Banana Pi BPI-M64

Banana Pi BPI-M64 is a 64-bit quad-core mini single board computer. It features 2GB of RAM and 8GB eMMC. It also has onboard WiFi and BT. On the ports side, the BPI-M64 has 2 USB A 2.0 ports, 1 USB OTG port, 1 HDMI port, 1 audio jack, and lastly a DC power port. The processor is pin-to-pin comptialbe with R18, so it comes with two versions:M64 and M64-R18.

Also being a member of the Banana Pi family, the M64 is a big jump from the octa-core BPI-M3. This is because this Banana Pi BPI is named after its 64-bit SoC. BPI-M64 will be reserved for an upcoming board

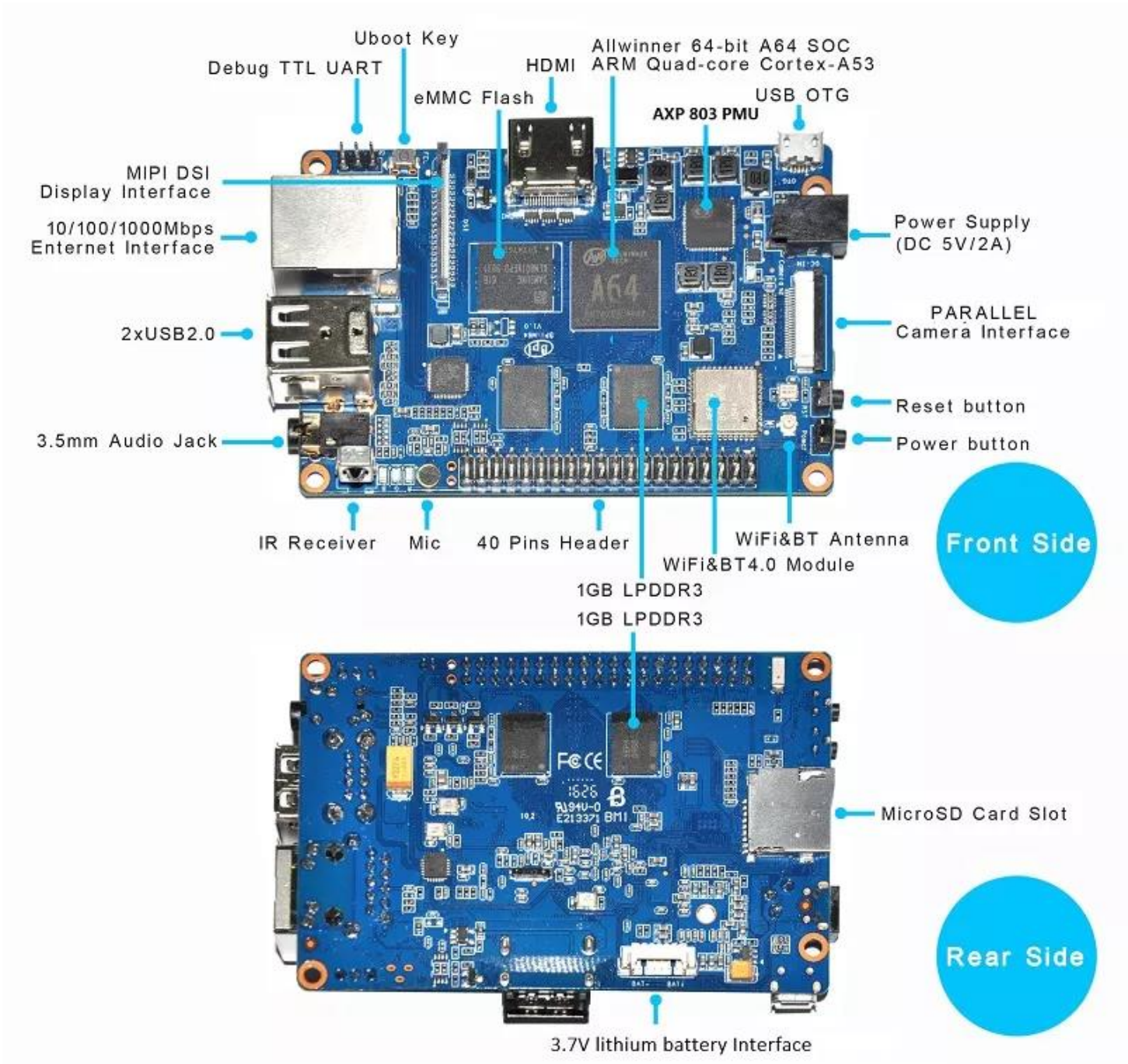
BPI-M64 is an open platform device, it is for anyone who wants to play and build with developer technology instead of simply using consumer technology. Backed by our community, starting a project and building servers is fun and rewarding. We welcome all companies, DIYers, and tech loving people within our community! Together, we can make a difference, we can discover our passions, inspire others, and build a practical project.

Key Features

- Allwinner A64 1.2 Ghz Quad-Core ARM Cortex A53 64-Bit Processor.
- 2 GB DDR3 SDRAM
- 8G EMMC
- 10/100/1000Mbps Ethernet
- WiFi (AP6212) & Bluetooth
- MIPI LCD interface
- Camera interface

How to burn image : Quick Start Banana pi SBC

Hardware interface



Hardware spec

HardWare Specification of Banana pi BPI-M64	
CPU	Allwinner 64 Bit Quad Core ARM Cortex A53 1.2 GHz CPU
GPU	Dual core Mali 400 MP2 GPU
Memory	2GB LPDDR3 (shared with GPU)

Storage	MicroSD slot with support for up to 256GB expansion and 8G eMMC flash with support for up to 64GB
Network	10/100/1000 Mbit/s Ethernet + Wi-Fi 802.11 b/g/n + Bluetooth 4.0
Video Input(s)	A CSI input connector allows for the connection of a designed camera module
Video Output(s)	1080p capable HDMI port and multi-channel audio output (NO H./X.265 capabilities)
Audio Input(s)	On board microphone
Audio Output(s)	3.5mm jack and HDMI
USB ports	USB 2.0 PORT (x2), USB OTG (x1)
Remote	IR Receiver
GPIO	40 Pin Header : GPIO (x28) and Power (+5V, +3.3V and GND). GPIO pins can be used for UART, I2C, SPI or PWM
Switches	Reset, Power and U-boot
LED	Power Status and 8P8C
Power Source	5 volt @2A via DC Power and/or Micro USB (OTG)
Size & Weight	92x60mm, 48g
OS	Android and Linux

