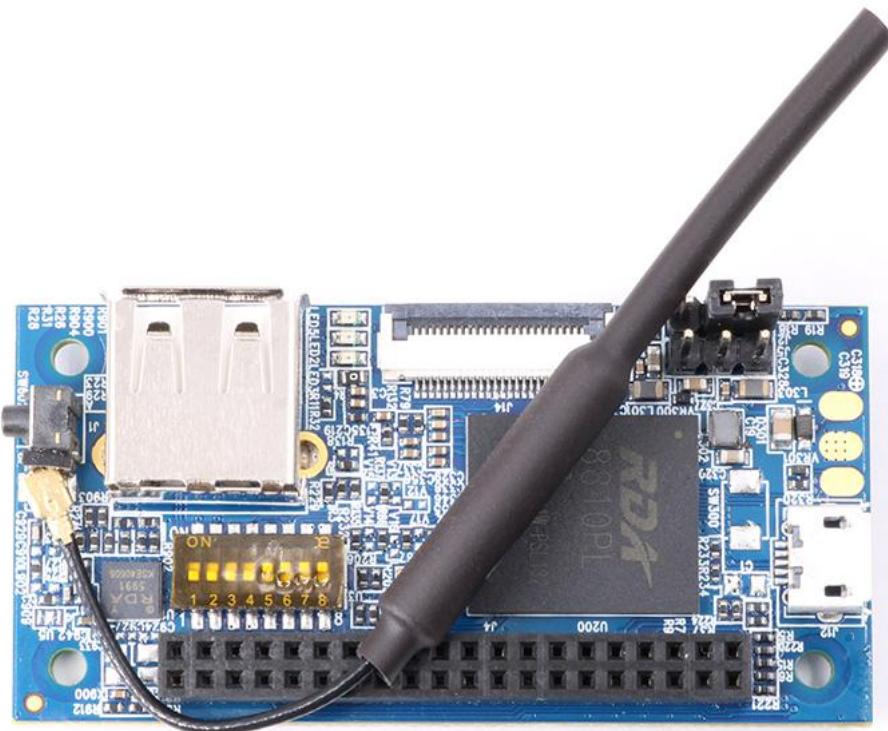




Orange Pi i96

User Manual





Content

I. Orange Pi Introduction.....	1
1. What is Orange Pi i96?	1
2. What can I do with Orange Pi i96?.....	1
3. Whom is it for?.....	1
4. Hardware Specification.....	1
5. GPIO Specifications.....	3
II. Using Method.....	4
1. Step 1: Prepare Accessories Needed.....	4
2. Step 2: Prepare a TF Card.....	4
3. Step 3: Start your Orange Pi.....	7
4. Step 4: Turn off your Orange Pi correctly.....	9
5. Initialize settings for your Linux system.....	9
6. Write Android into Nand.....	10
7. Android in no screen ADB mode.....	12
8. Universal software configuration.....	14
9. Usage of GPIO and WiringPi.....	17
III. Source Code Compilation of Android and Linux.....	20
1. Install JDK.....	20
2. Install Platform Supported Software.....	21
3. Download the Source Package and Unzip it.....	21
4. Android source code compiler.....	21
5. Compile Linux source Code.....	22
IV. Orange Pi Driver development.....	27
1. Device driver and application programming.....	27
2. Compile device driver.....	29
4. Running driver and application.....	31
V. Using Debug tools on OrangePi.....	32
1. Operations on Windows.....	32
2. Operations on Linux.....	36



I. Orange Pi Introduction

1. What is Orange Pi i96?

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 i96?

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 i96 is open source.

3. Whom is it for?

Orange Pi i96 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	RDA8810PL ARM Cortex-A5 32bit
GPU	Vivante GC860 3D GPU
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
Camera	MIPI CSI-2 Connector for camera sensor up to 2MP

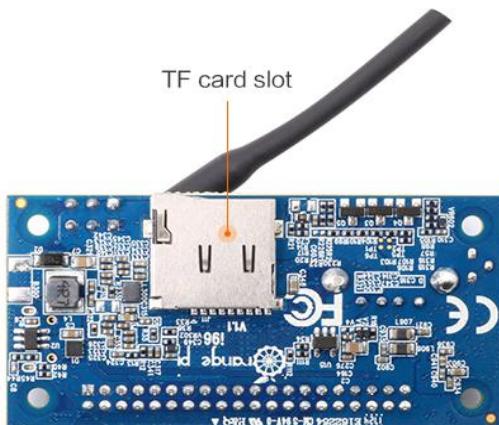
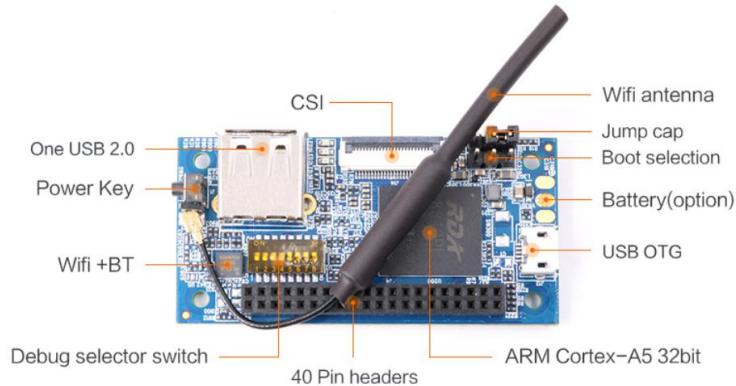


Power Source	USB OTG input can supply power Battery input can supply power (Optional)
USB 2.0 Ports	One USB 2.0 HOST, One USB 2.0 OTG
Buttons	Power Button(SW602)
Low-level peripherals	40 Pins Header
GPIO(1x3) pin	UART, ground.
LED	Power led
Supported OS	Android, Ubuntu, Debian, etc.

Interface definition

Product size	60mm × 30mm
Weight	30g
Orange Pi™ is a trademark of the Shenzhen Xunlong Software CO., Limited	

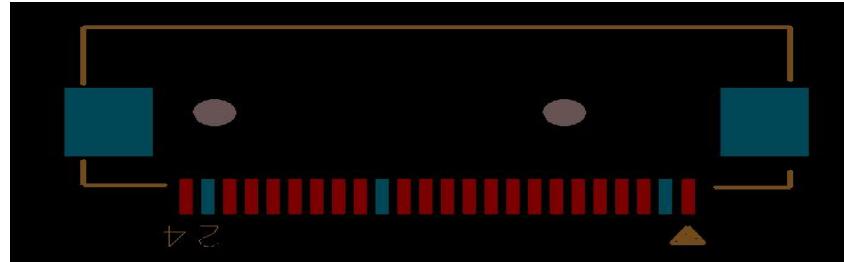
Interface instructions:





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 i96 is camera connector.



OrangePi i96-CS1

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 or bigger ; class 10. 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.



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:

- 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.

