

Getting Started with NanoPi Duo2

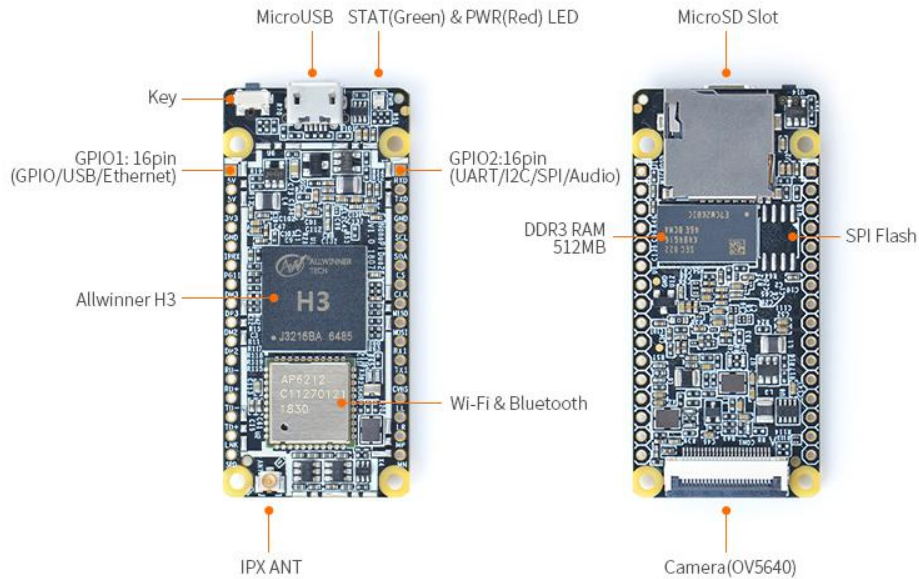
- The NanoPi Duo2("Duo2") is an ARM board designed and developed by FriendlyELEC for makers and hobbyists. It is only 55 x 25.4mm. It features Allwinner quad-core A7 processor H3, and has 256M/512M DDR3 RAM, onboard WiFi & bluetooth module and an OV5640 camera interface. It works with Linux variants such as Ubuntu Core.
- The NanoPi Duo2 is tiny and compact with rich interfaces and ports. It takes power input from its MicroUSB port and can be booted from a Micro SD card. It works with general bread-boards. Interface pins such as USB, SPI, UART, I2C, PWM, IR, audio input & output and Fast Ethernet etc are populated.
- The NanoPi Duo2 supports software utilities such as WiringNP and Python etc. These are all open source. It is suited for various IoT applications.

2 Hardware Spec

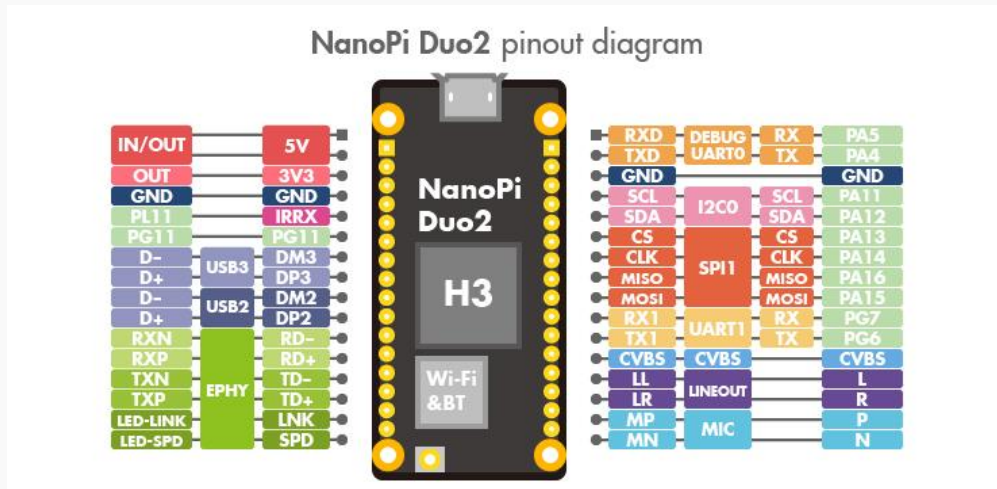
- CPU: Allwinner H3, Quad-core Cortex-A7 Up to 1.2GHz
- DDR3 RAM: 512M
- Connectivity: 10/100M Ethernet
- WiFi: 802.11b/g/n
- Bluetooth: Bluetooth V4.0 of 1, 2 and 3 Mbps.
- Camera: OV5640
- Key: GPIO Key
- USB Host: 2.54mm pin x2, exposed in 2.54mm pitch pin header
- MicroSD Slot x 1
- MicroUSB: OTG and power input
- Debug Serial Interface: exposed in 2.54mm pitch pin header
- Audio input/output Interface: exposed in 2.54mm pitch pin header
- GPIO1: 2.54mm spacing 16pin. It includes UART, SPI, I2C, Audio etc
- GPIO2: 2.54mm spacing 16pin. It includes USB,10/100M Ethernet, IO etc
- PCB Dimension: 25.4 x 55mm
- Power Supply: DC 5V/2A
- Temperature measuring range: -20°C to 70°C
- OS/Software: U-boot, Linux-4.14 / Linux-3.4, Ubuntu 16.04.2 LTS (Xenial)
- Weight: xgx(With Pin-headers)

