NanoPi NEO-AIR

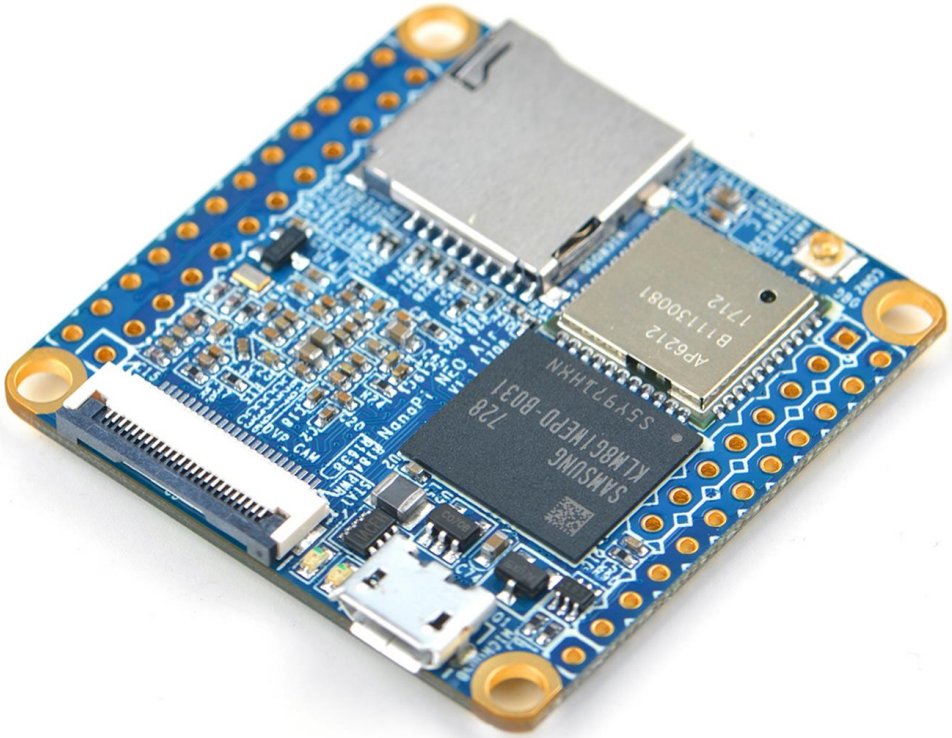
Download NanoPi NEO-AIR Files

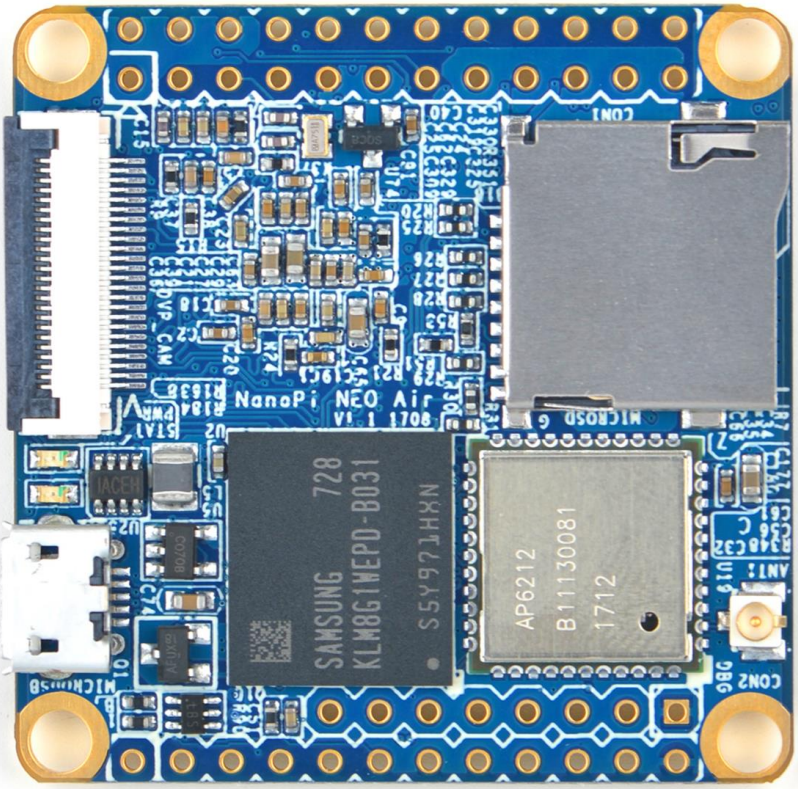
1 Introduction

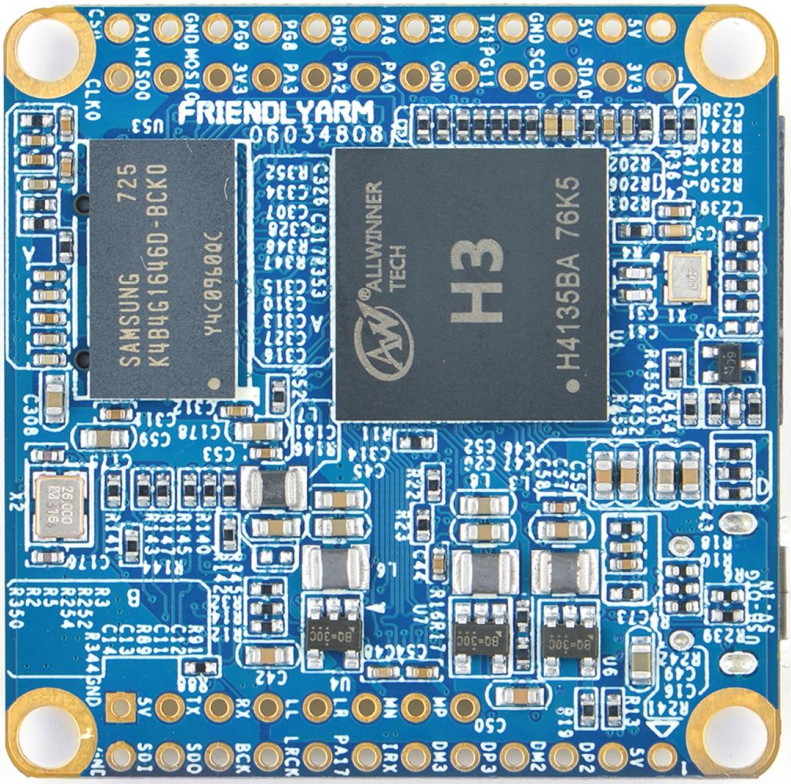
- The NanoPi NEO Air is a 40 x 40mm open source ARM board for makers.It uses an Allwinner H3 Quad Core A7 processor at 1.2GHz. Its pins are compatible with the NanoPi NEO(V 1.2) and its 24-pin header is compatible with Raspberry Pi's GPIO pin headers.
- The NanoPi NEO AIR features 512MB of 16bit wide DDR3 RAM, 8GB eMMC and one MicroSD slot. It has WiFi & Bluetooth and DVP camera interface(YUV422). The DVP camera interface can support friendlyarm's 5M-pixel camera module
- It has enhanced power circuit design and better heat dissipation.

2 Hardware Spec

- CPU: Allwinner H3, Quad-core Cortex-A7 Up to 1.2GHz
- RAM: 512MB DDR3 RAM
- Storage: 8GB eMMC
- WiFi: 802.11b/g/n
- Bluetooth: 4.0 dual mode
- DVP Camera: 0.5mm pitch 24 pin FPC seat
- MicroUSB: OTG and power input
- MicroSD Slot x 1
- Debug Serial Port: 4Pin,2.54mm pitch pin header
- GPIO1: 2.54mm spacing 24pin,It includes UART,SPI,I2C,GPIO
- GPIO2: 2.54mm spacing 12pin,It includes USBx2,IR,SPDIF,I2S
- PCB Size: 40 x 40mm
- PCB layer: 6
- Power Supply: DC 5V/2A
- Working Temperature: -20℃ to 70℃
- OS/Software: u-boot, UbuntuCore, eflasher
- Weight: 7.5g(WITHOUT Pin-headers)


AP6212
B1113008
1712
SAMSUNG
KLM8G1WEPD-B031
S5Y971HXN
728
NanoPi NEO Air


NanoPi NEO Air
V1.1 1708
SAMSUNG 728
KLM8G1WEPD-B031
S5Y971HXN
AP6212
B11130081
1712
CON1
CON2
ANT1



3 Software Features

3.1 uboot

- supports fastboot to update uboot

3.2 UbuntuCore 16.04

- mainline kernel: Linux-4.11.2
- rpi-monitor: check system status and information
- npi-config: system configuration utility for setting passwords, language, timezone, hostname, SSH and auto-login,and enabling/disabling i2c, spi, serial and PWM
- networkmanager: manage network
- software utility: RPi.GPIO_NP to access GPIO pins
- welcome window with basic system information and status
- auto-login with user account "pi" with access to npi-config
- on first system boot file system will be automatically extended.
- supports file system auto check and repair on system boot.

- supports FriendlyElec's NanoHat-PCM5102A
- supports FriendlyElec's Matrix_-_2.8_SPI_Key_TFT
- supports file transfer with Bluetooth
- supports FriendlyElec BakeBit modules
- supports dynamic frequency scaling and voltage regulation

3.3 Ubuntu OLED

- supports FriendlyElec's OLED module

3.4 Eflasher

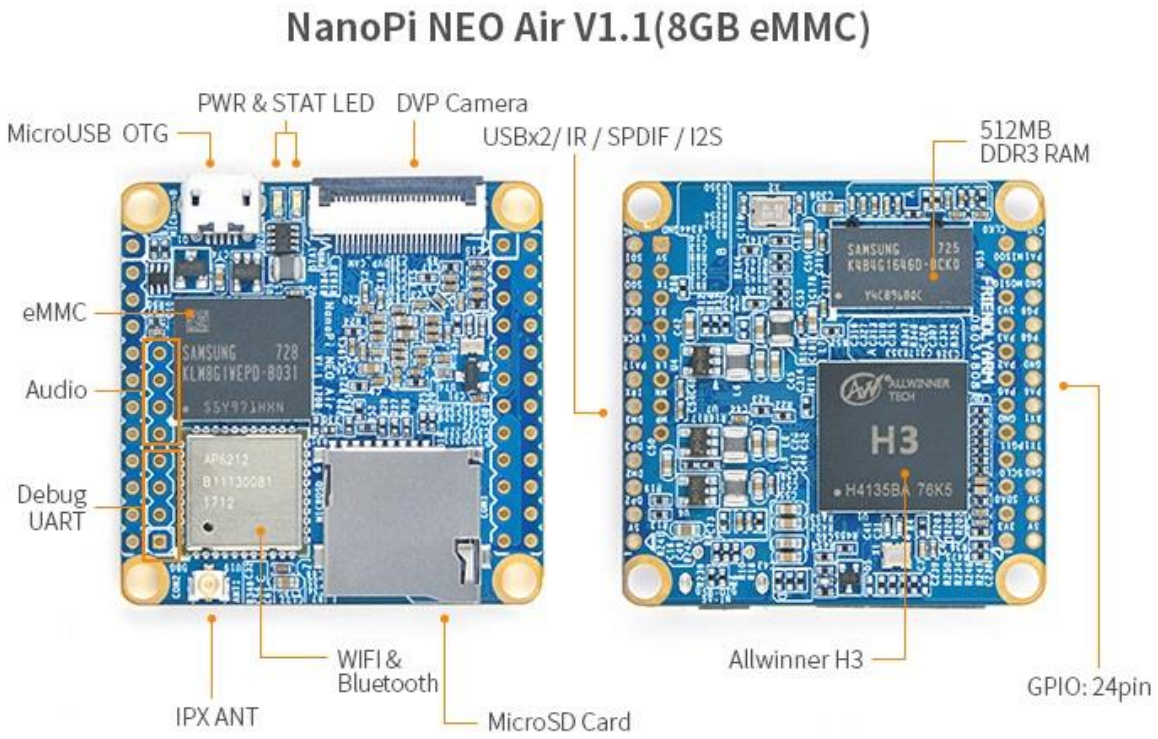
- supports flashing OS image to eMMC

3.5 Debian for NAS Dock

- supports FriendlyElec's NAS Dock

4 Diagram, Layout and Dimension

4.1 Layout



GPIO Pin Description

Pin#	Name	Linux gpio	Pin#	Name	Linux gpio
1	SYS_3.3V		2	VDD_5V	

3	I2C0_SDA/GPIOA12		4	VDD_5V	
5	I2C0_SCL/GPIOA11		6	GND	
7	GPIOG11	203	8	UART1_TX/GPIOG6	198
9	GND		10	UART1_RX/GPIOG7	199
11	UART2_TX/GPIOA0	0	12	GPIOA6	6
13	UART2_RTS/GPIOA2	2	14	GND	
15	UART2_CTS/GPIOA3	3	16	UART1_RTS/GPIOG8	200
17	SYS_3.3V		18	UART1_CTS/GPIOG9	201
19	SPI0_MOSI/GPIOC0	64	20	GND	
21	SPI0_MISO/GPIOC1	65	22	UART2_RX/GPIOA1	1
23	SPI0_CLK/GPIOC2	66	24	SPI0_CS/GPIOC3	67