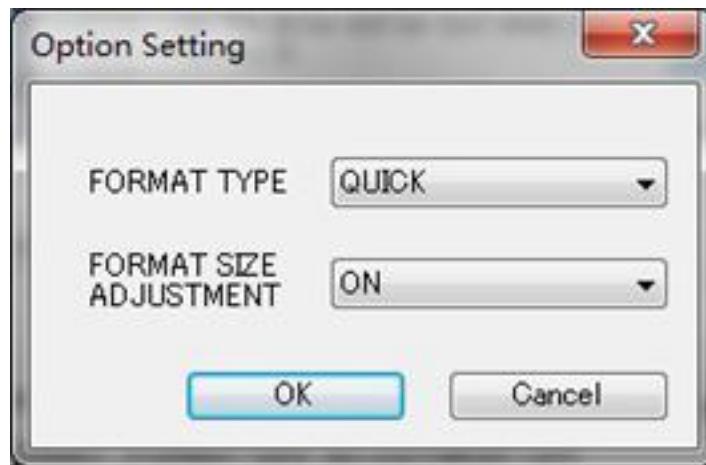


b. Formatting the TF card.

- 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)"



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

- v. Click the "Format" button.

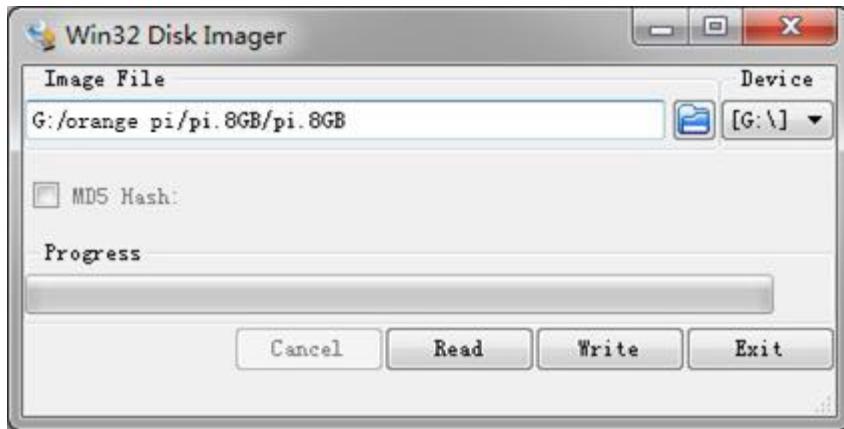
- c. Download the operating system image file from the download page, the page address is as follows: <http://www.orangepi.cn/downloadresourcescn/>
- d. Unzip the downloaded file (in addition to the Android system, this method can be used to burn to write, the Android system need another burn, the following 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.



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]

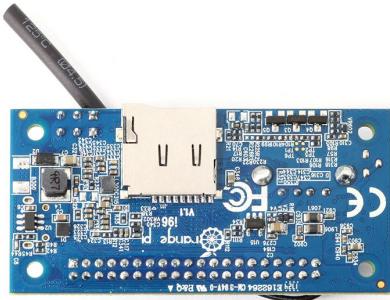


Here will be output some number which should be same as the image page line of "SHA-1"

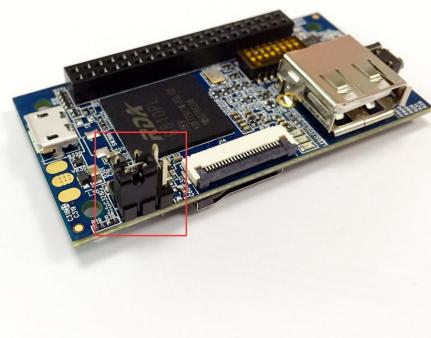
- iii Run umount /dev/sdxx command to uninstall all partitions in TF Card
- iv Run the command of sudo dd bs=4M if=[path]/[imagename] of=/dev/sdx to write image file and wait for it finished. You can run sudo pkill - USR1 - n - x dd command to check the procedure.

3. Step 3: Start your Orange Pi

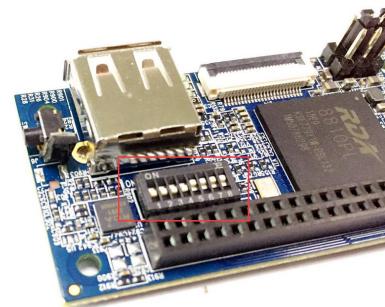
- Insert the TF card with written image into the TF card slot



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



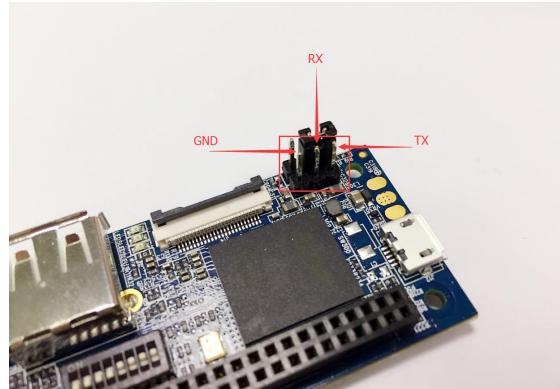
- Insert the keyboard or mouse into the USB port.



- Connect wifi antenna
- Connect to Camera(optional)



- 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 Orange Pi 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 your first time to use Linux on Orange Pi i96, like wifi setting, audio setting, user setting.

1) Wifi setting on serial port

After used serial to login system, entered the password the system will prompt you to use the orangepi_config tool to make some basic setting, including wifi setting. You could use the following command in the order line:

```
sudo orangepi_config  
> 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 i96 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 i96.

After getting the IP of Orange Pi i96, you could use SSH remote login in Linux PC or Windows PC. Command as following:

```
ssh orangepi@192.168.xxx.xxx
```

Password: orangepi



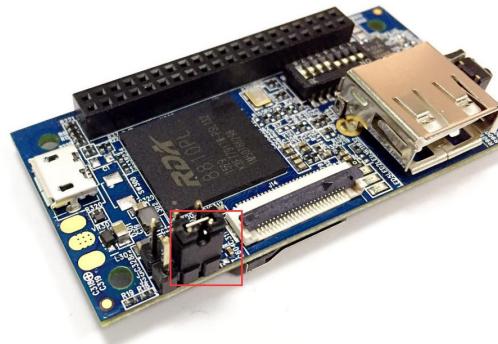
After enter the system via ssh, run the following command to connect to router:
sudo orangepi_config

6. Write Android into Nand

Orange Pi i96 is supported boot from Nand, and also supported update Android in Nand.

1) Boot Android from NAND

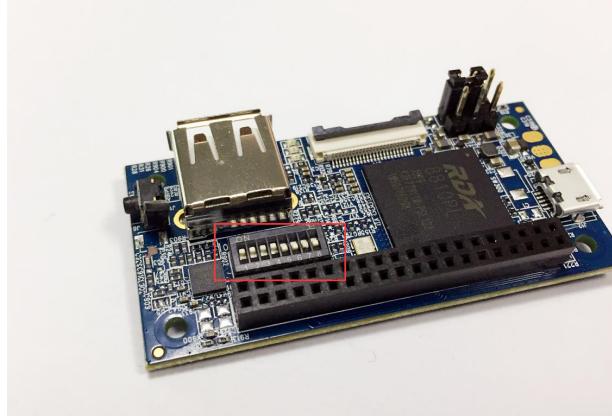
Switching the boot mode into NAND via short jumper cap.



Power it on, Orange Pi i96 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/>

The screenshot shows the Orange Pi official website's download resources page. At the top, there is a navigation bar with links for Home, News, Model, Resources (which is currently selected), partners, Forum, and Community. Below the navigation bar, there are sections for Orange Pi 2G-IoT and Orange Pi Zero Plus2 H3. Each section contains icons, names, update dates, and 'Download Now' buttons for Android-Tcard, Ubuntu Server, Android SDK source Code, Debian Desktop Jessie, and Ubuntu Desktop.

4) Install USB driver on Windows

Unzip the tool file, install the USB driver, here is the path:

`*/OrangePi_I96_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 link for Orange Pi i96 Nand version image:

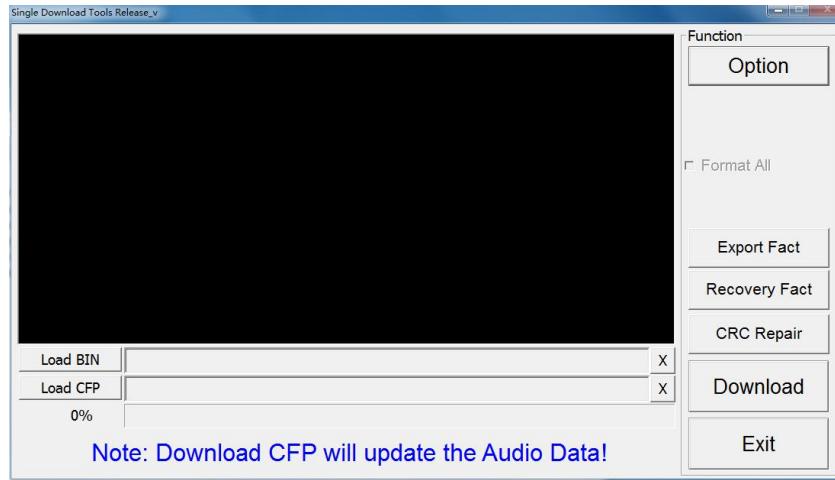
<http://www.orangepi.org/downloadresources/>

The screenshot shows the same download resources page as before, but with a red arrow pointing to the 'Android-Nand' download link for the Orange Pi 2G-IoT section. This indicates that the user should click on this link to download the Android Nand image.

6) Use writing tool

Use writing tool to write NAND:

`*/OrangePi_2G-IOT_Toolschain/OrangePi_2G-IOT_NandUpdate_Tools/OrangePi_2G-IOT_Update.e`

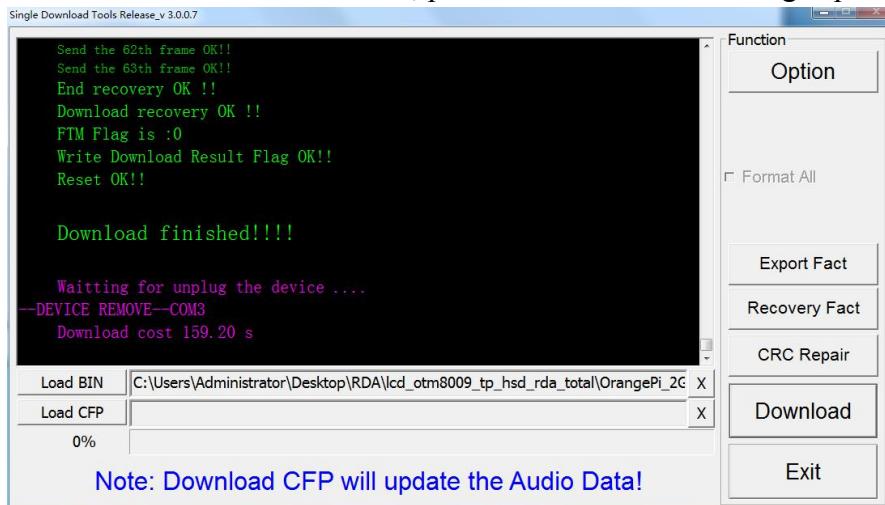


Click “load BIN” to import the image of NAND version into writing tool. After that, click Download button to download the image. Meanwhile, the tool is waiting for the download link of Orange Pi i96.

7) Download Image

Prepare an Android USB to DC cable, first connect to the OTG port of Orange Pi i96, 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.

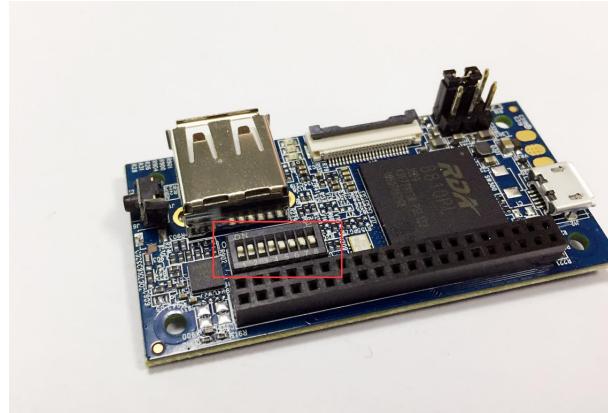
Note: If it could not download, please check the the shorting cap and switch.



7. Android in no screen ADB mode

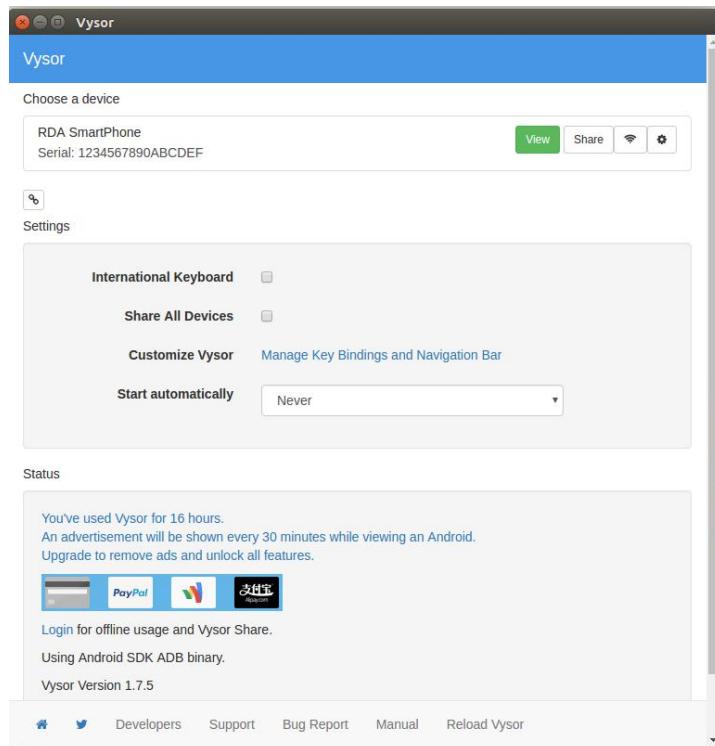


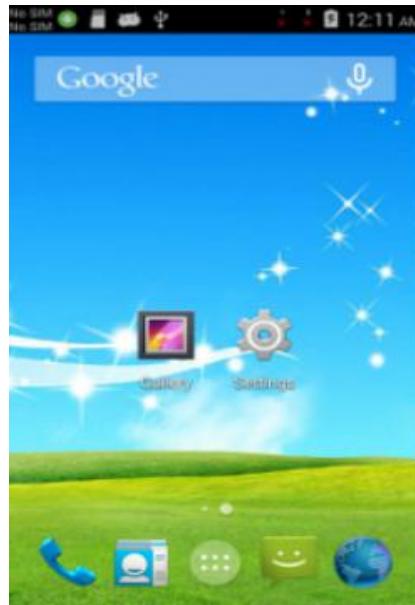
- ADB setting: Set the toggle switch into 1234 “UP”, 5678 “Down”, the system will switch into adb model, in this model, the USB is unable.



- Connect to the OTG port of Orange Pi I96 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 I96 via adb, you could refer to the adb debug method from the internet to enter into Orange Pi i96

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





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/ornagepi /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:

- Open the source file