3 Diagram, Layout and Dimension

3.1 Layout



NanoPi Duo2 Layout



pinout

• GPIO Pin Spec

Pin#	GPIO1	Name	Linux gpio	Pin#	GPIO2	Name	Linux gpio
1	5V	VDD_5V		2	RXD	DEBUG_RX(UART_RXD0)/GPIOA5/PWM0	5
3	5V	VDD_5V		4	TXD	DEBUG_TX(UART_TXD0)/GPIOA4	4
5	3V3	SYS_3.3V		6	GND	GND	
7	GND	GND		8	SCL	I2C0_SCL/GPIOA11	11

9	IRRX	GPIOL11/IR-RX	363		10	SDA	I2C0_SDA/GPIOA12	12
11	PG11	GPIOG11	203		12	CS	UART3_TX/SPI1_CS/GPIOA13	13
13	DM3	USB-DM3			14	CLK	UART3_RX/SPI1_CLK/GPIOA14	14
15	DP3	USB-DP3			16	MISO	UART3_CTS/SPI1_MISO/GPIOA16	16
17	DM2	USB-DM2			18	MOSI	UART3_RTS/SPI1_MOSI/GPIOA15	15
19	DP2	USB-DP2			20	RX1	UART1_RX/GPIOG7	199
21	RD-	EPHY-RXN			22	TX1	UART1_TX/GPIOG6	198
23	RD+	EPHY-RXP			24	CVBS	CVBS	
25	TD-	EPHY-TXN			26	LL	LINEOUT_L	
27	TD+	EPHY-TXP			28	LR	LINEOUT_R	
29	LNK	EPHY-LED-LINK			30	MP	MIC_P	
31	SPD	EPHY-LED-SPD			32	MN	MIC_N	

- **Camera (OV5640) Pin Spec**

Pin#	Name
1	NC1
2	AGND

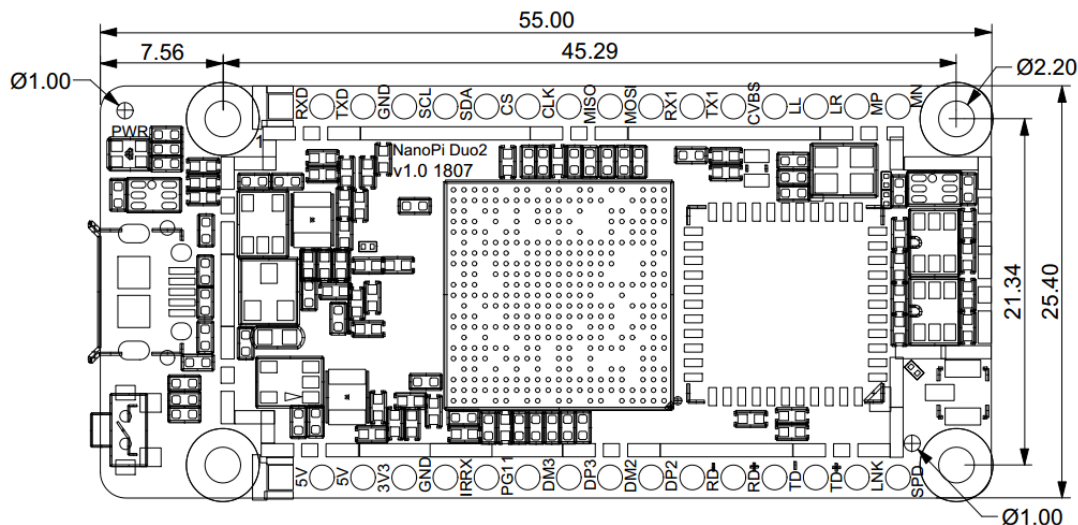
3	SIO-D
4	AVDD
5	SIO-C
6	RESER
7	VSYNC
8	PWDN
9	HREF
10	DVDD
11	DOVDD
12	Y9/MDP1
13	XCLK
14	Y8/MDN1
15	DGND
16	Y7/MCP
17	PCLK
18	Y6/MCN
19	Y2

