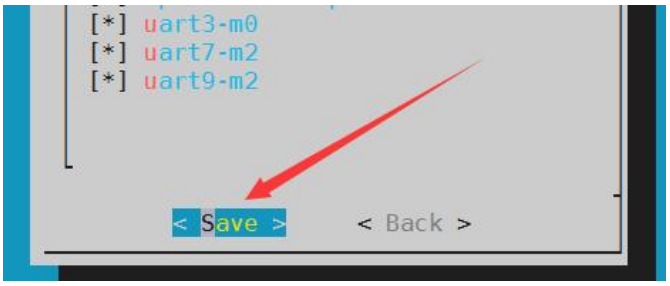


c. Then select **Hardware**

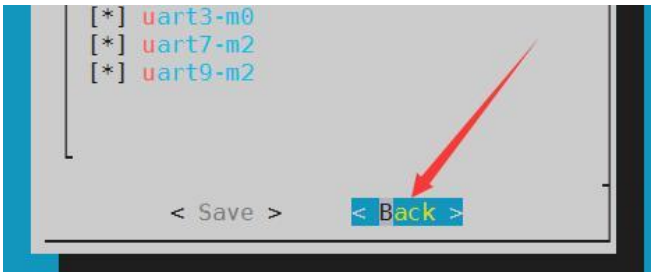


d. Then use the arrow keys on the keyboard to navigate to the position shown in the figure below, and then use the **space** to select the UART configuration you want to open

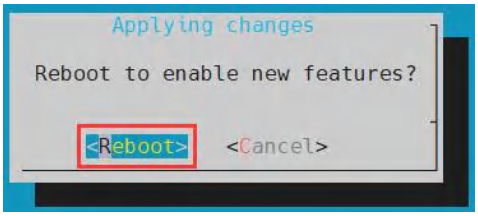
e. Then select **<Save>**to save



f. Then select **<Back>**



g. Then select **<Reboot>**to restart the system to make the configuration take effect



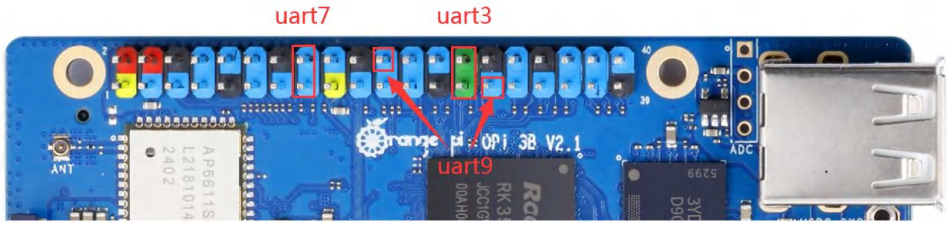
3) After entering the Linux system, first confirm whether there is a device node corresponding to uart under /dev

```
orangepi@orangepi:~# ls /dev/ttyS*
/dev/ttyS1   /dev/ttyS3   /dev/ttyS7   /dev/ttyS9
```

Note that /dev/ttyS0 in linux6.6 is uart9.

4) Then start to test the uart interface, first use the DuPont line to short the rx and tx of the uart interface to be tested

	uart3	uart7	uart9
tx Pin	Corresponding to the 28 pin	Corresponding to the 16 pin	Corresponding to the 29 pin
rx Pin	Corresponding to the 27 pin	Corresponding to the 15 pin	Corresponding to the 22 pin



5) Use the **serialTest.py** program in the examples to test the loopback function of the serial port as shown below. If you can see the following print, it means that the serial port communication is normal

a. Test UART3

```
root@orangepi:~/wiringOP-Python/examples# python3 serialTest.py --device \
"/dev/ttyS3"

Out:    0: ->    0
Out:    1: ->    1
Out:    2: ->    2
Out:    3: ->    3
Out:    4:^C
exit
```

b. Test UART7

```
root@orangepi:~/wiringOP-Python/examples# python3 serialTest.py --device \
"/dev/ttyS7"

Out:    0: ->    0
Out:    1: ->    1
Out:    2: ->    2
Out:    3: ->    3
Out:    4:^C
exit
```

c. Test UART9

```
root@orangepi:~/wiringOP-Python/examples# python3 serialTest.py --device \
"/dev/ttyS9"

Out:    0: ->    0
Out:    1: ->    1
Out:    2: ->    2
Out:    3: ->    3
Out:    4:^C
exit
```

3.19. Hardware watch the door dog test

Note that linux6.6 is not supported yet.

The watchdog_test program is pre-installed in the Linux system released by Orange Pi, which can be tested directly.

The method to run the watchdog_test program is as follows:

a. The second parameter 10 indicates the counting time of the watchdog. If the dog is not fed within this time, the system will restart
b. We can feed the dog by pressing any key on the keyboard (except ESC). After feeding the dog, the program will print a line of keep alive to indicate that the dog is fed successfully

```
orangepi@orangepi:~$ sudo watchdog_test 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
put_usr return,if 0,success:0
keep alive
keep alive
keep alive
```

3.20. Check the serial number of the RK3566 chip

The command to view the serial number of the RK3566 chip is as follows. The serial number of each chip is different, so the serial number can be used to distinguish multiple development boards.

```
orangepi@orangepi:~$ cat_serial.sh
Serial          : 8fa18eaf489041f0
```

3.21. The method of downloading and installing the balenaEtcher version of arm64

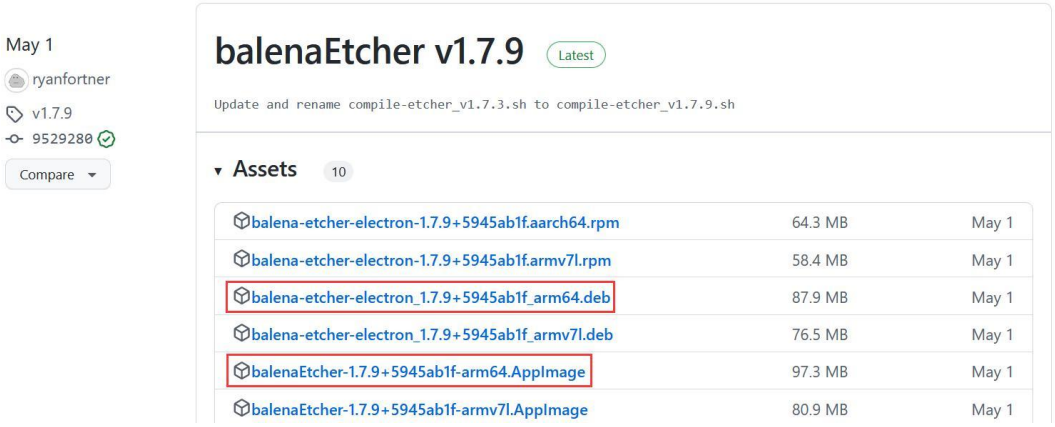
1) The download address of Balenaetcher ARM64 version is：
   a. The download address of the deb installation package is as follows, which needs

to be installed before it can be used

```
https://github.com/Itai-Nelken/BalenaEtcher-arm/releases/download/v1.7.9/balena-etcher-electron_1.7.9+5945ab1f_arm64.deb
```

b. The download address of the AppImage version that does not need to be installed is as follows:

```
https://github.com/Itai-Nelken/BalenaEtcher-arm/releases/download/v1.7.9/balenaEtcher-1.7.9+5945ab1f-arm64.AppImage
```

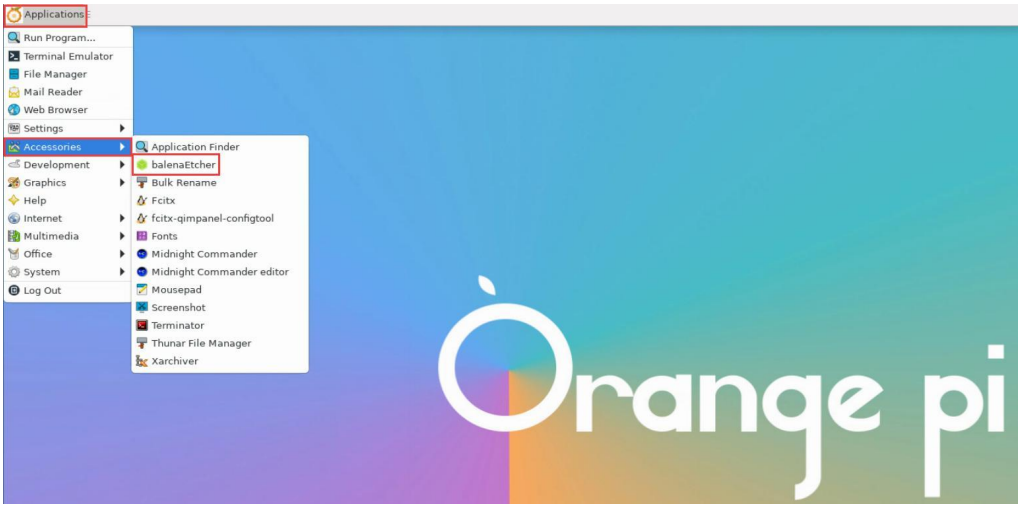


2) How to install and use the deb version of Balenaetcher：

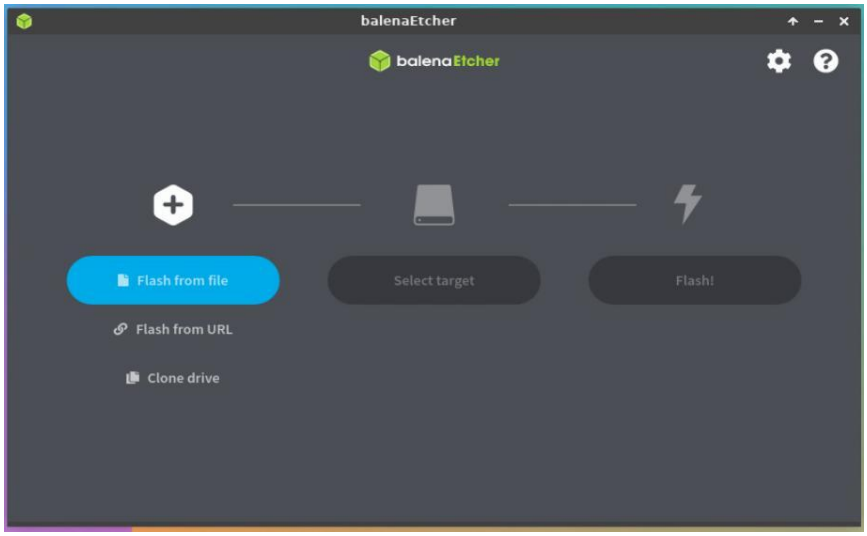
a. The deb version of balenaEtcher installation command is as follows:

```
orangepi@orangepi:~$ sudo apt install -y  \
--fix-broken ./balena-etcher-electron_1.7.9+5945ab1f_arm64.deb
```

b. After the deb version of balenaEtcher is installed, it can be opened in the Application



c. The interface after balenaEtcher is opened is as follows:

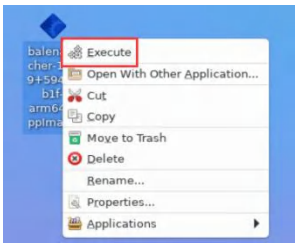


3) How to use the AppImage version of balenaEtcher：

a. First add permissions to Balenaetcher

```
orangepi@orangepi:~/Desktop$ chmod +x balenaEtcher-1.7.9+5945ab1f-arm64.AppImage
```

b. Then select the AppImage version balenaEtcher right-click the mouse, and then click Execute to open balenaEtcher



3. 22. The installation method of the Bt-Panel Linux panel

Bt-Panel Linux panel is a server management software that improves operation and maintenance efficiency, and supports one-click configuration of more than 100 server management functions such as LAMP/LNMP/cluster/monitoring/website/FTP/database/JAVA (excerpted from the official website of the Bt-Panel)

1) First, you need to expand the size of the **/tmp** space. After setting, you need to **restart the linux system of the development board**. The command is as follows:

```
orangepi@orangepi:~$ sudo sed -i 's/nosuid/&,size=2G/' /etc/fstab
```

```
orangepi@orangepi:~$ sudo reboot
```

2) After restarting, you can see that the size of the **/tmp** space has become 2G.

```
orangepi@orangepi:~$ df -h | grep "/tmp"
tmpfs                   2.0G     12K    2.0G      1% /tmp
```

3) Then enter the following command in the Linux system to start the installation of the Bt-Panel

```
orangepi@orangepi:~$ sudo install_bt_panel.sh
```

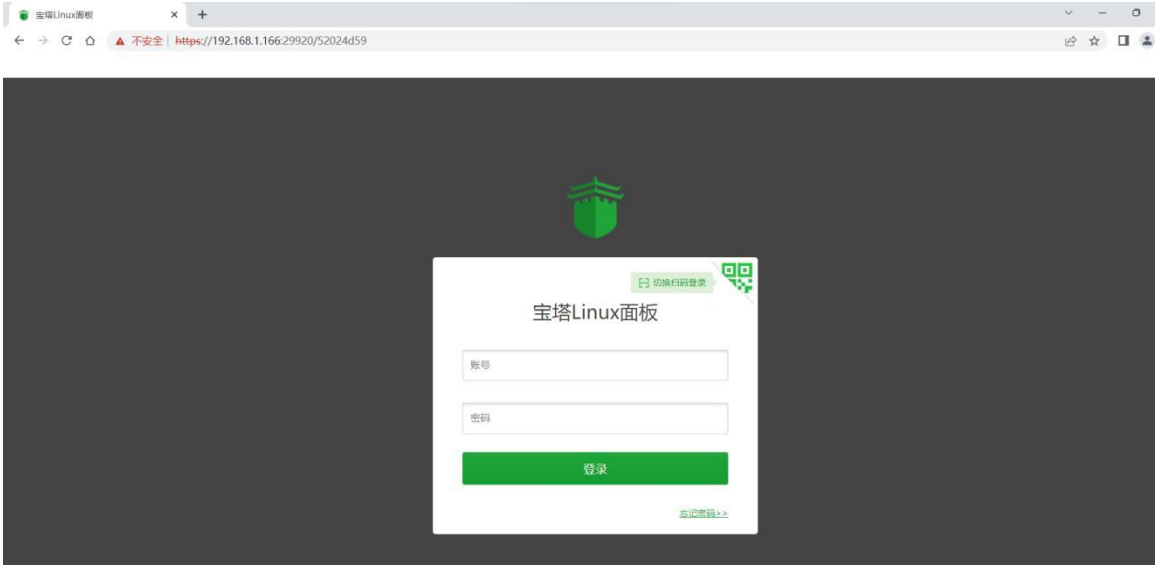
4) Then the Bt-Panel installation program reminds whether to install the **Bt-Panel** to the **/www** folder, and enter y at this time

```
+----------------------------------------------------------------------
| Bt-WebPanel FOR CentOS/Ubuntu/Debian
+----------------------------------------------------------------------
| Copyright © 2015-2099 BT-SOFT(http://www.bt.cn) All rights reserved.
+----------------------------------------------------------------------
| The WebPanel URL will be http://SERVER_IP:8888 when installed.
+----------------------------------------------------------------------

Do you want to install Bt-Panel to the /www directory now?(y/n): y
```

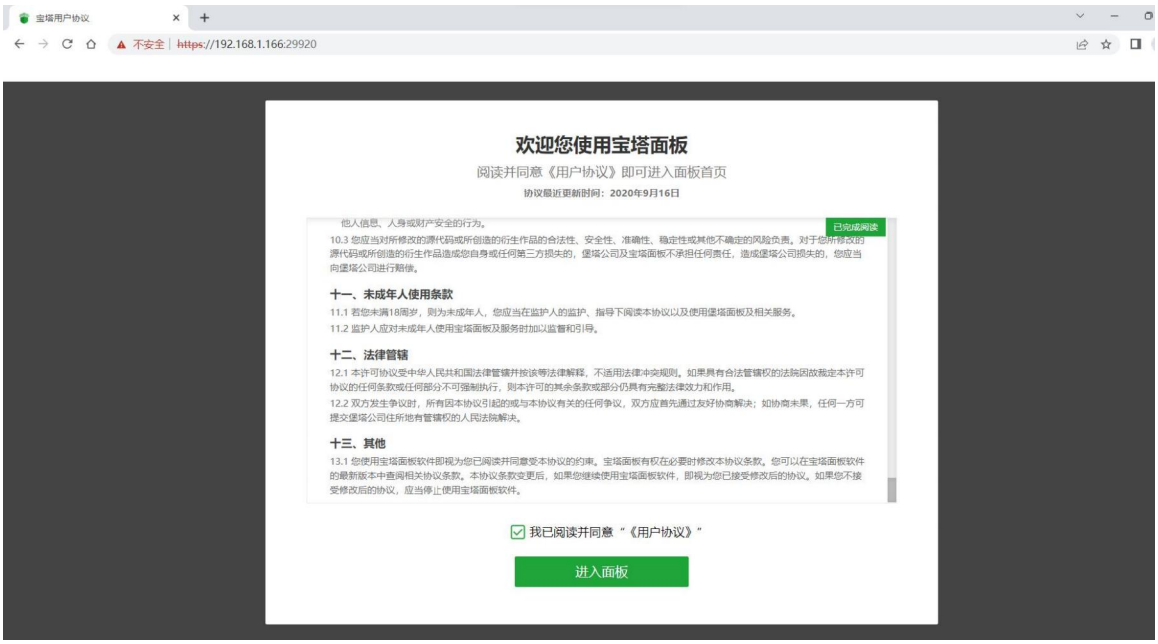
5) Then you have to wait patiently. When you see the printing information below the terminal output, it means that the Bt-Panel has been installed. The entire installation process takes about 34 minutes. There may be some differences according to the difference in network speed

```
==========================================================
Congratulations! Installed successfully!
==========================================================
外网面板地址: https://183.15.204.194:29920/52024d59
内网面板地址: https://192.168.1.166:29920/52024d59
username: 4qhagfrc
password: 27b2d026
If you cannot access the panel,
release the following panel port [29920] in the security group
若无法访问面板，请检查防火墙/安全组是否有放行面板[29920]端口
因已开启面板自签证书，访问面板会提示不匹配证书，请参考以下链接配置证书
https://www.bt.cn/bbs/thread-105443-1-1.html
==========================================================
Time consumed: 34 Minute!
orangepi@orangepi:~$
```

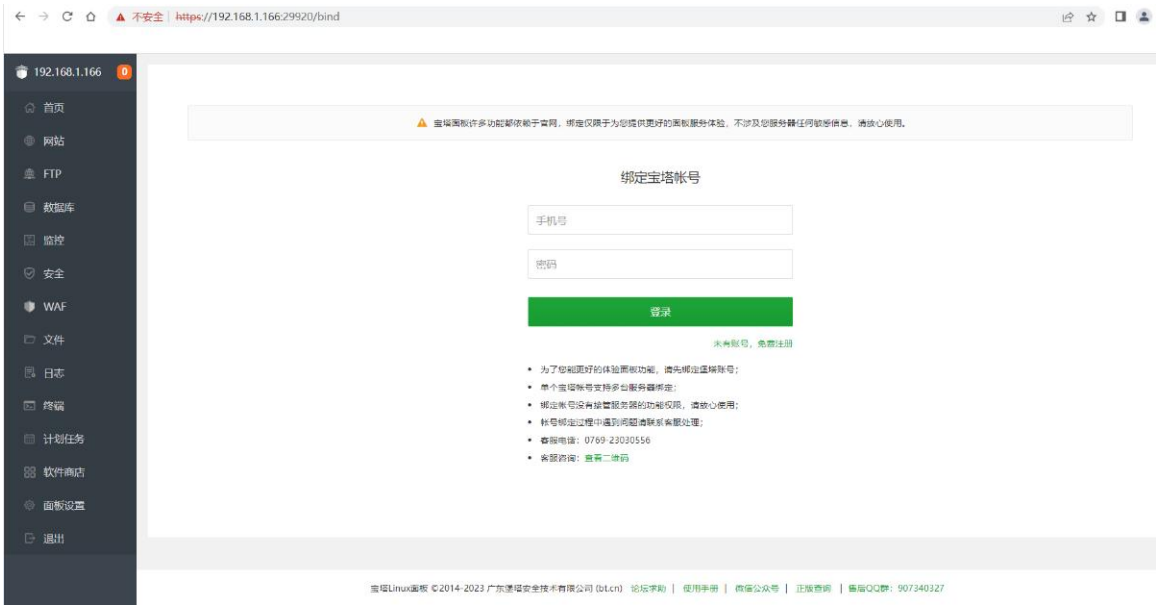
6) At this time, enter the **panel address** displayed above in the browser to open the login interface of the Bt-Panel Linux panel, and then enter the **username** and **password** displayed in the corresponding position to log in to the Bt-Panel



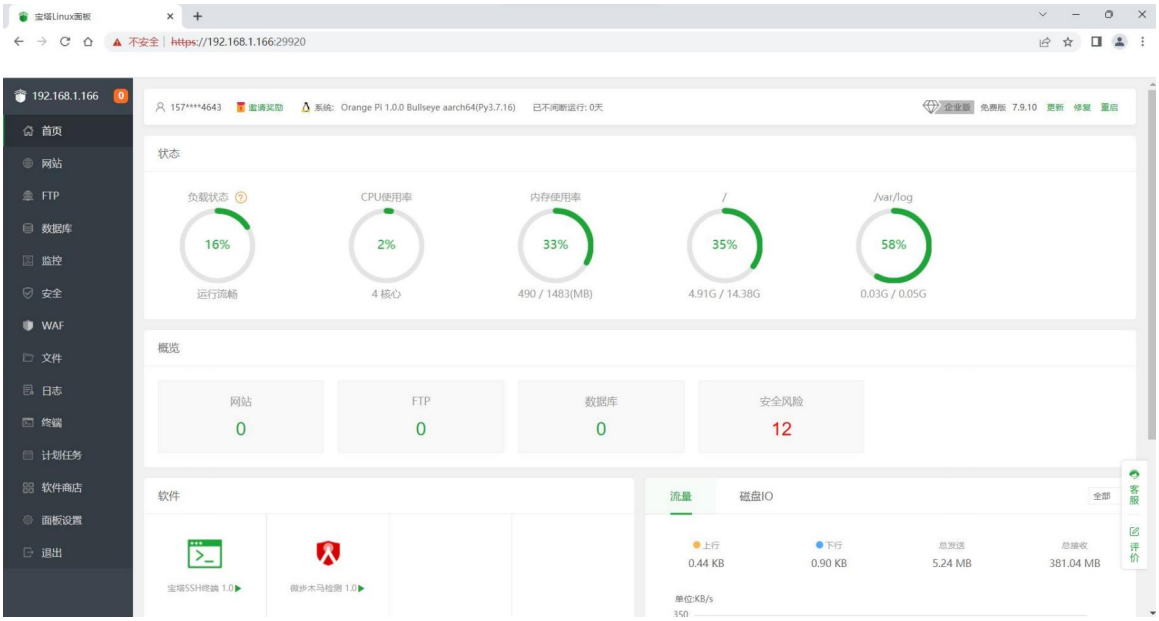
7) After successfully logging into the Bt-Panel, the following welcome interface will pop up. First, please read the user notice in the middle and drag it to the bottom, then you can select "I have agreed and read the "User Agreement"", and then click "Enter the Panel" You can enter the Bt-Panel