GPIO PIN define

Banana Pi BPI-M64 has a 40-pin GPIO header . Following is the Banana Pi GPIO Pinout:

40 PIN GPIO of Banana pi BPI-M64			
GPIO Pin Name	Default Function	Function2 ： GPIO	Function3
CON2-P01	VCC-3V3		
CON2-P02	VCC-5V		
CON2-P03	TWI1-SDA	PH3	
CON2-P04	VCC-5V		
CON2-P05	TWI1-SCK	PH2	

CON2-P06	GND		
CON2-P07	PH6	PH6	
CON2-P08	UART2-TX	PB0	
CON2-P09	GND		
CON2-P10	UART2-RX	PB1	
CON2-P11	PH7	PH7	
CON2-P12	UART2-CTS	PB3	
CON2-P13	DMIC-CLK	PH10	
CON2-P14	GND		
CON2-P15	DMIC-DIN	PH11	
CON2-P16	UART2-RTS	PB2	
CON2-P17	VCC-3V3		
CON2-P18	PD4	PD4	
CON2-P19	SPI1-MOSI	PD2	UART4-TX
CON2-P20	GND		
CON2-P21	SPI1-MISO	PD3	UART4-RX
CON2-P22	PC0	PC0	
CON2-P23	SPI1-CLK	PD1	UART3-RX
CON2-P24	SPI1-CS	PD0	UART3-TX
CON2-P25	GND		
CON2-P26	PC2	PC2	
CON2-P27	PC4	PC4	
CON2-P28	PC3	PC3	
CON2-P29	PC7	PC7	
CON2-P30	GND		

CON2-P31	PCM0-BCLK	PB5	
CON2-P32	PCM0-DIN	PB7	
CON2-P33	PCM0-SYNC	PB4	
CON2-P34	GND		
CON2-P35	PCM0-DOUT	PB6	
CON2-P36	PL9	PL9	
CON2-P37	PL12	PL12	
CON2-P38	PL7	PL7	
CON2-P39	GND		
CON2-P40	PL8	PL8	

CSI Camera Connector specification:

The CSI Camera Connector is a 40-pin FPC connector which can connect external camera module with proper signal pin mappings. The pin definitions of the CSI interface are shown as below. This is marked on the Banana Pi board as “Camera”.

CSI camer PIN define of Banana pi BPI-M64		
CSI Pin Name	Default Function	Function2 ： GPIO
CN5-P01	NC	
CN5-P02	GND	
CN5-P03	CSI0-SDA	PE13
CN5-P04	CSI0-AVDD	
CN5-P05	CSI0-SCK	PE12
CN5-P06	CSI0-Reset	PE16
CN5-P07	CSI0-VSYNC	
CN5-P08	CSI0-PWDN	PE17
CN5-P09	CSI0-HSYNC	PE2
CN5-P10	CSI0-DVDD	

CN5-P11	CSI0-DOVDD	
CN5-P12	CSI0-D7	PE11
CN5-P13	CSI0-MCLK	PE1
CN5-P14	CSI0-D6	PE10
CN5-P15	GND	
CN5-P16	CSI0-D5	PE9
CN5-P17	CSI0-PCLK	PE0
CN5-P18	CSI0-D4	PE8
CN5-P19	CSI0-D0	PE4
CN5-P20	CSI0-D3	PE7
CN5-P21	CSI0-D1	PE5
CN5-P22	CSI0-D2	PE6
CN5-P23	GND	
CN5-P24	CSI0-AFVCC	

MIPI DSI (Display Serial Interface)

The display Connector is a 40-pin FPC connector which can connect external LCD panel (MIPI DSI) and touch screen (I2C) module as well. The pin definitions of this connector are shown as below. This is marked on the Banana Pi board as “DSI”.

MIPI DSI display PIN define of Banana pi BPI-M64		
DSI Pin Name	Default Function	Function2 ： GPIO
CN6-P01	VCC	
CN6-P02	IPSOUT	
CN6-P03	VCC	
CN6-P04	IPSOUT	
CN6-P05	GND	
CN6-P06	IPSOUT	

CN6-P07	GND	
CN6-P08	IPSOUT	
CN6-P09	NC	
CN6-P10	GND	
CN6-P11	NC	
CN6-P12	DSI-D0N	
CN6-P13	NC	
CN6-P14	DSI-D0P	
CN6-P15	NC	
CN6-P16	GND	
CN6-P17	TWI0-SDA	PH1
CN6-P18	DSI-D1N	
CN6-P19	TWI0-SCK	PH0
CN6-P20	DSI-D1P	
CN6-P21	CTP-INT	PH4
CN6-P22	GND	
CN6-P23	CTP-RST	PH8
CN6-P24	DSI-CKN	
CN6-P25	GND	
CN6-P26	DSI-CKP	
CN6-P27	LCD-BL-EN	PD5
CN6-P28	GND	
CN6-P29	LCD-RST	PD6
CN6-P30	DSI-D2N	
CN6-P31	LCD-PWR-EN	PD7

CN6-P32	DSI-D2P	
CN6-P33	GND	
CN6-P34	GND	
CN6-P35	LCD-PWM	PL10
CN6-P36	DSI-D3N	
CN6-P37	GND	
CN6-P38	DSI-D3P	
CN6-P39	NC	
CN6-P40	GND	

UART specification:

The header CON2 is the UART interface. For developers of Banana Pi, this is an easy way to get the UART console output to check the system status and log message.