20	Y5/MDP0
21	Y3
22	Y4/MDN0
23	AF-VDD
24	NC2

Note

1. SYS_3.3V: 3.3V power output
2. VDD_5V: 5V power input/output. 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 :[NanoPi Duo2 Schematic](#)

3.2 Dimensional Diagram



For more details refer to :[NanoPi Duo2 V1.0 1807 pcb in dxf format](#)

4 Get Started

4.1 Essentials You Need

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

- NanoPi Duo2
- 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
- A serial communication board

4.2 TF Cards We Tested

To make your NanoPi Duo2 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 闪迪



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



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



4.3 Install OS

4.3.1 Get Image Files

Visit this link [download link](#) to download image files (under the official-ROMs directory) and the flashing utility(under the tools directory):

Image Files:	
nanopi-duo2_sd_friendlycore-xenial_4.14_armhf_YYYYMMDD.img.zip	FriendlyCore (base on UbuntuCore) Image File, Kernel: Linux-4.14
nanopi-duo2_sd_friendlywrt_4.14_armhf_YYYYMMDD.img.zip	Base on OpenWrt, kernel:Linux-4.14
Flash Utility:	
win32diskimager.rar	Windows utility for flashing Debian image. Under Linux users can use "dd"

4.4 Work with NanoPi Duo2 IoT-Box Carrier Board

FriendlyELEC developed a dedicated carrier board: NanoPi Duo2 IoT-Box. For more details about this carrier board refer to: [Introduction to NanoPi Duo2 IoT-Box](#). How is a hardware setup:



5 Work with FriendlyCore

5.1 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:

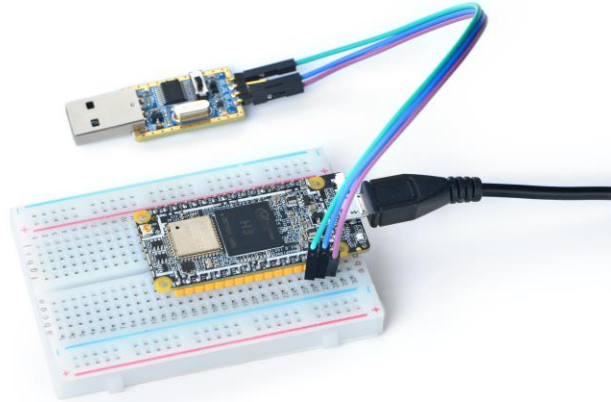
- it integrates Qt4.8;
- it integrates NetworkManager;
- it has bluez and Bluetooth related packages;
- it has alsa packages;
- it has npi-config;
- it has RPiGPIO, a Python GPIO module;
- it has some Python/C demo in /root/ directory;
- it enables 512M-swap partition;

5.2 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

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 npu-config" to disable auto login.

- Update packages

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

5.3 Configure System with npu-config

The npu-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 npu-config
```


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

```
NanoPi Software Configuration Tool (npf-config)

1 Change User Password Change password for the default user (pi)
2 Hostname Set the visible name for this Pi on a network
3 Boot Options Configure options for start-up
4 Localisation Options Set up language and regional settings to match your location
5 Interfacing Options Configure connections to peripherals
6 Advanced Options Configure advanced settings
7 Update Update this tool to the latest version
8 About npf-config Information about this configuration tool

<Select> <Finish>
```

5.4 Develop Qt Application

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

5.5 Setup Program to AutoRun

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

```
sudo npf-config
```

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

5.6 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
```

5.7 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
```

5.8 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.

5.9 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 nearby 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
```

5.10 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.

5.11 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](#).

5.12 WiringPi and Python Wrapper

- [WiringNP: NanoPi NEO/NEO2/Air GPIO Programming with C](#)
- [RPi.GPIO : NanoPi NEO/NEO2/Air GPIO Programming with Python](#)

5.13 Custom welcome message

The welcome message is printed from the script in this directory :

```
/etc/update-motd.d/
```

For example, to change the FriendlyELEC LOGO, you can change the file /etc/update-motd.d/10-header. For example, to change the LOGO to HELLO, you can change the following line :

```
TERM=linux toilet -f standard -F metal $BOARD_VENDOR
```