8) After entering the Bt-Panel, you will first be prompted to bind an account on the official website of the Bt-Panel. If you do not have an account, you can go to the official website of the Bt-Panel (**https://www.bt.cn**) to register one



9) The final display interface is shown in the figure below. You can intuitively see some status information of the development board Linux system, such as load state, CPU usage, memory usage and storage space usage



10) More functions of the Bt-Panel can refer to the following information to explore by yourself

Manual：**http://docs.bt.cn**
Forum address：**https://www.bt.cn/bbs**
GitHub Link：**https://github.com/aaPanel/BaoTa**

3. 23. Set the Chinese environment and install Chinese input method

Note that before installing the Chinese input method, please make sure that the Linux system used in the development board is the desktop version system。

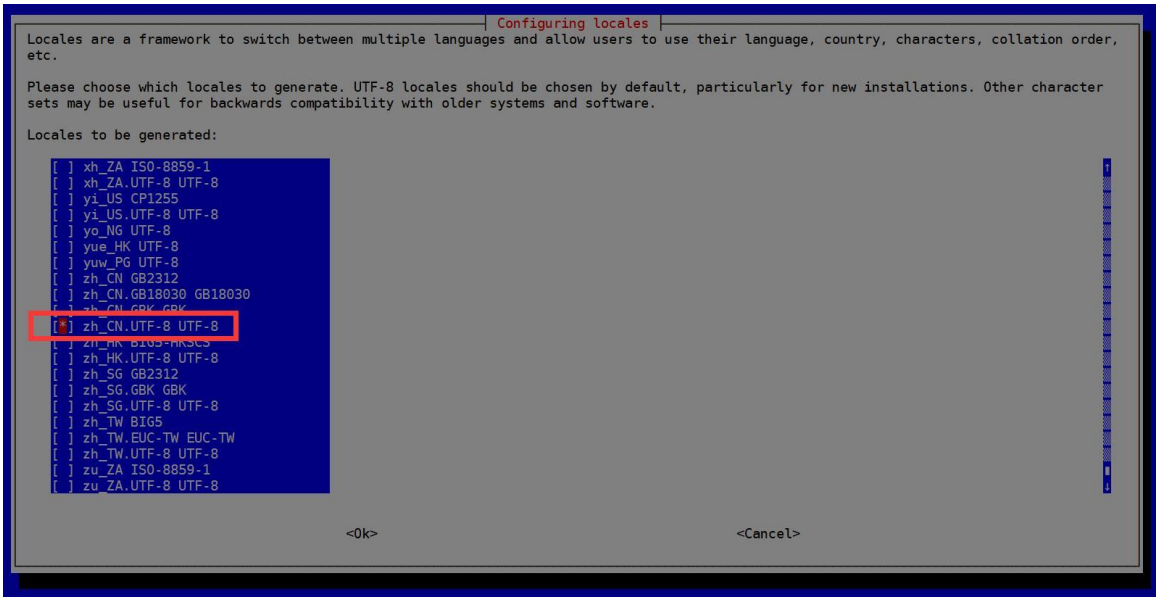
3. 23. 1. Debian system installation method

1) First set the default **locale** as Chinese

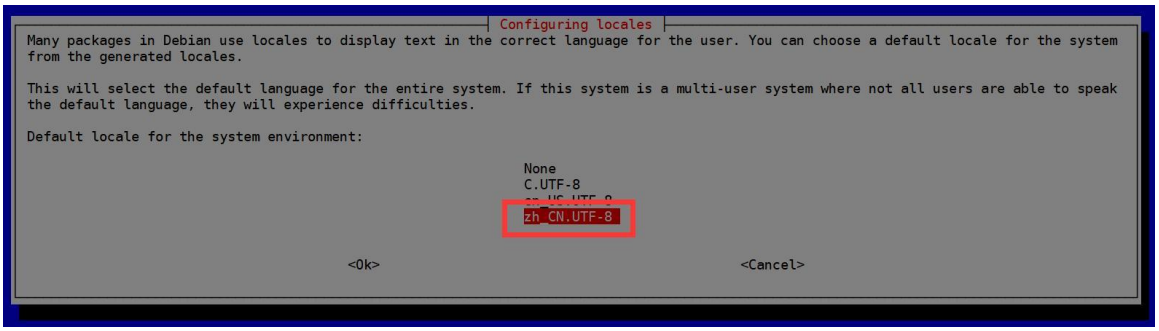
a. Enter the command below to start configured **locale**

```
orangepi@orangepi:~$ sudo dpkg-reconfigure locales
```

b. Then select **zh_CN.UTF-8 UTF-8** in the pop-up interface (to move up and down through the upper and lower direction buttons on the keyboard, select it through the space key, and finally move the cursor to **<OK>** through the TAB key, then press the ENTER key)



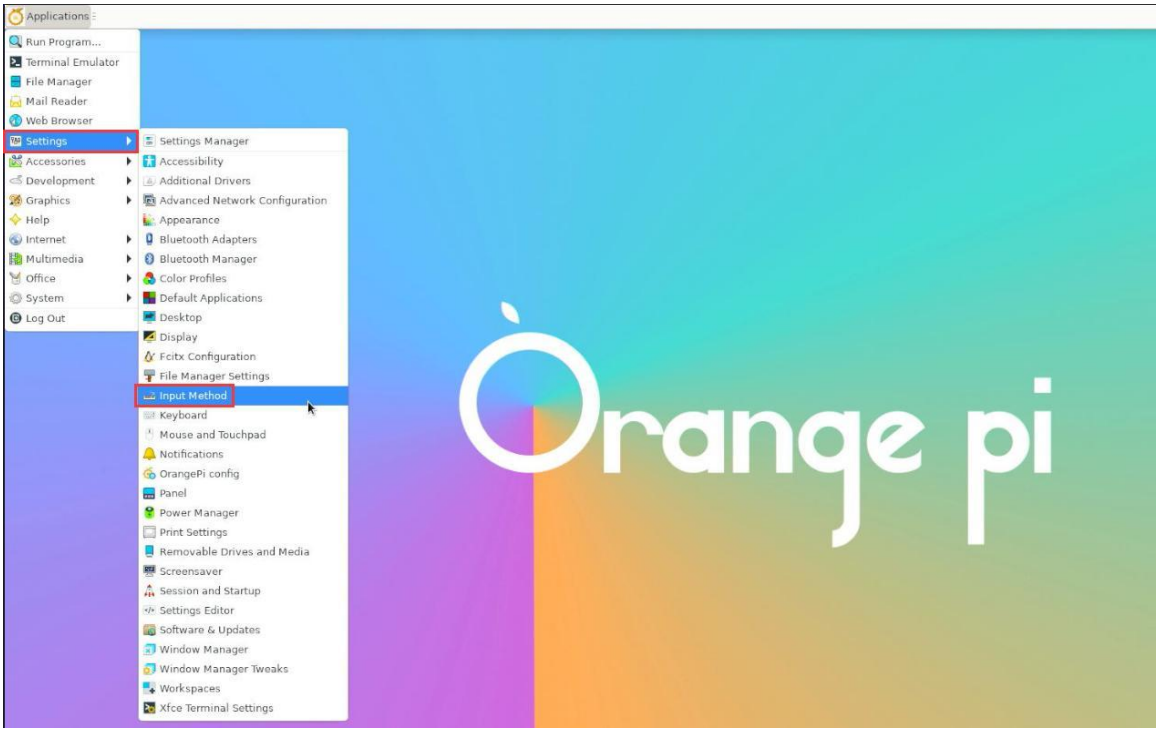
c. Then set the default **locale** as **zh_CN.UTF-8**



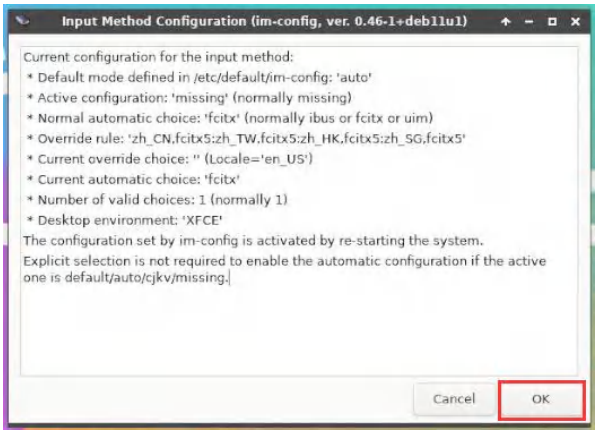
d. After exiting the interface, the **locale** settings will be started. The output displayed by the command line is shown below

```
orangepi@orangepi:~$ sudo dpkg-reconfigure locales
Generating locales (this might take a while)...
  en_US.UTF-8... done
  zh_CN.UTF-8... done
Generation complete.
```

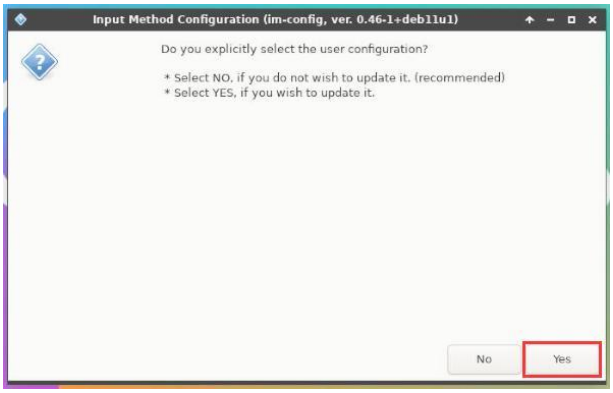
2) Then open **Input Method**



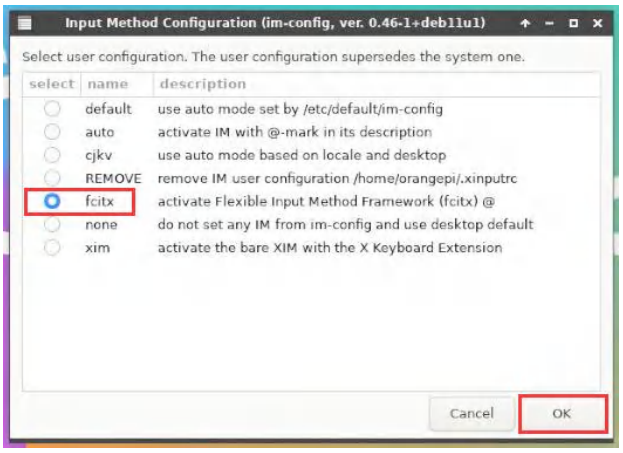
3) Then select **OK**



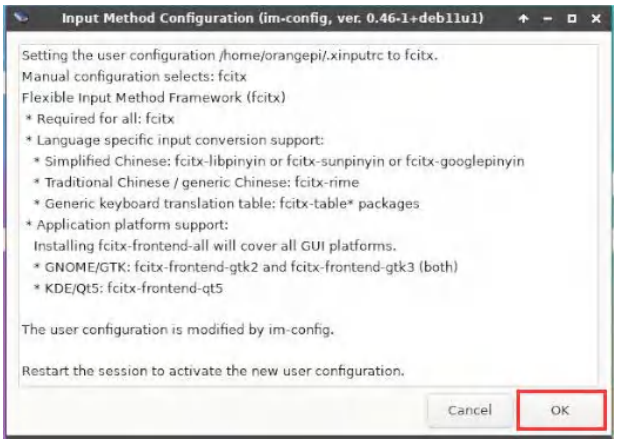
4) Then select **Yes**



5) Then select **fcitx**

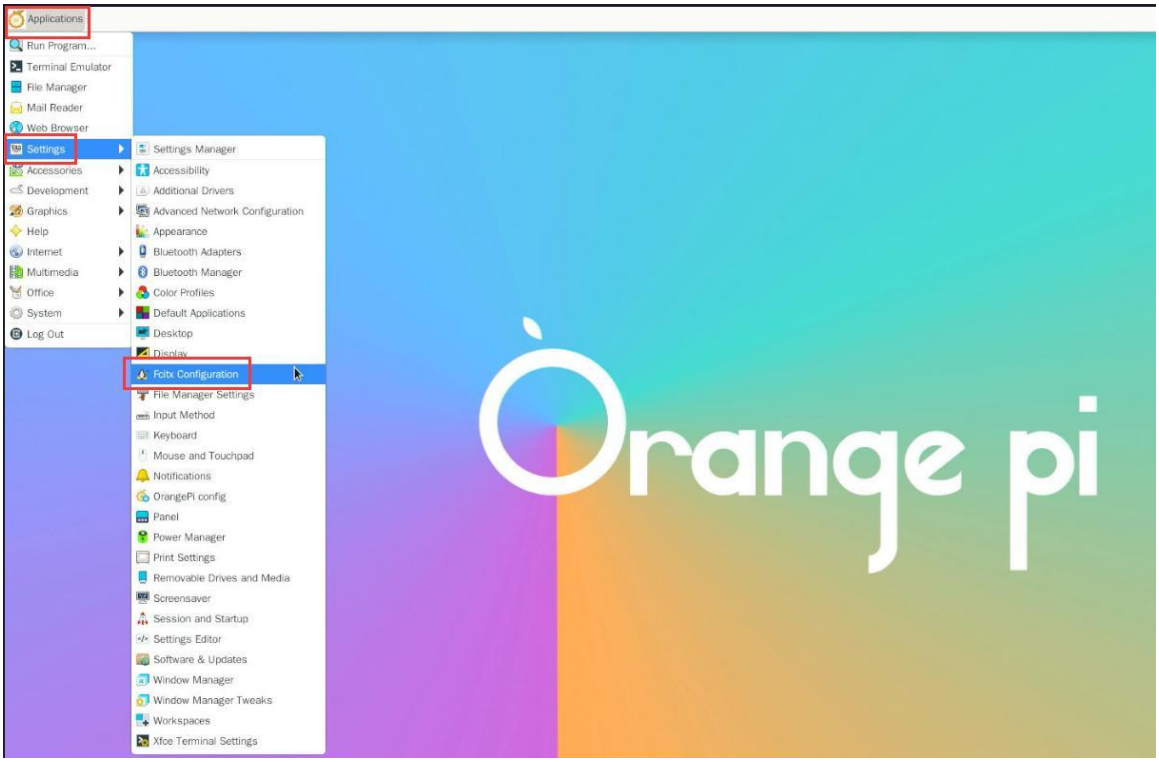


6) Then select **OK**

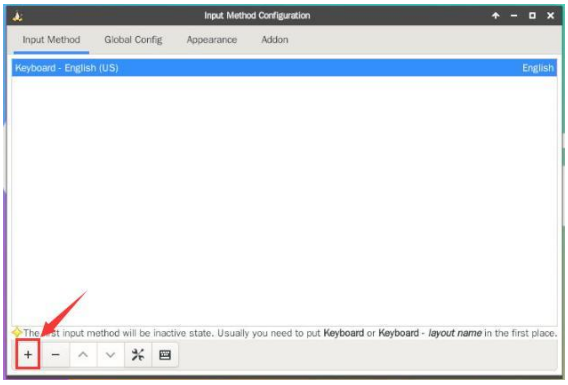


7) **Then restart the Linux system to make the configuration effective**

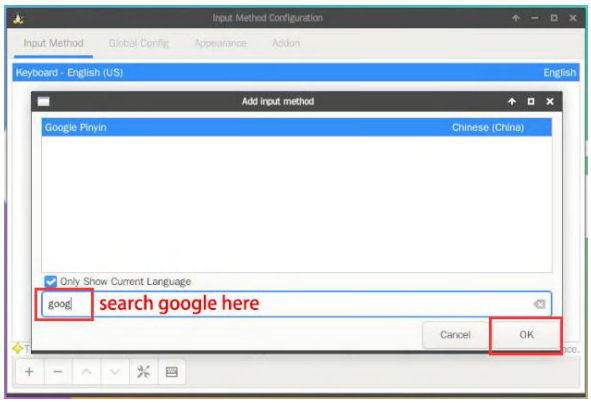
8) Then open **Fcitx configuration**



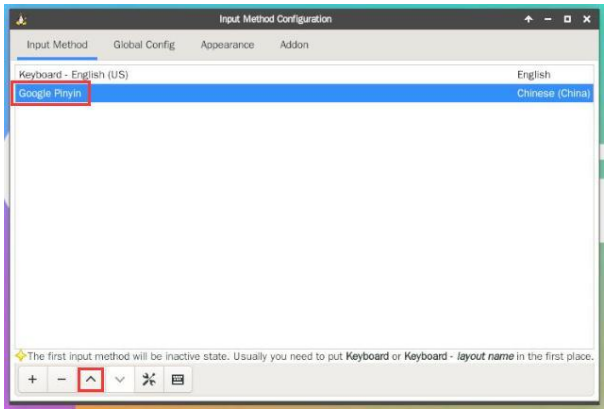
9) Then click the + of the position shown in the figure below

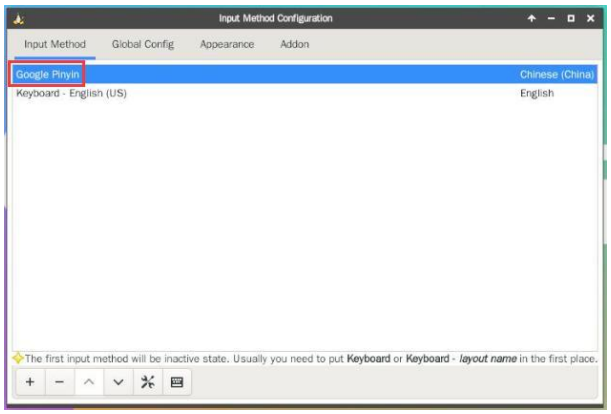


10) Then search **Google Pinyin** and click **OK**



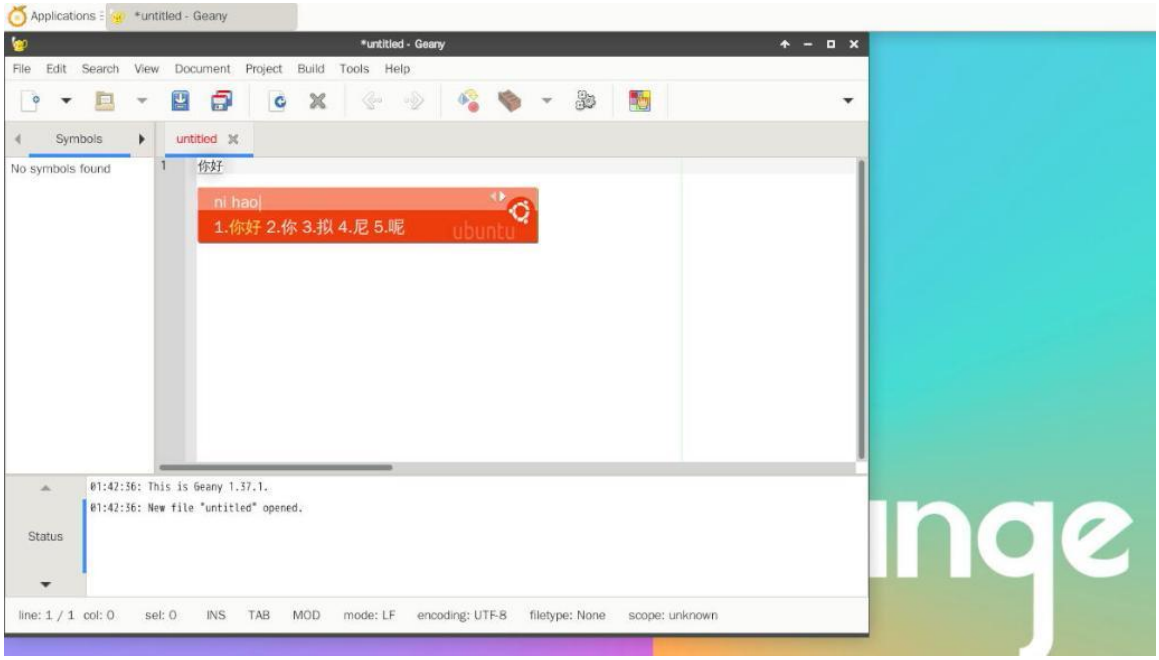
11) Then put **Google Pinyin** to the front





12) Then open the **Geany** editor to test the Chinese input method

13) The Chinese input method test is shown below



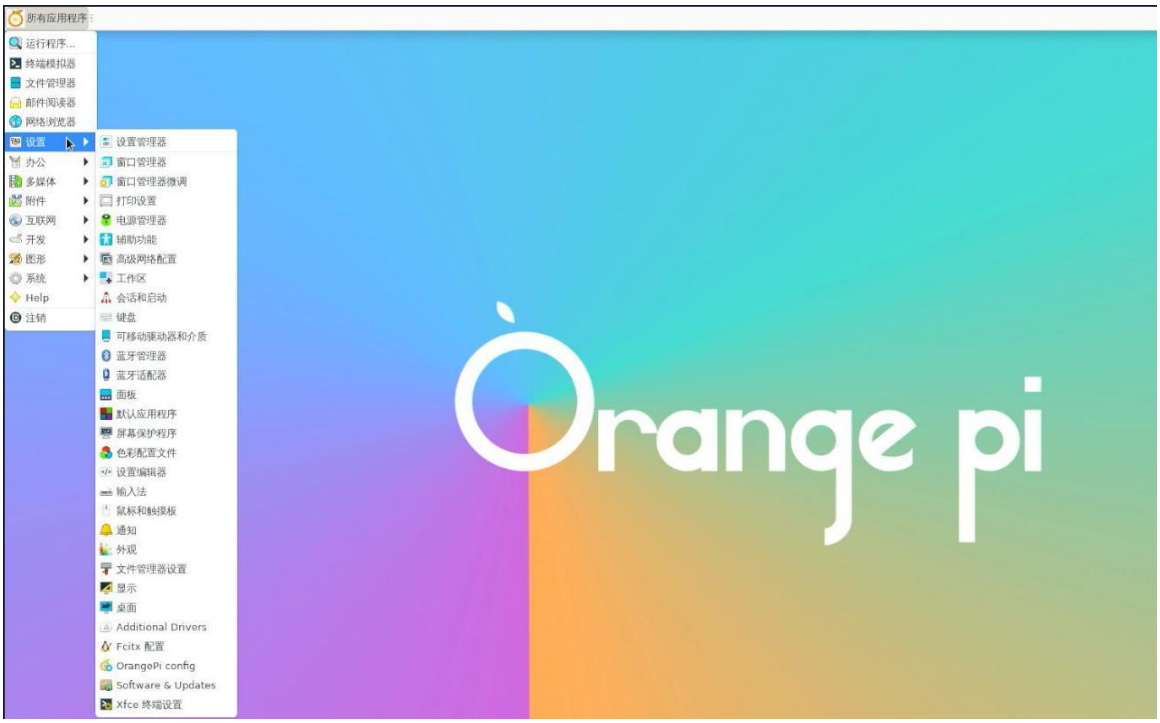
14) You can switch the Chinese and English input method through the **Ctrl+Space** shortcut