Uart PIN define of Banana pi BPI-M64		
CON2 Pin Name	Default Function	GPIO
CON2 P03	UART0-TXD	PB8
CON2 P02	UART0-RXD	PB9
CON2 P01	GND	

Software

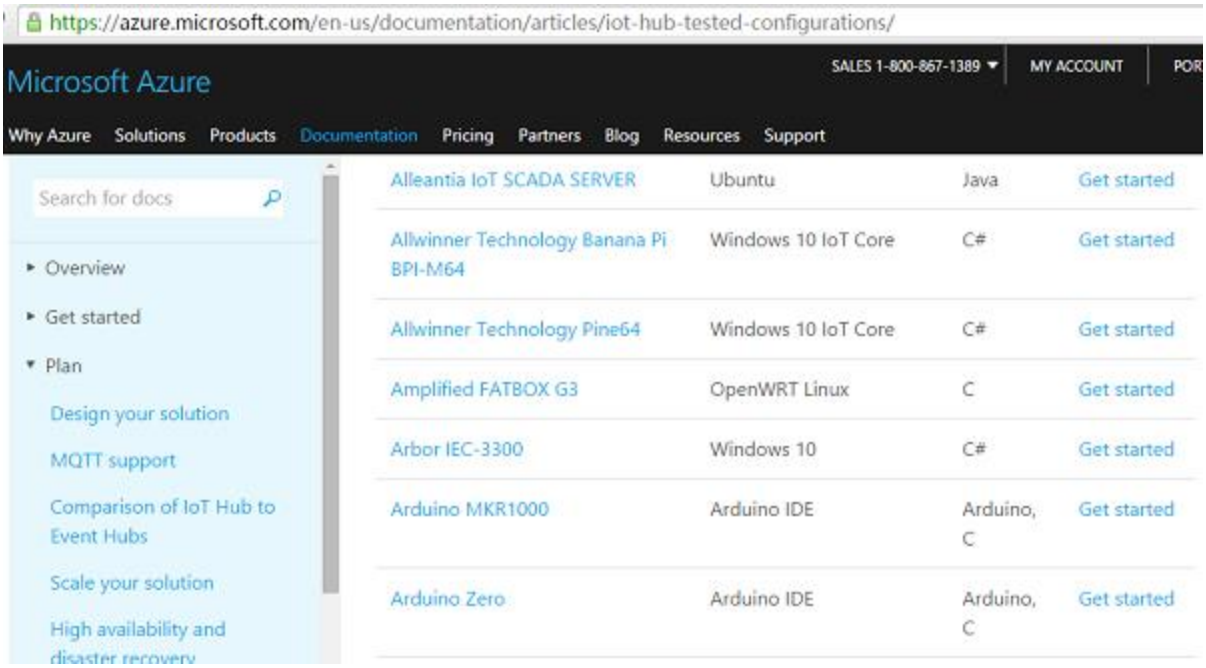
Development

Basic Development

Win 10 IoT

banana pi BPI-M64 IOT certifying pass by Microsoft:windows 10 iot core

BPI-M64 for Win10 IoT : https://catalog.azureiotsolutions.com/details?title=Allwinner_Banana_Pi_BPI_M64&source=all-devices-page



Development For Android

Install Android Image

Prepare

1. Prepare a USB-Serial cable, a MicroUSB cable and PC with Linux or WIndows 7/10

2. The USB-Serial cable is used for console debug and MicroUSB cable is used for Image download and ADB debug.

3. M64 board is only suport DC power supply bootup.

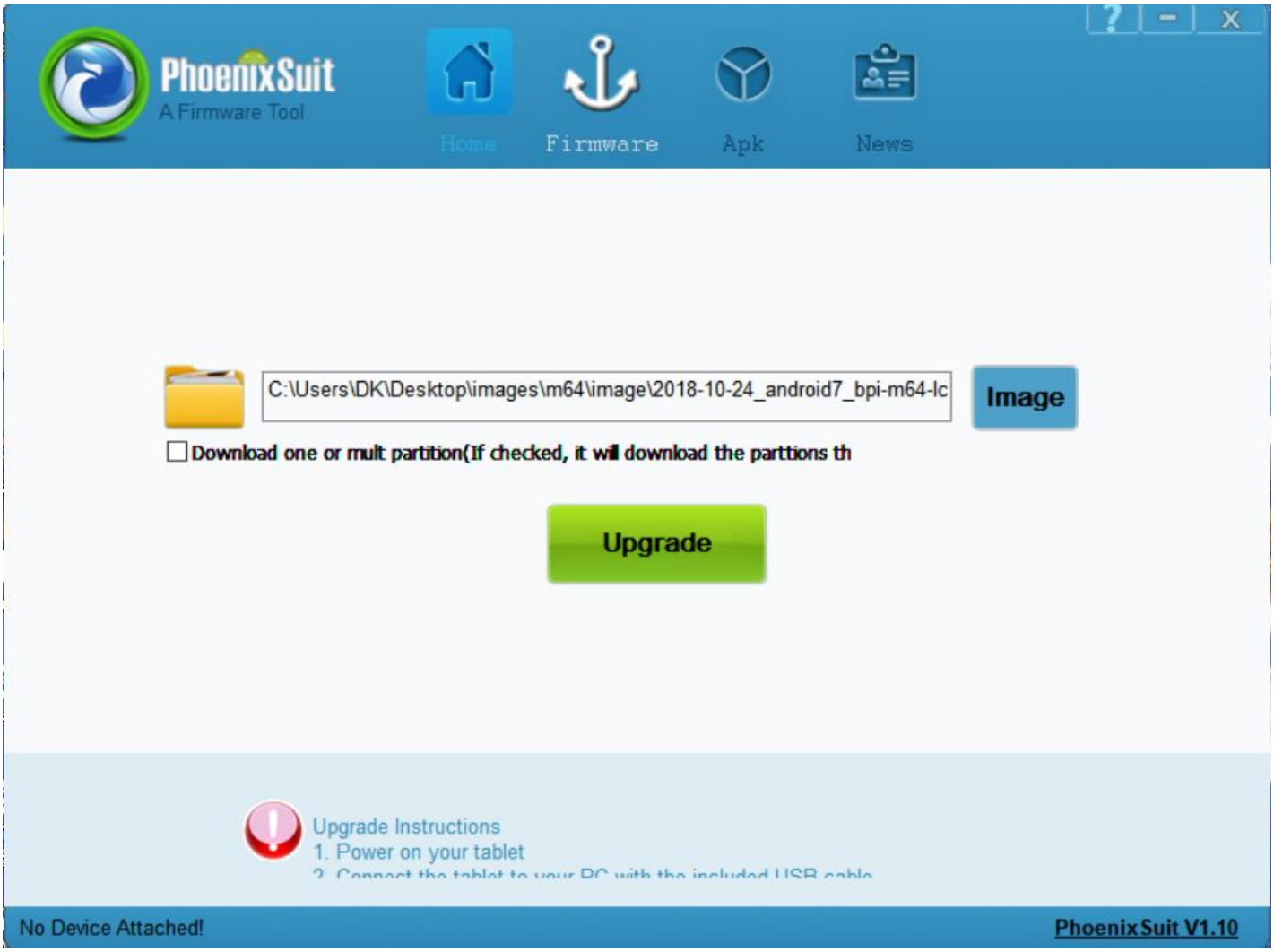
4. If you want insert a SDcard for Android storage using, and your SDcard was download Linux Image or any other allwinner bootable SDcard image, please format the SDcard start from block 0.