To:

```
TERM=linux toilet -f standard -F metal HELLO
```

5.14 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
```

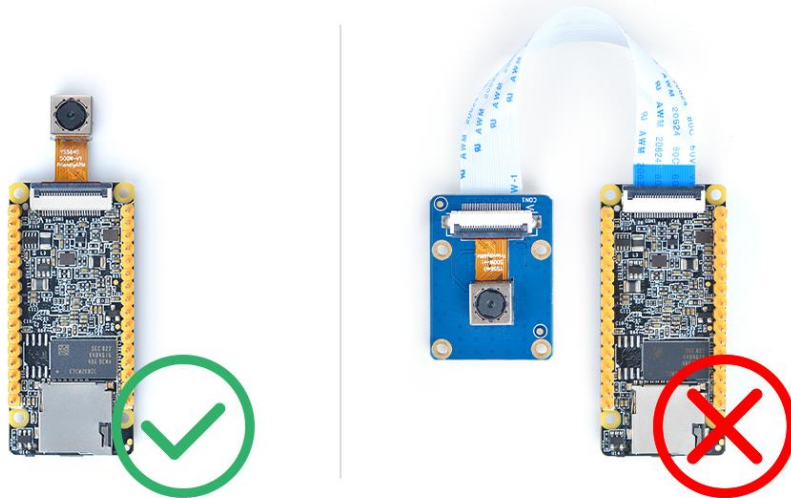
You will hear sounds from system's default audio device.

If you are using H3/H5/H2+ series board with mainline kernel, the easier way is using [npi-config](#).

5.15 Connect to DVP Camera OV5640

For NanoPi-Duo2 the OV5640 can work with Linux-4.14 Kernel.

The NanoPi-Duo2 has support for OV5640 cameras and you can directly connect an OV5640 camera to the board. Here is a hardware setup:



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 stream 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
```

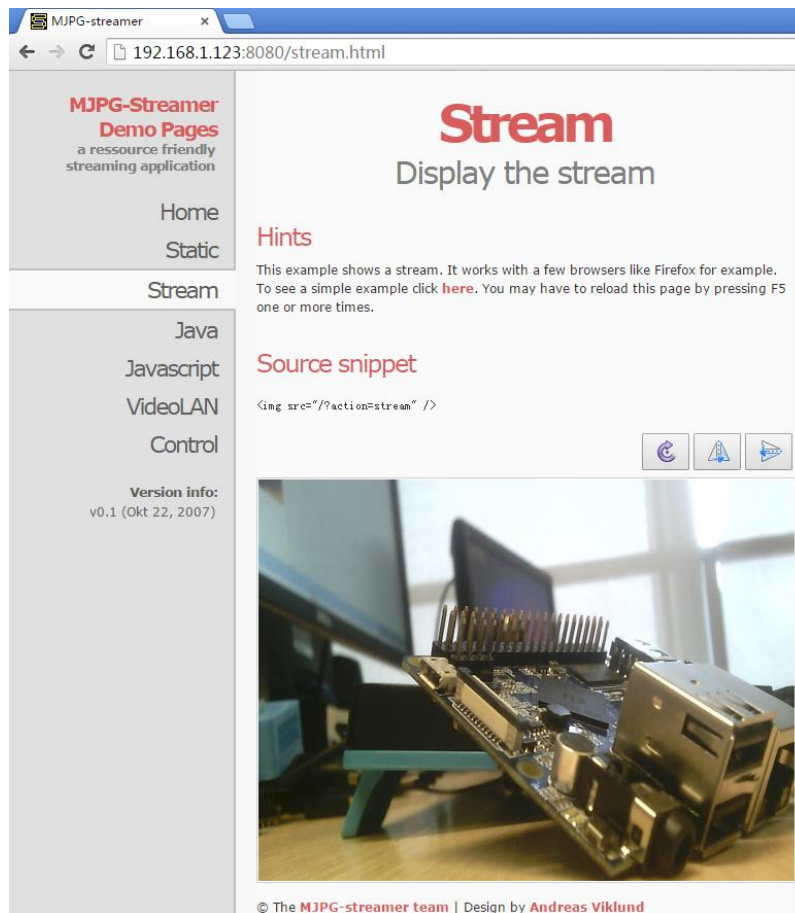
start.sh runs the following two commands:

```
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 dyncntrl's 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 stream 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.

5.16 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-lclb000.usb-1
...
```

The above messages indicate that `/dev/video0` is camera's device node. The `mjpg-streamer` application is an open source video stream 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:

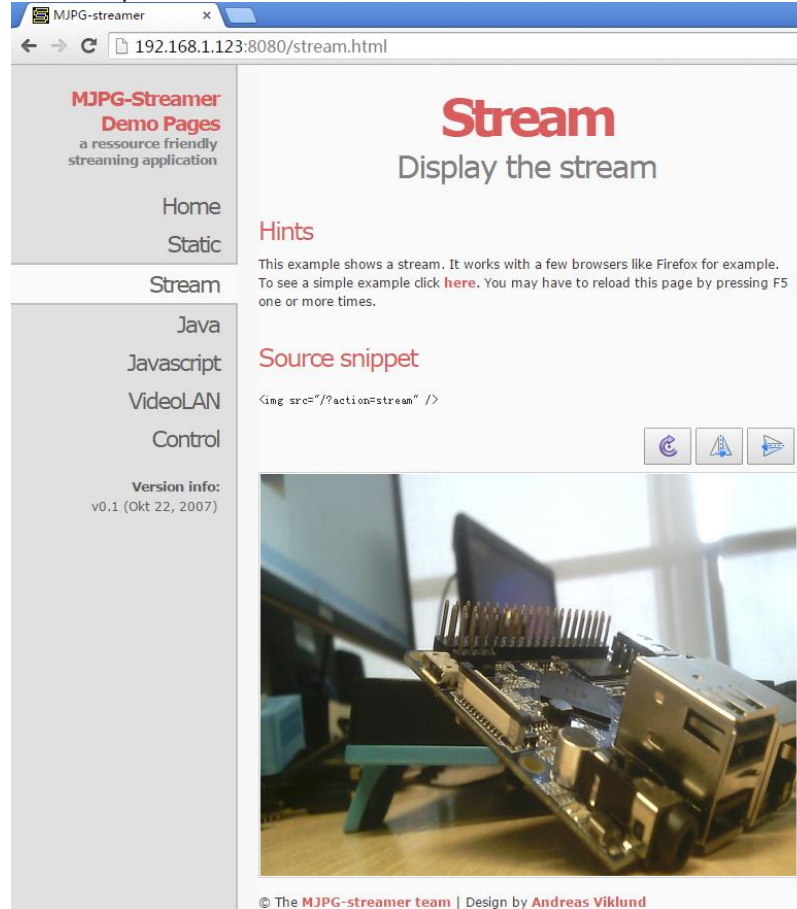
```
export LD_LIBRARY_PATH="$ (pwd) "
./mjpg_streamer -i "./input_uvc.so -d /dev/video0 -y 1 -r 1280x720
-f 30 -q 90 -n -fb 0" -o "./output_http.so -w ./www"
```

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:



5.17 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
```

5.18 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 np-i-config utility:

```
$ np-i-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
+xdp" > /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).
```

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

5.19.1 How to Install Docker

Run the following commands :

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

5.19.2 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
```

5.20 Using 4G Module EC20 on FriendlyCore

5.20.1 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/
```

5.20.2 Step2 : Add udhcpd script

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

```
#!/bin/sh

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

[ -z "$1" ] && echo "Error: should be called from udhcpd" && 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/udhcpd/default.script
```

5.20.3 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
```

5.20.4 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
```

5.20.5 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 Work with OpenWrt

6.1 Introduction

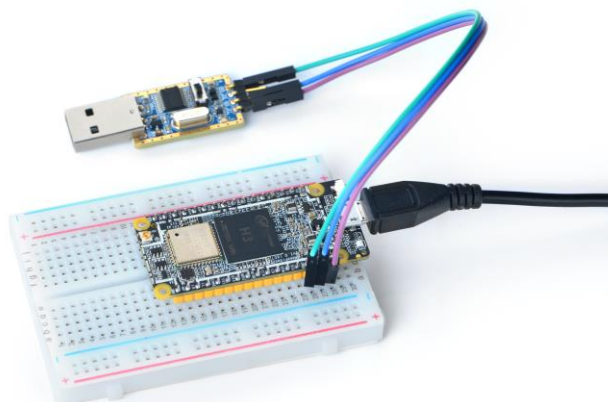
OpenWrt is a highly extensible GNU/Linux distribution for embedded devices. Unlike many other distributions for routers, OpenWrt is built from the ground up to be a full-featured, easily modifiable operating system for embedded devices. In practice, this means that you can have all the features you need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

6.2 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.

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.

```
BusyBox v1.28.3 () built-in shell (ash)

- _ - _ 
|_| W I R E L E S S   F R E E D O M 

-----
OpenWrt 18.06.1, r7258-5eb055306f
===== 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:/#
```

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

```
Begin: Resizing ext4 file system on /dev/mmcblk0p3 ... Model: SD SR64G (sd/mmc)
Disk /dev/mmcblk0: 100%
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type    File system  Flags
  1      0.04%  0.11%  0.07%   primary fat16
  2      0.11%  0.53%  0.42%   primary ext4
  3      0.53% 100%   99.5%   primary ext4

