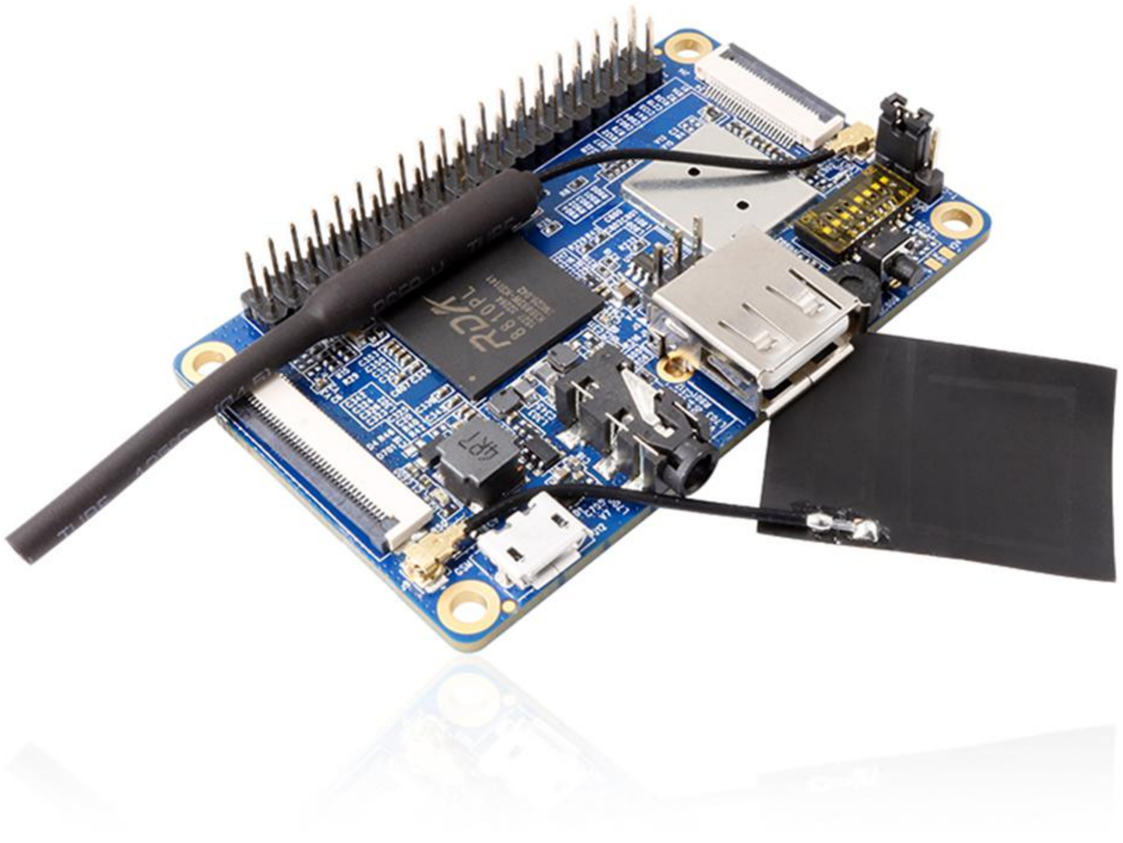
Orange Pi 2G-IOT User Manual



Content

I. Orange Pi Introduction

1. What is Orange Pi 2G-IOT?

It's an open-source single-board computer. It can run Android 4.4, Ubuntu, Debian, Raspberry Pi image. It uses the RDA8810 Soc, and has 256MB LPDDR2 SDRAM.

2. What can I do with Orange Pi 2G-IOT?

You can use it to build…

- A computer
- A wireless server
- Games
- Music and sounds
- HD video
- A speaker
- Android
- Scratch
- Pretty much anything else, because Orange Pi 2G-IOT is open source.

3. Whom is it for?

Orange Pi 2G-IOT is for anyone who wants to create with technology– not just consuming. It's a simple, fun, useful tool and you can use it to take control of the world around you.

4. Hardware specification

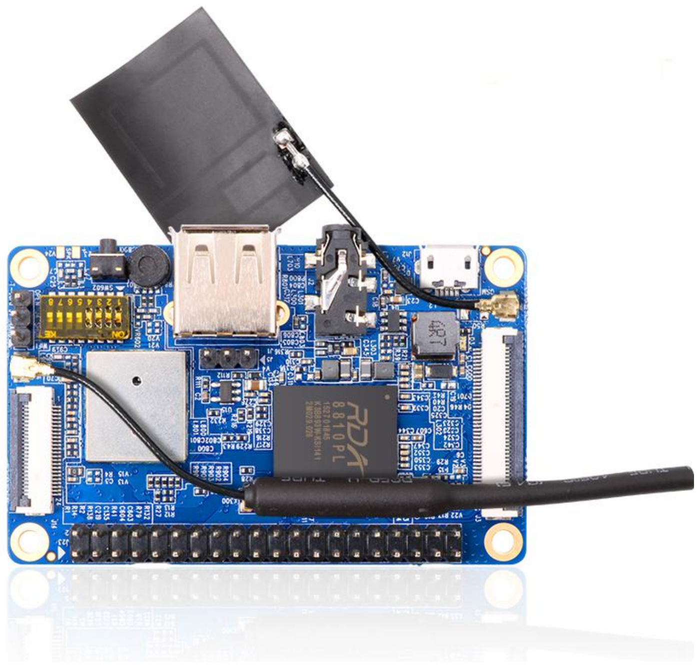
Hardware specification	
CPU	ARM Cortex-A5 32bit
GPU	Separate graphic processor, Vivante's GC860 support OpenGLES1.1/2.0 support OpenVG1.4 support DirectFB support GDI/DirecShow 30M Triangle/s, 250M Pixel/s
Memory (SDRAM)	Integrated 256MB LPDDR2 SDRAM
Onboard Storage	TF card / Integrated 500MB 8Bit 1.8V 4K SLC Nand Flash

Onboard WIFI+BT	RDA5991, WIFI+BT
2G model	The four frequency single card GSM/GPRS Dedicated accelerators SIM card
Video Input	A CSI input connector Camera： Supports 8-bit YUV422 CMOS sensor interface Supports CCIR656 protocol for NTSC and PAL Supports SM pixel camera sensor Supports video capture solution up to 1080p@30fps
Audio Input	MIC, 3.5 mm Jack
Video Outputs	LCD
Audio Output	3.5 mm Jack、 FM、SPEAK（Optional）
Power Source	USB OTG input can supply power Battery input can supply power（Optional）
USB 2.0 Ports	One USB 2.0 HOST, One USB 2.0 OTG
Buttons	Power Button(SW602)
Low-level peripherals	40 Pins Header, compatible with Raspberry Pi B+
GPIO(1x3) pin	UART, ground.
LED	Power led
Supported OS	Android, Ubuntu, Debian, Rasbian

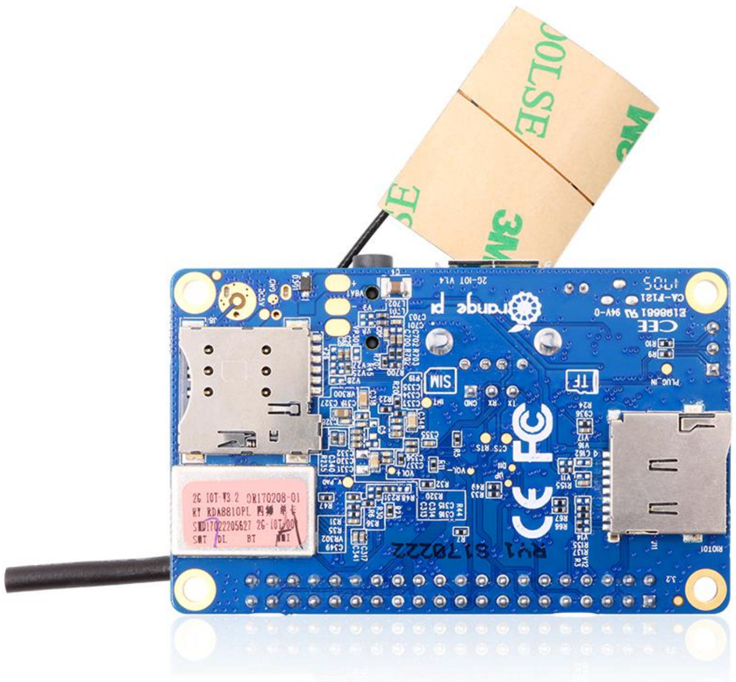
Interface definition

Product size	67mm × 42mm
Weight	35g
Orange Pi™ is a trademark of the Shenzhen Xunlong Software CO., Limited	

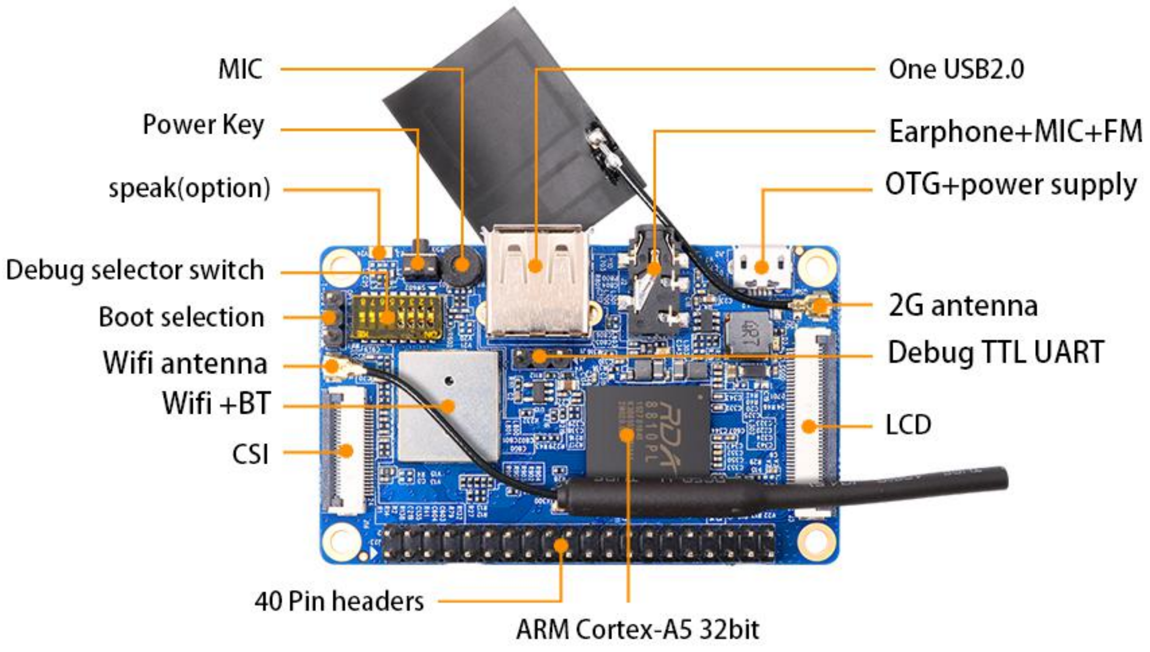
Top view

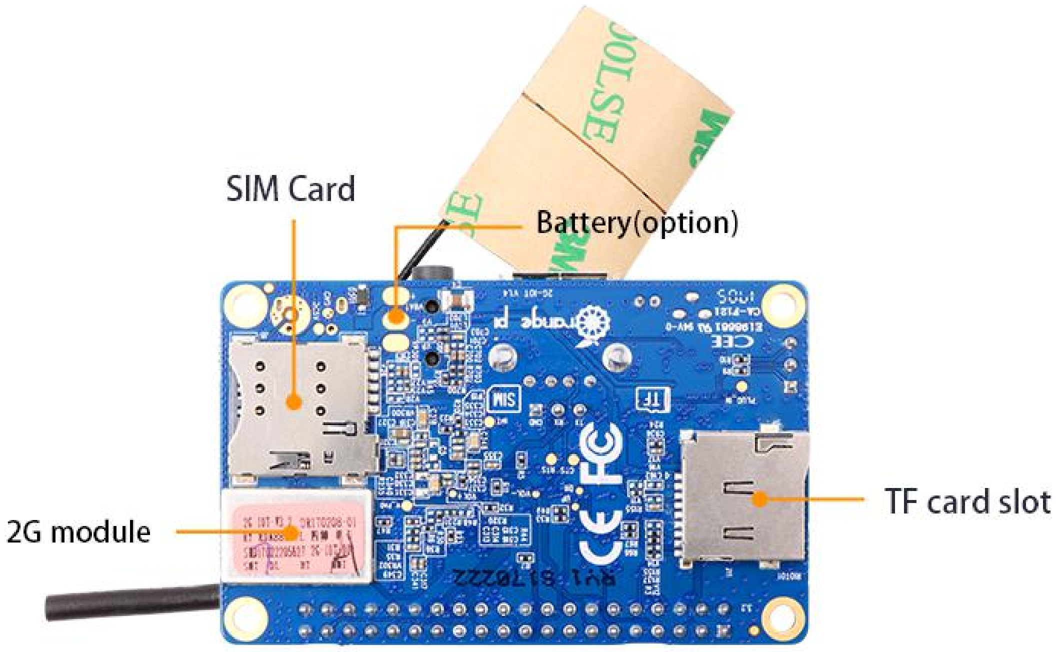


Bottom view



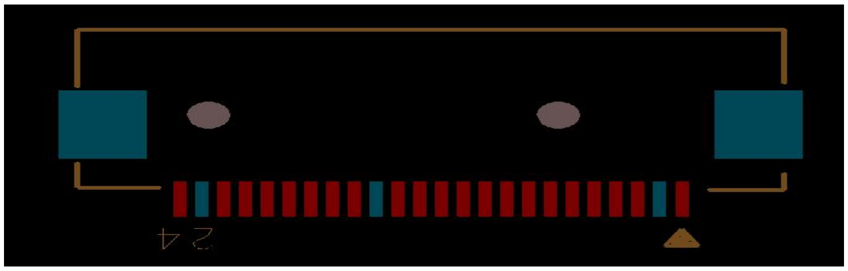
Interface instructions:





5. GPIO Specifications

The CSI Camera Connector is a 24-pin FPC connector which can connect external camera module with proper signal pin mappings. The pin of CIS connector can be defined as follows. The connector marked with "CON 1" on the Orange Pi 2G-IOT is camera connector.



OrangePi 2G-IOT-CSI

CON1-P01	NC	
CON1-P02	GND	
CON1-P03	TWI2-SDA	PE13
CON1-P04	VCC-CSI	
CON1-P05	TWI2-SCK	PE12
CON1-P06	CSI-RESET#	PE15
CON1-P07	CSI-VSYNC	PE3
CON1-P08	CSI-STBY-EN	PE15
CON1-P09	CSI-HSYNC	PE2
CON1-P10	VDD1V8-CSI	
CON1-P11	VCC-CSI	
CON1-P12	CSI-D7	PE11
CON1-P13	CSI-MCLK	PE1
CON1-P14	CSI-D6	PE10
CON1-P15	GND	
CON1-P16	CSI-D5	PE9
CON1-P17	CSI-PCLK	PE0
CON1-P18	CSI-D4	PE8
CON1-P19	CSI-D0	PE4
CON1-P20	CSI-D3	PE7
CON1-P21	CSI-D1	PE5
CON1-P22	CSI-D2	PE6
CON1-P23	GND	
CON1-P24	AFVCC-CSI	

II. Using Method

You can configure your Orange Pi in a very short period of time and use it according to the following steps. You need to fulfill the several steps before booting your Orange Pi.

1. Step 1: Prepare Accessories Needed

The first time you use the Orange Pi, you need at least some parts for the following:

No.	Items	Requirements and Instructions
1	TF card	8GB ; class 10 (for now it only supports 8GB SD card).Branded TF cards which are much more reliable are the good choice
2	Power adapter	At lease 5V/2A high quality power adapter, OTG could use as power supply.
3	Keyboard and mouse	Any keyboard and mouse with USB port is applicable; Keyboard and mouse are high-power, so a USB concentrator is required.
4	TTL to USB cable	Support debug log in.
5	Audio cable (Optional)	You can select an audio cable with 3.5mm jack to feel stereo audio.
6	SIM Card (Optional)	Support 2G SIM card

TF card

OTG power adapter

2. Step 2: Prepare a TF Card

In order to be able to us Orange Pi normally, you must first install the operating system into the TF card or Nand. The following instructions will teach you how to write the operating system image file to the Windows and Linux Platform. For now this board could support boot from TF card with Android and Linux distro, and could support boot from Nand with Android. It will illustrate about how to write image into Nand.

1) Writing image into a SD card on Windows:

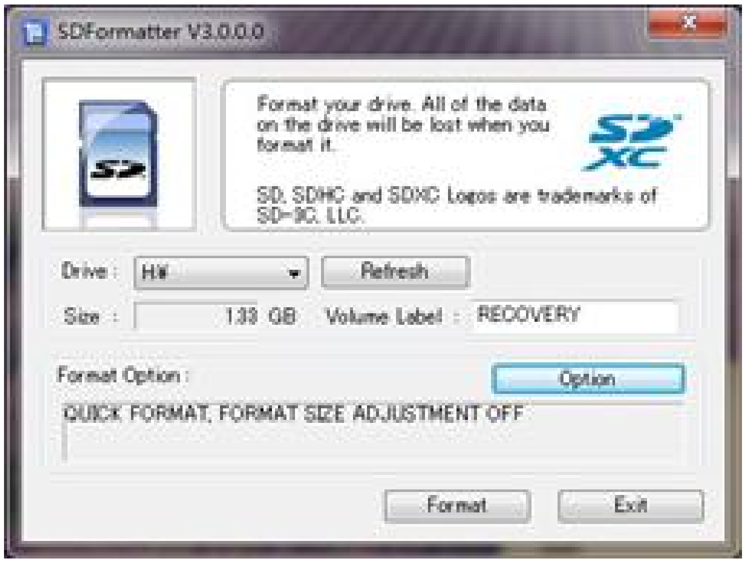
a. Inserting the TF card into the computer, the capacity of the card must be larger than the operating system image, usually requires 8GB or bigger capacity.

b. Formatting the TF card.

   i. Download tools for formatting TF card, such as TF Formatter, could be download from https://www.sdcard.org/downloads/formatter_4/eula_windows/

   ii. Unzip the downloaded files, and run *setup.exe*

   iii. In the *options settings* option set the format type option to quick formatting. *Logical size adjustment* option to open "(ON)”



SDFormatter V3.0.0.0

Format your drive. All of the data on the drive will be lost when you format it.

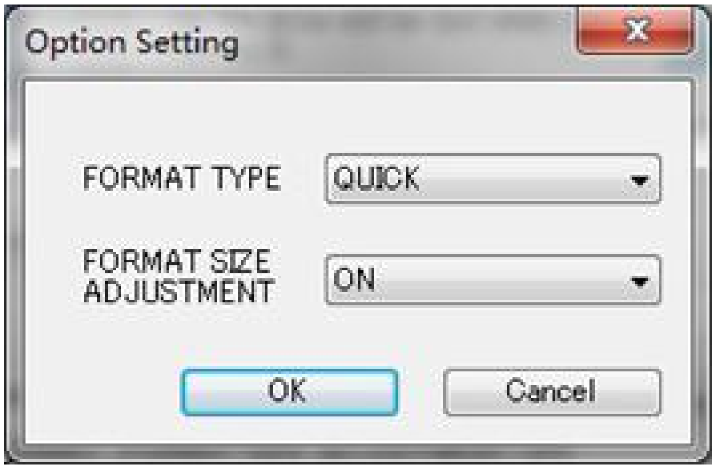
SD, SDHC and SDXC Logos are trademarks of SD-3C, LLC.

Drive : H:\

Size : 1.33 GB Volume Label : RECOVERY

Format Option :

QUICK FORMAT, FORMAT SIZE ADJUSTMENT OFF



Option Setting

FORMAT TYPE QUICK

FORMAT SIZE ADJUSTMENT ON

   iv. Make sure the inserted TF card codes are in accordance with the chosen codes.

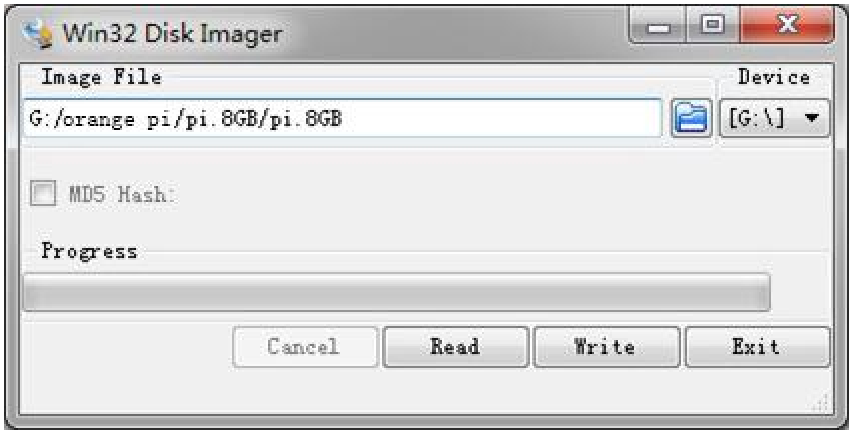
   v. Click the "Format"button.

c. Download the operating system image file from the download page, the page address is as follows: http://www.orangepi.cn/downloadresourcescn/

d. Unzip the downloaded file (in addition to the Android system, this method can be used to burn to write, the Android system need another burn, the following will introduce).

e. Right click the downloaded file, select "Unzip file" to write image to TF card.

i. Download tools to write image, such as **Win32 Diskimager**, http://sourceforge.net/projects/win32diskimager/files/Archive/.
ii. Select the path of image file that has been unzipped.



G:/orange pi/pi.8GB/pi.8GB

iii. Click the "Write" button and wait for the image writing.
iv. After the image is written, click the "Exit" button.

2) Writing image into a SD card on Linux:

a. Inserting the TF card into the computer, the capacity of the card must be larger than the operating system image, usually requires 8GB or bigger capacity.

b. Formatting the TF card.
  i. Run ***fdisk –l*** command to make sure TF disk.
  ii. Run ***umount /dev/sdxx*** to uninstall all partitions of TF Card.
  iii. Run ***sudo fdisk /dev/sdx*** command. Use director to delete all partitions of TF Card, and then us ***n*** command to add a new partition, finally use ***w*** command to save and exit.
  iv. Run ***sudo mkfs.vfat /dev/sdx1*** command to format the TF card partition set up last step to FAT32 form(according to your TF card disk to replace***x***). Or you could skip this step since command in Linux will format TF card automatic.

c. Download the image OS from download page:
  http://www.orangepi.cn/`downloadresourcescn/

d. Unzip the downloaded file and right click it, select " Unzip file"

e. Write image into TF card
  i. Run ***sudo fdisk –l*** command to make sure the TF card disk

ii. Make sure the image file ***hash key*** is the same as download page offered(optional) :

sha1sum [path]/[imagename]

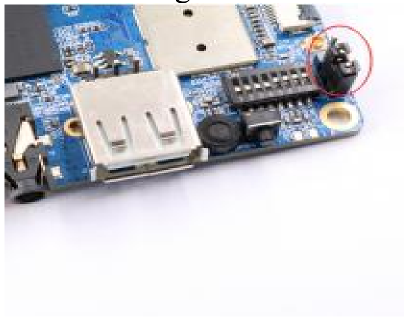
Here will be output some number which should be same as the image page line of *"SHA-1"*
iii. Run ***umount /dev/sdxx*** command to uninstall all partitions in TF Card

iv. Run the command of *sudo dd bs=4M if=[path]/[imagename] of=/dev/sdx* to write image file and wait for it finished. You can run *sudo pkill –USR1 –n –x dd* command to check the procedure.

3. Step 3: Start your Orange Pi

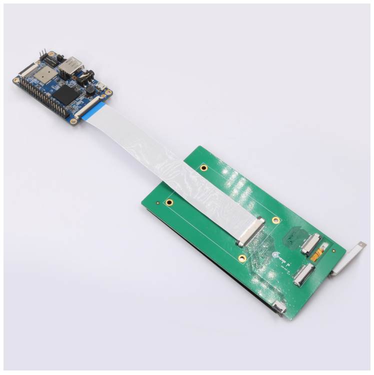
- Insert the TF card with written image into the TF card slot

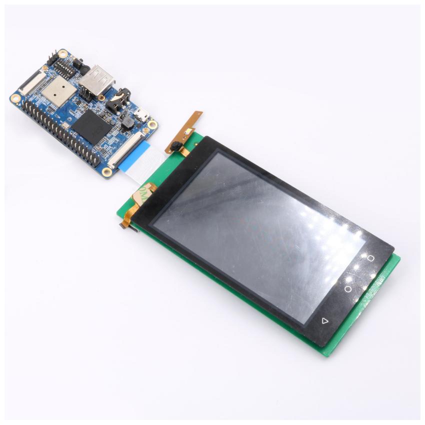
- Make sure the toggle switch is showing like the following, booting from SD card.

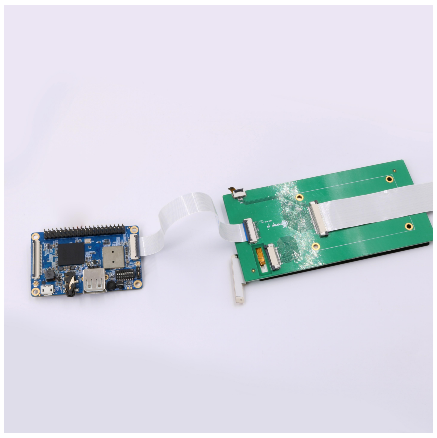


- Insert the keyboard or mouse into the USB port.
- Connect wifi antenna and base-band antenna

- Connect LCD and Camera

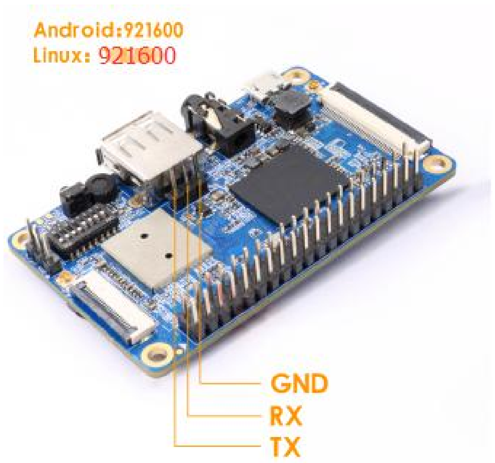




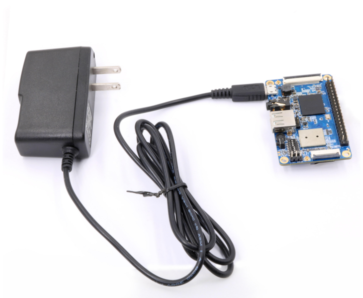


- Connect TTL cable, you could refer to the Debug method in this instruction.

Android and Linux use different Baud rate, please note the Baud rate setting.
Android Baud rate is 921600, Linux Baud rate is 921600
Serial port uses TTL to USB cable to connect.



- It is the power input interface on the right side for connecting a 5V and at least 2A or bigger than 2A power adapter. Avoid using smaller power GSM mobile phone charger, it is not able to output 2A even if it marked "5V/2A".



If the above steps are successful, the OrangePi will start in a few minutes. The monitor Graphical interface of display system. It may take a long time to start the first time, please wait patiently. The next time will boot very fast.

4. Step 4: Turn off your Orange Pi correctly

You can use the shutdown button on the interface to safety close the Orange Pi. You can also close the system by entering command in the shell:

```
sudo halt
or
sudo shutdown –h
```

It will be safety to turn off the Orange Pi. If directly use the power button to shut down the system may damage the file system on TF Card. After the system is closed, the power can be cut off by more than 5 seconds' press. If all the above steps run, then your Orange Pi could shut down.

5. Initialize settings for your Linux system

You need to make some basic settings when it is you first time to use Linux on Orange Pi 2G-IOT, like wifi setting, audio setting, user setting.

1) Wifi setting on serial port

In the use of serial login system, enter the login password the system will prompt you to use the OrangePi_Settings tool to make some basic setting, including wifi setting. You could use the following command in the order line:

```
sudo OrangePi_Settings
> wifi settings
```

This setting include the functions of WIFI statue setting, wifi searching and connect to AP. You could use this method to set wifi.

2) Use ssh to connect wifi

You need to use two cellphones if you want to use this function. Please refer to this:

Orange Pi 2G-IOT is defaulted to connect the hotspot of orangepi, the password is orangepi. Use another cellphone's hotspot function, setting the hot spot name as orangepi, password as orangepi. It will connect to orangepi hotspot default after booting the system. After that, use another cellphone to connect the hotspot, and use "wifi assistant" to check the IP of Orange Pi 2G-IOT.

After getting the IP of Orange Pi 2G-IOT, you could use SSH remote login in Linux

PC or Windows PC. Command as following:

ssh orangepi@192.168.xxx.xxx

Password: orangepi

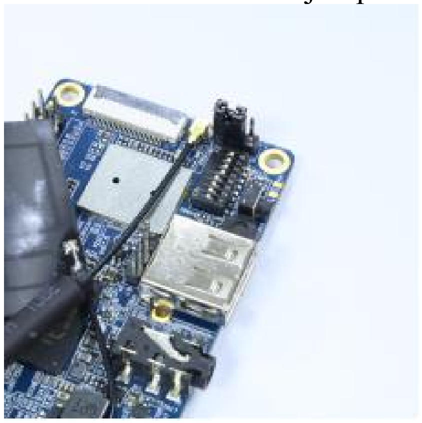
After enter the system via ssh, run the following command to connect to router:
sudo OrangePi_Settings

6. Write Android into Nand

Orange Pi 2G-IOT is supported boot from Nand, and also supported update Android in Nand.

1) Boot Android from NAND

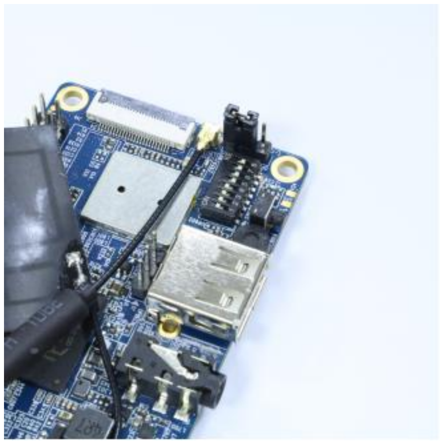
Switching the boot mode into NAND via short jumper cap.



Power it on, Orange Pi 2G-IOT will boot from NAND.

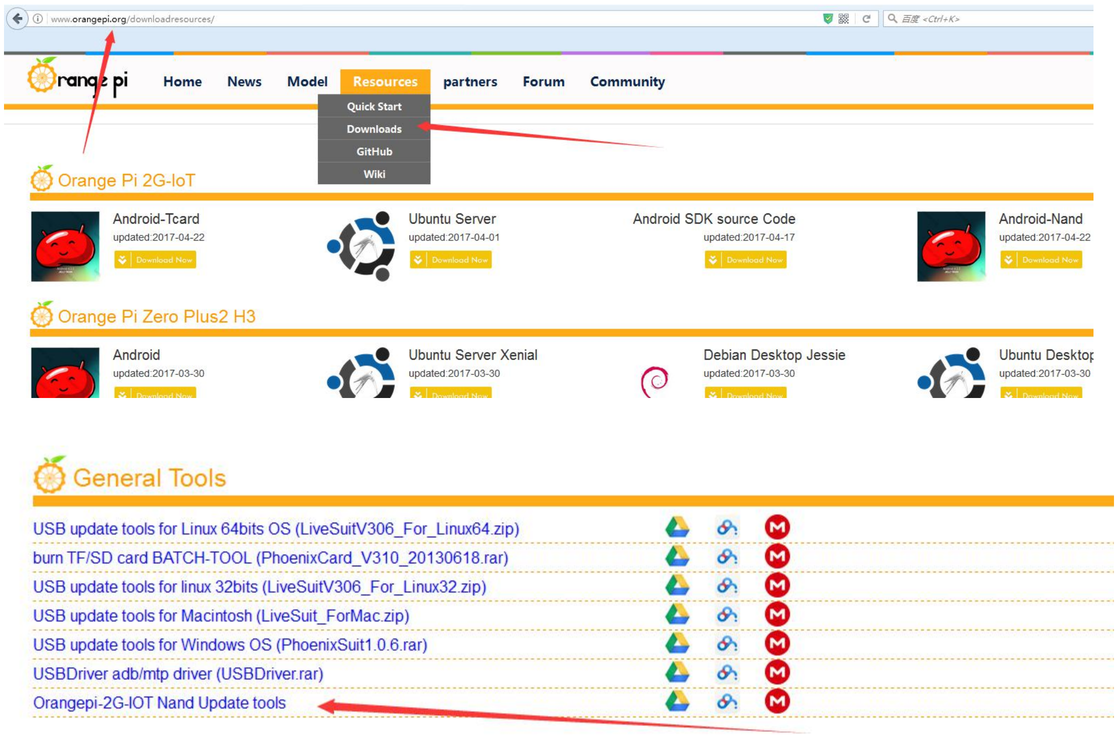
2) Update Android in NAND

- Short jumper cap to switch the system to boot from NAND, set toggle switch into 1234 UP, 5678 Down like the following:



3) Install writing tool on Windows

For now Nand writing tool could only support working on Windows, you could download the tool from official website: http://www.orangepi.org/downloadresources/



Orange Pi 2G-IoT

Android-Tcard updated:2017-04-22	Ubuntu Server updated:2017-04-01	Android SDK source Code updated:2017-04-17	Android-Nand updated:2017-04-22

Orange Pi Zero Plus2 H3

Android updated:2017-03-30	Ubuntu Server Xenial updated:2017-03-30	Debian Desktop Jessie updated:2017-03-30	Ubuntu Desktop updated:2017-03-30

General Tools

USB update tools for Linux 64bits OS (LiveSuitV306_For_Linux64.zip)
burn TF/SD card BATCH-TOOL (PhoenixCard_V310_20130618.rar)
USB update tools for linux 32bits (LiveSuitV306_For_Linux32.zip)
USB update tools for Macintosh (LiveSuit_ForMac.zip)
USB update tools for Windows OS (PhoenixSuit1.0.6.rar)
USBDriver adb/mtp driver (USBDriver.rar)
Orangepi-2G-IOT Nand Update tools

4) Install USB driver on Windows

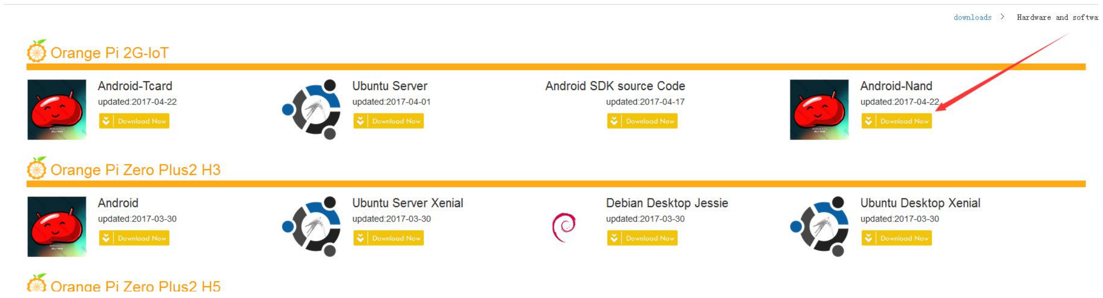
Unzip the tool file, install the USB driver, here is the path:
*/OrangePi_2G-IOT_Toolschain/USB_Driver/USB-driver/
You should install it according to your PC, if your PC is 32bit, then install x86 USB

driver, if it is 64bit, then is x64 USB driver.