5. Download and Install Allwinner Image Download Tools, PhoenixSuit is for window and LiveSuit is for Linux

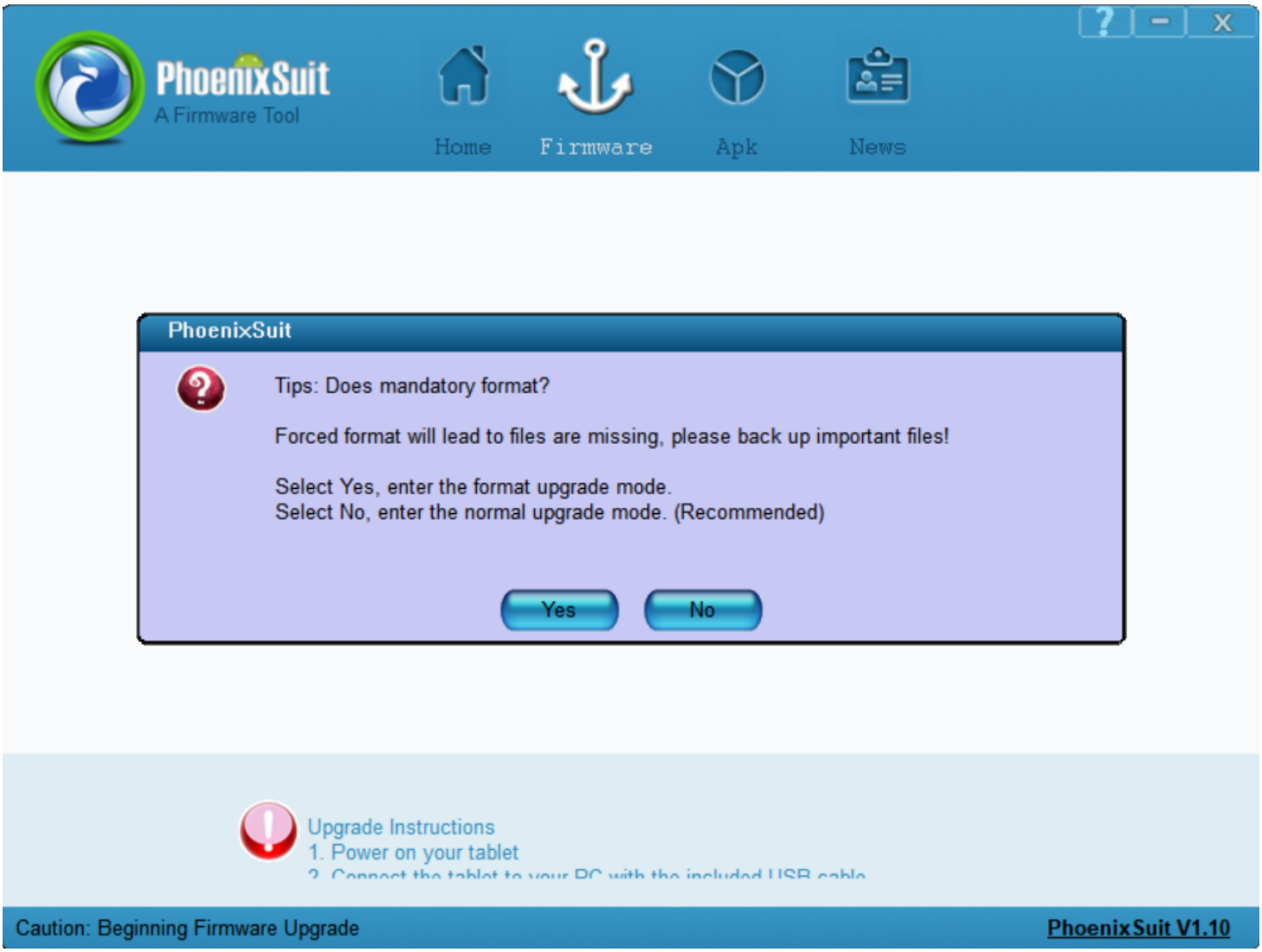
6. Download BPI latest Android Image

Install Image with PhoenixSuit on Windows

1. Open PhoenixSuit, click the Firmware icon to switching to firmware download panel, then click Image button and choose the Android Image file.