resize2fs 1.44.1 (24-Mar-2018)
[ 29.750417] random: crng init done
Resizing the filesystem on /dev/mmcblk0p3 to 62040064 (1k) blocks.
The filesystem on /dev/mmcblk0p3 is now 62040064 (1k) blocks long.
```

Please wait for this to be done.

- **Login via SSH**

By default in FriendlyElec's OpenWrt system the WiFi AP hotspot's name is like "OpenWrt-10:d0:7a:de:3d:92" and the network segment is 192.168.2.x. You can connect your device to it and login with SSH without a password by running the following command:

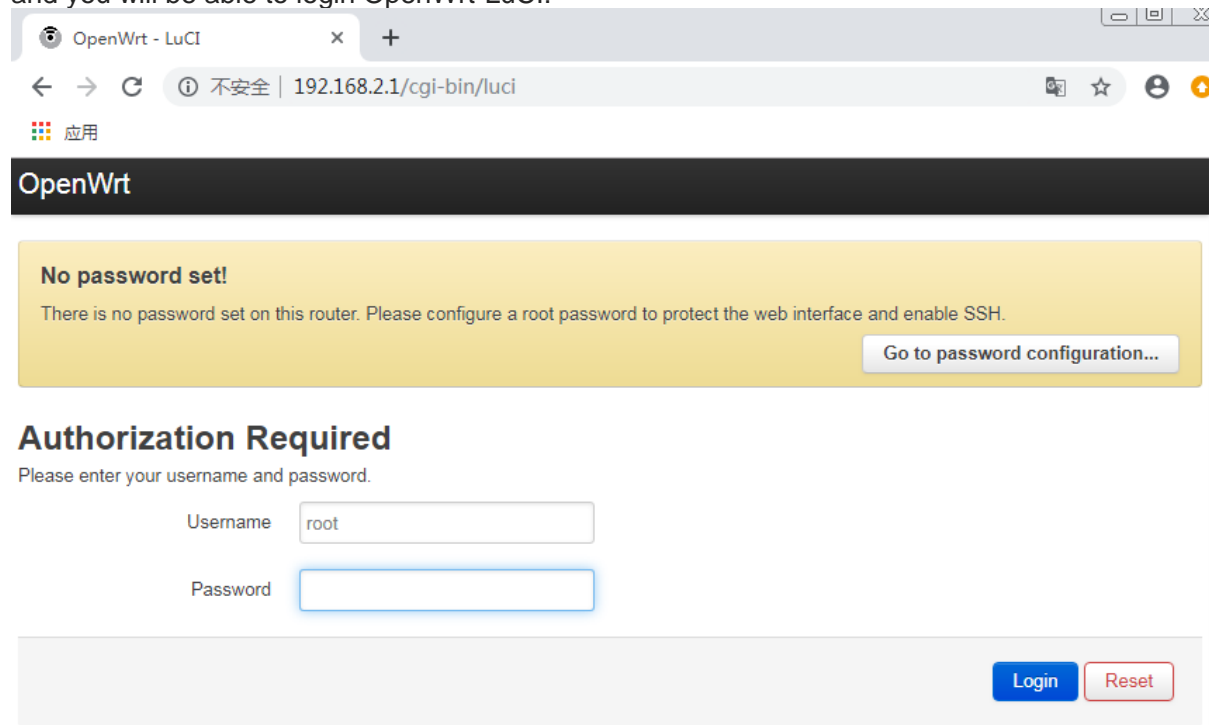
```
$ ssh root@192.168.2.1
```

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. 192.168.2.1 for the Ethernet connection, type this IP address in a browser's address bar and you will be able to login OpenWrt-LuCI:



OpenWrt

No password set!

There is no password set on this router. Please configure a root password to protect the web interface and enable SSH.

[Go to password configuration...](#)

Authorization Required

Please enter your username and password.

Username

Password

[Login](#) [Reset](#)

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

6.3 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>
```

<code>find <regex></code>	List packages whose name or description matches <regex>
<code>info [pkg regex]</code>	Display all info for <pkg>
<code>status [pkg regex]</code>	Display all status for <pkg>
<code>download <pkg></code>	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](#)

6.4 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:

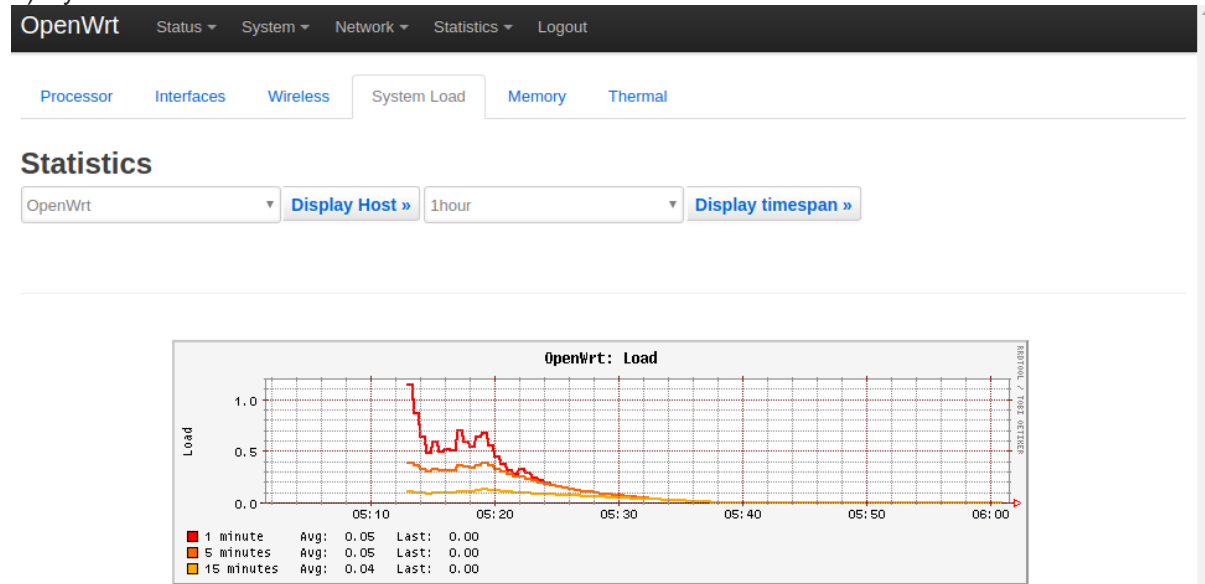
```
$ cpu_freq -s 1008000
Aavailable frequency(KHz):
    480000 624000 816000 1008000
Current frequency(KHz):
    CPU0 online=1 temp=36702C governor=userspace
freq=1008000KHz
    CPU1 online=1 temp=36702C governor=userspace
freq=1008000KHz
    CPU2 online=1 temp=36702C governor=userspace
freq=1008000KHz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

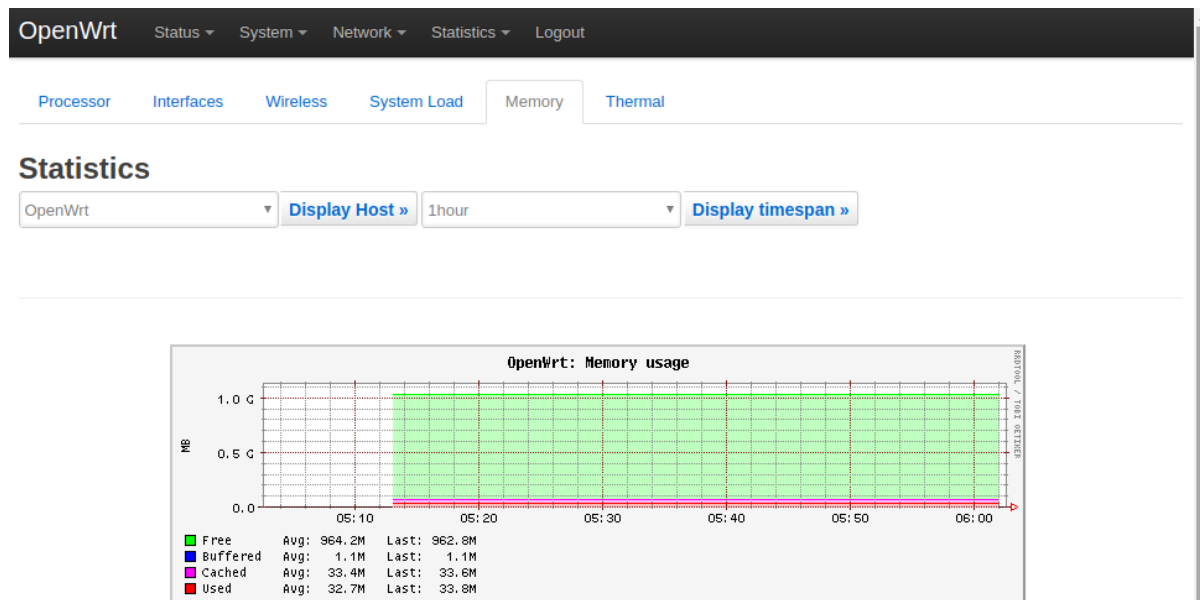
- **Check System Status on OpenWrt-LuCI Web Page**

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

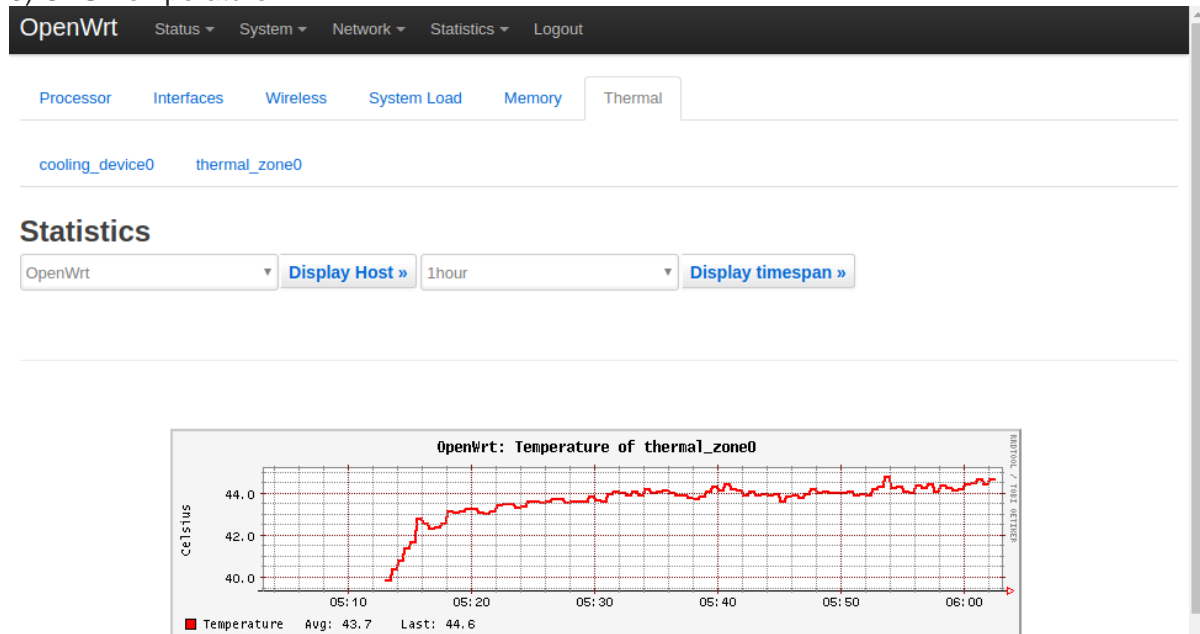
1) System Load:



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](#)

6.5 Check Network->Interfaces Configurations

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

OpenWrt
Status
System
Network
Logout
AUTO REFRESH ON

No password set!

There is no password set on this router. Please configure a root password to protect the web interface and enable SSH.

WAN
LAN

Interfaces

LAN

br-lan

Protocol: Static address
Uptime: 0h 24m 12s
MAC: 0A:EB:0A:8D:55:C8
RX: 0 B (0 Pkts.)
TX: 2.16 KB (14 Pkts.)
IPv4: 192.168.2.1/24
IPv6: fd0:45a0:1964::1/60

RestartStopEditDelete

WAN

eth0

Protocol: DHCP client
Uptime: 0h 20m 52s
MAC: 02:81:9A:D4:8A:08
RX: 525.60 KB (4022 Pkts.)
TX: 799.08 KB (2727 Pkts.)
IPv4: 192.168.1.136/24

RestartStopEditDelete

Add new interface...

IPv6 ULA-Prefix

fd0:45a0:1964::/48

Save & ApplySaveReset

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

6.6 Check Network->Wireless Configurations

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

OpenWrt
Status
System
Network
Statistics
Logout
AUTO REFRESH ON

No password set!

There is no password set on this router. Please configure a root password to protect the web interface and enable SSH.

Go to password configuration...

radio0: Master "OpenWrt-cc:b8:a8:b6:59:7c"