5) Download Android Nand image

Here is the ink for Orange Pi 2G-IOT Nand version image:

http://www.orangepi.org/downloadresources/



downloads > Hardware and softwa

Orange Pi 2G-IoT

Android-Tcard updated:2017-04-22 Download Now

Ubuntu Server updated:2017-04-01 Download Now

Android SDK source Code updated:2017-04-17 Download Now

Android-Nand updated:2017-04-22 Download Now

Orange Pi Zero Plus2 H3

Android updated:2017-03-30 Download Now

Ubuntu Server Xenial updated:2017-03-30 Download Now

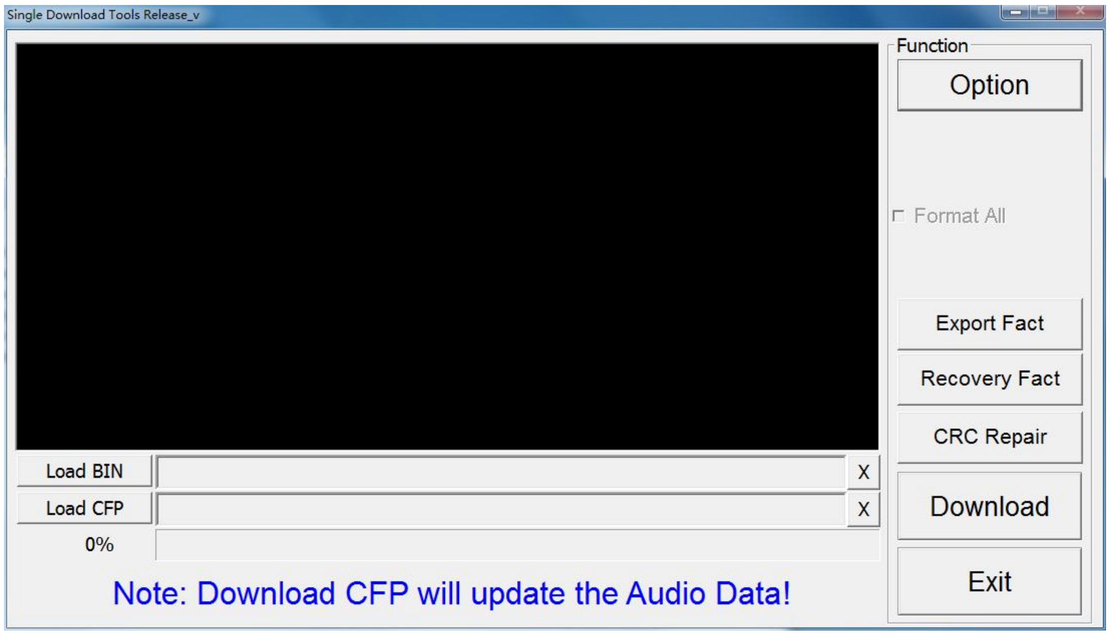
Debian Desktop Jessie updated:2017-03-30 Download Now

Ubuntu Desktop Xenial updated:2017-03-30 Download Now

Orange Pi Zero Plus2 H5

6) Use writing tool

Use writing tool to write NAND:
*/OrangePi_2G-IOT_Toolschain/OrangePi_2G-IOT_NandUpdate_Tools/OrangePi_2G-IOT_Update.exe.



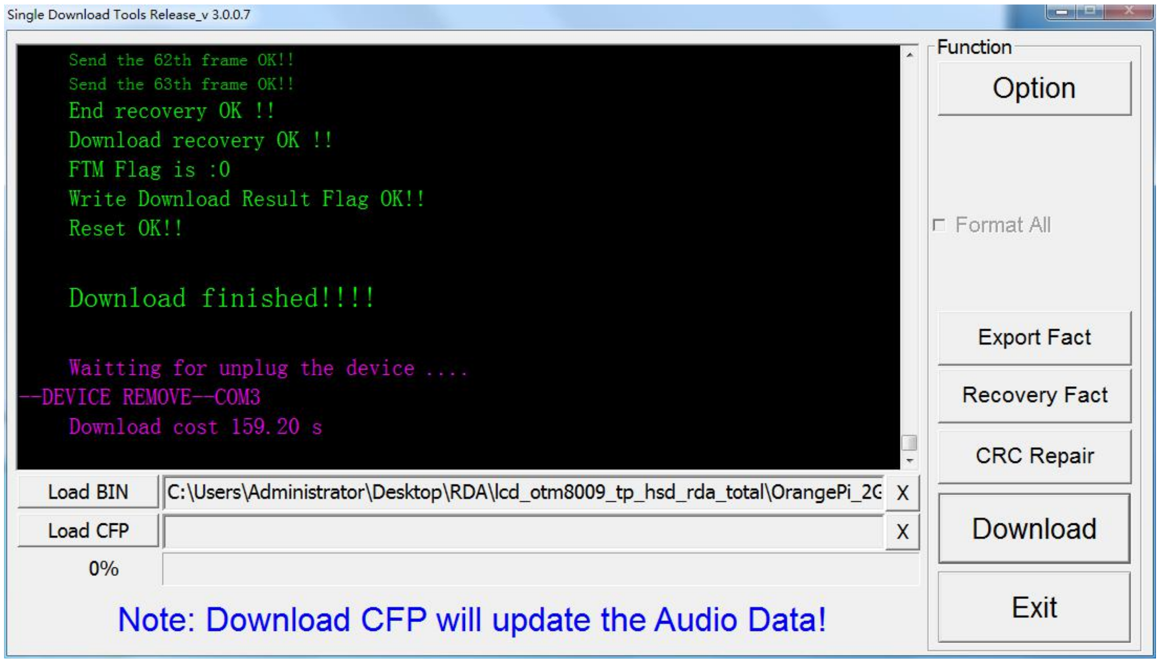
Note: Download CFP will update the Audio Data!

Click "load BIN" to import the image of NAND version into writing tool. After that, click Download button to download the image. Meanwhile, the tool is waiting for the download link of Orange Pi 2G-IOT.

7) Download Image

Prepare an Android USB to DC cable, first connect to the OTG port of Orange Pi 2G-IOT, then push on the power button for 5s, and connect the cable to the Windows PC. Meanwhile, the screen will indicate that connect successful and downloading. It will take around 3min to finished downloaded, after that, reboot the system and then the system will run on the update Android.

Note: If it could not download, please check the the shorting cap and switch.



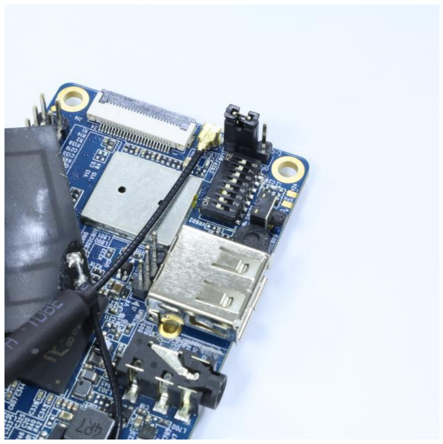
```
    Send the 62th frame OK!!
    Send the 63th frame OK!!
    End recovery OK !!
    Download recovery OK !!
    FTM Flag is :0
    Write Download Result Flag OK!!
    Reset OK!!

    Download finished!!!!

    Waitting for unplug the device ....
--DEVICE REMOVE--COM3
    Download cost 159.20 s
```

Load BIN: C:\Users\Administrator\Desktop\RDA\lcd_otm8009_tp_hsd_rda_total\OrangePi_2G

Note: Download CFP will update the Audio Data!

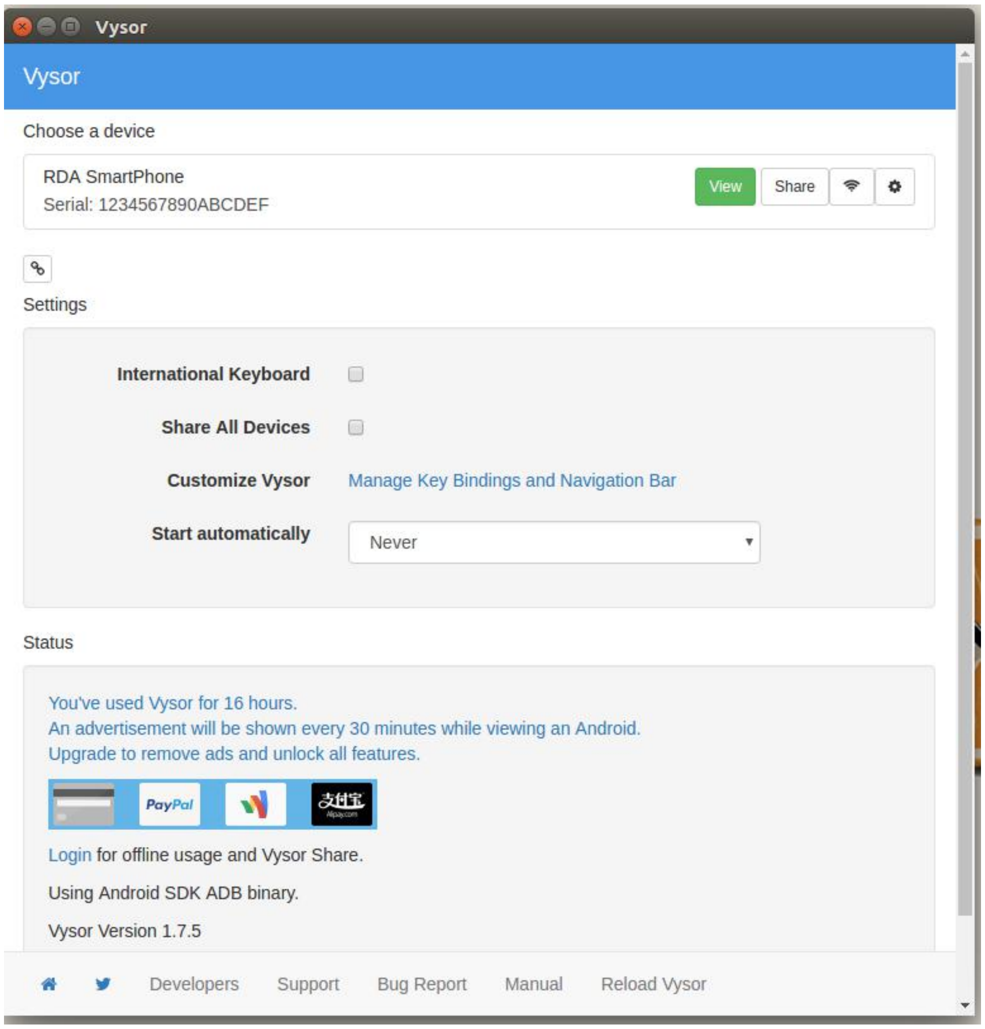
7. Android in no screen ADB mode

- ADB setting: Set the toggle switch into 1234 “UP”, 5678 “Down”, the system will switch into adb model, in this model, the USB is unable.



- Connect to the OTG port of Oragne Pi 2G-IOT with the USB to DC cable, the other side connect to PC, push the power button and then the system will be Android.
- If the PC haven't set on adb, then please refer to the teaching method of Ubuntu and Windows adb in internet. Use adb command in the PC terminal to connect the adb: adb shell
- After connect to OrangePi 2G-IOT via adb, you could refer to the adb debug method from the internet to enter into Orange Pi 2G-IOT

We would recommend you use Plug-in Vysor in Chrome browser, this tool could enter Android via adb:

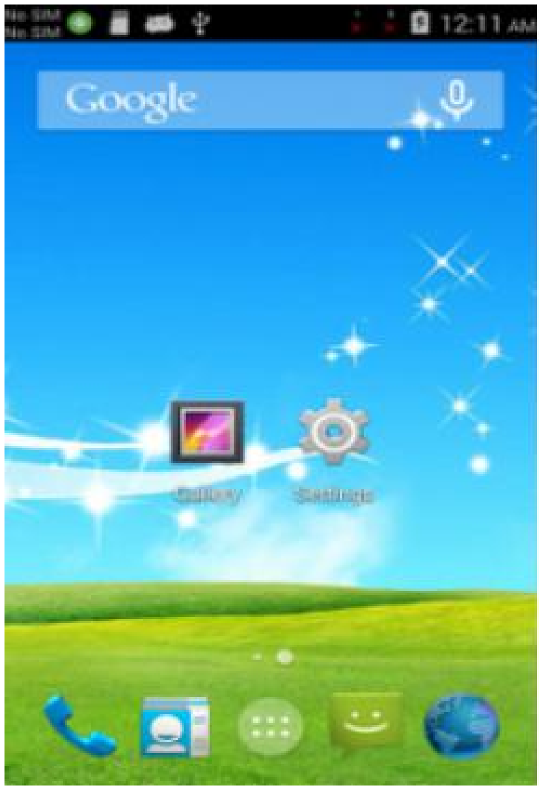


RDA SmartPhone
Serial: 1234567890ABCDEF

International Keyboard ☐
Share All Devices ☐
Customize Vysor: Manage Key Bindings and Navigation Bar
Start automatically: Never

You've used Vysor for 16 hours.
An advertisement will be shown every 30 minutes while viewing an Android.
Upgrade to remove ads and unlock all features.

Login for offline usage and Vysor Share.
Using Android SDK ADB binary.
Vysor Version 1.7.5



8. Universal software configuration

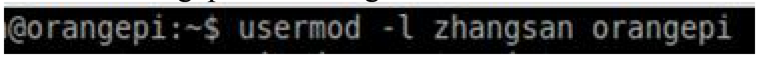
1) Change default account

The default log-in account and password is orangepi/orangepi or root/orangepi. It is recommended to modify the default orangepi account to your own account for secure sake. Take changing into Zhangsan as a sample. Steps are as follows:

a. Use root account to login Orange Pi

b. $ usermod -l zhangsan orangepi

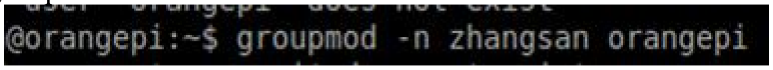
Change account of orangepi into Zhangsan



```
@orangepi:~$ usermod -l zhangsan orangepi
```

c. $ groupmod -n zhangsan orangepi

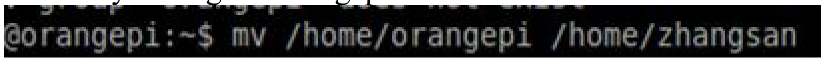
Change group



```
@orangepi:~$ groupmod -n zhangsan orangepi
```

d. $ mv /home/ornagepi /home/zhangsan

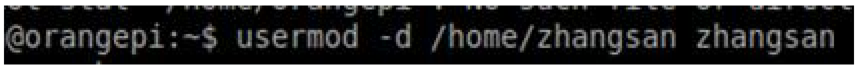
Change directory of original orangepi



```
@orangepi:~$ mv /home/orangepi /home/zhangsan
```

e. $ usermod -d /home/orangepi orangepi

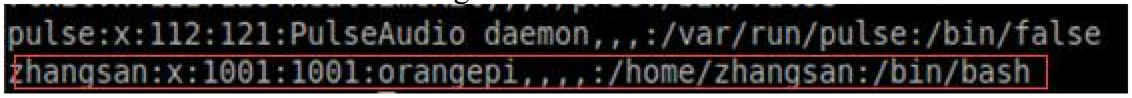
Set this directory into orangepi user's home directory



```
@orangepi:~$ usermod -d /home/zhangsan zhangsan
```

f. $ cat /etc/passwd

It should be shown as following:



```
pulse:x:112:121:PulseAudio daemon,,,:/var/run/pulse:/bin/false
zhangsan:x:1001:1001:orangepi,,,,:/home/zhangsan:/bin/bash
```

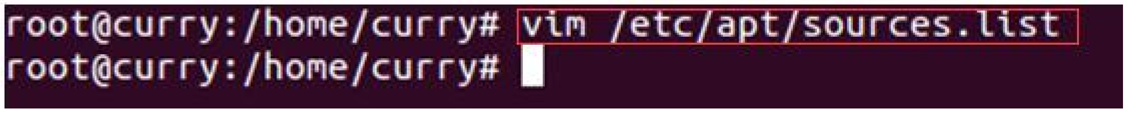
After the modification of the above steps, you could use the new account Zhangsan to log in.

2) System source configuration

This instruction will take Ubuntu as an example:

a. Open the source file

$ sudo vi /etc/apt/sources.list



```
root@curry:/home/curry# vim /etc/apt/sources.list
root@curry:/home/curry#
```

b. Edit source file

Replace the source file with your favourite source. Take an example of Ubuntu 16.04 on Zhonkeda source:

deb http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial main multiverse restricted universe

deb http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-backports main multiverse restricted universe

deb http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-proposed main multiverse restricted universe

deb http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-security main multiverse restricted universe

deb http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-updates main multiverse restricted universe

deb-src http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial main multiverse restricted universe

deb-src http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-backports main multiverse restricted universe

deb-src http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-proposed main multiverse restricted universe

deb-src http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-security main multiverse restricted universe

deb-src http://mirrors.ustc.edu.cn/ubuntu-ports/ xenial-updates main multiverse restricted universe

Note: xenial is the version of the code name in this source, if the other version of Ubuntu needs to replace the corresponding version code which can be found on the internet.

3) Enter the system via SSH

You could refer to the previous charter 5. 2)Use SSH to connect Wifi.

4) Modify the size of ext4 file system

It could promote system performance via expanding the rootfs partitions of file system after writing image, which could avoid the problems caused by insufficient space.

Expanding rootfs partitions on TF card of PC:

Using GParted to adjust the size:

Select the specified letter, right-click the corresponding letter, select "change the size" to adjust into the desired size, click "adjust the size", close the dialog box and click "apply to all operations", select the "apply" to complete the expansion operation.

a. Expand file system

i. Boot to Linux, umount /dev/sdb1 and /dev/sdb2, if it prompts disk busy, then use fuser to clean the using disk(we will recommend using another Linux booting disk to lead the system).

ii. Use fdisk /dev/sdb to adjust the partition size, after into it, enter p, and keep in mind about the initial position of needed extending size partition.

iii. Enter d to delete the partition need to change the size(my file system is /dev/sdb2, which is the 2 partition).

vi. Enter n to build a new partition, make sure the initial position is the same as you deleted, and enter the number as you desire.

v. Enter w to save the partition data.

vi. Use the following command to check the file system(make sure it is a right file system)

```
e2fsck -f /dev/sdb2
```

vii. Adjust the partition size

```
resize2fs /dev/sdb2
```

viii. It could mount a disk partition, you could check whether it has changed.

b. Shrink file system

i. Boot to Linux, umount /dev/sdb1 and /dev/sdb2, if it prompts disk busy, then use fuser to clean the using disk(we will recommend using another Linux booting disk to lead the system).

ii. Use the following command to check the file system(make sure it is a right file system)

```
e2fsck -f /dev/sdb2
```

iii. Modify the size of file system(Use resize2fs)

```
resize2fs /dev/sdb2 900M
```

The "s"after the number represents specifying the size of file system via the sectors(every sector calculated by 512 bite). You could also specify it into K(KB), M(MB), G(GB), etc.

vi. Use fdisk /dev/sdb to adjust the partition size, after into it, enter p, and keep in mind about the initial position of needed extending size partition. You need to first delete the partition then build a new one because the fdisk could not modify the size dynamic(you need to calculate the size, it have to enough to contain the file system adjusted in last step).

v. Enter d to delete the partition need to change the size(my file system is /dev/sdb2, which is the 2 partition).

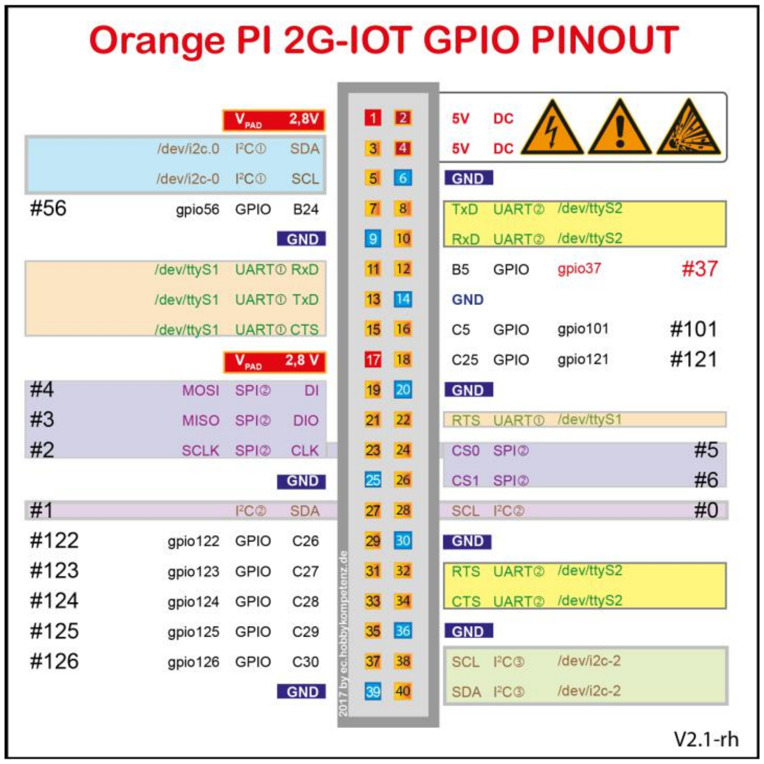
vi. Enter n to build a new partition, make sure the initial position is the same as you deleted, and enter the number as you desire. Besides, if it is boot-able partition you want to change, note that need to keep the boot-able mark in case cannot boot.

The above illustration is using fdisk and resize2fs to modify partition and file system, you could also use gparted. Gparted has graphical interface and it could help you to re-size file system at the same time of re-sizing partition. Goarted is much easier to use and reduce the change to make mistake. For now our official Lubuntu and Raspbian could not use it.

9. Usage of GPIO and WiringPi

1) Use WiringPi on OrangePi 2G-IOT to connect network

OrangePi 2G-IOT could support WiringPi, you could have a try according to the introduction on this section. The following is 40 Pin of OrangePi.



Orange PI 2G-IOT GPIO PINOUT

		V_{PAD}	2,8V	1	2	5V	DC		
	/dev/i2c.0	I^2C①	SDA	3	4	5V	DC		
	/dev/i2c-0	I^2C①	SCL	5	6	GND			
#56	gpio56	GPIO	B24	7	8	TxD	UART②	/dev/ttyS2	
			GND	9	10	RxD	UART②	/dev/ttyS2	
	/dev/ttyS1	UART①	RxD	11	12	B5	GPIO	gpio37	#37
	/dev/ttyS1	UART①	TxD	13	14	GND			
	/dev/ttyS1	UART①	CTS	15	16	C5	GPIO	gpio101	#101
		V_{PAD}	2,8 V	17	18	C25	GPIO	gpio121	#121
#4	MOSI	SPI②	DI	19	20	GND			
#3	MISO	SPI②	DIO	21	22	RTS	UART①	/dev/ttyS1	
#2	SCLK	SPI②	CLK	23	24	CS0	SPI②		#5
			GND	25	26	CS1	SPI②		#6
#1		I^2C②	SDA	27	28	SCL	I^2C②		#0
#122	gpio122	GPIO	C26	29	30	GND			
#123	gpio123	GPIO	C27	31	32	RTS	UART②	/dev/ttyS2	
#124	gpio124	GPIO	C28	33	34	CTS	UART②	/dev/ttyS2	
#125	gpio125	GPIO	C29	35	36	GND			
#126	gpio126	GPIO	C30	37	38	SCL	I^2C③	/dev/i2c-2	
			GND	39	40	SDA	I^2C③	/dev/i2c-2	

2017 by ec.hobbykompetenz.de

V2.1-rh

a. Download the latest WiringPi source code of Orange Pi 2G-IOT

Please confirm that the Orange Pi 2G-IOT has connect to wifi or network successfully, if no, then you need to first make it connect to wifi or network. If you connect to network, then you also need to install some essential tools:

```
sudo apt-get install git gcc make
```

b. Download the latest source code

You could download the latest WiringPi source code from official website:
www.orangepi.org
You could also download files from github:
https://github.com/OrangePiLibra/WiringPi.git with the following command:

```
env GIT_SSL_NO_VERIFY=true git clone
```

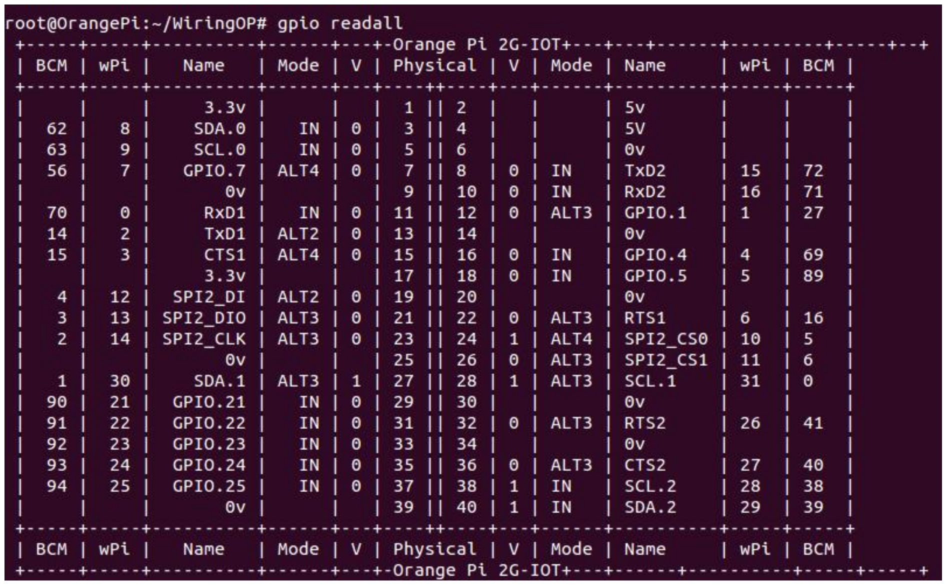
c. Compile and install WiringPi

Use the following command to compile and install the WiringPi after get the latest source code.

```
cd WiringOP/
./build OrangePi_2G-IOT
./build OrangePi_2G-IOT install
```

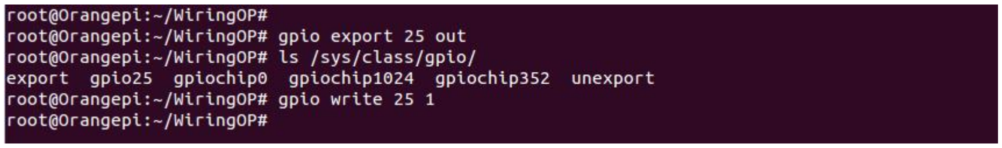
d. Test WiringPi with gpio command

You could use gpio command to test GPIO on 40pin on Orange Pi 2G-IOT with WiringPi installed.
i Use "gpio readall" command to print out all WiringPi pin mapping as following.
BCM line represents the actual hardware GPIO, there are 4 Groups GPIO and each Group have 32pins, and the serial number start from Group PA. PA0 corresponds to BCM colum number 0.
wPi line represents pins of wiringPi, you could use this group of data when use C library and gpio command on wiringPi. For example, number 37 pin is corresponding to number 25 pin of wiringPi, you could operate the 37 via operating 25pin.
Name line represents the definition name of Pin.
Mode line represents the mode of pin, it could but both input and output.
V line represents the voltage value of the current pin.
Physical line represents the actual hardware number.

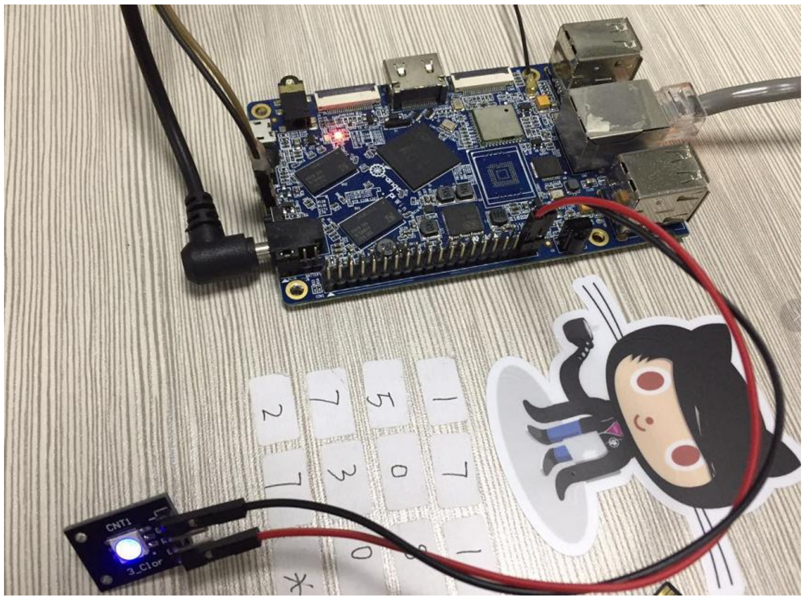


```
root@OrangePi:~/WiringOP# gpio readall
 +-----+-----+----------+------+---+-Orange Pi 2G-IOT+---+---+------+---------+-----+--+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 |     |     |     3.3v |      |   |  1 || 2  |   |      | 5v       |     |     |
 |  62 |   8 |    SDA.0 |   IN | 0 |  3 || 4  |   |      | 5V       |     |     |
 |  63 |   9 |    SCL.0 |   IN | 0 |  5 || 6  |   |      | 0v       |     |     |
 |  56 |   7 |   GPIO.7 | ALT4 | 0 |  7 || 8  | 0 | IN   | TxD2     | 15  | 72  |
 |     |     |       0v |      |   |  9 || 10 | 0 | IN   | RxD2     | 16  | 71  |
 |  70 |   0 |     RxD1 |   IN | 0 | 11 || 12 | 0 | ALT3 | GPIO.1   | 1   | 27  |
 |  14 |   2 |     TxD1 | ALT2 | 0 | 13 || 14 |   |      | 0v       |     |     |
 |  15 |   3 |     CTS1 | ALT4 | 0 | 15 || 16 | 0 | IN   | GPIO.4   | 4   | 69  |
 |     |     |     3.3v |      |   | 17 || 18 | 0 | IN   | GPIO.5   | 5   | 89  |
 |   4 |  12 |  SPI2_DI | ALT2 | 0 | 19 || 20 |   |      | 0v       |     |     |
 |   3 |  13 | SPI2_DIO | ALT3 | 0 | 21 || 22 | 0 | ALT3 | RTS1     | 6   | 16  |
 |   2 |  14 | SPI2_CLK | ALT3 | 0 | 23 || 24 | 1 | ALT4 | SPI2_CS0 | 10  | 5   |
 |     |     |       0v |      |   | 25 || 26 | 0 | ALT3 | SPI2_CS1 | 11  | 6   |
 |   1 |  30 |    SDA.1 | ALT3 | 1 | 27 || 28 | 1 | ALT3 | SCL.1    | 31  | 0   |
 |  90 |  21 |  GPIO.21 |   IN | 0 | 29 || 30 |   |      | 0v       |     |     |
 |  91 |  22 |  GPIO.22 |   IN | 0 | 31 || 32 | 0 | ALT3 | RTS2     | 26  | 41  |
 |  92 |  23 |  GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v       |     |     |
 |  93 |  24 |  GPIO.24 |   IN | 0 | 35 || 36 | 0 | ALT3 | CTS2     | 27  | 40  |
 |  94 |  25 |  GPIO.25 |   IN | 0 | 37 || 38 | 1 | IN   | SCL.2    | 28  | 38  |
 |     |     |       0v |      |   | 39 || 40 | 1 | IN   | SDA.2    | 29  | 39  |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+-Orange Pi 2G-IOT+---+------+----------+-----+-----+
```

ii Use "gpio export pin mode" to explore wiringPi GPIO to the directory of /sys/class/gpio and set the GPIO mode into mode.
According to the above WiringPi pin mapping, hardware pin number 37 is corresponding to WiringPi pin number GPIO 25, explore the number 25 and set it into output mode.