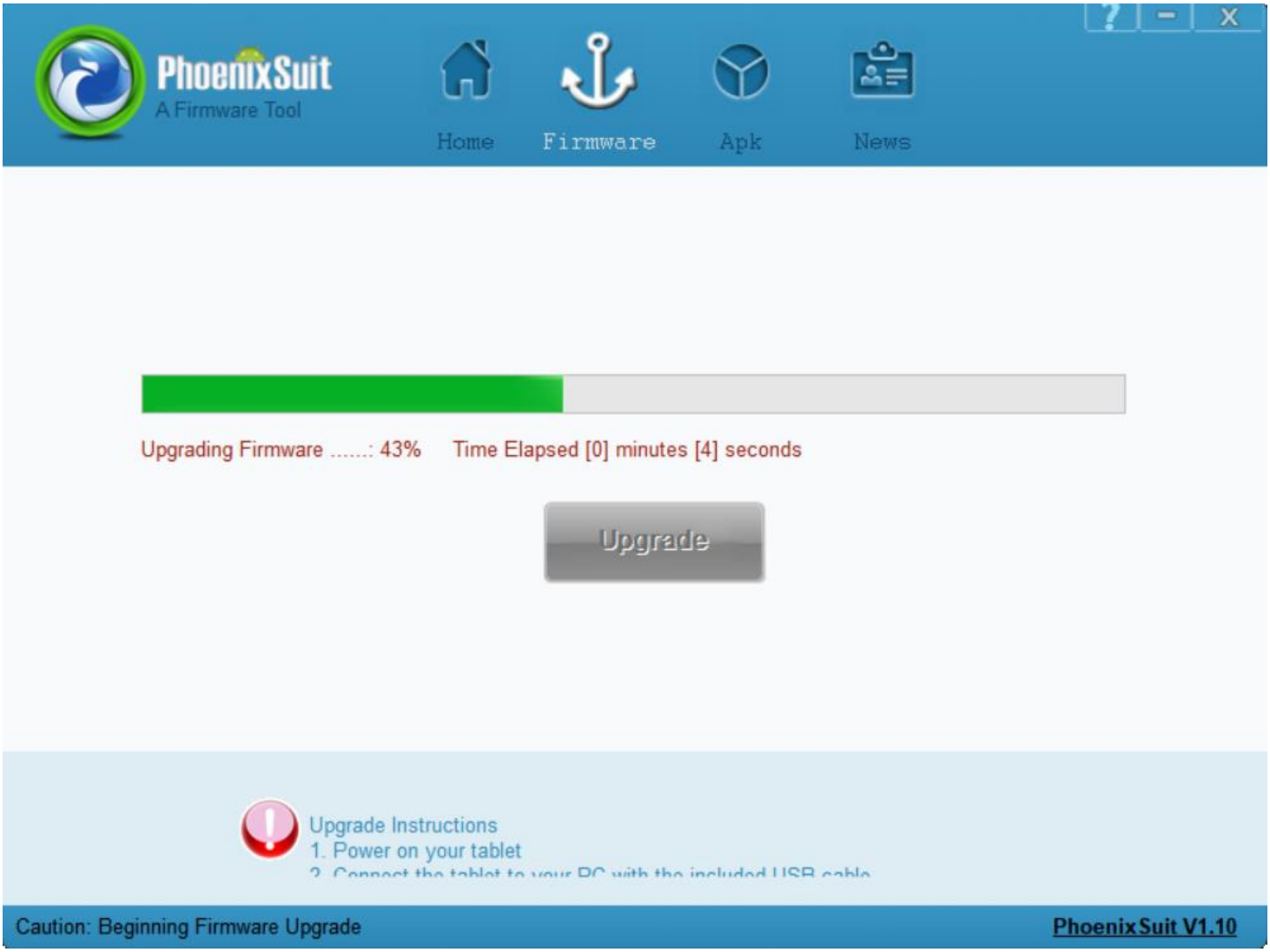


2. M64 board disconnect DC power, press and hold the uboot-key button(new uart debug pin), plugin mirco-usb cable to PC, popup a warning dialog.