Wireless Overview

radio0

Generic MAC80211 802.11bgn
Channel: 1 (2.412 GHz) | Bitrate: 54 Mbit/s

RestartScanAdd

100%

SSID: OpenWrt-cc:b8:a8:b6:59:7c | Mode: Master
BSSID: CC:B8:A8:B6:59:7C | Encryption: None

DisableEditRemove

Associated Stations

Network	MAC-Address	Host	Signal / Noise	RX Rate / TX Rate
Master "OpenWrt-cc:b8:a8:b6:59:7c" (wlan0)	00:26:F2:B1:57:46	wwd.lan (192.168.2.224)	-35 / 0 dBm	54.0 Mbit/s, 20MHz 54.0 Mbit/s, 20MHz

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.

6.7 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";

6.8 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.

7 Make Your Own Linux System

7.1 Make Image Based on Linux-4.14 BSP

The NanoPi Duo2 supports the Linux-4.14 kernel which is mainly maintained and supported by open source communities. FriendlyElec ported this kernel to the NanoPi Duo2.

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+](#)

7.2 Make Image Based on Linux-3.4 BSP

The Linux3.4 BSP is provided by Allwinner. FriendlyElec ported this to the NanoPi Duo2.

7.2.1 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.

7.2.2 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.

7.2.3 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.

7.2.4 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.

7.2.5 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".

7.2.6 Clean Source Code

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

8 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](#)

9 Resources

9.1 Datasheets & Schematics

- Schematic: [NanoPi Duo2 V1.0 1807 Schematic](#)
- Dimensional Diagram: [NanoPi Duo2 V1.0 1807 PCB Dimensional Diagram](#)
- H3's datasheet [Allwinner H3 Datasheet V1.2.pdf](#)

10 Hardware Update Versions

10.1 V1.0 1807

First Version

11 Update Log

11.1 Oct-10-2018

- Released English Version

11.2 Dec-19-2018

- Updated Section 6