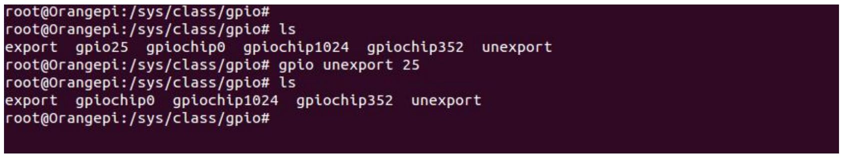


```
root@Orangepi:~/WiringOP#
root@Orangepi:~/WiringOP# gpio export 25 out
root@Orangepi:~/WiringOP# ls /sys/class/gpio/
export  gpio25  gpiochip0  gpiochip1024  gpiochip352  unexport
root@Orangepi:~/WiringOP# gpio write 25 1
root@Orangepi:~/WiringOP#
```



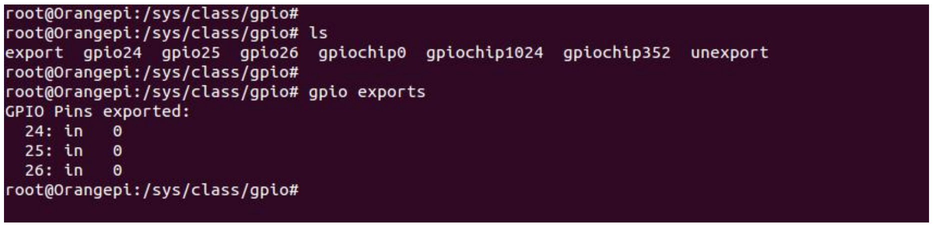
iii Use "gpio unexport pin" to cancel explore pin to /sys/class/gpio. For example:

orangepi# gpio unexport 25



```
root@Orangepi:/sys/class/gpio#
root@Orangepi:/sys/class/gpio# ls
export  gpio25  gpiochip0  gpiochip1024  gpiochip352  unexport
root@Orangepi:/sys/class/gpio# gpio unexport 25
root@Orangepi:/sys/class/gpio# ls
export  gpiochip0  gpiochip1024  gpiochip352  unexport
root@Orangepi:/sys/class/gpio#
```

iv Use "gpio exports" to check the current explored gpio. For expample:
orangepi# gpio exports



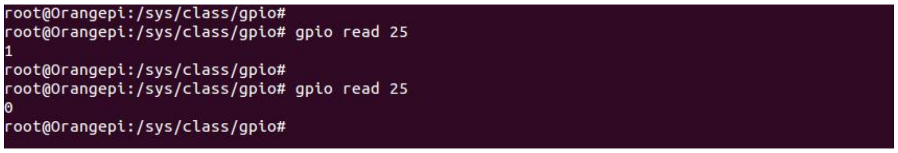
```
root@Orangepi:/sys/class/gpio#
root@Orangepi:/sys/class/gpio# ls
export  gpio24  gpio25  gpio26  gpiochip0  gpiochip1024  gpiochip352  unexport
root@Orangepi:/sys/class/gpio#
root@Orangepi:/sys/class/gpio# gpio exports
GPIO Pins exported:
  24: in   0
  25: in   0
  26: in   0
root@Orangepi:/sys/class/gpio#
```

v Use "gpio mode pin mode" command to configure wiringPi pin mode. For example:
Configure pin wiringPi 25 as output mode
orangepi# gpio mode 25 out
Configure pin wiringPi 26 as input mode
orangepi# gpio mode 26 in

vi. Use "gpio write pin value" to write value of output mode pin, for example:
Configure pin wiringPi 25 as output pin：
orangepi# gpio mode 25 out
Write 0 on wiringPi 25
orangepi# gpio write 25 0
Write 1 wiringPi 25
orangepi# gpio write 25 1

vii. Use "gpio read pin" command to read the value of input mode pin, for example:

Configure pin wiringPi 25 as input pin
orangepi# gpio mode 25 in
read the value from wiringPi 25
orangepi# gpio read 25



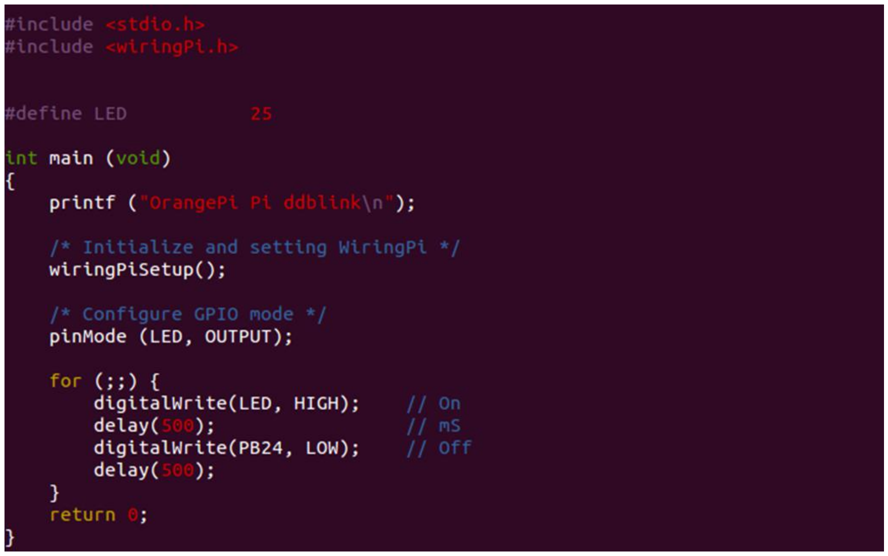
```
root@Orangepi:/sys/class/gpio#
root@Orangepi:/sys/class/gpio# gpio read 25
1
root@Orangepi:/sys/class/gpio#
root@Orangepi:/sys/class/gpio# gpio read 25
0
root@Orangepi:/sys/class/gpio#
```

viii. If you want to learn more "gpio" command, you could refer to "gpio -h" obtain.

e. Use WiringPi C Library

WiringPi support C library and python library, you could use C language to operation GPIO port, the example source code is on the directory of /example/OrangePi/. Here is an example for C library usage on GPIO:

Complie GPIO LED



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


#define LED           25

int main (void)
{
    printf ("OrangePi Pi ddblink\n");

    /* Initialize and setting WiringPi */
    wiringPiSetup();

    /* Configure GPIO mode */
    pinMode (LED, OUTPUT);

    for (;;) {
        digitalWrite(LED, HIGH);    // On
        delay(500);                 // mS
        digitalWrite(PB24, LOW);    // Off
        delay(500);
    }
    return 0;
}
```

Usage of C library on wiringPi:

In order to use wiringPi C library, first you need to import file of "wiringPi.h". You need to initialize wiringPi before using GPIO with function wiringPiSetup(). And then you could configure pin mode into INPUT or OUTPUT. And please note that the pin number should corresponding to wiringPi. Finally you could use function digitalWrite() and digitalRead() to read and write.

10. Connect to Network via GSM

1) Use 2G-IOT to send message

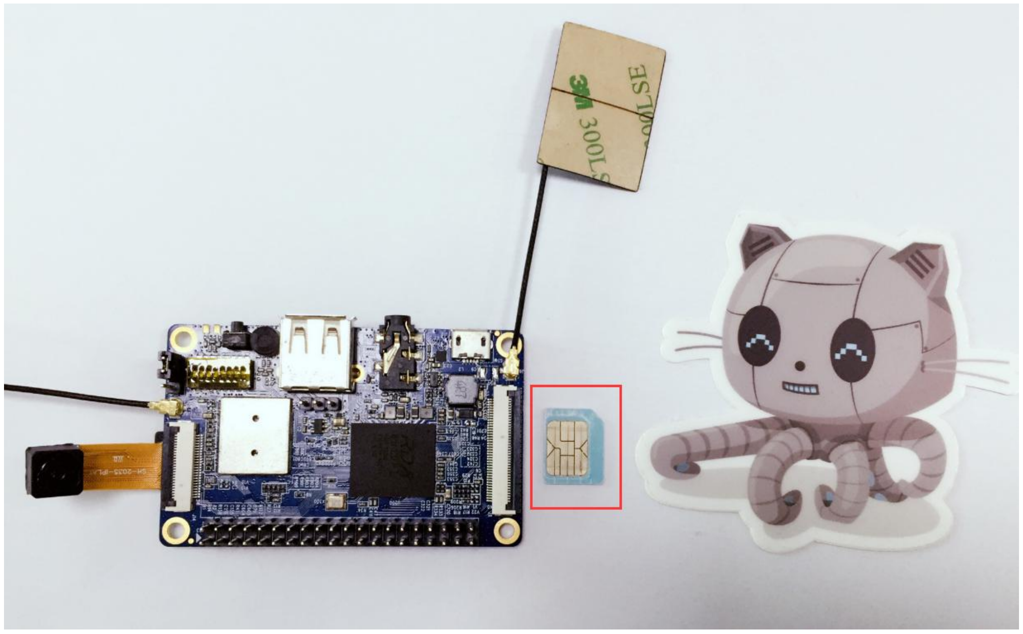
For now the version of Ubuntu,Debian and Raspbian could support SMS sending message. Before using the SMS function, please make sure the SIM card you use is active.

i. Prepare

First you need to prepare an activated Micro SIM card and insert into Orange Pi 2G-IOT, the Orange PI 2G-IOT could support the following frequency:

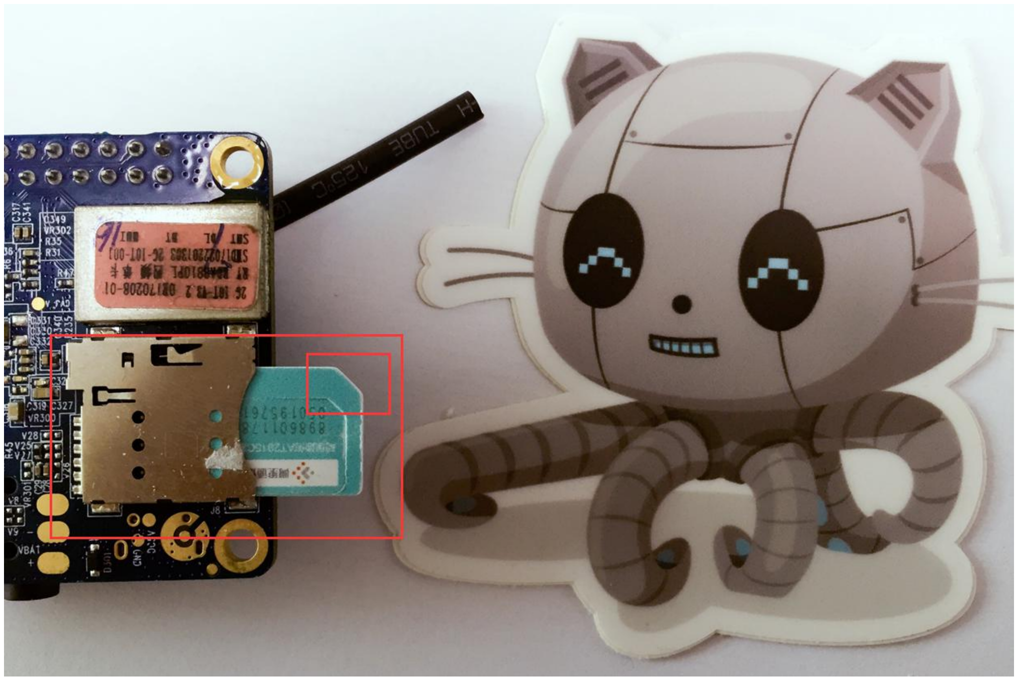
Frequency Range： 850,900,1800,1900

Standard: GSM800



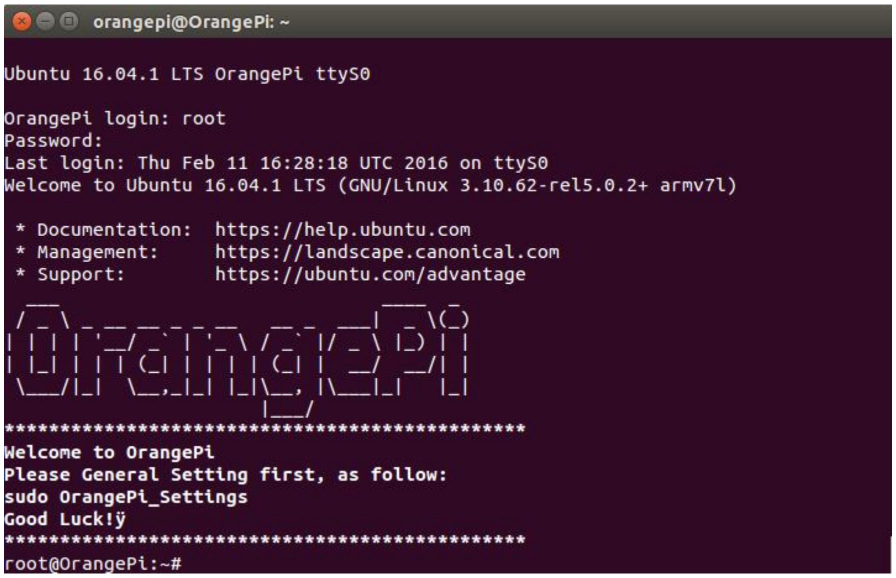
ii. Install SIM card

Please note the direction of when inserting SIM card.



iii. Login Linux system

After inserting SIM card, power on and enter into Linux system. You could login the system via serial port or SSH. If you are going to use serial port to login, please be remember to set the baud rate into 921600.



```
orangepi@OrangePi: ~

Ubuntu 16.04.1 LTS OrangePi ttyS0

OrangePi login: root
Password:
Last login: Thu Feb 11 16:28:18 UTC 2016 on ttyS0
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 3.10.62-rel5.0.2+ armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
  ___                            ____  _
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/
***********************************************
Welcome to OrangePi
Please General Setting first, as follow:
sudo OrangePi_Settings
Good Luck!ÿ
***********************************************
root@OrangePi:~#
```

iv. SMS Usage

There are many ways to use SMS, here will introduce the way with serial port and C language.

a. With Serial Port

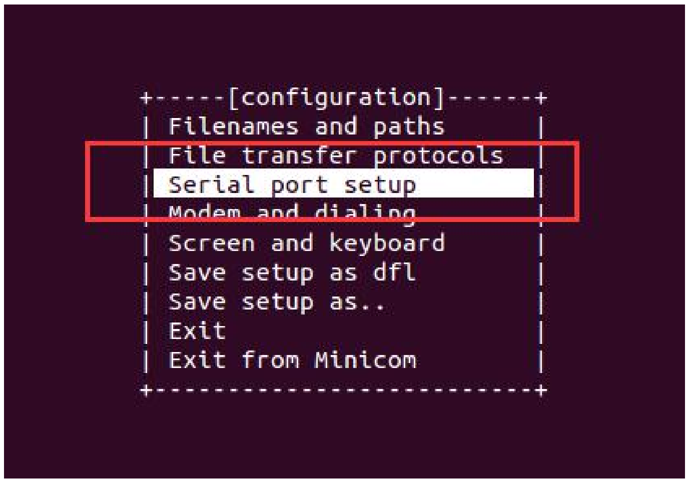
You need to install the tool of minicom on Orange Pi 2G-IOT before using SMS. Please make sure the board has already connect to wifi.
You could install minicom with the following command:

```
sudo apt-get install minico
```

The AP core and Modem will communicate via serial port. There will be a modem0 node after booting into Linux system. You could use serial port tool to connect to the /dev/modem0 node, which we recommend you use minicom, you could also try other tools.
If you are using the minicom or other tools on serial port, please refer to the following:
You could configure with the following command to connect /dev/modem0:

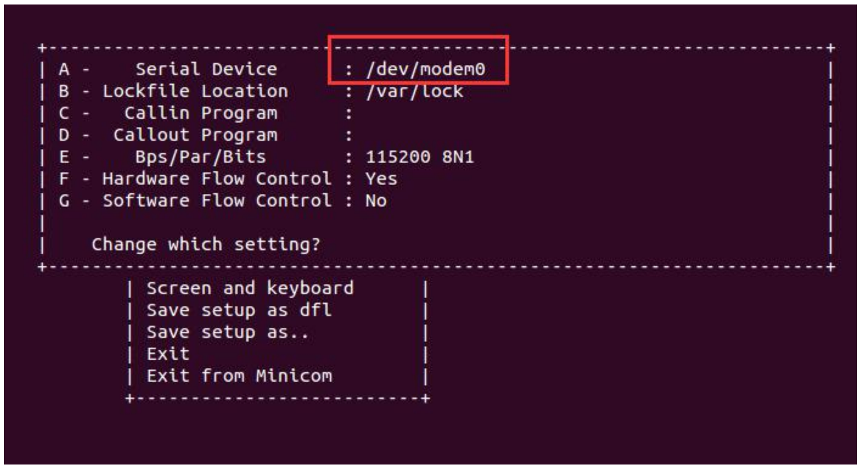
```
sudo minicmo -s
```

After input the command, you will enter the configure interface, select "**Serial port setup**"



```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

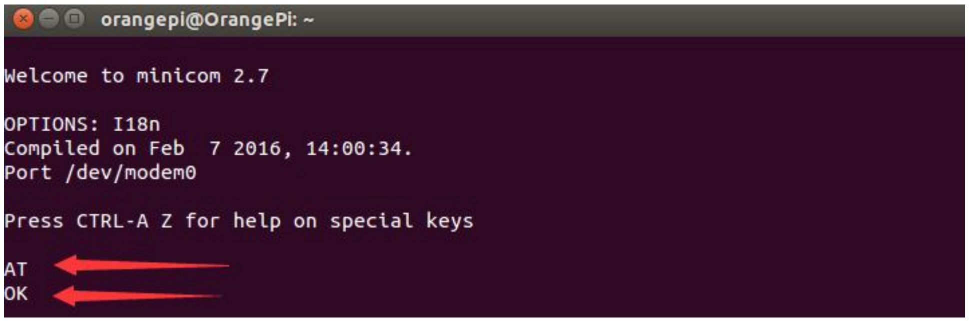
Push A to modify Serial Device into **/dev/modem0**



```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/modem0                               |
| B - Lockfile Location     : /var/lock                                 |
| C -   Callin Program      :                                           |
| D -  Callout Program      :                                           |
| E -    Bps/Par/Bits       : 115200 8N1                                |
| F - Hardware Flow Control : Yes                                       |
| G - Software Flow Control : No                                        |
|                                                                       |
|    Change which setting?                                              |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

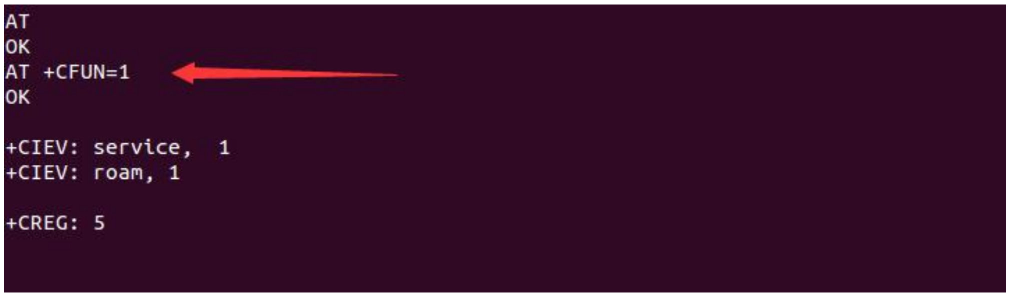
Use AT command to send message

First you need to input AT command to check the situation of Modem, if Modem replies OK, then means Modem is connect correctly.

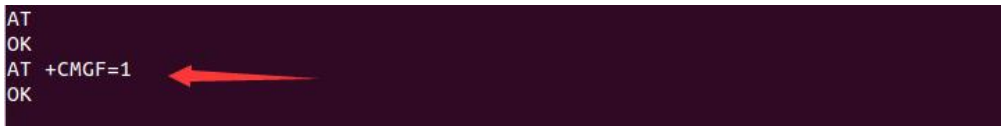


```
orangepi@OrangePi: ~

Welcome to minicom 2.7

OPTIONS: I18n
Compiled on Feb  7 2016, 14:00:34.
Port /dev/modem0

Press CTRL-A Z for help on special keys

AT
OK
```

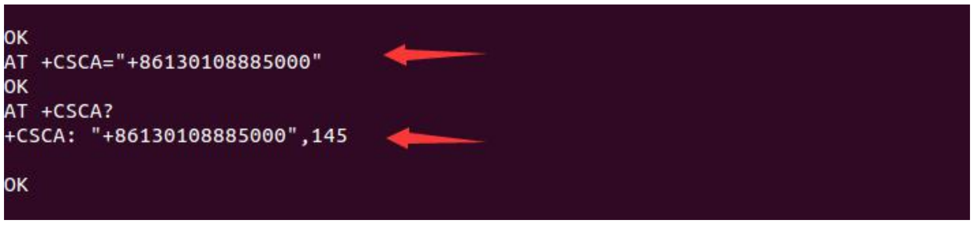
Input command of **AT +cfun=1** to open Modem full function



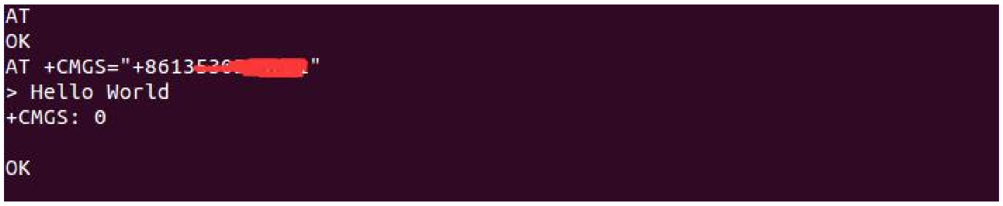
```
AT
OK
AT +CFUN=1
OK

+CIEV: service,  1
+CIEV: roam, 1

+CREG: 5
```



```
AT
OK
AT +CMGF=1
OK
```

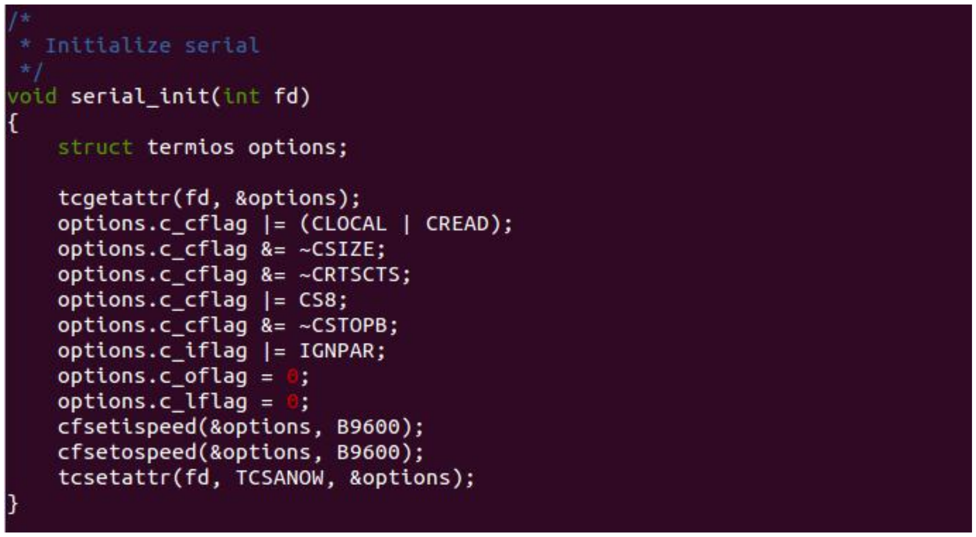


```
OK
AT +CSCA="+86130108885000"
OK
AT +CSCA?
+CSCA: "+86130108885000",145

OK
```



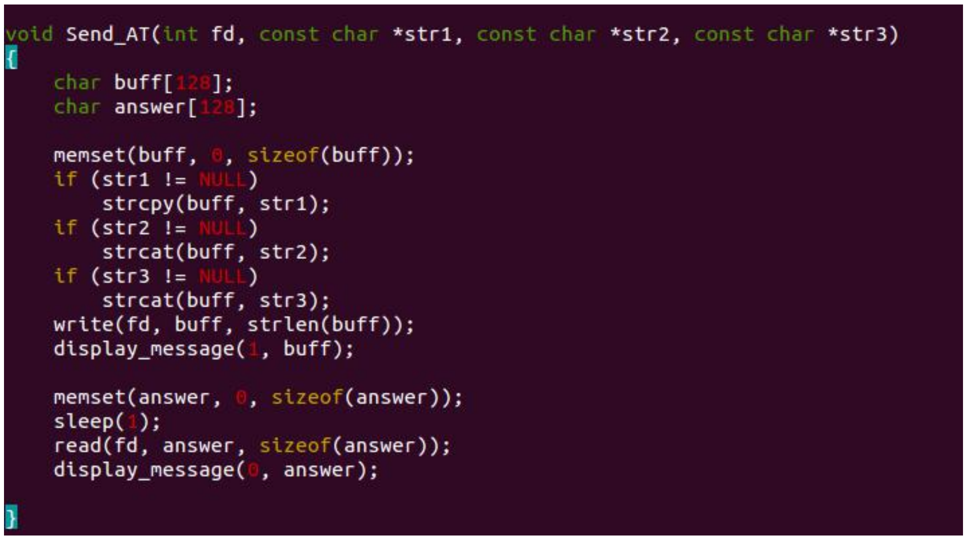
```
AT
OK
AT +CMGS="+86135[redacted]"
> Hello World
+CMGS: 0

OK
```

Initialize serial port:



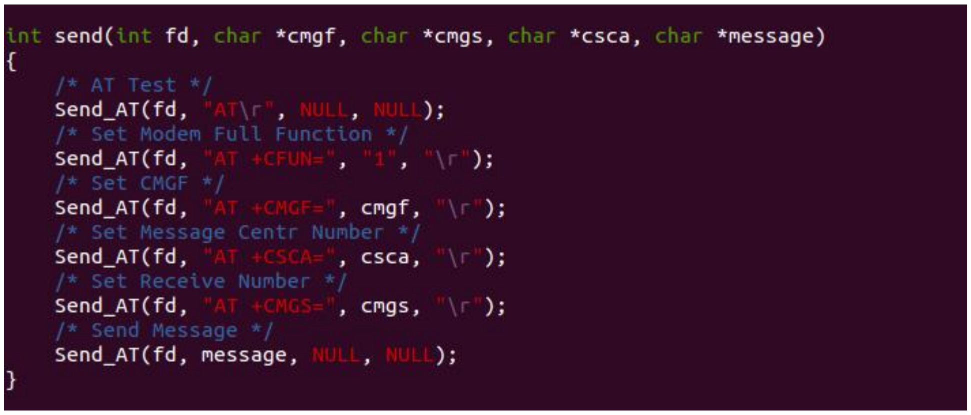
```
/*
 * Initialize serial
 */
void serial_init(int fd)
{
    struct termios options;

    tcgetattr(fd, &options);
    options.c_cflag |= (CLOCAL | CREAD);
    options.c_cflag &= ~CSIZE;
    options.c_cflag &= ~CRTSCTS;
    options.c_cflag |= CS8;
    options.c_cflag &= ~CSTOPB;
    options.c_iflag |= IGNPAR;
    options.c_oflag = 0;
    options.c_lflag = 0;
    cfsetispeed(&options, B9600);
    cfsetospeed(&options, B9600);
    tcsetattr(fd, TCSANOW, &options);
}
```

Send AT command function



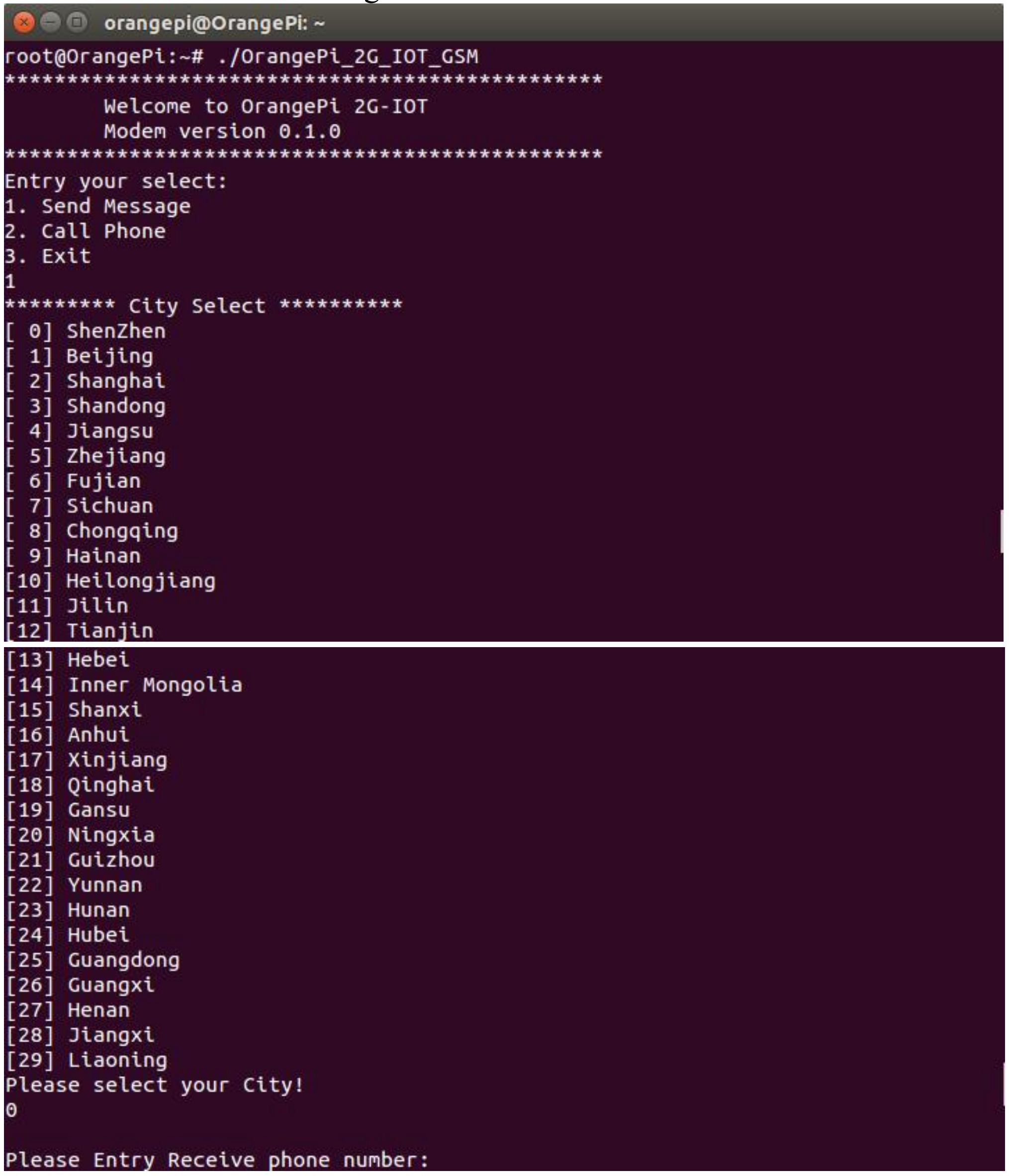
```
void Send_AT(int fd, const char *str1, const char *str2, const char *str3)
{
    char buff[128];
    char answer[128];

    memset(buff, 0, sizeof(buff));
    if (str1 != NULL)
        strcpy(buff, str1);
    if (str2 != NULL)
        strcat(buff, str2);
    if (str3 != NULL)
        strcat(buff, str3);
    write(fd, buff, strlen(buff));
    display_message(1, buff);

    memset(answer, 0, sizeof(answer));
    sleep(1);
    read(fd, answer, sizeof(answer));
    display_message(0, answer);

}
```

AT send message

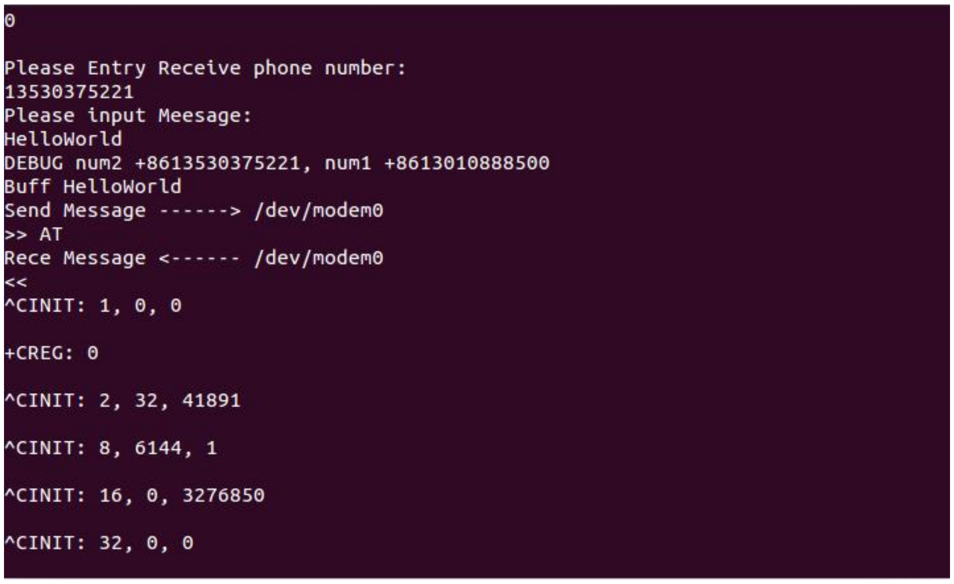


```
int send(int fd, char *cmgf, char *cmgs, char *csca, char *message)
{
    /* AT Test */
    Send_AT(fd, "AT\r", NULL, NULL);
    /* Set Modem Full Function */
    Send_AT(fd, "AT +CFUN=", "1", "\r");
    /* Set CMGF */
    Send_AT(fd, "AT +CMGF=", cmgf, "\r");
    /* Set Message Centr Number */
    Send_AT(fd, "AT +CSCA=", csca, "\r");
    /* Set Receive Number */
    Send_AT(fd, "AT +CMGS=", cmgs, "\r");
    /* Send Message */
    Send_AT(fd, message, NULL, NULL);
}
```

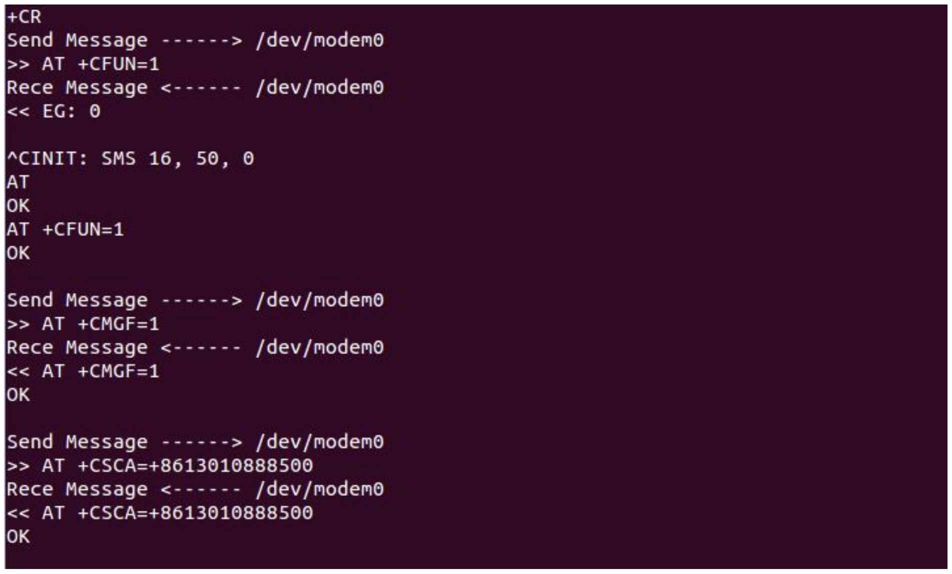
Detail steps you could refer to the following: "**OrangePi_2G_IOT_GSM_Demo.c**"
Here is the reflect of running C:



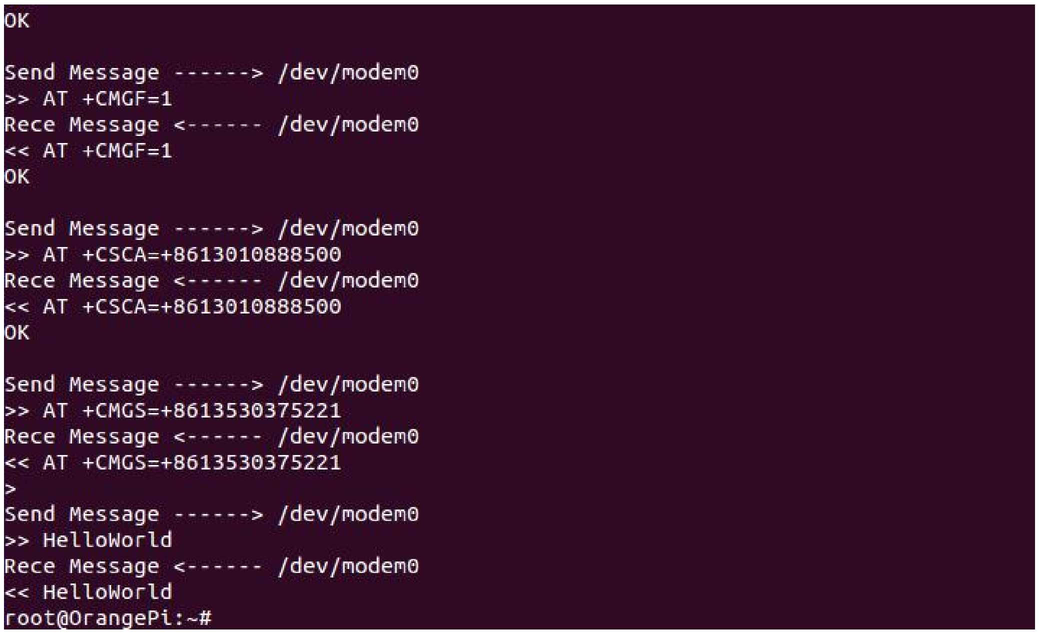
```
root@OrangePi:~# ./OrangePi_2G_IOT_GSM
***************************************************
        Welcome to OrangePi 2G-IOT
        Modem version 0.1.0
***************************************************
Entry your select:
1. Send Message
2. Call Phone
3. Exit
1
********* City Select **********
[ 0] ShenZhen
[ 1] Beijing
[ 2] Shanghai
[ 3] Shandong
[ 4] Jiangsu
[ 5] Zhejiang
[ 6] Fujian
[ 7] Sichuan
[ 8] Chongqing
[ 9] Hainan
[10] Heilongjiang
[11] Jilin
[12] Tianjin
[13] Hebei
[14] Inner Mongolia
[15] Shanxi
[16] Anhui
[17] Xinjiang
[18] Qinghai
[19] Gansu
[20] Ningxia
[21] Guizhou
[22] Yunnan
[23] Hunan
[24] Hubei
[25] Guangdong
[26] Guangxi
[27] Henan
[28] Jiangxi
[29] Liaoning
Please select your City!
0