```
$ sudo vi /etc/apt/sources.list
```

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

- 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).

iv 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.

iv 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,



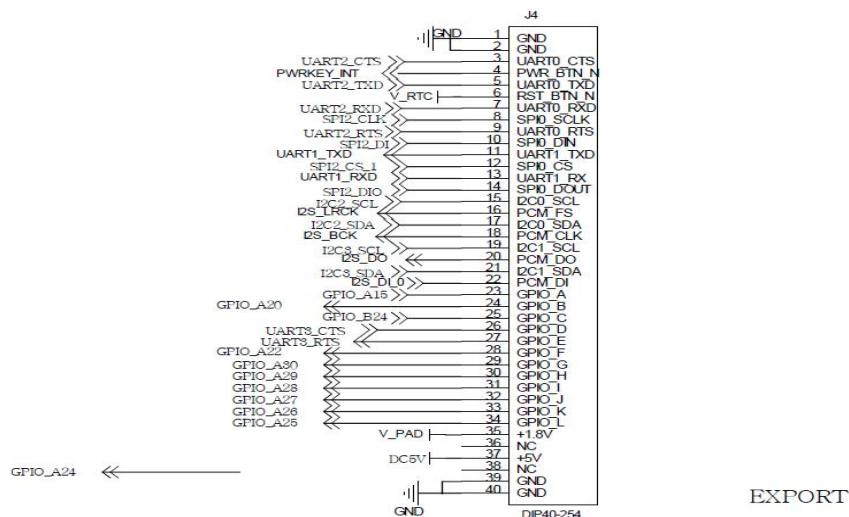
which is the 2 partition).

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 chance to make mistake. For now our official Lubuntu and Raspbian could not use it.

9. Usage of GPIO and WiringPi

OrangePi I96 could support WiringPi, you could have a try according to the introduction on this section. The following is 40 Pin of OrangePi.



- Download the latest WiringPi source code of Orange Pi i96

Please confirm that the Orange Pi i96 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
```

- 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
```

- Compile and install WiringPi

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



```
cd WiringOP/  
.build OrangePi_I96  
.build OrangePi_I96 install
```

● Test WiringPi with gpio command

You could use gpio command to test GPIO on 40pin on Orange Pi i96 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 be 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  
+---+---+---+---+---+---+---+---+---+---+---+  
| 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 |  
+---+---+---+---+---+---+---+---+---+---+---+
```

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 29 is corresponding to WiringPi pin number GPIO 25, explore the number 25 and set it into output mode.

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

```
orangepi# gpio unexport 26
```

iv Use "gpio exports" to check the current explored gpio. For example:

```
orangepi# gpio exports
```

v Use "gpio mode pin mode" command to configure wiringPi pin mode. For example:

Configure pin wiringPi 24 as output mode orangepi# gpio mode 24 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 26 as output pin: orangepi# gpio mode 26 out



- Write 0 on wiringPi 26 orangepi# gpio write 26 0
Write 1 wiringPi 26 orangepi# gpio write 26 1
- vii. Use "gpio read pin" command to read the value of input mode pin, for example:
Configure pin wiringPi 26 as input pin orangepi# gpio mode 26 in
read the value from wiringPi 26 orangepi# gpio read 26

```
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#
```

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

- 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(LED, LOW);       // Off
        delay(500);
    }
    return 0;
}
```

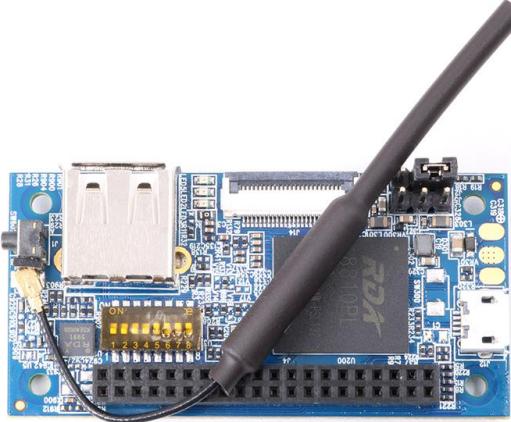
Usage of C library on wiringPi:

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.



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
```

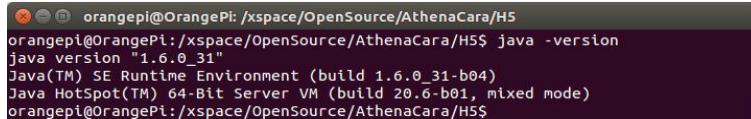
- Switch to java version and select version 1.6, use the following command:

```
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



A terminal window titled "orangePi@OrangePi: /xspace/OpenSource/AthenaCara/H5". The command "java -version" is run, and the output shows:

```
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 libcurl3-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 xsldproc 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 chapter to install first. After you install java 1.6 successfully, you could begin to compile Android source code.