15) If the entire system is required as Chinese, the variables in **/etc/default/locale** can be set to **zh_CN.UTF-8**

```
orangepi@orangepi:~$ sudo vim /etc/default/locale
#    File generated by update-locale
LC_MESSAGES=zh_CN.UTF-8
LANG=zh_CN.UTF-8
```

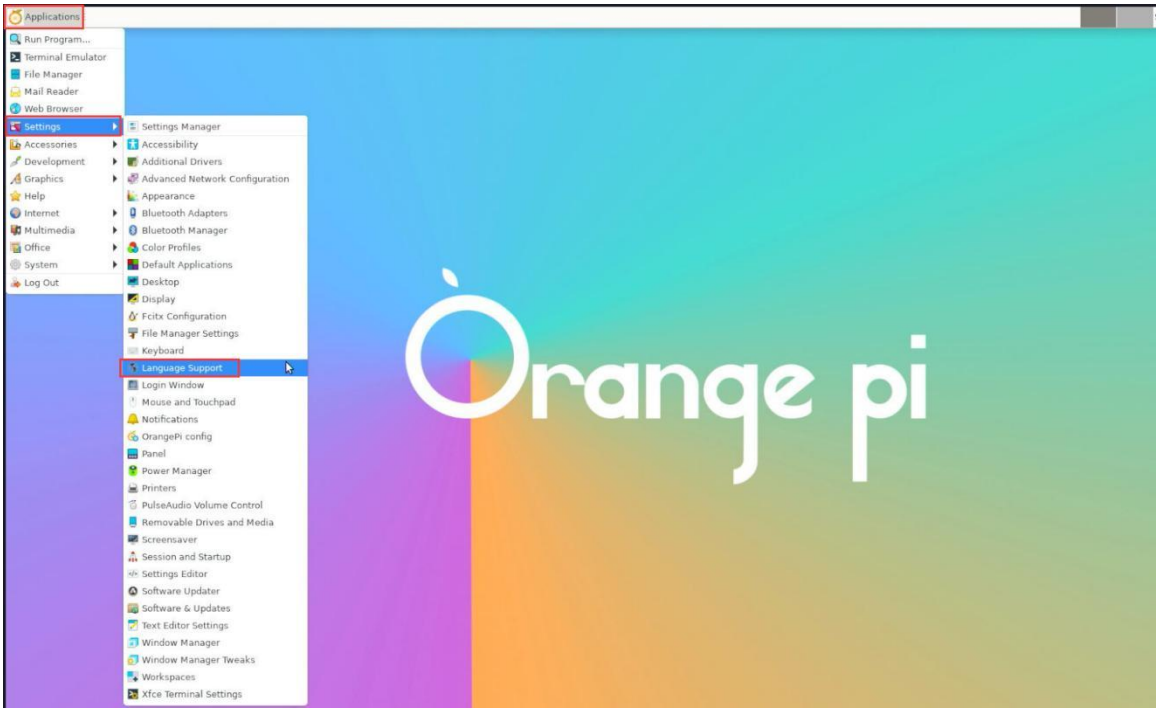
```
LANGUAGE=zh_CN.UTF-8
```

16) Then **restart the system** to see the system displayed as Chinese

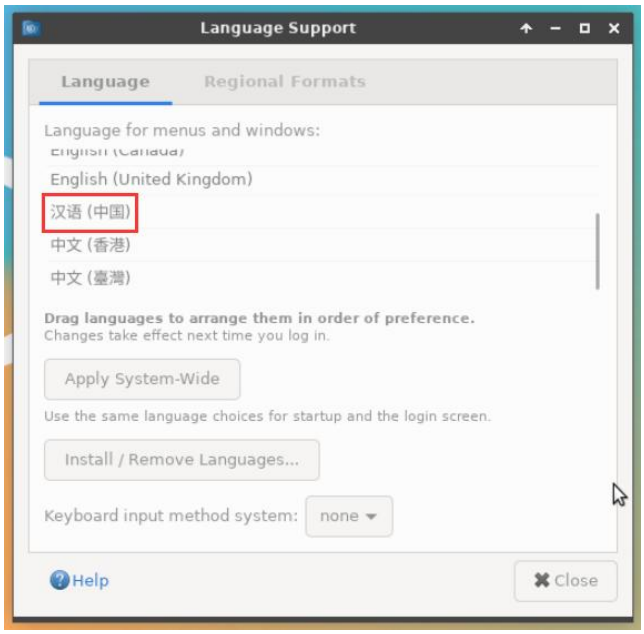


3. 23. 2. The installation method of Ubuntu 20.04 system

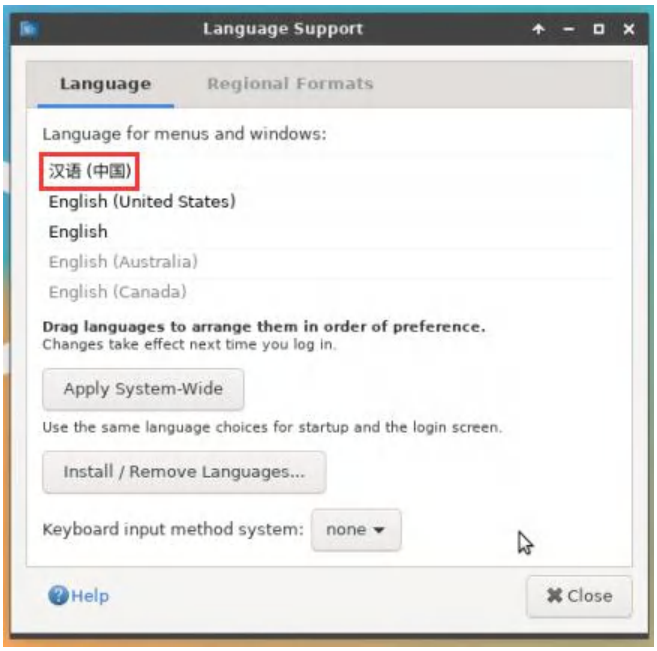
1) First open **Language Support**



2) Then find **Chinese (China)** option

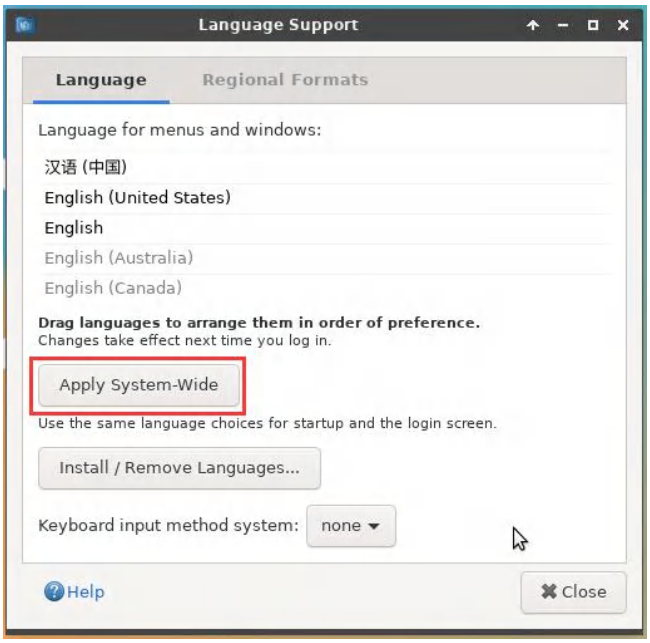


3) Then use the left mouse button to select **Chinese (China)** and hold it down, and then drag it up to the beginning. The display is shown below：



Note that this step is not easy to drag, please try more patiently。

4) Then select the **Apply System-Wide** to apply the Chinese settings to the entire system

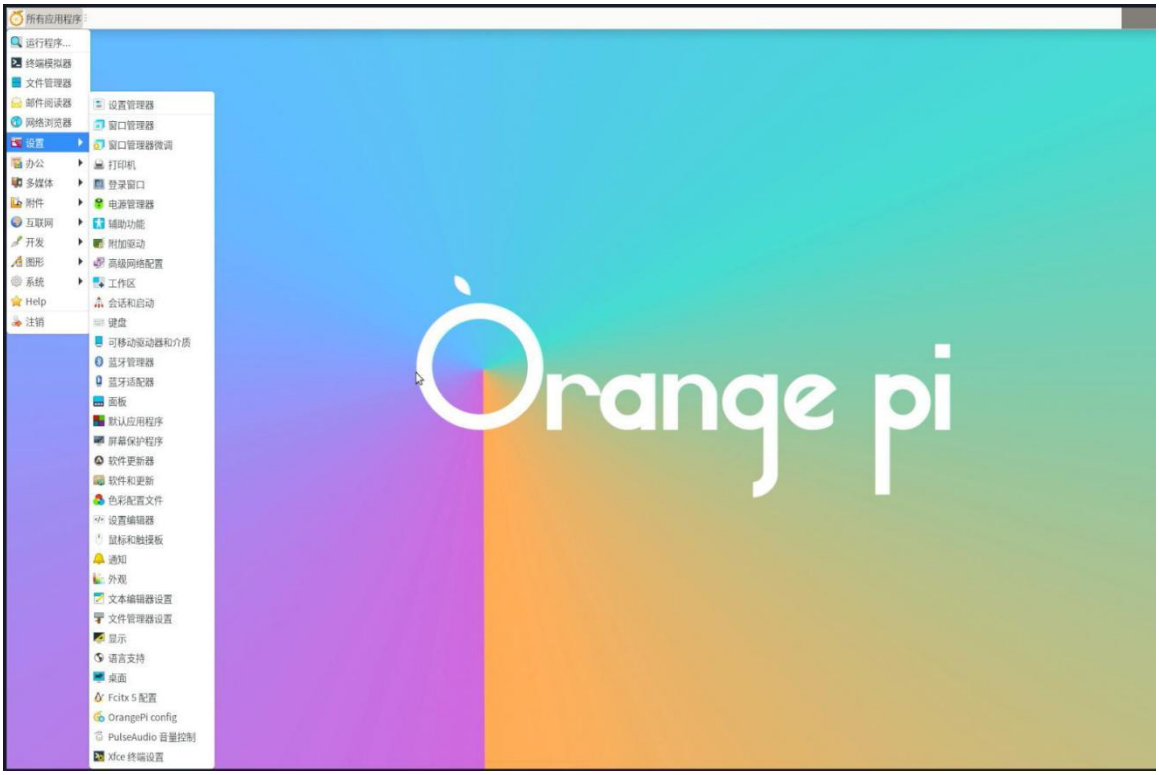


5) Then set the **Keyboard input method system** to **fcitx**

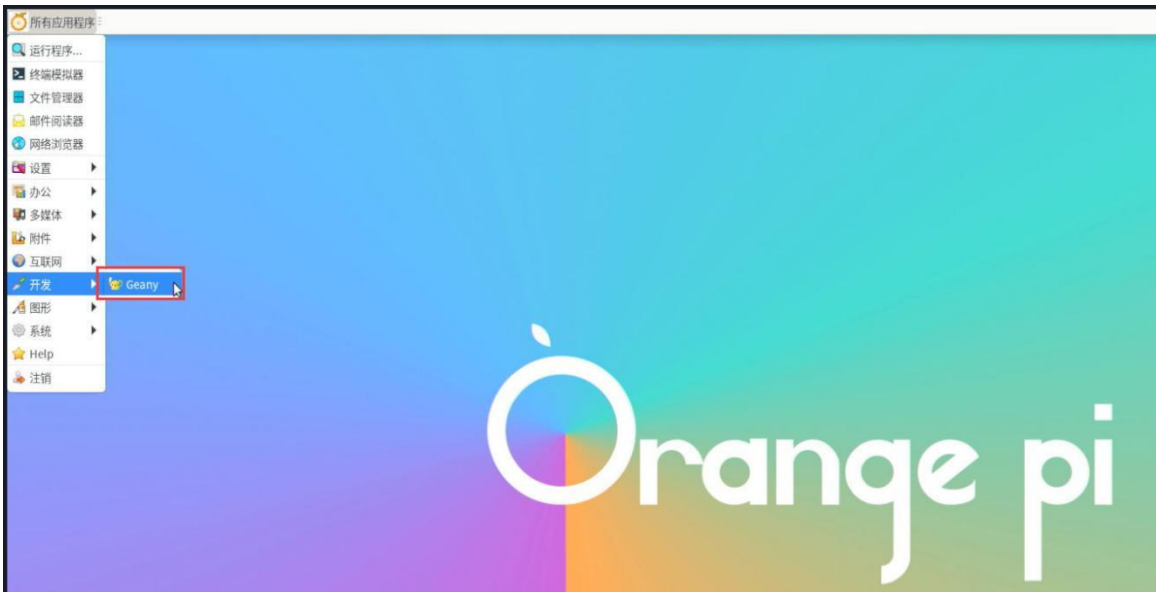
6) **Then restart the Linux system to make the configuration effective**

7) After re-entering the system, please do **not ask me again** at the interface below, and then determine whether the standard folder should be updated as Chinese based on your preference.

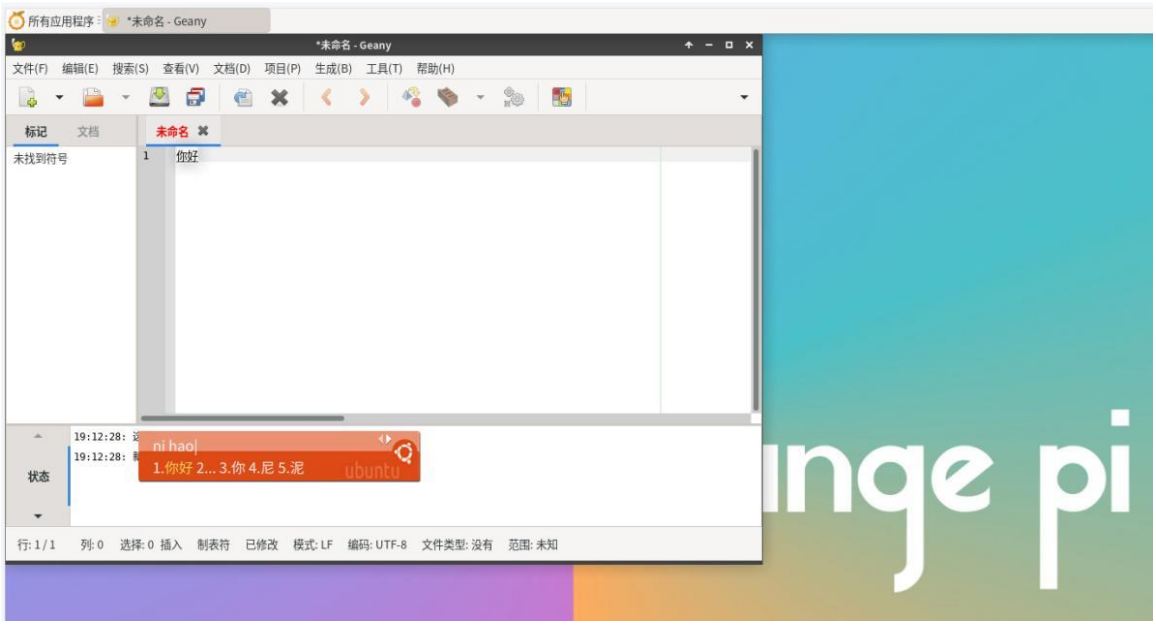
8) Then you can see that the desktop is displayed as Chinese



9) Then we can open the **Geany** to test Chinese input method , and the way to open is shown in the figure below

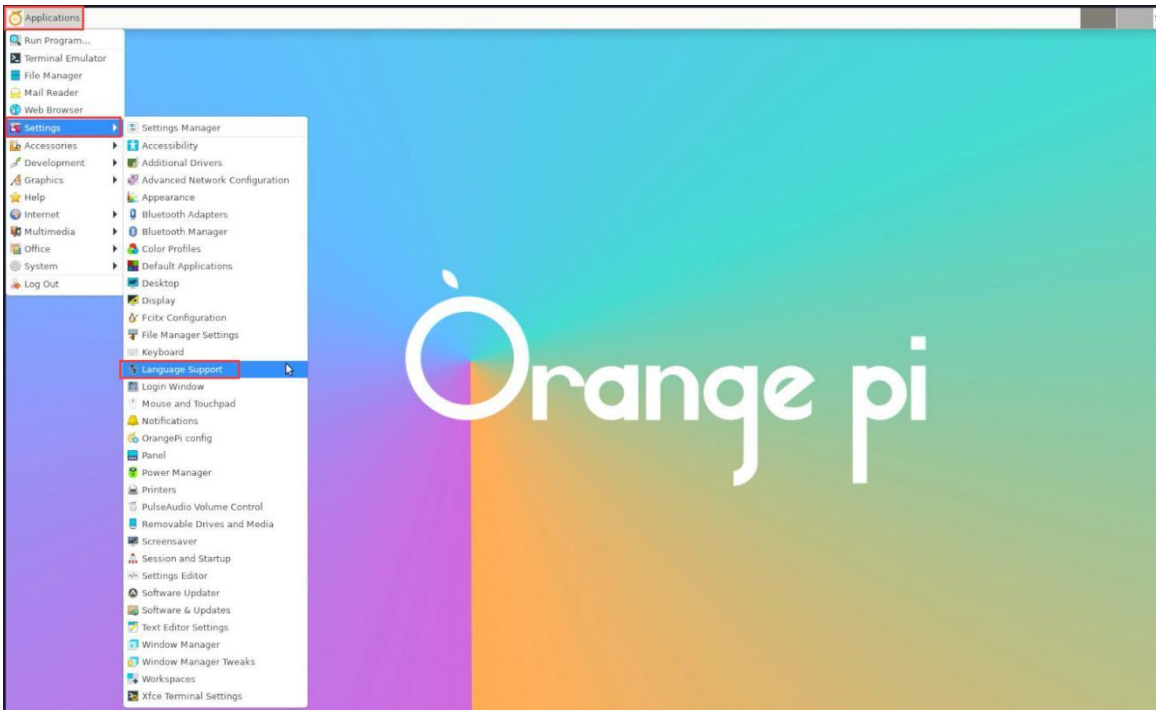


10) After opening **Geany**, the default is an English input method. We can switch into Chinese input method through the **Ctrl+Space** shortcut keys, and then we can enter Chinese

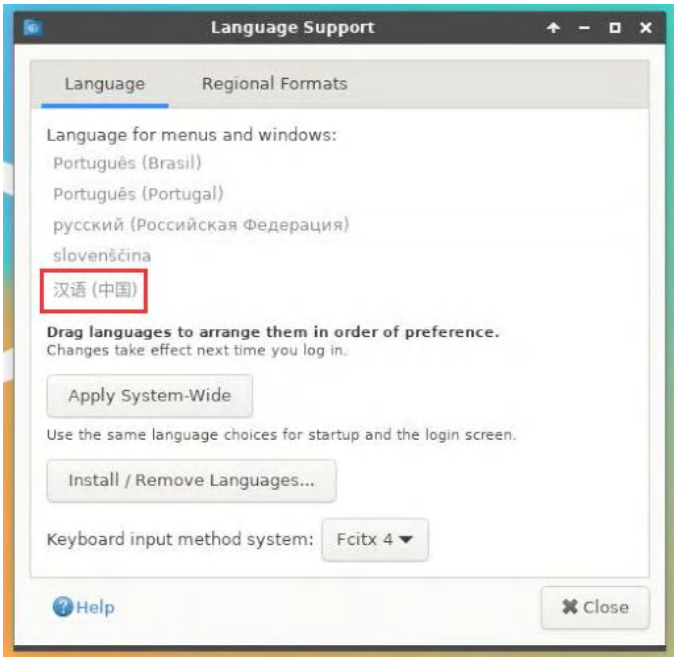


3. 23. 3. The installation method of ubuntu 22.04 system

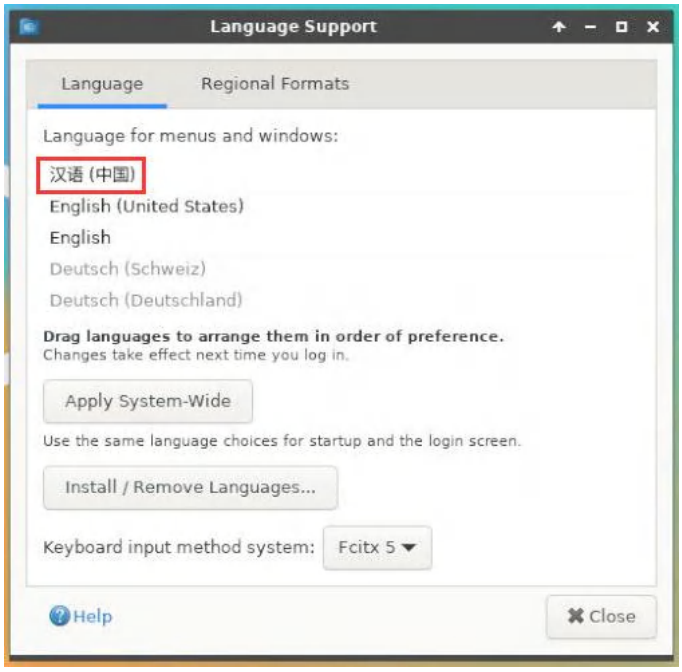
1) First open **Language Support**



2) Then find **Chinese (China)** option

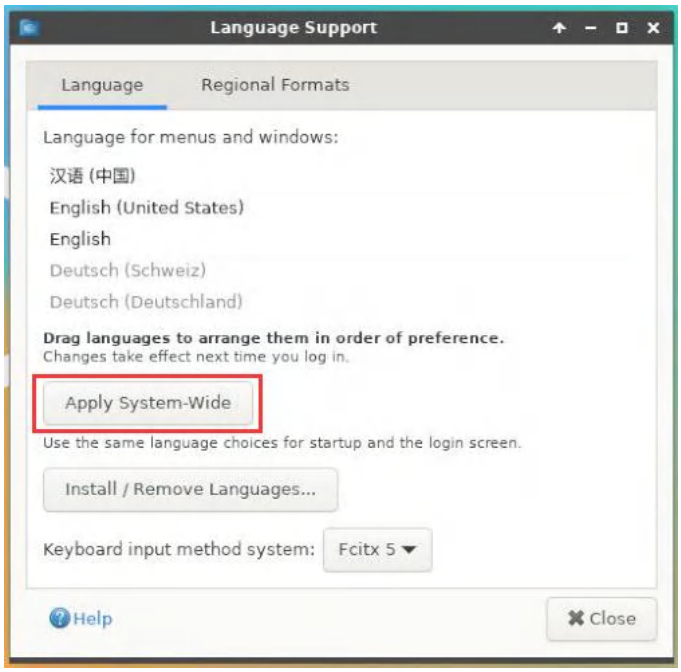


3) Then use the left mouse button to select **Chinese (China)** and hold it down, and then drag it up to the beginning. The display after dragging is shown in the figure below：



Note that this step is not easy to drag, please try more patiently.

