# NanoPi M1 Plus

[Download NanoPi M1 Plus Documents]



## 1 Introduction

- The NanoPi M1 Plus is designed and developed by FriendlyElec for professionals, enterprise users, makers and hobbyists. It is only two thirds the size of a Raspberry Pi. FriendlyElec has made a Debian, Ubuntu-Core and Android images ready for it.
- The NanoPi M1 Plus uses the Allwinner H3 Soc. It integrates Gbps Ethernet, IR receiver, video/audio output, WiFi & Bluetooth, onboard microphone and supports DVP/Camera/HDMI and CVBS. It has a serial debug port. Its GPIO pins are compatible with those of a Raspberry Pi.

## 2 Hardware Spec

- CPU: Allwinner H3, Quad-core Cortex-A7@1.2GHz
- GPU: Mali400MP2@600MHz，Supports OpenGL ES2.0
- DDR3 RAM: 1GB
- eMMC: 8GB
- Wireless: 802.11 b/g/n
- Bluetooth: 4.0 dual mode
- Antenna Interface: Shared by WiFi and Bluetooth, IPX interface
- Connectivity: 10/100/1000M Ethernet
- Audio: 3.5mm jack/Via HDMI
- Microphone: onboard microphone
- IR: onboard IR receiver
- USB Host: USB 2.0 x 3, 2 x USB Type A and 1 x 2.54mm pitch pin-header
- MicroSD Slot: x1
- MicroUSB: power input and data transmission, OTG
- Audio Output: HDMI 1.4 1080P, CVBS
- DVP Camera Interface: 24pin, 0.5mm pitch FPC seat
- Serial Debug Port: 4Pin, 2.54mm pitch pin-header
- GPIO： 40pin, 2.54mm pitch pin-header, compatible with RasberryPi 2's GPIO. It contains UART, SPI, I2C, I2S/PCM, SPDIF-OUT and IO
- User Button: 1 x Power Button and 1 x Reset Button

- LED: 1 x Power LED and 1 x System Status LED
- PCB Dimension: 64 x 60 mm, ENIG
- Power Supply: DC 5V/2A
- Working Temperature: -30℃ to 80℃
- OS/Software: u-boot, Debian, Ubuntu-Core, eflasher, Android

# 3 Software Features

## 3.1 uboot

- mainline uboot released on May 2017

## 3.2 UbuntuCore 16.04

- mainline kernel: Linux-4.14
- rpi-monitor: check system status and information
- npi-config: system configuration utility for setting passwords, language, timezone, hostname, SSH and auto-login
- networkmanager: manage network
- welcome window with basic system information and status
- auto-login with user account "pi" with access to npi-config
- supports USB WiFi module: refer to #Test USB WiFi
- fixed MAC address

## 3.3 Eflasher

- supports flashing OS image to eMMC

## 3.4 Debian

- rpi-monitor: check system status and information
- npi-config: system configuration utility for setting passwords, language, timezone, hostname, SSH and auto-login
- supports Ethernet
- supports USB WiFi module: refer to #Test USB WiFi
- supports FriendlyElec's CAM202 USB camera
- supports FriendlyElec's CAM500B DVP camera

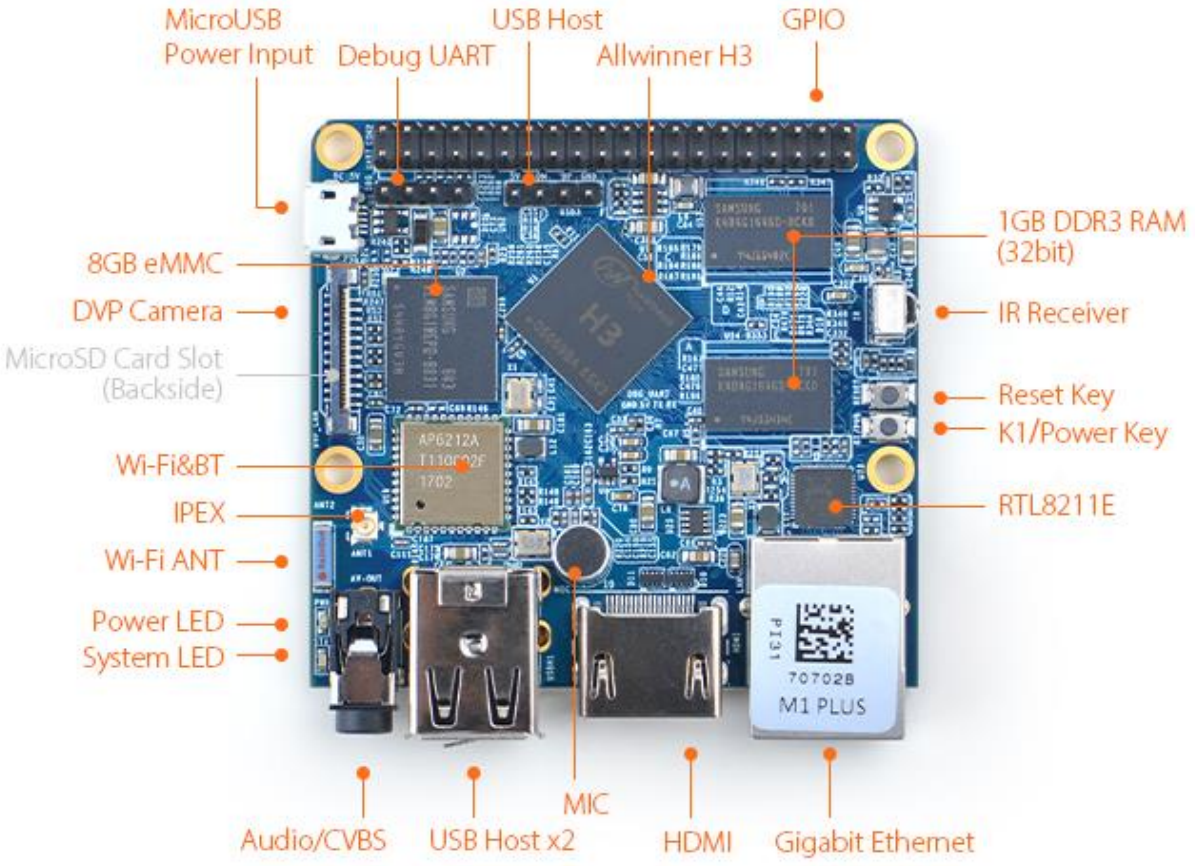
## 3.5 Debian for NAS Dock

- supports FriendlyElec's NAS Dock

## 3.6 Android

- supports Ethernet
- supports WiFi

## 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 |
| 25 | GND | | 26 | SPDIF-OUT/GPIOA17 | 17 |
| 27 | I2C1_SDA/GPIOA19/PCM0_CLK/I2S0_BCK | 19 | 28 | I2C1_SCL/GPIOA18/PCM0_SYNC/I2S0_LRCK | 18 |
| 29 | GPIOA20/PCM0_DOUT/I2S0_SDOUT | 20 | 30 | GND | |
| 31 | GPIOA21/PCM0_DIN/I2S0_SDIN | 21 | 32 | NC | |
| 33 | NC | | 34 | GND | |
| 35 | NC | | 36 | NC | |
| 37 | GPIOA9 | 9 | 38 | NC | |
| 39 | GND | | 40 | NC | |

- **Debug Port（UART0）**

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

- **USB Pin Header**

| Pin# | Name |
|---|---|
| 1 | 5V |
| 2 | DM |
| 3 | DP |
| 4 | GND |

- **DVP Camera Interface Pin Description**

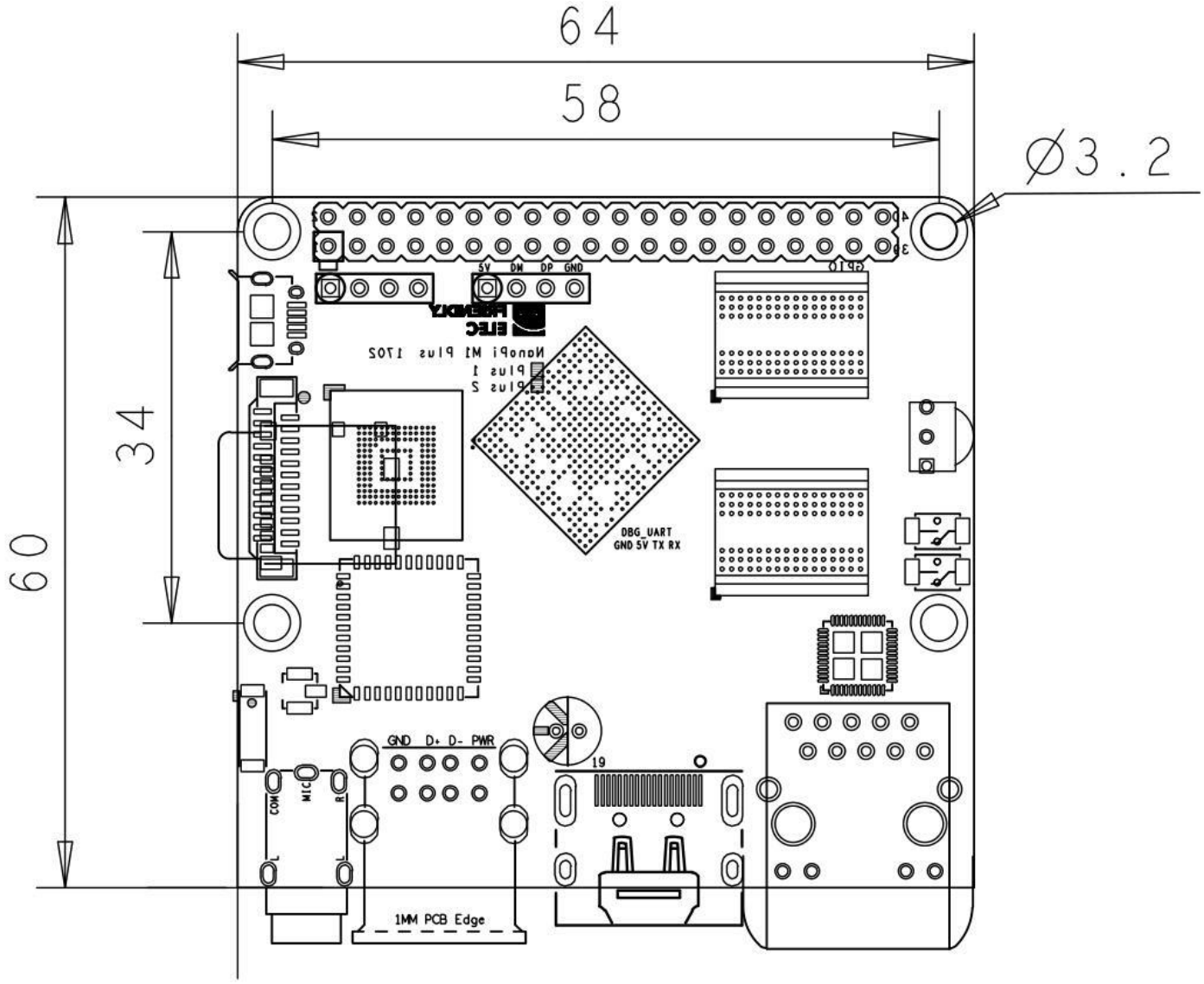
| Pin# | Name | Description |
|---|---|---|
| 1, 2 | SYS_3.3V | 3.3V Output, it can be used to power camera modules |
| 7,9,13,15,24 | GND | Ground, 0V |
| 3 | I2C2_SCL | I2C clock signal |