- 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 i96 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

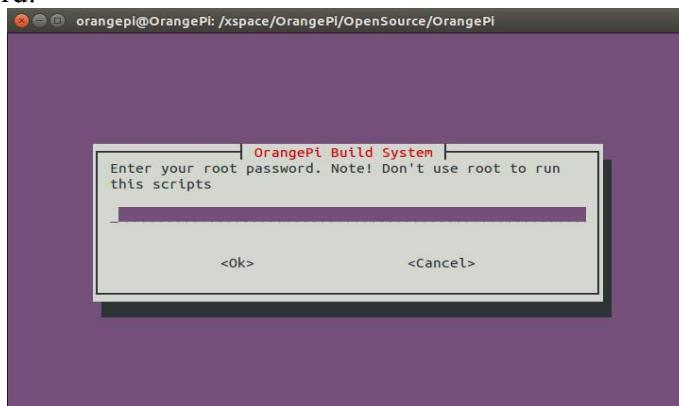
Use the following command to enter into source code directory:

```
cd */OrangePi
```

Execute the following script:

```
./Build_OrangePi.sh
```

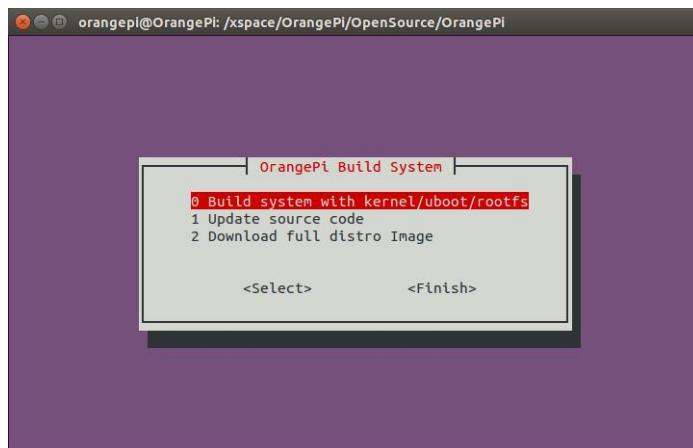
Input root password:



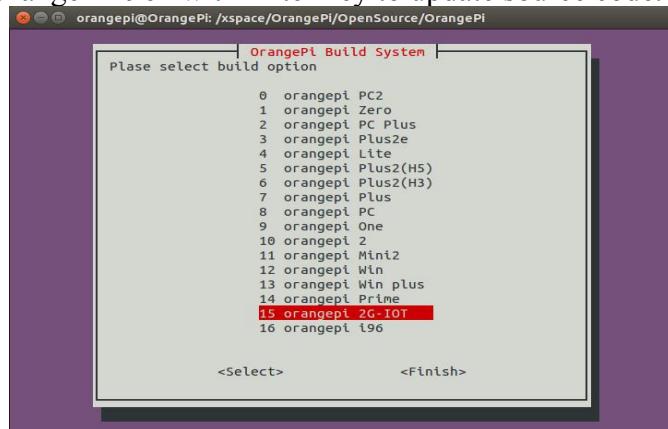
After root password recognize successful, enter inter main interface and use Enter key.



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



And then select “OrangePi I96” with Enter key to update source code.



It would take around 40minutes to update source code and corresponding scripts. After updated the source code, there will be generated a directory of OrangePi_i96. This directory contains both Linux source code and scripts:

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

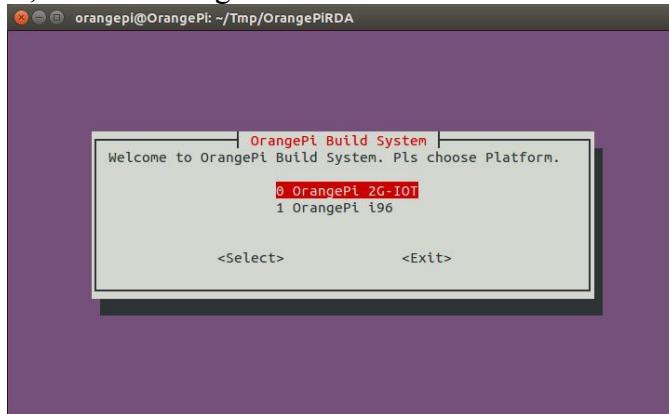
- Compile Linux



Execute the following command after enter into directory of OrangePiRDA:

`./build.sh`

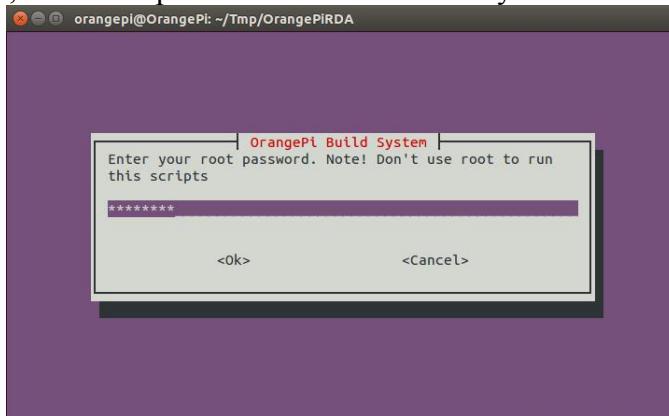
The script is an automatic script, you could select a corresponding board which you want to compile, here is “OrangePi I96”.



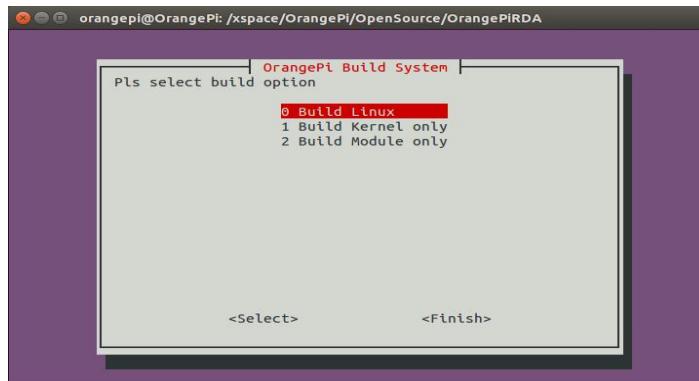
If it is the first time you run the script, the system would install development tool automatic to make sure the network is connecting.

```
orangepi@OrangePi: ~/Tmp/OrangePIRDA
build.sh kernel scripts toolchain
orangepi@OrangePi:~/Tmp/OrangePIRDA$ ./build.sh
[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 libtimezonestamp1
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.



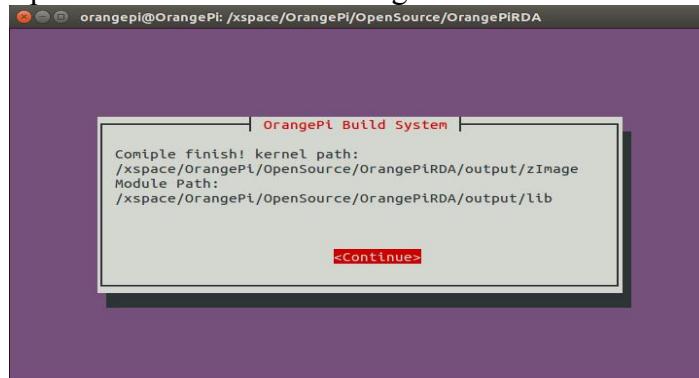
You will enter into the main interface after entering password, select what you are going to do:



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/alias.o
HOSTLD scripts/mod/nodpost.o
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.



● 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 partition 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.

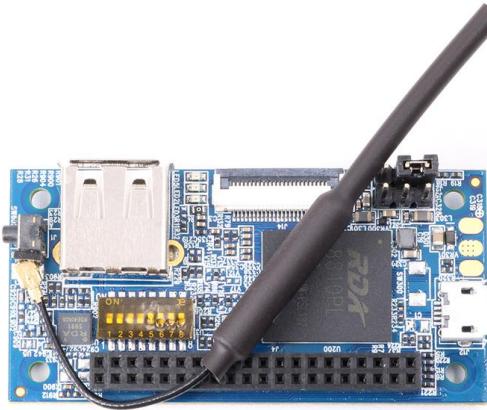




IV. Orange Pi Driver development

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:

```

43 obj-$(CONFIG_SPEAR13XX_PCIE_GADGET) += spear13xx_pcie_gadget.o
44 obj-$(CONFIG_VMWARE_BALLOON) += vmw_balloon.o
45 obj-$(CONFIG_ARM_CHARLCD) += arm-charlcd.o
46 obj-$(CONFIG_PCH_PHUB) += pch_phub.o
47 obj-y += ti-st/
48 obj-$(CONFIG_AB8500_PWM) += ab8500-pwm.o
49 obj-y += lis3lv02d/
50 obj-y += carma/
51 obj-$(CONFIG_USB_SWITCH_FSA9480) += fsa9480.o
52 obj-$(CONFIG_ALTERA_STAPL) += altera-stapl/
53 obj-$(CONFIG_MAX8997_MUIC) += max8997-muic.o
54 obj-$(CONFIG_WL127X_RFKILL) += wl127x-rfkill.o
55 obj-$(CONFIG_SENSORS_AK8975) += akm8975.o
56 obj-$(CONFIG_SUNXI_VIBRATOR) += sunxi-vibrator.o
57 obj-$(CONFIG_SUNXI_BROM_READ) += sunxi_brom_read.o
58 obj-$(CONFIG_NET) += rf_pm/
59 obj-$(CONFIG_ORANGEPI_MISC) += OrangePi_misc.o

```

Re-modify Makefile

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

```

Modify Kconfig

Back to the source code directory /OrangePi_i96

\$./build.sh

Please refer to last section about Linux source code compilation.

Update the new generated module file into Linux system.

It will generated a corresponding .ko file which is generated from the previous compilation of OrangePi_misc.c.

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.

The second partition is the rootfs partition





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
bug --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$
```

Downloaded package file

Unzip the downloaded file and enter 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
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$ ls
arm-linux-gnueabihf bin lib libexec share
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$
```

Unzip the package file

Enter to current directory to check files

Check the information after entering bin directory

```
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ cd bin/
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ ls
arm-linux-gnueabihf-bin lib libexec share
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ cd lib/
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$ ls
arm-linux-gnueabihf-addr2line  arm-linux-gnueabihf-dbg  arm-linux-gnueabihf-gcc  arm-linux-gnueabihf-ldd  arm-linux-gnueabihf-readelf
arm-linux-gnueabihf-ar    arm-linux-gnueabihf-elfedit  arm-linux-gnueabihf-gcov  arm-linux-gnueabihf-ld.gold   arm-linux-gnueabihf-size
arm-linux-gnueabihf-as    arm-linux-gnueabihf-g++    arm-linux-gnueabihf-gdb  arm-linux-gnueabihf-mn      arm-linux-gnueabihf-strip
arm-linux-gnueabihf-c++   arm-linux-gnueabihf-gcc     arm-linux-gnueabihf-gfortran arm-linux-gnueabihf-objcopy
arm-linux-gnueabihf-c++filt arm-linux-gnueabihf-gcc-4.9.1 arm-linux-gnueabihf-gprof  arm-linux-gnueabihf-objdump arm-linux-gnueabihf-strings
arm-linux-gnueabihf-cpp   arm-linux-gnueabihf-gcc-4.9.1 arm-linux-gnueabihf-ld     arm-linux-gnueabihf-pc
arm-linux-gnueabihf-cppconfig arm-linux-gnueabihf-gcc-4.9.1 arm-linux-gnueabihf-ld.MD  arm-linux-gnueabihf-pc-config
curry@curry:~/tools/1_arm-linux-gnueabihf-gcc$
```

Find out the tool for compiling

`pwd` shows the path and export it into the whole project



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

Indicate the path
Environment variables

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

```
root@curry:/home/curry/tools/i_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/i_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin# chmod 777 /etc/environment
root@curry:/home/curry/tools/i_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/i_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/i_arm-linux-gnueabihf-gcc/gcc-linaro-arm-linux-gnueabihf-4.9-2014.07_linux/bin"
```

Add path

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:                                     ← Switch to super user
root@orangepi:/home/orangepi# insmod /lib/modules/3.4.39/OrangePi misc.ko
```

Load driver module

\$ lsmod To check whether it is loaded

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

Check the loaded module
Check the character device driver

\$ 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
```

View details of the character device

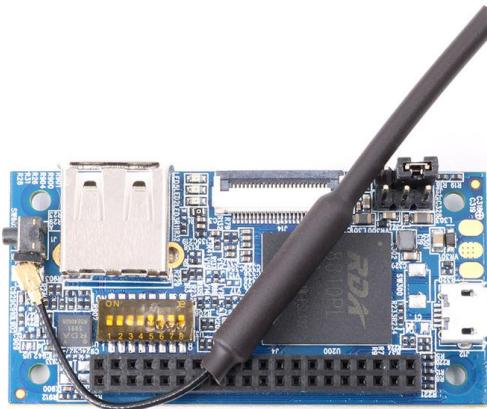
Executive application (note the use of the application, check the code for specify)

\$./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

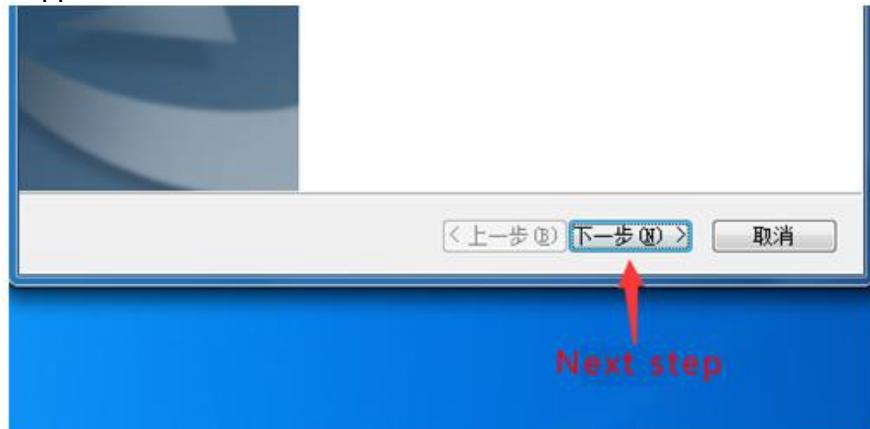


- Download and unzip the latest version of driver:
PL2303_Prolific_DriverInstaller_v130.zip