Please Entry Receive phone number:
```



```
0

Please Entry Receive phone number:
13530375221
Please input Meesage:
HelloWorld
DEBUG num2 +8613530375221, num1 +8613010888500
Buff HelloWorld
Send Message ------> /dev/modem0
>> AT
Rece Message <------ /dev/modem0
<<
^CINIT: 1, 0, 0

+CREG: 0

^CINIT: 2, 32, 41891

^CINIT: 8, 6144, 1

^CINIT: 16, 0, 3276850

^CINIT: 32, 0, 0
```

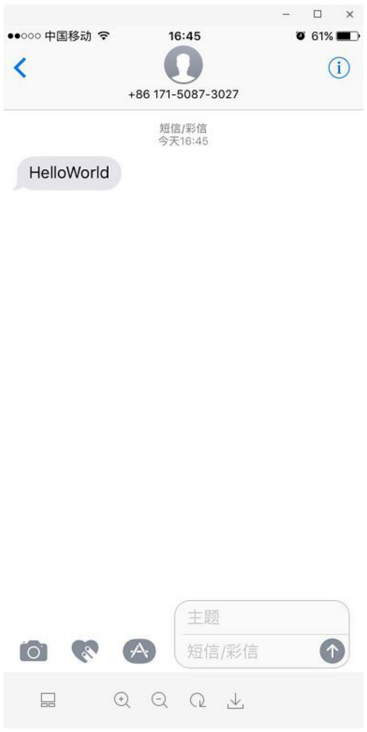


```
+CR
Send Message ------> /dev/modem0
>> AT +CFUN=1
Rece Message <------ /dev/modem0
<< EG: 0

^CINIT: SMS 16, 50, 0
AT
OK
AT +CFUN=1
OK

Send Message ------> /dev/modem0
>> AT +CMGF=1
Rece Message <------ /dev/modem0
<< AT +CMGF=1
OK

Send Message ------> /dev/modem0
>> AT +CSCA=+8613010888500
Rece Message <------ /dev/modem0
<< AT +CSCA=+8613010888500
OK
```



```
OK

Send Message ------> /dev/modem0
>> AT +CMGF=1
Rece Message <------ /dev/modem0
<< AT +CMGF=1
OK

Send Message ------> /dev/modem0
>> AT +CSCA=+8613010888500
Rece Message <------ /dev/modem0
<< AT +CSCA=+8613010888500
OK

Send Message ------> /dev/modem0
>> AT +CMGS=+8613530375221
Rece Message <------ /dev/modem0
<< AT +CMGS=+8613530375221
>
Send Message ------> /dev/modem0
>> HelloWorld
Rece Message <------ /dev/modem0
<< HelloWorld
root@OrangePi:~#
```

Finally you will receive the message from OrangePi 2G-IOT



- OrangePi_2G_IOT_GSM_Demo.c

```
/*
 * OrangePi 2G-IOT GSM Demo
 *  (C) Copyright 2017 OrangePi
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

```
#include <fcntl.h>
#include <string.h>
#include <termios.h>
#include <sys/types.h>
#include <sys/stat.h>

#define NR_CITY  30
#define MODEM_PATH  "/dev/modem0"
#define VERSION     "0.1.0"

struct Centry_number {
    char *city;
    char *number;
} City_Number[NR_CITY] = {
    { "ShenZhen",           "13010888500" },
    { "Beijing",            "13010112500" },
    { "Shanghai",           "13010314500" },
    { "Shandong",           "13010171500" },
    { "Jiangsu" ,           "13010341500" },
    { "Zhejiang",           "13010360500" },
    { "Fujian",             "13010380500" },
    { "Sichuan",            "13010811500" },
    { "Chongqing",          "13010831500" },
    { "Hainan" ,            "13010501500" },
    { "Heilongjiang",       "13010980500" },
    { "Jilin",              "13010911500" },
    { "Tianjin",            "13010130500" },
    { "Hebei",              "13010180500" },
    { "Inner Mongolia",     "13010950500" },
    { "Shanxi",             "13010701500" },
    { "Anhui",              "13010305500" },
    { "Xinjiang",           "13010969500" },
    { "Qinghai",            "13010776500" },
    { "Gansu",              "13010879500" },
    { "Ningxia",            "13010796500" },
    { "Guizhou",            "13010788500" },
    { "Yunnan",             "13010868500" },
    { "Hunan",              "13010731500" },
    { "Hubei",              "13010710500" },
    { "Guangdong",          "13010200500" },
    { "Guangxi",            "13010591500" },
    { "Henan",              "13010761500" },
    { "Jiangxi",            "13010720500" },
    { "Liaoning",           "13010240500"},
};

/*
 * Initialize serial
 */
void serial_init(int fd)
{
    struct termios options;

    tcgetattr(fd, &options);
    options.c_cflag |= (CLOCAL | CREAD);
    options.c_cflag &= ~CSIZE;
    options.c_cflag &= ~CRTSCTS;
    options.c_cflag |= CS8;
    options.c_cflag &= ~CSTOPB;
    options.c_iflag |= IGNPAR;
    options.c_oflag = 0;
    options.c_lflag = 0;
    cfsetispeed(&options, B9600);
    cfsetospeed(&options, B9600);
    tcsetattr(fd, TCSANOW, &options);
}

void display_message(int direction, const char *message)
{
    if (direction) {
        printf("Send Message ------> %s\n", MODEM_PATH);
        printf(">> %s\n", message);
    } else {
        printf("Rece Message <------ %s\n", MODEM_PATH);
        printf("<< %s\n", message);
    }
}

void Send_AT(int fd, const char *str1, const char *str2, const char *str3)
{
    char buff[128];
    char answer[128];

    memset(buff, 0, sizeof(buff));
    if (str1 != NULL)
        strcpy(buff, str1);
    if (str2 != NULL)
        strcat(buff, str2);
    if (str3 != NULL)
        strcat(buff, str3);
    write(fd, buff, strlen(buff));
    display_message(1, buff);

    memset(answer, 0, sizeof(answer));
    sleep(1);
    read(fd, answer, sizeof(answer));
    display_message(0, answer);

}

int send(int fd, char *cmgf, char *cmgs, char *csca, char *message)
{
    /* AT Test */
    Send_AT(fd, "AT\r", NULL, NULL);
    /* Set Modem Full Function */
    Send_AT(fd, "AT +CFUN=", "1", "\r");
    /* Set CMGF */
    Send_AT(fd, "AT +CMGF=", cmgf, "\r");
    /* Set Message Centr Number */
    Send_AT(fd, "AT +CSCA=", csca, "\r");
    /* Set Receive Number */
    Send_AT(fd, "AT +CMGS=", cmgs, "\r");
    /* Send Message */
    Send_AT(fd, message, NULL, NULL);
}

int Send_Message(int fd)
{
    char buff[128];
    char num1[64];
    char num2[64];
    int i;
    int choice;

    printf("********* City Select **********\n");
    for (i = 0; i < NR_CITY; i++)
```

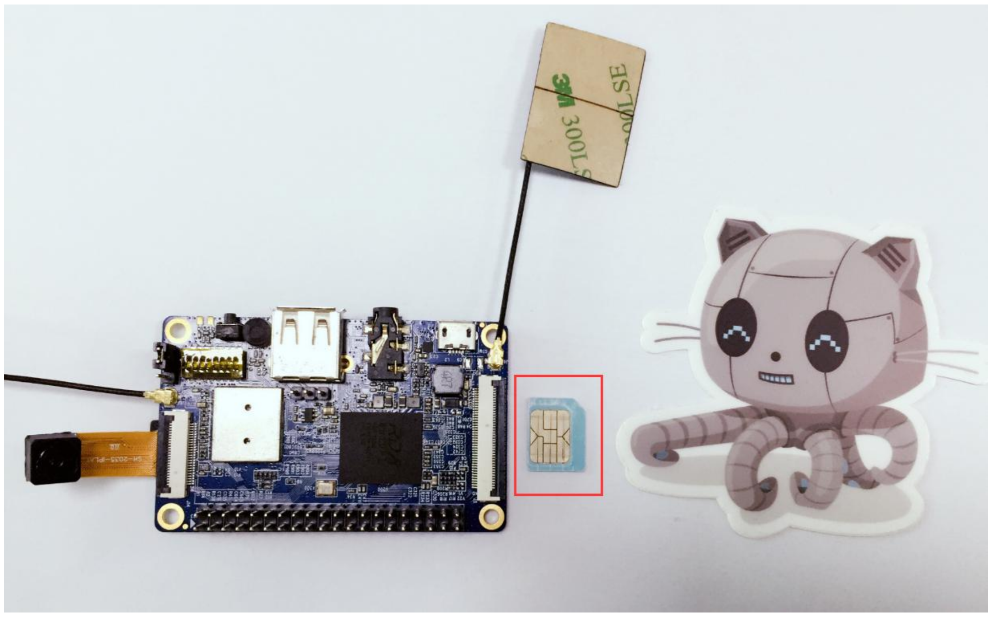
```
        printf("[%2d] %s\n", i, City_Number[i].city);
    printf("Please select your City!\n");
    scanf("%d", &choice);
    do {
        memset(num1, 0, sizeof(num1));
        printf("\nPlease Entry Receive phone number:\n");
        scanf("%s", num1);
    } while (strlen(num1) != 11);

    sleep(1);
    memset(buff, 0, sizeof(buff));
    printf("Please input Meesage:\n");
    scanf("%s", buff);

    /* Restruct buff */
    i = strlen(buff);
    buff[i] = 0x1A;
    buff[i+1] = '\r';
    buff[i+2] = '\0';

    memset(num2, 0, sizeof(num2));
    strcpy(num2, "+86");
    strcat(num2, num1);

    memset(num1, 0, sizeof(num1));
    strcpy(num1, "+86");
    strcat(num1, City_Number[choice].number);

    send(fd, "1", num2, num1, buff);
}

/*
 * Call Phone.
 */
void Call_Phone(int fd)
{
    char buff[128];
    char number[20];

    do {
        memset(number, 0, sizeof(number));
        printf("\nPlease input phone number:");
        scanf("%s", number);
    } while (strlen(number) != 11);

    memset(buff, 0, sizeof(buff));
    strcpy(buff, "+86");
    strcat(buff, number);
    strcat(buff, ";");

    /* AT Test */
    Send_AT(fd, "AT\r", NULL, NULL);
    /* Call */
    Send_AT(fd, "AT", " DT ", buff);
}

int main(int argc, char *argv[])
{
    int fd;
    char choice;

    fd = open(MODEM_PATH, O_RDWR | O_NOCTTY | O_NDELAY);
    if (fd < 0) {
        printf("Can't open %s\n", MODEM_PATH);
        return -1;
    }

    /* Initialize /dev/modem0 */
    serial_init(fd);

    printf("************************************************\n");
    printf("\tWelcome to OrangePi 2G-IOT\n");
    printf("\tModem version %s\n", VERSION);
    printf("************************************************\n");
    printf("Entry your select:\n");
    printf("1. Send Message\n");
    printf("2. Call Phone\n");
    printf("3. Exit\n");
    choice = getchar();

    switch (choice) {
    case '1':
            Send_Message(fd);
            break;
    case '2':
            Call_Phone(fd);
            break;
    default:
            break;

    }
    close(fd);

    return 0;
}
```

- **OrangePi 2G-IOT Linux 发行版打电话**

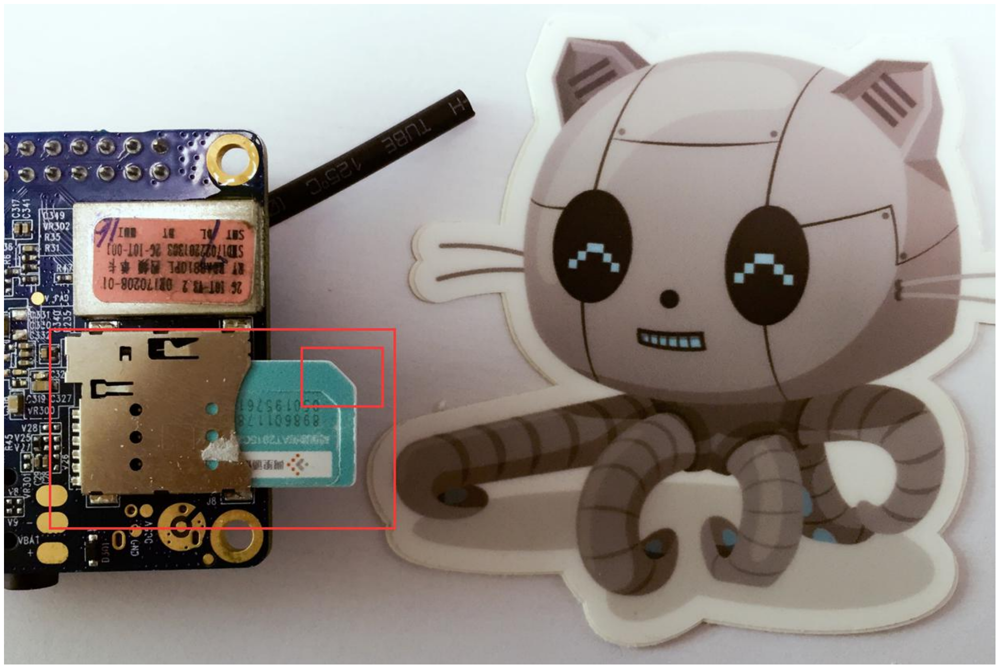
目前官方 OrangePi 2G-IOT Linux 发行版支持 Ubuntu， Debian 和 Raspbian， 这些发新版已经支持打电话功能，开发者请到官网(http://www.orangepi.cn/downloadresourcescn/) 下载最新的 Linux 发行版镜像。在使用打电话功能之前，请自行准备一张移动或联通 SIM 卡，并确保激活可以使用。

- SMS 前期准备
  准备一张中国联通或移动 SIM 卡，国外开发者可以根据频段选择对应的运用商 SIM 卡，OrangePi 2G-IOT 支持的频段如下：

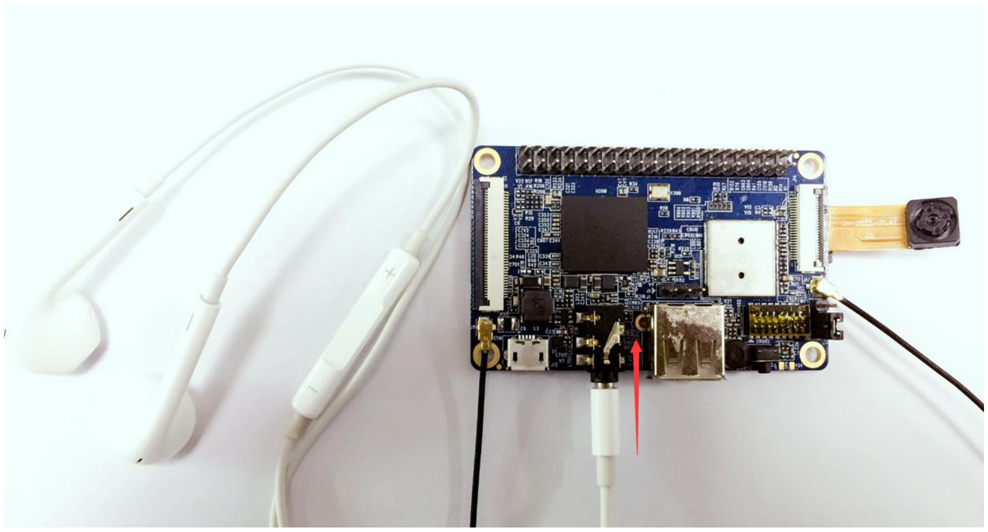
  准好 SIM 卡之后，将其安装到 OrangePi 2G-IOT 卡槽中，注意，OrangePi 2G-IOT SIM 卡槽支持 Nano 类型的卡。



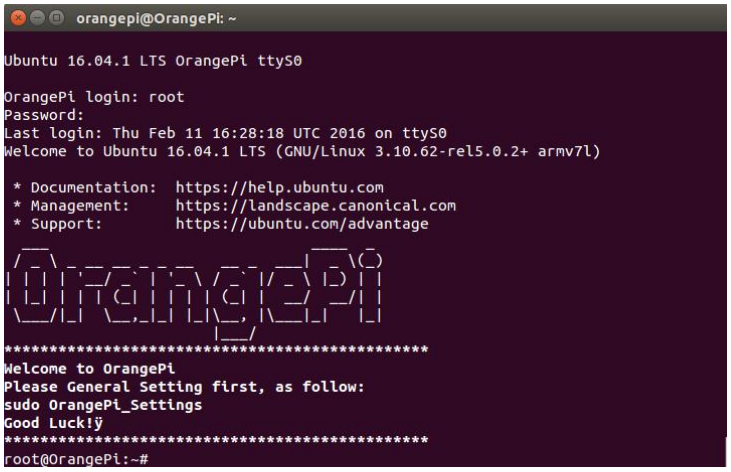
安装 SIM 卡，注意插入的方向，SIM 卡缺角方向在外侧



插入耳机



- 登录 Linux
  插好卡之后，上电启动 Linux 系统，用户可以使用多种方式连接系统，其中包括串口方式连接，ssh 方式连接等。具体连接方法，请参考官方 OrangePi 2G-IOT 的用户手册。如果使用串口方式连接，波特率请设置为 921600。



```
orangepi@OrangePi: ~

Ubuntu 16.04.1 LTS OrangePi ttyS0

OrangePi login: root
Password:
Last login: Thu Feb 11 16:28:18 UTC 2016 on ttyS0
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 3.10.62-rel5.0.2+ armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
  ___                              ____  _
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/
************************************************
Welcome to OrangePi
Please General Setting first, as follow:
sudo OrangePi_Settings
Good Luck!ÿ
************************************************
root@OrangePi:~#
```

- 多种方式打电话
  OrangePi 2G-IOT 的 Linux 发行版有多种方式可以使用通话功能，本文重点介绍串口方式和 C 程序方式。
  1. **串口方式**
  该方式需要在 OrangePi 2G-IOT 上安装串口工具 minicom，开发者可以在 OrangePi 2G-IOT 上进行安装，安装之前请确保 OrangePi 2G-IOT 已经连上 wifi。
  开发者使用如下命令进行 minicom 的安装：

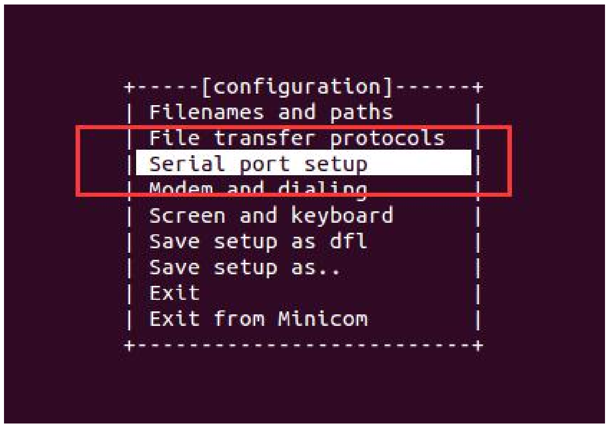
```
sudo apt-get install minicom
```

  OrangePi 中 AP 核与 Modem 之间通过串口进行通信，Linux 系统启动之后，会在 /dev/ 目录下生成 modem0 节点。
  开发者可以使用串口工具连接到 /dev/modem0 节点上。其中推荐使用 minicom 进行连接，开发者也可以参照使用其他串口工具。
  开发者在使用 minicom 或其他串口工具进行连接时，请参考一下步骤：
  1）在 OrangePi 2G-IOT Linux 系统上使用 minicom 连接 /dev/modem0, 用户可以使用

如下命令进行配置

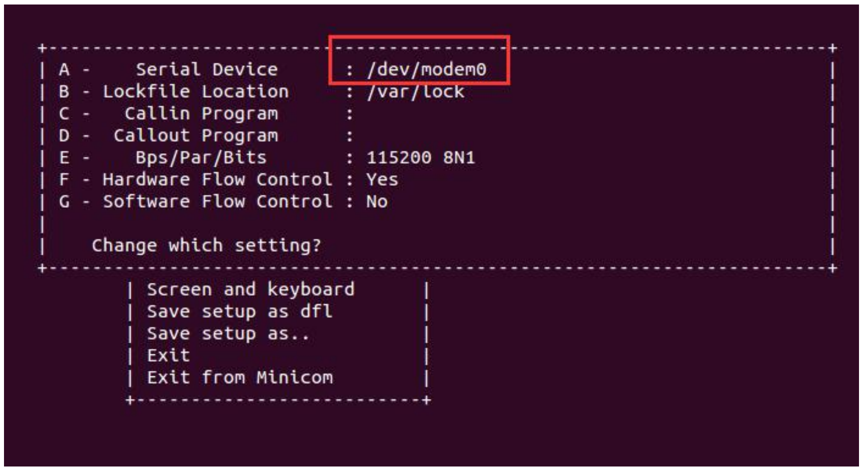
```
sudo minicmo -s
```

输入命令之后，进入配置界面，如下图，选择 **"Serial port setup"**



```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

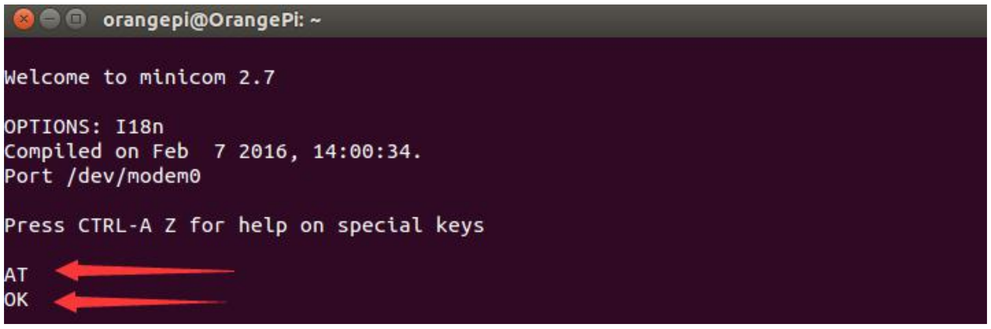
按下 “A” 修改 **Serial Device** 为 **/dev/modem0**



```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/modem0                                |
| B - Lockfile Location     : /var/lock                                  |
| C -   Callin Program      :                                            |
| D -  Callout Program      :                                            |
| E -    Bps/Par/Bits       : 115200 8N1                                 |
| F - Hardware Flow Control : Yes                                        |
| G - Software Flow Control : No                                         |
|                                                                        |
|    Change which setting?                                               |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

2) 使用 AT 指令集进行短信的发送

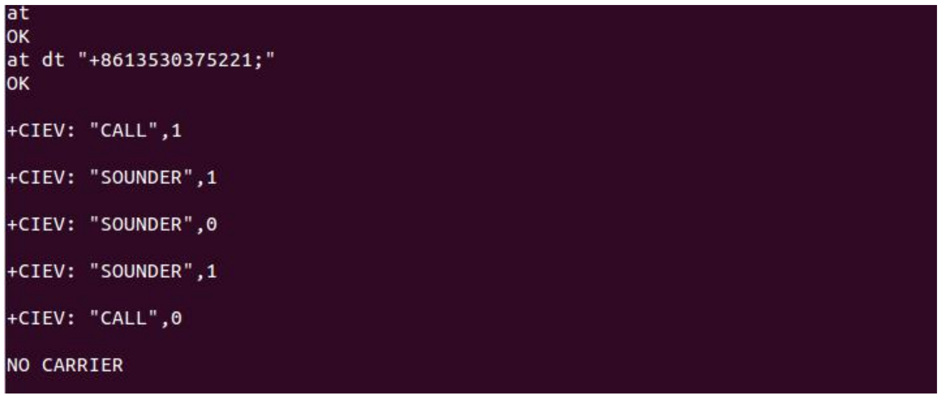
首先输入 **AT** 命令对 Modem 状态进行检查，如果 Modem 回复 OK，表示 Modem 连接正常。



```
orangepi@OrangePi: ~

Welcome to minicom 2.7

OPTIONS: I18n
Compiled on Feb  7 2016, 14:00:34.
Port /dev/modem0

Press CTRL-A Z for help on special keys

AT
OK
```

接着使用 **"AT DT"** 命令拨打电话，命令之后紧跟被呼叫的电话号码，并在电话号码前加上国家编号，中国区为 +86。

输入完电话号码之后，以 “; ” 结尾，例如 **AT DT "+86135xxxxxxx;"**



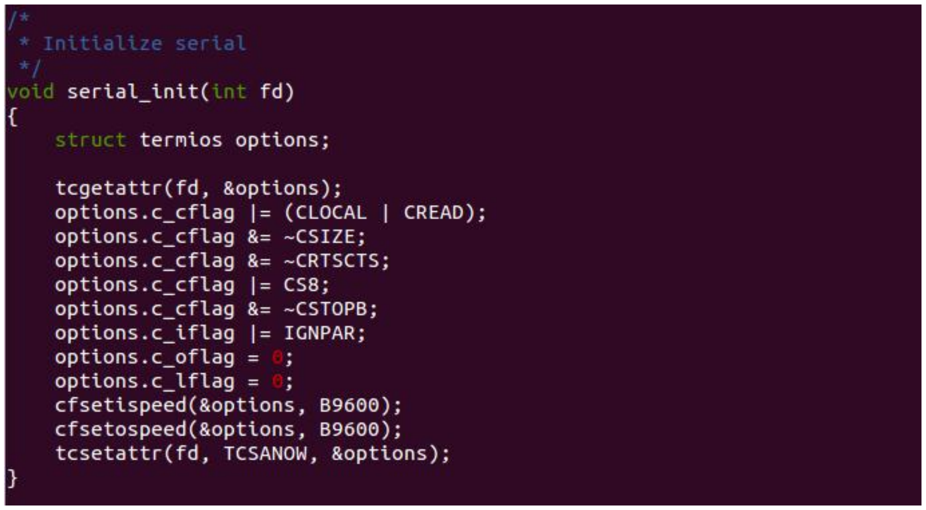
```
at
OK
at dt "+8613530375221;"
OK

+CIEV: "CALL",1

+CIEV: "SOUNDER",1

+CIEV: "SOUNDER",0

+CIEV: "SOUNDER",1

+CIEV: "CALL",0

NO CARRIER
```

以上为一次完整的通话过程。

2. C 语言方式

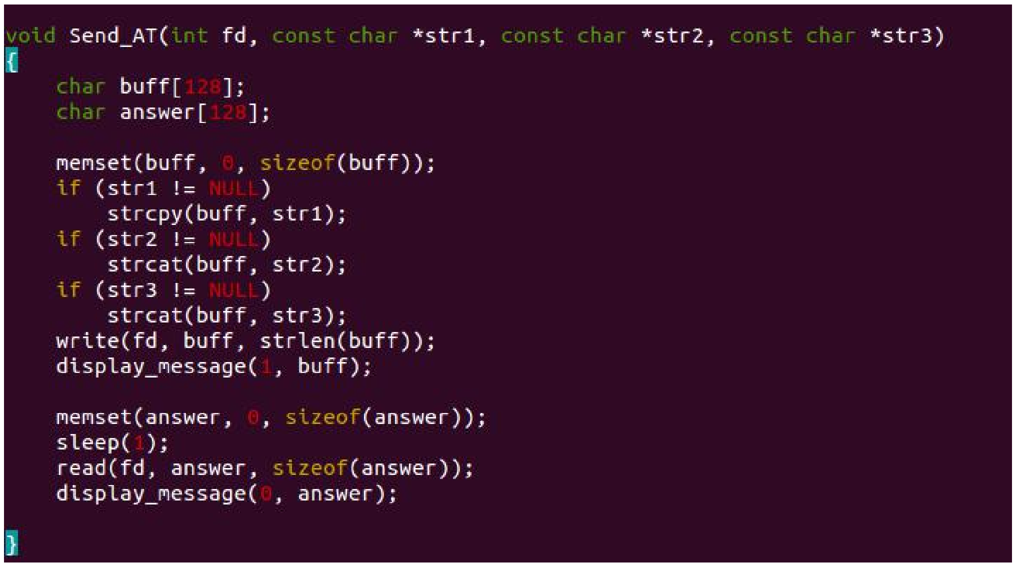
C 代码中发打电话的流程与 AT 发端的流程一样，也使用串口和 Modem 进行通信，然后交换 AT 命令。具体流程如下：

首先，初始化串口：



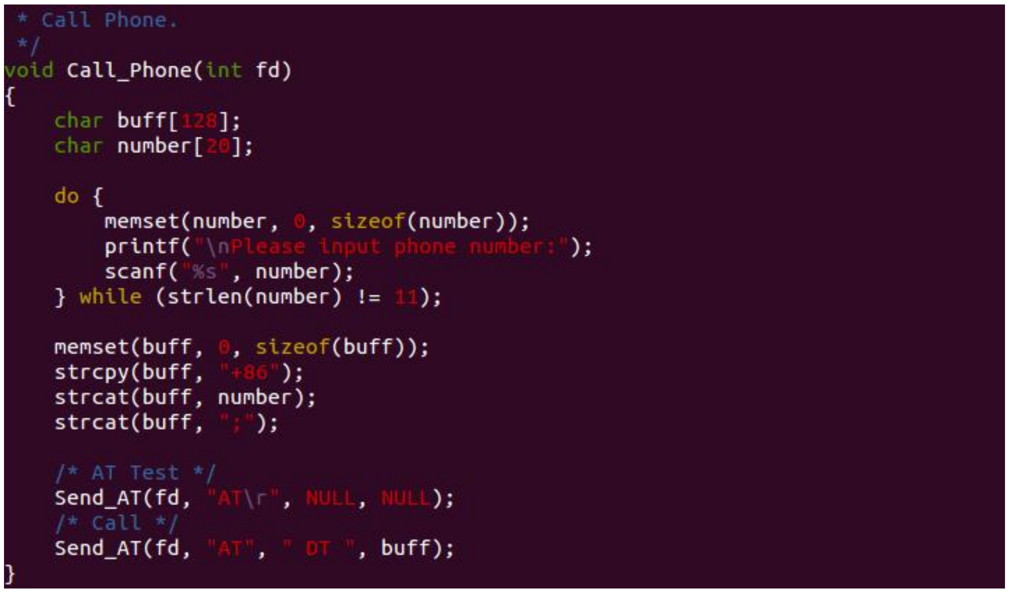
```
/*
 * Initialize serial
 */
void serial_init(int fd)
{
    struct termios options;

    tcgetattr(fd, &options);
    options.c_cflag |= (CLOCAL | CREAD);
    options.c_cflag &= ~CSIZE;
    options.c_cflag &= ~CRTSCTS;
    options.c_cflag |= CS8;
    options.c_cflag &= ~CSTOPB;
    options.c_iflag |= IGNPAR;
    options.c_oflag = 0;
    options.c_lflag = 0;
    cfsetispeed(&options, B9600);
    cfsetospeed(&options, B9600);
    tcsetattr(fd, TCSANOW, &options);
}
```

发送 AT 命令函数



```
void Send_AT(int fd, const char *str1, const char *str2, const char *str3)
{
    char buff[128];
    char answer[128];

    memset(buff, 0, sizeof(buff));
    if (str1 != NULL)
        strcpy(buff, str1);
    if (str2 != NULL)
        strcat(buff, str2);
    if (str3 != NULL)
        strcat(buff, str3);
    write(fd, buff, strlen(buff));
    display_message(1, buff);

    memset(answer, 0, sizeof(answer));
    sleep(1);
    read(fd, answer, sizeof(answer));
    display_message(0, answer);

}
```