- **USB/I2S/IR Pin Description**

Pin#	Name	Description
1	VDD_5V	5V Power Out
2	USB-DP2	USB2 DP Signal
3	USB-DM2	USB2 DM Signal

4	USB-DP3	USB3 DP Signal
5	USB-DM3	USB3 DM Signal
6	GPIOL11/IR-RX	GPIOL11 or IR Receive
7	SPDIF-OUT/GPIOA17	GPIOA17 or SPDIF-OUT
8	PCM0_SYNC/I2S0_LRC	I2S/PCM Sample Rate Clock/Sync
9	PCM0_CLK/I2S0_BCK	I2S/PCM Sample Rate Clock
10	PCM0_DOUT/I2S0_SDOUT	I2S/PCM Serial Bata Output
11	PCM0_DIN/I2S0_SDIN	I2S/PCM Serial Data Input
12	GND	0V

- **Debug Port（UART0）**

Pin#	Name
1	GND
2	VDD_5V
3	UART_TXD0/GPIOA4
4	UART_RXD0/GPIOA5/PWM0

- **Audio Port Description**

Pin#	Name	Description

1	LL	LINEOUTL, LINE-OUT Left Channel Output
2	LR	LINEOUTR, LINE-OUT Right Channel Output
3	MICN	MICIN1N, Microphone Negative Input
4	MICP	MICIN1P, Microphone Positive Input

- **DVP Camera IF Pin Spec**

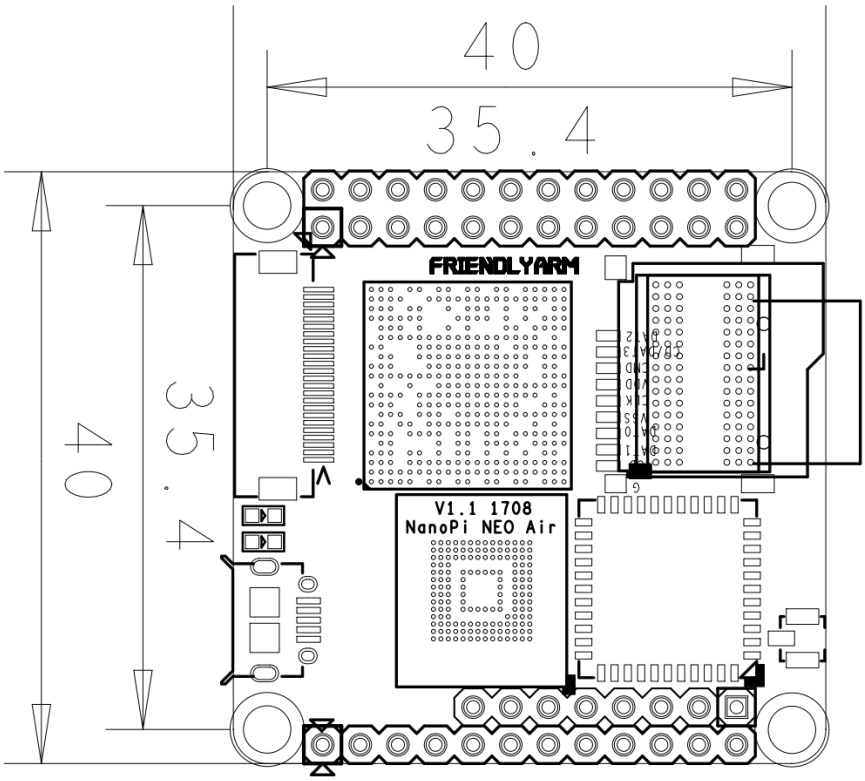
Pin#	Name	Description
1, 2	SYS_3.3V	3.3V power output, to camera modules
7,9,13,15,24	GND	Gound, 0V
3	I2C2_SCL	I2C Clock Signal
4	I2C2_SDA	I2C Data Signal
5	GPIOE15	Regular GPIO, control signals output to camera modules
6	GPIOE14	Regular GPIO, control signals output to camera modules
8	MCLK	Clock signals output to camera modules
10	NC	Not Connected
11	VSYNC	vertical synchronization to CPU from camera modules
12	HREF/HSYNC	HREF/HSYNC signal to CPU from camera modules

14	PCLK	PCLK signal to CPU from camera modules
16-23	Data bit7-0	data signals

Note:

1. SYS_3.3V: 3.3V power output
2. VVDD_5V: 5V power input/output. When the external device's voltage is greater than the MicroUSB's voltage the external device is charging the board otherwise the board powers the external device. The input range is 4.7V ~ 5.6V
3. All pins are 3.3V, output current is 5mA
4. For more details refer to the document:NanoPi-NEO-Air-1708-Schematic.pdf

4.2 Dimensional Diagram



- NanoPi-NEO-Air-V1.1-1708 pcb file in dxf format

5 Get Started

5.1 Essentials You Need

Before starting to use your NanoPi NEO AIR get the following items ready

- NanoPi NEO AIR
- microSD Card/TFCard: Class 10 or Above, minimum 8GB SDHC
- microUSB power. A 5V/2A power is a must
- A Host computer running Ubuntu 16.04 64 bit system

5.2 TF Cards We Tested

To make your NanoPi NEO AIR boot and run fast we highly recommend you use a Class10 8GB SDHC TF card or a better one. The following cards are what we used in all our test cases presented here:

- SanDisk TF 8G Class10 Micro/SD TF card:

- SanDisk TF128G MicroSDXC TF 128G Class10 48MB/S:

- 川宇 8G C10 High Speed class10 micro SD card:

5.3 Install OS

5.3.1 Download Image Files

- Get the following files from here download link to download image files and the flashing utility:

Image Files:	
nanopi-neo-air_sd_friendlycore-xenial_3.4_armhf_YYYYMMDD.img.zip	Base on UbuntuCore, kernel:Linux-3.4
nanopi-neo-air_sd_friendlycore-xenial_4.14_armhf_YYYYMMDD.img.zip	Base on UbuntuCore, kernel:Linux-4.14
nanopi-neo-air_sd_friendlywrt_4.14_armhf_YYYYMMDD.img.zip	Base on OpenWrt, kernel:Linux-4.14

nanopi-neo-air_eflasher_friendlycore-xenial_4.14_armhf_YYYYMMDD.img.zip	eflasher image, for flashing FriendlyCore(Linux-4.14) to eMMC
nanopi-neo-air_eflasher_openwrt_4.14_armhf_YYYYMMDD.img.zip	eflasher image, for flashing OpenWrt(Linux-4.14) to eMMC
Flash Utility:	
win32diskimager.rar	Windows utility. Under Linux users can use "dd"

5.3.2 Comparison of Linux-3.4 and Linux-4.14

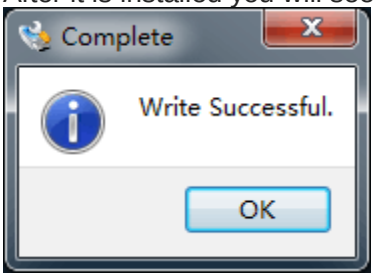
- Our Linux-3.4 is provided by Allwinner. Allwinner has done a lot of customization work which on one hand contains many features and functions but on the other hand incurs overheat issues. If your application needs to use VPU or GPU you need to use the 3.4 kernel based ROM and use a heat sink together with your board.
- Our Linux-4.14 is based on the mainline kernel. We will keep this kernel with the latest one released by Linus Torvalds. This kernel is stable and doesn't generate heat that much. If your application doesn't need to use VPU or GPU we recommend you to use this kernel.
- For more details about the Linux-4.14 kernel refer to: Building U-boot and Linux for H5/H3/H2+

5.3.3 Linux

5.3.3.1 Flash to TF

- FriendlyCore / Debian / Ubuntu / OpenWrt / DietPi are all based on a same Linux distribution and their installation methods are the same.
- Extract the Linux image and win32diskimager.rar files. Insert a TF card(at least 8G) into a Windows PC and run the win32diskimager utility as administrator. On the utility's main window select your TF card's drive, the wanted image file and click on "write" to start flashing the TF card.

After it is installed you will see the following window:



- Insert this card into your board's BOOT slot and power on (with a 5V/2A power source). If the PWR LED is on and the STAT LED is blinking this indicates your board has successfully booted.
  ;

5.3.3.2 Flash to eMMC

5.3.3.2.1 Flash OS with eflasher Utility

- For more details about eflasher refer to the wiki link: EFlasher。
- Extract the eflasher Image and win32diskimager.rar files. Insert a TF card(at least 4G) into a Windows PC and run the win32diskimager utility as administrator. On the utility's main window select your TF card's drive, the wanted image file and click on "write" to start flashing the TF card.
- Insert this card into your board's BOOT slot and power on (with a 5V/2A power source). If the green LED is on and the blue LED is blinking this indicates your board has successfully booted.
- Connect the board to an HDMI monitor or an LCD and a USB mouse, and select an OS to start installation.

- If your board doesn't support HDMI or no monitor is connected you can select an OS by running the following command:

```
$ su root
$ eflasher
```

The password for "root" is "fa".

We take "nanopi-neo-air_eflasher_friendlycore-xenial_4.14_armhf_YYYYMMDD.img" as an example. After you run the "eflasher" command you will see the following messages:

```
------------------------------------------------------------------------
  EFlasher v1.2 b190412 running on NanoPi
    Doc: http://wiki.friendlyarm.com/wiki/index.php/EFlasher
    eMMC: 14.56 GB
------------------------------------------------------------------------
  # Select an OS to install:
    1) friendlycore-xenial_4.14

  # Select your backup target device:
     tf) [*] TF card  (/dev/mmcblk0p3 - 790.69 MB free - 3.44 GB total )
    usb) [ ] USB disk  (<none>)

  # Backup eMMC flash to TF card:
    Not enough free disk space on your TF card

  # Restore eMMC flash from backup file:
    No backup files found

  # Configure automatic job:
    aui) Automatic installing (Curr:Off)
    aur) Automatic restoring (Curr:Off)

  # Format drive
    ftf) Format TF card back to original size
------------------------------------------------------------------------
>>> Enter an option (1/tf/usb/aui/aur/ftf) :
```

Type "1", select writing friendlycore system to eMMC you will see the following messages:

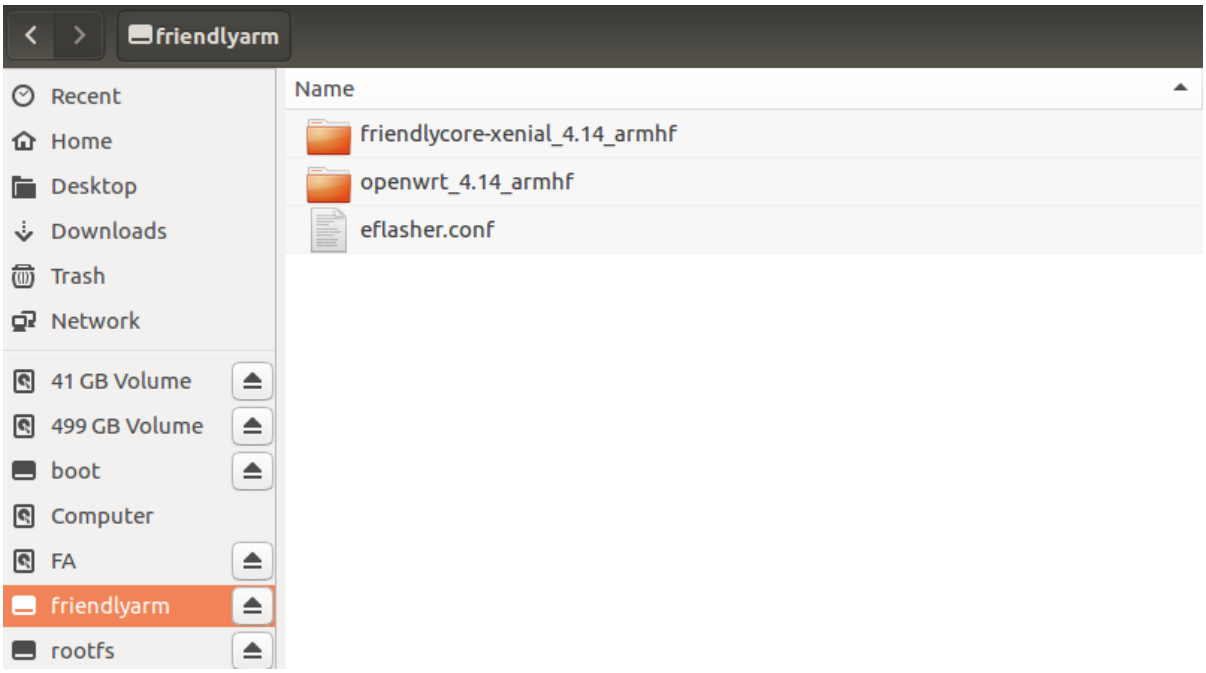
```
------------------------------------------------------------------------
  Ready to Go with FriendlyCore
------------------------------------------------------------------------
  Ready to install
  Version:
               2019-04-25
  Path:
               /mnt/sdcard/friendlycore-xenial_4.14_armhf
  Image files:
               u-boot-sunxi-with-spl.bin 1.99 MB
               boot.img 40.00 MB
               rootfs.img 2.44 GB

  Total size:
               2.48 GB
  Kernel parameter:
               Default
------------------------------------------------------------------------
>>> Do you wish to continue? (yes/no) :
```

Type "yes" to start installation:

```
------------------------------------------------------------------------
  Installing FriendlyCore
------------------------------------------------------------------------
  Speed: 17.65 MB/s
  Remaining Time: 00:02:10
  [=====_____________________________________________]  11%
------------------------------------------------------------------------
  >>>Enter "c" to cancel.
```

After it is done power off the system, take off the TF card, power on again your system will be booted from eMMC.

- If you want to flash other system to eMMC you can download the whole images-for-eflasher directory and extract the package under that directory to the FRIENDLYARM partition of an installation SD card.



6 Work with FriendlyCore

6.1 Introduction

FriendlyCore is a light Linux system without X-windows, based on ubuntu core, It uses the Qt-Embedded's GUI and is popular in industrial and enterprise applications.

Besides the regular Ubuntu Core's features FriendlyCore has the following additional features:

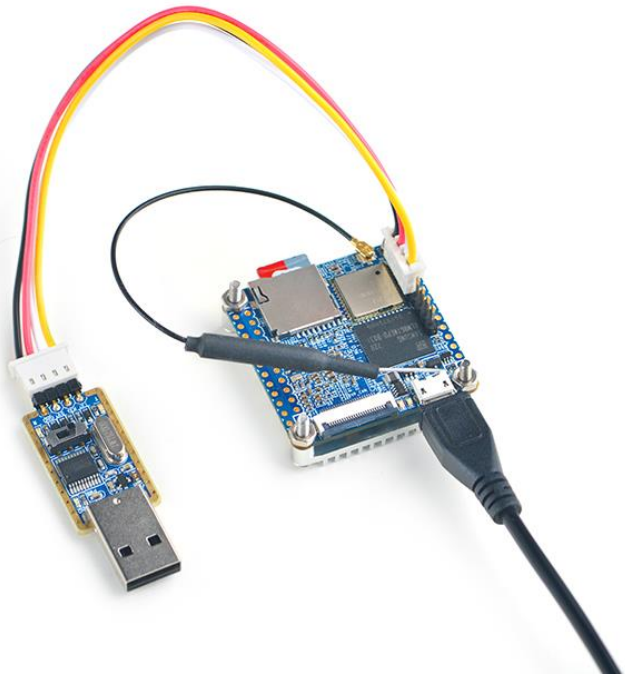
- it integrates Qt4.8;
- it integrates NetworkManager;
- it has bluez and Bluetooth related packages;
- it has alsa packages;
- it has npi-config;
- it has RPiGPIO, a Python GPIO module;
- it has some Python/C demo in /root/ directory;
- it enables 512M-swap partition;

6.2 System Login

- If your board is connected to an HDMI monitor you need to use a USB mouse and keyboard.
- If you want to do kernel development you need to use a serial communication board, ie a PSU-ONECOM board, which will

allow you to operate the board via a serial terminal.Here is a setup where we connect a board to a PC via the PSU-ONECOM and you can power on your board from either the PSU-ONECOM or its

MicroUSB:
You can use a USB to Serial conversion board too.
Make sure you use a 5V/2A power to power your board from its MicroUSB port:



- FriendlyCore User Accounts:

Non-root User:

```
User Name: pi
Password: pi
```

Root:

```
User Name: root
Password: fa
```

The system is automatically logged in as "pi". You can do "sudo npi-config" to disable auto login.

- Update packages

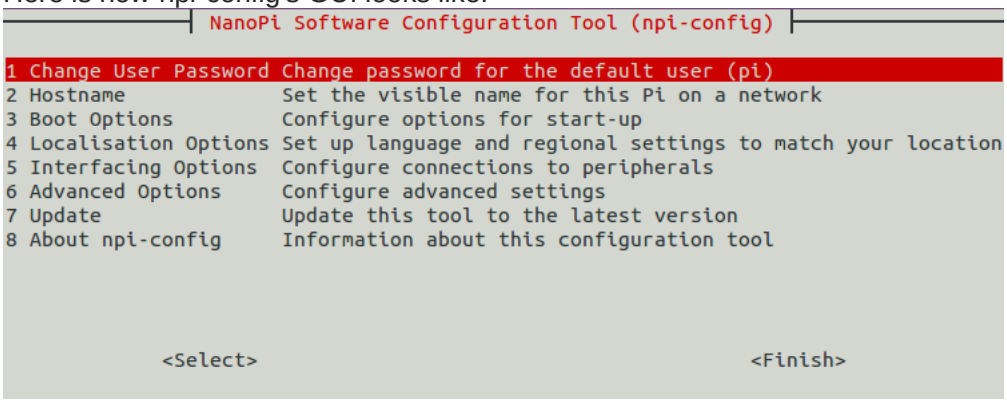
```
$ sudo apt-get update
```

6.3 Configure System with npi-config

The npi-config is a commandline utility which can be used to initialize system configurations such as user password, system language, time zone, Hostname, SSH switch , Auto login and etc. Type the following command to run this utility.

```
$ sudo npi-config
```

Here is how npi-config's GUI looks like:



6.4 Develop Qt Application

Please refer to: [How to Build and Install Qt Application for FriendlyELEC Boards]

6.5 Setup Program to AutoRun

You can setup a program to autorun on system boot with npi-config:

```
sudo npi-config
```

Go to Boot Options -> Autologin -> Qt/Embedded, select Enable and reboot.

6.6 Extend TF Card's Section

When FriendlyCore is loaded the TF card's section will be automatically extended.You can check the section's size by running the following command:

```
$ df -h
```

6.7 Transfer files using Bluetooth

Take the example of transferring files to the mobile phone. First, set your mobile phone Bluetooth to detectable status, then execute the following command to start Bluetooth search. :

```
hcitool scan
```

Search results look like :

```
Scanning ...
    2C:8A:72:1D:46:02   HTC6525LVW
```

This means that a mobile phone named HTC6525LVW is searched. We write down the MAC address in front of the phone name, and then use the sdptool command to view the Bluetooth service supported by the phone :

```
sdptool browser 2C:8A:72:1D:46:02
```

Note: Please replace the MAC address in the above command with the actual Bluetooth MAC address of the mobile phone.
This command will detail the protocols supported by Bluetooth for mobile phones. What we need to care about is a

file transfer service called OBEX Object Push. Take the HTC6525LVW mobile phone as an example. The results are as follows：

```
Service Name: OBEX Object Push
Service RecHandle: 0x1000b
Service Class ID List:
  "OBEX Object Push" (0x1105)
Protocol Descriptor List:
  "L2CAP" (0x0100)
  "RFCOMM" (0x0003)
    Channel: 12
  "OBEX" (0x0008)
Profile Descriptor List:
  "OBEX Object Push" (0x1105)
    Version: 0x0100
```

As can be seen from the above information, the channel used by the OBEX Object Push service of this mobile phone is 12, we need to pass it to the obexftp command, and finally the command to initiate the file transfer request is as follows：

```
obexftp --nopath --noconn --uuid none --bluetooth -b 2C:8A:72:1D:46:02 -B 12 -put
example.jpg
```

Note: Please replace the MAC address, channel and file name in the above command with the actual one.

After executing the above commands, please pay attention to the screen of the mobile phone. The mobile phone will pop up a prompt for pairing and receiving files. After confirming, the file transfer will start.

Bluetooth FAQ：
1) Bluetooth device not found on the development board, try to open Bluetooth with the following command：

```
rfkill unblock 0
```

2) Prompt can not find the relevant command, you can try to install related software with the following command：

```
apt-get install bluetooth bluez obexftp openobex-apps python-gobject ussp-push
```

6.8 WiFi

For either an SD WiFi or a USB WiFi you can connect it to your board in the same way. The APXX series WiFi chips are SD WiFi chips. By default FriendlyElec's system supports most popular USB WiFi modules. Here is a list of the USB WiFi modules we tested:

Index	Model
1	RTL8188CUS/8188EU 802.11n WLAN Adapter
2	RT2070 Wireless Adapter
3	RT2870/RT3070 Wireless Adapter

4	RTL8192CU Wireless Adapter
5	mi WiFi mt7601
6	5G USB WiFi RTL8821CU
7	5G USB WiFi RTL8812AU

You can use the NetworkManager utility to manage network. You can run "nmcli" in the commandline utility to start it. Here are the commands to start a WiFi connection:

- Change to root

```
$ su root
```

- Check device list

```
$ nmcli dev
```

Note: if the status of a device is "unmanaged" it means that device cannot be accessed by NetworkManager. To make it accessed you need to clear the settings under "/etc/network/interfaces" and reboot your system.

- Start WiFi

```
$ nmcli r wifi on
```

- Scan Surrounding WiFi Sources

```
$ nmcli dev wifi
```

- Connect to a WiFi Source

```
$ nmcli dev wifi connect "SSID" password "PASSWORD" ifname wlan0
```

The "SSID" and "PASSWORD" need to be replaced with your actual SSID and password.If you have multiple WiFi devices you need to specify the one you want to connect to a WiFi source with iface
If a connection succeeds it will be automatically setup on next system reboot.

For more details about NetworkManager refer to this link: Use NetworkManager to configure network settings

If your USB WiFi module doesn't work most likely your system doesn't have its driver. For a Debian system you can get a driver from Debian-WiFi and install it on your system. For a Ubuntu system you can install a driver by running the following commands:

```
$ apt-get install linux-firmware
```

In general all WiFi drivers are located at the "/lib/firmware" directory.

6.9 Setup Wi-Fi Hotspot

Run the following command to enter AP mode:

```
$ su root
$ turn-wifi-into-apmode yes
```

You will be prompted to type your WiFi hotspot's name and password and then proceed with default prompts. After this is done you will be able to find this hotspot in a neadby cell phone or PC. You can login to this board at 192.168.8.1:

```
$ ssh root@192.168.8.1
```

When asked to type a password you can type "fa".

To speed up your ssh login you can turn off your wifi by running the following command:

```
$ iwconfig wlan0 power off
```

To switch back to Station mode run the following command:

```
$ turn-wifi-into-apmode no
```

6.10 Bluetooth

Search for surrounding bluetooth devices by running the following command:

```
$ su root
$ hciconfig hci0 up
$ hcitool scan
```

You can run "hciconfig" to check bluetooth's status.