| | | |
|---|---|---|
| 4 | I2C2_SDA | I2C data signal |
| 5 | GPIOE15 | regular GPIO, used to control connected camera modules |
| 6 | GPIOE14 | regular GPIO, used to control connected camera modules |
| 8 | MCLK | Clock signals output to camera modules |
| 10 | NC | Not connected |
| 11 | VSYNC | vertical synchronization |
| 12 | HREF/HSYNC | horizontal synchronization==Make Your Own Debian/Ubuntu====Make Your Own Debian/Ubuntu== |
| 14 | PCLK | peripheral clock |
| 16-23 | Data bit7-0 | data bits |

**Notes**

1. SYS_3.3V: 3.3V power output
2. VDD_5V: 5V power output5V. When the external device's power is greater than the MicroUSB's the external device is charging the board otherwise the board powers the external device.The input range is 4.7V ~ 5.5V
3. All pins are 3.3V, output current is 5mA
4. For more details refer to the document:NanoPi-M1-Plus-1702-Schematic.pdf

## 4.2 Board Dimension




For more details please refer to the document:pcb in dxf format

# 5 Get Started

## 5.1 Essentials You Need

Before starting to use your NanoPi M1 Plus get the following items ready

- NanoPi M1 Plus
- microSD Card/TFCard: Class 10 or Above, minimum 8GB SDHC
- microUSB power. A 5V/2A power is a must
- HDMI monitor
- USB keyboard, mouse and possible a USB hub(or a TTL to serial board)
- A host computer running Ubuntu 16.04 64 bit system

## 5.2 TF Cards We Tested

To make your NanoPi M1 Plus 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 (under the "officail-ROMs" diirectory) and the flashing utility (under the "tools" diirectory):

| Image Files: | |
|---|---|
| nanopi-m1-plus_sd_friendlycore-xenial_3.4_armhf_YYYYMMDD.img.zip | FriendlyCore, kernel:Linux-3.4 |
| nanopi-m1-plus_sd_friendlycore-xenial_4.14_armhf_YYYYMMDD.img.zip | FriendlyCore, kernel:Linux-4.14 |
| nanopi-m1-plus_sd_debian-jessie_3.4_armhf_YYYYMMDD.img.zip | Debian-Desktop, kernel:Linux-3.4 |
| nanopi-m1-plus_sd_debian-jessie_4.14_armhf_YYYYMMDD.img.zip | Debian-Desktop, kernel:Linux-4.14 |
| nanopi-m1-plus_sd_friendlywrt_4.14_armhf_YYYYMMDD.img.zip | Base on OpenWrt, kernel:Linux-4.14 |

| | |
|---|---|
| nanopi-m1-plus_eflasher_debian-jessie_4.14_armhf_YYYYMMDD.img.zip | eflasher image, for flashing Debian(Linux-4.14) to eMMC |
| nanopi-m1-plus_eflasher_friendlycore-xenial_4.14_armhf_YYYYMMDD.img.zip | eflasher image, for flashing FriendlyCore(Linux-4.14) to eMMC |
| nanopi-m1-plus_eflasher_openwrt_4.14_armhf_YYYYMMDD.img.zip | eflasher image, for flashing OpenWrt(Linux-4.14) to eMMC |
| nanopi-m1-plus_eflasher_friendlycore-xenial_3.4_armhf_YYYYMMDD.img.zip | eflasher image, for flashing FriendlyCore(Linux-3.4) to eMMC |
| nanopi-m1-plus_sd_android_YYYYMMDD.img.zip | Android, kernel:Linux-3.4 |
| Flash Utility: | |
| win32diskimager.rar | Windows utility. Under Linux users can use "dd" |
| PhoenixCard_V310.rar | Windows utility for flashing Android image. Attention: the "dd" command under Linux doesn't work for flashing Android image |
| HDDLLF.4.40.exe | Windows utility for formatting a TF card |

*5.3.2 Comparison of Linux-3.4 and Linux-4.14*

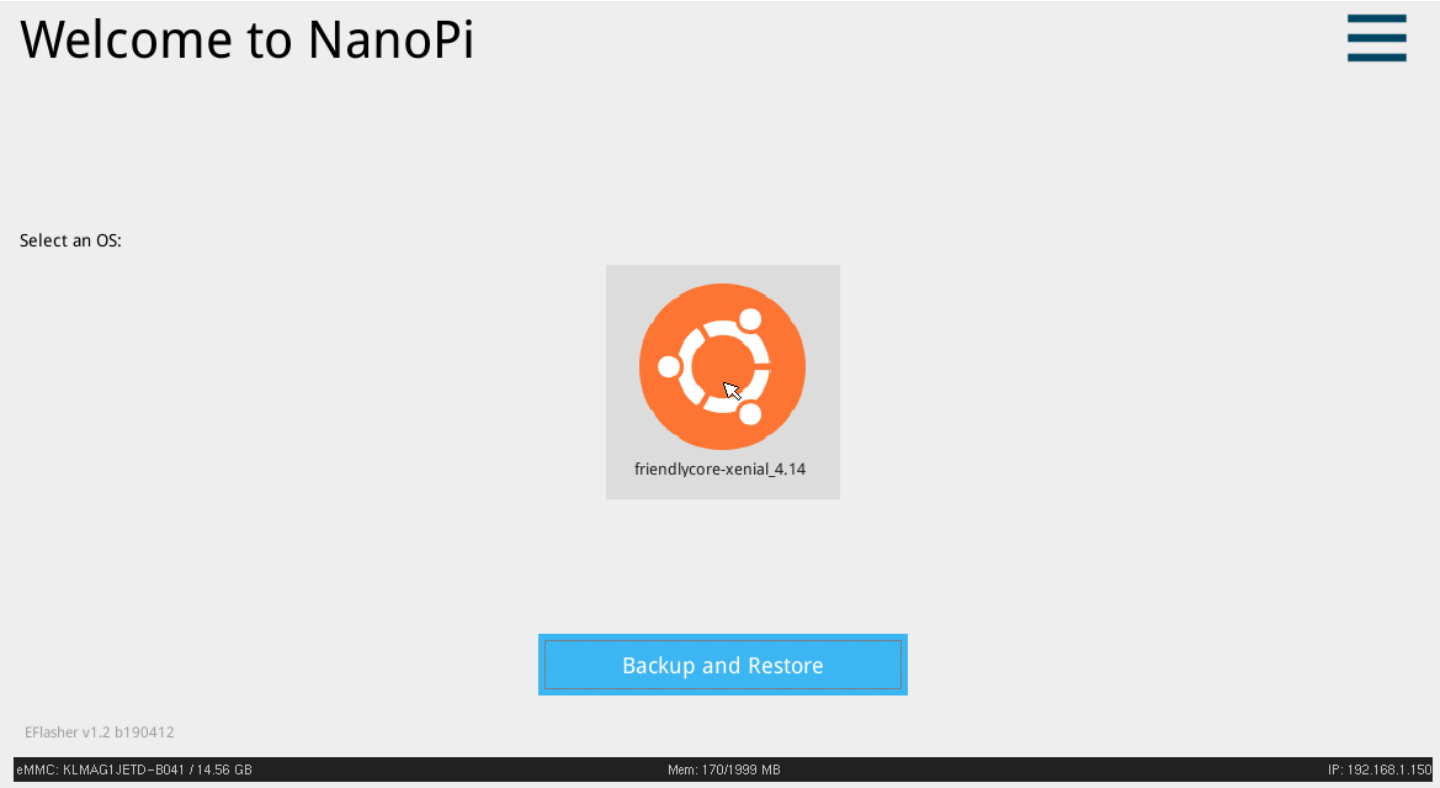
- Our Linux-3.4 is provided by Allwinner. Allwinner has done a lot of customization work which on one hand contains many features and functions but on the other hand incurs overheat issues. If your application needs to use VPU or GPU you need to use the 3.4 kernel based ROM and use a heat sink together 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+

5.3.2.1 Flash to eMMC

5.3.2.1.1 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".