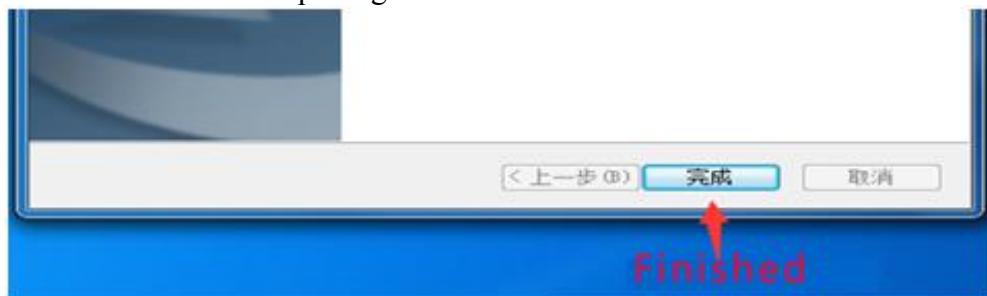
PL2303_Prolific_DriverInstaller_v130	2010/7/15 10:41	应用程序	3,099 KB
PL2303_Prolific_DriverInstaller_v130	2016/8/3 9:20	WinRAR ZIP 压缩文件	2,316 KB
releasenote	2010/7/22 10:14	文本文档	2 KB

Unzipped program
Downloaded package

- 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

Unzipped file

Putty package

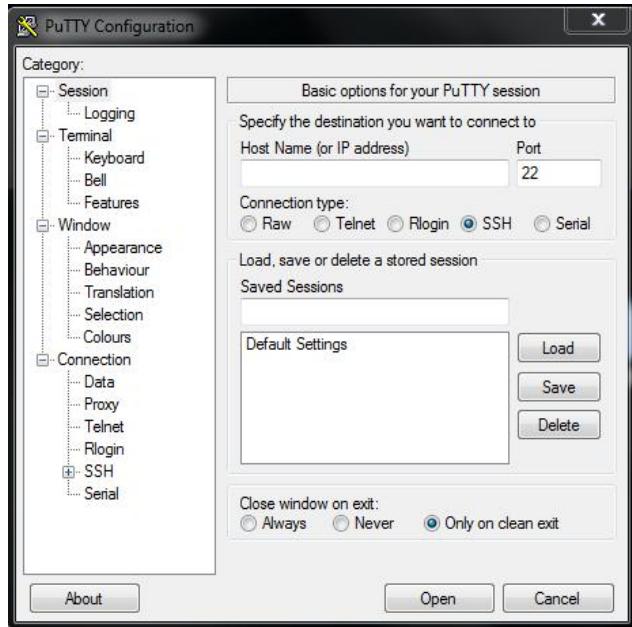
- Unzip and install it

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下载须知	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

Click to install

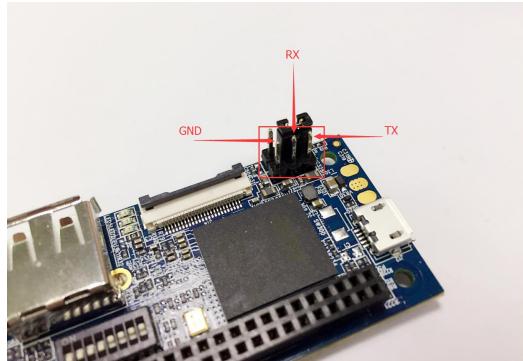


- Open it after installed, shown as below:



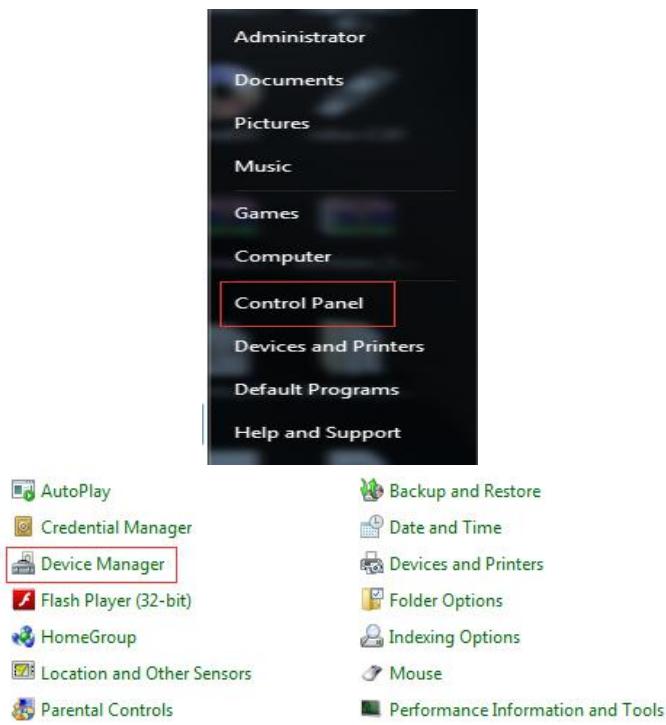
3) Connect method

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

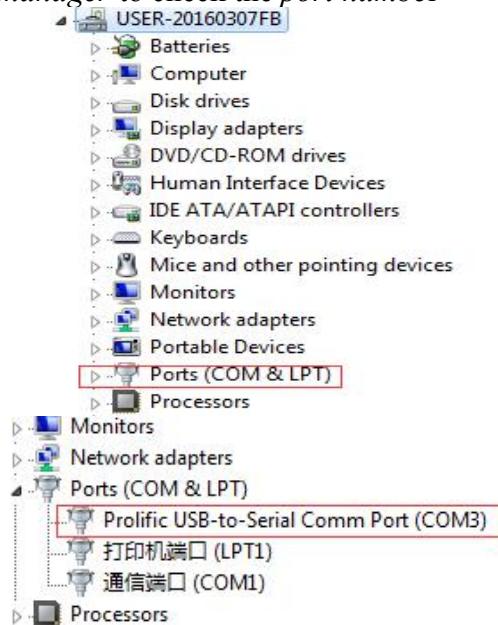


4) Equipment information acquisition

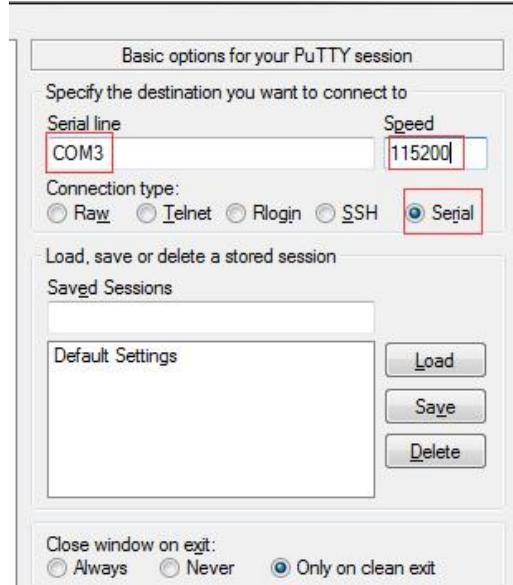
- Select *control panel* on *Start* menu