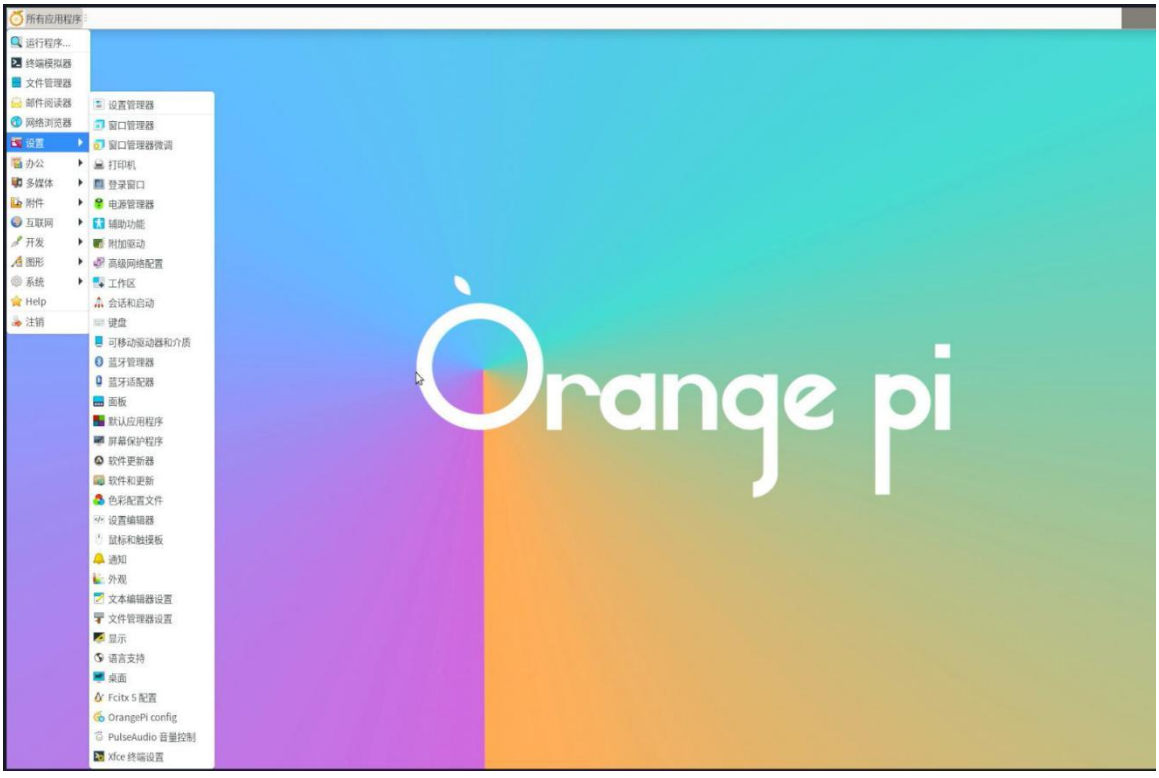
4) Then select the **Apply System-Wide** to apply the Chinese settings to the entire system



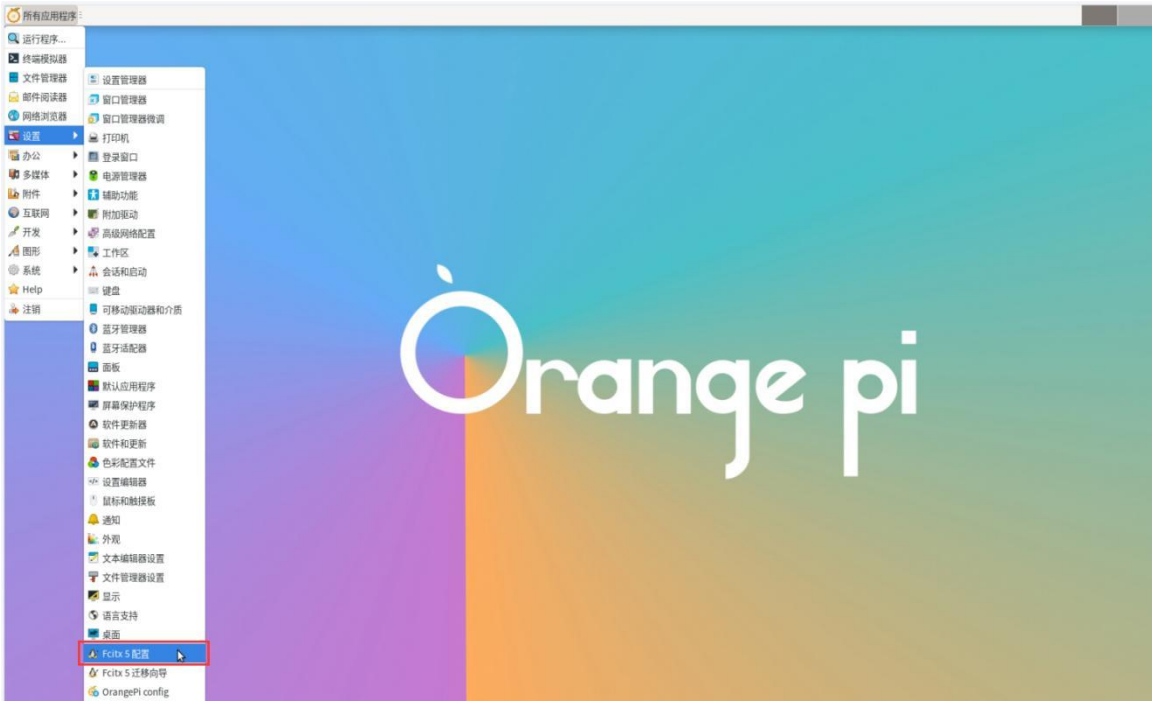
5) **Then restart the Linux system to make the configuration effective**

6) After re-entering the system, please **do not ask me again** at the interface below, and then determine whether the standard folder should be updated as Chinese based on your preference

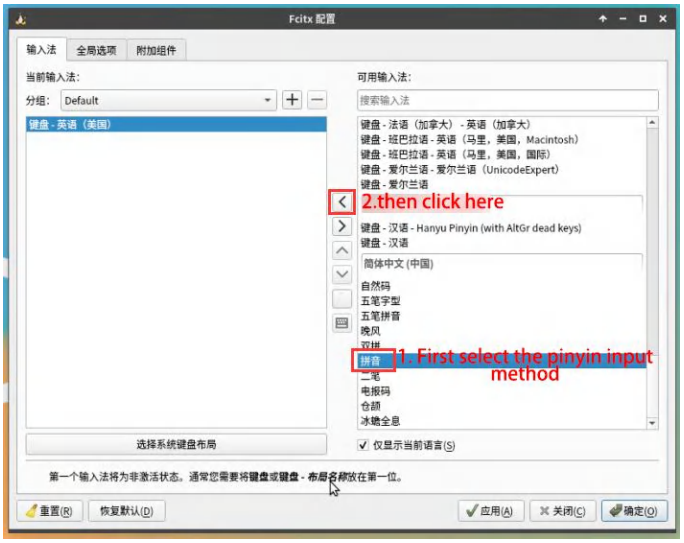
7) Then you can see that the desktop is displayed as Chinese



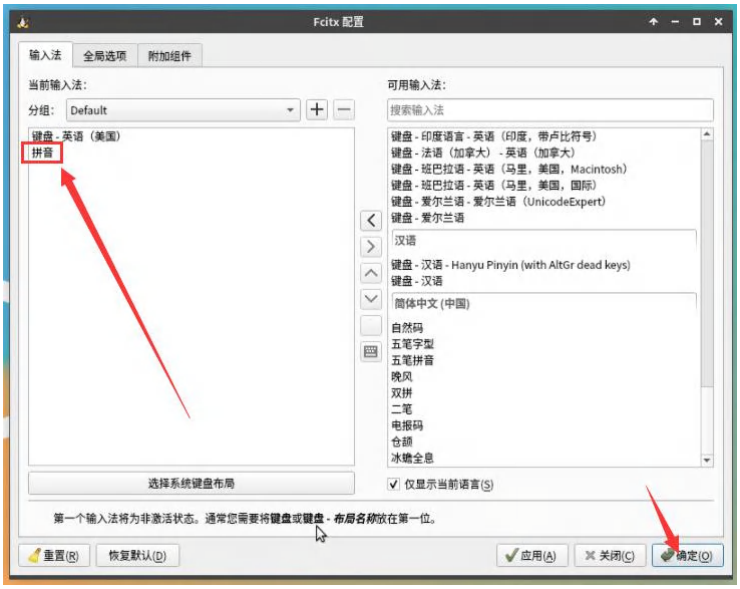
8) Then open the Fcitx5 configuration program



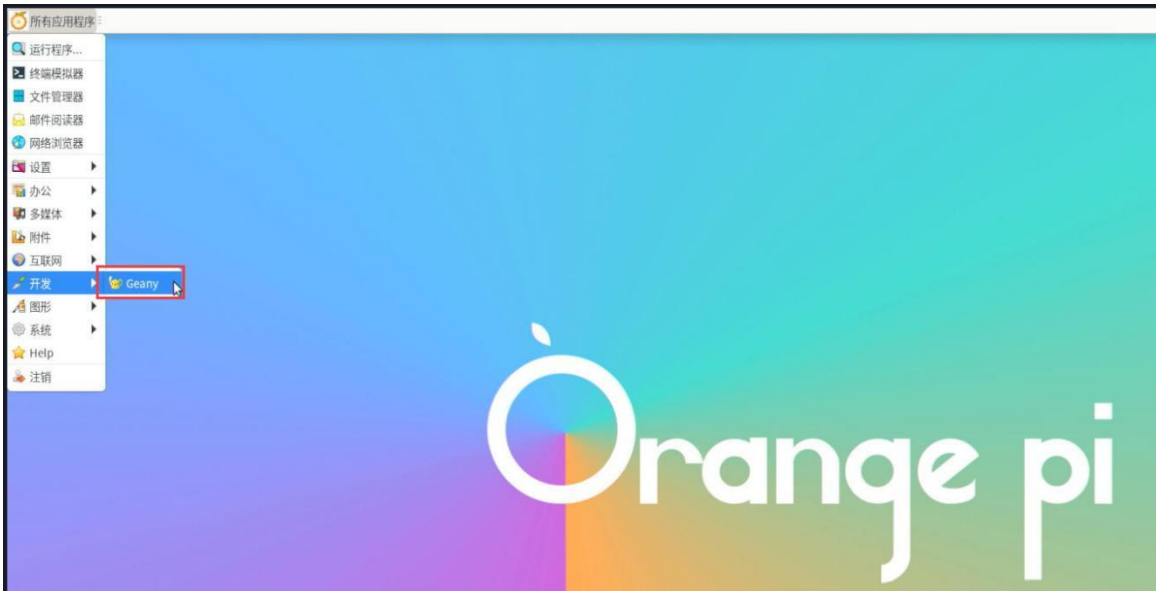
9) Then choose to use Pinyin input method



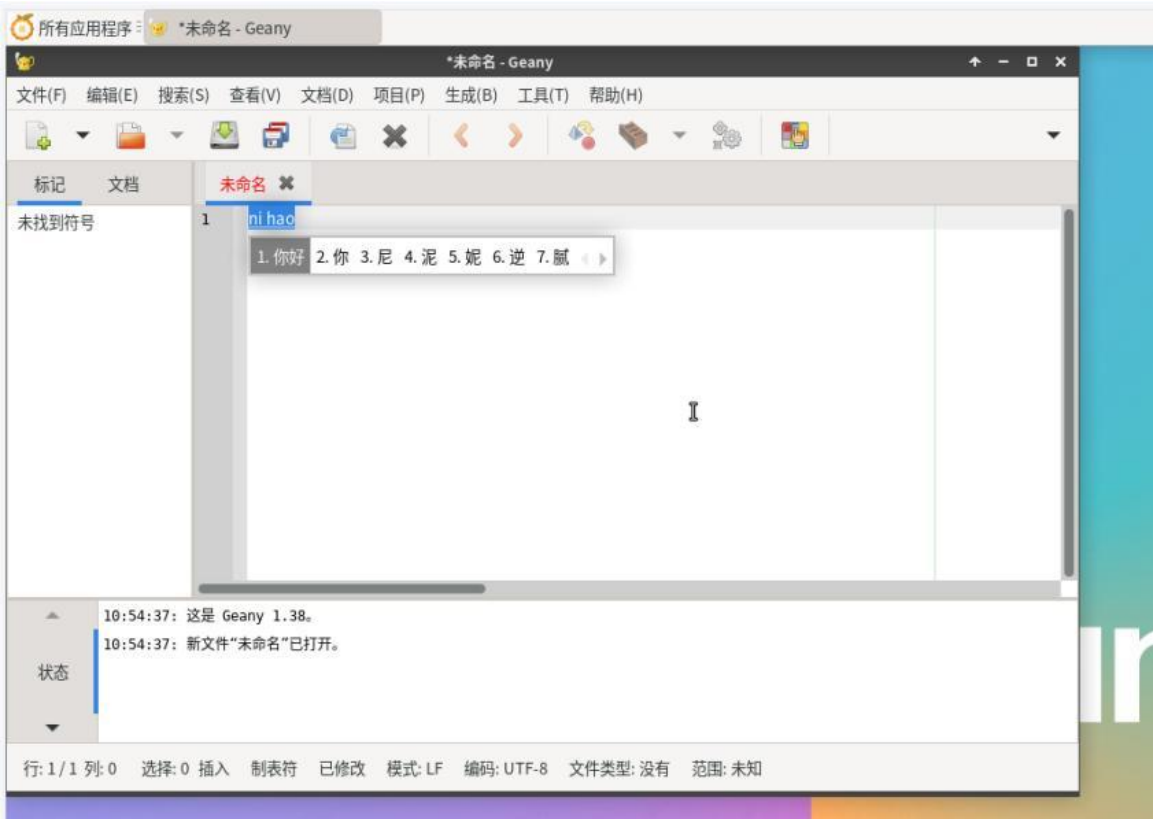
10) The interface after the selection is shown below, then click OK



11) Then we can open the **Geany** to test Chinese input method, and the way to open is shown in the figure below



12) After opening **Geany**, it is still an English input method by default. We can switch into Chinese input methods through the **Ctrl+Space** shortcut keys, and then we can enter Chinese



3.24. How to remotely log in to the Linux system desktop method

3.24.1. Use NoMachine remote login

Please make sure that the Ubuntu or Debian system installed on the development board is a desktop version. In addition, NoMachine also provides detailed usage documents. It is strongly recommended to read this document to familiarize yourself with the use of NoMachine. The document link is as follows:

https://knowledgebase.nomachine.com/DT10R00166

NoMachine supports Windows, Mac, Linux, iOS and Android platforms, so we can remotely log in and control the Orange Pi development board through NoMachine on a variety of devices. The following demonstrates how to remotely log in to the Linux system desktop of the Orange Pi development board through NoMachine in Windows. For installation methods on other platforms, please refer to the official documentation of NoMachine.

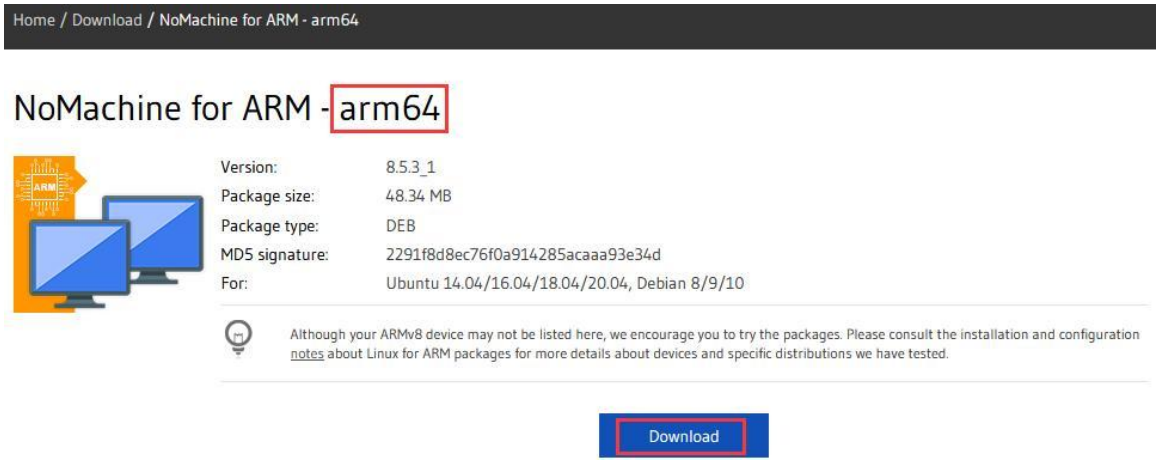
Before operation, please make sure that the Windows computer and the development board are in the same LAN, and you can log in to the Ubuntu or Debian system of the development board through ssh.

1) First download the installation package of the NoMachine software Linux **arm64** deb version, and then install it in the Linux system of the development board

a. Since RK3566 is an SOC with ARMv8 architecture, the system we use is Ubuntu or Debian, so here we need to download the **NoMachine for ARM ARMv8 DEB** installation package. The download link is as follows:

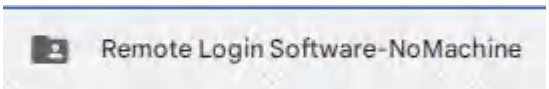
Note that this download link may change, please look for the deb package of the Armv8/Arm64 version.

https://downloads.nomachine.com/download/?id=116&distro=ARM

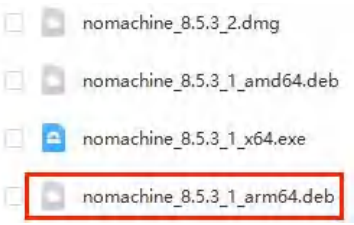


b. In addition, you can also download the installation package to **NoMachine** in the **official tools**.

First enter the **remote login software-Nomachine** folder



Then download the ARM64 version of the DEB installation package



c. Then upload the downloaded **nomachine_x.x.x_x_arm64.deb** to the Linux system of the development board.

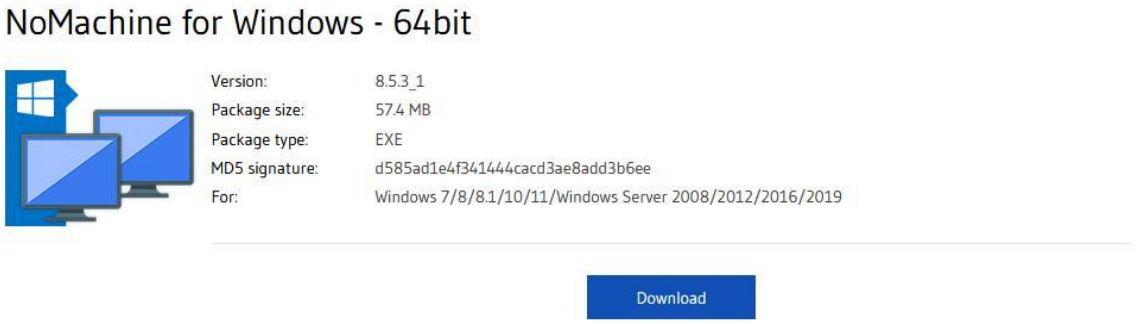
d. Then use the following command to install **NoMachine** in the Linux system of the development board

```
orangepi@orangepi:~$ sudo dpkg -i nomachine_x.x.x_x_arm64_arm64.deb
```

2) Then download the installation package of the Windows version of the NoMachine software, the download address is as follows

Note that this download link may change.

https://downloads.nomachine.com/download/?id=8

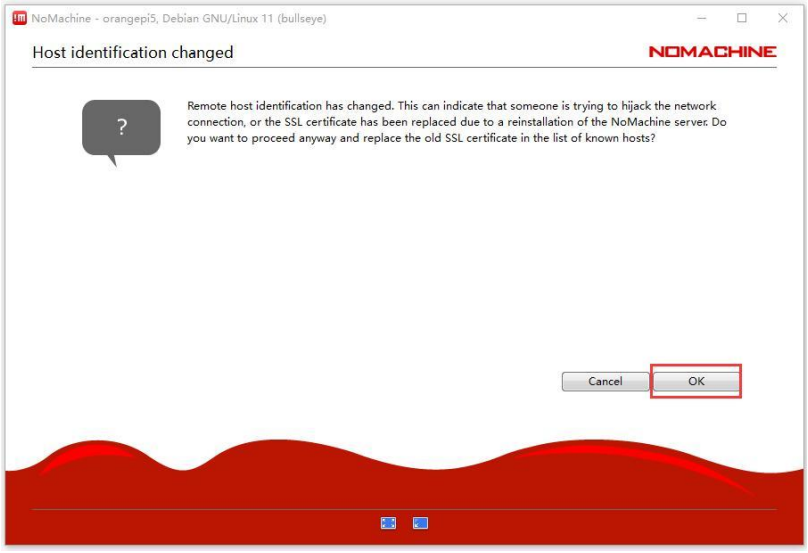


3) Then install nomachine in Windows. **Please restart the computer after installation**

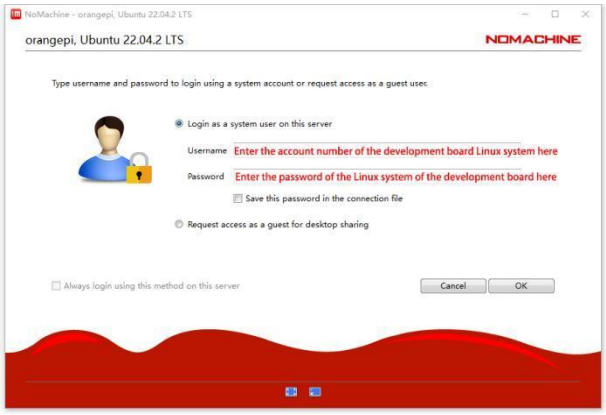
4) Then open **NoMachine** in Window

5) After NoMachine starts, it will automatically scan other devices installed with NoMachine in the LAN. After entering the main interface of NoMachine, you can see that the development board is already in the list of connectable devices, and then click the position shown in the red box in the figure below You can start to log in to the Linux system desktop of the development board

6) Then click **OK**

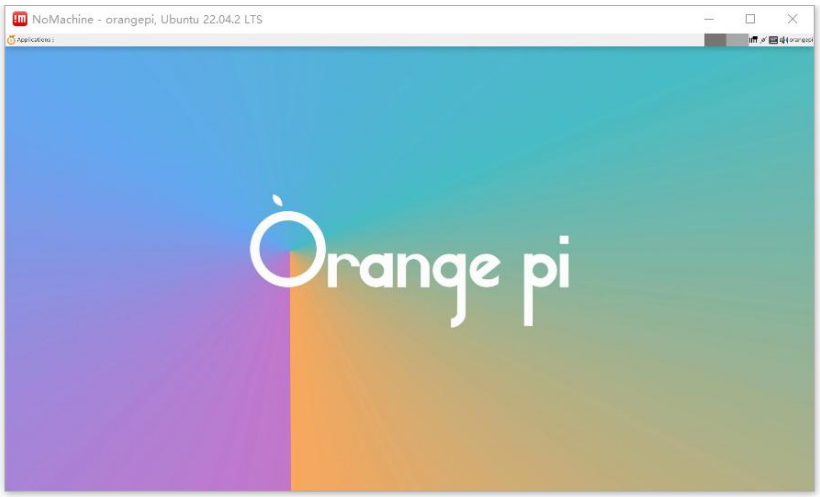


7) Then enter the username and password of the Linux system in the corresponding position in the figure below, and then click **OK** to start logging in



8) Then click OK in the next interface

9) Finally, you can see the desktop of the development board Linux system



3. 24. 2. Use VNC remote login

Before operation, please make sure that the Windows computer and the development board are in the same LAN, and you can log in to the Ubuntu or Debian system of the development board through ssh.