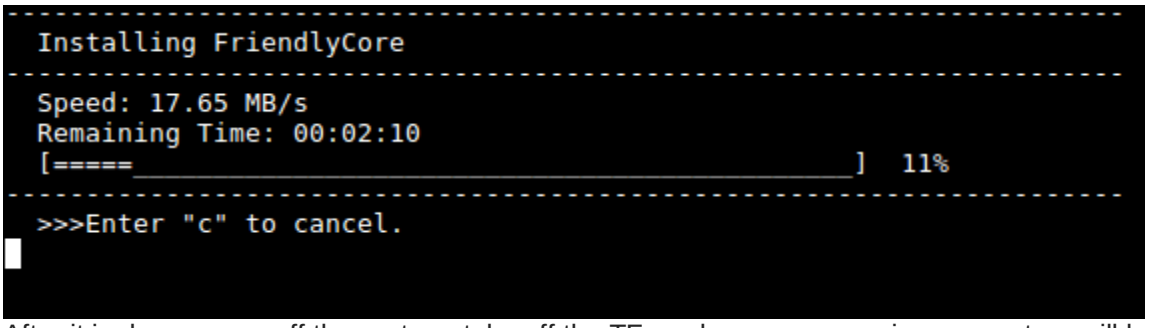
We take "nanopi-m1-plus_eflasher_friendlycore-xenial_4.14_armhf_YYYYMMDD.img" as an example. After you run the "eflasher" command you will see the following messages:

```
------------------------------------------------------------------------
  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:

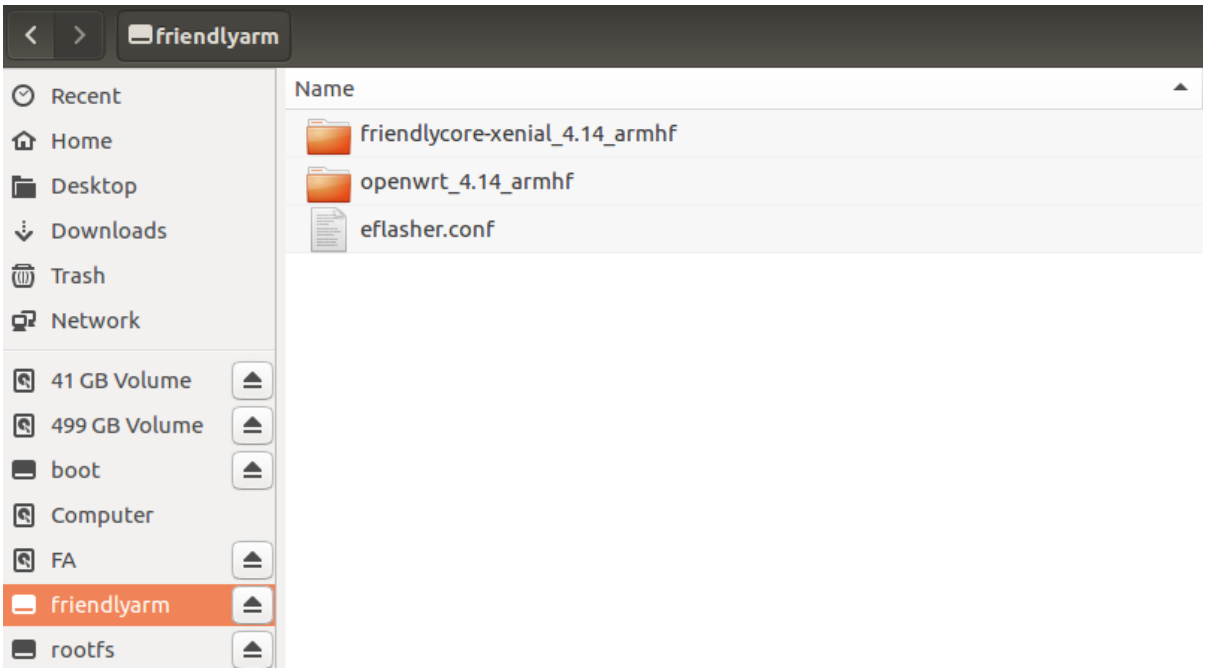
```
------------------------------------------------------------------------
  Ready to Go with FriendlyCore
------------------------------------------------------------------------
  Ready to install
  Version:
                2019-04-25
  Path:
                /mnt/sdcard/friendlycore-xenial_4.14_armhf
  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:




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.

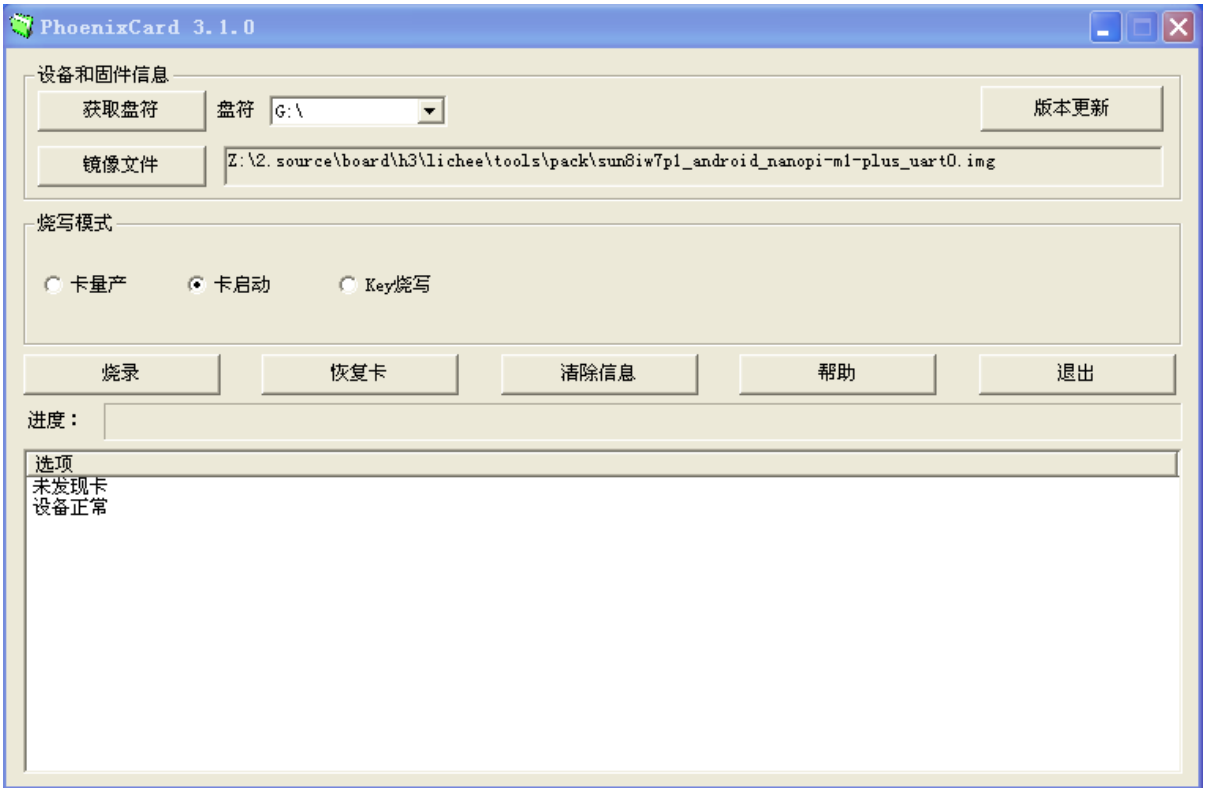



*5.3.3 Android*

5.3.3.1 Flash to TF

Note:before make a MicroSD card to an Android image card you need to format this card.

- On a Windows PC run the HDDLLF.4.40 utility as administrator. Insert a TF card(at least 8G) into this PC and format it. After formatting is done take out the TF card, insert it into the PC again and format it with Windows internal format utility to format it to FAT32. After this formatting is done take out the card.
- Extract the the Android image and PhoenixCard_V310.rar . Insert the TF card you made in the previous step into a Windows PC and run the PhoenixCard_V310 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.



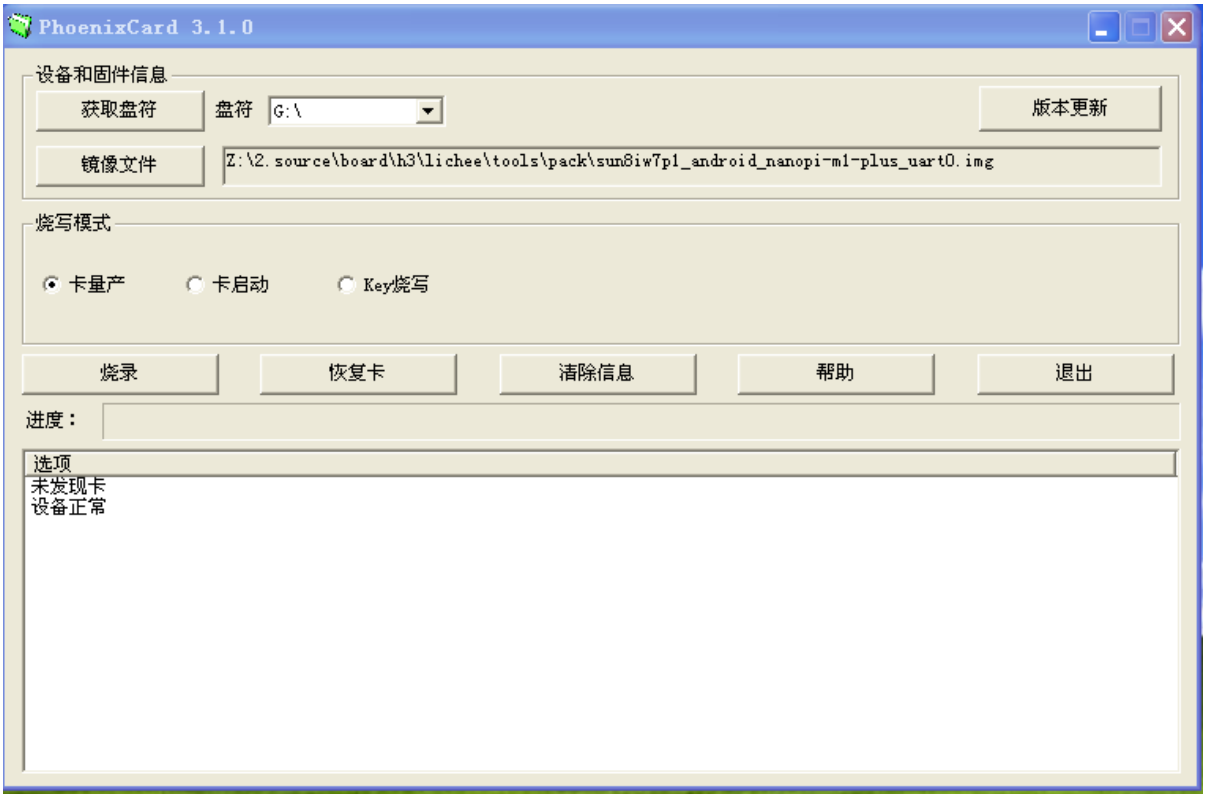
(In the screenshot an Android image file for the NanoPi M1 Plus was selected. You need to select a correct image file for your board.)

- Insert this card into your board' 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.

5.3.3.2 Flash to eMMC

5.3.3.2.1 Install Android to eMMC from TF Card

- Extract an Android image file and the flashing utility PhoenixCard_V310.rar, insert a TF card(at least 8G) to a windows PC and run PhoenixCard as administrator. On the PhoenixCard Window select your TF card's device and your wanted Android image file, set the mode to "卡量产(factory production)" and click on "烧录(Flash)" to start flashing the Android image to TF card.