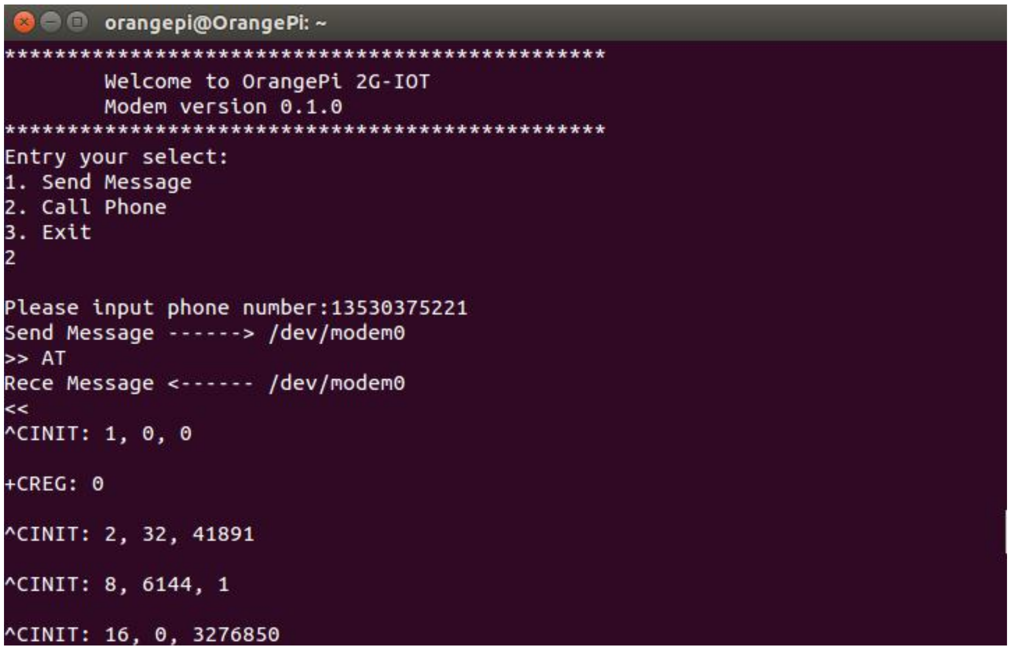
AT 发短信流程



```
 * Call Phone.
 */
void Call_Phone(int fd)
{
    char buff[128];
    char number[20];

    do {
        memset(number, 0, sizeof(number));
        printf("\nPlease input phone number:");
        scanf("%s", number);
    } while (strlen(number) != 11);

    memset(buff, 0, sizeof(buff));
    strcpy(buff, "+86");
    strcat(buff, number);
    strcat(buff, ";");

    /* AT Test */
    Send_AT(fd, "AT\r", NULL, NULL);
    /* Call */
    Send_AT(fd, "AT", " DT ", buff);
}
```

完整程序见附录 **“OrangePi_2G_IOT_GSM_Demo.c”**

C 代码运行效果如下图：



```
orangepi@OrangePi: ~
**************************************************
        Welcome to OrangePi 2G-IOT
        Modem version 0.1.0
**************************************************
Entry your select:
1. Send Message
2. Call Phone
3. Exit
2

Please input phone number:13530375221
Send Message ------> /dev/modem0
>> AT
Rece Message <------ /dev/modem0
<<
^CINIT: 1, 0, 0

+CREG: 0

^CINIT: 2, 32, 41891

^CINIT: 8, 6144, 1

^CINIT: 16, 0, 3276850
```

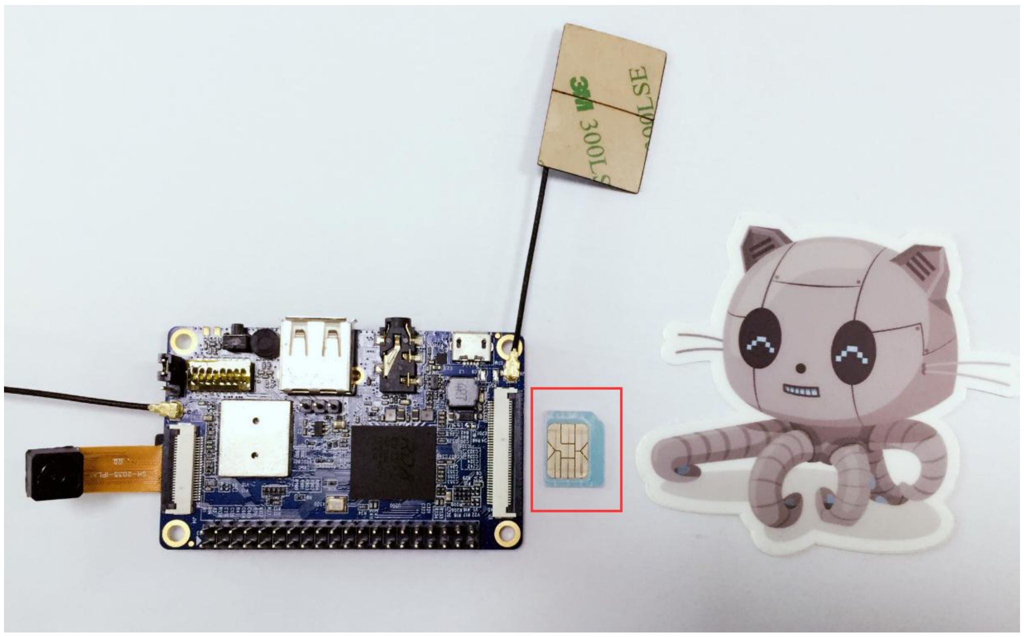
- OrangePi_2G_IOT_GSM_Demo.c
  Demo 程序请参考上一节。

- **OrangePi 2G-IOT Linux 发行版 GSM 上网**

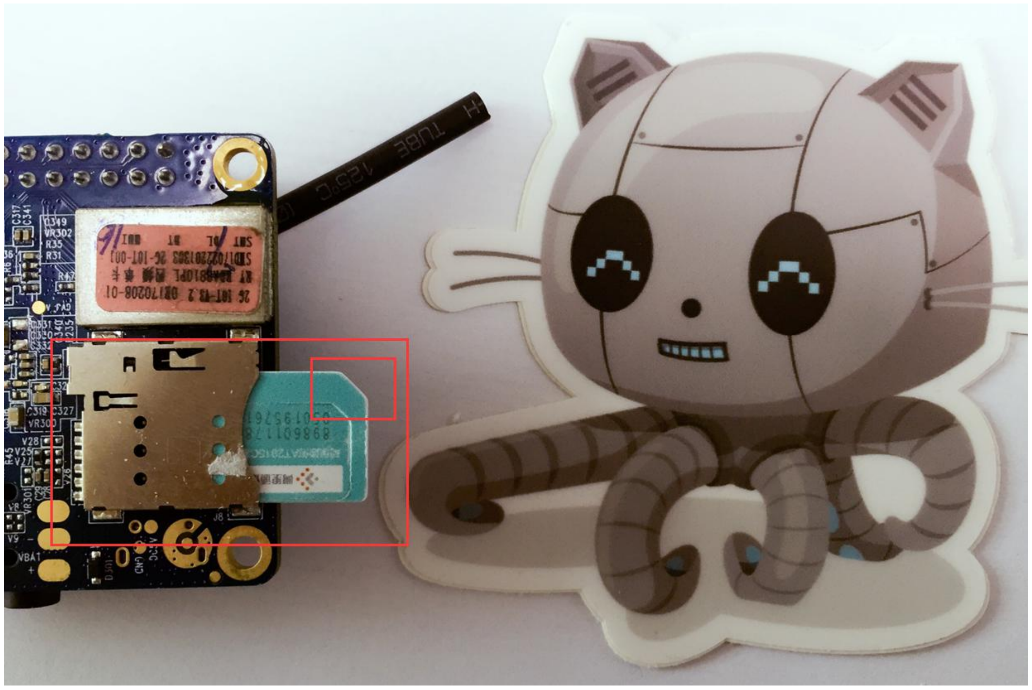
目前官方 OrangePi 2G-IOT Linux 发行版支持 Ubuntu， Debian 和 Raspbian， 这些发新版已经支持 GSM 上网功能，开发者请到官网(http://www.orangepi.cn/downloadresourcescn/) 下载最新的 Linux 发行版镜像。在使用 GSM 上网功能之前，请自行准备一张移动或联通 SIM 卡，并确保激活可以使用。

- SMS 前期准备
  准备一张中国联通或移动 SIM 卡，国外开发者可以根据频段选择对应的运用商 SIM 卡，OrangePi 2G-IOT 支持的频段如下：

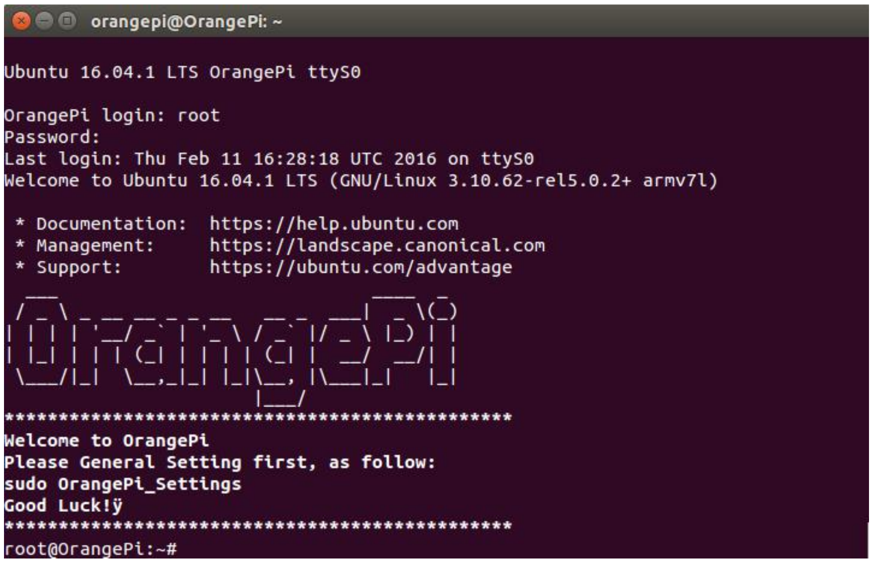
  准好 SIM 卡之后，将其安装到 OrangePi 2G-IOT 卡槽中，注意，OrangePi 2G-IOT SIM 卡槽支持 Nano 类型的卡。



安装 SIM 卡，注意插入的方向，SIM 卡缺角方向在外侧



- 登录 Linux
  插好卡之后，上电启动 Linux 系统，用户可以使用多种方式连接系统，其中包括串口方式连接，ssh 方式连接等。具体连接方法，请参考官方 OrangePi 2G-IOT 的用户手册。如果使用串口方式连接，波特率请设置为 921600。



```
orangepi@OrangePi: ~

Ubuntu 16.04.1 LTS OrangePi ttyS0

OrangePi login: root
Password:
Last login: Thu Feb 11 16:28:18 UTC 2016 on ttyS0
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 3.10.62-rel5.0.2+ armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
  ___                              ____  _
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/
**************************************************
Welcome to OrangePi
Please General Setting first, as follow:
sudo OrangePi_Settings
Good Luck!ÿ
**************************************************
root@OrangePi:~#
```

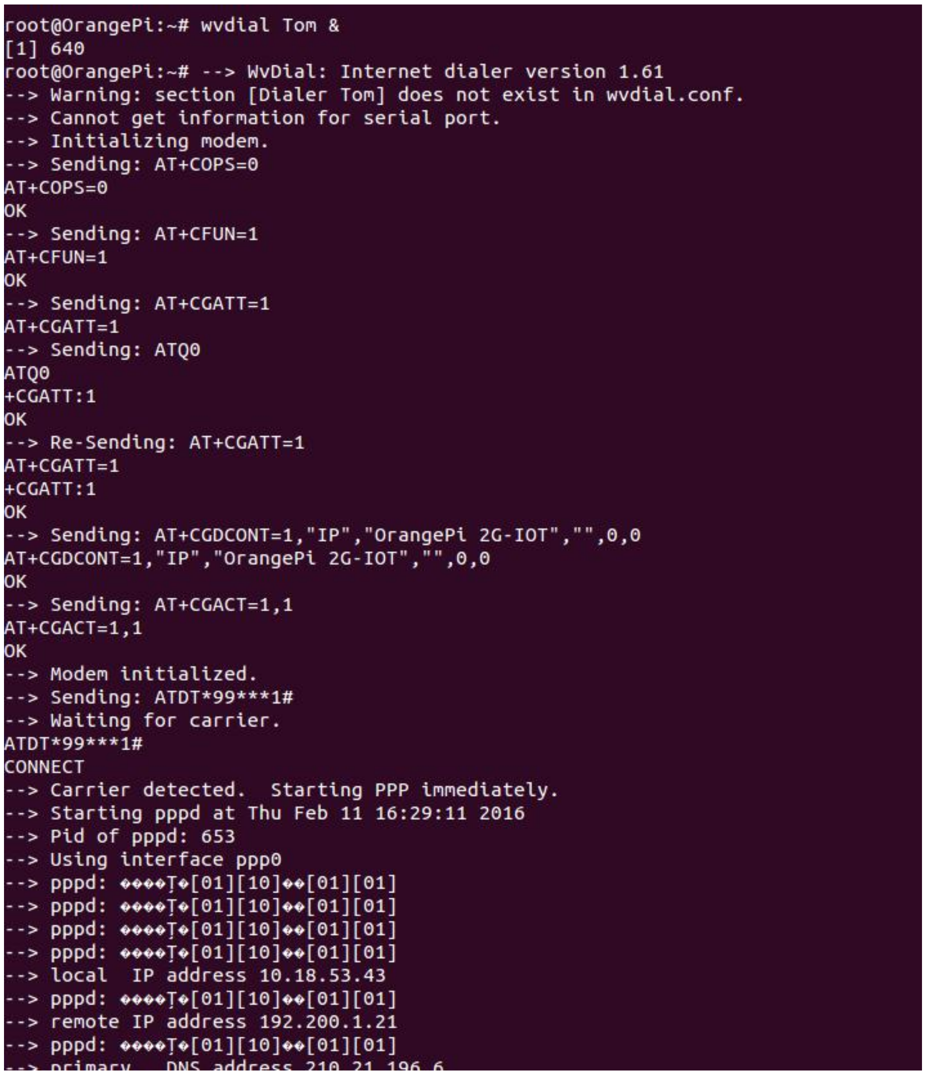
- 使用开源工具拨号上网
  OrangePi 2G-IOT 目前支持 ppp 方式上网，其上网基于 ppp 和 wvdial 两个开源工具。开发者首先在 OrangePi 2G-IOT 上安装这两个开源工具，可使用如下命令：
  sudo apt-get install ppp wvdial
- 配置 wvdial
  安装完毕后，对 wvdial 工具进行配置，修改 /etc/wvdial.conf 文件，如下

```
[Dialer defaults]
ISDN = 0
Modem Type = Analog Modem
Phone = *99***1#
Stupid Mode = 1
Dial Command = ATDT
Modem = /dev/modem0
Baud = 460800
Init1 = AT+COPS=0
Init2 = AT+CFUN=1
Init3 = AT+CGATT=1
Init4 = AT+CGDCONT=1,"IP","OrangePi_2G-IOT","",0,0
Init5 = AT+CGACT=1,1
Username = " "
Password = " "
```

- 配置 ppp 工具
  安装完 ppp 之后，对 ppp 进行配置，修改 /etc/ppp/peers/wvdial 文件，如下：

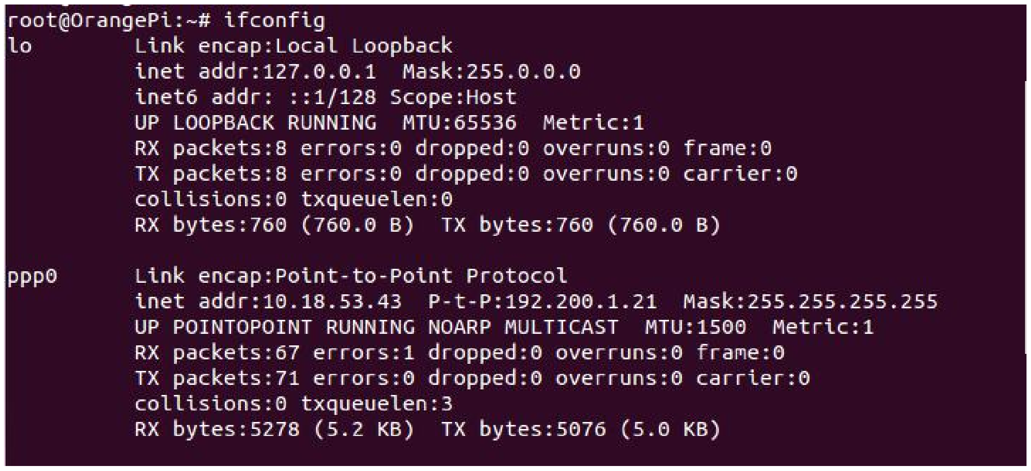
```
 noauth
 name wvdial
 defaultroute
 replacedefaultroute
```

- 拨号上网
  配置好 ppp 和 wvdial 之后，使用命令进行拨号上网，开发者可以使用如下命令：
  wvdial Tom &



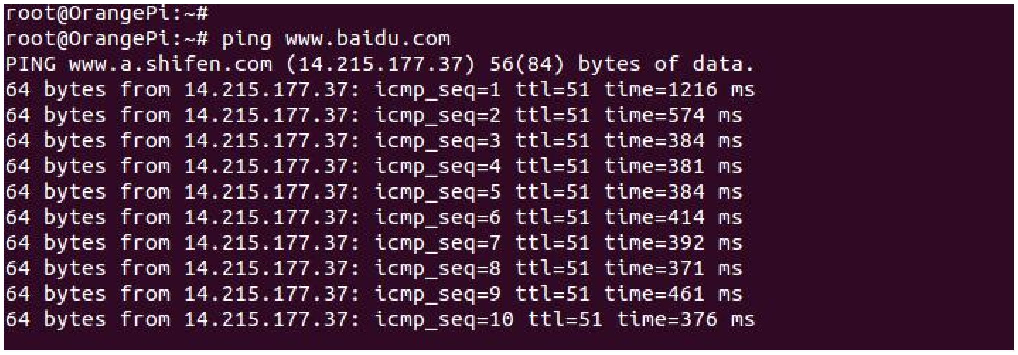
```
root@OrangePi:~# wvdial Tom &
[1] 640
root@OrangePi:~# --> WvDial: Internet dialer version 1.61
--> Warning: section [Dialer Tom] does not exist in wvdial.conf.
--> Cannot get information for serial port.
--> Initializing modem.
--> Sending: AT+COPS=0
AT+COPS=0
OK
--> Sending: AT+CFUN=1
AT+CFUN=1
OK
--> Sending: AT+CGATT=1
AT+CGATT=1
--> Sending: ATQ0
ATQ0
+CGATT:1
OK
--> Re-Sending: AT+CGATT=1
AT+CGATT=1
+CGATT:1
OK
--> Sending: AT+CGDCONT=1,"IP","OrangePi 2G-IOT","",0,0
AT+CGDCONT=1,"IP","OrangePi 2G-IOT","",0,0
OK
--> Sending: AT+CGACT=1,1
AT+CGACT=1,1
OK
--> Modem initialized.
--> Sending: ATDT*99***1#
--> Waiting for carrier.
ATDT*99***1#
CONNECT
--> Carrier detected.  Starting PPP immediately.
--> Starting pppd at Thu Feb 11 16:29:11 2016
--> Pid of pppd: 653
--> Using interface ppp0
--> pppd: ����T�[01][10]��[01][01]
--> pppd: ����T�[01][10]��[01][01]
--> pppd: ����T�[01][10]��[01][01]
--> pppd: ����T�[01][10]��[01][01]
--> local  IP address 10.18.53.43
--> pppd: ����T�[01][10]��[01][01]
--> remote IP address 192.200.1.21
--> pppd: ����T�[01][10]��[01][01]
--> primary   DNS address 210.21.196.6
```

使用 ifconfig 查看网络信息



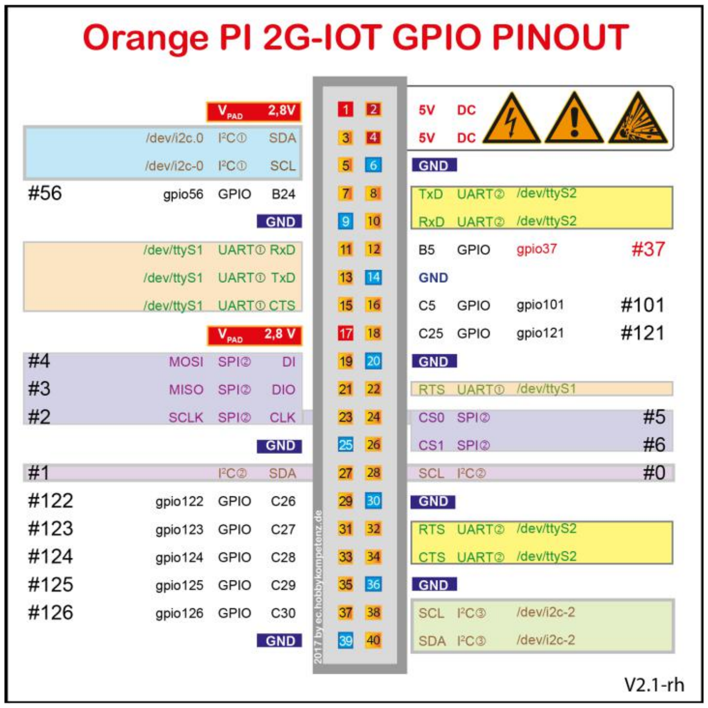
```
root@OrangePi:~# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:760 (760.0 B)  TX bytes:760 (760.0 B)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.18.53.43  P-t-P:192.200.1.21  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:67 errors:1 dropped:0 overruns:0 frame:0
          TX packets:71 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:5278 (5.2 KB)  TX bytes:5076 (5.0 KB)
```

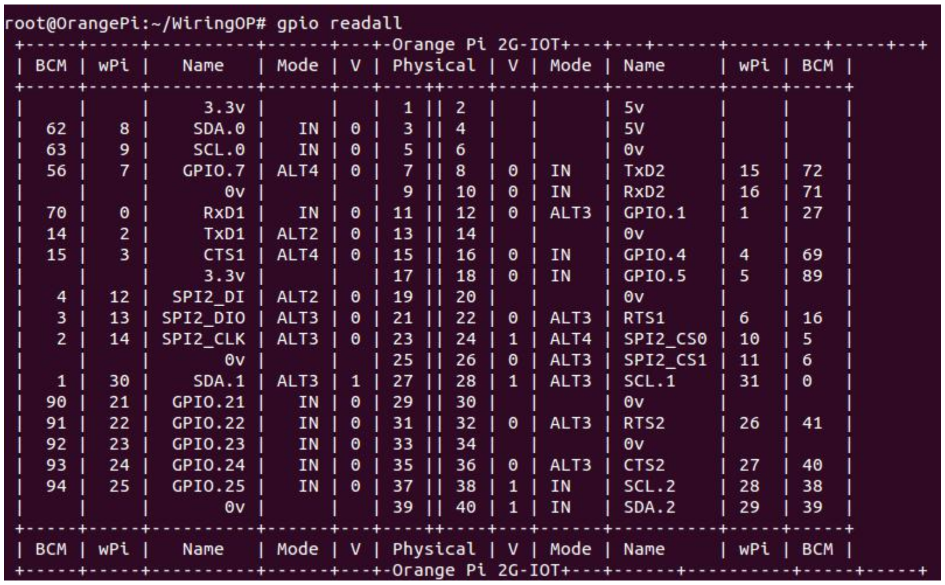
使用 ping 命令测试连通状态



```
root@OrangePi:~#
root@OrangePi:~# ping www.baidu.com
PING www.a.shifen.com (14.215.177.37) 56(84) bytes of data.
64 bytes from 14.215.177.37: icmp_seq=1 ttl=51 time=1216 ms
64 bytes from 14.215.177.37: icmp_seq=2 ttl=51 time=574 ms
64 bytes from 14.215.177.37: icmp_seq=3 ttl=51 time=384 ms
64 bytes from 14.215.177.37: icmp_seq=4 ttl=51 time=381 ms
64 bytes from 14.215.177.37: icmp_seq=5 ttl=51 time=384 ms
64 bytes from 14.215.177.37: icmp_seq=6 ttl=51 time=414 ms
64 bytes from 14.215.177.37: icmp_seq=7 ttl=51 time=392 ms
64 bytes from 14.215.177.37: icmp_seq=8 ttl=51 time=371 ms
64 bytes from 14.215.177.37: icmp_seq=9 ttl=51 time=461 ms
64 bytes from 14.215.177.37: icmp_seq=10 ttl=51 time=376 ms
```

The open source project WiringPi is working on OrangePi 2G-IOT. This section will introduce how to utilize WiringPi on OrangePi 2G-IOT. The following is definition of 40pin.


Orange PI 2G-IOT GPIO PINOUT
VPAD 2,8V 1 2 5V DC
/dev/i2c.0 I²C① SDA 3 4 5V DC
/dev/i2c-0 I²C① SCL 5 6 GND
#56 gpio56 GPIO B24 7 8 TxD UART② /dev/ttyS2
GND 9 10 RxD UART② /dev/ttyS2
/dev/ttyS1 UART① RxD 11 12 B5 GPIO gpio37 #37
/dev/ttyS1 UART① TxD 13 14 GND
/dev/ttyS1 UART① CTS 15 16 C5 GPIO gpio101 #101
VPAD 2,8 V 17 18 C25 GPIO gpio121 #121
#4 MOSI SPI② DI 19 20 GND
#3 MISO SPI② DIO 21 22 RTS UART① /dev/ttyS1
#2 SCLK SPI② CLK 23 24 CS0 SPI② #5
GND 25 26 CS1 SPI② #6
#1 I²C② SDA 27 28 SCL I²C② #0
#122 gpio122 GPIO C26 29 30 GND
#123 gpio123 GPIO C27 31 32 RTS UART② /dev/ttyS2
#124 gpio124 GPIO C28 33 34 CTS UART② /dev/ttyS2
#125 gpio125 GPIO C29 35 36 GND
#126 gpio126 GPIO C30 37 38 SCL I²C③ /dev/i2c-2
GND 39 40 SDA I²C③ /dev/i2c-2
2017 by ec.hobbykompetenz.de
V2.1-rh



```
root@OrangePi:~/WiringOP# gpio readall
 +-----+-----+----------+------+---+-Orange Pi 2G-IOT+---+---+------+---------+-----+--+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 |     |     |     3.3v |      |   |  1 || 2  |   |      | 5v       |     |     |
 |  62 |   8 |    SDA.0 |   IN | 0 |  3 || 4  |   |      | 5V       |     |     |
 |  63 |   9 |    SCL.0 |   IN | 0 |  5 || 6  |   |      | 0v       |     |     |
 |  56 |   7 |   GPIO.7 | ALT4 | 0 |  7 || 8  | 0 | IN   | TxD2     | 15  | 72  |
 |     |     |       0v |      |   |  9 || 10 | 0 | IN   | RxD2     | 16  | 71  |
 |  70 |   0 |     RxD1 |   IN | 0 | 11 || 12 | 0 | ALT3 | GPIO.1   | 1   | 27  |
 |  14 |   2 |     TxD1 | ALT2 | 0 | 13 || 14 |   |      | 0v       |     |     |
 |  15 |   3 |     CTS1 | ALT4 | 0 | 15 || 16 | 0 | IN   | GPIO.4   | 4   | 69  |
 |     |     |     3.3v |      |   | 17 || 18 | 0 | IN   | GPIO.5   | 5   | 89  |
 |   4 |  12 |  SPI2_DI | ALT2 | 0 | 19 || 20 |   |      | 0v       |     |     |
 |   3 |  13 | SPI2_DIO | ALT3 | 0 | 21 || 22 | 0 | ALT3 | RTS1     | 6   | 16  |
 |   2 |  14 | SPI2_CLK | ALT3 | 0 | 23 || 24 | 1 | ALT4 | SPI2_CS0 | 10  | 5   |
 |     |     |       0v |      |   | 25 || 26 | 0 | ALT3 | SPI2_CS1 | 11  | 6   |
 |   1 |  30 |    SDA.1 | ALT3 | 1 | 27 || 28 | 1 | ALT3 | SCL.1    | 31  | 0   |
 |  90 |  21 |  GPIO.21 |   IN | 0 | 29 || 30 |   |      | 0v       |     |     |
 |  91 |  22 |  GPIO.22 |   IN | 0 | 31 || 32 | 0 | ALT3 | RTS2     | 26  | 41  |
 |  92 |  23 |  GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v       |     |     |
 |  93 |  24 |  GPIO.24 |   IN | 0 | 35 || 36 | 0 | ALT3 | CTS2     | 27  | 40  |
 |  94 |  25 |  GPIO.25 |   IN | 0 | 37 || 38 | 1 | IN   | SCL.2    | 28  | 38  |
 |     |     |       0v |      |   | 39 || 40 | 1 | IN   | SDA.2    | 29  | 39  |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+-Orange Pi 2G-IOT+---+------+----------+-----+-----+
```

Usage:

1) Download WiringPi image of OrangePi 2G-IOT

```
env GIT_SSL_NO_VERIFY=true git clone https://github.com/OrangePiLibra/WiringPi.git
```

2) Compile and install WiringPi

```
cd WiringPi
sudo ./build OrangePi_2G-IOT
sudo ./build install
```

3) Utilze WiringPi

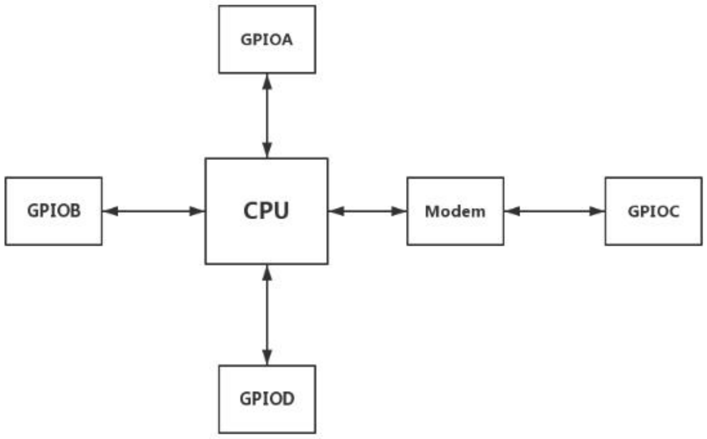
```
cd WiringPi/example/OrangePi
make OrangePi
./OrangePi
```

4) Details GPIO infromation

```
cd  WiringPi/example/OrangePi
cat README.md
```

Reality Test

OrangePi 2G-IOT contains GPIOA, GPIOB, GPIOC and GPIOD. Each of group has 32 gpio. The type of GPIO is "Input", "Output" and "specify function" such as "I2C", "I2S" and so on. On board, OrangePi 2G-IOT exports 40 pins as different function. User can utilize these, GPIO on different application scenarios. For example, User can configure the type of GPIO as "Input", and get current voltage from program. Another hand, User can configure GPIO as specify function, such as "Uart", "I2C" and "SPI".



Please note that: On OrangePi 2G-IOT, GPIOA, GPIOB and GPIOD tract as general GPIO, but GPIOC as non-general GPIO. Because of some hardware design. The host of GPIOC is modem not CPU. So, If CPU wanna control GPIOC, it must send message to Modem, and Moden get message and control GPIOC, it's not good news. In other words. on version 0.1, GPIOC only support "OUTPUT" mode.

The size of GPIOx group is 32, so we can get gpio map:

```
GPIOA: 0   -  31
GPIOB: 32  -  63
GPIOC: 64  -  95
GPIOD: 96  -  127
```

Each gpio have a unique ID, and the way of caculate as follow:

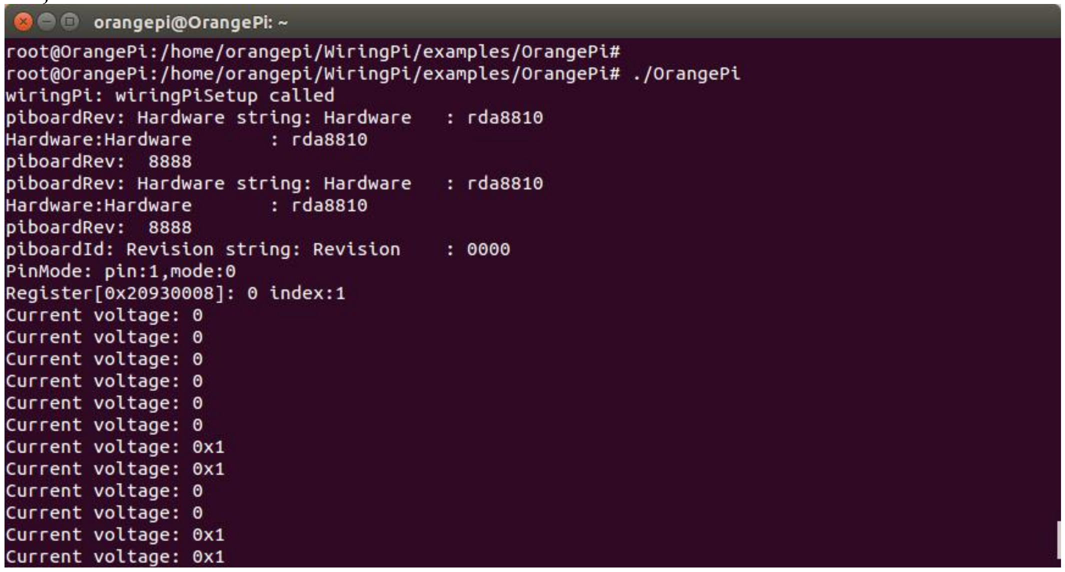
```
GPIO_A_x:   ID = 0    + x
GPIO_B_x:   ID = 32  + x
GPIO_C_x:   ID = 64  + x
GPIO_D_x:   ID = 96  + x
```

1) Set gpio as Input mode

As General GPIO, we only offer the number of GPIOx. then, use library of wiringPi, User can easily to control GPIO. GPIOC can not support "Input" mode.

```
#include <wiringPi.h>
/* Defind unique ID */
#define PA1          1
#define PB5          37
#define PD2          98
int main(void)
 {
```

```
unsigned int vol;

/* Setup wiringPi */
 wiringPiSetup();

 /* Set Input Mode */
pinMode(PA1, INPUT);
pinMode(PB5, INPUT);
pinMode(PD2, INPUT);

/* Get value from GPIO */
vol = digitalRead(PA1);
vol = digitalRead(PB5);
vol = digitalRead(PD2);

return 0;
}
```



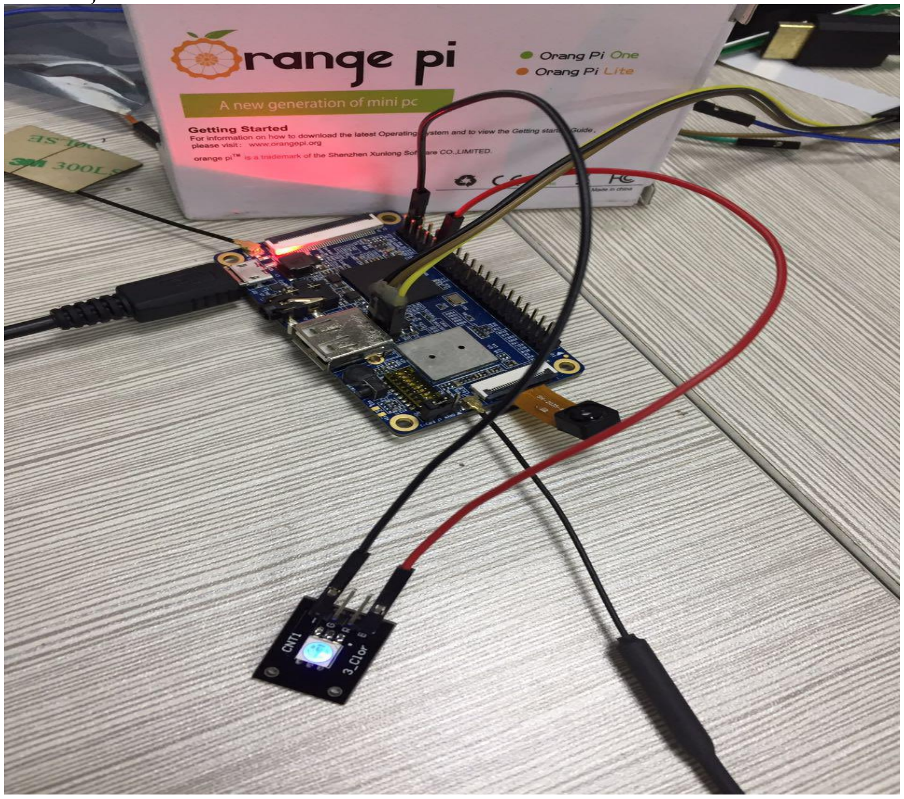
```
orangepi@OrangePi: ~
root@OrangePi:/home/orangepi/WiringPi/examples/OrangePi#
root@OrangePi:/home/orangepi/WiringPi/examples/OrangePi# ./OrangePi
wiringPi: wiringPiSetup called
piboardRev: Hardware string: Hardware   : rda8810
Hardware:Hardware       : rda8810
piboardRev:  8888
piboardRev: Hardware string: Hardware   : rda8810
Hardware:Hardware       : rda8810
piboardRev:  8888
piboardId: Revision string: Revision    : 0000
PinMode: pin:1,mode:0
Register[0x20930008]: 0 index:1
Current voltage: 0
Current voltage: 0
Current voltage: 0
Current voltage: 0
Current voltage: 0
Current voltage: 0
Current voltage: 0x1
Current voltage: 0x1
Current voltage: 0
Current voltage: 0
Current voltage: 0x1
Current voltage: 0x1
```

2) Set gpio as "OUTPUT" mode

All GPIO support "OUTPUT" mode. The demo code as follow:

```
#include <wiringPi.h>
/* Defind unique ID */
#define PA1          1
#define PB5          37
#define PC27           91
#defien PD2          98
int main(void)
{
/* Setup wiringPi */
 iringPiSetup();
/* Set GPIO Mode */
pinMode(PA1,  OUTPUT);
pinMode(PB5,  OUTPUT);
```

```
        pinMode(PC27, OUTPUT);
         pinMode(PD2,  OUTPUT);

        digitalWrite(PA1,  HIGH);
        digitalWrite(PB5,  LOW);
        digitalWrite(PC27, HIGH);
        digitalWrite(PD2,  LOW);

        return 0;
        }
```



11. Use GSM to Connect Network

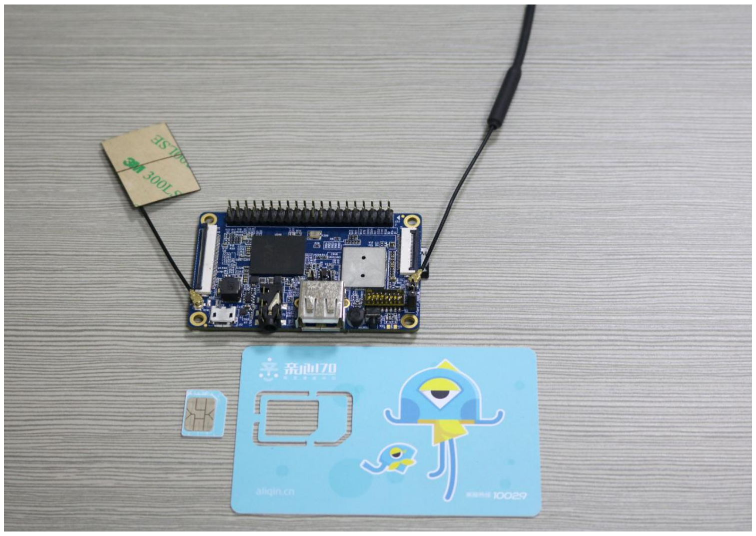
Orange Pi 2G-IOT Linux Dist has debugged to connect network via GSM. You could first download Linux Dist from official website, Debian, Ubuntu and Raspbian could support GSM function.