6.11 Ethernet Connection

If a board is connected to a network via Ethernet before it is powered on it will automatically obtain an IP with DHCP activated after it is powered up. If you want to set up a static IP refer to: Use NetworkManager to configure network settings。

6.12 WiringPi and Python Wrapper

- WiringNP: NanoPi NEO/NEO2/Air GPIO Programming with C
- RPi.GPIO : NanoPi NEO/NEO2/Air GPIO Programming with Python

6.13 Set Audio Device

If your system has multiple audio devices such as HDMI-Audio, 3.5mm audio jack and I2S-Codec you can set system's default audio device by running the following commands.

- After your board is booted run the following commands to install alsa packages:

```
$ apt-get update
$ apt-get install libasound2
$ apt-get install alsa-base
$ apt-get install alsa-utils
```

- After installation is done you can list all the audio devices by running the following command. Here is a similar list you may see after you run the command:

```
$ aplay -l
card 0: HDMI
card 1: 3.5mm codec
card 2: I2S codec
```

"card 0" is HDMI-Audio, "card 1" is 3.5mm audio jack and "card 2" is I2S-Codec. You can set default audio device to HDMI-Audio by changing the "/etc/asound.conf" file as follows:

```
pcm.!default {
    type hw
    card 0
    device 0
}
```

```
ctl.!default {
    type hw
    card 0
}
```

If you change "card 0" to "card 1" the 3.5mm audio jack will be set to the default device.
Copy a .wav file to your board and test it by running the following command:

```
$ aplay /root/Music/test.wav
```

You will hear sounds from system's default audio device.
If you are using H3/H5/H2+ series board with mainline kernel, the easier way is using npi-config。

6.14 Connect to DVP Camera CAM500B

For NanoPi-NEO-Air the CAM500B can work with both Linux-3.4 Kernel and Linux-4.14 Kernel.
The CAM500B camera module is a 5M-pixel camera with DVP interface. For more tech details about it you can refer to Matrix - CAM500B.

connect your board to camera module. Then boot OS, connect your board to a network, log into the board as root and run "mjpg-streamer":

```
$ cd /root/C/mjpg-streamer
$ make
$ ./start.sh
```

You need to change the start.sh script and make sure it uses a correct /dev/videoX node. You can check your camera's node by running the following commands:

```
$ apt-get install v4l-utils
$ v4l2-ctl -d /dev/video0 -D
Driver Info (not using libv4l2):
        Driver name   : sun6i-video
        Card type     : sun6i-csi
        Bus info      : platform:camera
        Driver version: 4.14.0
    ...
```

The above messages indicate that "/dev/video0" is camera's device node.The mjpg-streamer application is an open source video steam server. After it is successfully started the following messages will be popped up:

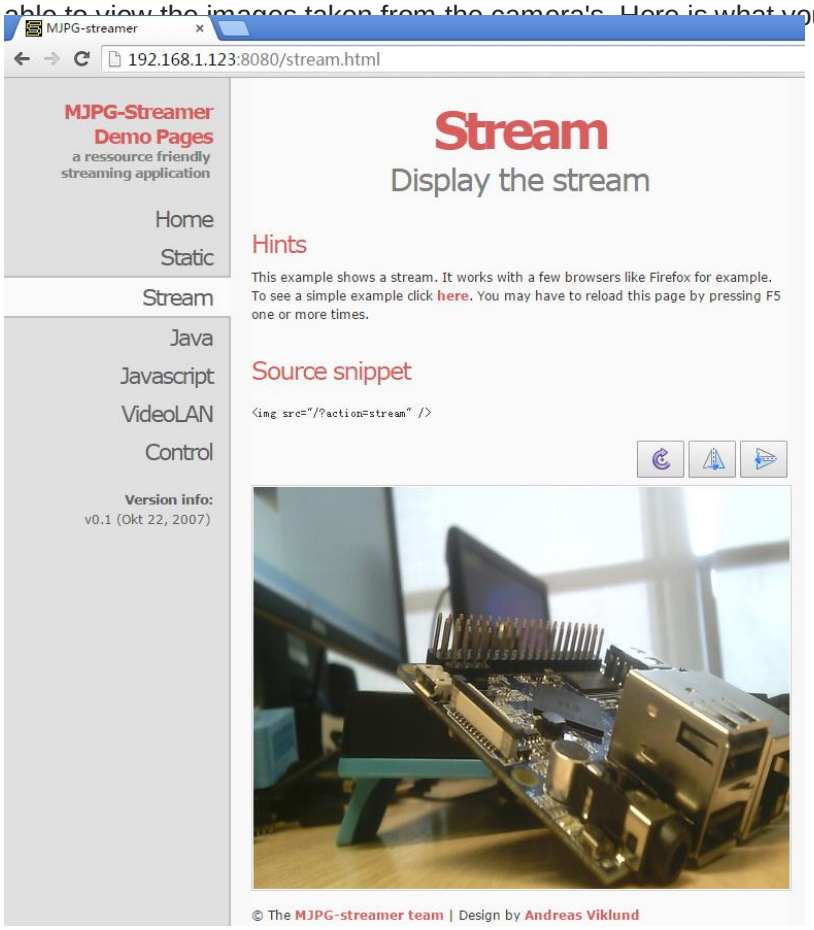
```
$ ./start.sh
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 1280 x 720
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 90
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8080
 o: username:password.: disabled
 o: commands..........: enabled
```

start.sh runs the following two commands:

```
export LD_LIBRARY_PATH="$(pwd)"
./mjpg_streamer -i "./input_uvc.so -d /dev/video0 -y 1 -r 1280x720 -f 30 -q 90 -n
-fb 0" -o "./output_http.so -w ./www"
```

Here are some details for mjpg_streamer's major options:
-i: input device. For example "input_uvc.so" means it takes input from a camera;
-o: output device. For example "output_http.so" means the it transmits data via http;
-d: input device's subparameter. It defines a camera's device node;
-y: input device's subparameter. It defines a camera's data format: 1:yuyv, 2:yvyu, 3:uyvy 4:vyuy. If this option isn't defined MJPEG will be set as the data format;
-r: input device's subparameter. It defines a camera's resolution;
-f: input device's subparameter. It defines a camera's fps. But whether this fps is supported depends on its driver;
-q: input device's subparameter. It defines the quality of an image generated by libjpeg soft-encoding;
-n: input device's subparameter. It disables the dynctrls function;
-fb: input device's subparameter. It specifies whether an input image is displayed at "/dev/fbX";
-w: output device's subparameter. It defines a directory to hold web pages;

In our case the board's IP address was 192.168.1.230. We typed 192.168.1.230:8080 in a browser and were able to view the images taken from the camera's. Here is what you would expect to observe:



The mjpg-streamer utility uses libjpeg to software-encode steam data. The Linux-4.14 based ROM currently doesn't support hardware-encoding. If you use a H3 boards with Linux-3.4 based ROM you can use the ffmpeg utility to hardware-encode stream data and this can greatly release CPU's resources and speed up encoding:

```
$ ffmpeg -t 30 -f v4l2 -channel 0 -video_size 1280x720 -i /dev/video0 -pix_fmt
nv12 -r 30 \
        -b:v 64k -c:v cedrus264 test.mp4
```

By default it records a 30-second video. Typing "q" stops video recording. After recording is stopped a test.mp4 file will be generated.

6.15 Connect to USB Camera(FA-CAM202)

The FA-CAM202 is a 200M USB camera. Connect your board to camera module. Then boot OS, connect your board to a network, log into the board as root and run "mjpg-streamer":

```
$ cd /root/C/mjpg-streamer
$ make
$ ./start.sh
```

You need to change the start.sh script and make sure it uses a correct /dev/videoX node. You can check your camera's node by running the following commands:

```
$ apt-get install v4l-utils
$ v4l2-ctl -d /dev/video0 -D
Driver Info (not using libv4l2):
        Driver name   : uvcvideo
        Card type     : HC 3358+2100: HC 3358+2100  / USB 2.0 Camera: USB 2.0
Camera
        Bus info      : usb-1c1b000.usb-1
    ...
```

The above messages indicate that "/dev/video0" is camera's device node.The mjpg-streamer application is an open source video steam server. After it is successfully started the following messages will be popped up:

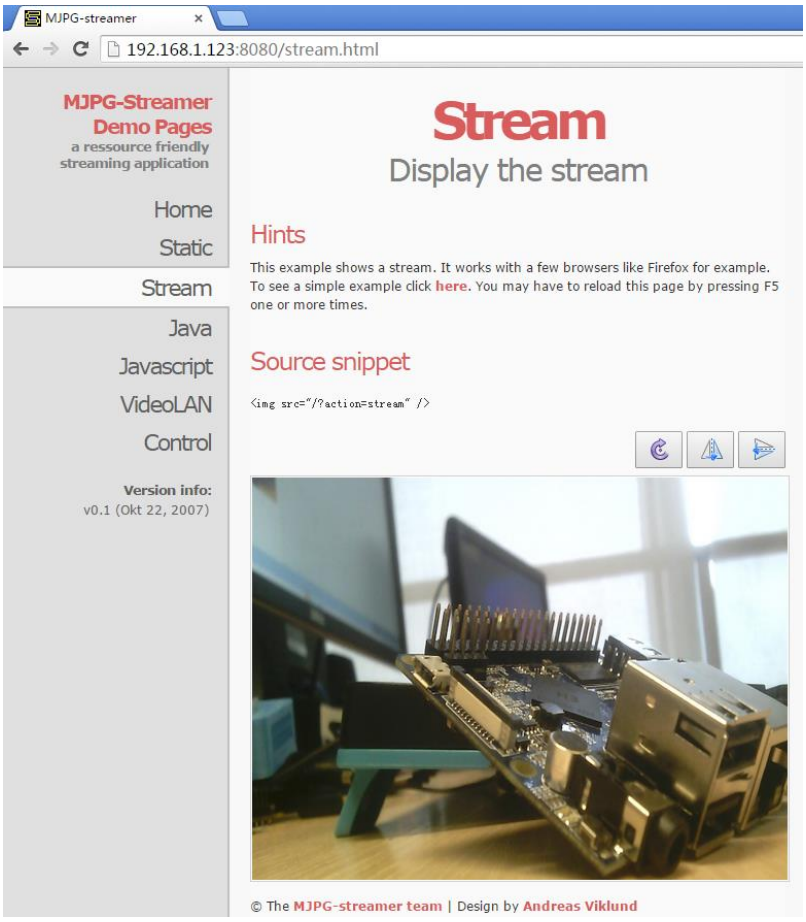
```
$ ./start.sh
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 1280 x 720
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 90
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8080
 o: username:password.: disabled
 o: commands..........: enabled
```

start.sh runs the following two commands:

```
export LD_LIBRARY_PATH="$(pwd)"
./mjpg_streamer -i "./input_uvc.so -d /dev/video0 -y 1 -r 1280x720 -f 30 -q 90 -n
-fb 0" -o "./output_http.so -w ./www"
```

Here are some details for mjpg_streamer's major options:
-i: input device. For example "input_uvc.so" means it takes input from a camera;
-o: output device. For example "output_http.so" means the it transmits data via http;
-d: input device's subparameter. It defines a camera's device node;
-y: input device's subparameter. It defines a camera's data format: 1:yuyv, 2:yvyu, 3:uyvy 4:vyuy. If this option isn't defined MJPEG will be set as the data format;
-r: input device's subparameter. It defines a camera's resolution;
-f: input device's subparameter. It defines a camera's fps. But whether this fps is supported depends on its driver;
-q: input device's subparameter. It defines the quality of an image generated by libjpeg soft-encoding;
-n: input device's subparameter. It disables the dynctrls function;
-fb: input device's subparameter. It specifies whether an input image is displayed at "/dev/fbX";
-w: output device's subparameter. It defines a directory to hold web pages;

In our case the board's IP address was 192.168.1.230. We typed 192.168.1.230:8080 in a browser and were able to view the images taken from the camera's. Here is what you would expect to observe:



6.16 Check CPU's Working Temperature

You can get CPU's working temperature by running the following command:

```
$ cpu_freq
Aavailable frequency(KHz):
        480000 624000 816000 1008000
Current frequency(KHz):
        CPU0 online=1 temp=26548C governor=ondemand freq=624000KHz
        CPU1 online=1 temp=26548C governor=ondemand freq=624000KHz
        CPU2 online=1 temp=26548C governor=ondemand freq=624000KHz
        CPU3 online=1 temp=26548C governor=ondemand freq=624000KHz
```