(In the screenshot an Android image file for the NanoPi M1 Plus was selected. You need to select a correct image file for your board.)

- After flashing is done, the TF card has the ability to burn the Android system to eMMC. Take out the TF card and insert it to your board's TF card slot. Connect your board to an HDMI monitor, power on your board(note:you need a 5V/2A power adapter) and you will see a green process bar on the HDMI monitor. After flashing is done take out the TF card and reboot your board and it will be rebooted from eMMC.

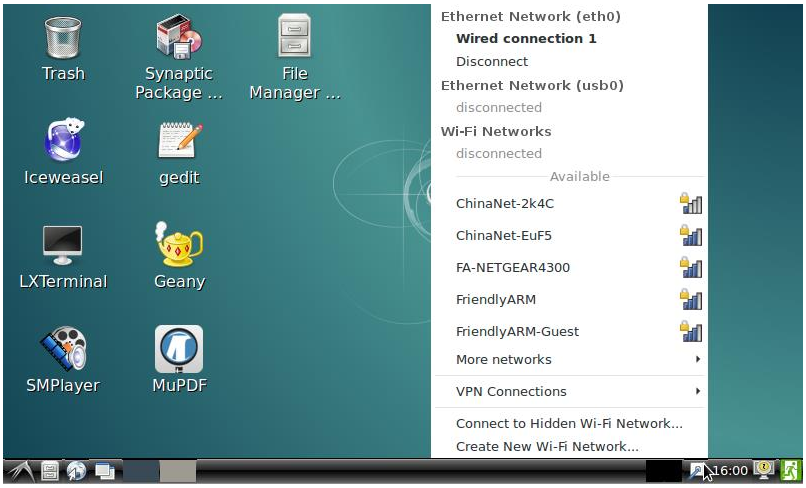
# 6 Working with Debian

## 6.1 Ethernet Connection

- If the board is connected to a network via Ethernet before it is powered on, it will automatically obtain an IP after it is powered up.

## 6.2 Wireless Connection

Under Debian you can manage your network with NetworkManager.
After Debian boots click on the network icon on the bottom right of the task bar a NetworkManger menu will pop up and all the available networks will be listed. If there is an active wireless network you will see something similar to the following screenshot:




You can click on a WiFI AP and connect your board to it.
For more details refer to:NetworkManager.

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.

## 6.3 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
```

## 6.4 Install Debian Packages

We provide a Debian Jessie image. You can install Jessie's packages by commanding "apt-get". If this is your first installation you need to update the package list by running the following command

```
apt-get update
```

You can install your preferred packages. For example if you want to install an FTP server you can do this:

```
apt-get install vsftpd
```

Note: you can change your download server by editting "/etc/apt/sources.list". You can get a complete server list from [1]. You need to select the one with "armhf".

## 6.5 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。

## 6.6 Login via VNC and SSH

If your board is not connected to a display device you can login to your board from a mobile phone. You need to download and install a "VNC Viewer" from here on a mobile phone and login to the board via VNC at port 1. Its default password is "fa123456".
Here is a screenshot which shows how it looks like when users login to the board from an iPhone via VNC:


In our case our board's IP address is 192.168.1.230. You can login via SSH by running the following commands:

```
$ ssh root@192.168.1.230
```

The password is fa.

## 6.7 Connect to USB Camera(FA-CAM202)

The FA-CAM202 is a 200M USB camera.
Refer to this link for more details on how to connect to a FA-CAM202: Connect NanoPi M1 to DVP Camera CAM500B
In Debian, click on "other"-->"xawtv" on the left bottom of the GUI and the USB Camera application will be started. After enter "welcome to xawtv！" click on "OK" to start exploring.

## 6.8 Use OpenCV to Access Camera

- The full name of "OpenCV" is Open Source Computer Vision Library and it is a cross platform vision library.
- Make sure your board is connected to the internet and an HDMI monitor, Boot Debian and login.
- Install OpenCV libraries:

```
$ apt-get update
$ apt-get install libcv-dev libopencv-dev
```

- Refer to the instructions in the previous sections to make sure the camera works
- Compile and run a code sample（Official Code Sample in C++ provided by the OpenCV organization):

```
$ cd /home/fa/Documents/opencv-demo
$ make
$ ./demo
```

## 6.9 Connect to DVP Camera CAM500B

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 H3 board to a CAM500B. Then boot OS, connect your board to a network, log into the board as root and run "mjpg-streamer":

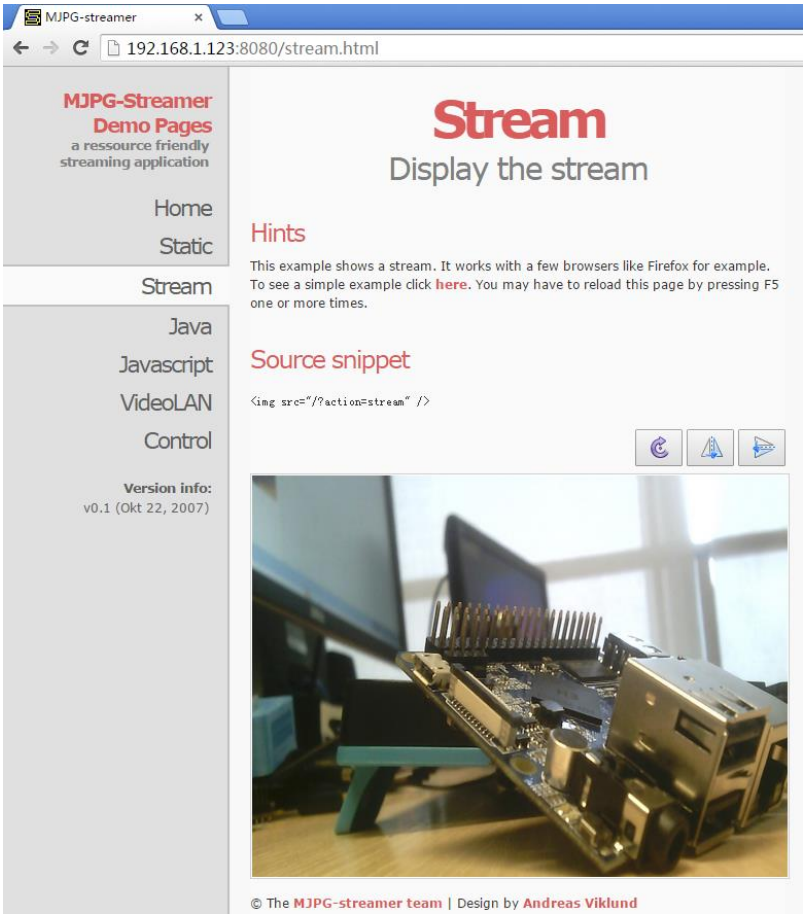
```
$ cd /root/mjpg-streamer
$ make
$ ./start.sh
```

The mjpg-streamer application is an open source video steam server. After it is successfully started the following messages will be popped up:

```
 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
```

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.x based ROM currently doesn't support hardware-encoding. If you use a Linux-3.x 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.

## 6.10 Check CPU's Working Temperature

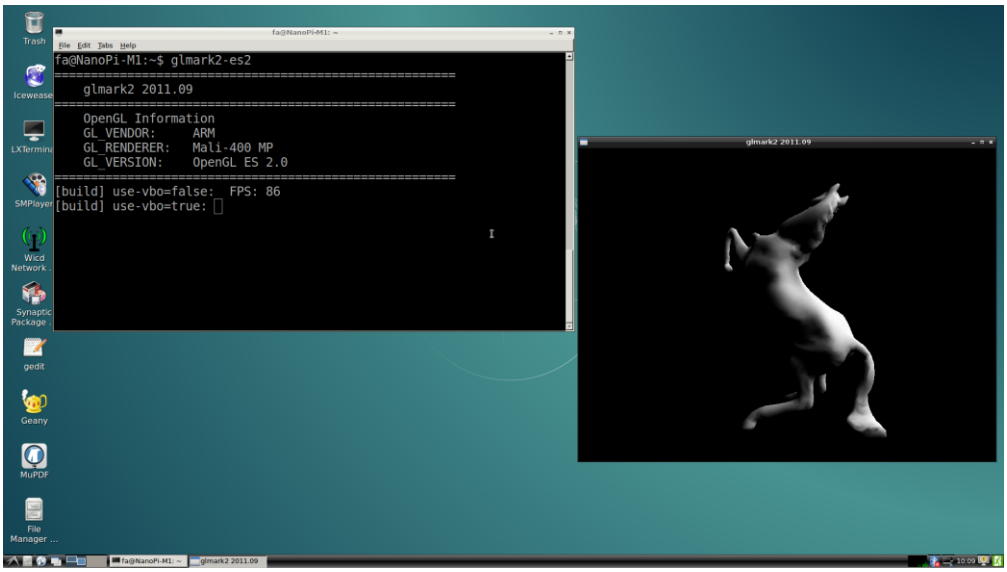
You can use the following command to read H3's temperature and frequency

```
cpu_freq
```

## 6.11 Test GPU

Note: this function is only supported in Allwinner Linux-3.4.y.
After OS loads please login from a terminal and run the following command:

```
glmark2-es2
```




## 6.12 Test VPU

Note: this function is only supported in Allwinner Linux-3.4.y
Visit this link download link to download files
After OS is loaded login from a terminal and run the following commands:

```
$ sudo apt-get install mpv
$ video_play mpv ./big_buck_bunny_1080p_H264_AAC_25fps_7200K.MP4
```

In our test it could do hard coding and play 1080P video fluently.

# 7 Work with FriendlyCore

## 7.1 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.

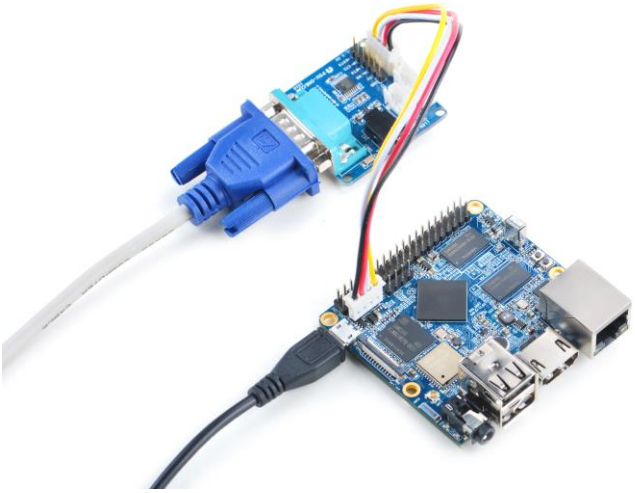
Besides the regular Ubuntu Core's features FriendlyCore has the following additional features:

- 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;

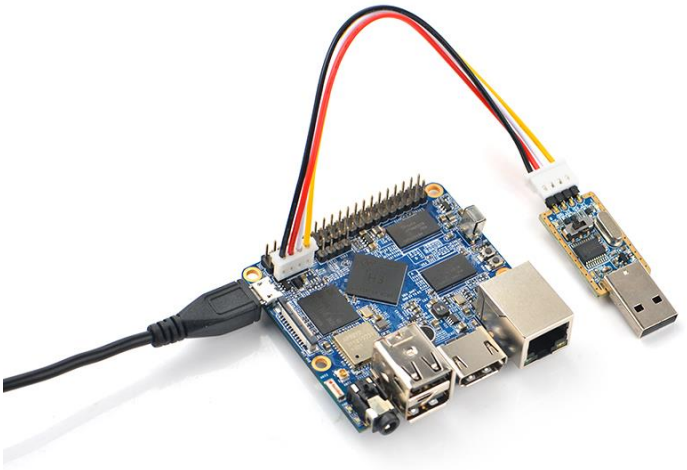
## 7.2 System Login

- 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

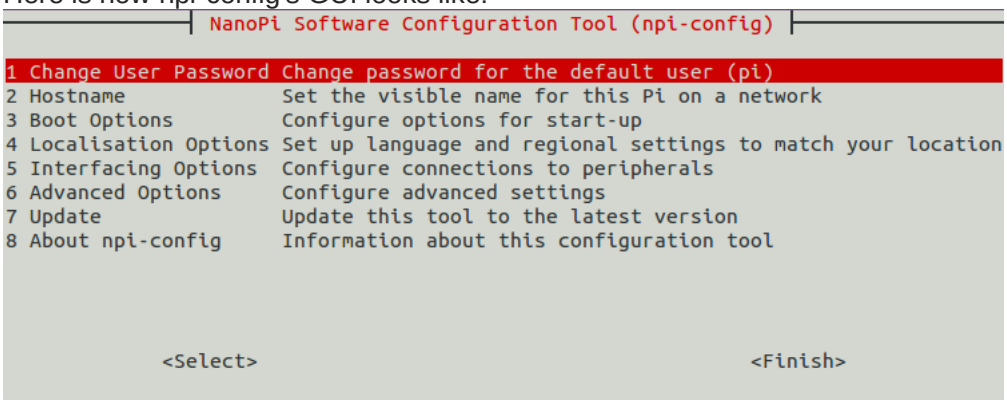
```
$ sudo apt-get update
```

## 7.3 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:




## 7.4 Develop Qt Application

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

## 7.5 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.

## 7.6 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
```

## 7.7 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
```

## 7.8 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.

## 7.9 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
```

## 7.10 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.

## 7.11 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。

## 7.12 WiringPi and Python Wrapper

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

## 7.13 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。

## 7.14 Connect to DVP Camera CAM500B

For NanoPi-M1-Plus 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
```

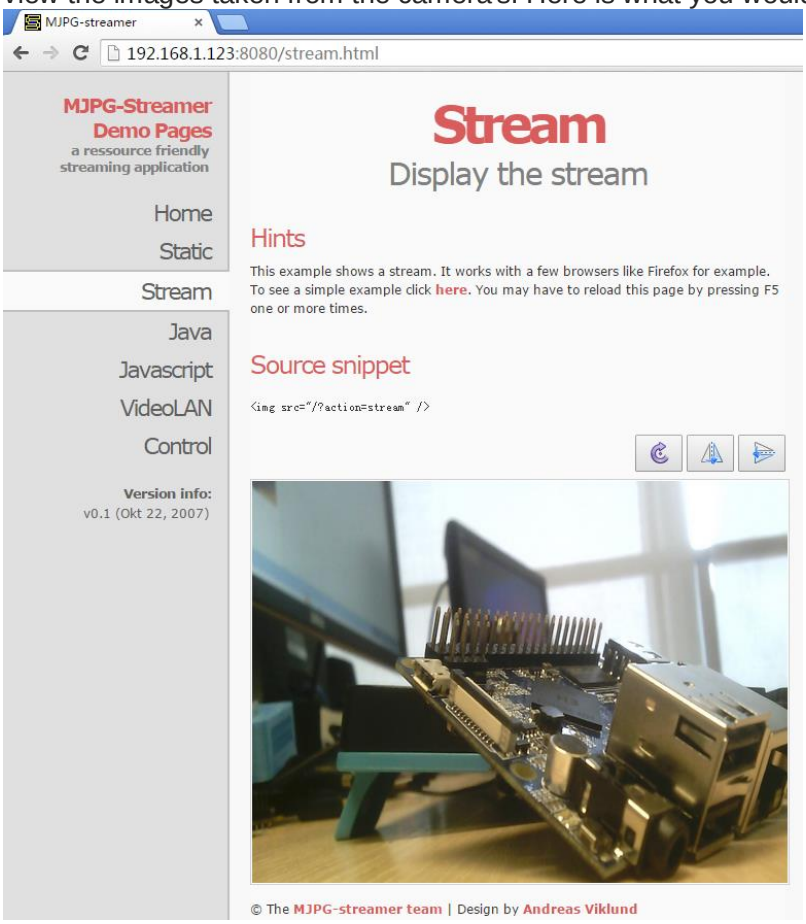
start.sh runs the following two commands:

```
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.

## 7.15 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
```

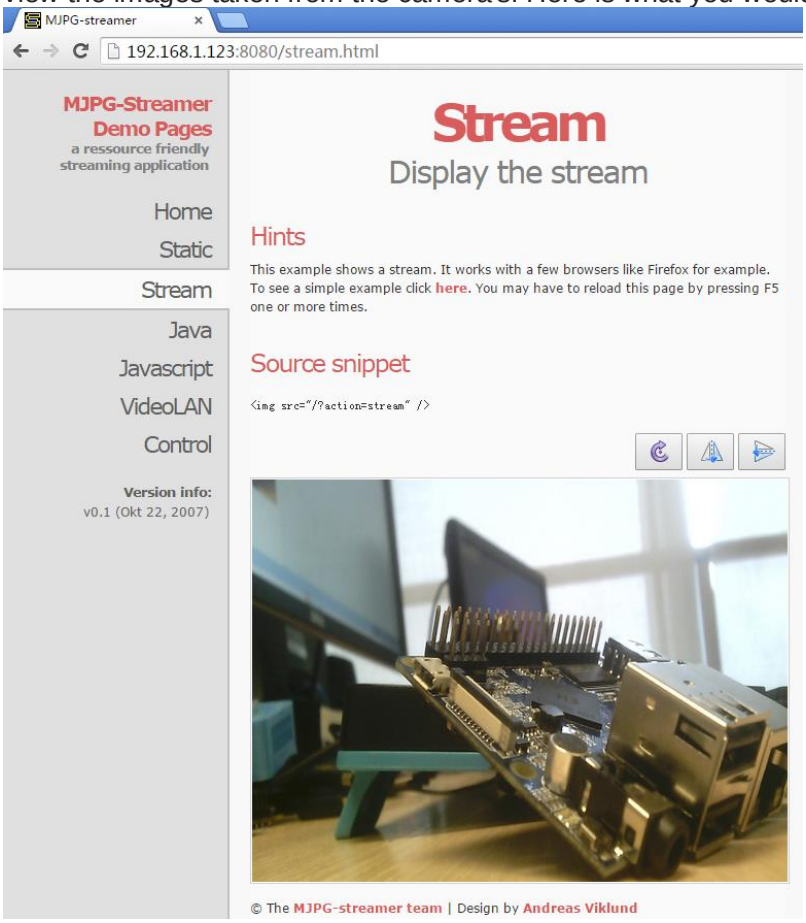
start.sh runs the following two commands:

```
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:




## 7.16 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
```

## 7.17 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:

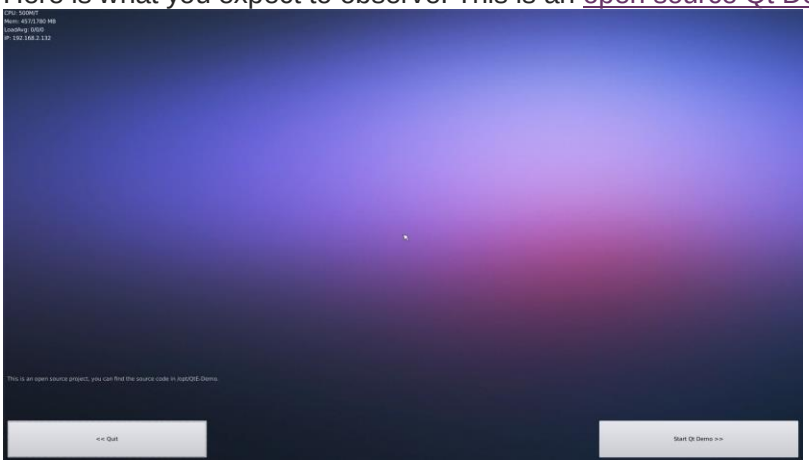
```
1522404275.767215: event type EV_MSC(0x04): scancode = 0xe0e43
1522404275.767215: event type EV_SYN(0x00).
1522404278.911267: event type EV_MSC(0x04): scancode = 0xe0e42
1522404278.911267: event type EV_SYN(0x00).
```

## 7.18 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:



## 7.19 How to install and use docker (for armhf system)

*7.19.1 How to Install Docker*

Run the following commands：

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

*7.19.2 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
```

## 7.20 Using 4G Module EC20 on FriendlyCore

*7.20.1 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/
```

*7.20.2 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
```

*7.20.3 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
```

*7.20.4 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
```