- 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 921600

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_hdcp=1
boot_disp.hDMI_mode_check=1
boot_disp.output_type=3
```

2. Operations on Linux

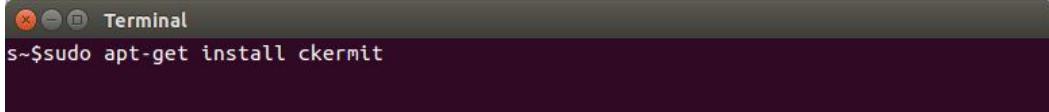
There are Minicom and Kermit serial debugging tools for Linux, this section will take Kermit as an example to have an illustrate.

1) Install Kermit

- Install the Kermit by execute command:



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



```
Terminal
$ sudo apt-get install ckermit
```

- Configure Kermit

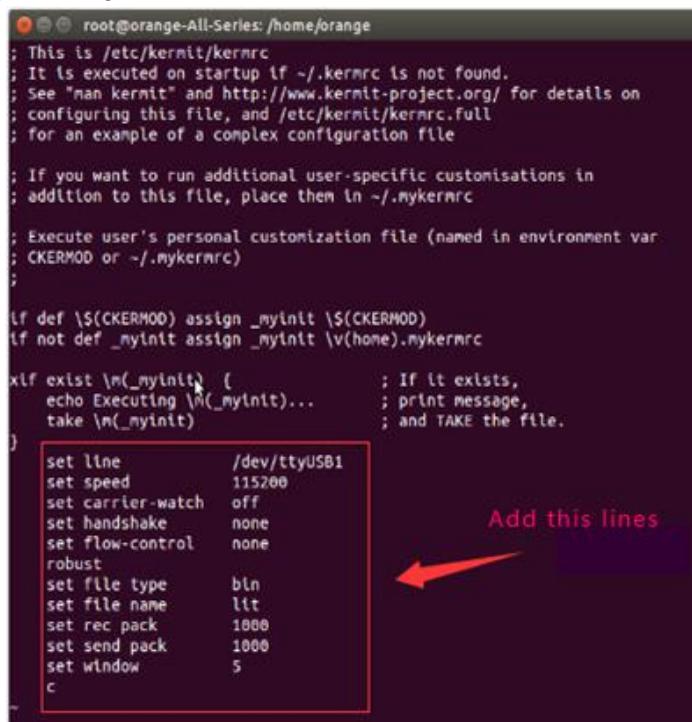
```
$ sudo vi /etc/kermit/kermrc
```



```
Terminal
~$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
```



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

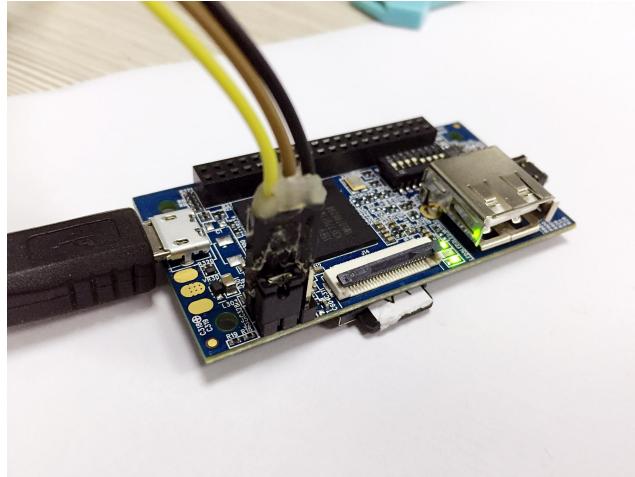
xlf exist ${_myinit} {
    echo Executing ${_myinit}...           ; If it exists,
    take ${_myinit}                      ; print message,
}                                         ; and TAKE the file.

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
```



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# ls /dev
root@orange-All-Series:/home/orange# ls /dev
autofs          i2c-4      psaux    sda7     tty21   tty47    tty513  uhid
block           i2c-5      ptmx     sda8     tty22   tty48    tty514  uinput
bsg             input      pts      sda9     tty23   tty49    tty515  urandom
btrfs-control  kmsg      ram0     serial   tty24   tty5     tty516  v4l
bus             log       ram1     sg0      tty25   tty50    tty517  vboxusb
cdrom          loop0     ram10    sgi     tty26   tty51    tty518  vcs
char            loop1     ram11    shm     tty27   tty52    tty519  vcs1
console         loop2     ram12    snapshot  tty28   tty53    tty520  vcs2
core            loop3     ram13    snd      tty29   tty54    tty521  vcs3
cpu             loop4     ram14    sr0      tty3    tty55    tty522  vcs4
cpu_dma_latency loop5     ram15    stderr   tty30   tty56    tty523  vcs5
cuse            loop6     ram2     stdin    tty31   tty57    tty524  vcsa
disk            loop7     ram3     stdout   tty32   tty58    tty525  vcsa1
dri             loop-control ram4     tty     tty33   tty59    tty526  vcsa2
ecryptfs        lp0      ram5     tty0    tty34   tty6     tty527  vcsa3
fb0             mapper   ram6     tty1    tty35   tty60    tty528  vcsa4
fd             mcelog   ram7     tty10   tty36   tty61    tty529  vcsa5
full            mem      ram8     tty11   tty37   tty62    tty530  vfat
fuse            mem      ram9     tty12   tty38   tty63    tty531  vga_arbiter
hidraw0         memory_bandwidth ram13    random  tty39   tty7     tty532  vhci
hidraw1         ndctl0   rfkill  tty14    tty4    tty8     tty533  vhost-net
hidraw2         net      rtc     tty15    tty40   tty9     tty534  video0
hpet            network_latency rtc0    tty16    tty41   ttyprintk  tty535  zero
hwrmg           network_throughput sda    tty17    tty42   tty50    tty536
i2c-0           null     sda1    tty18    tty43   tty51    tty537
i2c-1           parport0 sda2    tty19    tty44   tty510   tty538
i2c-2           port     sda5    tty2    tty45   tty511   tty539
i2c-3           ppp      sda6    tty20   tty46   tty512   ttyUSB0
root@orange-All-Series:/home/orange#
```

Serial number

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



```
vim kermrc (etc/kermit) - VIM
: CKERMOD or -./mykermrc
:
if def \$(CKERMOD) assign _myinit \$(CKERMOD)
if not def _myinit assign _myinit \v(home).nykermrc
:
atf exist \n(_myinit) (
    echo Executing \n(_myinit)...      : If it exists,
    take \n(_myinit)                 : print message,
    : 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 ltt
set rec pack 1800
set send pack 1800
set window   5
c
```

Set serial number

4) Start debug

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

```
vim kermrc (etc/kermit) - VIM
: CKERMOD or -./mykermrc
:
if def \$(CKERMOD) assign _myinit \$(CKERMOD)
if not def _myinit assign _myinit \v(home).nykermrc
:
atf exist \n(_myinit) (
    echo Executing \n(_myinit)...      : If it exists,
    take \n(_myinit)                 : print message,
    : 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 ltt
set rec pack 1800
set send pack 1800
set window   5
c
```

Set serial number

- Power it on and boot Orange Pi, the serial port will automatic print debug log, the account and password are 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
```