Ubuntu 20.04 tests many problems with VNC, please do not use this method。

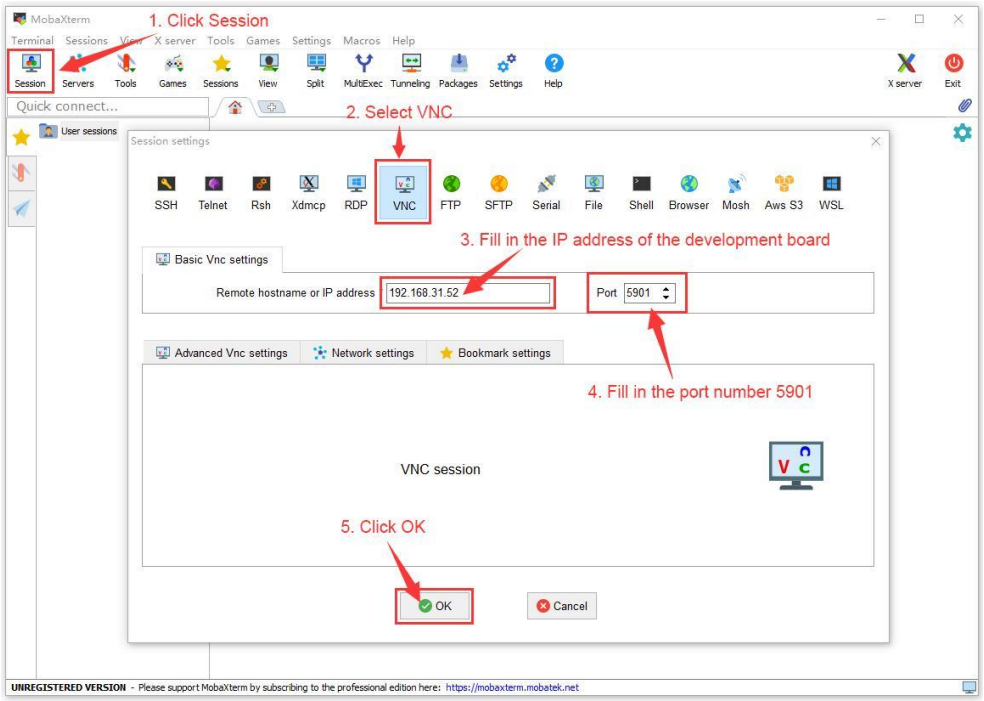
1) First run the **set_vnc.sh** script settings, and **remember to add Sudo permissions**

```
orangepi@orangepi:~$ sudo set_vnc.sh
You will require a password to access your desktops.

Password:          #Set the VNC password here, 8 -bit characters
Verify:            #Set the VNC password here, 8 -bit characters
Would you like to enter a view-only password (y/n)? n
xauth:    file /root/.Xauthority does not exist

New 'X' desktop is orangepi3b:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/orangepi3b:1.log

Killing Xtightvnc process ID 3047

New 'X' desktop is orangepi3b:1
```

```
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/orangepi3b:1.log
```

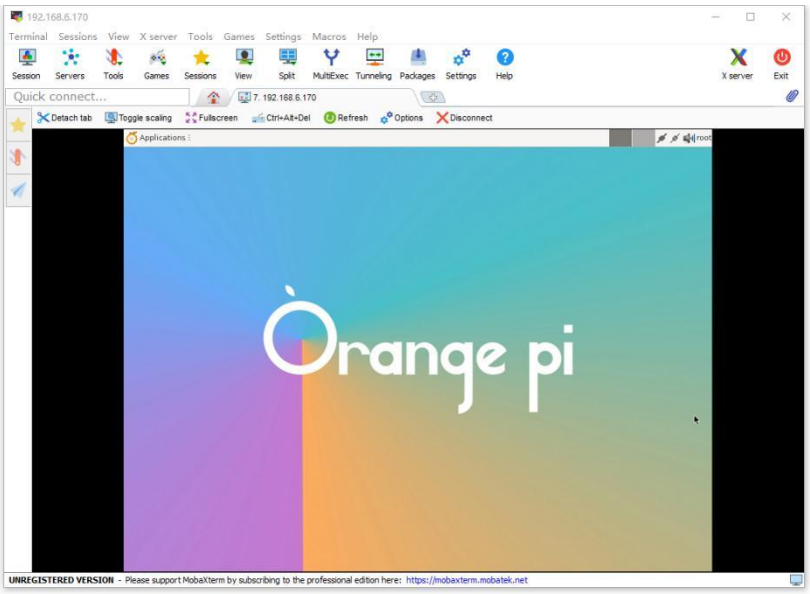
2) The steps to use the MobaXterm software to connect to the Linux system desktop of the development board are as follows:

a. First click on Session, then select VNC, then fill in the IP address and port of the development board, and finally click OK to confirm



b. Then enter the password of the previously set VNC

c. The interface after the login is shown as shown in the figure below, and then the desktop of the Linux system can be remotely operated



3.25. Some programming language tests supported by Linux system

3.25.1. Debian Bullseye system

1) Debian Bullseye has a gcc compilation tool chain installed by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 10.2.1-6) 10.2.1 20210110
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write C language of **Hello_world.c** program

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");
```

```
        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Debian Bullseye has Python3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. **hello_world.py** program in Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Bullseye does not install Java compilation tools and operating environment by default

a. You can use the following command to install openjdk, the latest version in Debian Bullseye is openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
```

c. Edit the **hello_world.java** of the Jave version

```
orangepi@orangepi:~$ vim hello_world.java
public class hello_world
{
        public static void main(String[] args)
        {
                System.out.println("Hello World!");
```

```
        }
}
```

d. Then compile and run **hello_world.java**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3.25.2. Debian Bookworm system

1) Debian Bookworm is installed with a gcc compilation tool chain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 12.2.0-14) 12.2.0
Copyright (C) 2022 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Debian Bookworm has Python3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.11.2 (main, Mar 13 2023, 12:18:29) [GCC 12.2.0] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Use the Ctrl+D shortcut to exit python's interactive mode.

b. Write the **hello_world.py** program in Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Bookworm does not install Java compilation tools and operating environment by default

a. You can use the following command to install openjdk, the latest version in Debian Bookworm is openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
```

c. Write the Java version of **hello_world.java**

```
orangepi@orangepi:~$ vim hello_world.java
public class hello_world
{
        public static void main(String[] args)
        {
                System.out.println("Hello World!");
        }
}
```

d. Then compile and run **hello_world.java**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3. 25. 3. Ubuntu Focal system

1) Ubuntu Focal has a gcc compilation tool chain installed by default, which can directly compile C language programs in the Linux system of the development board

a. The version of a.gcc is as follows

```
orangepi@orangepi:~$ gcc --version
```

```
gcc (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. **hello_world.c** program to write C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Ubuntu Focal defaults to install Python3

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. **hello_world.py** program in Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Ubuntu Focal's compilation tool and operating environment without the installation of

Java default

a. You can use the following command to install **openjdk-17**

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 17.0.2 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-120.04, mixed mode, sharing)
```

c. Edit the **hello_world.java** of Jave version

```
orangepi@orangepi:~$ vim hello_world.java
public class hello_world
{
        public static void main(String[] args)
        {
                System.out.println("Hello World!");
        }
}
```

d. Then compile and run **hello_world.java**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3.25.4. Ubuntu Jammy system

1) Ubuntu Jammy is installed with a gcc compilation tool chain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu 11.2.0-19ubuntu1) 11.2.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program of c language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>
```

```
int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Ubuntu jammy is installed with Python3 by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.10.4 (main, Apr  2 2022, 09:04:19) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Edit **hello_world.py** program in Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Ubuntu jammy defaults to compile tools and operating environments that are not installed in Java

a. You can use the following command to install openjdk-18

```
orangepi@orangepi:~$ sudo apt install -y openjdk-18-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 18-ea 2022-03-22
OpenJDK Runtime Environment (build 18-ea+36-Ubuntu-1)
OpenJDK 64-Bit Server VM (build 18-ea+36-Ubuntu-1, mixed mode, sharing)
```

c. Edit the **hello_world.java** of the Jave version

```
orangepi@orangepi:~$ vim hello_world.java
```

```
public class hello_world
{
        public static void main(String[] args)
        {
                System.out.println("Hello World!");
        }
}
```

d. Then compile and run **hello_world.java**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3. 26. QT installation method

1) Use the following script to install QT5 and QT Creator

```
orangepi@orangepi:~$ install_qt.sh
```

2) The QT version number will be automatically printed after installation

a. Ubuntu20.04 comes with QT version **5.12.8**

```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.12.8 in /usr/lib/aarch64-linux-gnu
```

b. Ubuntu22.04 comes with QT version **5.15.3**

```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.15.3 in /usr/lib/aarch64-linux-gnu
```

c. Debian11 comes with QT version **5.15.2**

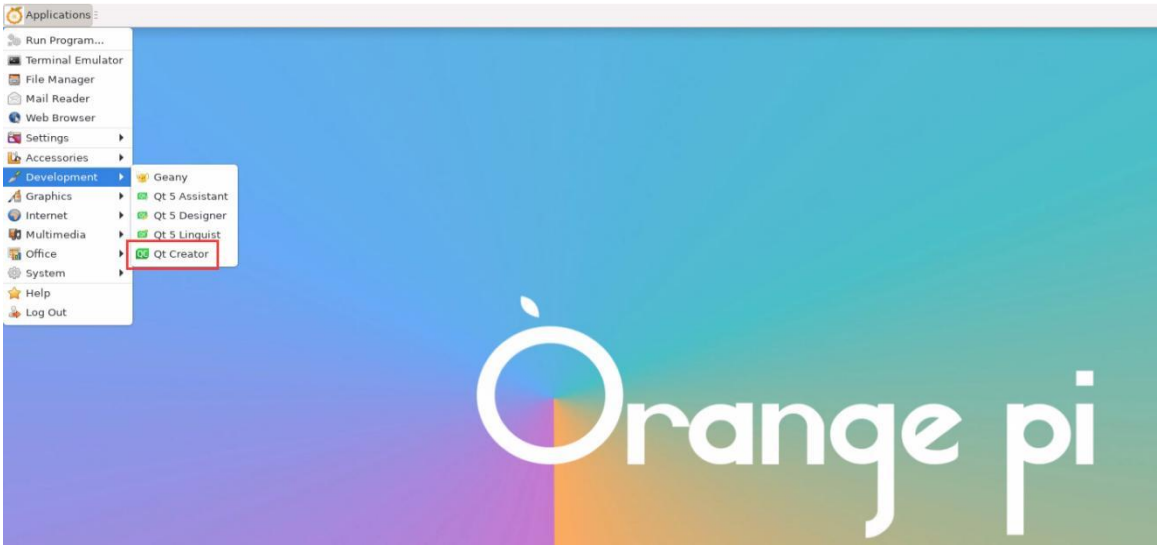
```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.15.2 in /usr/lib/aarch64-linux-gnu
```

d. Debian12 comes with QT version **5.15.8**

```
orangepi@orangepi:~$ install_qt.sh
```

```
.....
QMake version 3.1
Using Qt version 5.15.8 in /usr/lib/aarch64-linux-gnu
```

3) Then you can see the QT Creator launch icon in **Applications**



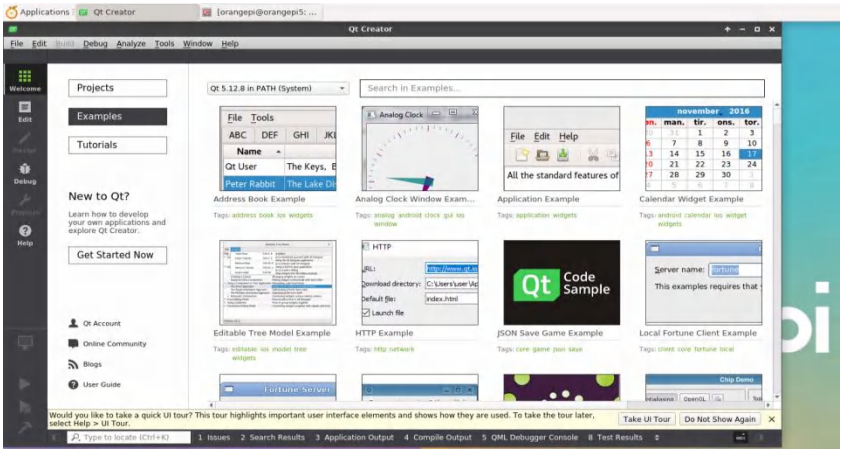
QT Creator can also be opened using the following command

```
orangepi@orangepi:~$ qtcreator
```

During the startup of QT and QT applications, if the following error is displayed, ignore it. This error has no impact on application running.

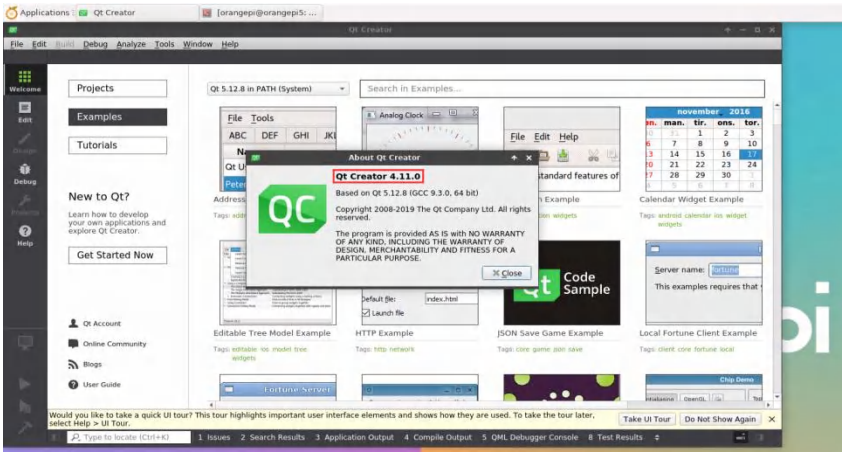
libGL error: failed to create dri screen
libGL error: failed to load driver: rockchip
libGL error: failed to create dri screen
libGL error: failed to load driver: rockchip

4) The interface after QT Creator is opened is as follows

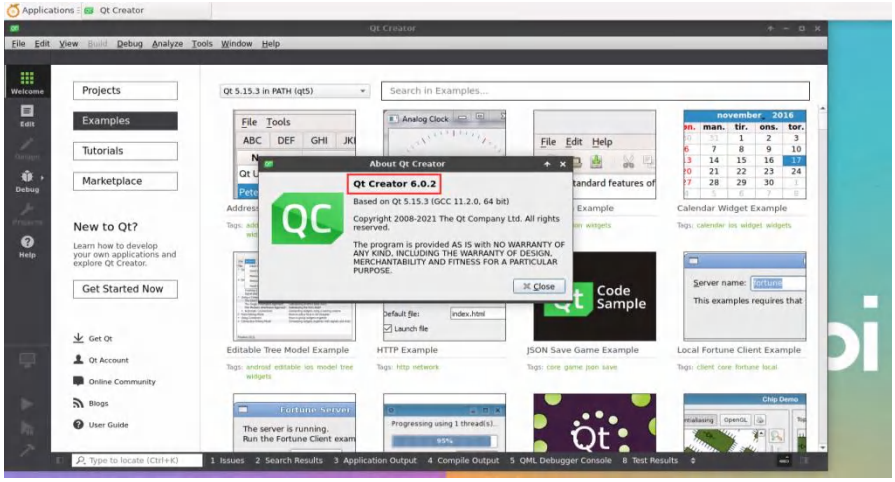


5) The QT Creator version is shown below

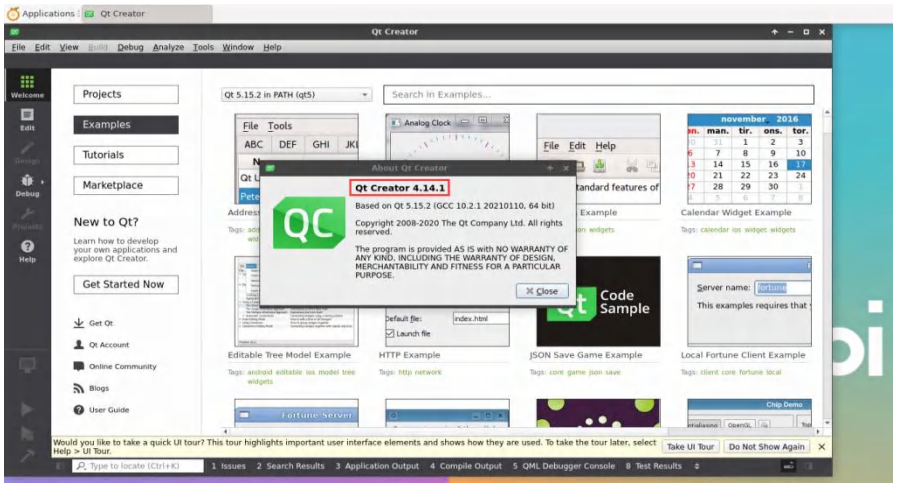
a. The default version of QT Creator in **Ubuntu20.04** is as follows