*7.20.5 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
```

## 7.21 Play & Record Audio

You can play and record audio by running the following commands
Check 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 which is recognized as [H3 Audio Codec]. You need to use the actual device name that your [H3 Audio Codec] device is recognized as in your system.

Play Audio:

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

Parameter "-D plughw:0" means the "card 0" device is used to play the audio file. You need to choose a device from the list obtained by running "aplay -l".
Record Audio:

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

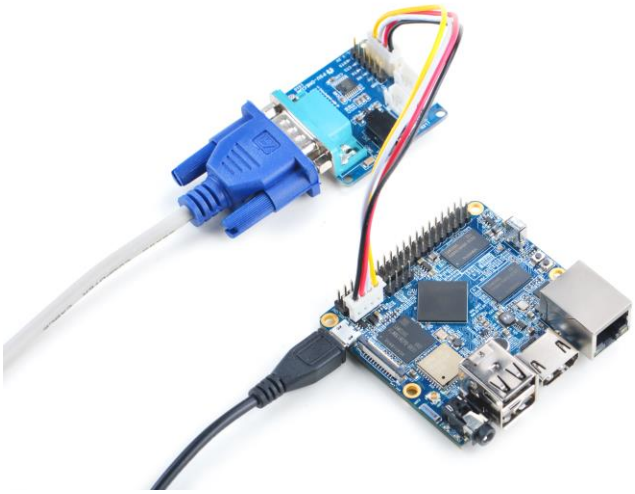
## 8.1 Introduction

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.

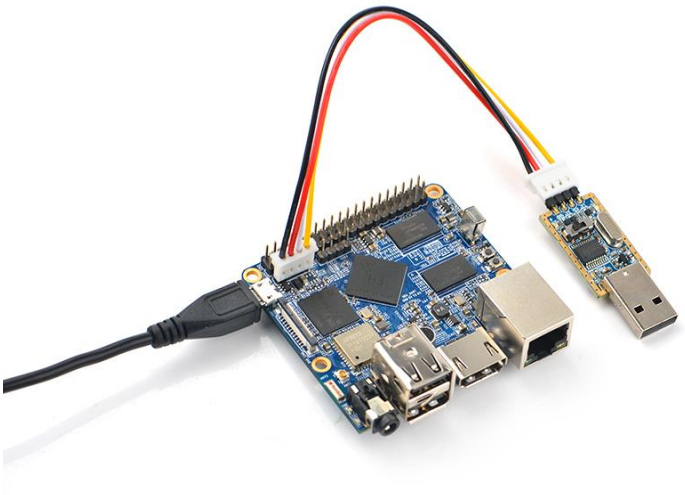
## 8.2 System Login

- **Login via Serial Port**

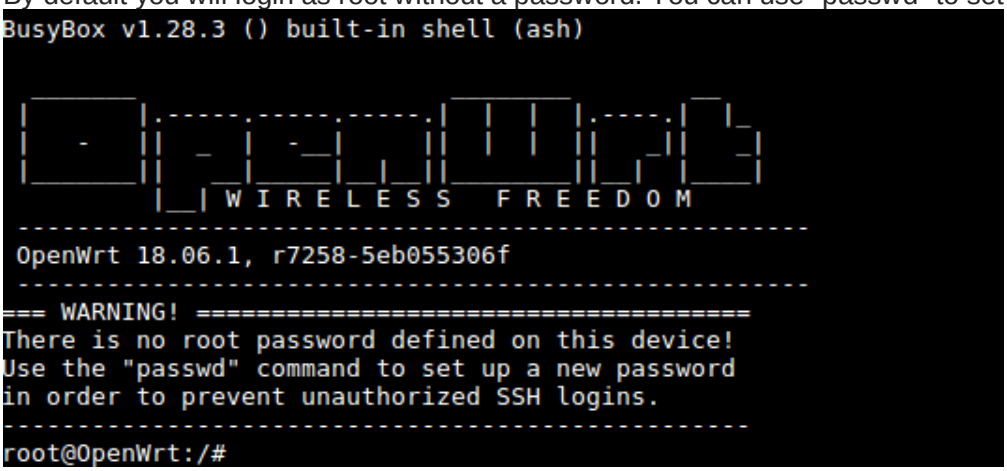
When you do kernel development you'd better get a serial communication board. After you connect your board to a serial communication board you will be able to do development work from a commandline utility.
Here is a hardware setup:
After you connect your board to a serial communication board (e.g. FriendlyElec's serial communication board) you can power the whole system from either the DC port on the serial communication board or the MicroUSB port(if there is one) on your board:



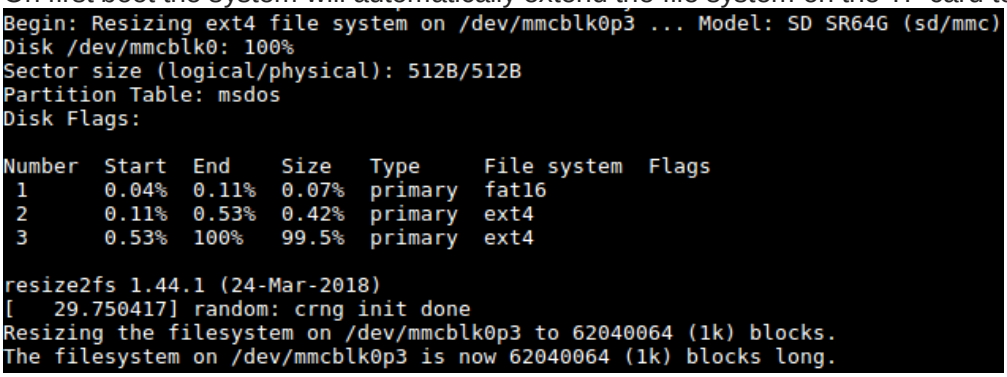
or you can use a USB to serial board and power on the whole system at the MicroUSB port with a 5V/2A power:



By default you will login as root without a password. You can use "passwd" to set a password for root.




On first boot the system will automatically extend the file system on the TF card to the max capacity:




Please wait for this to be done.

- **Login via SSH**

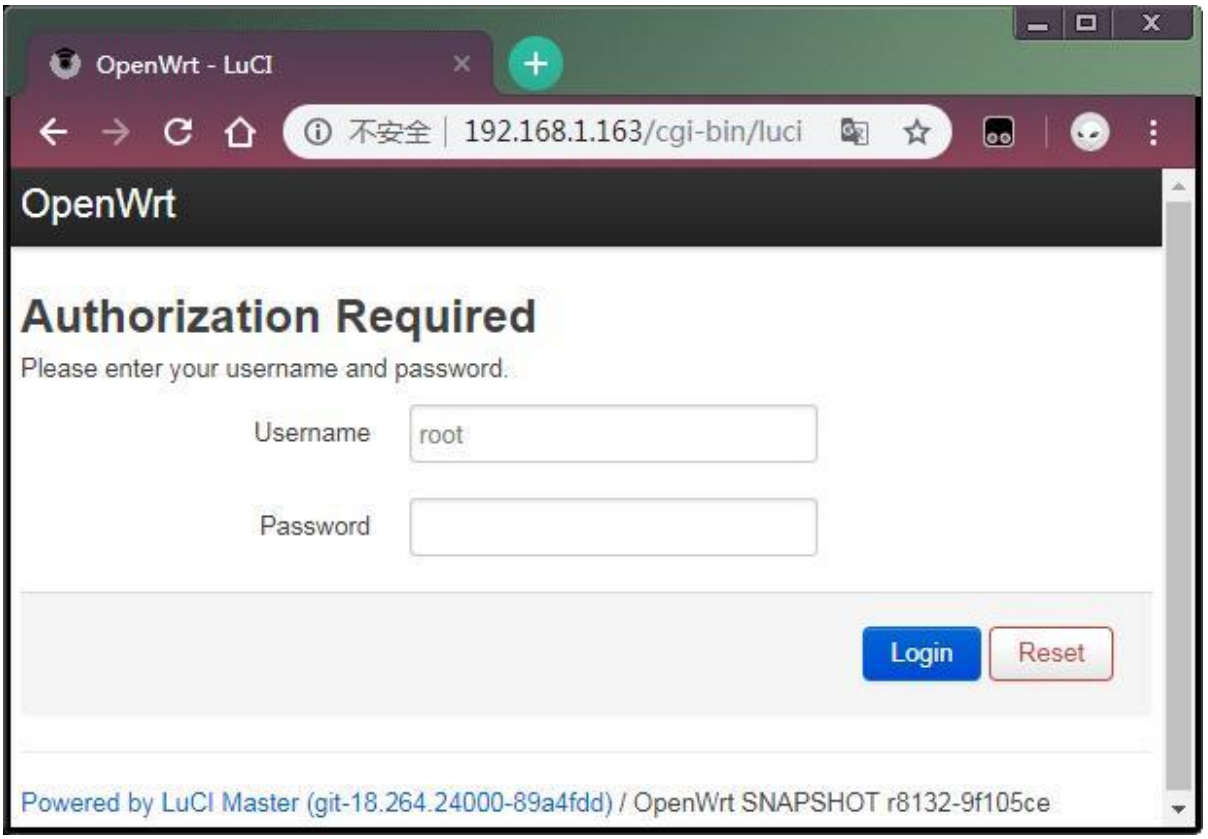
In FriendlyElec's OpenWrt system the Ethernet(eth0) is configured as WAN.
Before power on your board make sure your board is connected to a master router's LAN with an Ethernet cable and the eth0 will be assigned an IP address by DHCP.
For example, if your eth0 is assigned an IP address 192.168.1.163 you can login with SSH by running the following command:

```
$ ssh root@192.168.1.163
```

You can login without a password.

- **Login via Web**

You can login OpenWrt via a LuCI Web page.
After you go through all the steps in <Login via SSH> and get an IP address e.g. 192.168.1.163 for the Ethernet connection, type this IP address in a browser's address bar and you will be able to login OpenWrt-LuCI:




By default you will login as root without a password, just click on "Login" to login.

## 8.3 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

## 8.4 Check System Status

- **Check CPU Temperature & Frequency via Commandline**