First you need to prepare an activated Micro SIM card, the Orange PI 2G-IOT could support the following frequency:

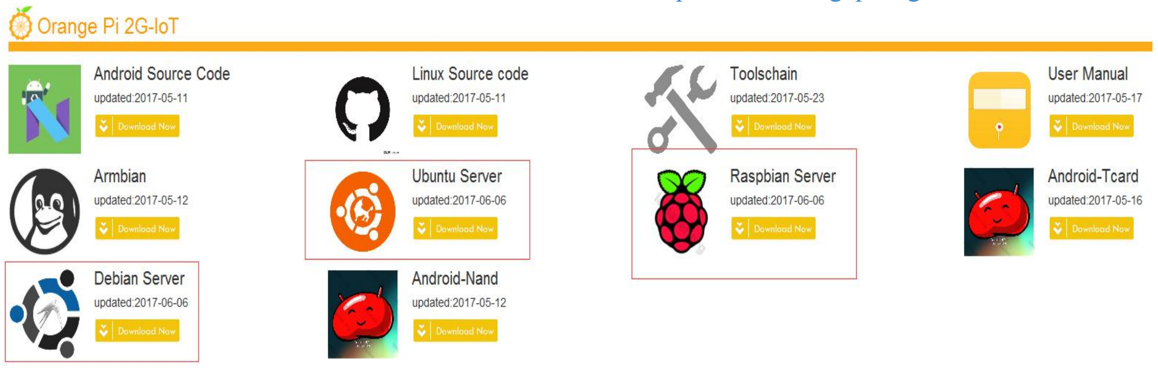
Frequency Range： 850,900,1800,1900

Standard: GSM800

The phone card we used is launched by China Unicom and Alibaba Group.



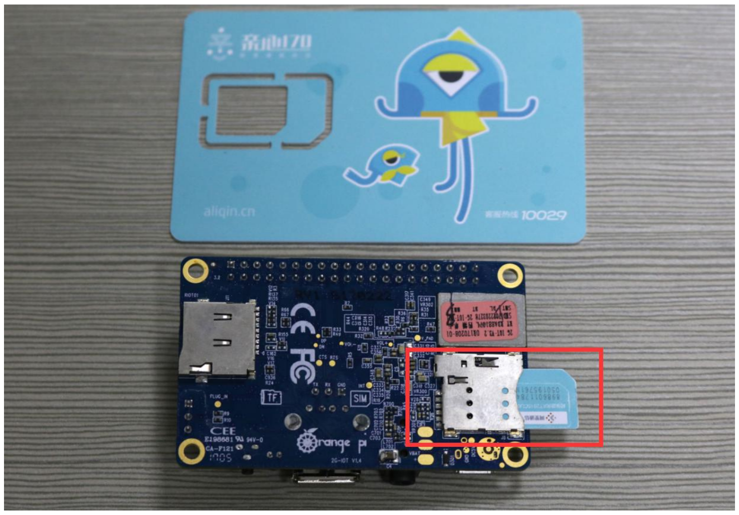
Download Linux Dist from official website: http://www.orangepi.org/downloadresources/

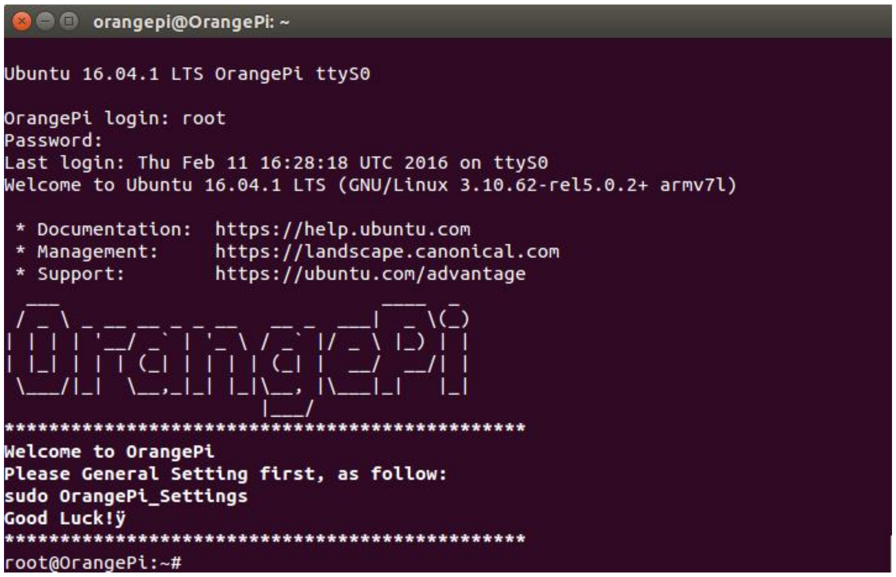


Orange Pi 2G-IoT

Android Source Code updated:2017-05-11 Download Now	Linux Source code updated:2017-05-11 Download Now	Toolschain updated:2017-05-23 Download Now	User Manual updated:2017-05-17 Download Now
Armbian updated:2017-05-12 Download Now	Ubuntu Server updated:2017-06-06 Download Now	Raspbian Server updated:2017-06-06 Download Now	Android-Tcard updated:2017-05-16 Download Now
Debian Server updated:2017-06-06 Download Now	Android-Nand updated:2017-05-12 Download Now		

Download and write image into SD card, then insert SD card and SIM card into the board.
You could login the system via serial port and SSH. If via serial port, please remember to set the baud rate into 921600.



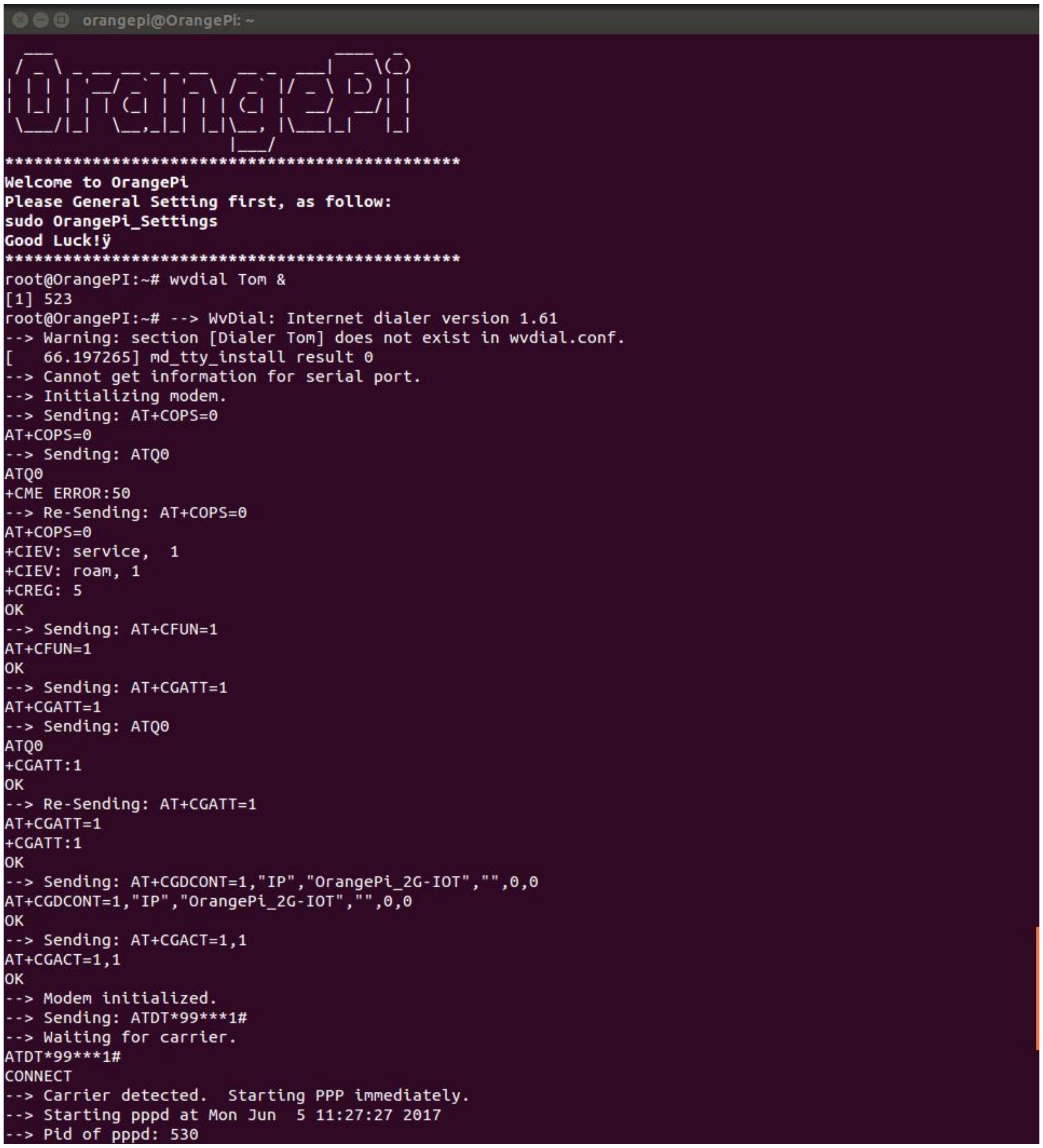


```
Ubuntu 16.04.1 LTS OrangePi ttyS0

OrangePi login: root
Password:
Last login: Thu Feb 11 16:28:18 UTC 2016 on ttyS0
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 3.10.62-rel5.0.2+ armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
  ___                              ____  _
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/
***********************************************
Welcome to OrangePi
Please General Setting first, as follow:
sudo OrangePi_Settings
Good Luck!ÿ
***********************************************
root@OrangePi:~#
```

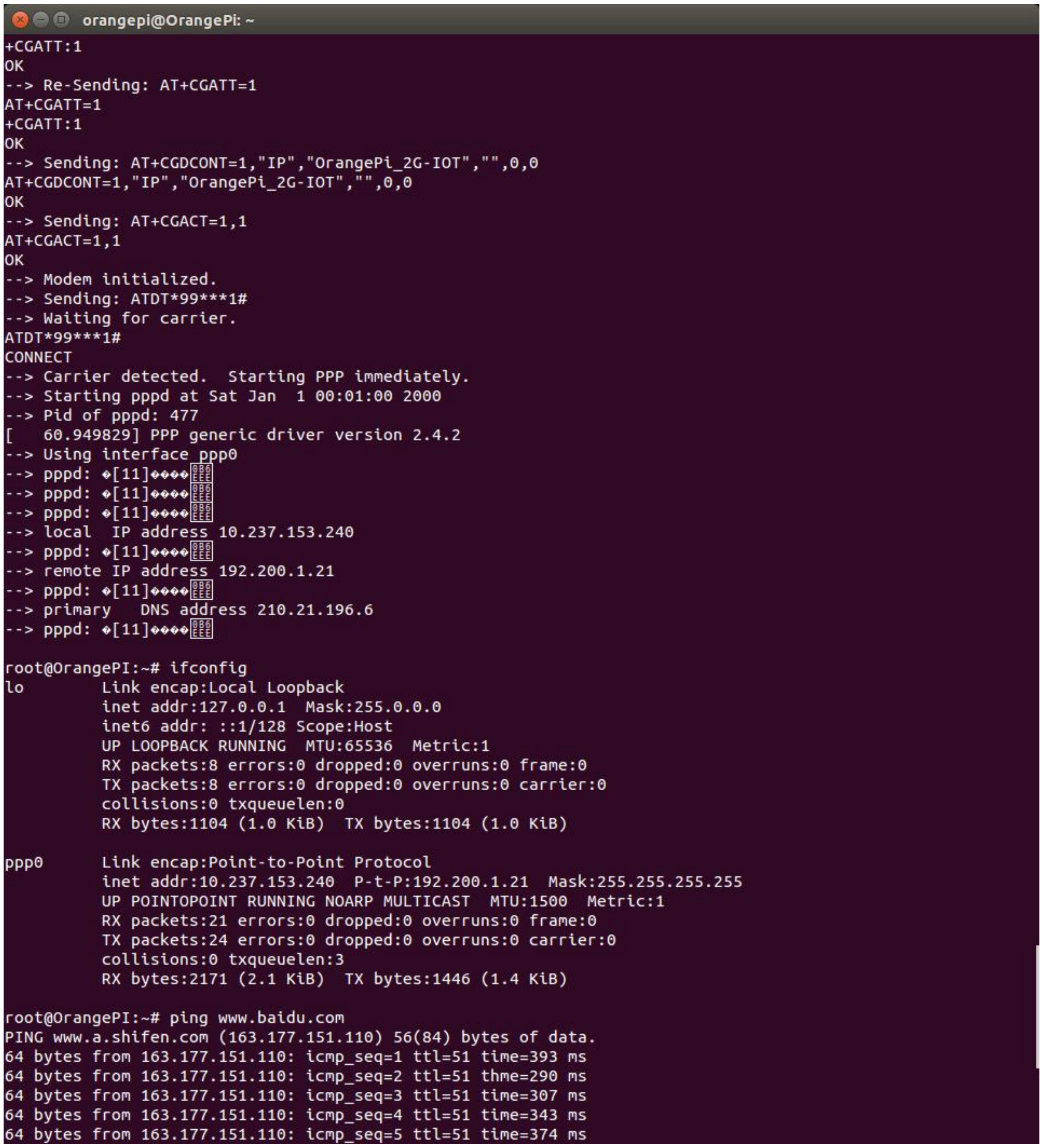
Execute the following command after power on and login to ppp dial up.

```
wvdia Tom &
```

This command will run at the backstage for dial up, after dial-up access, you could use ping and ifconfig to check the network. If network connect successful, then you could use GSM to connect network.



```
  ___                            ____  _ 
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/              
***********************************************
Welcome to OrangePi
Please General Setting first, as follow:
sudo OrangePi_Settings
Good Luck!ÿ
***********************************************
root@OrangePI:~# wvdial Tom &
[1] 523
root@OrangePI:~# --> WvDial: Internet dialer version 1.61
--> Warning: section [Dialer Tom] does not exist in wvdial.conf.
[   66.197265] md_tty_install result 0
--> Cannot get information for serial port.
--> Initializing modem.
--> Sending: AT+COPS=0
AT+COPS=0
--> Sending: ATQ0
ATQ0
+CME ERROR:50
--> Re-Sending: AT+COPS=0
AT+COPS=0
+CIEV: service,  1
+CIEV: roam, 1
+CREG: 5
OK
--> Sending: AT+CFUN=1
AT+CFUN=1
OK
--> Sending: AT+CGATT=1
AT+CGATT=1
--> Sending: ATQ0
ATQ0
+CGATT:1
OK
--> Re-Sending: AT+CGATT=1
AT+CGATT=1
+CGATT:1
OK
--> Sending: AT+CGDCONT=1,"IP","OrangePi_2G-IOT","",0,0
AT+CGDCONT=1,"IP","OrangePi_2G-IOT","",0,0
OK
--> Sending: AT+CGACT=1,1
AT+CGACT=1,1
OK
--> Modem initialized.
--> Sending: ATDT*99***1#
--> Waiting for carrier.
ATDT*99***1#
CONNECT
--> Carrier detected.  Starting PPP immediately.
--> Starting pppd at Mon Jun  5 11:27:27 2017
--> Pid of pppd: 530
```



```
+CGATT:1
OK
--> Re-Sending: AT+CGATT=1
AT+CGATT=1
+CGATT:1
OK
--> Sending: AT+CGDCONT=1,"IP","OrangePi_2G-IOT","",0,0
AT+CGDCONT=1,"IP","OrangePi_2G-IOT","",0,0
OK
--> Sending: AT+CGACT=1,1
AT+CGACT=1,1
OK
--> Modem initialized.
--> Sending: ATDT*99***1#
--> Waiting for carrier.
ATDT*99***1#
CONNECT
--> Carrier detected.  Starting PPP immediately.
--> Starting pppd at Sat Jan  1 00:01:00 2000
--> Pid of pppd: 477
[   60.949829] PPP generic driver version 2.4.2
--> Using interface ppp0
--> pppd: �[11]����
--> pppd: �[11]����
--> pppd: �[11]����
--> local  IP address 10.237.153.240
--> pppd: �[11]����
--> remote IP address 192.200.1.21
--> pppd: �[11]����
--> primary   DNS address 210.21.196.6
--> pppd: �[11]����

root@OrangePI:~# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1104 (1.0 KiB)  TX bytes:1104 (1.0 KiB)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.237.153.240  P-t-P:192.200.1.21  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:21 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:2171 (2.1 KiB)  TX bytes:1446 (1.4 KiB)

root@OrangePI:~# ping www.baidu.com
PING www.a.shifen.com (163.177.151.110) 56(84) bytes of data.
64 bytes from 163.177.151.110: icmp_seq=1 ttl=51 time=393 ms
64 bytes from 163.177.151.110: icmp_seq=2 ttl=51 thme=290 ms
64 bytes from 163.177.151.110: icmp_seq=3 ttl=51 time=307 ms
64 bytes from 163.177.151.110: icmp_seq=4 ttl=51 time=343 ms
64 bytes from 163.177.151.110: icmp_seq=5 ttl=51 time=374 ms
```

III. Source Code Compilation of Android and Linux

Hardware: Orange Pi development board*1, Card reader*1, TF card*1, power supply*1

Software: Linux host computer, which hard disk space at least 50G (to meet a fully compiled need)

Linux host computer needs:
Version 2.7.3 of Python;
Version 3.81-3.82 of GNU Make;
JDK1.6;
Version 1.7 or higher version of Git.

1. Install JDK

- Download and unzip JDK, you will get jdk-6u31-linux-x64.bin, copy it to the directory of /opt
- Modify the permission of jdk-6u31-linux-x64.bin with following command:

```
sudo chmod 755    jdk-6u31-linux-x64.bin
```

- Install jdk1.6

```
/jdk-6u31-linux-x64.bin
```

- Configuration multi Java version coexistence mode with the following command:

```
sudo update-alternatives --install /user/bin/java java /opt/jdk1.6.0_31/bin/java 300
sudo update-alternatives --install /user/bin/javap javap /opt/jdk1.6.0_31/bin/javap 300
sudo update-alternatives --install /user/bin/javac javac /opt/jdk1.6.0_31/bin/javac 300
sudo update-alternatives --install /user/bin/jar jar /opt/jdk1.6.0_31/bin/jar 300
sudo update-alternatives --install /user/bin/javaws javaws /opt/jdk1.6.0_31/bin/javaws 300
sudo update-alternatives --install /user/bin/javapdoc javadoc /opt/jdk1.6.0_31/bin/javadoc
```

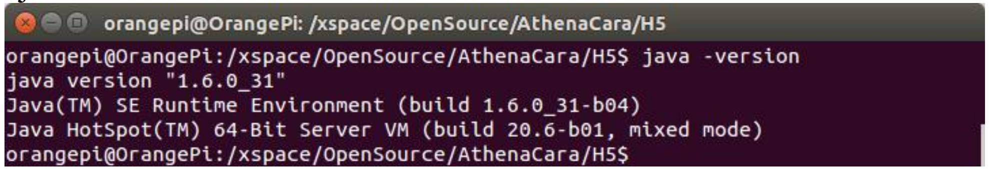
300

- Switch to java version and select version 1.6, use the following command:

```
sudo update-alternatives --config java
sudo update-alternatives --config javac
sudo update-alternatives --config jar
sudo update-alternatives --config javap
sudo update-alternatives --config javaws
sudo update-alternatives --config javadoc
```

- After confirmed it is version 1.6, you could use the following command:

```
java -version
```



```
orangepi@OrangePi: /xspace/OpenSource/AthenaCara/H5
orangepi@OrangePi:/xspace/OpenSource/AthenaCara/H5$ java -version
java version "1.6.0_31"
Java(TM) SE Runtime Environment (build 1.6.0_31-b04)
Java HotSpot(TM) 64-Bit Server VM (build 20.6-b01, mixed mode)
orangepi@OrangePi:/xspace/OpenSource/AthenaCara/H5$
```

2. Install Platform Supported Software

```
    $ sudo apt-get install git gnupg flex bison gperf build-essential \
       zip curl libc6-dev libncurses5-dev:i386 x11proto-core-dev \
       libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 \
       libgl1-mesa-dev g++-multilib mingw32 tofrodos \
       python-markdown libxml2-utils xsltproc zlib1g-dev:i386
$ sudo ln -s /usr/lib/i386-linux-gnu/mesa/libGL.so.1/usr/lib/i386-linux-gnu/libGL.so
```

3. Download the Source Package and Unzip it

Download website: http://www.orangepi.org/downloadresources/
Downloaded source package and use the following command:

```
$cat OrangePi_2G-IOT* > tar.tar.gz
$ tar –xvzf    tar.tar.gz
```

Unzip the file you will get the trunk directory, enter it via the terminal.

4. Android source code compiler

Before compiling Android source code, please make sure you have already installed JAVA 1.6 version, if not, please refer to previous charter to install first. After you install jave 1.6 successfully, you could begin to compile Android source code.

Before compile Android source code, you need to first compile modem kernel for both Tcard Startup and Nand Startup. modem kernel is misp framework, you need to use corresponding cross compilation tool.Youo could use the following command to unzip the cross compilation tool file:

```
$ tar -xvjf OrangePi_Doc.tar.bz2
$ cd Platform/RDA/modem-cross
$ tar -xvzf modem-cross-compiler-linux.tar.gz
$ cd cross-compiler
```

After you get the corresponding tool, install the cross compilation tool of modem as the following steps:

Step 1: Cross-Compiler Installation

```
$ sudo tar -xzf ~/modem-cross-compiler-linux.tar.gz -C /opt
$ ls /opt/cross-compiler
$ sudo chown -R root:root /opt/cross-compiler
$ sudo chmod +x /opt/cross-compiler/bin/*
```

Step 2: Environment Configuration

```
$ sudo sh -c 'echo export PATH=$PATH:/opt/cross-compiler/bin >
/etc/profile.d/cross-compiler.sh'
$ sudo sh -c 'echo /opt/cross-compiler/lib > /etc/ld.so.conf.d/cross-compiler.conf'
$ sudo ldconfig -v
$ sudo ldconfig -v | grep '/opt/cross-compiler/lib'
$ mips-elf-gcc -v
```

- Select source code:
  Use command to switch to Android source code:
  ```
  cd */trunk/
  ```
- Import development variables
  ```
  $ source build/envsetup.sh
  ```
- Select project
  ```
  $ lunch
  ```

If boot from TF card, select slt-userdebug, then select NollecA9V2VV8810P_ext4
If boot from Nand, select etu-userdebug, then select NollecA9V2VV8810P

- Compile system
  ```
  $ make –j
  ```
- Update image if boot from TF card

After compile Android source code for booting from TF card, you will get a new image on the directory of:

```
*/trunk/out/target/product/slt**/
```

And use the following commands to update it:

```
sudo dd if=bootloader.img of=/dev/sdc bs=512 seek=256 count=4096 && sync
sudo dd if=modem.img of=/dev/sdc bs=512 seek=12544 count=8192 && sync
sudo dd if=boot.img of=/dev/sdc bs=512 seek=20736 count=16384 && sync
sudo dd if=recovery.img of=/dev/sdc bs=512 seek=37120 count=20480 && sync
sudo dd if=system.ext4.img of=/dev/sdc bs=512 seek=57600  count=512000 && sync
sudo dd if=vendor.ext4.img of=/dev/sdc bs=512 seek=569600 count=512000 && sync
```

/dev/sdc is the mounted number on system of SD card.

- Nand update

There will be corespondent image on the directory of */trunk/out/target/product/etu**/ after compilation. Update the image into system with NAND update tool. About the details steps you could refer to How to update Android Nand in the manual.

5. Compile Linux source Code

Linux source code of Orange Pi 2G-IOT has been updated to github, you could download from github. Compile Linux would require you work under Linux environment. We would recommend you use Ubuntu 16.04 of Linux PC.

- Download Linux source code

You could download Linux source code from github:
https://github.com/OrangePiLibra/OrangePi

You could also use git command to update:

```
git clone https://github.com/OrangePiLibra/OrangePi.git
```

- Compile source code

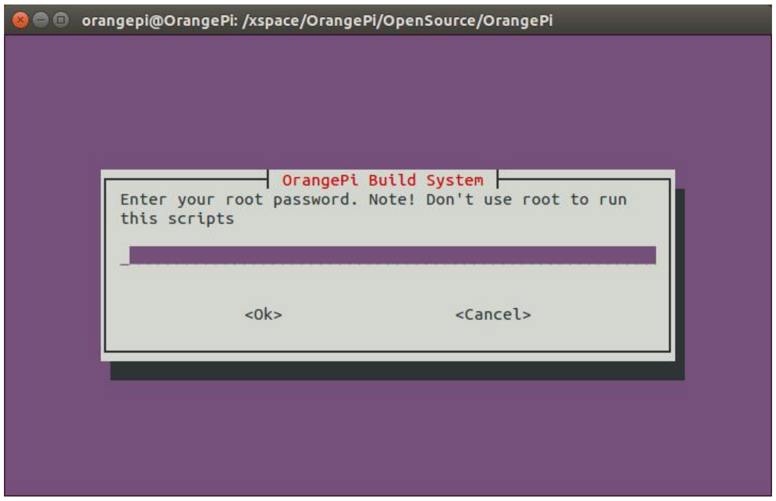
Use the following command to enter into source code directory after you get the source code:

```
cd */OrangePi
```

Execute the following script:

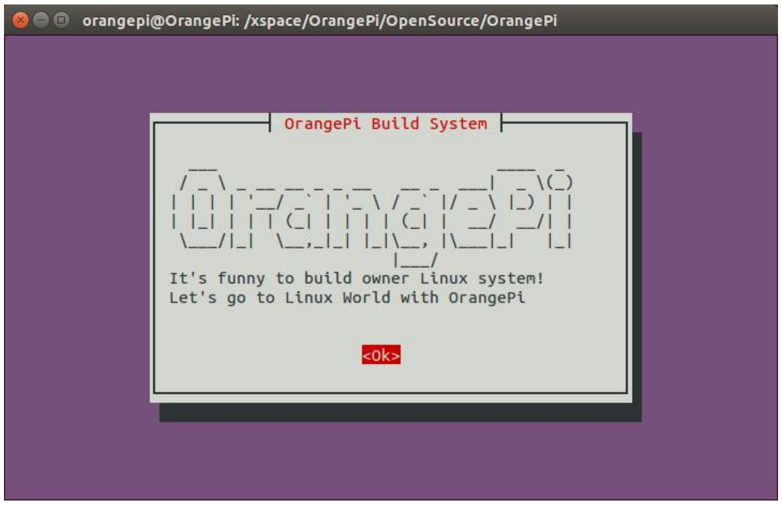
```
./Build_OrangePi.sh
```

Input root password:



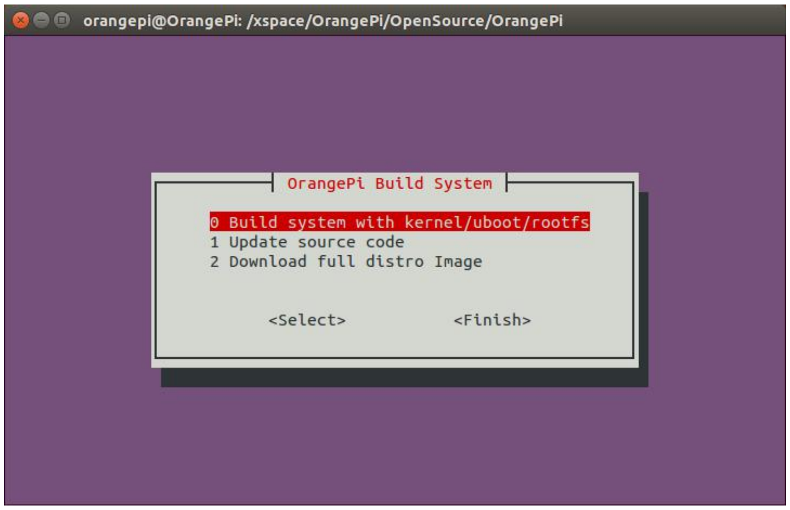
```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePi

┤ OrangePi Build System ├
Enter your root password. Note! Don't use root to run
this scripts

_

        <Ok>                    <Cancel>
```

After root password recognize successful, enter inter main interface and use Enter key.



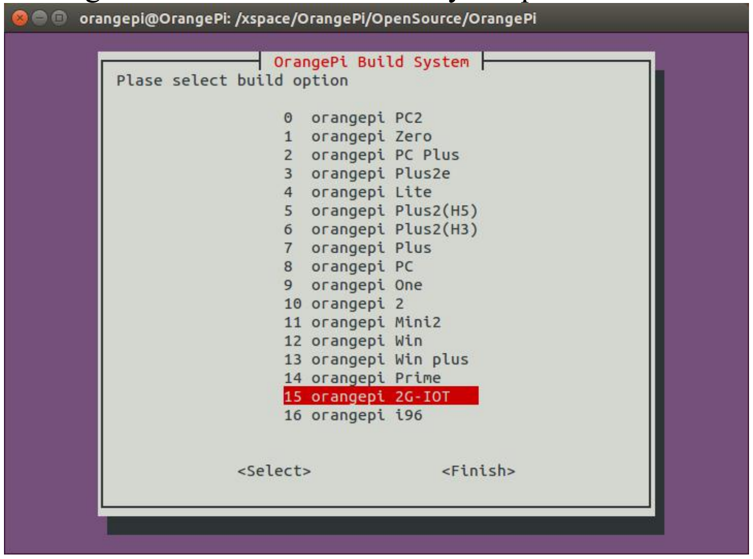
```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePi

                    OrangePi Build System
  ___                            ____  _
 / _ \ _ __ __ _ _ __   __ _  ___|  _ \(_)
| | | | '__/ _` | '_ \ / _` |/ _ \ |_) | |
| |_| | | | (_| | | | | (_| |  __/  __/| |
 \___/|_|  \__,_|_| |_|\__, |\___|_|   |_|
                       |___/
It's funny to build owner Linux system!
Let's go to Linux World with OrangePi

                           <Ok>
```

Select "Build system with kernel/uboot/rootfs" on main functional interface and use Enter key.