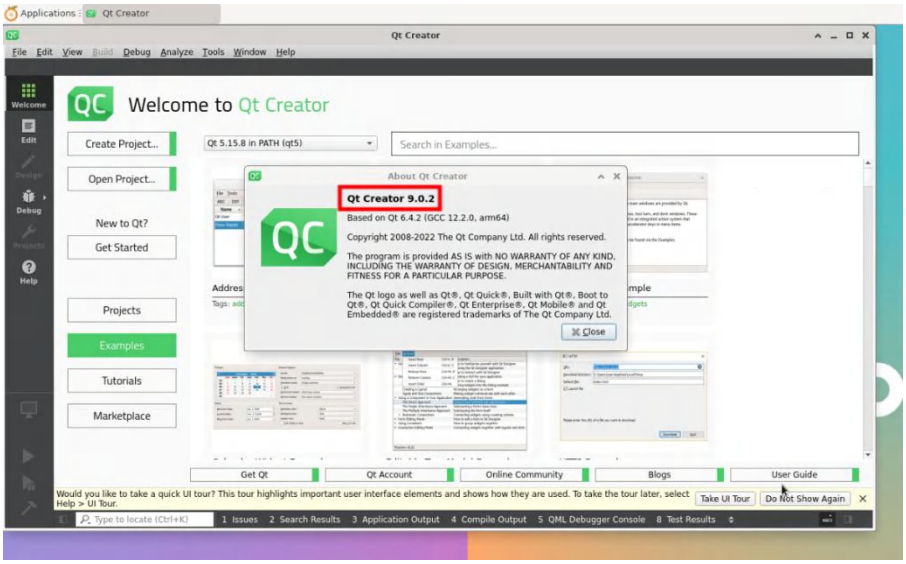
b. The default version of QT Creator in **Ubuntu22.04** is as follows



c. The default version of QT Creator in **Debian11** is as follows

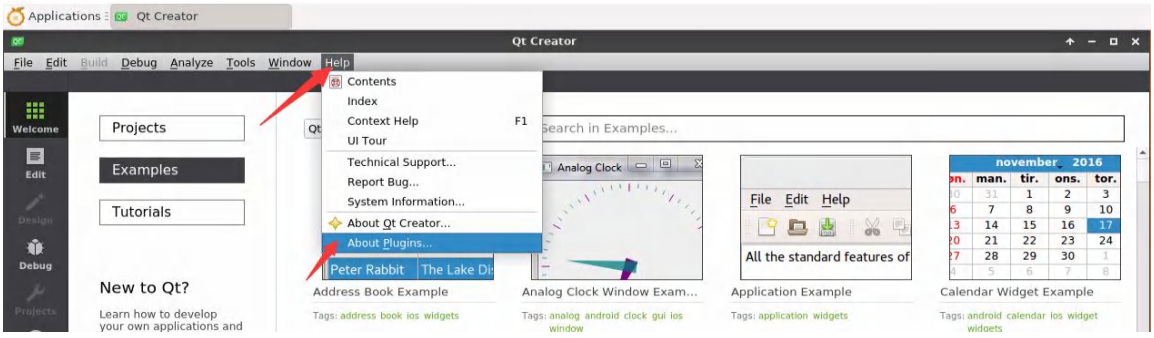


d. The default version of QT Creator in **Debian12** is as follows

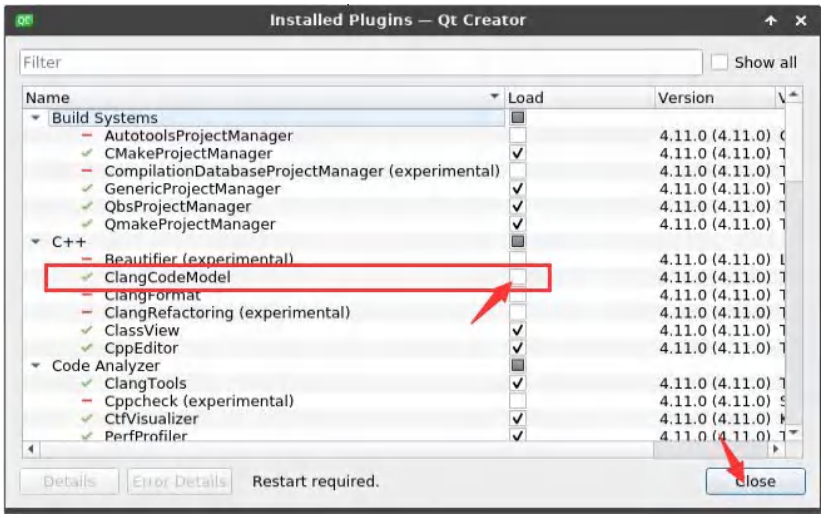


6) Then set QT

a. First open **Help->About Plugins...**



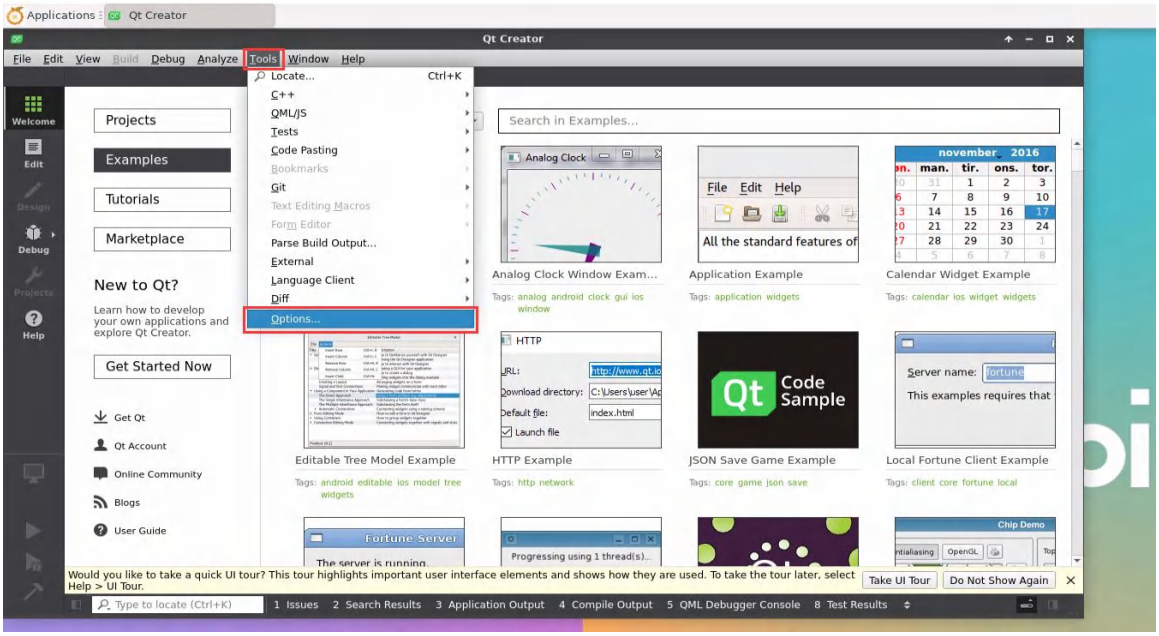
b. Then remove the check box for **ClangCodeModel**

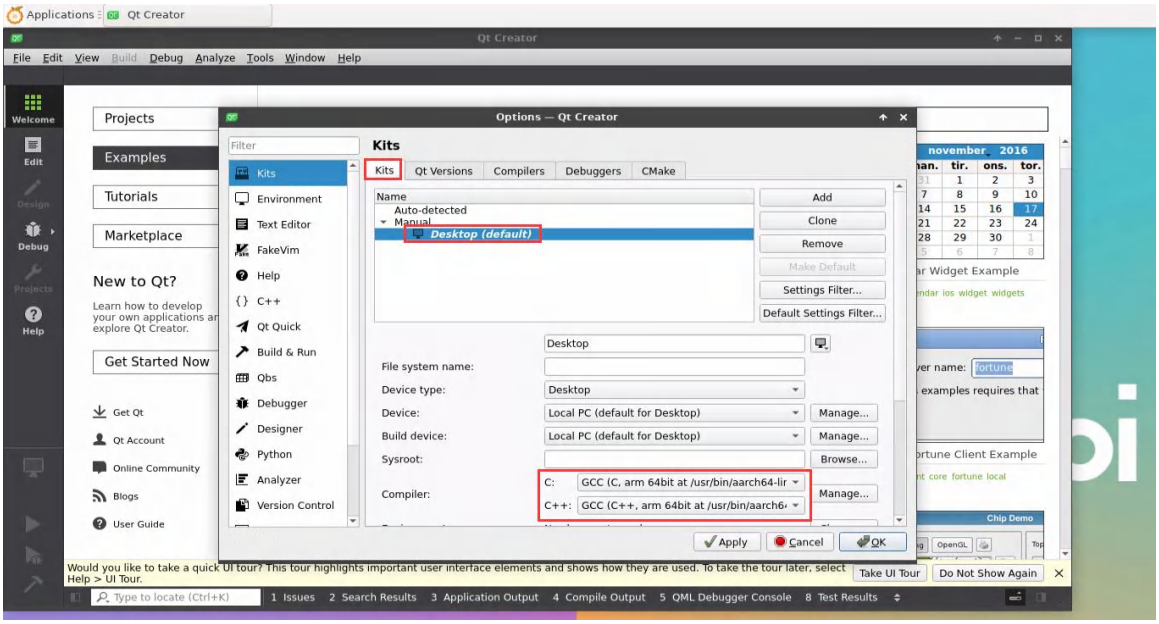


c. **Restart QT Creator after the Settings are complete**

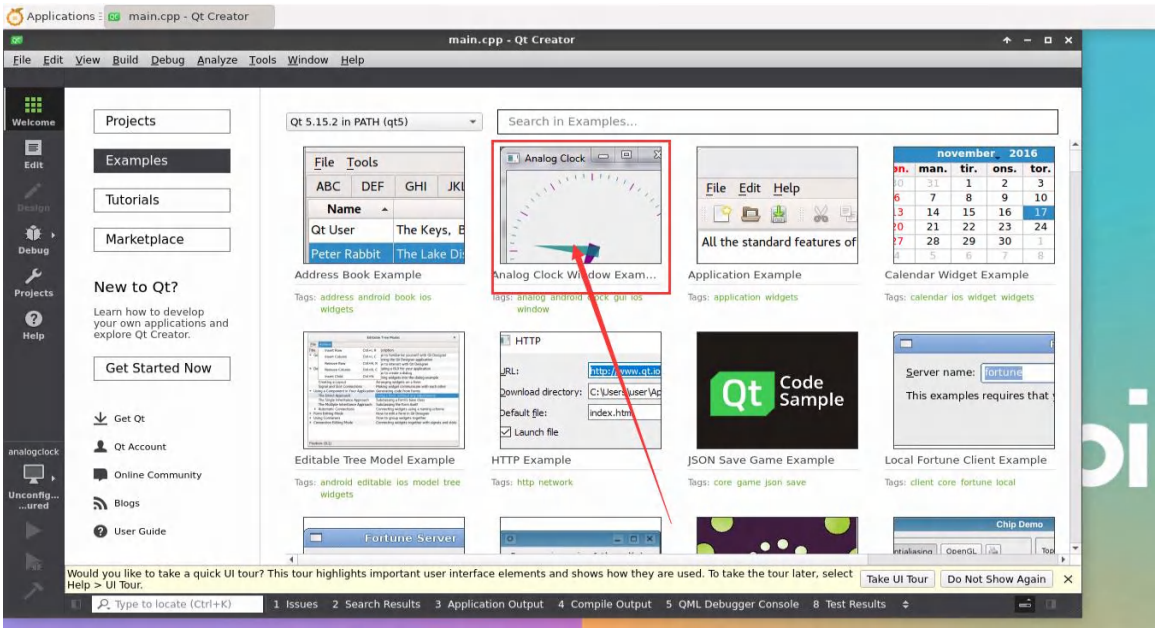
d. Then make sure that QT Creator uses the GCC compiler, if the default is Clang, change it to GCC

Debian12 Please skip this step.

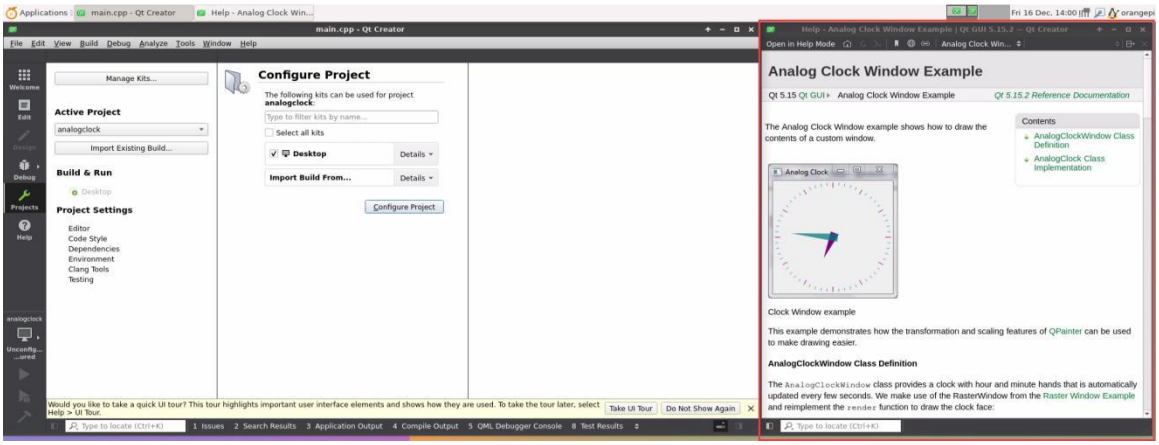




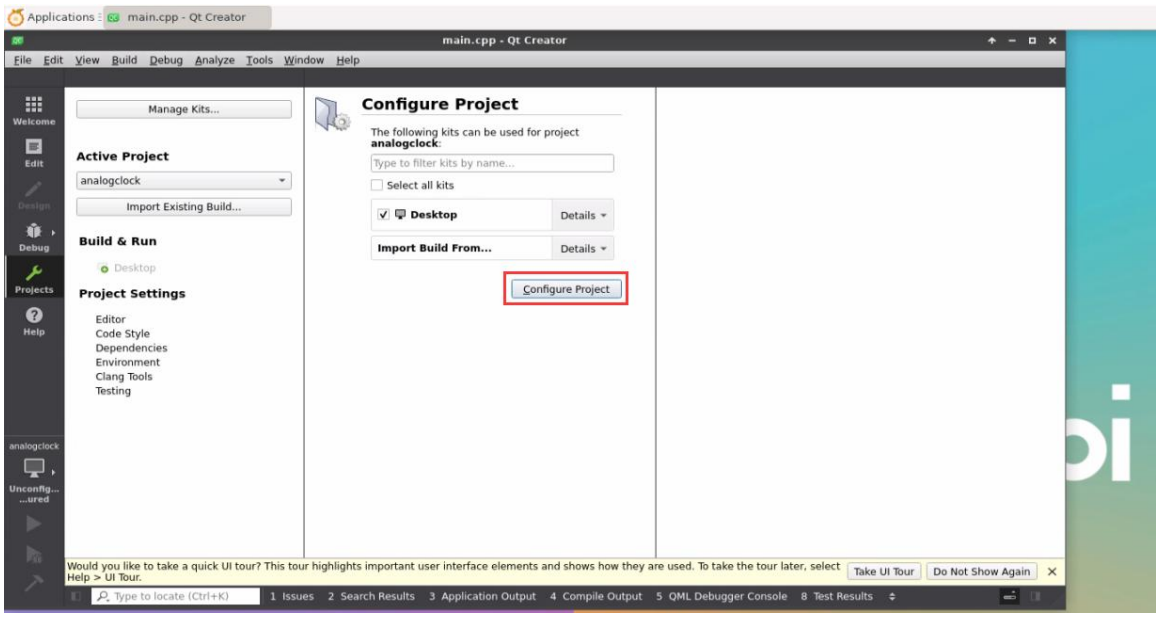
7) You can then open a sample code



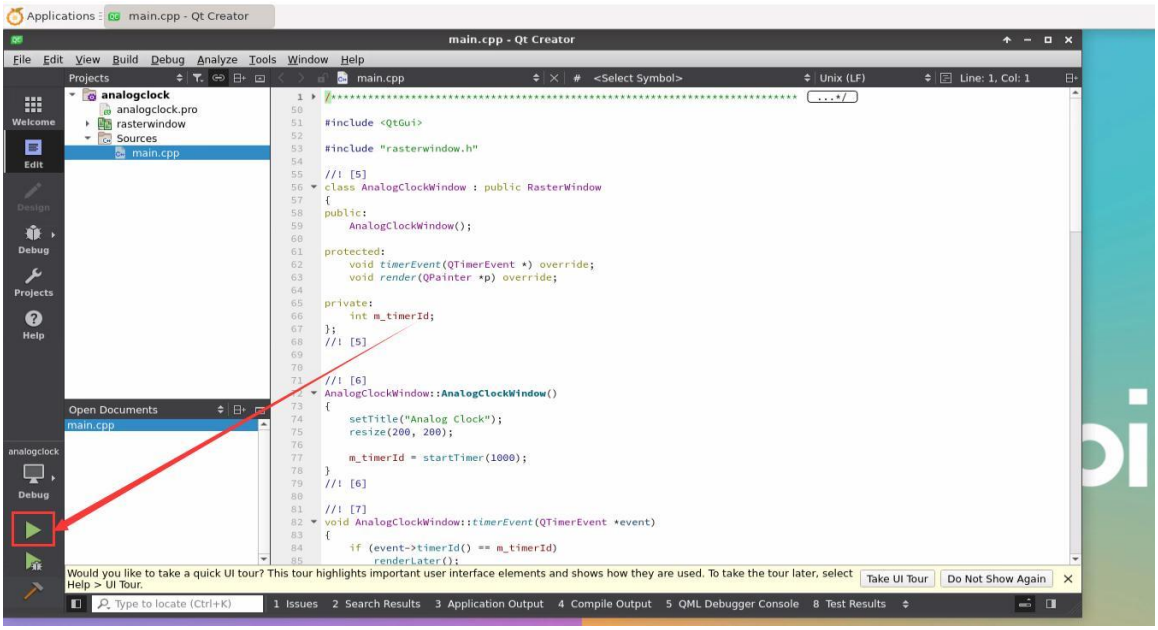
8) Clicking on the example code will automatically open the corresponding instruction document, you can carefully read the instructions



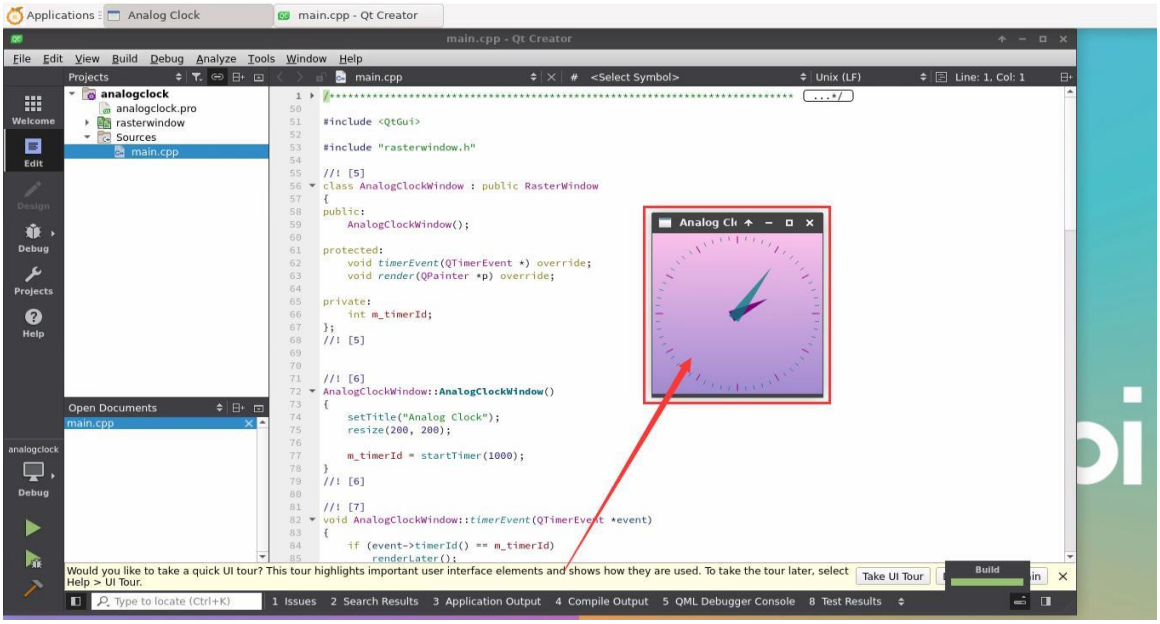
9) Then click **Configure Project**



10) Then click the green triangle in the lower left corner to compile and run the sample code



11) After waiting for a period of time, the interface shown in the following figure will pop up, which indicates that QT can compile and run normally



12) Reference documents

https://wiki.qt.io/Install_Qt_5_on_Ubuntu
https://download.qt.io/archive/qtcreator
https://download.qt.io/archive/qt

3. 27. ROS Installation Method

3. 27. 1. How to install ROS 1 Noetic on Ubuntu 20.04

1) The current active version of ROS 1 is as follows, the recommended version is **Noetic Ninjemys**

Distro	Release date	Poster	*Tuturtle*, turtle in tutorial	EOL date
ROS Noetic Ninjemys (**Recommended**)	May 23rd, 2020			May, 2025 (Focal EOL)
ROS Melodic Morenia	May 23rd, 2018			May, 2023 (Bionic EOL)

http://docs.ros.org
https://wiki.ros.org/Distributions

2) The official installation document link of ROS 1 **Noetic Ninjemys** is as follows:

http://wiki.ros.org/noetic/Installation/Ubuntu

3) In the official installation document of ROS **Noetic Ninjemys**, Ubuntu recommends using Ubuntu20.04, so please make sure that the system used by the development board is **Ubuntu20.04 desktop system**

http://wiki.ros.org/noetic/Installation

4) Then use the script below to install ros1

```
orangepi@orangepi3b:~$ install_ros.sh ros1
```

5) Before using the ROS tool, you first need to initialize rosdep, and then you can quickly install some system dependencies and some core components in ROS when compiling the source code

Note that running the following command needs to ensure that the development board can access github normally, otherwise an error will be reported due to network problems.

The install_ros.sh script will try to modify /etc/hosts and automatically run the following commands. However, this method cannot guarantee normal access to github every time. If the following error is displayed after installing ros1 in install_ros.sh, please find other ways to allow the Linux system of the development board to access github normally, and then manually run the following Order.