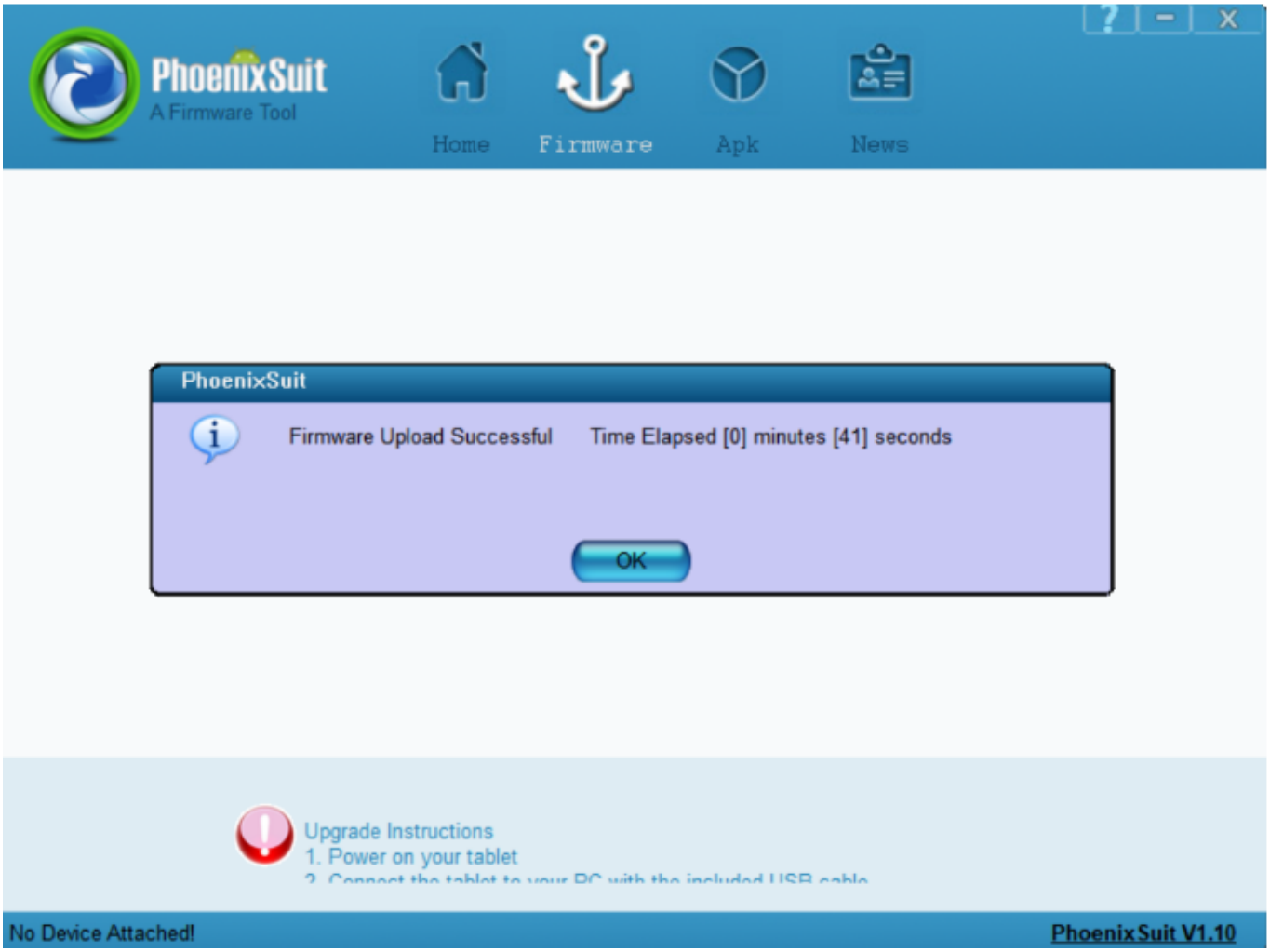


3 Press Yes to continue and popup another waring dialog, Press Yes to continue

4 Downloading

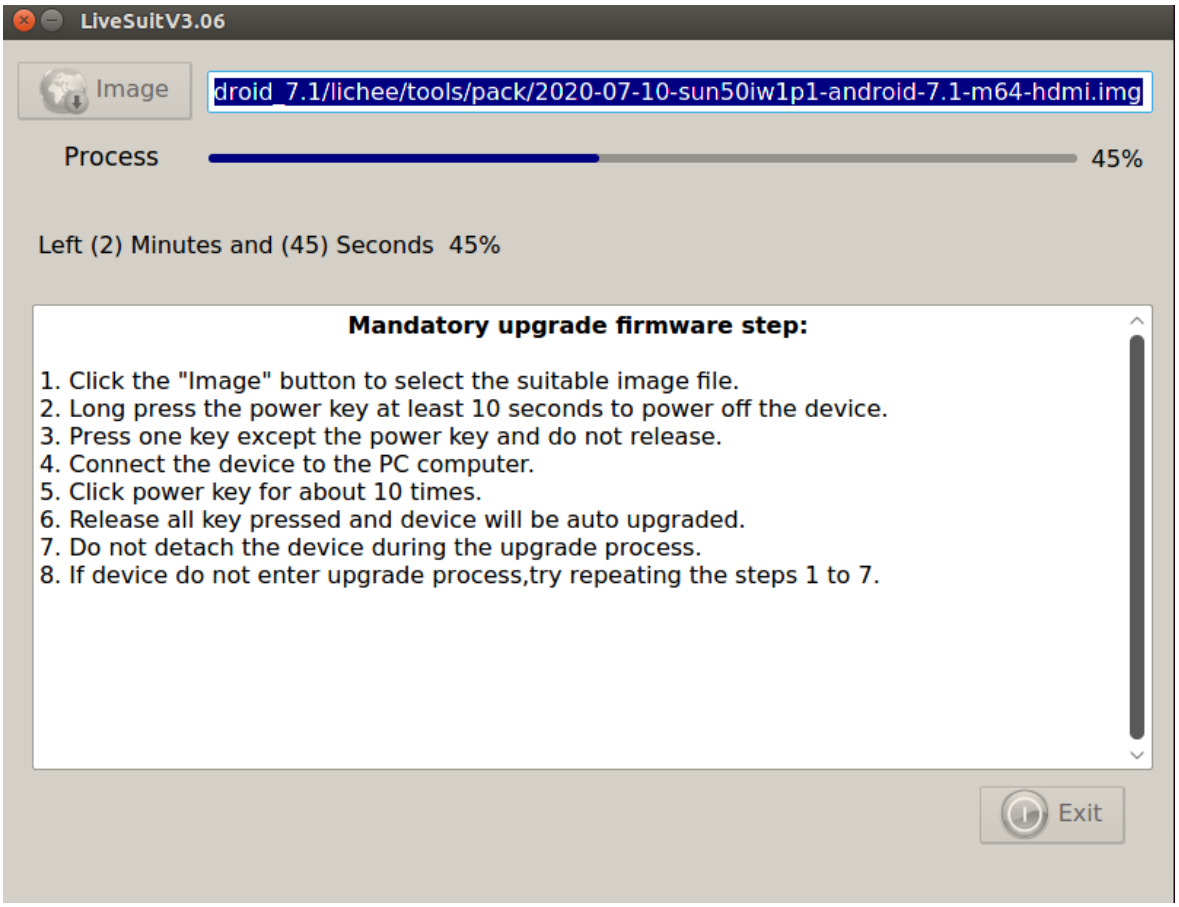


5. Finish Download



Install Image with LiveSuit on Linux

According to the Readme.pdf in LiveSuit Install package, After install the LiveSuit successfully please run LiveSuit.sh with **root permission**, then the download process is almost the same as PhoenixSuit.



Build Android source code

Get Android source code

Android 7.1

```
$ git clone https://github.com/BPI-SINOVOIP/BPI-A64-Android7
```

Android 6.1

```
$ git clone https://github.com/BPI-SINOVOIP/BPI-A64-Android
```

Build Android Source code

Please read the source code README.md

Development For Linux

Install Linux Image

Prepare

1. Prepare 8G/above TF card, USB-Serial interface, PC with Ubuntu System
2. Using your USB-Serial Connect debug console on M64
3. M64 board is only suport DC power supply bootup
4. Install bpi-tools on your Linux PC. If you can't access this URL or any other problems, please go to bpi-tools repo and install this tools manually.

```
$ apt-get install pv
$ curl -sL https://github.com/BPI-SINOVOIP/bpi-tools/raw/master/bpi-tools | sudo -E bash
```

5. Download BPI latest image
6. Login user/password: pi/bananapi or root/bananapi

Install Image to SDcard

1. Install image with bpi-tools on Linux

plug your sd card to your Linux PC, and run

```
$ sudo bpi-copy  xxx.img /dev/sdX
```

2. Install bpi image with Etcher on Windows, Linux and MacOS

Balena Etcher is an open source project by Balena, Flash OS images to SD cards & USB drives

Install Image to EMMC

1. Prepare a sd which is installed Linux image and bootup with sdcard
2. Copy emmc image to udisk then plug in M64, then mount udisk.
3. After mount udisk, use command "bpi-copy xxx-emmc-xxx.img" to install image on Emmc.
4. After success install, power off M64, eject the sdcard and poweron with emmc boot.

Switch to LCD boot type

1. The default release images are HDMI boot type, you can switch to LCD boot type for BPI 7" LCD support after first boot.

```
$ sudo bpi-bootsel /usr/lib/u-boot/bananapi/bpi-m64/BPI-M64-LCD7-linux4.4-8k.img.gz
$ reboot
```

2. Load the Touchscreen driver if you want to using TP

```
$ sudo modprobe gt9xxnew_ts.ko
```

Build Linux source code

Get the bsp source code

```
$ git clone https://github.com/BPI-SINOVOIP/BPI-M64-bsp-4.4
```

Build the bsp source code

Please read the source code README.md

Other development and test

GMAC

Use iperf3 to test gmac

1. On PC Terminal:

- Execute "iperf3 -s"

2. On M2U console:

- TCP test: "iperf3 -c serverIP"
- UDP test: "iperf3 -u -c serverIP"

```
root@bpi-iot-ros-ai:~# iperf3 -c 192.168.30.199
Connecting to host 192.168.30.199, port 5201
[  4] local 192.168.30.132 port 48991 connected to 192.168.30.199 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec   111 MBytes   928 Mbits/sec    0   1.24 MBytes
[  4]   1.00-2.00   sec   110 MBytes   925 Mbits/sec    0   1.24 MBytes
[  4]   2.00-3.00   sec   110 MBytes   923 Mbits/sec    0   1.24 MBytes
[  4]   3.00-4.00   sec   111 MBytes   929 Mbits/sec    0   1.24 MBytes
[  4]   4.00-5.00   sec   110 MBytes   925 Mbits/sec    0   1.24 MBytes
[  4]   5.00-6.00   sec   110 MBytes   924 Mbits/sec    0   1.24 MBytes
[  4]   6.00-7.00   sec   110 MBytes   923 Mbits/sec    0   1.24 MBytes
[  4]   7.00-8.00   sec   111 MBytes   931 Mbits/sec    0   1.24 MBytes
[  4]   8.00-9.00   sec   110 MBytes   925 Mbits/sec    0   1.24 MBytes
[  4]   9.00-10.00  sec   110 MBytes   923 Mbits/sec    0   1.24 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  1.08 GBytes   926 Mbits/sec    0             sender
[  4]   0.00-10.00  sec  1.07 GBytes   923 Mbits/sec                  receiver

iperf Done.
root@bpi-iot-ros-ai:~# iperf3 -u -c 192.168.30.199
Connecting to host 192.168.30.199, port 5201
[  4] local 192.168.30.132 port 35288 connected to 192.168.30.199 port 5201
[ ID] Interval           Transfer     Bandwidth       Total Datagrams
[  4]   0.00-1.00   sec   120 KBytes   983 Kbits/sec  15
[  4]   1.00-2.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   2.00-3.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   3.00-4.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   4.00-5.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   5.00-6.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   6.00-7.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   7.00-8.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   8.00-9.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   9.00-10.00  sec   128 KBytes  1.05 Mbits/sec  16
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  4]   0.00-10.00  sec  1.24 MBytes  1.04 Mbits/sec  0.134 ms  0/159 (0%)
[  4] Sent 159 datagrams

iperf Done.
```

Bluetooth

- Use bluetoothctl tool to operate BT
- Execute "bluetoothctl"
- If you don't know how to use bluetoothctl, type "help", you will see more commands
- Execute these commands:

```
[NEW] Device 00:1F:20:FF:E3:44 Bluetooth Mouse M557
[NEW] Device 40:70:4A:48:6F:43 RG100
[CHG] Device 38:59:F9:58:6A:CC RSSI: -83
[CHG] Device AC:BC:32:CF:7A:D0 Class: 0x38010c
[CHG] Device AC:BC:32:CF:7A:D0 Icon: computer
[CHG] Device 08:7C:BE:83:1B:7B RSSI: -94
[bluetooth]# connect 00:1F:20:FF:E3:44
Attempting to connect to 00:1F:20:FF:E3:44
[CHG] Device 00:1F:20:FF:E3:44 Connected: yes
[CHG] Device 00:1F:20:FF:E3:44 Modalias: usb:v046DpB010d1001
[CHG] Device 00:1F:20:FF:E3:44 UUIDs:
        00001000-0000-1000-8000-00805f9b34fb
        00001124-0000-1000-8000-00805f9b34fb
        00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 00:1F:20:FF:E3:44 Paired: yes
[CHG] Device 38:59:F9:58:6A:CC RSSI: -75
Connection successful
[CHG] Device 40:70:4A:48:6F:43 RSSI: -89
[bluetooth]# info 00:1F:20:FF:E3:44
Device 00:1F:20:FF:E3:44
        Name: Bluetooth Mouse M557
        Alias: Bluetooth Mouse M557
        Class: 0x002580
        Icon: input-mouse
        Paired: yes
        Trusted: no
        Blocked: no
        Connected: yes
        LegacyPairing: yes
        UUID: Service Discovery Serve.. (00001000-0000-1000-8000-00805f9b34fb)
        UUID: Human Interface Device... (00001124-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v046DpB010d1001
[bluetooth]#
```

WiFi on A64

WiFi Client

You have two ways to setup WiFi Client

1. Use commands to setup WiFi client

- ip link set wlan0 up
- iw dev wlan0 scan | grep SSID
- vim /etc/wpasupplicant/wpa_supplicant.conf

```
network={
ssid="ssid"
psk="password"
priority=1
}
```

- wpa_supplicant -iwlan0 -c /etc/wpa_supplicant/wpa_supplicant.conf
- dhclient wlan0

2. Use UI interface to setup WiFi Client

Ap Mode

1.Install hostapd and create hostapd configuration file hostapd.conf:

```
interface=wlan0
```

```
driver=nl80211
ssid=test
hw_mode=g
channel=1
```

2.Execute command："hostapd -d /<path>/hostapd.conf" If you meet problem like this：

```
root@bpi-iot-ros-ai:~# hostapd -d /etc/hostapd/hostapd.conf
random: Trying to read entropy from /dev/random
Configuration file: /etc/hostapd/hostapd.conf
rfkill: initial event: idx=0 type=2 op=0 soft=1 hard=0
rfkill: initial event: idx=1 type=1 op=0 soft=0 hard=0
rfkill: initial event: idx=2 type=1 op=0 soft=0 hard=0
nl80211: TDLS supported
nl80211: Supported cipher 00-0f-ac:1
nl80211: Supported cipher 00-0f-ac:5
nl80211: Supported cipher 00-0f-ac:2
nl80211: Supported cipher 00-0f-ac:4
nl80211: Supported cipher 00-0f-ac:6
nl80211: Using driver-based off-channel TX
nl80211: Use separate P2P group interface (driver advertised support)
nl80211: Enable multi-channel concurrent (driver advertised support)
nl80211: use P2P_DEVICE support
nl80211: Disable use_monitor with device_ap_sme since no monitor mode support detected
nl80211: interface wlan0 in phy phy0
nl80211: Set mode ifindex 9 iftype 3 (AP)
nl80211: Setup AP(wlan0) - device_ap_sme=1 use_monitor=0
nl80211: Subscribe to mgmt frames with AP handle 0x555da681e0 (device SME)
nl80211: Register frame type=0xd0 (WLAN_FC_STYPE_ACTION) nl_handle=0x555da681e0 match=
nl80211: Register frame command failed (type=208): ret=-114 (Operation already in progres
s)
nl80211: Register frame match - hexdump(len=0): [NULL]
nl80211: Could not configure driver mode
```

Then, you could solve by following command：

- nmcli radio wifi off

```
root@bpi-iot-ros-ai:~# rfkill list
0: sunxi-bt: Bluetooth
        Soft blocked: yes
        Hard blocked: no
1: phy0: Wireless LAN
        Soft blocked: yes
        Hard blocked: no
2: brcmfmac-wifi: Wireless LAN
        Soft blocked: yes
        Hard blocked: no
```

- rfkill unblock 1
- rfkill unblock 2
- ifconfig wlan0 up
- hostapd -d hostapd.conf

```
root@bpi-iot-ros-ai:~#
root@bpi-iot-ros-ai:~# hostapd /etc/hostapd/hostapd.conf
Configuration file: /etc/hostapd/hostapd.conf
Using interface wlan0 with hwaddr 8c:f7:10:1d:e4:80 and ssid "test"
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

Clear boot

- git clone https://github.com/BPI-SINOVOIP/BPI-files/tree/master/SD/100MB
- bpi-bootsel BPI-cleanboot-8k.img.gz /dev/sdX

GPIO Control

- To access a GPIO pin you first need to export it with

```
echo XX > /sys/class/gpio/export
```

- with XX being the number of the desired pin. To obtain the correct number you have to calculate it from the pin name (like PH18)

```
(position of letter in alphabet - 1) * 32 + pin number
 for PH18 this would be ( 8 - 1) * 32 + 18 = 224 + 18 = 242 (since 'h' is the 8th letter)
```

- echo "out/in" > /sys/class/gpio/gpio*NUMBER*/direction
- echo "0/1" > /sys/class/gpio/gpio*NUMBER*/value

Camara function

We use HDF5640 camara.

Guvcview

- Use your UI interface to operate camara
- Applications -> Sound & Video -> guvcview

Shell

- We also have built-in command in "/usr/local/bin" to test camara
- "./test_ov5640_image_mode.sh" to test picture taking function
- "./cameratest.sh" to test video recording function

IR function

- Execute "getevent"
- Use your IR device to send information to A64

RPi.GPIO

Install RPi.GPIO

- Execute "git clone https://github.com/BPI-SINOVOIP/RPi.GPIO"
- after clone the repo, cd RPi.GPIO
- Execute "sudo apt-get update"

- Execute "sudo apt-get install python-dev python3-dev"
- Execute "sudo python setup.py install" or "sudo python3 setup.py install" to install the module

Using RPi.GPIO

- cd /usr/local/bin
- Execute "./bpi_test_g40.py" to test RPi.GPIO

```
root@bpi-iot-ros-ai:/usr/local/bin# ./bpi_test_g40.py
Pi Board Information
--------------------
P1_REVISION => 3
RAM => 2048MB
REVISION => 4001
TYPE => Banana Pi M3[A83T]
PROCESSOR => Allwinner
MANUFACTURER => BPI-Sinovoip

Is this board info correct (y/n) ? y
8 GPIO.setup GPIO.OUT
./bpi_test_g40.py:21: RuntimeWarning: This channel is already in use, continu
disable warnings.
  GPIO.setup(pin, GPIO.OUT)
10 GPIO.setup GPIO.OUT
12 GPIO.setup GPIO.OUT
16 GPIO.setup GPIO.OUT
18 GPIO.setup GPIO.OUT
22 GPIO.setup GPIO.OUT
24 GPIO.setup GPIO.OUT
26 GPIO.setup GPIO.OUT
32 GPIO.setup GPIO.OUT
36 GPIO.setup GPIO.OUT
```

WringPi

- GitHub: https://github.com/BPI-SINOVOIP/BPI-WiringPi2.git
- We also have built-in test command in "/usr/local/bin"

How to Update WiringPi

- Execute "bpi-update -c pkglist.conf"

```
root@bpi-iot-ros-ai:/usr/local/bin# bpi-update -c pkglist.conf
CONFFILE=pkglist.conf
wait for download pkglist.conf ...
https://github.com/BPI-SINOVOIP/BPI-files/raw/master/others/for-bpi-tools/cont
OK!!\n
APP=/usr/bin/bpi-update
PKGLIST:
bpi-pkg-addons.conf
bpi-pkg-bpi-apps.conf
bpi-pkg-bpi-r2-wifi-firmware-tools.conf
bpi-pkg-bpi-service.conf
bpi-pkg-bpi-test-rfid.conf
bpi-pkg-bpi-tools.conf
bpi-pkg-bpi-w2-tools.conf
bpi-pkg-bpi-wiringpi-arm64.conf
bpi-pkg-bpi-wiringpi.conf
bpi-pkg-brcm.conf
bpi-pkg-bt-arm64.conf
bpi-pkg-bt.conf
bpi-pkg-camera-apps.conf
bpi-pkg-camera.conf
bpi-pkg-libvdpau_sunxi-arm64.conf
bpi-pkg-libvdpau_sunxi.conf
bpi-pkg-ov8865.conf
bpi-pkg-ov8865-enable.conf
```

- If your image is 32bit please do this command to install wring pi
- Execute "bpi-update -c bpi-pkg-bpi-wiringpi.conf"

- If your image is 64bit please do：“bpi-update -c bpi-pkg-bpi-wiringpi-arm64.conf”

```
root@bpi-iot-ros-ai:/usr/local/bin# chmod +x bpi_test_gpio40
root@bpi-iot-ros-ai:/usr/local/bin# ls
a10disp                     bt_reset.sh           test_ov5640_image_mode.sh
adbd                        cameratest.sh         test_ov5640.sh
adbd.sh                     cap                   test_ov8865.sh
apple.dat                   ffmpeg-3.1.4          tinacameratest
bpi-bt-on                   getevent              tinaplayerdemo
bpi-bt-patch                gpio                  tinarecorderdemo
bpi-pkg-bpi-wiringpi.conf   gpio40                tinymembench
bpi_test_52pi               guvcview              tusbd.ko
bpi_test_gpio40             guvcview.u1604        usbclnt
bpi_test_hello              h3disp                usbsrv
bpi_test_lcd1602            irtester              usbsrvd
bpi-wiringpi.tgz            pkglist.conf          usbsrvd-cl
brcm_bt_reset               realtinaplayerdemo    usbsrvd-srv
brcm_patchram_plus          sun8i-corekeeper.sh
root@bpi-iot-ros-ai:/usr/local/bin# chmod +x gpio40
root@bpi-iot-ros-ai:/usr/local/bin# ./bpi_test_gpio40
[BPI] phy led test
```

RGB 1602 LCD

- Execute "/usr/local/bin/bpi_test_lcd1602.sh"

0.96 Inch OLED Display

- Execute "/usr/local/bin/bpi_test_52pi.sh"

8x8 RGB LED Martix

- Firstly you need a GPIO Extend Board for 8x8 LED Martix