```
$ 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
```

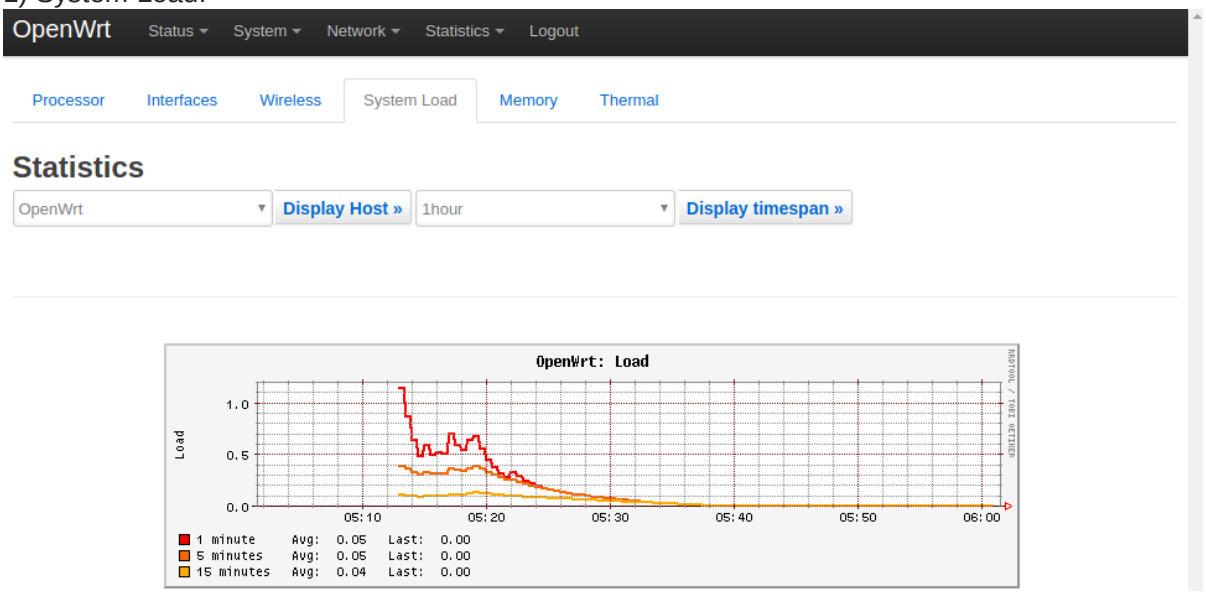
These messages mean that there are four CPU cores working online simultaneously. Each core's temperature is 26.5 degrees in Celsius, the scheduling policy is on-demand and the working frequency is 624MHz. You can set the frequency by running the following command:

```
$ 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
```

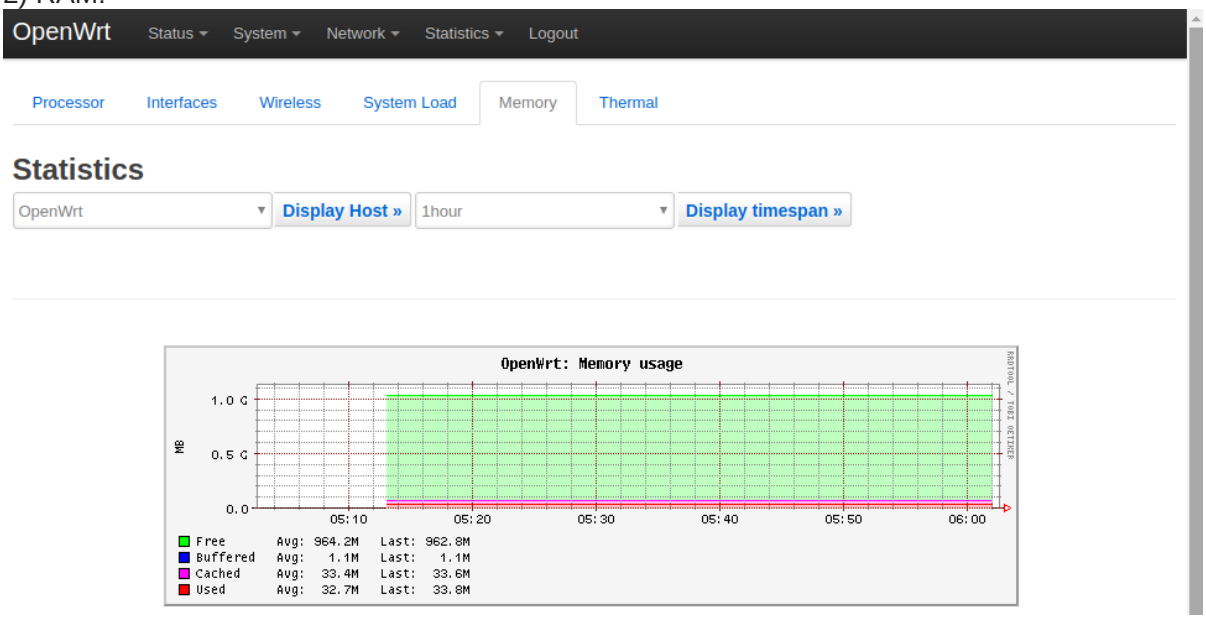
These messages mean four CPU cores are working online. Each core's temperature is 26.5 degrees. Each core's governor is on demand and the frequency is 480 MHz.

- **Check System Status on OpenWrt-LuCI Web Page**

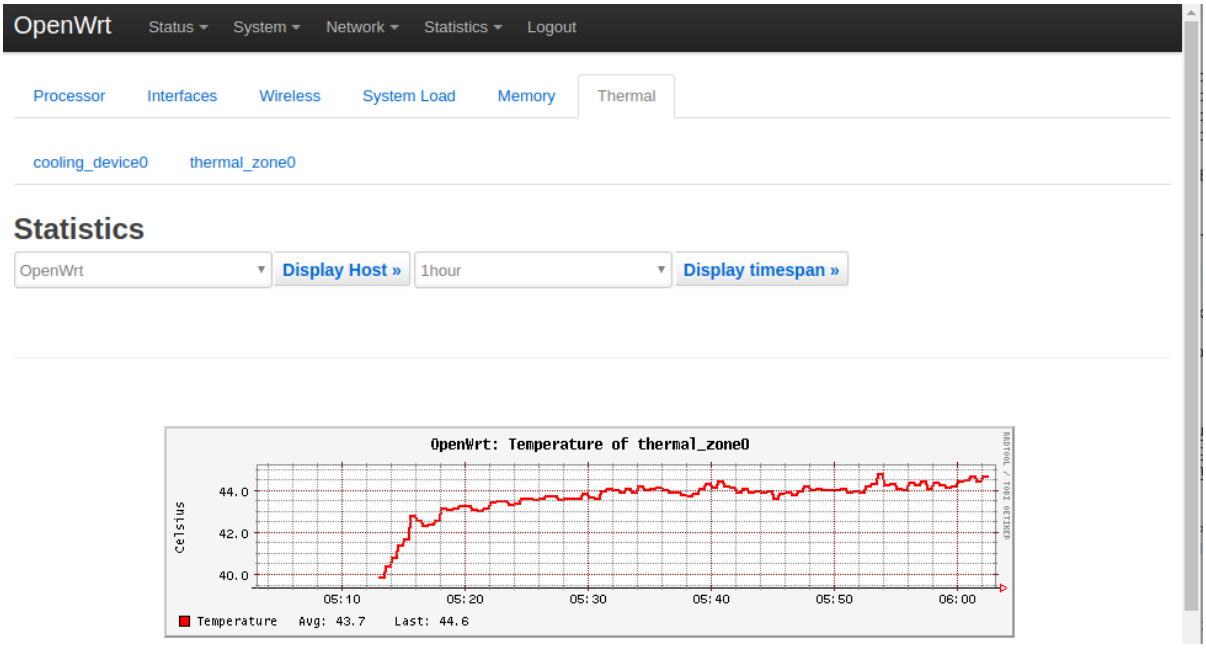
After open the OpenWrt-LuCI page, go to "Statistics ---> Graphs" and you will see various system statistics e.g.:
1) System Load:




2) RAM:




3) CPU Temperature:

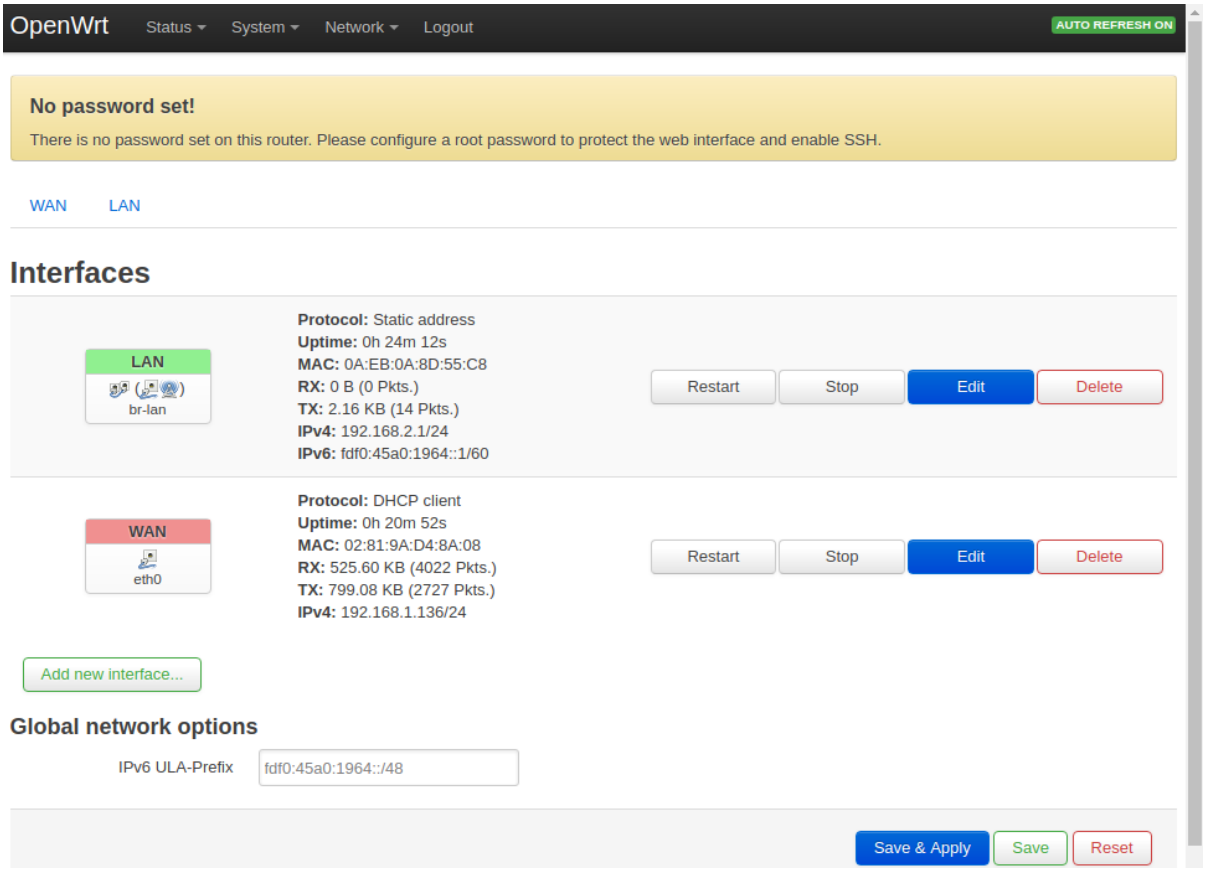



All the statistics listed on the Statistics page are presented by the luci-app-statistics package which uses the Collectd utility to collect data and presents them with the RRDtool utility.
If you want to get more statistics you can install other collectd-mod-* packages. All collectd-mod-* packages use the same configuration file: /etc/config/luci_statistics.

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

## 8.5 Check Network->Interfaces Configurations

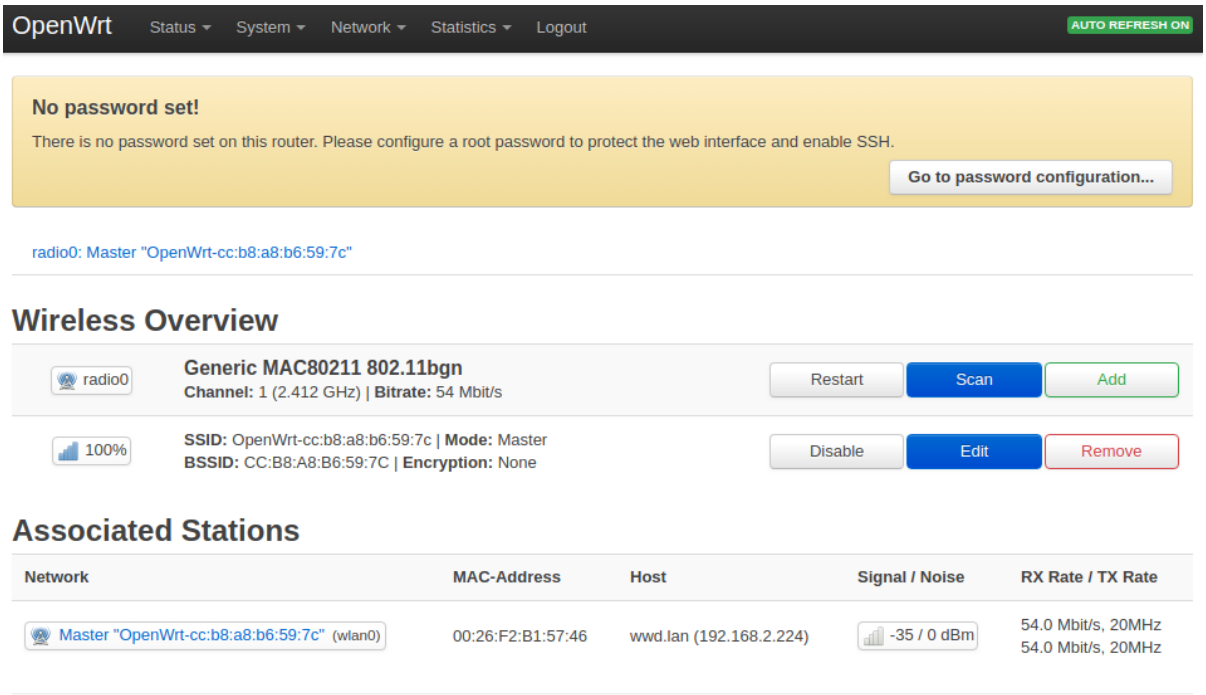
- After open the OpenWrt-LuCI page, go to "Network" ---> "Interfaces" and you will see the current network's configurations:




- All the configurations listed on the Network->Interfaces page are stored in the "/etc/config/network" file.

## 8.6 Check Netwrok->Wireless Configurations

- After open the OpenWrt-LuCI page, go to Network ---> Wireless and you will see the WiFi hotspot's configurations:




A default WiFi AP's hotspot name looks like "OpenWrt-10:d0:7a:de:3d:92". It doesn't have a password. You can connect your smart phone to it and browse the internet.

- All the configurations listed on the Network->Wireless page are stored in the "/etc/config/wireless" file.

## 8.7 USB WiFi

Currently the NanoPi NEO2 Black only works with a RTL8821CU USB WiFi dongle, plug and play. After this module is connected to the board it will by default work under AP mode and the hotspot's name is "rtl8821cu-mac address" and the password is "password";

## 8.8 Huawei's WiFi 2 mini(E8372H-155) Module

After this module is connected to the board it will be plug and play. The hotspot's name is "HUAWEI-8DA5". You can connect a device to the internet by connecting to this hotspot.

# 9 Build Linux System

## 9.1 Based on Linux-4.14 BSP

The NanoPi M1-Plus supports the Linux-4.14 kernel which is mainly maintained and supported by open source communities. FriendlyElec ported this kernel to the NanoPi M1-Plus.
Here is a reference link to more details about how to make image files for Allwinner H3 based on mainline U-boot and Linux-4.14 kernel:Building U-boot and Linux for H5/H3/H2+

## 9.2 Based on Linux-3.4 BSP

The Linux3.4 BSP is provided by Allwinner. FriendlyElec ported this to the NanoPi M1-Plus.

*9.2.1 Preparations*

Get lichee source:

```
$ git clone https://github.com/friendlyarm/h3_lichee.git lichee --depth 1
```

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.

*9.2.2 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.

*9.2.3 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
```

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

```
$ cd lichee/fa_tools
$ ./build.sh -b nanopi-m1-plus -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 you compile the source code it will automatically call this cross-compiler.

*9.2.4 Compile U-boot*

Note:you need to compile the whole lichee directory before you can compile U-boot individually.
You can run the following commands to compile U-boot:

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-m1-plus -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 the MicroSD 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.

*9.2.5 Compile Linux Kernel*

Note:you need to compile the whole lichee directory before you can compile Linux kernel individually.
If you want to compile the Linux kernel run the following command:

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-m1-plus -p linux -t kernel
```

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

*9.2.6 Clean Source Code*

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-m1-plus -p linux -t clean
```

# 10 Applications under Android

## 10.1 IR Controller(RC-100)

You can use FriendlyARM's IR controller(RC-100) to navigate the Android system.
Here is a list of the function keys on the RC-100 IR controller

| Key | Function |
|---|---|
| POWER | On/Off |
| F1 | Search |
| F2 | Open Browser |
| F3 | Enable/Disable Mouse |
| UP | Move Up |
| DOWN | Move Down |

| | |
|---|---|
| LEFT | Move Left |
| RIGHT | Move Right |
| OK | OK |
| Volume- | Turn Down Volume |
| Mute | Mute |
| Volume+ | Turn Up Volume |
| SETTING | Go to Setting Window |
| HOME | Go to Home Window |
| BACK | Go Back to the Previous Window |

After Android is loaded for the first time you need to follow the prompts on Android's GUI to enter the main window and then press F3 to enable mouse and complete the setup process by navigating "up", "down", "left", "right" and "OK".

## 10.2 Play 4K Video

Visit this the test-video directory of this link download link and download the 4K video file: 4K-Chimei-inn-60mbps.mp4 and copy it to an SD card or USB drive.
Boot Android on your M1 Plus and insert this SD card or USB drive to it. After locate the 4K video file with ESFileExplorer click on and play it with Android's Gallery player.
In our test playing this 4K video file from a USB drive worked better.

# 11 Make Your Own Android

## 11.1 Preparations

- 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
```

- Packaging an Android image relies on the scripts in the lichee's source code. Therefore you need to clone lichee's source code:

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

Note:lichee is the name of the project in which Allwinner provides support for its CPUs. The lichee source code includes the source code of U-boot, Linux and various scripts. You cannot rename the "lichee" directory.

- Clone Android Source Code:

```
$ git clone https://gitlab.com/friendlyelec/h3_android-4.4 android
```

Since packaging an Android image relies on the scripts in the lichee's source code. Therefore you need to clone the Android source code under the same directory where lichee is located and name the cloned directory "android":

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

- Install Cross Compiler:

In order to compile the lichee source code you need to 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.

## 11.2 Compile Android

- Setup Environment

Run the following commands on a host PC running 64-bit Ubuntu-14.04 LTS-64bit:

```
$ sudo apt-get install bison g++-multilib git gperf libxml2-utils make python-networkx zip flex libncurses5-dev zlib1g-dev gawk minicom
```

For more details refer to:android_initializing。

- Install JDK

We used the JDK1.6.0_45. You can get it from Oracle: Oracle JDK . In our test we installed it in the /usr/lib/jvm/ directory.

- Compile System

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-m1-plus -p android -t all            # compile lichee's source code and this will generate a kernel and drivers for Android.
$ cd ../../android
$ export PATH=/usr/lib/jvm/jdk1.6.0_45/bin:$PATH
$ ./build.sh -b nanopi-m1-plus                              # compile android's source code and this will generate an Android image file.
```

After the above commands are finished an Android image "sun8iw7p1_android_nanopi-m1-plus_uart0.img" will be generated under the "lichee/tools/pack/" directory.

## 11.3 Clean Source Code

```
$ cd lichee/fa_tools/
$ ./build.sh -b nanopi-m1-plus -p android -t clean
```

# 12 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

## 13 3D Housing Printing Files

- NanoPi M1 Plus 3D housing printing files:[2]

## 14 Resources

- Schematic
    - NanoPi-M1-Plus-1702-Schematic.pdf
- Dimensional Diagram
    - NanoPi-M1-Plus-1702-Dimensional in dxf format
- Allwinner H3 datasheet
    - Allwinner_H3_Datasheet_V1.2.pdf
- 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

# 15 Update Log

## 15.1 March-1-2017

- Released English Version

## 15.2 May-24-2017

- Added section 3: software features

## 15.3 June-4-2017

- Updated section 3: added OS features

## 15.4 June-13-2017

- Added section 7: working with UbuntuCore

## 15.5 July-8-2017

- Updated sections 6.4 & 6.9

## 15.6 Jan-30-2018

- Reorganized wiki
- Updated section 7.6

## 15.7 April-13-2018

- Added sections 6.6, 7.11, 7.16 and 7.17

## 15.8 August-1-2018

- Updated section 10

## 15.9 Dec-19-2018

- Updated section 8

## 15.10 March-21-2019

- Updated section 8