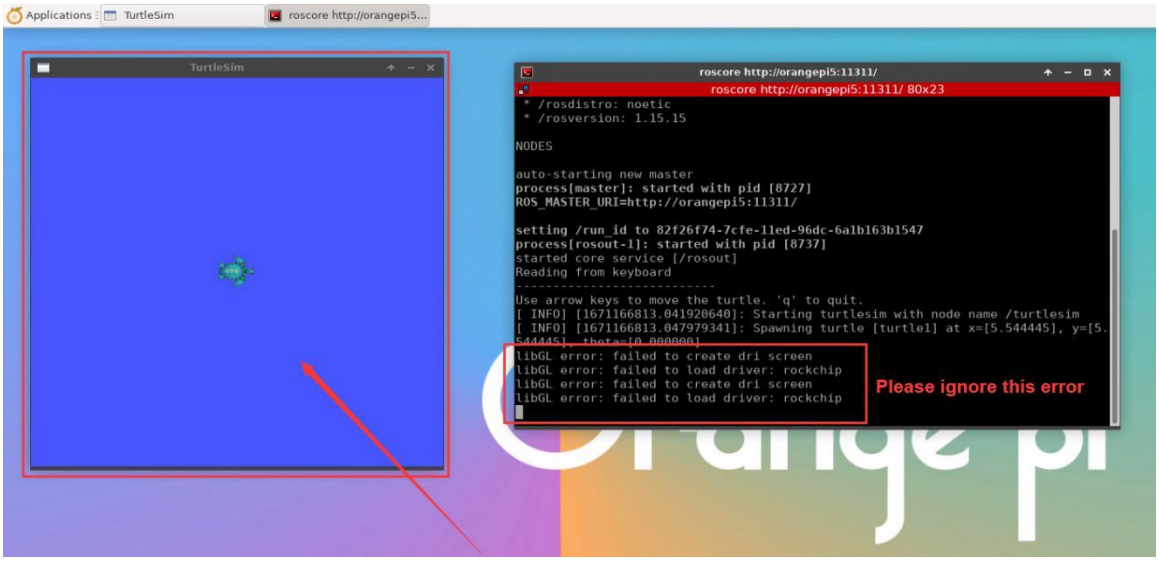
https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
ERROR: error loading sources list:
The read operation timed out

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ sudo rosdep init
Wrote /etc/ros/rosdep/sources.list.d/20-default.list
Recommended: please run
```

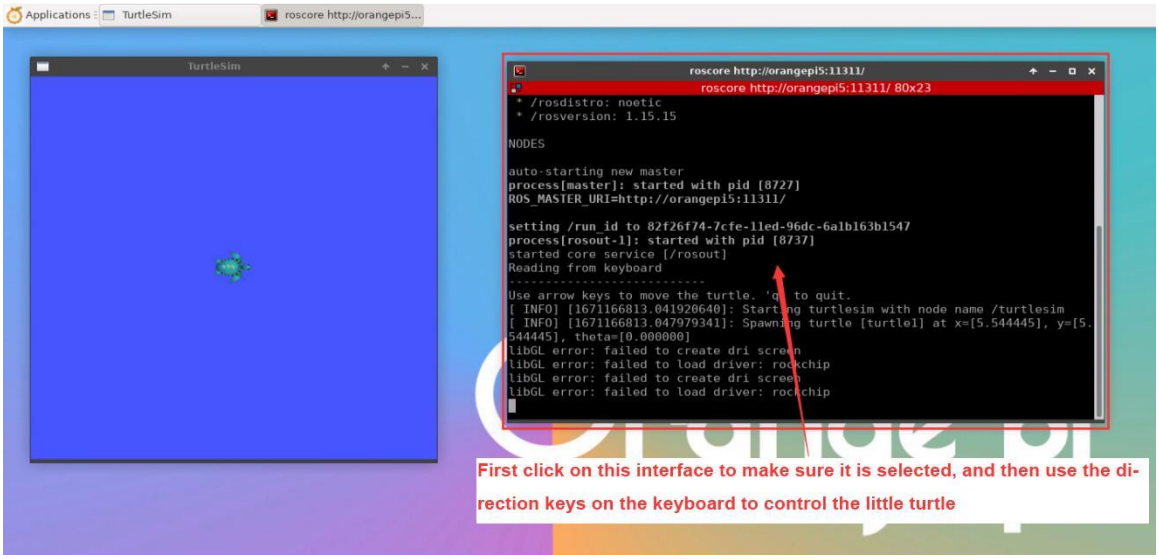
```
            rosdep update
orangepi@orangepi:~$ rosdep update
reading in sources list data from /etc/ros/rosdep/sources.list.d
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml
Query rosdistro index
https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml
Skip end-of-life distro "ardent"
Skip end-of-life distro "bouncy"
Skip end-of-life distro "crystal"
Skip end-of-life distro "dashing"
Skip end-of-life distro "eloquent"
Add distro "foxy"
Add distro "galactic"
Skip end-of-life distro "groovy"
Add distro "humble"
Skip end-of-life distro "hydro"
Skip end-of-life distro "indigo"
Skip end-of-life distro "jade"
Skip end-of-life distro "kinetic"
Skip end-of-life distro "lunar"
Add distro "melodic"
Add distro "noetic"
Add distro "rolling"
updated cache in /home/orangepi/.ros/rosdep/sources.cache
```

6) Then open a command line terminal window on the desktop, and then use the test_ros.sh script to start a small turtle routine to test whether ROS can be used normallyorangepi@orangepi:~$ **test_ros.sh**

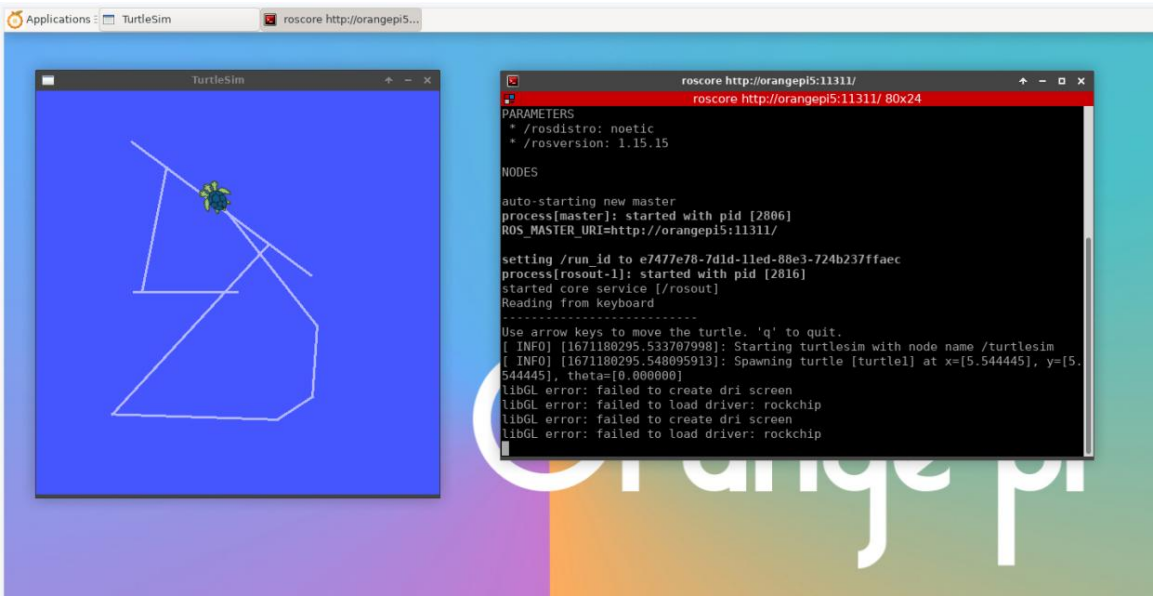
7) After running the **test_ros.sh** script, a little turtle as shown in the figure below will pop up



8) Then please keep the terminal window just opened at the top

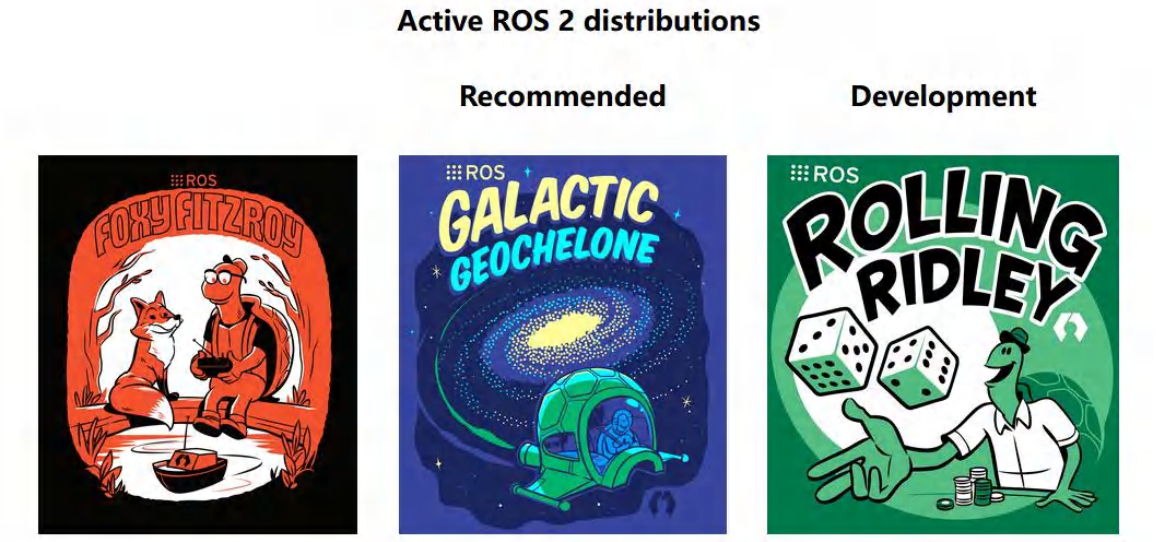


9) At this time, press the direction keys on the keyboard to control the little turtle to move up, down, left, and right



3.27.2. How to install ROS 2 Galactic on Ubuntu 20.04

1) The current active version of ROS 2 is as follows, the recommended version is **Galactic Geochelone**



Distro	Release date	Logo	EOL date
Humble Hawksbill	May 23rd, 2022		May 2027
Galactic Geochelone	May 23rd, 2021		November 2022
Foxy Fitzroy	June 5th, 2020		May 2023

http://docs.ros.org
http://docs.ros.org/en/galactic/Releases.html

2) The link to the official ROS 2 **Galactic Geochelone** installation documentation is as follows:

docs.ros.org/en/galactic/Installation.html
http://docs.ros.org/en/galactic/Installation/Ubuntu-Install-Debians.html

3) It is recommended to use Ubuntu20.04 in the official installation document of ROS 2 **Galactic Geochelone**, so please ensure that the system used by the development board is **Ubuntu20.04 desktop version**. There are several ways to install ROS 2. The following shows how to install ROS 2 **Galactic Geochelone** using **Debian packages**

4) Use the **install_ros.sh** script to install ros2

```
orangepi@orangepi:~$ install_ros.sh ros2
```

5) The **install_ros.sh** script will automatically run the **ros2 -h** command after installing ros2. If you can see the following print, it means that the ros2 installation is complete

```
usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...

ros2 is an extensible command-line tool for ROS 2.

optional arguments:
```

```
  -h, --help            show this help message and exit

Commands:
  action          Various action related sub-commands
  bag             Various rosbag related sub-commands
  component       Various component related sub-commands
  daemon          Various daemon related sub-commands
  doctor          Check ROS setup and other potential issues
  interface       Show information about ROS interfaces
  launch          Run a launch file
  lifecycle       Various lifecycle related sub-commands
  multicast       Various multicast related sub-commands
  node            Various node related sub-commands
  param           Various param related sub-commands
  pkg             Various package related sub-commands
  run             Run a package specific executable
  security        Various security related sub-commands
  service         Various service related sub-commands
  topic           Various topic related sub-commands
  wtf             Use `wtf` as alias to `doctor`

  Call `ros2 <command> -h` for more detailed usage.
```

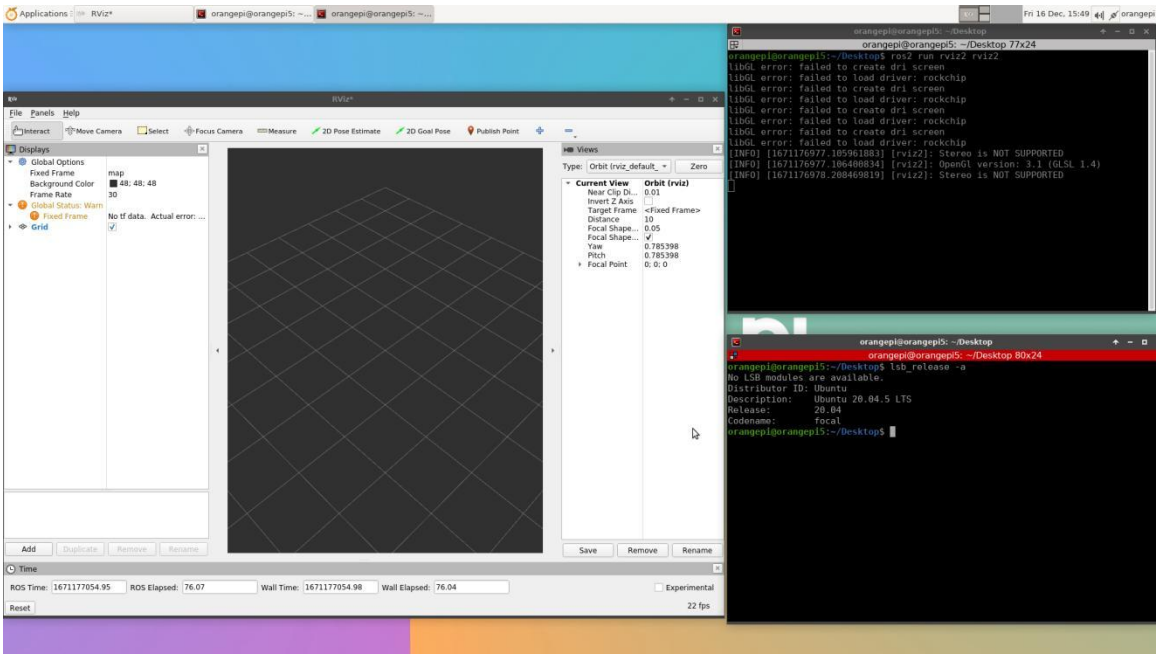
6) Then you can use the **test_ros.sh** script to test whether ROS 2 is installed successfully. If you can see the following print, it means that ROS 2 can run normally

```
orangepi@orangepi3b:~$ test_ros.sh
[INFO] [1671174101.200091527] [talker]: Publishing: 'Hello World: 1'
[INFO] [1671174101.235661048] [listener]: I heard: [Hello World: 1]
[INFO] [1671174102.199572327] [talker]: Publishing: 'Hello World: 2'
[INFO] [1671174102.204196299] [listener]: I heard: [Hello World: 2]
[INFO] [1671174103.199580322] [talker]: Publishing: 'Hello World: 3'
[INFO] [1671174103.204019965] [listener]: I heard: [Hello World: 3]
```

7) Run the following command to open rviz2

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
```

```
orangepi@orangepi:~$ ros2 run rviz2 rviz2
```



8) For the usage of ROS, please refer to the documentation of ROS 2

```
http://docs.ros.org/en/galactic/Tutorials.html
```

3. 27. 3. How to install ROS 2 Humble on Ubuntu 22.04

1) Use the **install_ros.sh** script to install ros2

```
orangepi@orangepi:~$ install_ros.sh ros2
```

2) The **install_ros.sh** script will automatically run the **ros2 -h** command after installing ros2. If you can see the following print, it means that the ros2 installation is complete

```
usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...

ros2 is an extensible command-line tool for ROS 2.

optional arguments:
  -h, --help            show this help message and exit

Commands:
  action      Various action related sub-commands
  bag         Various rosbag related sub-commands
  component   Various component related sub-commands
```

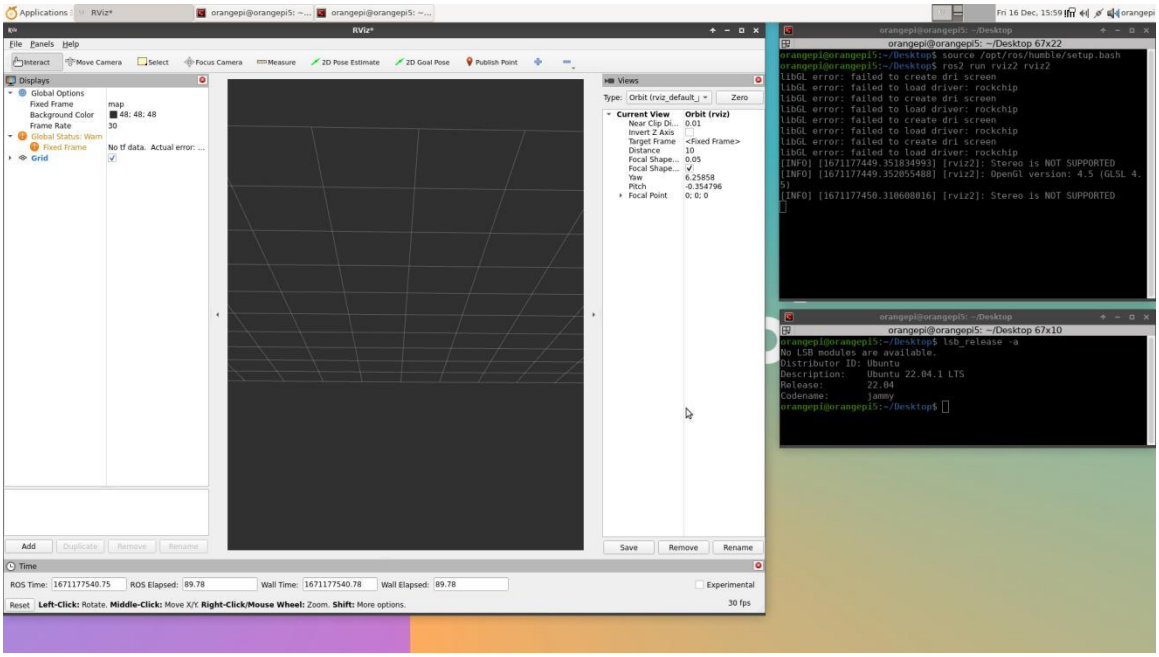
```
daemon      Various daemon related sub-commands
doctor      Check ROS setup and other potential issues
interface   Show information about ROS interfaces
launch      Run a launch file
lifecycle   Various lifecycle related sub-commands
multicast   Various multicast related sub-commands
node        Various node related sub-commands
param       Various param related sub-commands
pkg         Various package related sub-commands
run         Run a package specific executable
security    Various security related sub-commands
service     Various service related sub-commands
topic       Various topic related sub-commands
wtf         Use `wtf` as alias to `doctor`

Call `ros2 <command> -h` for more detailed usage.
```

3) Then you can use the **test_ros.sh** script to test whether ROS 2 is installed successfully. If you can see the following print, it means that ROS 2 can run normally

```
orangepi@orangepi3b:~$ test_ros.sh
[INFO] [1671174101.200091527] [talker]: Publishing: 'Hello World: 1'
[INFO] [1671174101.235661048] [listener]: I heard: [Hello World: 1]
[INFO] [1671174102.199572327] [talker]: Publishing: 'Hello World: 2'
[INFO] [1671174102.204196299] [listener]: I heard: [Hello World: 2]
[INFO] [1671174103.199580322] [talker]: Publishing: 'Hello World: 3'
[INFO] [1671174103.204019965] [listener]: I heard: [Hello World: 3]
```

4) Run the following command to open rviz2

```
orangepi@orangepi:~$ source /opt/ros/humble/setup.bash
orangepi@orangepi:~$ ros2 run rviz2 rviz2
```



5) Reference documents

http://docs.ros.org/en/humble/index.html
http://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html

3. 28. How to install kernel header files

1) The Linux image released by OPi comes with the deb package of the kernel header file by default, and the storage location is **/opt/**

```
orangepi@orangepi:~$ ls /opt/linux-headers*
/opt/linux-headers-legacy-rockchip-rk356x_x.x.x_arm64.deb
```

2) Use the following command to install the deb package of the kernel header file

The name of the kernel header file deb package needs to be replaced with the actual name, please do not copy it.

```
orangepi@orangepi:~$ sudo dpkg -i /opt/linux-headers-legacy-rockchip-rk356x_1.x.x_arm64.deb
```

3) After installation, you can see the folder where the kernel header files are located under **/usr/src**

```
orangepi@orangepi:~$ ls /usr/src
linux-headers-5.10.160-rockchip-rk356x
```

4) Then you can write a hello kernel module to test the kernel header file

a. First write the code of the hello kernel module, as follows:

```
orangepi@orangepi:~$ vim hello.c
#include <linux/init.h>
#include <linux/module.h>

static int hello_init(void)
{
        printk("Hello Orange Pi -- init\n");

        return 0;
}
static void hello_exit(void)
{
        printk("Hello Orange Pi -- exit\n");

        return;
}

module_init(hello_init);
module_exit(hello_exit);

MODULE_LICENSE("GPL");
```

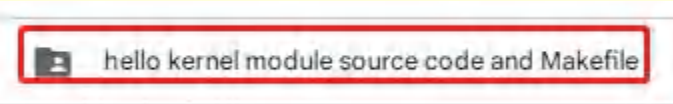
b. Then write the Makefile for compiling the hello kernel module, as follows:

```
orangepi@orangepi:~$ vim Makefile
ifneq ($(KERNELRELEASE),)
obj-m:=hello.o
else
KDIR :=/lib/modules/$(shell uname -r)/build
PWD  :=$(shell pwd)
all:
    make -C $(KDIR) M=$(PWD) modules
clean:
    rm -f *.ko *.o *.mod.o *.mod *.symvers *.cmd  *.mod.c *.order
```

```
endif
```

c. Then use the make command to compile the hello kernel module. The output of the compilation process is as follows:

If there is a problem with compiling the code you copied here, please go to the official tool to download the source code and upload it to the Linux system of the development board for testing.




```
orangepi@orangepi:~$ make
make -C /lib/modules/5.10.160-rockchip-35xx/build M=/home/orangepi modules
make[1]: Entering directory '/usr/src/linux-headers-5.10.160-rockchip-rk35xx'
  CC [M]  /home/orangepi/hello.o
  MODPOST /home/orangepi/Module.symvers
  CC [M]  /home/orangepi/hello.mod.o
  LD [M]  /home/orangepi/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.10.160-rockchip-rk35xx'
```

d. After compiling, the **hello.ko** kernel module will be generated

```
orangepi@orangepi:~$ ls *.ko
hello.ko
```

e. Use the **insmod** command to insert the **hello.ko** kernel module into the kernel

```
orangepi@orangepi:~$ sudo insmod hello.ko
```

f. Then use the **demsg** command to view the output of the **hello.ko** kernel module. If you can see the output below, it means that the **hello.ko** kernel module is loaded correctly.

```
orangepi@orangepi:~$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
```

g. Use the **rmmod** command to uninstall the **hello.ko** kernel module

```
orangepi@orangepi:~$ sudo rmmod hello
orangepi@orangepi:~$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
[ 3173.800892] Hello Orange Pi -- exit
```

3. 29. Use of the Raspberry PI's 5-inch screen

3. 29. 1. Assembly method of Raspberry PI 5-inch screen

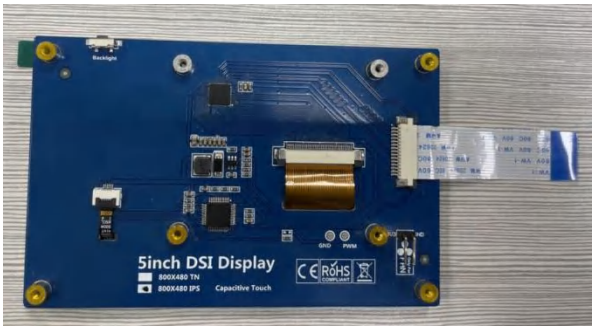
Note that linux6.6 is not supported yet.