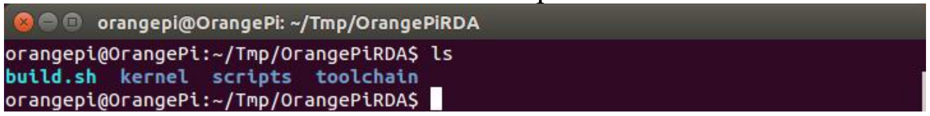
```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePi

                    OrangePi Build System

        0 Build system with kernel/uboot/rootfs
        1 Update source code
        2 Download full distro Image


              <Select>              <Finish>
```

And then select "OrangePi 2G-IOT" with Enter key to update source code.



```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePi

                    OrangePi Build System
Plase select build option

                  0  orangepi PC2
                  1  orangepi Zero
                  2  orangepi PC Plus
                  3  orangepi Plus2e
                  4  orangepi Lite
                  5  orangepi Plus2(H5)
                  6  orangepi Plus2(H3)
                  7  orangepi Plus
                  8  orangepi PC
                  9  orangepi One
                  10 orangepi 2
                  11 orangepi Mini2
                  12 orangepi Win
                  13 orangepi Win plus
                  14 orangepi Prime
                  15 orangepi 2G-IOT
                  16 orangepi i96


             <Select>              <Finish>
```

It would take around 40minutes to update source code and corresponding scripts. After updated the source code, there will be generated a directory of OrangePiRDA. This directory contains both Linux source code and scripts:



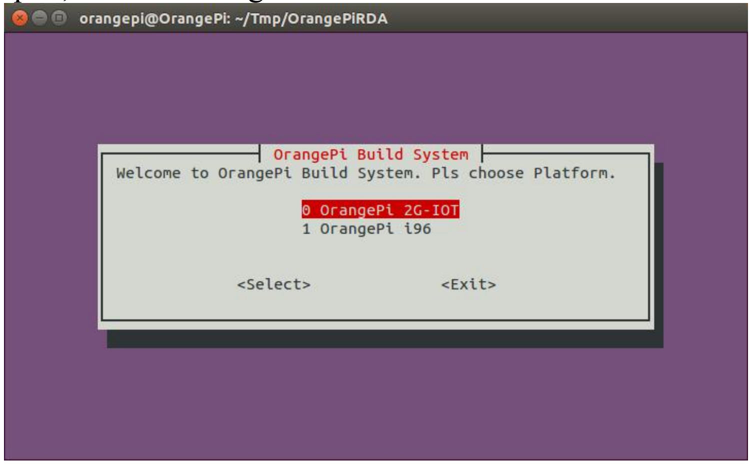
```
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ls
build.sh  kernel  scripts  toolchain
orangepi@OrangePi:~/Tmp/OrangePiRDA$
```

- Compile Linux

Execute the following command after enter into directory of OrangePiRDA:

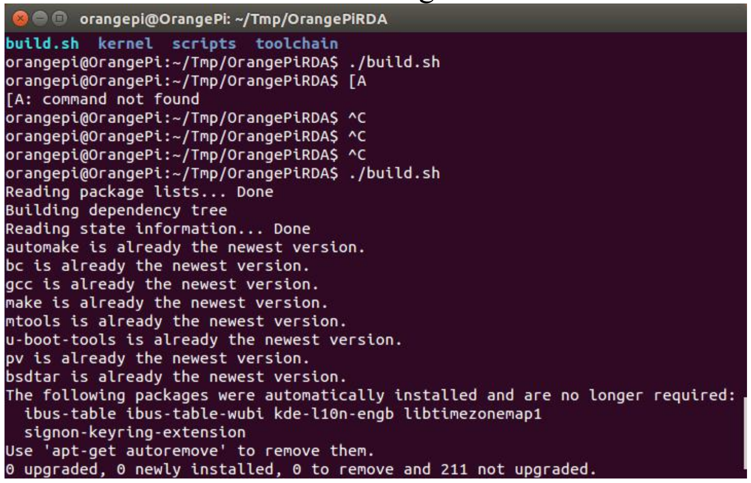
```
./build.sh
```

The script is is an automatic script, you could select a corresponding board which you want to compile, here is “OrangePi 2G-IOT”.



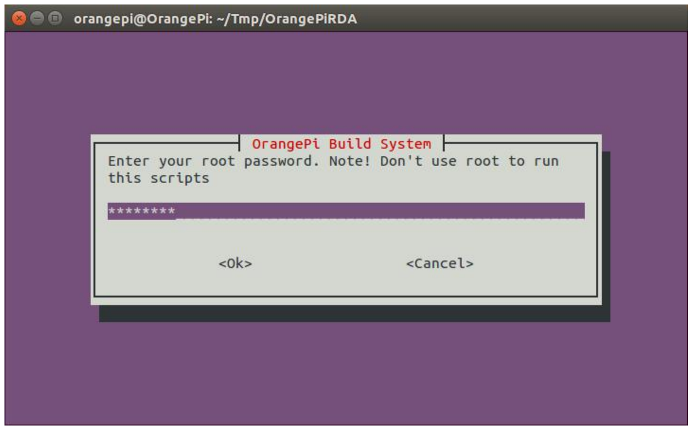
```
OrangePi Build System
Welcome to OrangePi Build System. Pls choose Platform.

0 OrangePi 2G-IOT
1 OrangePi i96

<Select>          <Exit>
```

If it is the first time you run the script, the system would install development tool automatic to make sure the network is connecting.



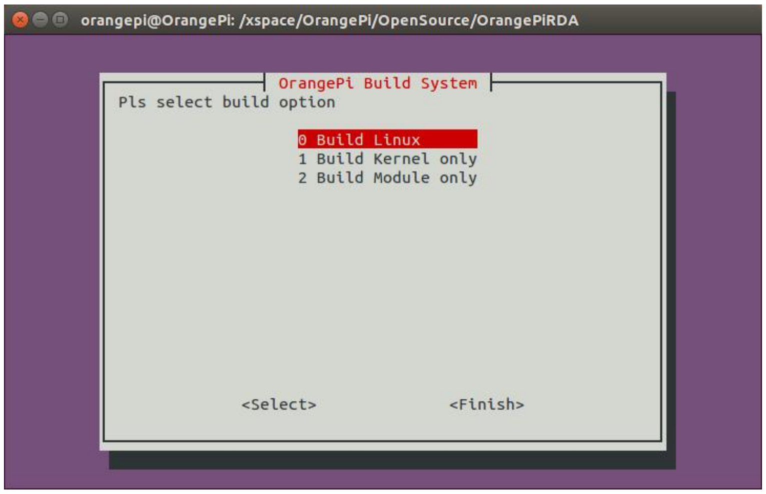
```
build.sh  kernel  scripts  toolchain
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ./build.sh
orangepi@OrangePi:~/Tmp/OrangePiRDA$ [A
[A: command not found
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ^C
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ^C
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ^C
orangepi@OrangePi:~/Tmp/OrangePiRDA$ ./build.sh
Reading package lists... Done
Building dependency tree
Reading state information... Done
automake is already the newest version.
bc is already the newest version.
gcc is already the newest version.
make is already the newest version.
mtools is already the newest version.
u-boot-tools is already the newest version.
pv is already the newest version.
bsdtar is already the newest version.
The following packages were automatically installed and are no longer required:
  ibus-table ibus-table-wubi kde-l10n-engb libtimezonemap1
  signon-keyring-extension
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 211 not upgraded.
```

After installed tool, enter root password and use Enter key.



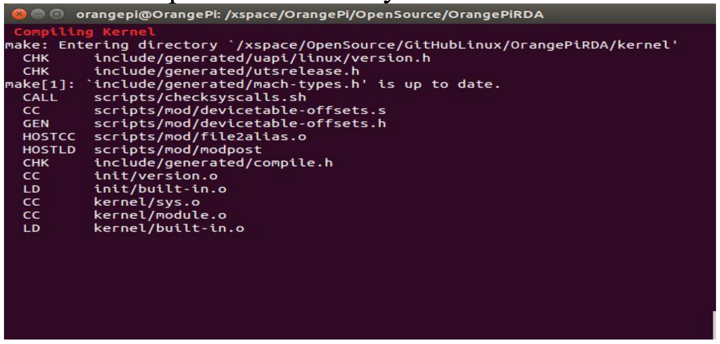
```
orangepi@OrangePi: ~/Tmp/OrangePiRDA

┤ OrangePi Build System ├
Enter your root password. Note! Don't use root to run
this scripts

********

        <Ok>                    <Cancel>
```

You will enter into the main interface after entering password, select what you are going to do:



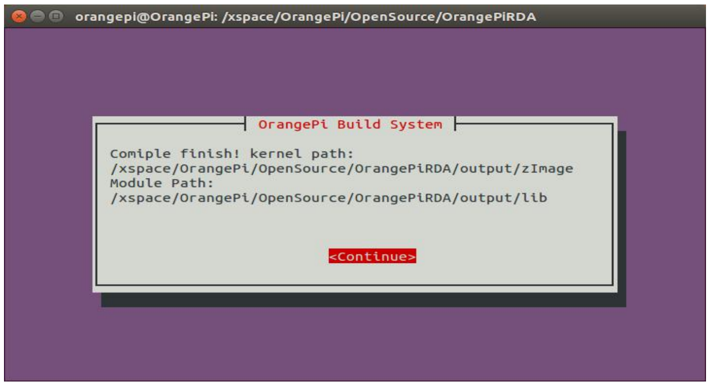
```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePiRDA

┤ OrangePi Build System ├
Pls select build option

                  0 Build Linux
                  1 Build Kernel only
                  2 Build Module only

        <Select>                <Finish>
```

This version is only support the above three options. After selecting the corresponding option, the system would compile automatically.



```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePiRDA
 Compiling Kernel
make: Entering directory `/xspace/OpenSource/GitHubLinux/OrangePiRDA/kernel'
  CHK     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
make[1]: `include/generated/mach-types.h' is up to date.
  CALL    scripts/checksyscalls.sh
  CC      scripts/mod/devicetable-offsets.s
  GEN     scripts/mod/devicetable-offsets.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTLD  scripts/mod/modpost
  CHK     include/generated/compile.h
  CC      init/version.o
  LD      init/built-in.o
  CC      kernel/sys.o
  CC      kernel/module.o
  LD      kernel/built-in.o
```

There will be prompt the location of kernel image and module after compilation.



```
orangepi@OrangePi: /xspace/OrangePi/OpenSource/OrangePiRDA

OrangePi Build System

Comiple finish! kernel path:
/xspace/OrangePi/OpenSource/OrangePiRDA/output/zImage
Module Path:
/xspace/OrangePi/OpenSource/OrangePiRDA/output/lib

<Continue>
```

- Update Linux Kernel and module

After finished the above compilation steps, you could update the new kernel and module into the board to run it. Before this, you could refer to the charter about Linux image writing section to write a Linux distro into SD card. After written image, insert SD card into PC and till now it would recognize there are two partitions, one is boot partition with file of uboot, kernel and Ramdisk. The other partition is rootfs partion which contains root file system.

There is already marked the location of generated kernel, you only need to copy the generated zImage into first partition of SD card and replace zImage inside. Till now the kernel has been updated.

And there is already marked the location of new generated module, the second SD card partition is Rootfs partition, you need to have root permission to delete the directory of rootfs/lib/modules/3.xxx with following command:

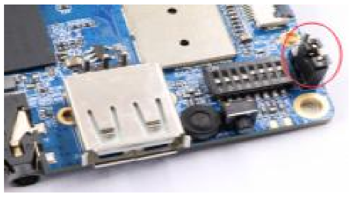
```
sudo rm -rf    */rootfs/lib/modules/3.xxx
```

Copy the new generated module into rootfs partition you need to use the following command:

```
sudo cp -rf    */OrangePiRDA/output/lib/modules/3.xxx    */rootfs/lib/modules/ sync
```

After all above steps, kernel and module update have been finished.

You could insert SD card into Orange pi, and make the jumper like the following, after booting, it would enter into Linux.



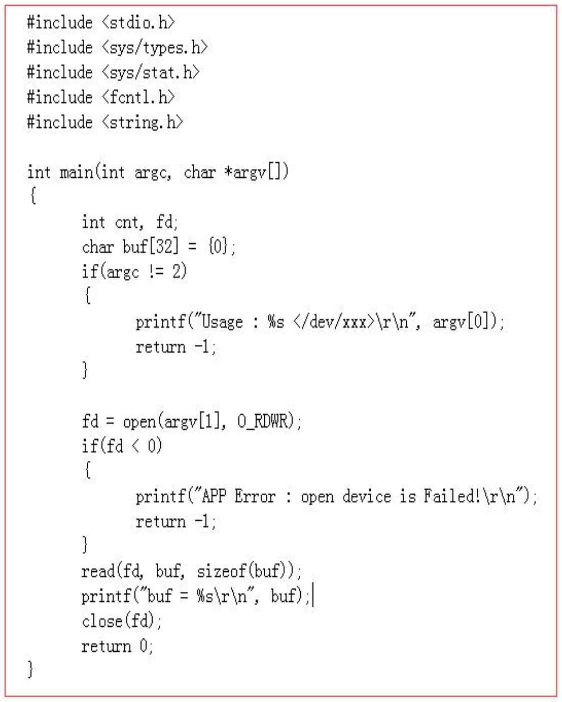
IV. Orange Pi Driver development

In order to help developers more familiar with Orange Pi, this instruction will make a brief illustration on device driver module and application program.

Hardware: Orange Pi development board*1, Card reader*1, TF card*1, power supply*1

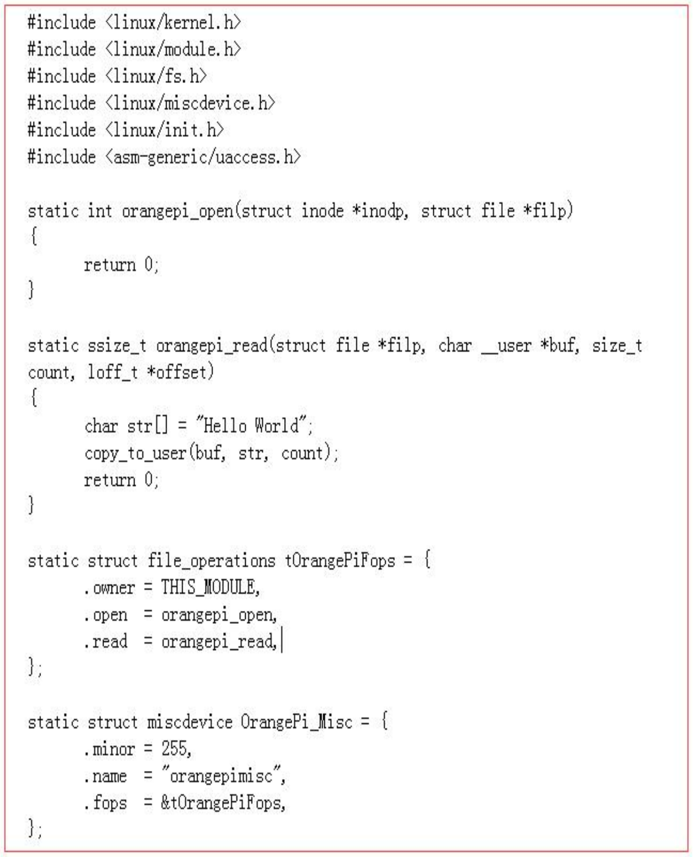
1. Device driver and application programming

1) Application Program (app.c)

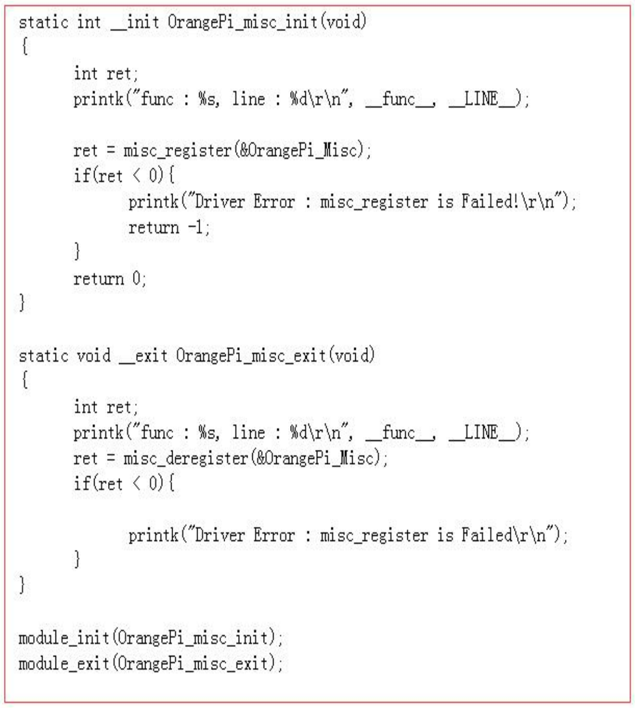


```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>

int main(int argc, char *argv[])
{
        int cnt, fd;
        char buf[32] = {0};
        if(argc != 2)
        {
                printf("Usage : %s </dev/xxx>\r\n", argv[0]);
                return -1;
        }

        fd = open(argv[1], O_RDWR);
        if(fd < 0)
        {
                printf("APP Error : open device is Failed!\r\n");
                return -1;
        }
        read(fd, buf, sizeof(buf));
        printf("buf = %s\r\n", buf);
        close(fd);
        return 0;
}
```

2) Driver Program (OrangePi_misc.c)



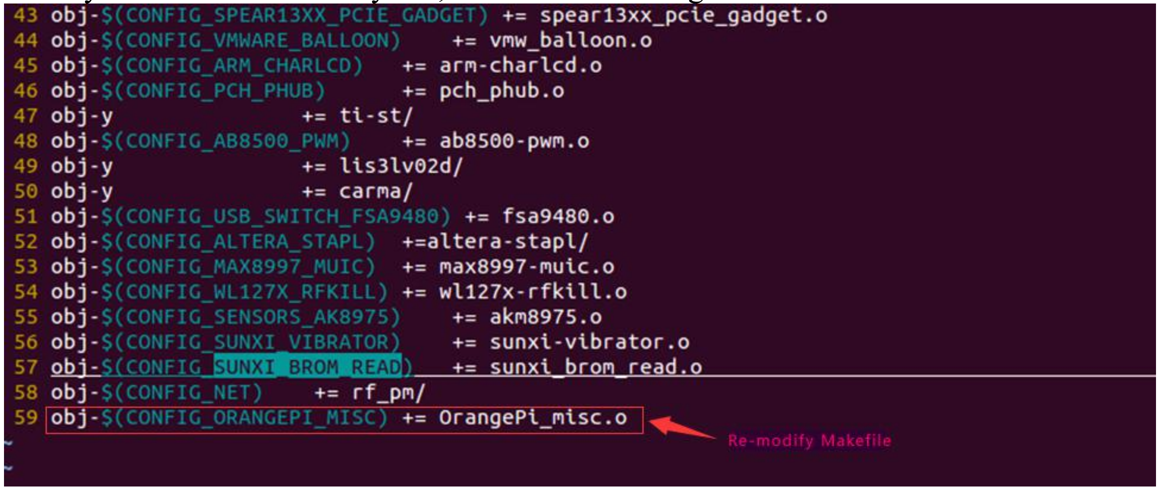
```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/fs.h>
#include <linux/miscdevice.h>
#include <linux/init.h>
#include <asm-generic/uaccess.h>

static int orangepi_open(struct inode *inodp, struct file *filp)
{
        return 0;
}

static ssize_t orangepi_read(struct file *filp, char __user *buf, size_t
count, loff_t *offset)
{
        char str[] = "Hello World";
        copy_to_user(buf, str, count);
        return 0;
}

static struct file_operations tOrangePiFops = {
        .owner = THIS_MODULE,
        .open  = orangepi_open,
        .read  = orangepi_read,
};

static struct miscdevice OrangePi_Misc = {
        .minor = 255,
        .name  = "orangepimisc",
        .fops  = &tOrangePiFops,
};
```



```
static int __init OrangePi_misc_init(void)
{
        int ret;
        printk("func : %s, line : %d\r\n", __func__, __LINE__);

        ret = misc_register(&OrangePi_Misc);
        if(ret < 0){
                printk("Driver Error : misc_register is Failed!\r\n");
                return -1;
        }
        return 0;
}

static void __exit OrangePi_misc_exit(void)
{
        int ret;
        printk("func : %s, line : %d\r\n", __func__, __LINE__);
        ret = misc_deregister(&OrangePi_Misc);
        if(ret < 0){

                printk("Driver Error : misc_register is Failed\r\n");
        }
}

module_init(OrangePi_misc_init);
module_exit(OrangePi_misc_exit);
```

2. Compile device driver

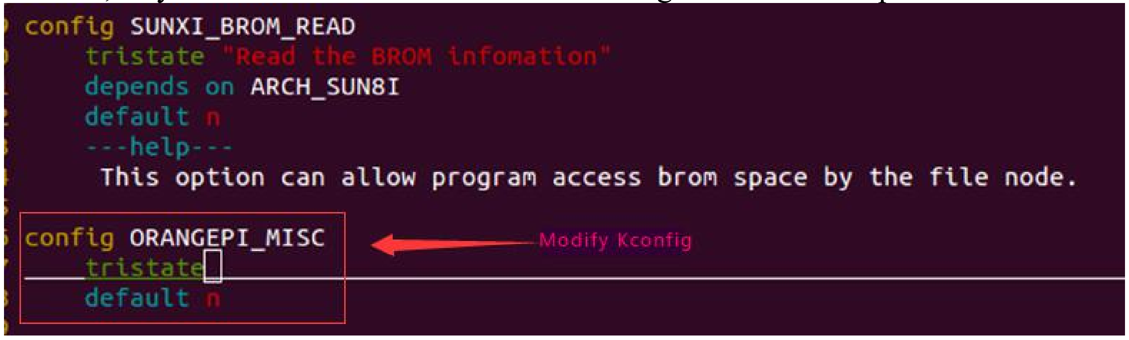
Copy the OrangePi_misc.c to the */trunk/kernel/driver/misc:
Enter to */trunk/kernel/driver/misc
Modify Makefile on currently file, shown as following:



```
obj-$(CONFIG_SPEAR13XX_PCIE_GADGET) += spear13xx_pcie_gadget.o
obj-$(CONFIG_VMWARE_BALLOON)     += vmw_balloon.o
obj-$(CONFIG_ARM_CHARLCD)     += arm-charlcd.o
obj-$(CONFIG_PCH_PHUB)       += pch_phub.o
obj-y                   += ti-st/
obj-$(CONFIG_AB8500_PWM)     += ab8500-pwm.o
obj-y                   += lis3lv02d/
obj-y                   += carma/
obj-$(CONFIG_USB_SWITCH_FSA9480) += fsa9480.o
obj-$(CONFIG_ALTERA_STAPL)  +=altera-stapl/
obj-$(CONFIG_MAX8997_MUIC)  += max8997-muic.o
obj-$(CONFIG_WL127X_RFKILL) += wl127x-rfkill.o
obj-$(CONFIG_SENSORS_AK8975)     += akm8975.o
obj-$(CONFIG_SUNXI_VIBRATOR)     += sunxi-vibrator.o
obj-$(CONFIG_SUNXI_BROM_READ)    += sunxi_brom_read.o
obj-$(CONFIG_NET)     += rf_pm/
obj-$(CONFIG_ORANGEPI_MISC) += OrangePi_misc.o
```

There is Kconfig on the same sibling folders with Makefile. Each Kconfig respectively describes the the source directory file related kernel configuration menu. In the kernel configuration making menuconfig, it read from the Kconfig config menu and the user configuration saved to the config. In the kernel compile, the main Makefile by calling this.Config could know the user's configuration of the kernel.

Kconfig is corresponding to the kernel configuration menu. Add a new driver to the kernel source code, you can modify the Kconfig to increase the configuration menu for your drive, so you can choose whether the menuconfig driver was compiled or not.



```
config SUNXI_BROM_READ
    tristate "Read the BROM infomation"
    depends on ARCH_SUN8I
    default n
    ---help---
     This option can allow program access brom space by the file node.

config ORANGEPI_MISC
    tristate
    default n
```

Back to the source code directory /trunk:

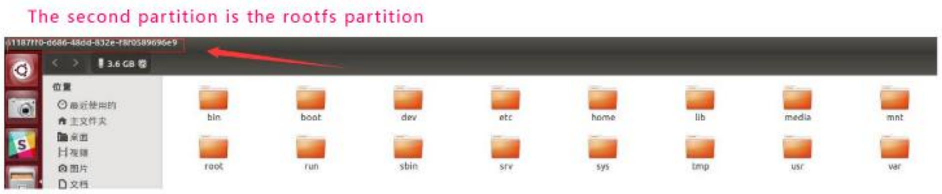
$ make bootimage

Make sure it have already finished make-engineer-configuration before execute this command, if not, please refer to last section about Linux source code compilation.

Update the new generated module file into Linux system.

It will show on *cd /trunk/out/target/product/slt-NollecA9V2VV8810P_ext4/obj/KERNEL/out/target/product/slt-NollecA9V2VV8810P_ext4/obj/KERNEL/modules/lib/modules/3.10.62-rel5.0.2/ generated corresponding .ko file, it is the module that generated after OrangePi_misc.c compilation.

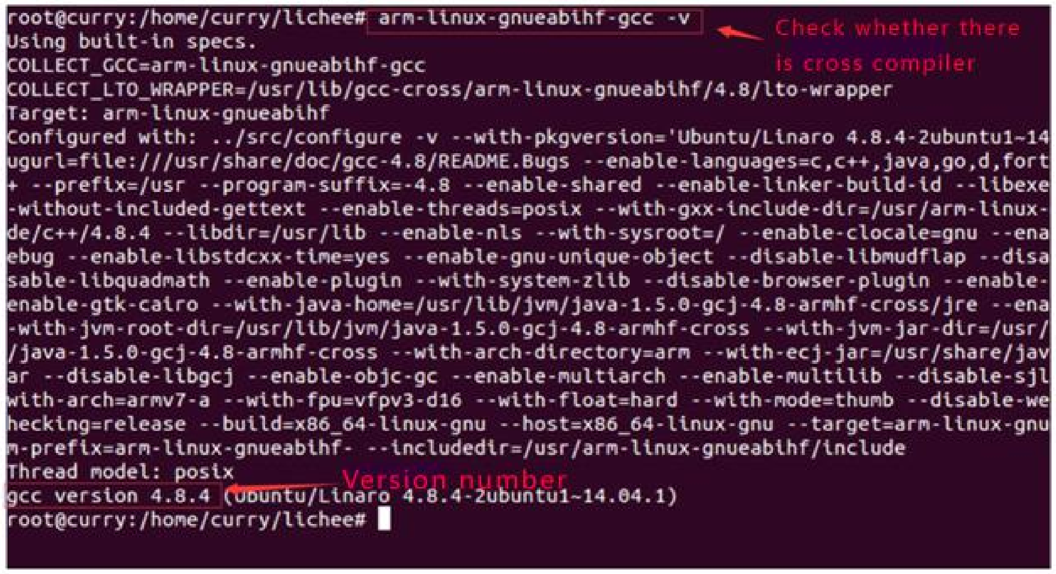
Insert U disk (please note the SD card should have written image) if the SD card is mounted to the directory system of /dev/sdc, then SD card will mount to rootfs, which is /dev/sdc7, and mounted to rootfs partition automatic.



Copy the directory file:
/trunk/out/target/product/slt-NollecA9V2VV8810P_ext4/obj/KERNEL/out/target/product/slt-NollecA9V2VV8810P_ext4/obj/KERNEL/modules/lib/modules/3.10.62-rel5.0.2/
into:
/media/*/lib/modules/

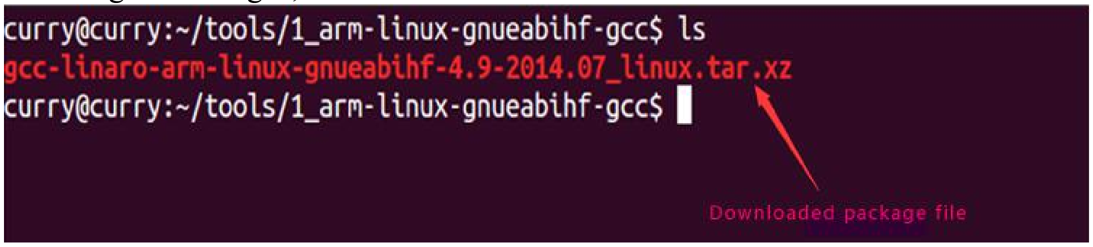
3. Compiling method of application

Check whether there is the cross compiler, if not, then download and install it.
$ arm-linux-gnueabihf-gcc -v



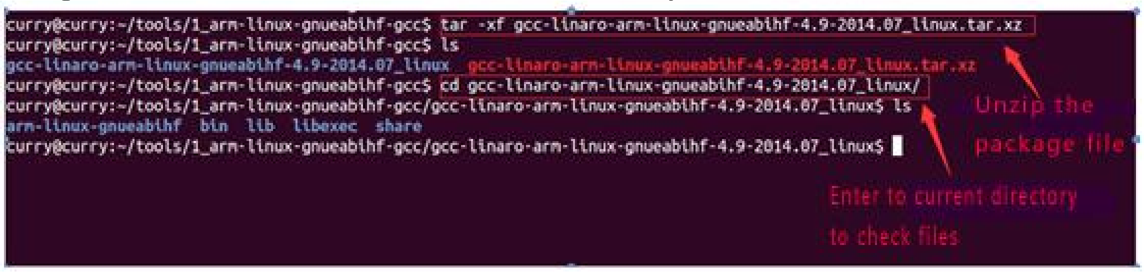
```
root@curry:/home/curry/lichee# arm-linux-gnueabihf-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gnueabihf-gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc-cross/arm-linux-gnueabihf/4.8/lto-wrapper
Target: arm-linux-gnueabihf
Configured with: ../src/configure -v --with-pkgversion='Ubuntu/Linaro 4.8.4-2ubuntu1~14
ugurl=file:///usr/share/doc/gcc-4.8/README.Bugs --enable-languages=c,c++,java,go,d,fort
+ --prefix=/usr --program-suffix=-4.8 --enable-shared --enable-linker-build-id --libexe
-without-included-gettext --enable-threads=posix --with-gxx-include-dir=/usr/arm-linux-
de/c++/4.8.4 --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --ena
ebug --enable-libstdcxx-time=yes --enable-gnu-unique-object --disable-libmudflap --disa
sable-libquadmath --enable-plugin --with-system-zlib --disable-browser-plugin --enable-
enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-4.8-armhf-cross/jre --ena
-with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-4.8-armhf-cross --with-jvm-jar-dir=/usr/
/java-1.5.0-gcj-4.8-armhf-cross --with-arch-directory=arm --with-ecj-jar=/usr/share/jav
ar --disable-libgcj --enable-objc-gc --enable-multiarch --enable-multilib --disable-sjl
with-arch=armv7-a --with-fpu=vfpv3-d16 --with-float=hard --with-mode=thumb --disable-we
hecking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=arm-linux-gnu
m-prefix=arm-linux-gnueabihf- --includedir=/usr/arm-linux-gnueabihf/include
Thread model: posix
gcc version 4.8.4 (Ubuntu/Linaro 4.8.4-2ubuntu1~14.04.1)
root@curry:/home/curry/lichee#
```

While compiling the application, you will fill that you need the cross compiler arm-linux-gnueabihf-gcc, download and install it.



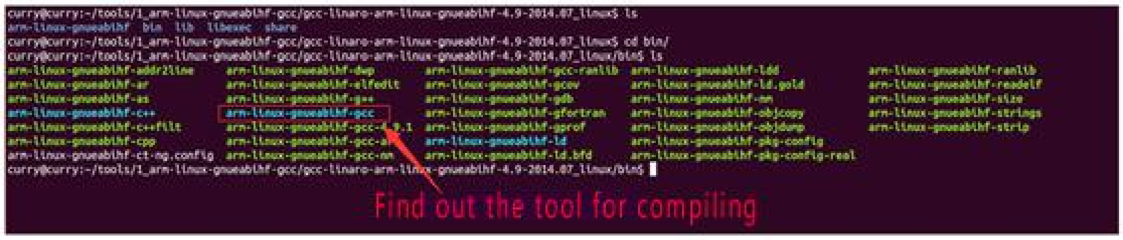
```
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ ls
gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux.tar.xz
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$
```

Unzip the downloaded file and enter the the directory



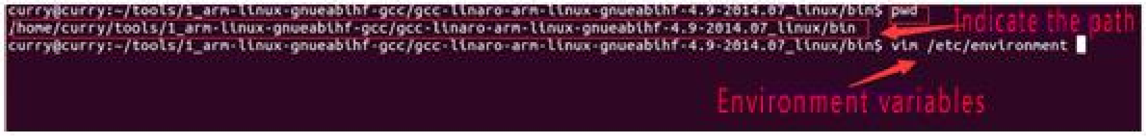
```
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ tar -xf gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux.tar.xz
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ ls
gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux  gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux.tar.xz
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ cd gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux$ ls
arm-linux-gnueabihf  bin  lib  libexec  share
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux$
```

Check the information after entering bin directory



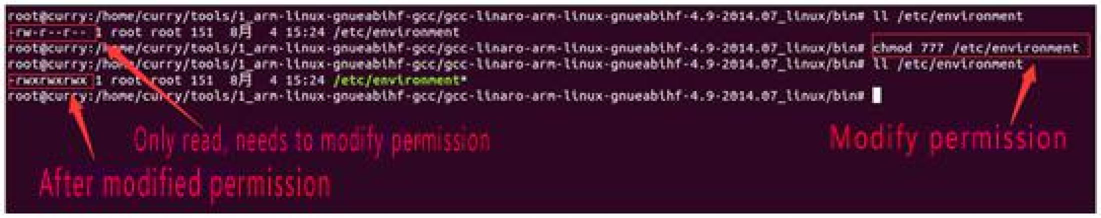
```
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux$ ls
arm-linux-gnueabihf  bin  lib  libexec  share
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux$ cd bin/
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin$ ls
arm-linux-gnueabihf-addr2line     arm-linux-gnueabihf-dwp         arm-linux-gnueabihf-gcc-ranlib  arm-linux-gnueabihf-ldd              arm-linux-gnueabihf-ranlib
arm-linux-gnueabihf-ar            arm-linux-gnueabihf-elfedit     arm-linux-gnueabihf-gcov        arm-linux-gnueabihf-ld.gold          arm-linux-gnueabihf-readelf
arm-linux-gnueabihf-as            arm-linux-gnueabihf-g++         arm-linux-gnueabihf-gdb         arm-linux-gnueabihf-nm               arm-linux-gnueabihf-size
arm-linux-gnueabihf-c++           arm-linux-gnueabihf-gcc         arm-linux-gnueabihf-gfortran    arm-linux-gnueabihf-objcopy          arm-linux-gnueabihf-strings
arm-linux-gnueabihf-c++filt       arm-linux-gnueabihf-gcc-4.9.1   arm-linux-gnueabihf-gprof       arm-linux-gnueabihf-objdump          arm-linux-gnueabihf-strip
arm-linux-gnueabihf-cpp           arm-linux-gnueabihf-gcc-ar      arm-linux-gnueabihf-ld          arm-linux-gnueabihf-pkg-config
arm-linux-gnueabihf-ct-ng.config  arm-linux-gnueabihf-gcc-nm      arm-linux-gnueabihf-ld.bfd      arm-linux-gnueabihf-pkg-config-real
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin$
```

pwd hows the path and export it into the whole project



```
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin$ pwd
/home/curry/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin$ vim /etc/environment
```

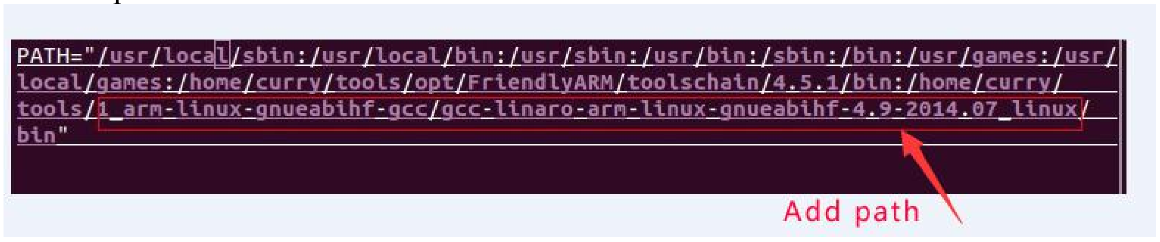
$ ll /etc/environment shows that the file can only read, need to modify permissions
$ chmod 755 /etc/environment



```
root@curry:/home/curry/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin# ll /etc/environment
-rw-r--r-- 1 root root 151  8月  4 15:24 /etc/environment
root@curry:/home/curry/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin# chmod 777 /etc/environment
root@curry:/home/curry/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin# ll /etc/environment
-rwxrwxrwx 1 root root 151  8月  4 15:24 /etc/environment*
root@curry:/home/curry/tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin#
```

Only read, needs to modify permission
After modified permission
Modify permission

Add the path to the whole environment variable



```
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/
local/games:/home/curry/tools/opt/FriendlyARM/toolschain/4.5.1/bin:/home/curry/
tools/1_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/
bin"
```

Compile the application with cross compiler

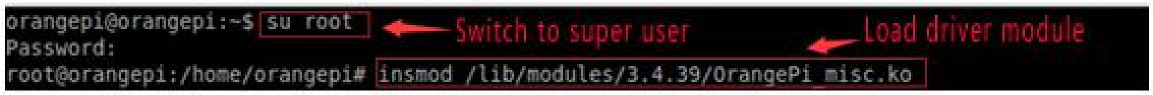
$ arm-linux-gnueabihf-gcc app.c –o aq
There will be an ap application generated in the directory, copy it to the development board file system(on the rootfs directory of /home/orangepi/)
$ cp aq /media/*/home/orangepi/

4. Running driver and application

Removed the SD card and inserted it into the development board and power on.

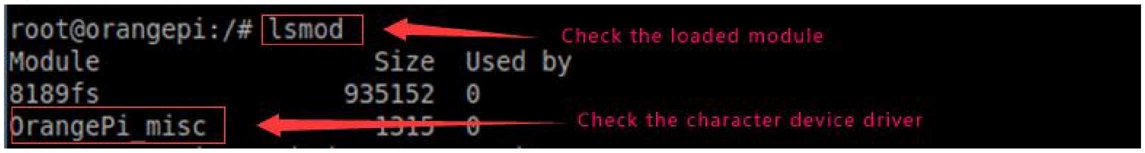
You need to switch to root users and load module driver module to the development board first.

$ insmod /lib/modules/orangepi.ko



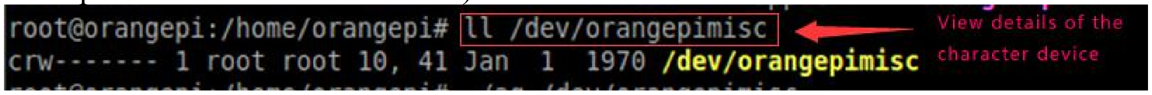
```
orangepi@orangepi:~$ su root
Password:
root@orangepi:/home/orangepi# insmod /lib/modules/3.4.39/OrangePi_misc.ko
```

$ lsmod To check whether it is loaded



```
root@orangepi:/# lsmod
Module                  Size  Used by
8189fs                935152  0
OrangePi_misc           1315  0
```

$ ll /dev/orangepimisc(Miscellaneous equipment automatically generated device files, the specific look at the driver code)



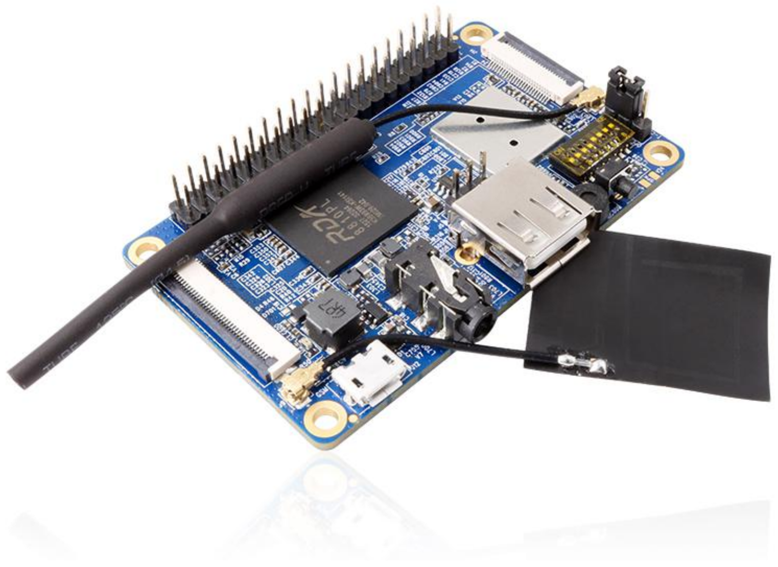
```
root@orangepi:/home/orangepi# ll /dev/orangepimisc
crw------- 1 root root 10, 41 Jan  1  1970 /dev/orangepimisc
```

Executive application (note the use of the application, check the code for specify)

$ ./aq /dev/orangepimisc

V. Using Debug tools on OrangePi

Hardware: Orange Pi development board*1, Card reader*1, TF card*1, power supply*1, TTL to USB cable*1



TTL to USB cable

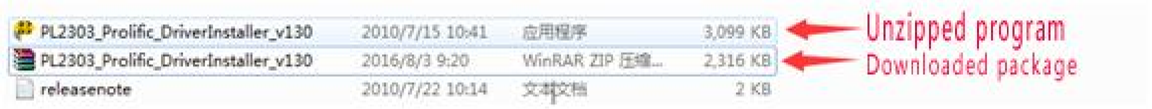
1. Operations on Windows

In order to get more debugging information in the project development process of using OrangePi, OrangePi default support for serial information debugging. For developers, you can simply get the serial port debugging information with the materials mentioned above. The host computer using different serial debugging tools are similar, basically can reference with the following manual for deployment. There are a lot of debugging tools for Windows platform, the most commonly used tool is putty. This section takes putty as an example to explain the deployment.

Android Baud rate set as 921600
Linux Baud rate set as 921600

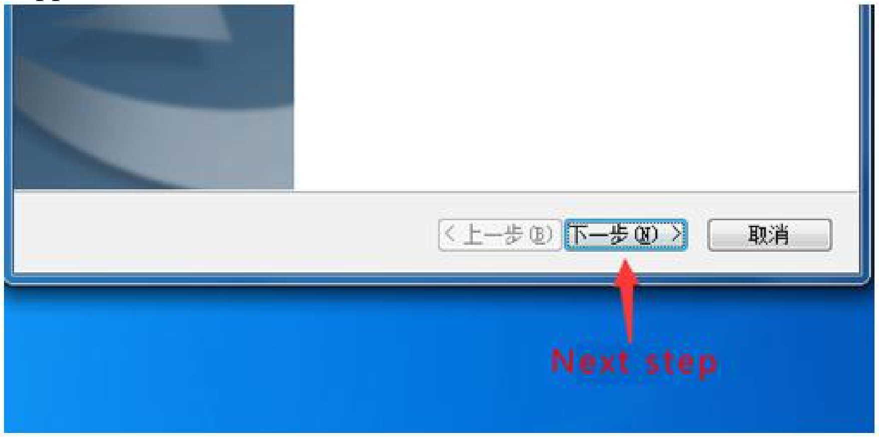
1) Install USB driver on Windows

- Download and unzip the latest version of driver: PL2303_Prolific_DriverInstaller_v130.zip

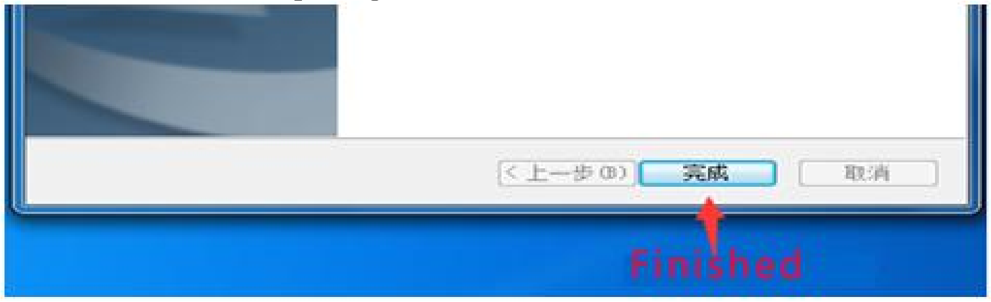


PL2303_Prolific_DriverInstaller_v130	2010/7/15 10:41	应用程序	3,099 KB
PL2303_Prolific_DriverInstaller_v130	2016/8/3 9:20	WinRAR ZIP 压缩...	2,316 KB
releasenote	2010/7/22 10:14	文本文档	2 KB

- Choose application installation as Administrator

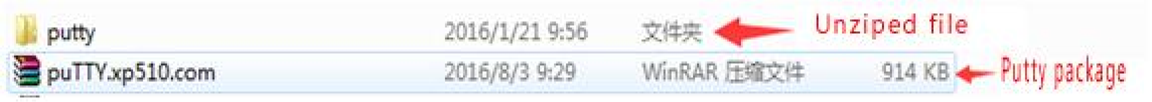


- Wait for installation completing



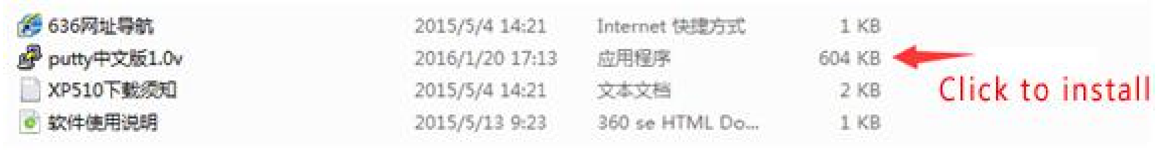
2) Install putty on Windows

- Download putty installation package



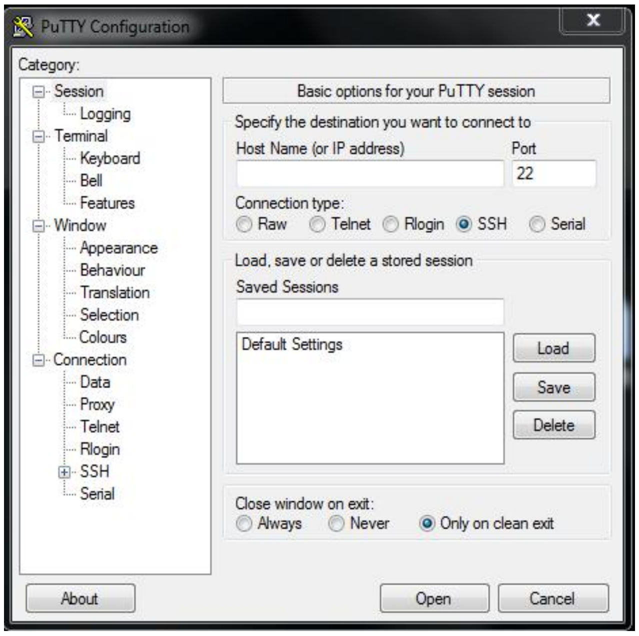
putty	2016/1/21 9:56	文件夹	
puTTY.xp510.com	2016/8/3 9:29	WinRAR 压缩文件	914 KB

- Unzip and install it



636网址导航	2015/5/4 14:21	Internet 快捷方式	1 KB
putty中文版1.0v	2016/1/20 17:13	应用程序	604 KB
XP510下載须知	2015/5/4 14:21	文本文档	2 KB
软件使用说明	2015/5/13 9:23	360 se HTML Do...	1 KB

- Open it after installed, shown as below:



PuTTY Configuration

Category: Session, Logging, Terminal, Keyboard, Bell, Features, Window, Appearance, Behaviour, Translation, Selection, Colours, Connection, Data, Proxy, Telnet, Rlogin, SSH, Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

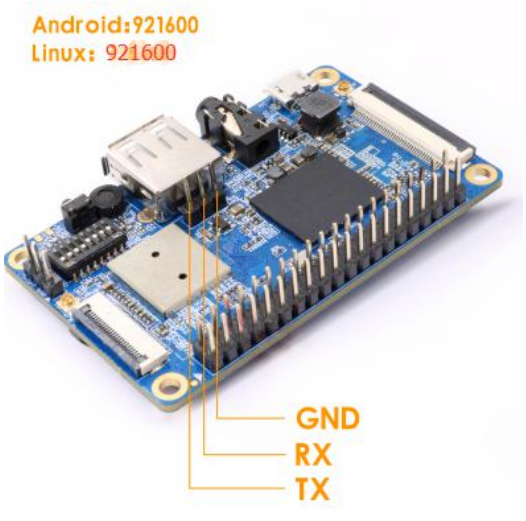
Saved Sessions

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

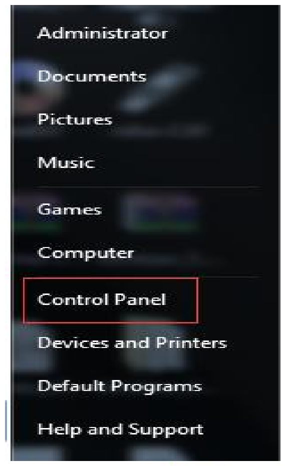
3) Connect method

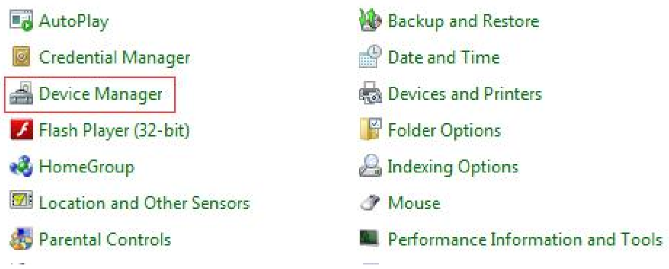
Use the TTL to the serial port cable, one end connected to OrangePi, the other end connected to PC



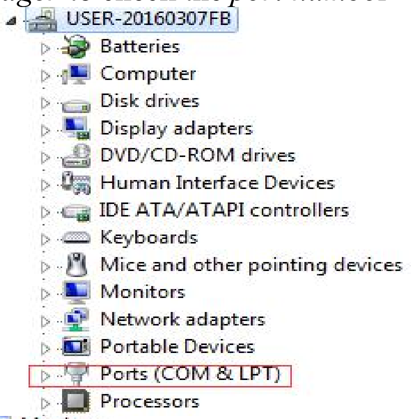
4) Equipment information acquisition

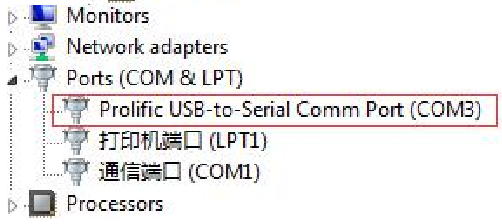
- Select *control panel on Start* menu



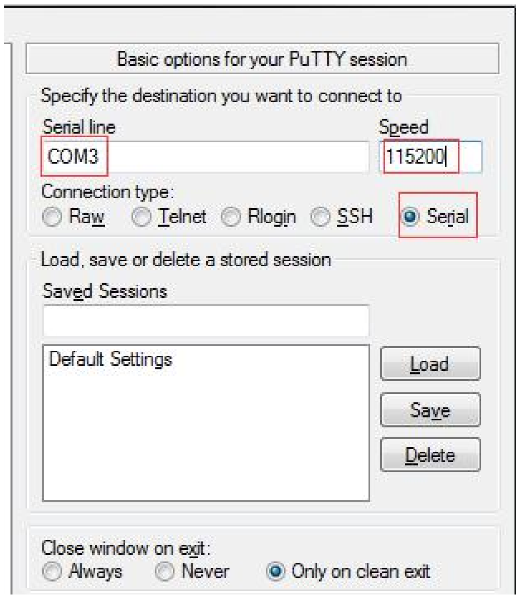


- Click on the *device manager* to check the *port number*





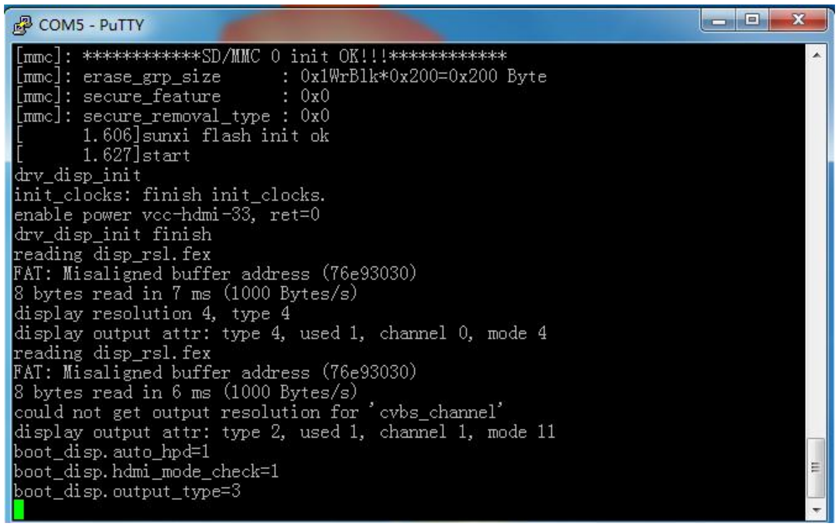
5) Putty configuration



Serial port should set to the corresponding port number (COM5), the speed should set to 115200

6) Start debug

Power Orange Pi on and boot it, the serial port will automatic print out debug log.



```
[mmc]: ************SD/MMC 0 init OK!!!************
[mmc]: erase_grp_size      : 0x1WrBlk*0x200=0x200 Byte
[mmc]: secure_feature      : 0x0
[mmc]: secure_removal_type : 0x0
[      1.606]sunxi flash init ok
[      1.627]start
drv_disp_init
init_clocks: finish init_clocks.
enable power vcc-hdmi-33, ret=0
drv_disp_init finish
reading disp_rsl.fex
FAT: Misaligned buffer address (76e93030)
8 bytes read in 7 ms (1000 Bytes/s)
display resolution 4, type 4
display output attr: type 4, used 1, channel 0, mode 4
reading disp_rsl.fex
FAT: Misaligned buffer address (76e93030)
8 bytes read in 6 ms (1000 Bytes/s)
could not get output resolution for 'cvbs_channel'
display output attr: type 2, used 1, channel 1, mode 11
boot_disp.auto_hpd=1
boot_disp.hdmi_mode_check=1
boot_disp.output_type=3
```

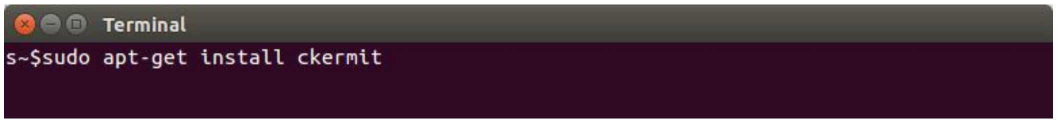
2. Operations on Linux

There are Minicom and Kermit serial debugging tools for Linux, this section will take Kermit as an example to have an illustrate.

1) Install Kermit

- Install the Kermit by execute command:

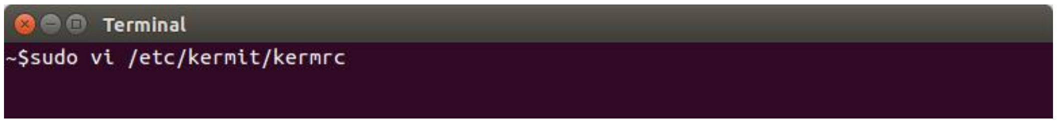
  $ sudo apt-get install ckermit



```
s~$sudo apt-get install ckermit
```

- Configurate Kermit

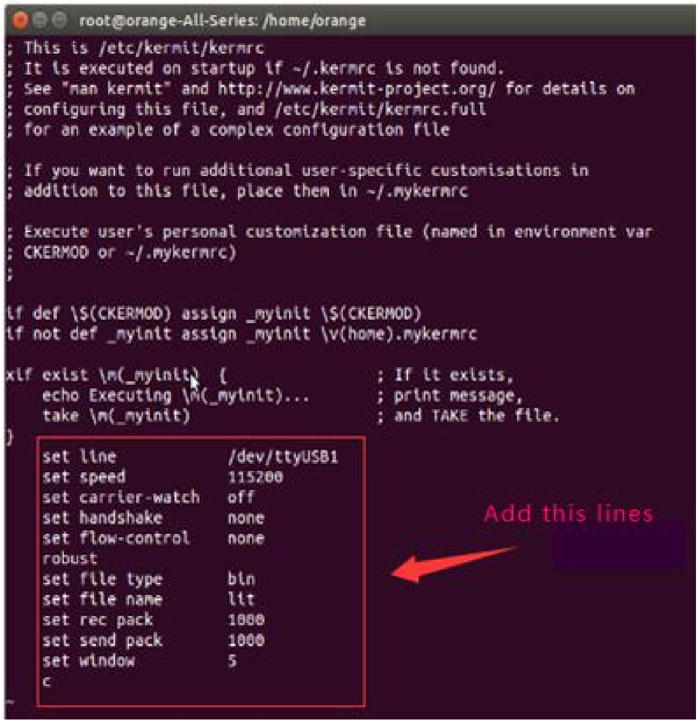
  $ sudo vi /etc/kermit/kermrc



```
~$sudo vi /etc/kermit/kermrc
```

- Add lines：

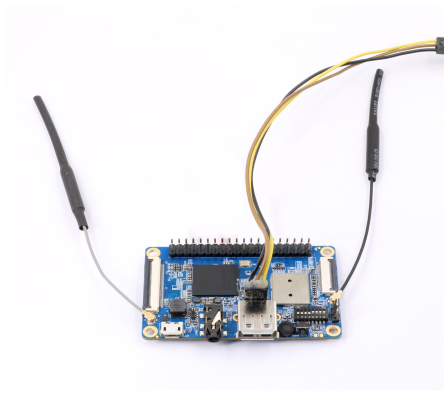
```
set line            /dev/ttyUSB1
set speed           921600
set carrier-watch   off
set handshake       none
set flow-control    none
robust
set file type       bin
set file name       lit
set rec pack        1000
set send pack       1000
set window          5
c
```



```
; This is /etc/kermit/kermrc
; It is executed on startup if ~/.kermrc is not found.
; See "man kermit" and http://www.kermit-project.org/ for details on
; configuring this file, and /etc/kermit/kermrc.full
; for an example of a complex configuration file

; If you want to run additional user-specific customisations in
; addition to this file, place them in ~/.mykermrc

; Execute user's personal customization file (named in environment var
; CKERMOD or ~/.mykermrc)
;

if def \$(CKERMOD) assign _myinit \$(CKERMOD)
if not def _myinit assign _myinit \v(home).mykermrc

xif exist \m(_myinit) {                 ; If it exists,
    echo Executing \m(_myinit)...       ; print message,
    take \m(_myinit)                    ; and TAKE the file.
}
    set line            /dev/ttyUSB1
    set speed           115200
    set carrier-watch   off
    set handshake       none
    set flow-control    none
    robust
    set file type       bin
    set file name       lit
    set rec pack        1000
    set send pack       1000
    set window          5
    c
```

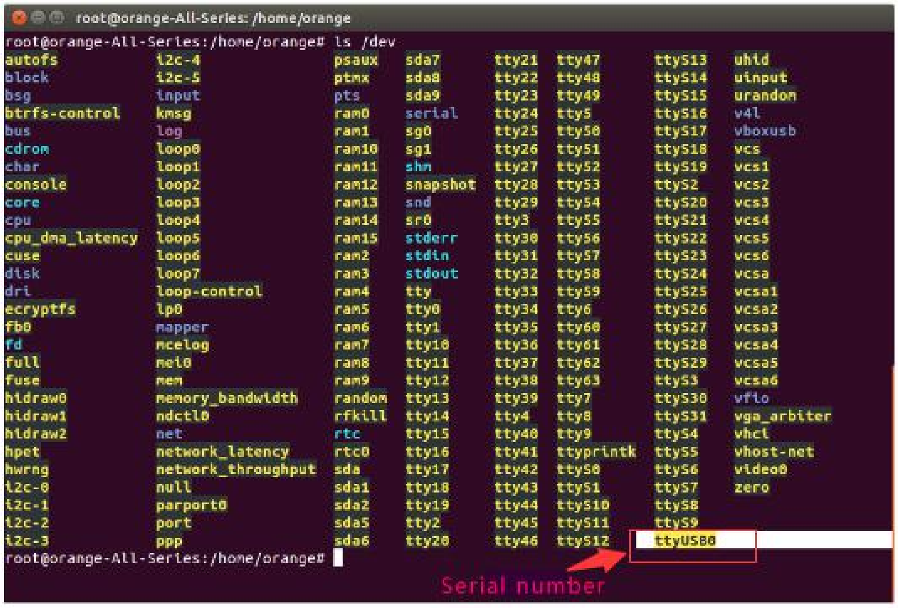
2) Connect method for debug

Use the TTL to the serial port cable, one end connected to OrangePi, the other end connected to PC



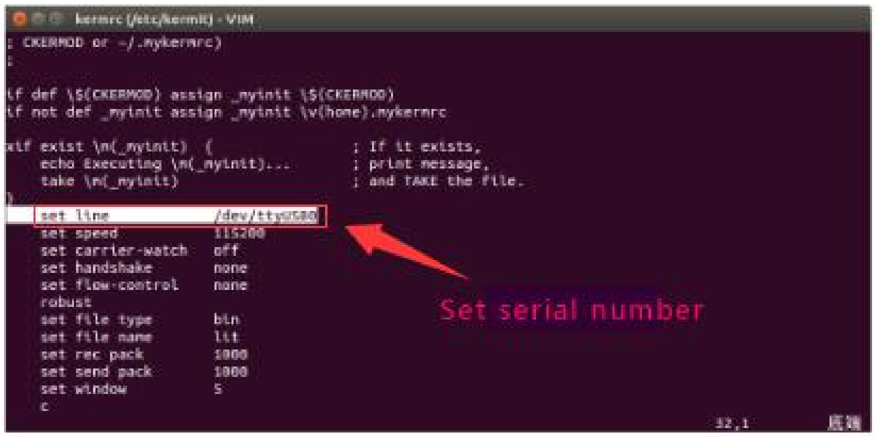
3) Equipment information acquisition

$ ls /dev/ (Input command in the PC terminal to check the device number of TTL to the serial cable)



```
root@orange-All-Series: /home/orange
root@orange-All-Series:/home/orange# ls /dev
autofs           i2c-4               psaux    sda7      tty21  tty47      ttyS13   uhid
block            i2c-5               ptmx     sda8      tty22  tty48      ttyS14   uinput
bsg              input               pts      sda9      tty23  tty49      ttyS15   urandom
btrfs-control    kmsg                ram0     serial    tty24  tty5       ttyS16   v4l
bus              log                 ram1     sg0       tty25  tty50      ttyS17   vboxusb
cdrom            loop0               ram10    sg1       tty26  tty51      ttyS18   vcs
char             loop1               ram11    shm       tty27  tty52      ttyS19   vcs1
console          loop2               ram12    snapshot  tty28  tty53      ttyS2    vcs2
core             loop3               ram13    snd       tty29  tty54      ttyS20   vcs3
cpu              loop4               ram14    sr0       tty3   tty55      ttyS21   vcs4
cpu_dma_latency  loop5               ram15    stderr    tty30  tty56      ttyS22   vcs5
cuse             loop6               ram2     stdin     tty31  tty57      ttyS23   vcs6
disk             loop7               ram3     stdout    tty32  tty58      ttyS24   vcsa
dri              loop-control        ram4     tty       tty33  tty59      ttyS25   vcsa1
ecryptfs         lp0                 ram5     tty0      tty34  tty6       ttyS26   vcsa2
fb0              mapper              ram6     tty1      tty35  tty60      ttyS27   vcsa3
fd               mcelog              ram7     tty10     tty36  tty61      ttyS28   vcsa4
full             mei0                ram8     tty11     tty37  tty62      ttyS29   vcsa5
fuse             mem                 ram9     tty12     tty38  tty63      ttyS3    vcsa6
hidraw0          memory_bandwidth    random   tty13     tty39  tty7       ttyS30   vfio
hidraw1          ndctl0              rfkill   tty14     tty4   tty8       ttyS31   vga_arbiter
hidraw2          net                 rtc      tty15     tty40  tty9       ttyS4    vhci
hpet             network_latency     rtc0     tty16     tty41  ttyprintk  ttyS5    vhost-net
hwrng            network_throughput  sda      tty17     tty42  ttyS0      ttyS6    video0
i2c-0            null                sda1     tty18     tty43  ttyS1      ttyS7    zero
i2c-1            parport0            sda2     tty19     tty44  ttyS10     ttyS8
i2c-2            port                sda5     tty2      tty45  ttyS11     ttyS9
i2c-3            ppp                 sda6     tty20     tty46  ttyS12     ttyUSB0
root@orange-All-Series:/home/orange# 
```

- It can be seen from the figure that TTL to the serial port cable is identified as ttyUSB0, configure the /ect/kermit/kermitc file, update the serial port information.
  $ sudo vi /etc/kermit/kermitc
- Set the value of setline into /dev/ttyUSB0

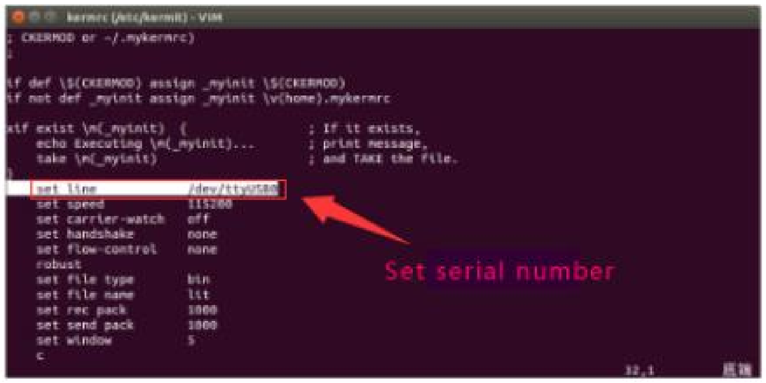


kermrc (/etc/kermit) - VIM

```
; CKERMOD or ~/.mykermrc)
;

if def \$(CKERMOD) assign _myinit \$(CKERMOD)
if not def _myinit assign _myinit \v(home).mykermrc

xif exist \m(_myinit)  {                      ; If it exists,
    echo Executing \m(_myinit)...             ; print message,
    take \m(_myinit)                          ; and TAKE the file.
}
    set line             /dev/ttyUSB0
    set speed            115200
    set carrier-watch    off
    set handshake        none
    set flow-control     none
    robust
    set file type        bin
    set file name        lit
    set rec pack         1000
    set send pack        1000
    set window           5
    c
                                                                  32,1          底端
```

4) Start debug

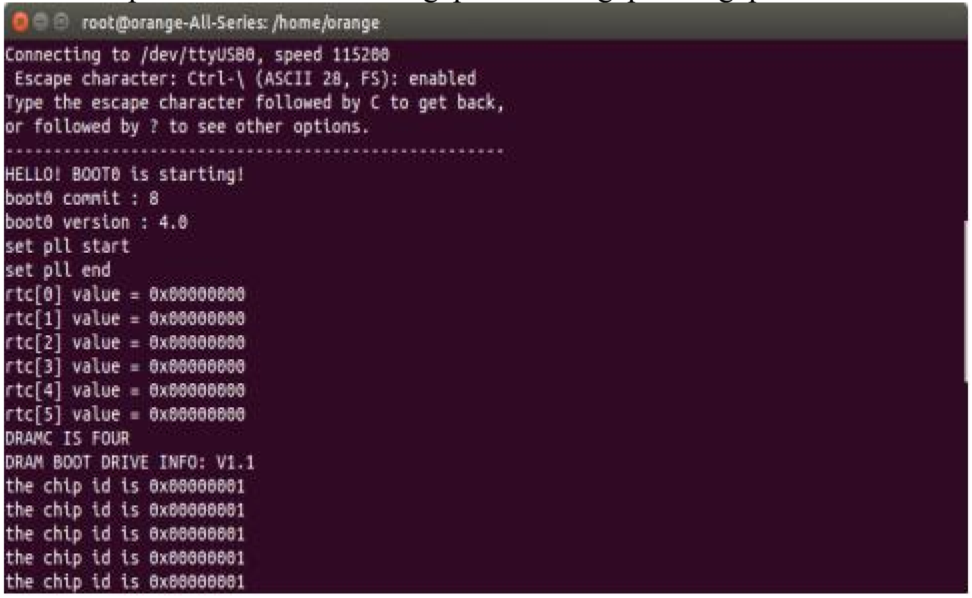
- Input command in the host computer terminal, enter the Kermit mode:
  $ sudo kermit –c



kermrc (/etc/kermit) - VIM

```
; CKERMOD or ~/.mykermrc)
;

if def \$(CKERMOD) assign _myinit \$(CKERMOD)
if not def _myinit assign _myinit \v(home).mykermrc

xif exist \m(_myinit)  {                      ; If it exists,
    echo Executing \m(_myinit)...             ; print message,
    take \m(_myinit)                          ; and TAKE the file.
}
    set line             /dev/ttyUSB0
    set speed            115200
    set carrier-watch    off
    set handshake        none
    set flow-control     none
    robust
    set file type        bin
    set file name        lit
    set rec pack         1000
    set send pack        1000
    set window           5
    c
                                                                  32,1          底端
```

- Power it on and boot Orange Pi, the serial port will automatic print debug log, the account and password ard root/orangepi and orangepi/orangepi



root@orange-All-Series: /home/orange

```
Connecting to /dev/ttyUSB0, speed 115200
 Escape character: Ctrl-\ (ASCII 28, FS): enabled
Type the escape character followed by C to get back,
or followed by ? to see other options.
----------------------------------------------------
HELLO! BOOT0 is starting!
boot0 commit : 8
boot0 version : 4.0
set pll start
set pll end
rtc[0] value = 0x00000000
rtc[1] value = 0x00000000
rtc[2] value = 0x00000000
rtc[3] value = 0x00000000
rtc[4] value = 0x00000000
rtc[5] value = 0x00000000
DRAMC IS FOUR
DRAM BOOT DRIVE INFO: V1.1
the chip id is 0x00000001
the chip id is 0x00000001
the chip id is 0x00000001
the chip id is 0x00000001
the chip id is 0x00000001
```