This message means there are currently four CPUs working. All of their working temperature is 26.5 degree in Celsius and each one's clock is 624MHz.
Set CPU frequency:

```
$ cpu_freq -s 1008000
Aavailable frequency(KHz):
        480000 624000 816000 1008000
Current frequency(KHz):
        CPU0 online=1 temp=36702C governor=userspace freq=1008000KHz
        CPU1 online=1 temp=36702C governor=userspace freq=1008000KHz
        CPU2 online=1 temp=36702C governor=userspace freq=1008000KHz
        CPU3 online=1 temp=36702C governor=userspace freq=1008000KHz
```

6.17 Test Infrared Receiver

Note: Please Check your board if IR receiver exist.
By default the infrared function is disabled you can enable it by using the npi-config utility:

```
$ npi-config
    6 Advanced Options     Configure advanced settings
        A8 IR              Enable/Disable IR
            ir Enable/Disable ir[enabled]
```

Reboot your system and test its infrared function by running the following commands:

```
$ apt-get install ir-keytable
$ echo "+rc-5 +nec +rc-6 +jvc +sony +rc-5-sz +sanyo +sharp +mce_kbd +xmp" >
/sys/class/rc/rc0/protocols    # Enable infrared
$ ir-keytable -t
Testing events. Please, press CTRL-C to abort.
```

"ir-keytable -t" is used to check whether the receiver receives infrared signals. You can use a remote control to send infrared signals to the receiver. If it works you will see similar messages as follows:

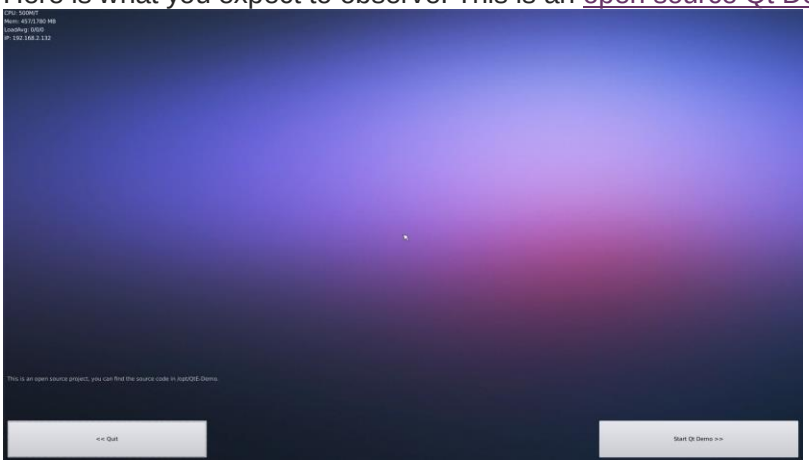
```
1522404275.767215: event type EV_MSC(0x04): scancode = 0xe0e43
1522404275.767215: event type EV_SYN(0x00).
1522404278.911267: event type EV_MSC(0x04): scancode = 0xe0e42
1522404278.911267: event type EV_SYN(0x00).
```

6.18 Run Qt Demo

Run the following command

```
$ sudo /opt/QtE-Demo/run.sh
```

Here is what you expect to observe. This is an open source Qt Demo:



6.19 How to install and use docker (for armhf system)

6.19.1 How to Install Docker

Run the following commands：

```
sudo apt-get update
sudo apt-get install docker.io
```

6.19.2 Test Docker installation

Test that your installation works by running the simple docker image:

```
git clone https://github.com/friendlyarm/debian-jessie-arm-docker
cd debian-jessie-arm-docker
```

```
./rebuild-image.sh
./run.sh
```

6.20 Using 4G Module EC20 on FriendlyCore

6.20.1 Step1：Compile the quectel-CM command line tool on the development board

Compile and install quectel-CM into the /usr/bin/ directory by entering the following command：

```
git clone https://github.com/friendlyarm/quectel-cm.git
cd quectel-cm/
make
cp quectel-CM /usr/bin/
```

6.20.2 Step2：Add udhcpc script

The quectel-CM tool will call the udhcpc script. we need to create a udhcpc script for it. Please create a new file with the editor you are familiar with. The file name is: /usr/share/udhcpc/default.script, the content is as follows：

```
#!/bin/sh

# udhcpc script edited by Tim Riker <Tim@Rikers.org>

[ -z "$1" ] && echo "Error: should be called from udhcpc" && exit 1

RESOLV_CONF="/etc/resolv.conf"
[ -n "$broadcast" ] && BROADCAST="broadcast $broadcast"
[ -n "$subnet" ] && NETMASK="netmask $subnet"

case "$1" in
  deconfig)
    /sbin/ifconfig $interface 0.0.0.0
    ;;

  renew|bound)
    /sbin/ifconfig $interface $ip $BROADCAST $NETMASK

    if [ -n "$router" ] ; then
      echo "deleting routers"
      while route del default gw 0.0.0.0 dev $interface ; do
        :
      done

      for i in $router ; do
        route add default gw $i dev $interface
      done
    fi

    echo -n > $RESOLV_CONF
    [ -n "$domain" ] && echo search $domain >> $RESOLV_CONF
    for i in $dns ; do
      echo adding dns $i
      echo nameserver $i >> $RESOLV_CONF
    done
```

```
    ;;
esac

exit 0
```

Assign executable permissions with the following command：

```
chmod 755 /usr/share/udhcpc/default.script
```

6.20.3 Step3：Start 4G dialing

Start the dialing by entering the following command:

```
quectel-CM &
```

If the dialing is successful, the screen will output information such as the IP address, as shown below:

```
root@NanoPC-T4:~# quectel-CM &
[1] 5364
root@NanoPC-T4:~# [05-15_08:23:13:719]
WCDMA&LTE_QConnectManager_Linux&Android_V1.1.34
[05-15_08:23:13:720] quectel-CM profile[1] = (null)/(null)/(null)/0, pincode = (null)
[05-15_08:23:13:721] Find /sys/bus/usb/devices/3-1 idVendor=2c7c idProduct=0125
[05-15_08:23:13:722] Find /sys/bus/usb/devices/3-1:1.4/net/wwan0
[05-15_08:23:13:722] Find usbnet_adapter = wwan0
[05-15_08:23:13:723] Find /sys/bus/usb/devices/3-1:1.4/usbmisc/cdc-wdm0
[05-15_08:23:13:723] Find qmichannel = /dev/cdc-wdm0
[05-15_08:23:13:739] cdc_wdm_fd = 7
[05-15_08:23:13:819] Get clientWDS = 18
[05-15_08:23:13:851] Get clientDMS = 2
[05-15_08:23:13:884] Get clientNAS = 2
[05-15_08:23:13:915] Get clientUIM = 1
[05-15_08:23:13:947] Get clientWDA = 1
[05-15_08:23:13:979] requestBaseBandVersion EC20CEFHLGR06A01M1G_OCPU_BETA1210
[05-15_08:23:14:043] requestSetEthMode QMUXResult = 0x1, QMUXError = 0x46
[05-15_08:23:14:075] requestGetSIMStatus SIMStatus: SIM_READY
[05-15_08:23:14:107] requestGetProfile[1] cmnet///0
[05-15_08:23:14:139] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached,
DataCap: LTE
[05-15_08:23:14:171] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[05-15_08:23:14:235] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached,
DataCap: LTE
[05-15_08:23:14:938] requestSetupDataCall WdsConnectionIPv4Handle: 0xe16e4540
[05-15_08:23:15:002] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[05-15_08:23:15:036] ifconfig wwan0 up
[05-15_08:23:15:052] busybox udhcpc -f -n -q -t 5 -i wwan0
[05-15_08:23:15:062] udhcpc (v1.23.2) started
[05-15_08:23:15:077] Sending discover...
[05-15_08:23:15:093] Sending select for 10.22.195.252...
[05-15_08:23:15:105] Lease of 10.22.195.252 obtained, lease time 7200
[05-15_08:23:15:118] deleting routers
SIOCDELRT: No such process
[05-15_08:23:15:132] adding dns 221.179.38.7
[05-15_08:23:15:132] adding dns 120.196.165.7
```

6.20.4 Test 4G connection

Ping a domain name to see if DNS resolution is already working：

```
root@NanoPC-T4:~# ping www.baidu.com
PING www.a.shifen.com (183.232.231.174) 56(84) bytes of data.
64 bytes from 183.232.231.174 (183.232.231.174): icmp_seq=1 ttl=56 time=74.3 ms
64 bytes from 183.232.231.174 (183.232.231.174): icmp_seq=2 ttl=56 time=25.1 ms
64 bytes from 183.232.231.174 (183.232.231.174): icmp_seq=3 ttl=56 time=30.8 ms
64 bytes from 183.232.231.174 (183.232.231.174): icmp_seq=4 ttl=56 time=29.1 ms
64 bytes from 183.232.231.174 (183.232.231.174): icmp_seq=5 ttl=56 time=29.2 ms
```

6.20.5 Test the speed of 4G

```
wget -O - https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py | python
```

The test results obtained are as follows：

```
Retrieving speedtest.net configuration...
Testing from China Mobile Guangdong (117.136.40.167)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by ChinaTelecom-GZ (Guangzhou) [2.51 km]: 62.726 ms
Testing download
speed.....................................................................................
Download: 32.93 Mbit/s
Testing upload
speed.....................................................................................
.................
Upload: 5.58 Mbit/s
```

6.21 Video Play and Recording

The NanoPi NEO Air have an audio interface (2.54mm pin header) whose pin spec is as follows:

Pin#	Name	Description
1	LL	LINE-OUT Left Channel Output
2	LR	LINE-OUT Right Channel Output
3	MN	Microphone Negative Input
4	MP	Microphone Positive Input

Here is a hardware setup:
Earphone

Make sure your board is connected to an audio device before proceed with the following steps
List audio devices:

```
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: Codec [H3 Audio Codec], device 0: CDC PCM Codec-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Both Allwinner H5 and H3 have an internal codec device which is recognized as an [H3 Audio Codec] card device in the kernel.

Play video:

```
$ aplay /root/Music/test.wav -D plughw:0
```

Parameter -D plughw:0 means device "card 0" is in use. Make sure you use a correctly recognized card device.
Record video:

```
$ arecord -f cd -d 5 test.wav
```

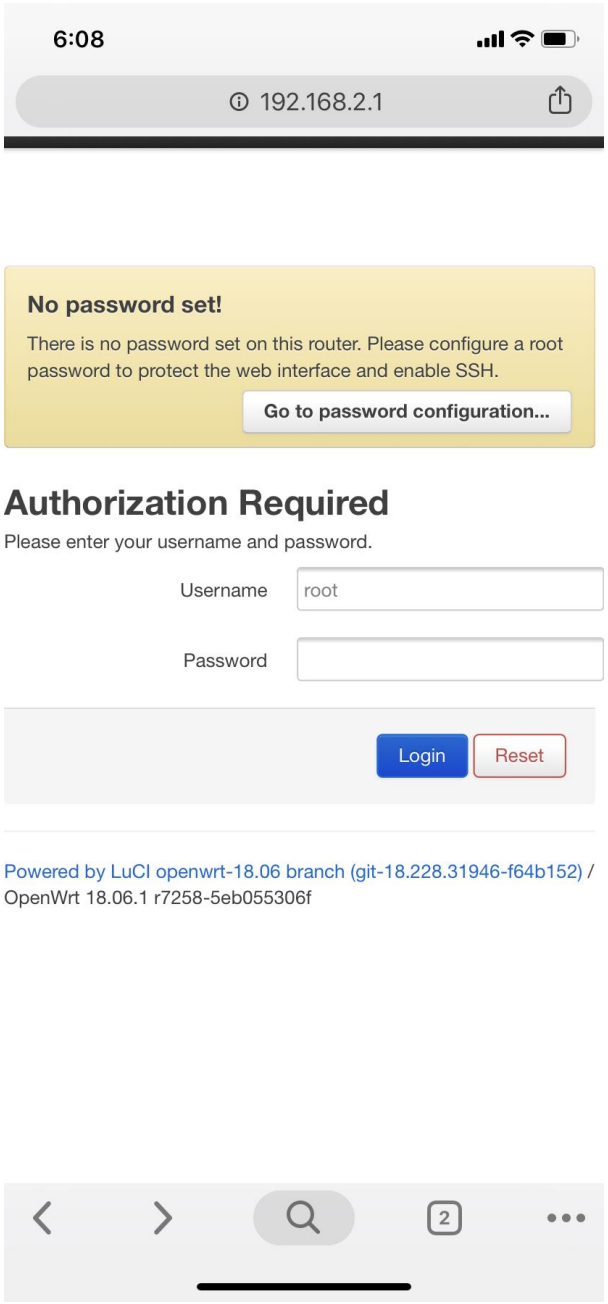
7 Work with OpenWrt

7.1 Introduction

OpenWrt is a highly extensible GNU/Linux distribution for embedded devices.Unlike many other distributions for routers, OpenWrt is built from the ground up to be a full-featured, easily modifiable operating system for embedded devices. In practice, this means that you can have all the features you need with none of the bloat, powered by a modern Linux kernel. For more details you can refer to:OpenWrt website.

7.2 Configure WiFi

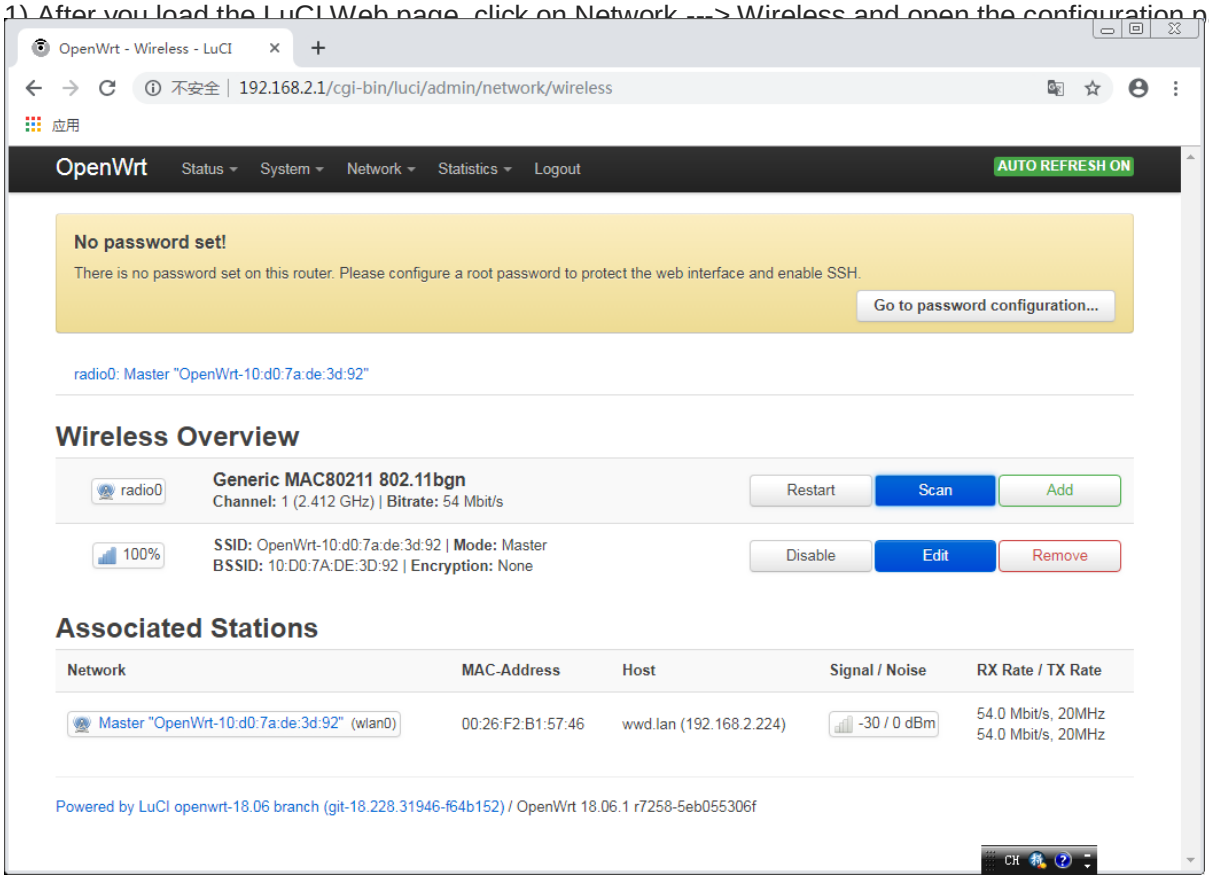
- Make sure you install a WiFi antenna on your board.
- By default the NanoPi NEO Air's OpenWrt is set to AP mode. The default AP hotspot's name is like "OpenWrt-10:d0:7a:de:3d:92" and it doesn't have a password. You can connect a smart phone or PC to it.
- The board's AP hotspot IP address falls into 192.168.2.x. Open a browser in your phone or PC, type 192.168.2.1 in the address bar and you will be able to visit the LuCI GUI:



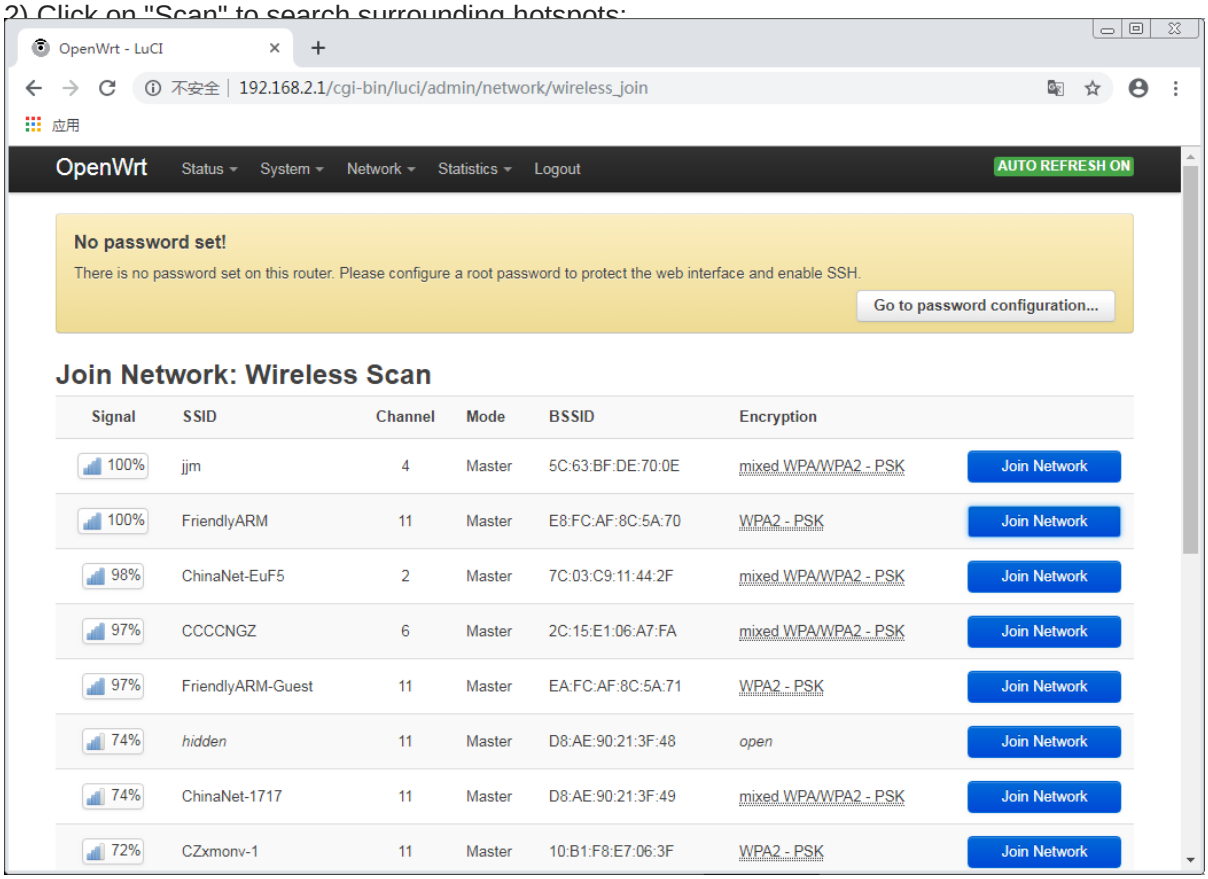
The default user is root without a password. You can click on "Login" to sign in.

- The NEO-Air has only one network device: wlan0. Therefore if you want to visit the internet from the board you need to switch from AP mode to STA mode. Here are the steps.

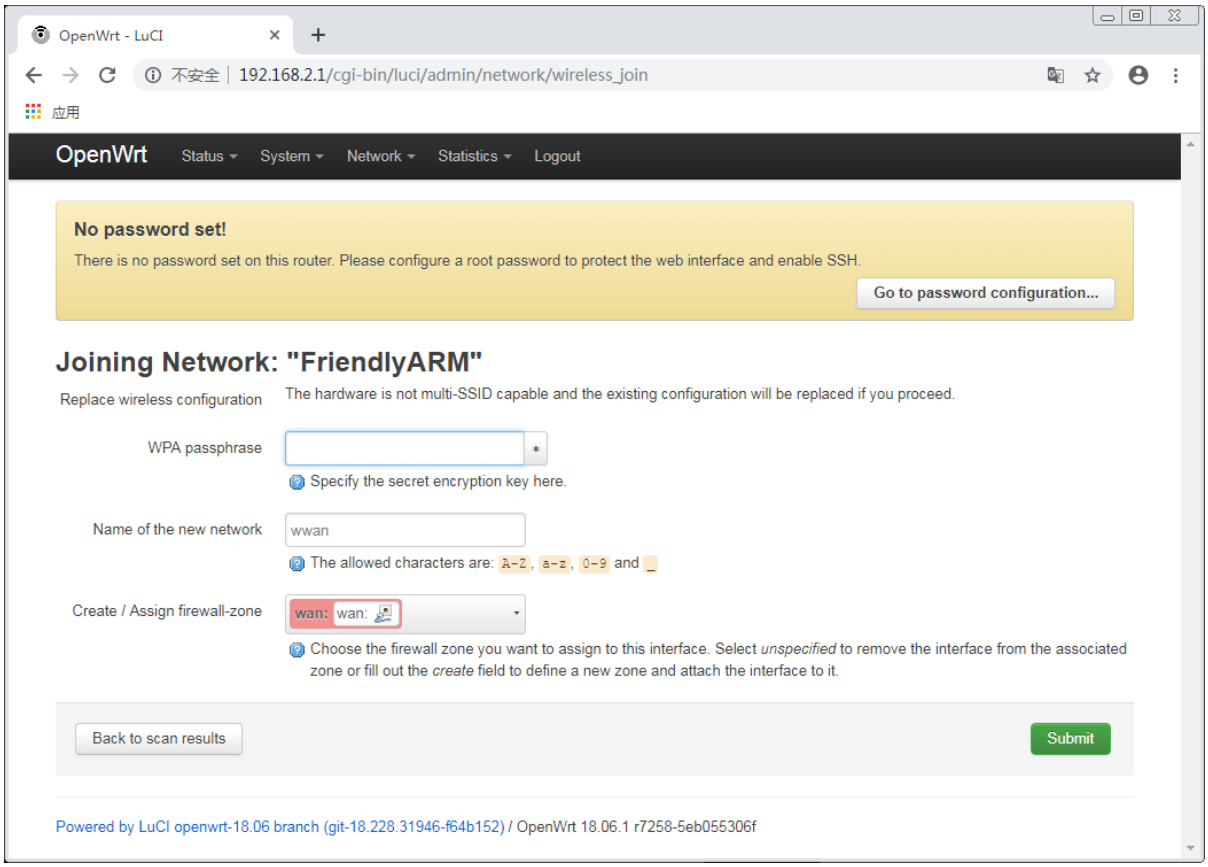
1) After you load the LuCI Web page, click on Network ---> Wireless and open the configuration page:



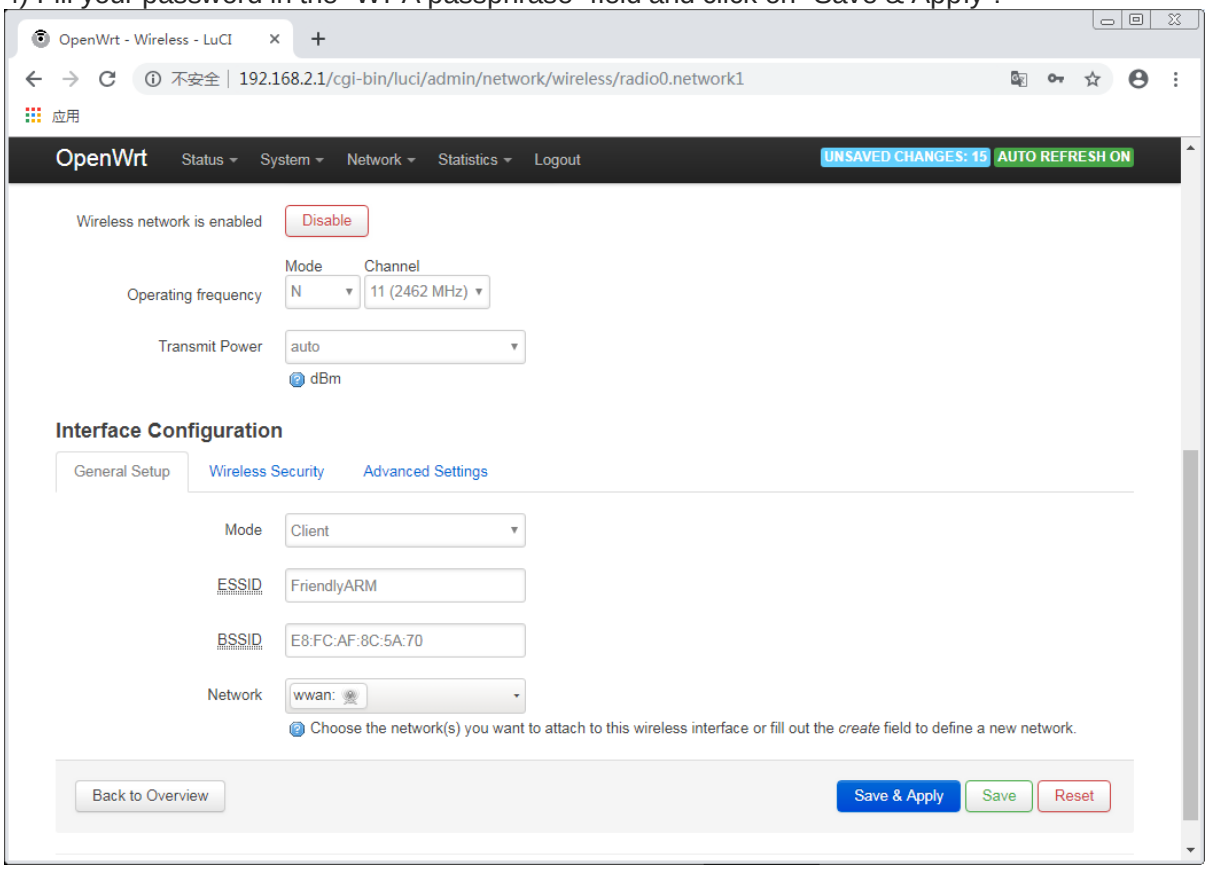
2) Click on "Scan" to search surrounding hotspots:



3) Click on "Join network":



4) Fill your password in the "WPA passphrase" field and click on "Save & Apply":



Your board will switch from AP mode to STA mode, "OpenWrt-10:d0:7a:de:3d:92" will be removed” and the page will be disconnected. In 30 seconds you can check if your board's wlan0 will be assigned an IP.

7.3 System Login

- Login via Serial Communication

Here is a hardware setup:
Connect the following serial communication board to your board and power on the whole system from the serial board's DC or MicroUSB port:

By default you will login as root without a password. You can set your password by commanding "passwd".

```
BusyBox v1.29.3 () built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt SNAPSHOT, r8132-9f105ce
 -----------------------------------------------------
=== WARNING! =====================================
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
--------------------------------------------------
root@OpenWrt:/#
```

When your board loads OS for the first time the TF card's rootfs system in your board will be automatically partitioned to its max capacity:

```
Begin: Resizing ext4 file system on /dev/mmcblk0p2 ... Model: SD SP32G (sd/mmc)
Disk /dev/mmcblk0: 100%
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start  End[    4.687657]  mmcblk0: p1 p2
    Size   Type     File system  Flags
 1      0.08%  0.21%  0.13%  primary  fat16
 2      0.21%  100%   99.8%  primary  ext4

resize2fs 1.42.12 (29-Aug-2014)
```

Be patient for this process to be done.

- Login via SSH

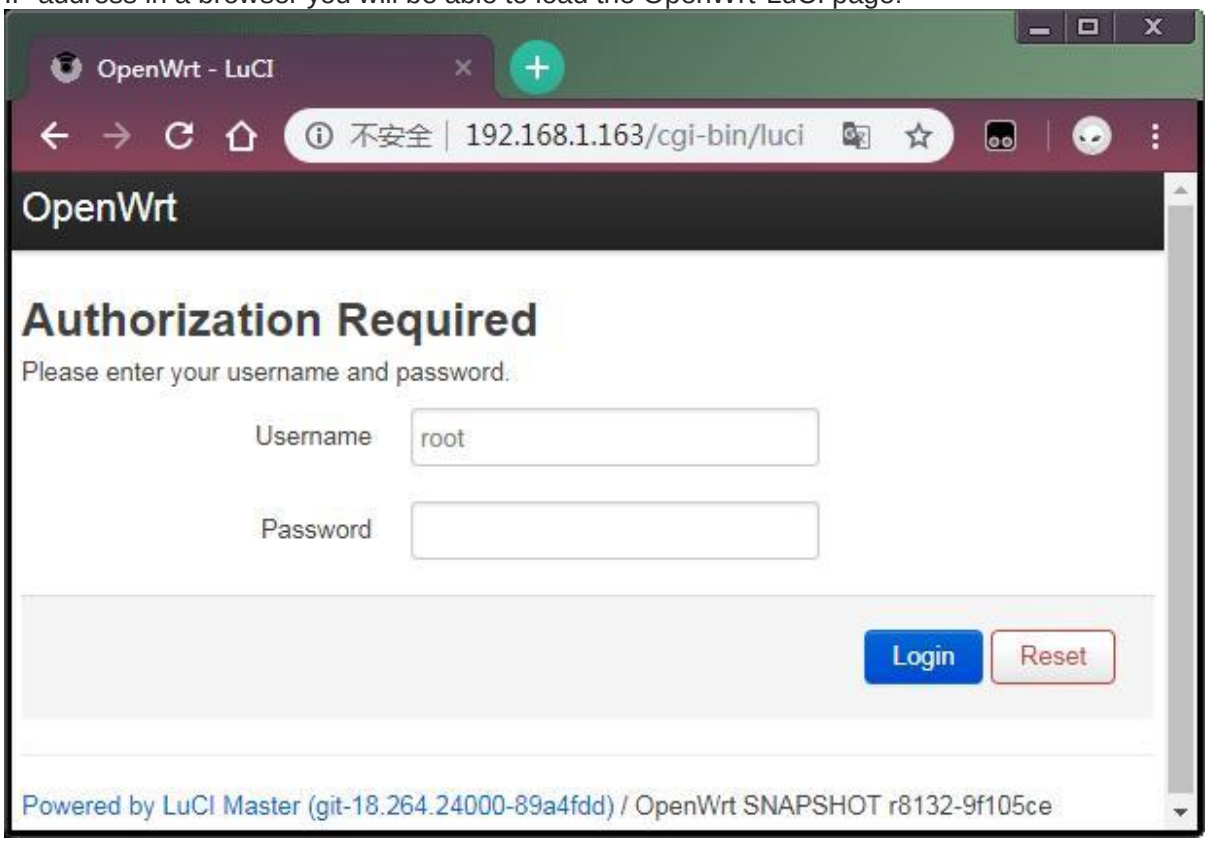
Make sure you connect your board to the internet following the steps in <Configure WiFi> and your board's wlan0 IP address is 192.168.1.163. Run the following commands to login via SSH:

```
$ ssh root@192.168.1.163
```

By default you will login as root without a password.

- Login via Web

OpenWrt supports web login via the LuCI Web GUI. Configure your system following the steps in <Login via SSH>. In our test the board's wlan0 IP address was 192.168.1.163. Type your IP address in a browser and you will be able to load the OpenWrt-LuCI page:
If your board is connected to the internet and your board's wlan0 is assigned an IP address, after you type the IP address in a browser you will be able to load the OpenWrt-LuCI page:



By default you will login as root without a password. After click on "Login" you will sign in.

7.4 Manage Software Packages

OpenWrt has a package management utility: opkg. You can get its details by running the following command:

```
$ opkg
Package Manipulation:
        update                  Update list of available packages
        upgrade <pkgs>          Upgrade packages
        install <pkgs>          Install package(s)
        configure <pkgs>        Configure unpacked package(s)
        remove <pkgs|regexp>    Remove package(s)
        flag <flag> <pkgs>      Flag package(s)
         <flag>=hold|noprune|user|ok|installed|unpacked (one per invocation)

Informational Commands:
        list                    List available packages
        list-installed          List installed packages
        list-upgradable         List installed and upgradable packages
        list-changed-conffiles  List user modified configuration files
```

```
        files <pkg>             List files belonging to <pkg>
        search <file|regexp>    List package providing <file>
        find <regexp>           List packages whose name or description matches
<regexp>
        info [pkg|regexp]       Display all info for <pkg>
        status [pkg|regexp]     Display all status for <pkg>
        download <pkg>          Download <pkg> to current directory
...
```

These are just part of the manual. Here are some popular opkg commands.

- Update Package List

Before you install a package you'd better update the package list:

```
$ opkg update
```

- Check Available Packages

```
$ opkg list
```

At the time of writing there are 3241 packages available.

- Check Installed Packages

```
$ opkg list-installed
```

At the time of writing 124 packages have been installed.

- Install/Delete Packages:

```
$ opkg install <pkgs>
$ opkg remove <pkgs>
```

- Check Files Contained in Installed Packages:

```
$ opkg files <pkg>
```

- Install Chinese Language Package for LuCI:

```
$ opkg install luci-i18n-base-zh-cn
```

- Check Changed Files:

```
$ opkg list-changed-conffiles
```

- Reference Links:
  - openwrt opkg

7.5 Check System Status

- Check CPU Temperature & Frequency:

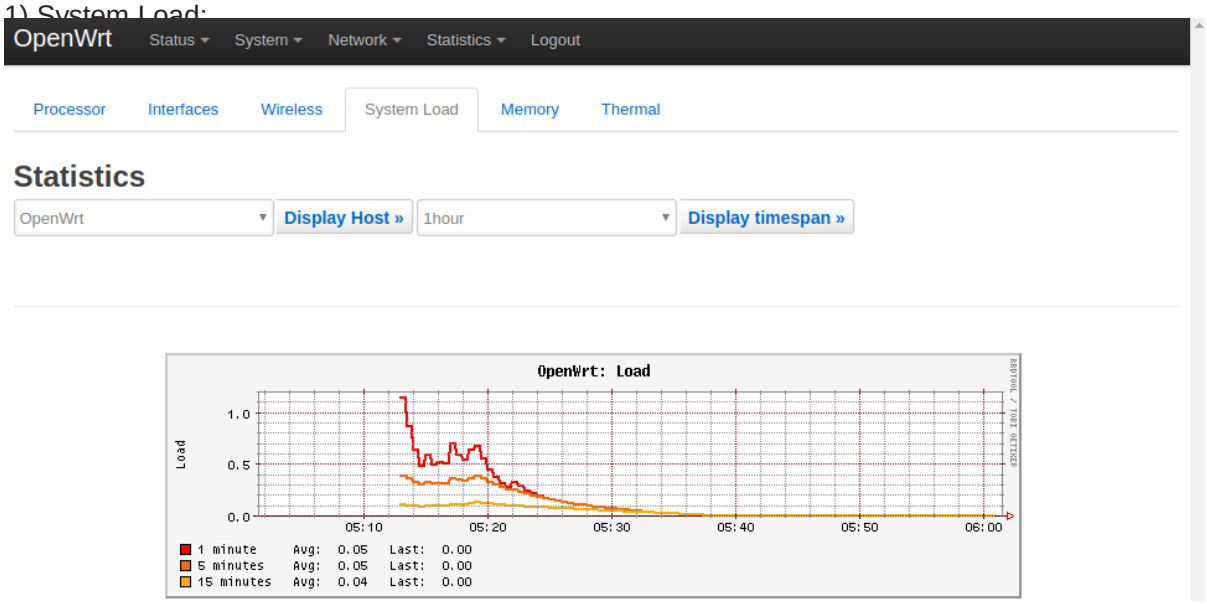
```
$ cpu_freq
CPU0 online=1 temp=26581 governor=ondemand cur_freq=480000
CPU1 online=1 temp=26581 governor=ondemand cur_freq=480000
CPU2 online=1 temp=26581 governor=ondemand cur_freq=480000
CPU3 online=1 temp=26581 governor=ondemand cur_freq=480000
```

These messages mean four CPU cores are working online. Each core's temperature is 26.5 degrees. Each core's governor is on demand and the frequency is 480 MHz。

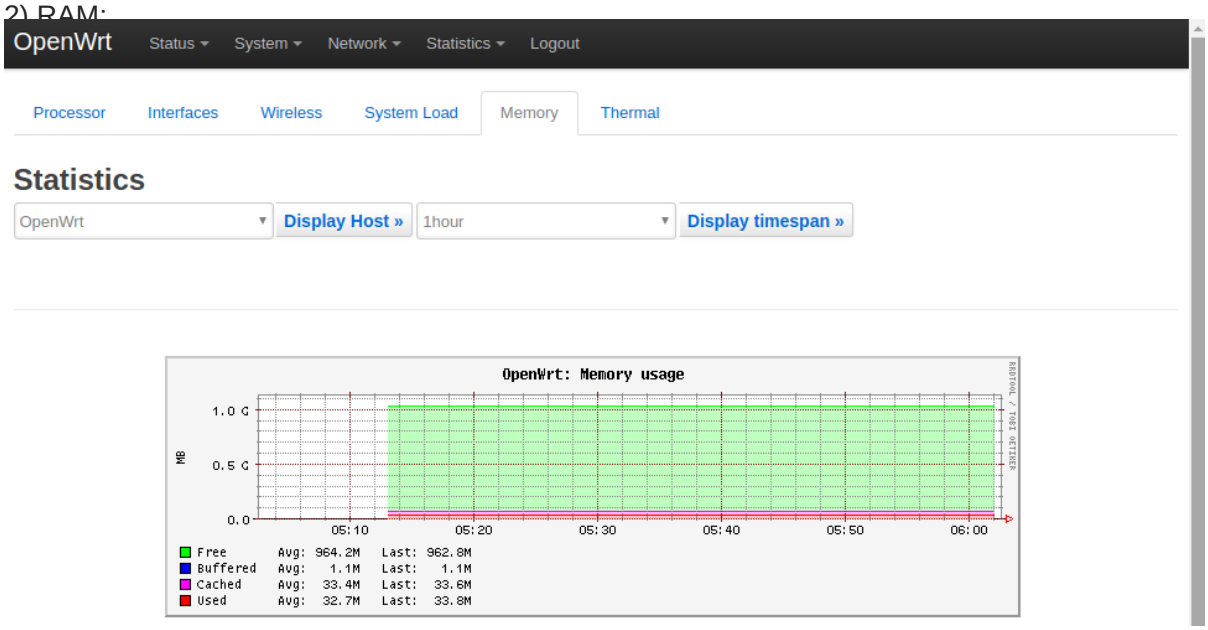
- Check System Status on OpenWrt-LuCI Web Page:

After you load the OpenWrt-LuCI page, click on "Statistics ---> Graphs" and you will be able to check all the statistics:

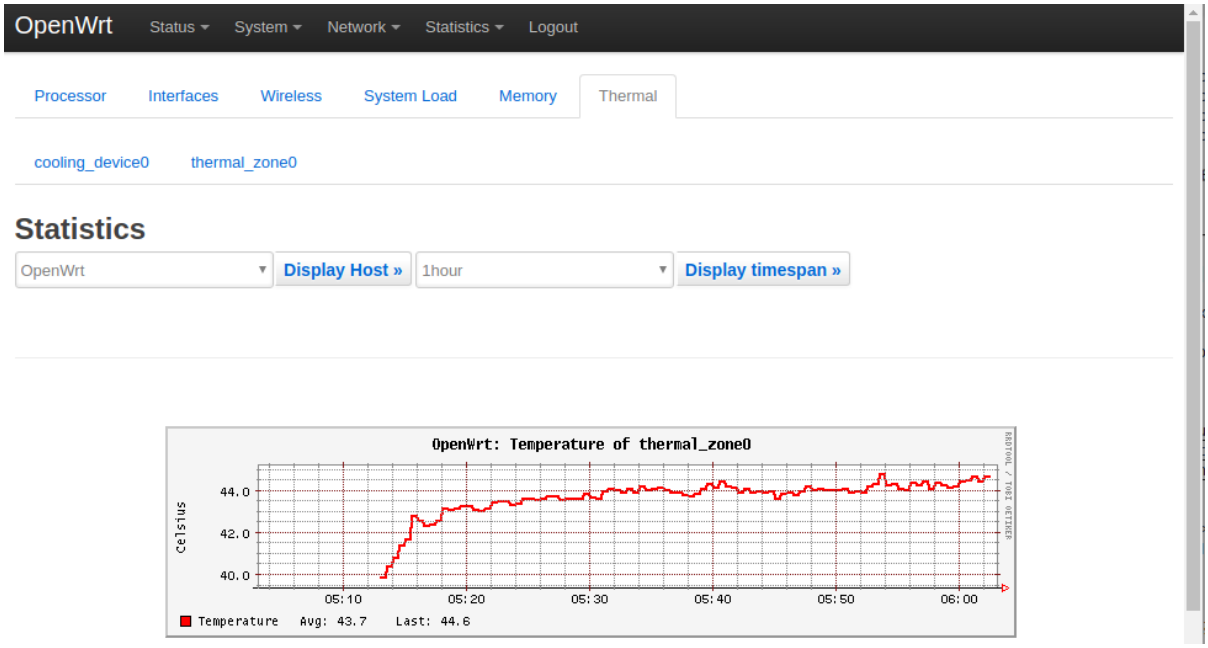
1) System Load:



2) RAM:



3) CPU Temperature:



"Statistics" is collected by the luci-app-statistics package. The luci-app-statistics package uses the Collectd utility to collect statistics and presents them using the RRDtool utility.
If you want to get more statistics you can install various collectd-mod-* packages. All collectd-mod-* packages use the same configuration file: /etc/config/luci_statistics.

- Reference Links:
  - openwrt luci_app_statistics
  - openwrt statistics.chart.public
  - openwrt statistic.custom

8 Make Your Own FriendlyCore

8.1 Use Linux-4.14 BSP

The NanoPi NEO Air has gotten support for kernel Linux-4.14. For more details about how to use mainline U-boot and Linux-4.14 refer to :Building U-boot and Linux for H5/H3/H2+

8.2 Use Allwinner's Linux-3.4 BSP

8.2.1 Preparations

Visit this link download link and enter the "sources/nanopi-H3-bsp" directory and download all the source code.Use the 7-zip utility to extract it and a lihee directory and an Android directory will be generated. You can check that by running the following command:

```
$ ls ./
android lichee
```

Or you can get it from our github:

```
$ git clone https://github.com/friendlyarm/h3_lichee.git lichee
```

Note: "lichee" is the project name named by Allwinner for its CPU's source code which contains the source code of U-boot, Linux kernel and various scripts.

8.2.2 Install Cross Compiler

Visit this site download link, enter the "toolchain" directory, download the cross compiler "gcc-linaro-arm.tar.xz" and copy it to the "lichee/brandy/toochain/" directory.

8.2.3 Compile lichee Source Code

Compilation of the H3's BSP source code must be done under a PC running a 64-bit Linux.The following cases were tested on Ubuntu-14.04 LTS-64bit:

```
$ sudo apt-get install gawk 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 u-boot-tools
```

Enter the lichee directory and un the following command to compile the whole package:

```
$ cd lichee/fa_tools
$ ./build.sh -b nanopi-air -p linux -t all
```

After this compilation succeeds a u-boot, Linux kernel and kernel modules will be generated.
Note: the lichee directory contains a cross-compiler we have setup. When the build.sh script runs it will automatically call this cross-compiler.

Type the following command to update the U-boot on your TF card:

```
$ cd lichee/fa_tools/
$ ./fuse.sh -d /dev/sdX -p linux -t u-boot
```

Note: you need to replace "/dev/sdx" with the device name in your system.
The boot.img and kernel modules are under the "linux-3.4/output" directory. You can copy the new boot.img file to your TF card's boot partition.

8.3 Compile U-boot

You can compile u-boot individually by using the following command:

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-air -p linux -t u-boot
```

The gen_script.sh script patches the U-boot with Allwinner features. A U-boot without these features cannot work.
Type the following command to update the U-boot on your TF card:

```
$ cd lichee/fa_tools/
$ ./fuse.sh -d /dev/sdX -p linux -t u-boot
```

Note: you need to replace "/dev/sdx" with the device name in your system.

8.4 Compile Linux Kernel

If you want to compile the Linux kernel run the following command:

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-air -p linux -t kernel
```

After the compilation is done a uImage and its kernel modules will be generated under "linux-3.4/output".

8.5 Clean Source Code

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-air -p linux -t clean
```

9 3D Printing Files

NanoPi NEO Air V1.0 3D printing files
[http:// NanoPi NEO Air V1.1 3D printing files]

10 Other OS Support

10.1 DietPi_NanoPiNEO-armv7-(Jessie)

DietPi is an extremely lightweight Debian Jessie OS. Its image file starts at 400MB and nearly 3x lighter than 'Raspbian Lite'.It is pre-installed with DietPi-RAMLog. These features enable users to get the best performance of a device.
The following steps are for reference only. FriendlyElec doesn't provide technical support for them.
Installation guide:

- Download the image file "DietPi_NanoPiNEO-armv7-(Jessie)" from DietPi_NanoPiNEO-armv7-(Jessie)
- Extract the package and use the win32diskimager to write it to a MicroSD card under Windows.
- Insert this MicroSD card to your NanoPi NEO and power up.

Username:root , Password: dietpi

11 Developer's Guide

- System Development
  - Building U-boot and Linux for H5/H3/H2+
  - How to Build FriendlyWrt
  - Qt dev: How to Build, Install and Setting Qt Application

- Image Utilities
  - How to make your own SD-bootable ROM
  - How to use overlayfs on Linux
  - EFlasher

- System Configurations
  - npi-config
  - Use NetworkManager to configure network settings

- Hardware Access
  - WiringNP: NanoPi NEO/NEO2/Air GPIO Programming with C
  - RPi.GPIO : NanoPi NEO/NEO2/Air GPIO Programming with Python
  - Hardware Misc
  - Matrix
  - BakeBit
  - HATs&Docks

12 Resources

- Schematics
  - NanoPi-NEO-Air-1608-Schematic.pdf
  - NanoPi-NEO-Air-V1.1-1708-Schematic.pdf
- Dimensional Diagram
  - pcb file in dxf format
  - NanoPi-NEO-Air-V1.1-1708 pcb file in dxf format
- H3 datasheet
  - Allwinner_H3_Datasheet_V1.2.pdf

The following BakeBit modules can work with BakeBit - NanoHat Hub:

- 1.Button
- 2.Buzzer
- 3.Green LED
- 4.JoyStick
- 5.LED Bar
- 6.Light Sensor
- 7.OLED
- 8.Red LED
- 9.Rotary Angle Sensor
- 10.Servo
- 11.Sound Sendor
- 12.Ultrasonic Ranger

- Matrix Modules & Wiki Sites:
  - Button
  - LED
  - A/D Converter
  - Relay
  - 3-Axis Digital Accelerometer
  - 3-Axis Digital Compass
  - Temperature Sensor
  - Temperature & Humidity Sensor
  - Buzzer
  - Joystick
  - I2C(PCF8574)+LCD1602
  - Sound Sensor
  - Ultrasonic Ranger
  - GPS
  - Matrix - Compact Kit
  - Fire Sensor
  - CAM500A Camera
  - BAll Rolling Switch
  - 2'8 SPI Key TFT 2.8" SPI LCD
  - IR Counter
  - IR Receiver
  - L298N Motor Driver
  - MQ-2 Gas Sensor
  - MQ-3 Gas Sensor
  - One_Touch_Sensor
  - _Photoresistor
  - _Potentiometer

- Pressure & Temperature Sensor
- RGB LED
- RTC
- Rotary Encoder
- Soil Moisture Sensor
- Thermistor
- USB WiFi
- Water Sensor