1) First prepare the required accessories

a. Raspberry PI 5-inch MIPI LCD display + touch screen

b. 15pin MIPI cable

2) Then connect the 15pin MIPI cable to the Raspberry PI 5-inch screen in the way shown below (note the orientation of the insulation surface)



3) Finally connect to the LCD interface of the Orange Pi 3B development board

3. 29. 2. Open the Raspberry PI 5-inch screen configuration method

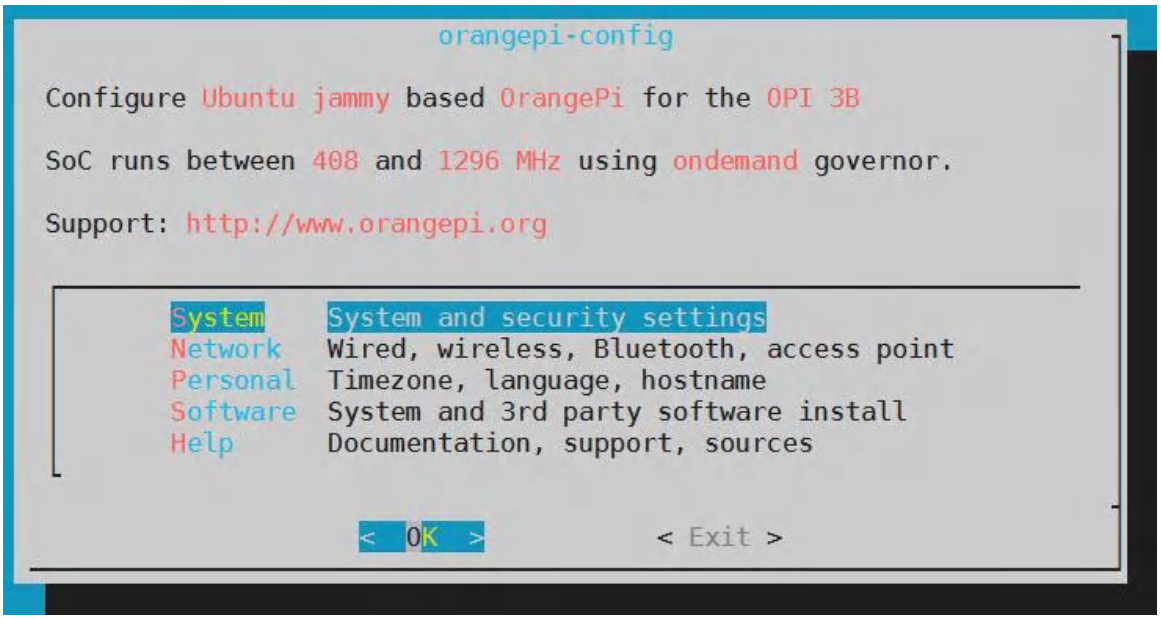
1) The Linux image does not open the 5-inch screen of the Raspberry PI by default. If you need to use the 5-inch screen of the Raspberry PI, you need to open it manually.

2) The steps to open the mipi lcd configuration are as follows:

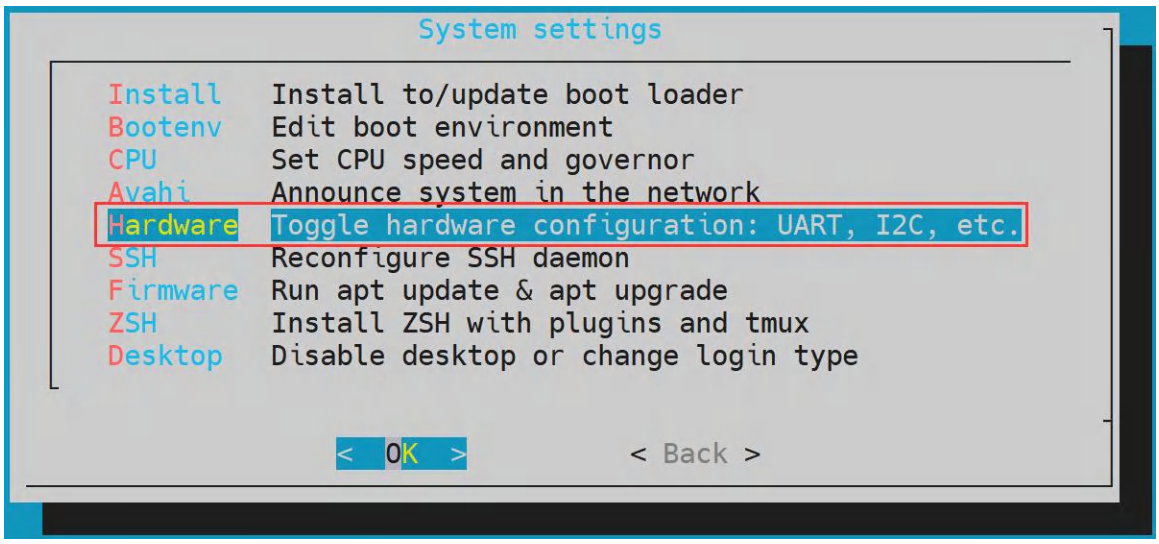
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

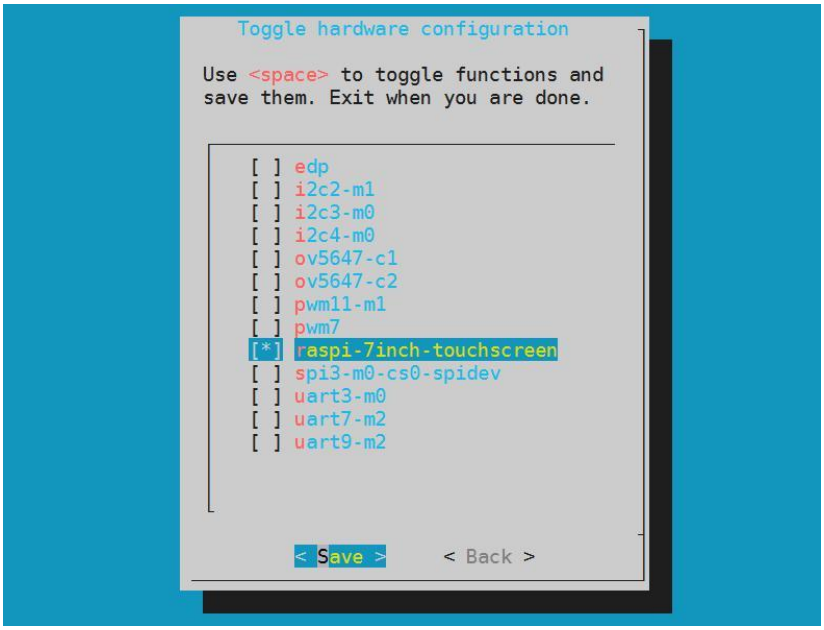
b. Then select **System**



c. Then select **Hardware**

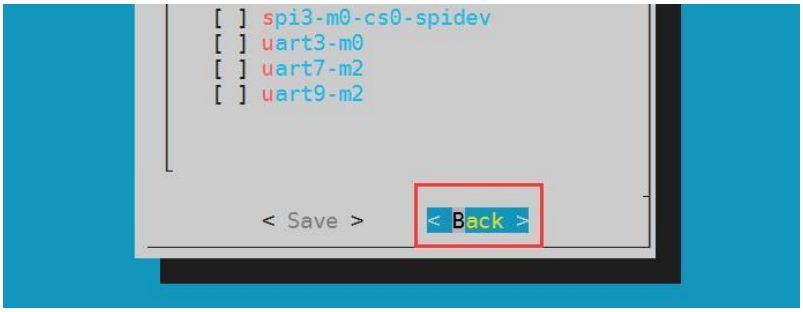


d. Then use the arrow keys of the keyboard to navigate to raspi-7inch-touchscreen, and then use the **space** to select

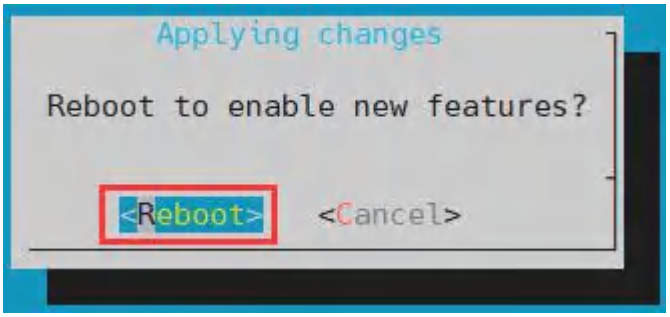


e. Then select **<Save>** to save

f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



The above settings will eventually add the configuration of overlays=raspi-7inch-touchscreen to /boot/orangepiEnv.txt. After setting, you can check it first. If this configuration does not exist, then there is a problem with the settings.

If you find it troublesome to use orangepi-config, you can also use the vim editor to open /boot/orangepiEnv.txt, and then add the configuration of overlays=raspi-7inch-touchscreen is also possible.

orangepi@orangepi:~$ **cat /boot/orangepiEnv.txt | grep "raspi"**
overlays=raspi-7inch-touchscreen #Sample configuration

3) After startup, you can see the lcd screen display as follows:

3. 29. 3. The method of server version image rotation display direction

1) Add **extraargs=fbcon=rotate:the direction to rotate** in **/boot/orangepiEnv.txt** This line configuration can set the direction displayed by the server version of the Linux system, where the number after **fbcon=rotate:** can be set as:

a. 0: normal screen (default is landscape)
b. 1: Turn clockwise 90 degrees
c. 2: Flip 180 degrees
d. 3: Turn clockwise 270 degrees

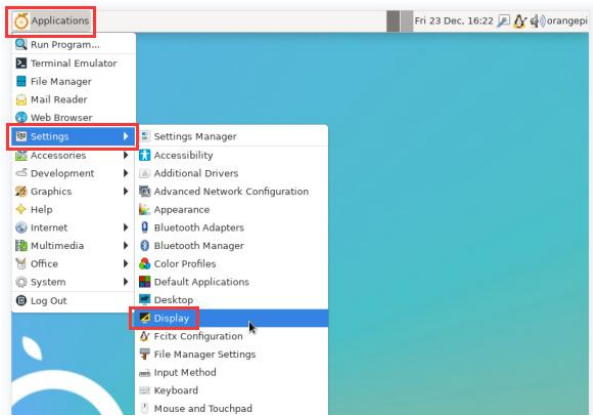
```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=lcd1
extraargs=cma=64M fbcon=rotate:3
```

Note that if there is the line extraargs=cma=64M in /boot/orangepiEnv.txt by default, the configuration fbcon=rotate:3 can be added after extraargs=cma=64M (separated by spaces).

2) Then **restart** the Linux system and you can see that the direction displayed on the LCD screen has been rotated

3. 29. 4. Method of rotating display and touch direction of desktop version image

1) First open **Display** Settings in Linux



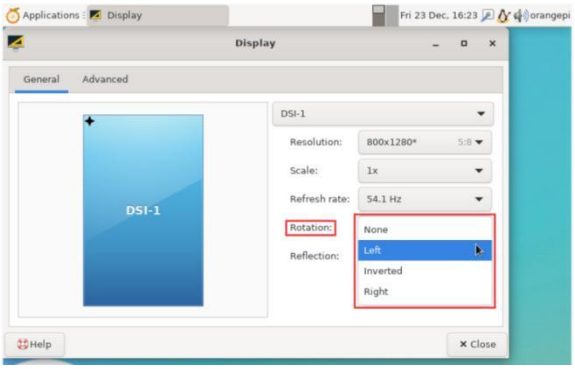
2) Then select the direction you want to rotate in **Rotation**

a. **None**: no rotation

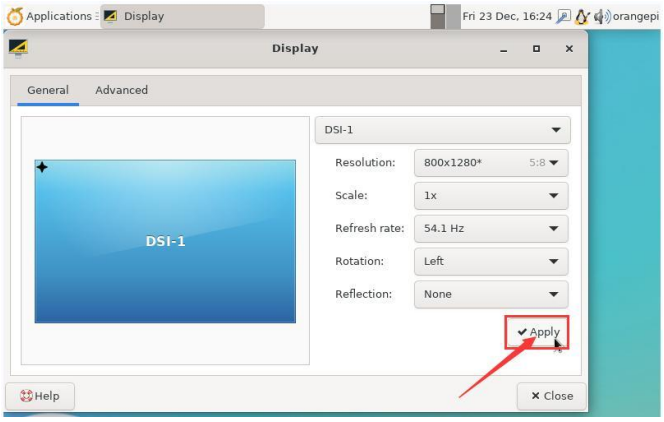
b. **Left**: rotate left 90 degrees

c. **Inverted**: Flip up and down, which is equivalent to rotating 180 degrees

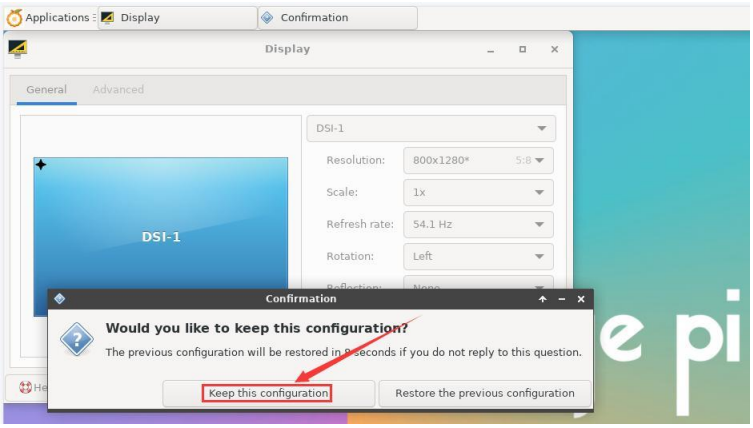
d. **Right**: rotate right 90 degrees



3) Then click **Apply**



4) Then select **Keep this configuration**



5) At this point, the screen display has been rotated, and then close the **Display** program

6) The above steps will only select the display direction, and will not rotate the direction of the touch. Use the **set_lcd_rotate.sh** script to rotate the direction of the touch. After the script is set, it will automatically restart, and then you can test whether the touch has been used normally.

a. **None**: no rotation

```
orangepi@orangepi:~$ set_lcd_rotate.sh none
```

b. **Left**: rotate left 90 degrees

```
orangepi@orangepi:~$ set_lcd_rotate.sh left
```

c. **Inverted**: Flip up and down, which is equivalent to rotating 180 degrees

```
orangepi@orangepi:~$ set_lcd_rotate.sh inverted
```

d. **Right**: rotate right 90 degrees

```
orangepi@orangepi:~$ set_lcd_rotate.sh right
```

The set_lcd_rotate.sh script mainly does four things:

1. **Rotate the direction displayed by the framebuffer**
2. **Rotate the direction of the touch**
3. **Turn off the boot logo**
4. **Restart the system**

Rotating the touch direction is achieved by adding the line Option "TransformationMatrix" "x x x x x x x x x" to /usr/share/X11/xorg.conf.d/40-libinput.conf Where "x x x x x x x x x" is configured

differently for different directions.

7) Touch rotation reference

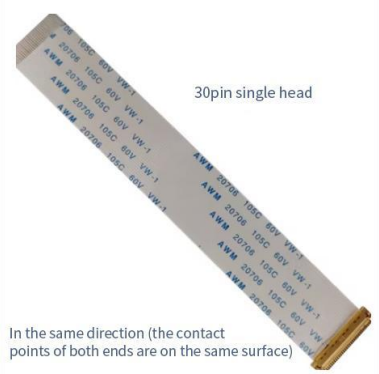
https://wiki.ubuntu.com/X/InputCoordinateTransformation

3. 30. How to use the eDP screen

3. 30. 1. Assembly method of eDP screen

Note that linux6.6 is not supported yet.

1) Currently only one eDP screen is compatible, including the following accessories:
   a. 0.5 pitch 30pin single-head cable in the same direction



   b. eDP display

2) Connect the 30pin eDP interface of the screen and the eDP interface of the development board with a 30pin single-head cable in the same direction

3. 30. 2. How to open the eDP screen configuration

Note that the method described below is only applicable to the adapted eDP screen. If the customer uses an unadapted screen, it cannot be turned on according to the following method.

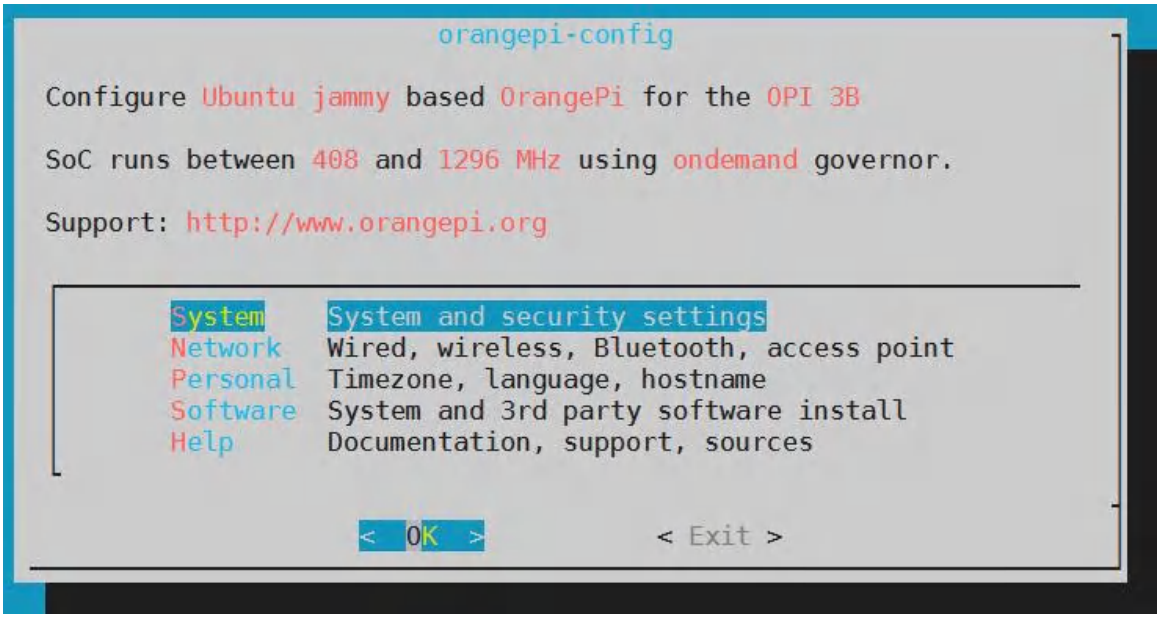
1) By default, the Linux image does not have the configuration to open the eDP screen. If you need to use the eDP screen, you need to open it manually.

2) The steps to open the eDP screen configuration are as follows:

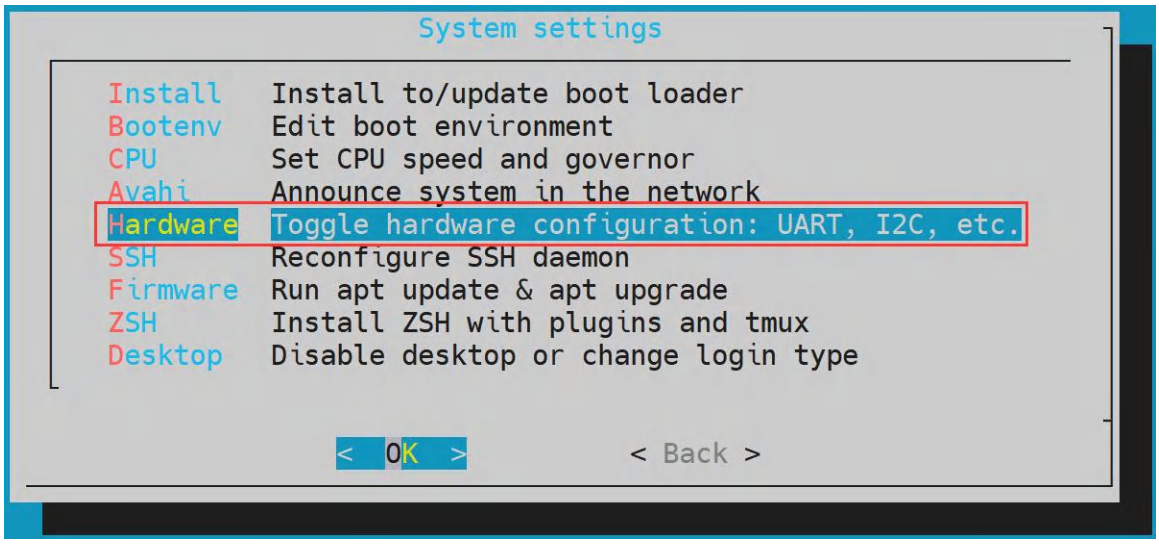
a. First run **orangepi-config**, ordinary users remember to add **sudo** permission

```
orangepi@orangepi:~$ sudo orangepi-config
```

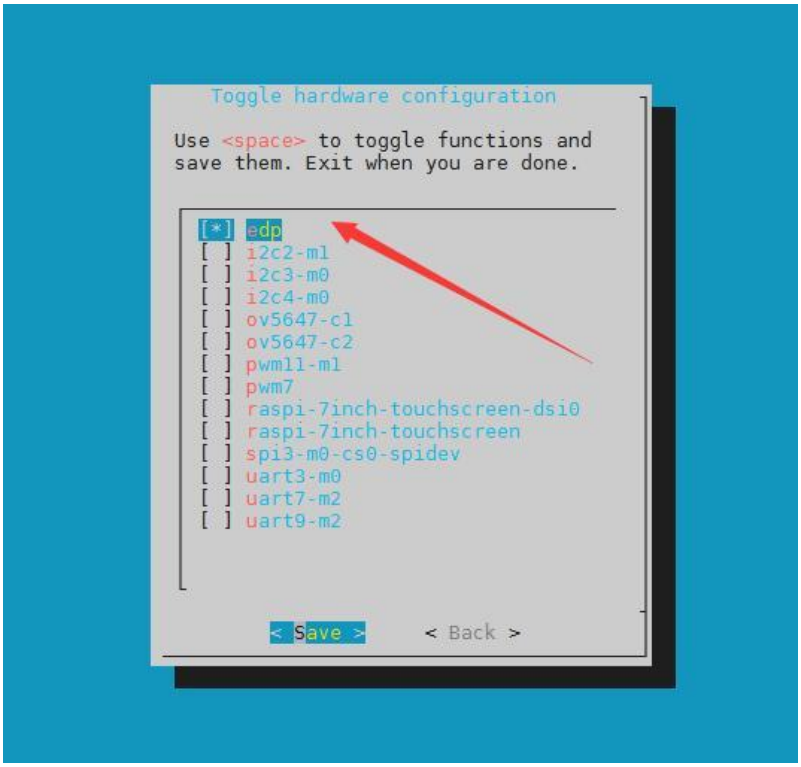
b. Then select **System**



c. Then select **Hardware**

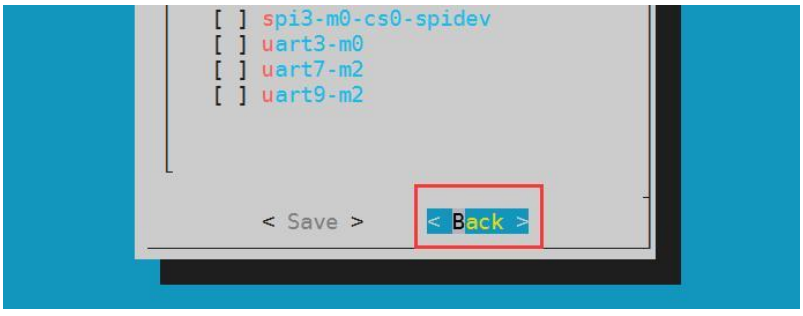


d. Then use the arrow keys on the keyboard to navigate to edp, and then use the **space** to select



e. Then select **<Save>** to save

f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect