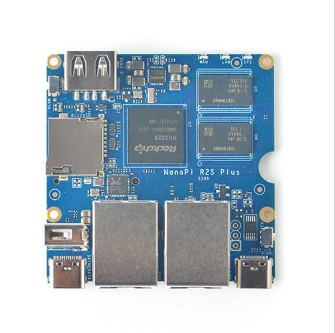
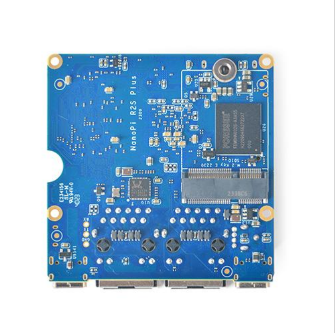
# NanoPi R2S Plus

## Contents

Overview


Front


Back

w/ Wi-Fi

Case

# 1 Introduction

The NanoPi R2S PLUS is a FriendlyElec mini-router with edge-computing and dual Gbps Ethernet ports.

The NanoPi R2S PLUS uses Rockchip's quad-core A53 RK3328 SoC with powerful performance. Its default frequency is 1.2GHz. The NanoPi R2S PLUS has 1GB RAM, 32G eMMC flash, dual Gbps Ethernet ports, a GPIO connector etc. It uses RK805 PMU chip and supports dynamic frequency scaling. It has two USB 2.0 port that can interface with 4G modules, USB HD cameras, USB WiFi modules etc.

The NanoPi R2S Plus can be optionally equipped with an RTL8822CS WiFi module (SDIO Interface), which can work in AP mode or wireless repeater mode for wireless range expansion.

The NanoPi R2S Plus suppors TF card booting. It works with FriendlyWrt, OpenMediaVault, Debian Core and Ubuntu core etc. These systems are based on Linux-6.1(LTS). The NanoPi R2S Plus is powered through USB Type-C. It is a good choice for applications that need to be deployed in compact space and need strong networking performance.

# 2 Hardware Spec

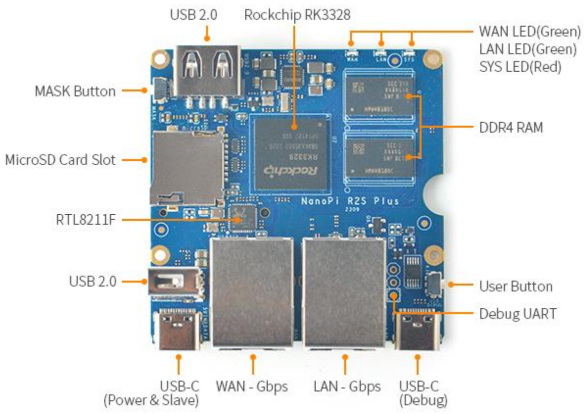
- CPU: Rockchip RK3328, Quad-core Cortex-A53
- DDR4 RAM: 1GB
- Flash: 32GB eMMC 5.1 Flash
- Network：
  - 1 x Internal 10/100/1000M Ethernet
  - 1 x USB 3.0 converted 10/100/1000M Ethernet
  - Optional M.2 SDIO Wi-Fi
- 2 x USB 2.0 Host Type-A
- 1 x microSD Slot
- USB-C Power: 5V power input and USB update in maskrom mode
- USB-C Bebug: onboard USB to Debug UART converter, 1500000bps bauds
- 3 x LED, for WAN, LAN and SYSTEM
- 1 x GPIO Button, 1 x Maskrom mode button
- PCB Size: 58*58*1.6mm
- Power Supply: DC 5V/2A
- Temperature measuring range: 0℃ to 70℃
- OS/Software: U-boot， Ubuntu-Core， OpenWrt

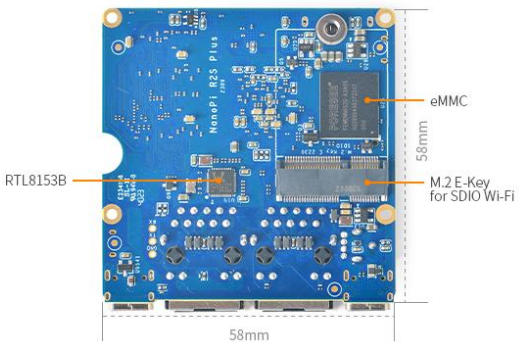
- **Network Transmission Rates**

| | TX | RX |
|---|---|---|
| WAN | 941 Mbps | 941 Mbps |
| LAN | 941 Mbps | 941 Mbps |
| Notes： | 1. test utility: iperf<br>2. use indepedent IP address section and test with a PC in simplex communication mode | |

# 3 Diagram, Layout and Dimension

## 3.1 Layout





For more details refer to:NanoPi_R2S_Plus_2309_SCH.PDF (https://wiki.friendlyelec.com/wiki/images/d/da/NanoPi_R2S_Plus_2309_SCH.PDF)

Dimensional Diagram： Nanopi_r2s_plus_2309_dxf.zip (https://wiki.friendlyelec.com/wiki/images/0/0c/Nanopi_r2s_plus_2309_dxf.zip)

# 4 Get Started

## 4.1 Essentials You Need

Before starting to use your NanoPi R2S Plus get the following items ready

- NanoPi R2S Plus
- MicroSD Card/TF Card: Class 10 or Above, minimum 8GB SDHC
- 5V/2A and above USB Type-C interface power adapter
- If you need to develop and compile,you need a computer that can connect to the Internet. It is recommended to install Ubuntu 20.04 64-bit system and use the following script to initialize the development environment, or use docker container:
    - How to setup the Compiling Environment on Ubuntu bionic (https://github.com/friendlyarm/build-env-on-ubuntu-bionic)
    - docker-cross-compiler-novnc (https://github.com/friendlyarm/docker-cross-compiler-novnc)

## 4.2 TF Cards We Tested

Refer to： TFCardsWeTested

## 4.3 Configure parameters for serial port

Use the following serial parameters:

| | |
|---|---|
| Baud rate | 1500000 |
| Data bit | 8 |
| Parity check | None |
| Stop bit | 1 |
| Flow control | None |

## 4.4 Install OS

### 4.4.1 Downloads

#### 4.4.1.1 Official image

Visit download link (http://download.friendlyelec.com/NanoPiR2SPlus) to download official image files (in the "01_Official images" directory).
The table below lists all official images, the word 'XYZ' in image filename meaning:

- **sd**: Use it when you need to boot the entire OS from the SD card
- **eflasher**: Use it when you need to flash the OS to eMMC via TF card
- **usb**: Use it when you need to flash the OS to eMMC via USB

| Icon | Image Filename | Version | Description | Kernel Version |
|---|---|---|---|---|
| | rk3328-XYZ-debian-bookworm-core-6.1-arm64-YYYYMMDD.img.gz | bookworm | **Debian 12 Core**, No desktop environment, command line only | 6.1.y |
| | rk3328-XYZ-friendlycore-focal-4.19-arm64-YYYYMMDD.img.gz | focal | 64-bit FriendlyCore image file(Qt 5.10.0) based on Ubuntu core 20.04 64bit | 4.19.y |
| | rk3328-XYZ-ubuntu-noble-core-6.1-arm64-YYYYMMDD.img.zip | noble | 64-bit Ubuntu image file based on Ubuntu core 24.04 64bit | 6.1.y |
| | rk3328-XYZ-openmediavault-6.1-YYYYMMDD.img.gz | Shaitan | OpenMediaVault NAS system, base on Debian 12 | 6.1.y |
| | rk3328-XYZ-buildroot-4.19-arm64-YYYYMMDD.img.gz | -- | Buildroot image file with Qt5-wayland (base on Rockchip Linux SDK) | 4.19.y |
| | rk3328-XYZ-friendlywrt-24.10-YYYYMMDD.img.gz | 24.10 | FriendlyWrt, based on OpenWrt 24.10 | 6.6.y |
| | rk3328-XYZ-friendlywrt-24.10-docker-YYYYMMDD.img.gz | 24.10 | FriendlyWrt with Docker, based on OpenWrt 24.10 | 6.6.y |
| | rk3328-XYZ-friendlywrt-23.05-YYYYMMDD.img.gz | 23.05 | FriendlyWrt, based on OpenWrt 23.05 | 6.6.y |
| | rk3328-XYZ-friendlywrt-23.05-docker-YYYYMMDD.img.gz | 23.05 | FriendlyWrt with Docker, based on OpenWrt 23.05 | 6.6.y |
| **Other Image** | | | | |
| | FriendlyWrt (Github Actions) | 24.10,23.05 | FriendlyWrt (https://github.com/friendlyarm/Actions-FriendlyWrt/releases) | 6.6.y |
| | Alpine-Linux (Github Actions) | - | Alpine-Linux (https://github.com/friendlyarm/Actions-Alpine-Linux/releases) | 6.6.y |
| | rk3328-XYZ-multiple-os-YYYYMMDD-25g.img.gz | - | It contains multiple OS image files, making it convenient for testing different operating systems, this image disables automatic flashing at startup; you will need to manually select the OS to flash. | |

#### 4.4.1.2 Tools (optional)

Visit download link (http://download.friendlyelec.com/NanoPiR2SPlus) to download tools (in the "05_Tools" directory).

| Filename | Description |
|---|---|
| win32diskimager.rar | This program is designed to write a raw disk image to a removable device or backup a removable device to a raw image file |
| SD Card Formatter | A program (application) that allows easy and quick clear the SD card |
| RKDevTool_Release_v2.84.zip | Rockchip flashing tool, for USB upgrade |

### 4.4.2 Flashing the OS to the microSD card

Follow the steps below:

- Get an 8G microSD card;
- Visit download link (http://download.friendlyelec.com/NanoPiR2SPlus)to download image files (in the "01_Official images/01_SD card images" directory);
- Download the win32diskimager tool (in the "05_Tools" directory), or use your preferred tool;
- Extract the .gz format compressed file to get the .img format image file;
- Run the win32diskimager utility under Windows as administrator. On the utility's main window select your SD card's drive, the wanted image file and click on "write" to start flashing the SD card.
- Take out the SD and insert it to NanoPi-R2S-Plus's microSD card slot;

- Power on NanoPi-R2S-Plus and it will be booted from your TF card, some models may require pressing the Power button to start;

### 4.4.3 Install OS to eMMC

#### 4.4.3.1 Option 1: Install OS via TF Card

This method firstly boots a mini Linux from a TF card and then automatically runs an EFlasher utility to install the OS to eMMC. You can connect your system to an HDMI monitor and watch its progress.
This is optional. You can watch its progress by observing its LEDs as well:

By default, flashing starts automatically upon power-up, so be sure to back up the data in eMMC. If you don't want it to start automatically, you can use image file with a filename containing the words 'multiple-os' and manually select the OS you want to flash on the interface.

##### 4.4.3.1.1 Flash Official OS to eMMC

Follow the steps below:

- Get an SDHC card with a minimum capacity of 8G
- Visit download link (http://download.friendlyelec.com/NanoPiR2SPlus)to download image files (in the "01_Official images/02_SD-to-eMMC images" directory) and win32diskimager tool (in the "05_Tools" directory);
- Extract the .gz format compressed file to get the .img format image file;
- Run the win32diskimager utility under Windows as administrator. On the utility's main window select your SD card's drive, the wanted image file and click on "write" to start flashing the SD card.
- Eject your SD card and insert it to NanoPi-R2S-Plus's microSD card slot.
- Turn on NanoPi-R2S-Plus, it will boot from the SD card and automatically run EFlasher to install the OS to the board's eMMC.
- After flashing is complete, eject the SD card from NanoPi-R2S-Plus, NanoPi-R2S-Plus will automatically reboot and boot from eMMC.

##### 4.4.3.1.2 Flash third party OS (Image file) to eMMC

- Auto Install (Default Behavior)

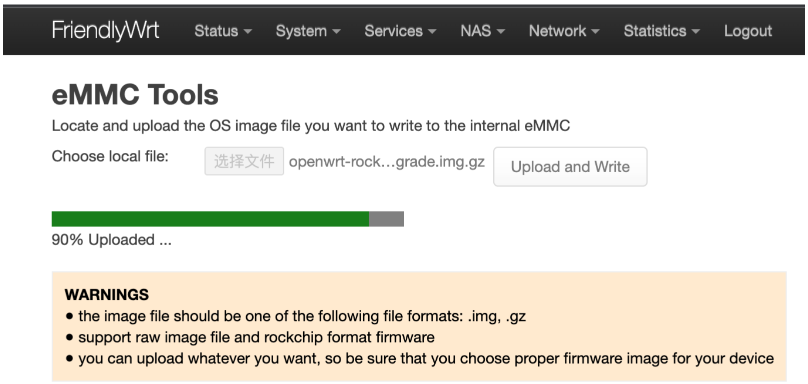
1) Download an "eflasher" firmware from network drive (http://download.friendlyelec.com/NanoPiR2SPlus)(in the "01_Official images/02_SD-to-eMMC images" directory), extract it and install it to a TF card ;
2) Eject and insert the TF card to your PC, after a "FriendlyARM" device shows up(Under Linux, it is a "FriendlyARM" directory), copy the image file ending with .raw or .gz into the directory (Note: if your file is in .img format, please rename it to .raw format).
3) Open the eflasher.conf file on the TF card, set "autoStart=" to the name of your image file, such as:

```
autoStart=openwrt-rockchip-armv8_nanopi-ext4-sysupgrade.img.gz
```

In addition to third-party image, official image files which with the '-sd-' word in the filename are also supported, for example: rk3NNN-sd-friendlywrt-24.10-YYYYMMDD.img.gz
4) Eject the TF card, insert the TF card to NanoPi-R2S-Plus, power it on it will automatically install your firmware. You can watch the installation progress by observing the LEDs' status.

#### 4.4.3.2 Option 2: Install OS on Web Page

Get a TF card which has been installed with FriendlyWrt, log in FriendlyWrt on the web page, click on "System" ->"eMMC Tools". Click on "Select file" to select your wanted image file, either an official image (filename containing '-sd-') or a third party image. The file should be a ".gz" or ".img" file.
After a file is selected, click on "Upload and Write" to start installing an OS.



eMMC Tools

Locate and upload the OS image file you want to write to the internal eMMC

Choose local file: 选择文件 openwrt-rock…grade.img.gz Upload and Write

90% Uploaded ...

**WARNINGS**

- the image file should be one of the following file formats: .img, .gz
- support raw image file and rockchip format firmware
- you can upload whatever you want, so be sure that you choose proper firmware image for your device

After installation is done, eject the SD card, the system will automatically reboot and load the OS from eMMC. After the OS begins to load, if the system LED is flashing and the network LED is on, it means the the OS has loaded successfully. If the OS is FriendlyWrt, you can click on “Go to Homepage” to enter the homepage. For official OS, you need select the file with the filename containing '-sd-', for example: rk3NNN-sd-friendlywrt-24.10-YYYYMMDD.img.gz, the compression file only supports the .gz format. If the file is too large, you can compress it into .gz format before uploading.

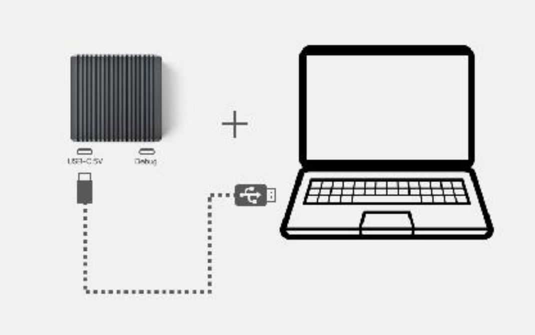
#### 4.4.3.3 Option 3: Install OS via USB

##### 4.4.3.3.1 Step 1: Install USB Driver and Tools/Utilities

Download a driver file DriverAssitant_v5.12.zip under the “tools” directory from network drive (http://download.friendlyelec.com/NanoPiR2SPlus), extract and install it.
Under the same directory, download a utility RKDevTool_Release_v2.84.zip and extract it.

##### 4.4.3.3.2 Step 2: Connect NanoPi-R2S-Plus to PC and Enter Installation Mode

Press and hold the “Mask” key, Use a USB C-to-A cable, connect NanoPi-R2S-Plus to a PC， After the status LED has been on for at least 3 seconds, release the Mask key;



##### 4.4.3.3.3 Step 3: Install image to eMMC

A firmware in general is packaged in either of the two options: the first is an whole image (ie, update.img) which is often offered by third party developers, the second is that an image is packaged and placed in multiple partition images. FriendlyElec offers an image in the latter option.

- Option 1: Install whole image (ie, update.img)

On a PC which has the extracted RKDevTool_Release_v2.84 utility, go to the RKDevTool_Release_v2.84 directory, run the RKDevTool.exe file. If everything works, you will see a “Found a new Maskrom device” message on the utility;
Go to “Upgrade Firmware(升级固件)”, click on “Firmware(固件)”, select your wanted image file, and click on “Upgrade(升级)” to install. After installation is done, your board will reboot automatically and load the system from eMMC;

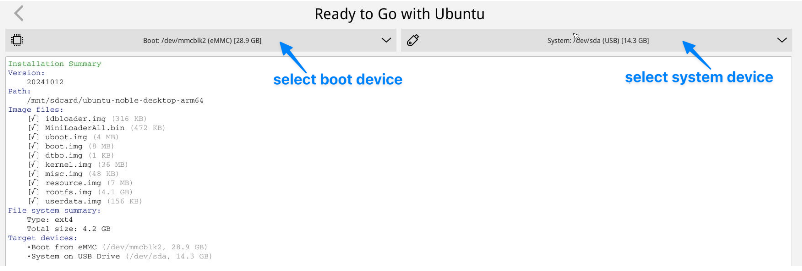
- Option 2: Install OS that is packaged & placed in multiple partition images

Go to network drive (http://download.friendlyelec.com/NanoPiR2SPlus) to download your needed package and extract it (in the "01_Official images/03_USB upgrade images). After it is extracted, you will see some utilities and a configuration file under the directory. double click on RKDevTool.exe, you will see a “Found a new Maskrom device” message on the utility. Click on the “Execute”, wait a moment and it will be installed. After installation is done your system will automatically reboot and load the system from eMMC.

### 4.4.4 Installing the System to M.2 or USB Drive

You can use a TF card to boot the eFlasher system, allowing the boot and system to be installed on different storage devices. However, since the CPU doesn’t support booting directly from M.2 and USB devices, the system can be installed on M.2 and USB devices, but the boot must still be installed on eMMC or a TF card.
Steps are as follows:

- Prepare a TF card with a capacity of 32GB or larger.
- Visit [the download link here](http://download.friendlyelec.com/NanoPiR2SPlus) to download the firmware file named XXXX-eflasher-multiple-os-YYYYMMDD-30g.img.gz (located in the “01_Official images/02_SD-to-eMMC images” directory).
- Flash the firmware to the TF card, connect the storage device you intend to use on NanoPi-R2S-Plus, insert the TF card and power on, we need to perform the operations in the eFlasher GUI. If your NanoPi-R2S-Plus does not have a display interface, you can use VNC; refer to Using VNC to Operate eFlasher (https://wiki.friendlyelec.com/wiki/index.php/EFlasher#Remote_Control_eFlasher_via_VNC).
- In the eFlasher GUI, select the OS to install, and in the OS settings interface, choose the destination for boot installation (typically eMMC), then choose the destination for system installation (options include eMMC, M.2 hard drive, USB storage, etc.), as shown below:



Ready to Go with Ubuntu

Boot: /dev/mmcblk2 (eMMC) [28.9 GB]

System: /dev/sda (USB) [14.3 GB]

select boot device

select system device

```
Installation Summary
Version:
    20241012
Path:
    /mnt/sdcard/ubuntu-noble-desktop-arm64
Image files:
    [√] idbloader.img (316 KB)
    [√] MiniLoaderAll.bin (472 KB)
    [√] uboot.img (4 MB)
    [√] boot.img (8 MB)
    [√] dtbo.img (1 KB)
    [√] kernel.img (36 MB)
    [√] misc.img (48 KB)
    [√] resource.img (7 MB)
    [√] rootfs.img (4.1 GB)
    [√] userdata.img (156 KB)
File system summary:
    Type: ext4
    Total size: 4.2 GB
Target devices:
    •Boot from eMMC (/dev/mmcblk2, 28.9 GB)
    •System on USB Drive (/dev/sda, 14.3 GB)
```

- If no eMMC is available, the TF card can serve as the boot by inserting another TF card into the USB port via a USB card reader and selecting it as the boot destination, enabling booting from the TF card with the system stored on the M.2 or USB drive.
- After flashing, eject the SD card from NanoPi-R2S-Plus. If booting from eMMC, NanoPi-R2S-Plus will automatically restart into the newly flashed system. If boot installation is on a TF card, power off, insert the boot TF card, and power on again.
- For a more detailed installation guide, please refer to this link (https://wiki.friendlyelec.com/wiki/index.php/EFlasher#Installing_the_System_to_M.2_or_USB_Drive_.28Rockchip_Platform_Only.29).

# 5 Work with FriendlyWrt

## 5.1 Introduction to FriendlyWrt

FriendlyWrt is a customized system made by FriendlyElec based on an OpenWrt distribution. It is open source and well suitable for developing IoT applications, NAS applications etc.

## 5.2 First boot

For the first boot, the system needs to do the following initialization work：
1）Extended root file system
2）Initial setup（will execute /root/setup.sh)
So you need to wait for a while (about 2~3 minutes) to boot up for the first time, and then set FriendlyWrt, you can enter the ttyd terminal on the openwrt webpage, when the prompt is displayed as root@FriendlyWrt, it means the system has been initialized.

```
root@FriendlyWrt
```

## 5.3 Account & Password

The default password is password (empty password in some versions). Please set or change a safer password for web login and ssh login. It is recommended to complete this setting before connecting NanoPi-R2S-Plus to the Internet.

## 5.4 Login FriendlyWrt

Connect the PC to the LAN port of NanoPi-R2S-Plus. If your PC without a built-in ethernet port, connect the LAN port of the wireless AP to the LAN port of NanoPi-R2S-Plus, and then connect your PC to the wireless AP via WiFi , Enter the following URL on your PC's browser to access the admin page:

- http://friendlywrt/
- http://192.168.2.1/
- http://[fd00:ab:cd::1]

The above is the LAN port address of NanoPi-R2S-Plus. The IP address of the WAN port will be dynamically obtained from your main router through DHCP.

## 5.5 Recommended security settings

The following settings are highly recommended to complete before connecting NanoPi-R2S-Plus to the Internet。

- Set a secure password
- Only allow access to ssh from lan, change the port
- Check the firewall settings

Set up as you wish.

## 5.6 Change LAN IP in LuCI

1) Click on Network → Interfaces, then click on the Edit button of the LAN Network;
2) In General Setup tab, input new IP address (for example: 192.168.11.1), click "Save" and then click "Save & Apply";
3) On the pop-up window with the title “Connectivity change“, click "Apply and revert on connectivity loss";
4) Wait a moment, enter the new address in your computer's browser and login to FriendlyWrt;

## 5.7 Safe shutdown operation

Enter the "Services" -> "Terminal", enter the "poweroff" command and hit enter, wait until the led light is off, and then unplug the power supply.

## 5.8 Soft Factory Reset

Enter "System"->"Backup/Flash firmware"，Click “Perform reset“ Button, Your device's settings will be reset to defaults like when FriendlyWrt was first installed. You can also do this in the terminal:

```
firstboot && reboot
```

## 5.9 Install Software Packages

### 5.9.1 Set up openwrt official opkg source

```
sed -i -e 's/mirrors.cloud.tencent.com/downloads.openwrt.org/g' /etc/opkg/distfeeds.conf
opkg update
```

### 5.9.2 Update Package List

Before install software packages update the package list:

```
$ opkg update
```

### 5.9.3 List Available Packages

```
$ opkg list
```

### 5.9.4 List Installed Packages

```
$ opkg list-installed
```

### 5.9.5 Install Packages

```
$ opkg install <package names>
```

### 5.9.6 Remove Packages

```
$ opkg remove <package names>
```

## 5.10 Disable IPv6

```
. /root/setup.sh
disable_ipv6
reboot
```

## 5.11 Configure the function of the user button

By default, the user button is configured to reboot the device, as shown below:

```
echo 'BTN_1 1 /sbin/reboot' >> /etc/triggerhappy/triggers.d/example.conf
```

You can change its behavior by changing the configuration file above.

## 5.12 Configuring Quectel EC20 (4G module) dial-up networking

- Go to "Network" -> "Interfaces"
- Click "Delete" next to "WAN6", then click "Save & Apply"
- Click "Edit" next to "WAN", in the "Device" drop-down menu, select "Ethernet Adapter: wwan0", in the "Protocol" drop-down menu, select "QMI Cellular" and click "Switch Protocol"
- Click the "Modem Device" drop-down menu, select "/dev/cdc-wdm0", fill in the APN information (e.g. for China Mobile, enter "cmnet")
- Click "Save" to close the dialog, Finally, click "Save & Apply" at the bottom of the page to initiate the dial-up process
- Devices connected to LAN will have access to the Internet, If your device has a WiFi module, you can enable wireless AP functionality on the "Wireless" page and connect to the Internet via devices connected wirelessly

## 5.13 Some common issues of FriendlyWrt

- Unable to dial up
  - Go to "Network" -> "Firewall" and set "Inbound Data", "Outbound Data" and "Forwarding" in "WAN Zone" to "Accept";
  - If you still cannot access the Internet, you can try to turn off IPV6;
- Dial-up successful, but no outgoing traffic
  - Go to "Services" -> "Terminal" and type "fw4 reload" to try to reload the firewall settings again;
- Unable to power on
  - Try to replace the power adapter and cable. It is recommended to use a power supply with specifications above 5V/2A;
  - Note that some fast chargers with Type-C interface will have a delay, it may take a few seconds to start providing power;
- When doing secondary routing, the computer cannot connect to the Internet
  - If your main network is IPv4, and NanoPi-R2S-Plus works in IPv6, the computer may not be able to connect to the Internet. It is recommended to turn off IPv6 (the method is described later in this WiKi), or switch the main route to IPv6;
- If you have questions or have better suggestions, please send an email to techsupport@friendlyarm.com;

### 5.14 Use USB2LCD to view IP and temperature

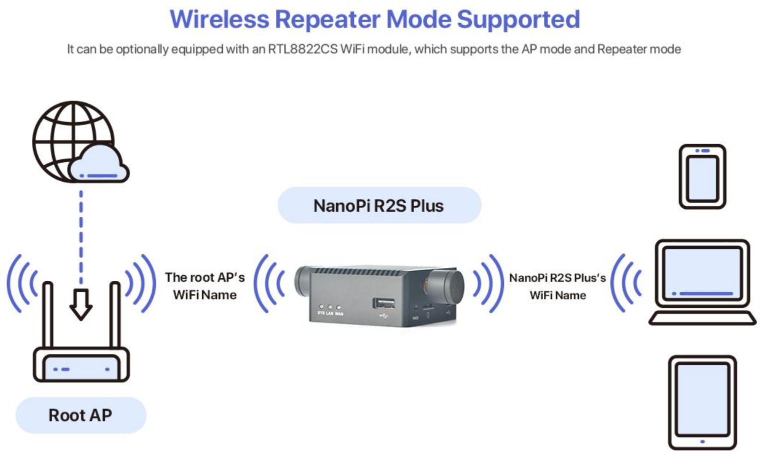
Plug the USB2LCD module to the USB interface ofNanoPi-R2S-Plus and power on, the IP address and CPU temperature will be displayed on the LCD:

### 5.15 How to use SDIO WiFi

#### 5.15.1 AP Mode

- FriendlyWrt's wireless function is disabled by default, to enable WiFi, you can click "Network > Wireless", and then click the "Enable" button.
- Search for a WiFi hotspot with a name like FriendlyWrt-xx:yy:zz on your phone, enter the default password "password" to connect.
- SDIO WiFi models supported: rtl8822cs

#### 5.15.2 Wireless Repeater Mode


Wireless Repeater Mode Supported
It can be optionally equipped with an RTL8822CS WiFi module, which supports the AP mode and Repeater mode
NanoPi R2S Plus
The root AP's WiFi Name
NanoPi R2S Plus's WiFi Name
Root AP

- Connect to the wireless router

Click "Network" -> "Wireless", Click the "Scan" button next to "Generic MAC80211 802.11ac/b/g/n" to scan for networks, select the router you want to connect and then click the "Join Network" button,
In the network joining configuration page, turn on "Replace wireless configuration", enter the key in the "WPA passphrase" field, keep other configurations as default, click "Submit",
Click "Advanced Settings", input "wlan0" in "Interface Name" field, and then click "Save",
Finally, click "Save and Apply", if normal, you can see the IP address on the FriendlyWrt homepage.

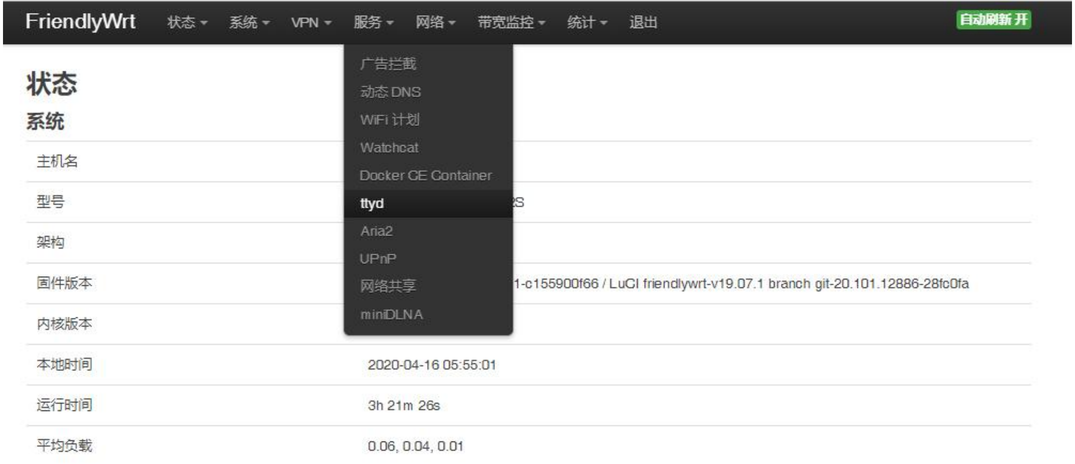
- Create wireless hotspot

Click "Network" -> "Wireless", Click the "Add" button next to "Generic MAC80211 802.11ac/b/g/n"， the "Edit wireless network" page will pop up，
Set the wireless channel in the "Operating frequency" field (e.g., "40 (5200 MHz)"),
In the "ESSID" field, input the hotspot name,
In the "Network" field, select "lan",
Click on "Wireless Security", set the encryption type,
Click "Advanced Settings", input "wlan1" in "Interface Name" field, then click "Save" and "Save and Apply",
It requires a reboot to take effect, click on "System" menu, then select "Reboot".

## 5.16 How to use USB WiFi

### 5.16.1 Check USB WiFi Device with Command Line Utility

(1) Click on "services>ttyd" to start the command line utility



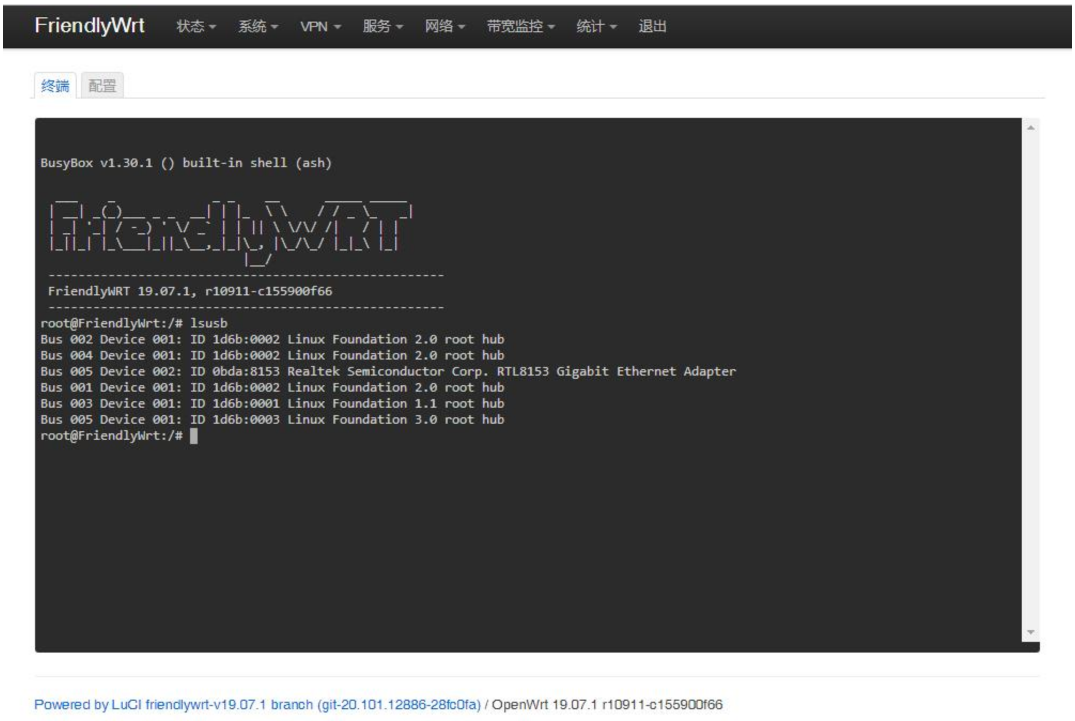
状态

系统

| | |
|---|---|
| 主机名 | |
| 型号 | S |
| 架构 | |
| 固件版本 | 1-c155900f66 / LuCI friendlywrt-v19.07.1 branch git-20.101.12886-28fc0fa |
| 内核版本 | |
| 本地时间 | 2020-04-16 05:55:01 |
| 运行时间 | 3h 21m 26s |
| 平均负载 | 0.06, 0.04, 0.01 |

(2) Make sure no USB devices are connected to your board and run the following command to check if any USB devices are connected or not

```
lsusb
```



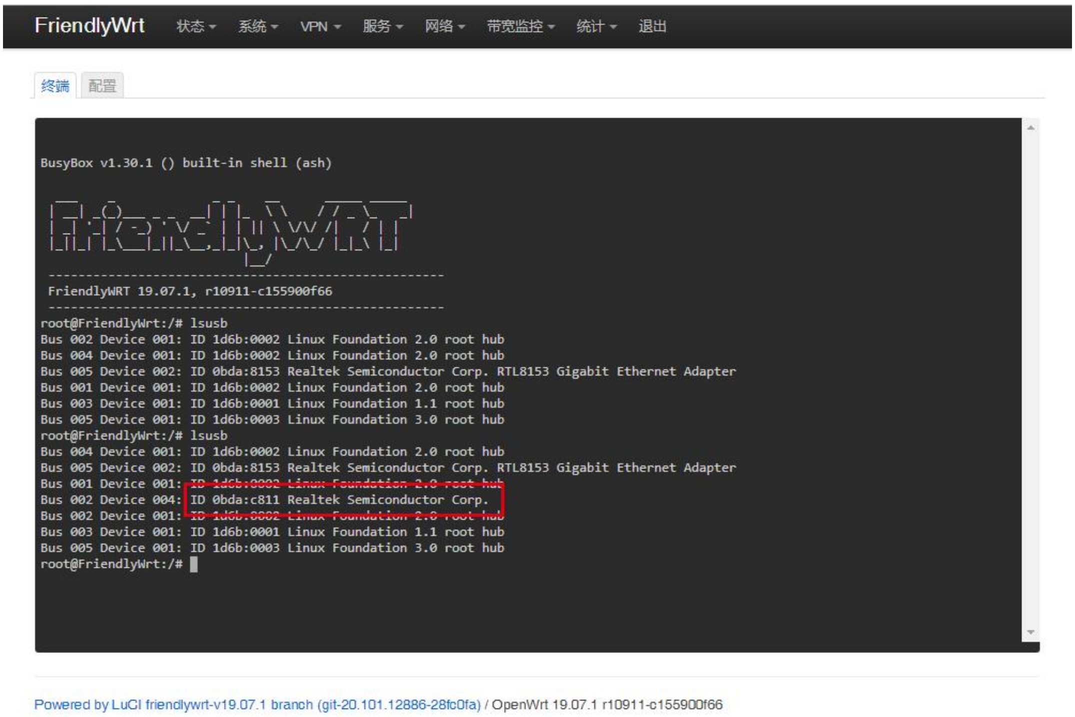
```
BusyBox v1.30.1 () built-in shell (ash)

 -----------------------------------------------------
 FriendlyWRT 19.07.1, r10911-c155900f66
 -----------------------------------------------------
root@FriendlyWrt:/# lsusb
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 002: ID 0bda:8153 Realtek Semiconductor Corp. RTL8153 Gigabit Ethernet Adapter
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
root@FriendlyWrt:/#
```

Powered by LuCI friendlywrt-v19.07.1 branch (git-20.101.12886-28fc0fa) / OpenWrt 19.07.1 r10911-c155900f66

(3) Connect a USB WiFi device to the board and run the command again

```
lsusb
```

You will see a new device is detected. In our test the device's ID was 0BDA:C811



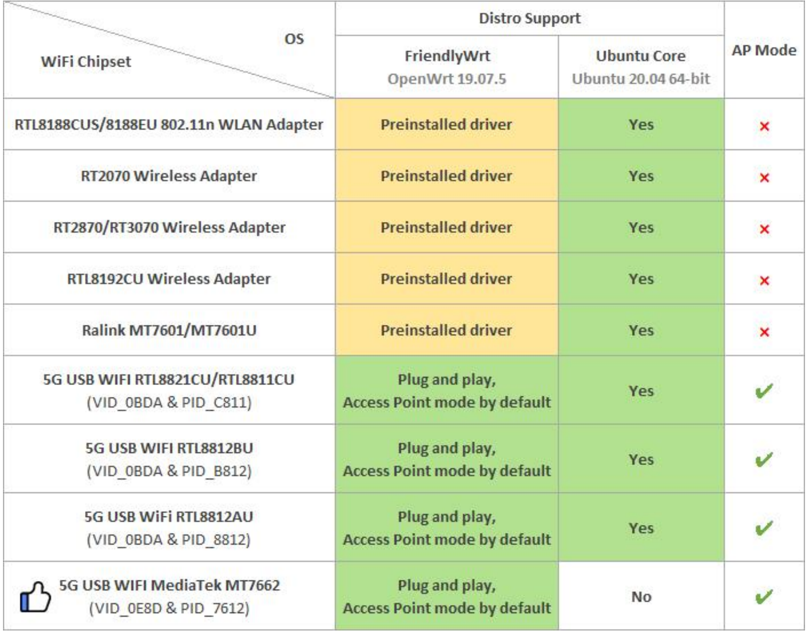
```
BusyBox v1.30.1 () built-in shell (ash)

  ___    _             _ _      __      _____ _____
 | __| _(_)___ _ _  __| | |_  _ \ \    / / _ \_   _|
 | _| '_| / -_) ' \/ _` | | || | \ \/\/ /|   / | |
 |_||_| |_\___|_||_\__,_|_|\_, |  \_/\_/ |_|_\ |_|
                           |__/
 -----------------------------------------------------
 FriendlyWRT 19.07.1, r10911-c155900f66
 -----------------------------------------------------
root@FriendlyWrt:/# lsusb
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 002: ID 0bda:8153 Realtek Semiconductor Corp. RTL8153 Gigabit Ethernet Adapter
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
root@FriendlyWrt:/# lsusb
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 002: ID 0bda:8153 Realtek Semiconductor Corp. RTL8153 Gigabit Ethernet Adapter
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 004: ID 0bda:c811 Realtek Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
root@FriendlyWrt:/#
```

Powered by LuCI friendlywrt-v19.07.1 branch (git-20.101.12886-28fc0fa) / OpenWrt 19.07.1 r10911-c155900f66

(4) Type your device's ID (in our case it was "0BDA:C811" or "VID_0BDA&PID_C811") in a search engine and you may find a device that matches the ID. In our case the device we got was Realtek 8811CU.

### 5.16.2 Configure a USB WiFi Device as AP

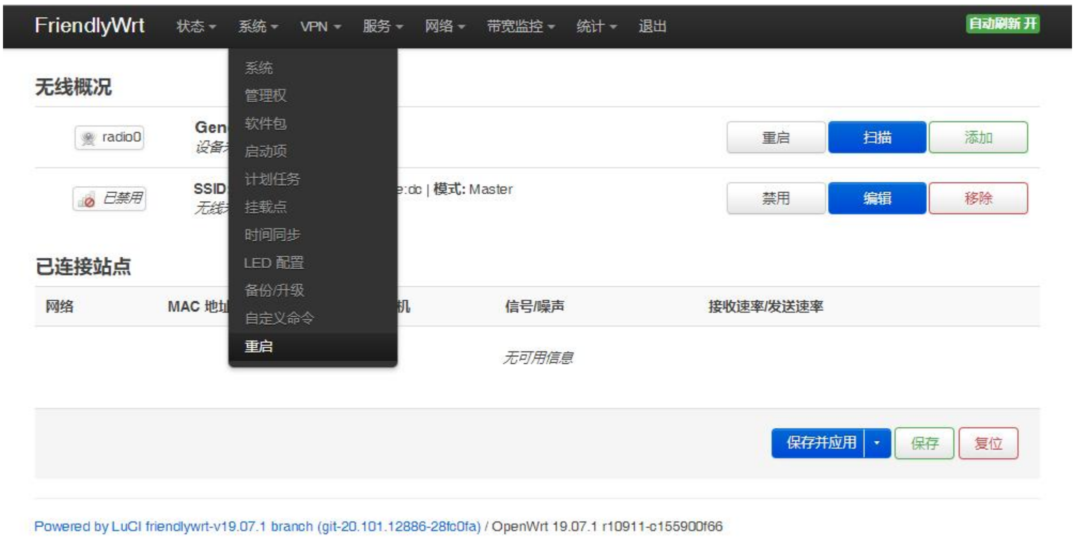
(1) Connect a USB WiFi device to the NanoPi-R2S-Plus. We recommend you to use the following devices:



| OS / WiFi Chipset | Distro Support | | AP Mode |
|---|---|---|---|
| | FriendlyWrt<br>OpenWrt 19.07.5 | Ubuntu Core<br>Ubuntu 20.04 64-bit | |
| RTL8188CUS/8188EU 802.11n WLAN Adapter | Preinstalled driver | Yes | ✗ |
| RT2070 Wireless Adapter | Preinstalled driver | Yes | ✗ |
| RT2870/RT3070 Wireless Adapter | Preinstalled driver | Yes | ✗ |
| RTL8192CU Wireless Adapter | Preinstalled driver | Yes | ✗ |
| Ralink MT7601/MT7601U | Preinstalled driver | Yes | ✗ |
| 5G USB WIFI RTL8821CU/RTL8811CU<br>(VID_0BDA & PID_C811) | Plug and play,<br>Access Point mode by default | Yes | ✓ |
| 5G USB WIFI RTL8812BU<br>(VID_0BDA & PID_B812) | Plug and play,<br>Access Point mode by default | Yes | ✓ |
| 5G USB WiFi RTL8812AU<br>(VID_0BDA & PID_8812) | Plug and play,<br>Access Point mode by default | Yes | ✓ |
| 👍 5G USB WIFI MediaTek MT7662<br>(VID_0E8D & PID_7612) | Plug and play,<br>Access Point mode by default | No | ✓ |

Note: devices that match these VID&PIDs would most likely work.

(2) Click on "System>Reboot" and reboot your NanoPi-R2S-Plus



无线概况

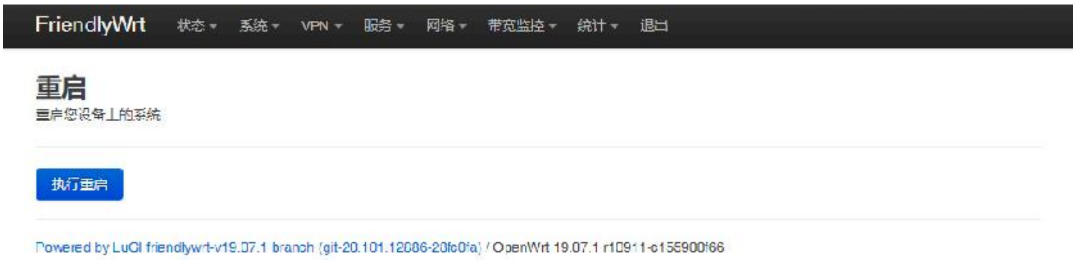
radio0 Gen… 设备…

已禁用 SSID… 无线… …e:do | **模式:** Master

已连接站点

| 网络 | MAC 地址 | …机 | 信号/噪声 | 接收速率/发送速率 |
|---|---|---|---|---|
| | | | *无可用信息* | |

Powered by LuCI friendlywrt-v19.07.1 branch (git-20.101.12886-28fc0fa) / OpenWrt 19.07.1 r10911-c155900f66

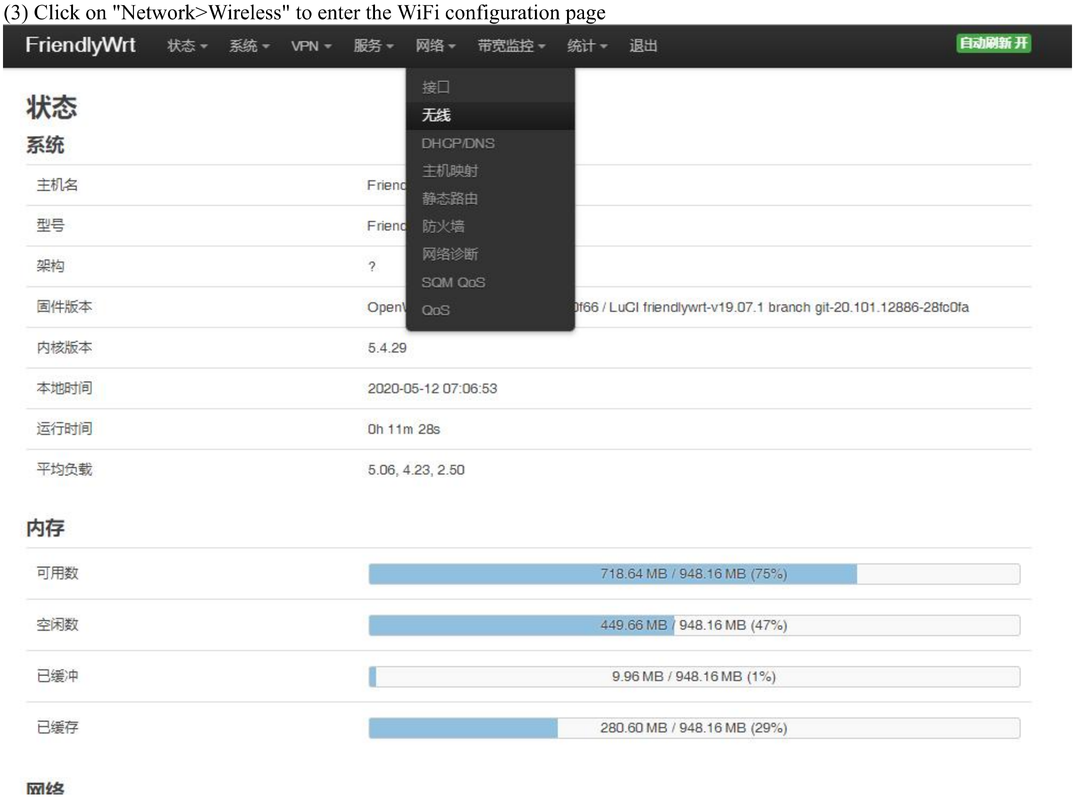


重启

重启您设备上的系统

Powered by LuCI friendlywrt-v19.07.1 branch (git-20.101.12886-28fc0fa) / OpenWrt 19.07.1 r10911-c155900f66

(3) Click on "Network>Wireless" to enter the WiFi configuration page



状态

系统

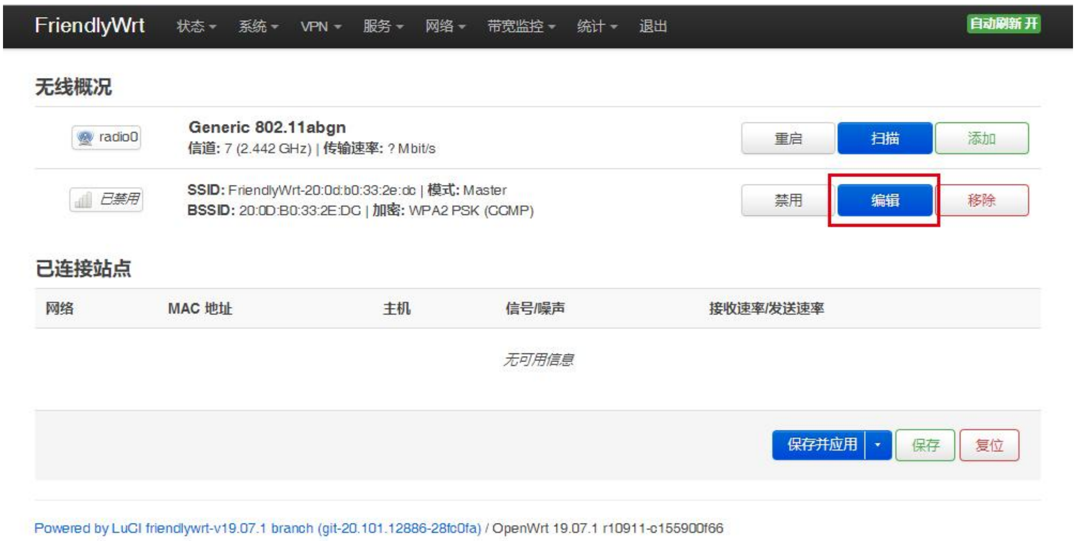
| | |
|---|---|
| 主机名 | Friend… |
| 型号 | Friend… |
| 架构 | ? |
| 固件版本 | OpenW… …0f66 / LuCI friendlywrt-v19.07.1 branch git-20.101.12886-28fc0fa |
| 内核版本 | 5.4.29 |
| 本地时间 | 2020-05-12 07:06:53 |
| 运行时间 | 0h 11m 28s |
| 平均负载 | 5.06, 4.23, 2.50 |

内存

| | |
|---|---|
| 可用数 | 718.64 MB / 948.16 MB (75%) |
| 空闲数 | 449.66 MB / 948.16 MB (47%) |
| 已缓冲 | 9.96 MB / 948.16 MB (1%) |
| 已缓存 | 280.60 MB / 948.16 MB (29%) |

网络

(4) Click on "Edit" to edit the configuration



无线概况

radio0 Generic 802.11abgn
信道: 7 (2.442 GHz) | 传输速率: ? Mbit/s

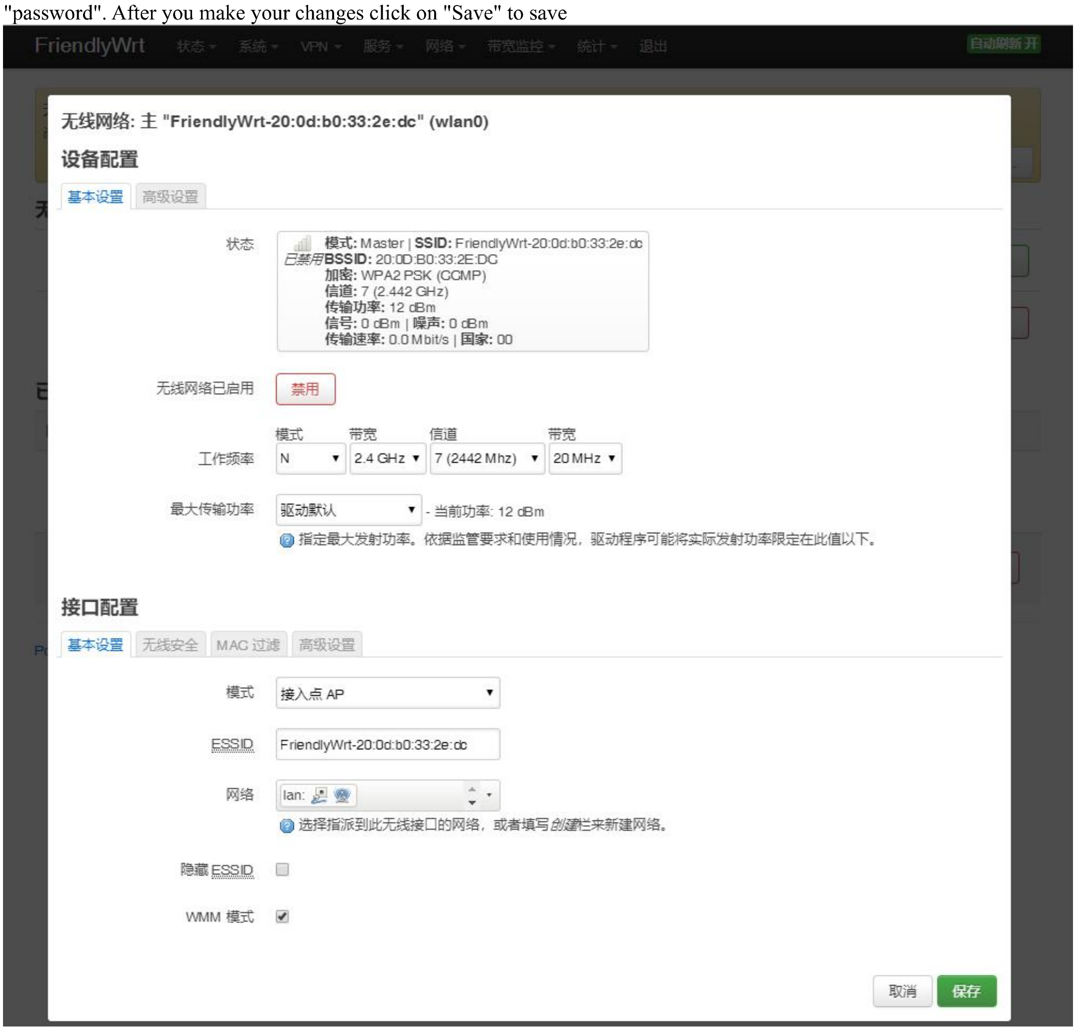
已禁用 SSID: FriendlyWrt-20:0d:b0:33:2e:dc | 模式: Master
BSSID: 20:0D:B0:33:2E:DC | 加密: WPA2 PSK (CCMP)

已连接站点

| 网络 | MAC 地址 | 主机 | 信号/噪声 | 接收速率/发送速率 |
|---|---|---|---|---|
| | | 无可用信息 | | |

Powered by LuCI friendlywrt-v19.07.1 branch (git-20.101.12886-28fc0fa) / OpenWrt 19.07.1 r10911-c155900f66

(5) On the "Interface Configuration" page you can set the WiFi mode and SSID, and then go to "Wireless Security" to change the password. By default the password is "password". After you make your changes click on "Save" to save



无线网络: 主 "FriendlyWrt-20:0d:b0:33:2e:dc" (wlan0)

设备配置

状态: 已禁用 模式: Master | SSID: FriendlyWrt-20:0d:b0:33:2e:dc
BSSID: 20:0D:B0:33:2E:DC
加密: WPA2 PSK (CCMP)
信道: 7 (2.442 GHz)
传输功率: 12 dBm
信号: 0 dBm | 噪声: 0 dBm
传输速率: 0.0 Mbit/s | 国家: 00

无线网络已启用

工作频率: 模式 N | 带宽 2.4 GHz | 信道 7 (2442 Mhz) | 带宽 20 MHz

最大传输功率: 驱动默认 - 当前功率: 12 dBm
指定最大发射功率。依据监管要求和使用情况，驱动程序可能将实际发射功率限定在此值以下。

接口配置

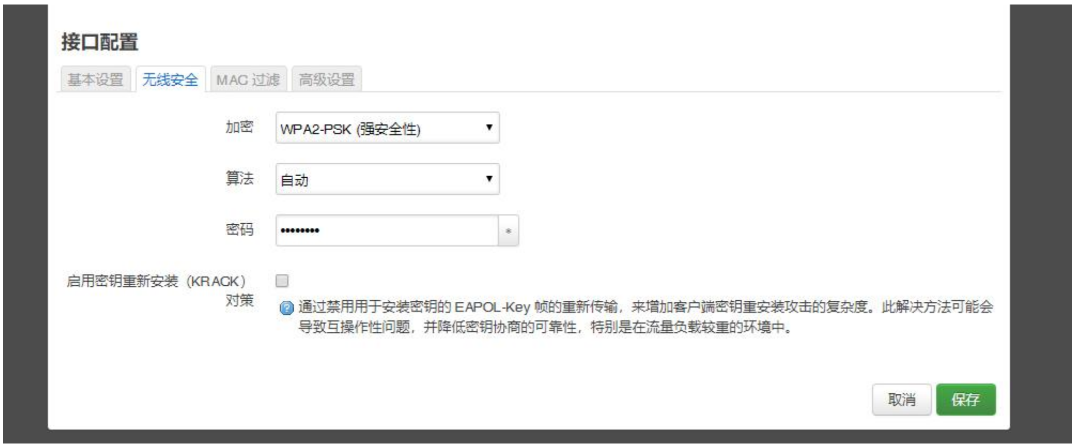
模式: 接入点 AP

ESSID: FriendlyWrt-20:0d:b0:33:2e:dc

网络: lan:
选择指派到此无线接口的网络，或者填写*创建*栏来新建网络。

隐藏 ESSID ☐

WMM 模式 ☑



接口配置

加密: WPA2-PSK (强安全性)

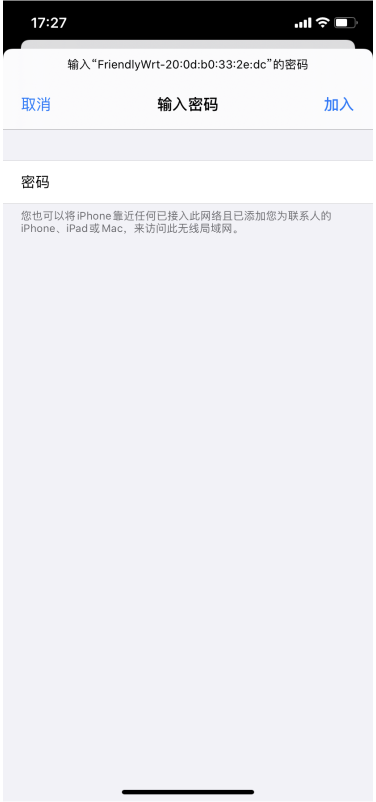
算法: 自动

密码: ••••••••

启用密钥重新安装（KRACK）对策 □

通过禁用用于安装密钥的 EAPOL-Key 帧的重新传输，来增加客户端密钥重安装攻击的复杂度。此解决方法可能会导致互操作性问题，并降低密钥协商的可靠性，特别是在流量负载较重的环境中。

(6) After you change the settings you can use a smartphone or PC to search for WiFi



输入“FriendlyWrt-20:0d:b0:33:2e:dc”的密码

输入密码

密码

您也可以将iPhone靠近任何已接入此网络且已添加您为联系人的iPhone、iPad或Mac，来访问此无线局域网。

### 5.16.3 Common USB WiFi issues

1) It is recommended to plug in the usb wifi in the off state, then power it on, FriendlyWrt will automatically generate the configuration file /etc/config/wireless, if not, see if there is wlan0 by ifconfig -a, if there is no wlan0, usually there is no driver.
2) If ifconfig -a sees wlan0, but the hotspot is not working properly, try changing the channel and country code, an inappropriate country code can also cause the WiFi to not work.
3) Some USB WiFis (e.g. MTK MT7662) work in CD-ROM mode by default and need to be switched by usb_modeswitch, you can try to add usb_modeswitch configuration to the following directory: /etc/usb_modeswitch.d.

### 5.16.4 Change the default WiFi hotspot configuration

FriendlyWrt sets the country, hotspot name and other parameters for USB WiFi by default, with the aim of being as plug-and-play as possible, but this does not guarantee that all modules will be compatible with this setting, you can change these behaviors by modifying the following file：

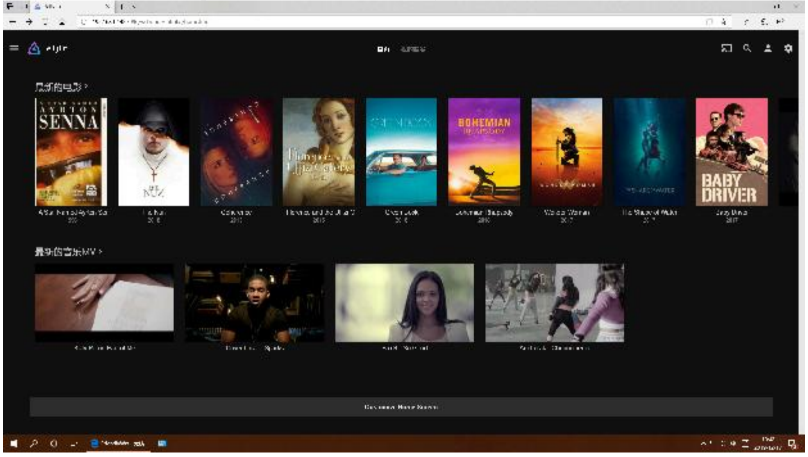
```
/lib/wifi/mac80211.sh
```

## 5.17 Work with Docker Applications

### 5.17.1 Work with Docker: Install JellyFin

```
mkdir -p /jellyfin/config
mkdir -p /jellyfin/videos
docker run --restart=always -d -p 8096:8096 -v /jellyfin/config:/config -v /jellyfin/videos:/videos jellyfin/jellyfin:10.1.0-arm64 -name myjellyfin
```

After installation, visit port 8096 and here is what you would find:



### 5.17.2 Work with Docker: Install Personal Nextcloud

```
mkdir /nextcloud -p
docker run -d -p 8888:80  --name nextcloud  -v /nextcloud/:/var/www/html/ --restart=always --privileged=true  arm64v8/nextcloud
```

After installtion, visit port 8888.

### 5.17.3 Expand Docker Storage

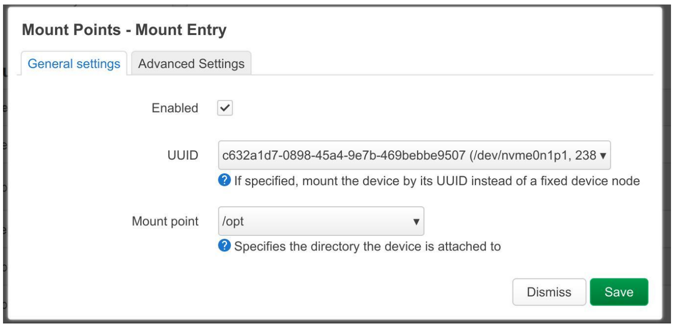
- Stop docker service first:

```
/etc/init.d/dockerd stop
```

- Rename the original /opt directory, create an empty /opt directory:

```
mv /opt /opt-old && mkdir /opt
```

- Format your drive as ext4, and mount it to the /opt directory:



Mount Points - Mount Entry

General settings | Advanced Settings

Enabled ☑

UUID: c632a1d7-0898-45a4-9e7b-469bebbe9507 (/dev/nvme0n1p1, 238 ▾

If specified, mount the device by its UUID instead of a fixed device node

Mount point: /opt ▾

Specifies the directory the device is attached to

Dismiss | Save

- Enter the command "mount | grep /opt" to check the mount status:

```
root@FriendlyWrt:~# mount | grep /opt
/dev/nvme0n1p1 on /opt type ext4 (rw,relatime)
root@FriendlyWrt:~#
```

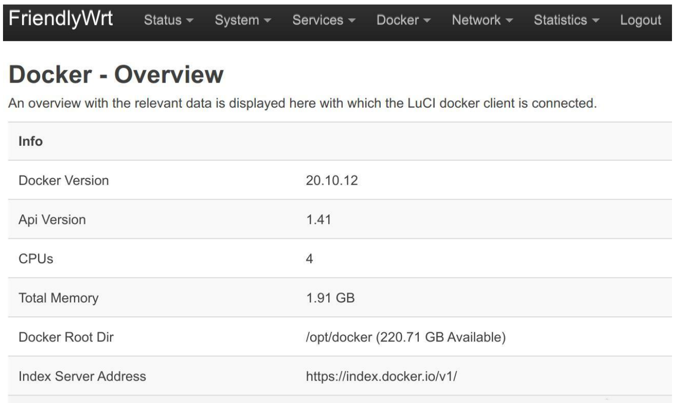
- Copy the files from the original /opt directory to the new /opt directory:

```
cp -af /opt-old/* /opt/ && rm -rf /opt-old
```

- Reboot the device

```
reboot
```

- After reboot, go to the "Docker" -> "Overview" page, check the information in the "Docker Root Dir" line, you can see that the Docker space has been expanded:



Docker - Overview

An overview with the relevant data is displayed here with which the LuCI docker client is connected.

| Info | |
|---|---|
| Docker Version | 20.10.12 |
| Api Version | 1.41 |
| CPUs | 4 |
| Total Memory | 1.91 GB |
| Docker Root Dir | /opt/docker (220.71 GB Available) |
| Index Server Address | https://index.docker.io/v1/ |

### 5.17.4 Docker FAQ and solutions

#### 5.17.4.1 Unable to access the network services provided by the Docker container

Solution:

- Go to the "Firewall" settings and set "Forwarding" to "Accept";
- Turn off "Software Offload";

## 5.18 Mount smbfs

```
mount -t cifs //192.168.1.10/shared /movie -o username=xxx,password=yyy,file_mode=0644
```

## 5.19 Use sdk to compile the package

### 5.19.1 Install the compilation environment

Download and run the following script on 64-bit Ubuntu (version 18.04+): How to setup the Compiling Environment on Ubuntu bionic (https://github.com/friendlyarm/build-env-on-ubuntu-bionic)

### 5.19.2 Download and decompress sdk from the network disk

The sdk is located in the toolchain directory of the network disk:

```
tar xvf openwrt-sdk-*-rockchip-armv8_gcc-11.2.0_musl.Linux-x86_64.tar.xz
# If the path is too long, it will cause some package compilation errors, so change the directory name here
mv openwrt-sdk-*-rockchip-armv8_gcc-11.2.0_musl.Linux-x86_64 sdk
cd sdk
./scripts/feeds update -a
./scripts/feeds install -a
```

### 5.19.3 Compile the package

download the source code of the example (a total of 3 examples are example1, example2, example3), and copy to the package directory:

```
git clone https://github.com/mwarning/openwrt-examples.git
cp -rf openwrt-examples/example* package/
rm -rf openwrt-examples/
```

Then enter the configuration menu through the following command:

```
make menuconfig
```

In the menu, select the following packages we want to compile (actually selected by default):

```
"Utilities" => "example1"
"Utilities" => "example3"
"Network" => "VPN" => "example2"
```

execute the following commands to compile the three software packages:

```
make package/example1/compile V=99
make package/example2/compile V=99
make package/example3/compile V=99
```

After the compilation is successful, you can find the ipk file in the bin directory, as shown below:

```
$ find ./bin -name example*.ipk
./bin/packages/aarch64_generic/base/example3_1.0.0-220420.38257_aarch64_generic.ipk
./bin/packages/aarch64_generic/base/example1_1.0.0-220420.38257_aarch64_generic.ipk
./bin/packages/aarch64_generic/base/example2_1.0.0-220420.38257_aarch64_generic.ipk
```

### 5.19.4 Install the ipk to NanoPi

You can use the scp command to upload the ipk file to NanoPi:

```
cd ./bin/packages/aarch64_generic/base/
scp example*.ipk root@192.168.2.1:/root/
```

Then use the opkg command to install them:

```
cd /root/
opkg install example3_1.0.0-220420.38257_aarch64_generic.ipk
opkg install example1_1.0.0-220420.38257_aarch64_generic.ipk
opkg install example2_1.0.0-220420.38257_aarch64_generic.ipk
```

## 5.20 Build FriendlyWrt using GitHub Actions

Please refre this link: https://github.com/friendlyarm/Actions-FriendlyWrt

# 6 Work with FriendlyCore

## 6.1 FriendlyCore User Account

- Non-root User:

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

- Root:

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

## 6.2 Update Software Packages

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

## 6.3 Setup Network Configurations

### 6.3.1 Set static IP address

By default "eth0" is assigned an IP address obtained via dhcp. If you want to change the setting you need to change the following file:

```
vi /etc/network/interfaces.d/eth0
```

For example if you want to assign a static IP to it you can run the following commands:

```
auto eth0
iface eth0 inet static
    address 192.168.1.231
    netmask 255.255.255.0
    gateway 192.168.1.1
```

### 6.3.2 Set a DNS

You also need to modify the following file to add the DNS configuration:

```
vi /etc/systemd/resolved.conf
```

For example, set to 192.168.1.1:

```
[Resolve]
DNS=192.168.1.1
```

Restart the systemd-resolved service with the following command:

```
sudo systemctl restart systemd-resolved.service
sudo systemctl enable systemd-resolved.service
```

### 6.3.3 Set up to use another network interface

To change the setting of "eth1" you can add a new file similar to eth0's configuration file under the /etc/network/interfaces.d/ directory.

## 6.4 Setup Wi-Fi

First, use the following command to check if Network-Manager is installed on your system:

```
which nmcli
```

If you have installed it, refer to this link to connect to WiFi: Use NetworkManager to configure network settings, If you do not have Network-Manager installed on your system, please refer to the following method to configure WiFi,
By default the WiFi device is "wlan0". You need to create a configuration file under "/etc/network/interfaces.d/" for WiFi:

```
vi /etc/network/interfaces.d/wlan0
```

Here is a sample wlan0 file:

```
auto lo
iface lo inet loopback
auto wlan0
iface wlan0 inet dhcp
wpa-driver wext
wpa-ssid YourWiFiESSID
wpa-ap-scan 1
wpa-proto RSN
wpa-pairwise CCMP
wpa-group CCMP
wpa-key-mgmt WPA-PSK
wpa-psk YourWiFiPassword
```

Please replace "YourWiFiESSID" and "YourWiFiPassword" with your WiFiESSID and password. After save and close the file you can connect to your WiFi source by running the following command:

```
sudo systemctl daemon-reload
sudo systemctl restart networking
```

After you power on your board it will automatically connect to your WiFi source.
Please note that if you use one TF card to boot multiple boards the WiFi device name will likely be named to "wlan1", "wlan2" and etc. You can reset it to "wlan0" by deleting the contents of the following file and reboot your board: /etc/udev/rules.d/70-persistent-net.rules

### 6.4.1 WiFi models supported

#### 6.4.1.1 M.2 WiFi Module

- RTL8822CE

#### 6.4.1.2 Usb Dongle

- RTL8821CU (Vid: 0BDA, Pid: C811) (Test sample:TP-Link TL-WDN5200H)
- RTL8812AU (Vid: 0BDA, Pid: 8812)
- MediaTek MT7662 (Vid: 0E8D, Pid: 7612) (Test sample:COMFAST CF-WU782AC V2)

## 6.5 Install the kernel-header package

```
sudo dpkg -i /opt/archives/linux-headers-*.deb
```

## 6.6 Build kernel-header deb package

Please refre to: https://github.com/friendlyarm/sd-fuse_rk3328/blob/master/test/test-build-kernel-header-deb.sh

## 6.7 Config status LEDs

First determine whether the system already exists the leds initialization service:

```
sudo systemctl status leds
```

If the leds service already exists, change the default behavior of the LEDs by editing the following file:

```
/etc/init.d/leds.sh
```

Since there is no leds service in the early firmware, you need to refer to the following guide to manually configure the LEDs. First, set the following kernel modules to be automatically loaded at boot:

```
modprobe ledtrig-netdev
echo ledtrig-netdev > /etc/modules-load.d/ledtrig-netdev.conf
```

Put the following into the autorun script to associate the status leds with the ethernet interface, and you can configure it to behave in other ways by referring to these content:

```
echo netdev > /sys/class/leds/wan_led/trigger
echo eth0 > /sys/class/leds/wan_led/device_name
echo 1 > /sys/class/leds/wan_led/link

echo netdev > /sys/class/leds/lan_led/trigger
echo eth1 > /sys/class/leds/lan_led/device_name
echo 1 > /sys/class/leds/lan_led/link
```

## 6.8 Delete Qt5 and related files

Execute the following commands：

```
su root
cd /
rm -rf usr/local/Trolltech/Qt-5.10.0-rk64one usr/local/Trolltech/Qt-5.10.0-rk64one-sdk usr/bin/setqt5env* usr/bin/qt5demo etc/qt5
rm -rf opt/{qt5-browser,Qt5_CinematicExperience,qt5-multi-screen-demo,qt5-nmapper,qt5-player,qt5-smarthome,QtE-Demo,qt5-qml-image-viewer,dual-camera}
```

# 7 Work with Debian Core

## 7.1 Account & Password

Regular Account:
  User Name: pi
  Password: pi

Root:
  the root user account is disabled by default, you may configure the root password through the 'sudo passwd root' command.

## 7.2 View IP address

Since the Debian Bullseye hostname is the hardware model by default, you can use the ping command to get the IP address:`ping NanoPi-R2S-Plus`
Debian Bullseye uses network-manager to manage the network, and the network ports are configured to automatically obtain IP addresses by DHCP (including devices with multiple network ports).

## 7.3 Connect to Debian via SSH

Run the following command`ssh pi@NanoPi-R2S-Plus`
The default password is: pi

## 7.4 Update Software Packages

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

## 7.5 Change time zone

### 7.5.1 Check the current time zone

```
timedatectl
```

### 7.5.2 List all available time zones

```
timedatectl list-timezones
```

### 7.5.3 Set the time zone (e.g. Shanghai)

```
sudo timedatectl set-timezone Asia/Shanghai
```

## 7.6 Change startup LOGO

Replace the following two files in the kernel source code directory and recompile the kernel：
kernel/logo.bmp
kernel/logo_kernel.bmp
Or use the script to operate, as shown below：

- Download scripts:

```
git clone https://github.com/friendlyarm/sd-fuse_rk3328.git -b kernel-4.19 --single-branch
cd sd-fuse_rk3328
```

- Compile kernel and repackage firmware

```
convert files/logo.jpg -type truecolor /tmp/logo.bmp
convert files/logo.jpg -type truecolor /tmp/logo_kernel.bmp
sudo LOGO=/tmp/logo.bmp KERNEL_LOGO=/tmp/logo_kernel.bmp ./build-kernel.sh debian-bookworm-core-arm64
sudo ./mk-sd-image.sh debian-bookworm-core-arm64
sudo ./mk-emmc-image.sh debian-bookworm-core-arm64
```

## 7.7 Soft Factory Reset

Execute the following command in a terminal:

```
sudo firstboot && sudo reboot
```

## 7.8 Install Docker on Debian

Please refer to: How to Install Docker on Debian

# 8 Buildroot Linux

Buildroot is a simple, efficient and easy-to-use tool to generate embedded Linux systems through cross-compilation. It contains a boot-loader, kernel, rootfs, various libraries and utilities(e.g. qt, gstreamer, busybox etc).

FriendlyELEC's Buildroot is based on Rockchip's version which is made with linux-sdk and maintained with git. FriendlyELEC's version is synced with Rockchip's version;

- Rockchip's Buildroot: https://github.com/rockchip-linux/buildroot
- Buildroot's official site: https://buildroot.org

For a more detailed description of the Buildroot system, please refer to: Buildroot

# 9 How to Compile

## 9.1 Setup Development Environment

### 9.1.1 Method 1: Using docker to cross-compile

Please refre to docker-cross-compiler-novnc (https://github.com/friendlyarm/docker-cross-compiler-novnc)

### 9.1.2 Method 2: Setup build environment on the host machine

#### 9.1.2.1 Install required packages

Install and run requirements ubuntu 20.04, install required packages using the following commands:

```
sudo apt-get -y update
sudo apt-get install -y sudo curl
sudo bash -c \
  "$(curl -fsSL https://raw.githubusercontent.com/friendlyarm/build-env-on-ubuntu-bionic/master/install.sh)"
```

The following cross-compilers will be installed:

| Version | Architecture | Compiler path | Purpose |
|---|---|---|---|
| 4.9.3 | armhf | /opt/FriendlyARM/toolchain/4.9.3 | Can be used to build 32-bit ARM applications |
| 6.4 | aarch64 | /opt/FriendlyARM/toolchain/6.4-aarch64 | Can be used to build kernel 4.4 |
| 11.3 | aarch64 | /opt/FriendlyARM/toolchain/11.3-aarch64 | Can be used to build kernel 4.19 or higher and U-Boot |

#### 9.1.2.2 Setting the compiler path

Based on the table in the previous section, select the appropriate version of the compiler and add the compiler's path to PATH. For example, if you want to use the 11.3 cross-compiler, edit ~/.bashrc using vi and add the following content to the end:

```
export PATH=/opt/FriendlyARM/toolchain/11.3-aarch64/bin:$PATH
export GCC_COLORS=auto
```

Run the ~/.bashrc script to make it effective in the current commandline. Note: there is a space after ".":

```
. ~/.bashrc
```

To verify if the installation was successful:

```
$ aarch64-linux-gcc -v
Using built-in specs.
COLLECT_GCC=aarch64-linux-gcc
COLLECT_LTO_WRAPPER=/opt/FriendlyARM/toolchain/11.3-aarch64/libexec/gcc/aarch64-cortexa53-linux-gnu/11.3.0/lto-wrapper
Target: aarch64-cortexa53-linux-gnu
Configured with: /home/cross/arm64/src/gcc/configure --build=x86_64-build_pc-linux-gnu --host=x86_64-build_pc-linux-gnu --target=aarch64-cortexa53-linux-gnu --prefix=/opt/FriendlyARM/t
Thread model: posix
Supported LTO compression algorithms: zlib
gcc version 11.3.0 (ctng-1.25.0-119g-FA)
```

## 9.2 Build Openwrt/Friendlywrt

### 9.2.1 Download Code

Two versions are available, please choose as required:

#### 9.2.1.1 FriendlyWrt 24.10

```
mkdir friendlywrt24-rk3328
cd friendlywrt24-rk3328
git clone https://github.com/friendlyarm/repo --depth 1 tools
tools/repo init -u https://github.com/friendlyarm/friendlywrt_manifests -b master-v24.10 \
        -m rk3328.xml --repo-url=https://github.com/friendlyarm/repo  --no-clone-bundle
tools/repo sync -c  --no-clone-bundle
```

#### 9.2.1.2 FriendlyWrt 23.05

```
mkdir friendlywrt23-rk3328
cd friendlywrt23-rk3328
git clone https://github.com/friendlyarm/repo --depth 1 tools
tools/repo init -u https://github.com/friendlyarm/friendlywrt_manifests -b master-v23.05 \
        -m rk3328.xml --repo-url=https://github.com/friendlyarm/repo  --no-clone-bundle
tools/repo sync -c  --no-clone-bundle
```

### 9.2.2 First compilation step

```
./build.sh rk3328.mk  # or rk3328-docker.mk
```

All the components (including u-boot, kernel, and friendlywrt) are compiled and the sd card image will be generated, then execute the following command to generate the image file for installing the system into the emmc:

```
./build.sh emmc-img
```

After making changes to the project, the sd card image needs to be repackaged by running the following command:

```
./build.sh sd-img
```

### 9.2.3 Secondary compilation steps

```
cd friendlywrt
make menuconfig
rm -rf ./tmp
make -j${nproc}
cd ../
./build.sh sd-img
./build.sh emmc-img
```

### 9.2.4 Build u-boot only

```
./build.sh uboot
```

### 9.2.5 Build kernel only

```
./build.sh kernel
```

### 9.2.6 Build friendlywrt only

```
./build.sh friendlywrt
```

Or go to the friendlywrt directory and follow the standard openwrt commands. If you get an error with the above command, try using the following command to compile in a single thread:

```
cd friendlywrt
make -j1 V=s
```

## 9.3 Build Buildroot

please refer to: Buildroot

## 9.4 Build Other Linux

### 9.4.1 Kernel and u-boot versions

| Operating System | Kernel Version | U-boot version | Cross-compiler | Partition type | Packaging Tool | Kernel branch | Ker |
|---|---|---|---|---|---|---|---|
| buildroot | linux v4.19.y | u-boot v2017.09 | 11.3-aarch64 | GPT (https://github.com/friendlyarm/sd-fuse_rk3328/blob/kernel-4.19/prebuilt/parameter.template) | sd-fuse (https://github.com/friendlyarm/sd-fuse_rk3328/tree/kernel-4.19) | nanopi4-v4.19.y (https://github.com/friendlyarm/kernel-rockchip/tree/nanopi4-v4.19.y) | nanop |
| friendlycore-focal-arm64 | | | | | | | |
| openmediavault-arm64 | linux v6.1.y | | | GPT (https://github.com/friendlyarm/sd-fuse_rk3328/blob/kernel-6.1.y/prebuilt/parameter-ext4.txt) | sd-fuse (https://github.com/friendlyarm/sd-fuse_rk3328/tree/kernel-6.1.y) | nanopi-r2-v6.1.y (https://github.com/friendlyarm/kernel-rockchip/tree/nanopi-r2-v6.1.y) | nanop r2_lin |
| ubuntu-noble-core-arm64 | | | | GPT (https://github.com/friendlyarm/sd-fuse_rk3328/blob/kernel-6.1.y/prebuilt/parameter.template) | | | |
| debian-bookworm-core-arm64 | | | | | | | |
| friendlywrt21 | | | | | | | nanop r2_lin +frien |
| friendlywrt21-docker | | | | | | | |
| friendlywrt23 | | | | | | | |
| friendlywrt23-docker | | | | | | | |

- Kernel git repo：https://github.com/friendlyarm/kernel-rockchip
- U-boot git repo：https://github.com/friendlyarm/uboot-rockchip
- The cross-compile toolchain is located in the path: /opt/FriendlyARM/toolchain/
- The SD-Fuse is a helper script to make bootable SD card image.

### 9.4.2 Build kernel linux-v4.19.y

This section applies to the following operating systems:

| | |
|---|---|
| buildroot | friendlycore-focal-arm64 |

Clone the repository to your local drive then build:

```
git clone https://github.com/friendlyarm/kernel-rockchip --single-branch --depth 1 -b nanopi4-v4.19.y kernel-rockchip
cd kernel-rockchip
export PATH=/opt/FriendlyARM/toolchain/11.3-aarch64/bin/:$PATH
touch .scmversion
# Configuring the Kernel
# Load default configuration
make ARCH=arm64 CROSS_COMPILE=aarch64-linux- nanopi4_linux_defconfig
# Optionally, load configuration for FriendlyWrt
# make ARCH=arm64 CROSS_COMPILE=aarch64-linux- nanopi4_linux_defconfig friendlywrt.config
# Optionally, if you want to change the default kernel config
# make ARCH=arm64 CROSS_COMPILE=aarch64-linux- menuconfig
# Start building kernel
make ARCH=arm64 CROSS_COMPILE=aarch64-linux- nanopi4-images -j$(nproc)
# Start building kernel modules
mkdir -p out-modules
make ARCH=arm64 CROSS_COMPILE=aarch64-linux- INSTALL_MOD_PATH="$PWD/out-modules" modules -j$(nproc)
make ARCH=arm64 CROSS_COMPILE=aarch64-linux- INSTALL_MOD_PATH="$PWD/out-modules" modules_install
KERNEL_VER=$(make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 kernelrelease)
rm -rf $PWD/out-modules/lib/modules/${KERNEL_VER}/kernel/drivers/gpu/arm/mali400/
[ ! -f "$PWD/out-modules/lib/modules/${KERNEL_VER}/modules.dep" ] && depmod -b $PWD/out-modules -E Module.symvers -F System.map -w ${KERNEL_VER}
(cd $PWD/out-modules && find . -name \*.ko | xargs aarch64-linux-strip --strip-unneeded)
```

After the compilation, the following files will be generated:

| | | |
|---|---|---|
| kernel.img | resource.img | The kernel modules are located in the out-modules directory |

Run your build:
Please refre to #Running the build

### 9.4.3 Build kernel linux-v6.1.y

This section applies to the following operating systems:

| | | | | | | |
|---|---|---|---|---|---|---|
| friendlywrt21 | friendlywrt21-docker | friendlywrt23 | friendlywrt23-docker | ubuntu-noble-core-arm64 | openmediavault-arm64 | debian-bookworm-core-arm64 |

Clone the repository to your local drive then build:

```
git clone https://github.com/friendlyarm/kernel-rockchip --single-branch --depth 1 -b nanopi-r2-v6.1.y kernel-rockchip
cd kernel-rockchip
export PATH=/opt/FriendlyARM/toolchain/11.3-aarch64/bin/:$PATH
touch .scmversion
# Configuring the Kernel
# Load default configuration
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 nanopi-r2_linux_defconfig
# Optionally, load configuration for FriendlyWrt
# make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 nanopi-r2_linux_defconfig friendlywrt.config
# Optionally, if you want to change the default kernel config
# make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 menuconfig
# Start building kernel
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 -j$(nproc)
```

```
# Start building kernel modules
mkdir -p out-modules && rm -rf out-modules/*
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 INSTALL_MOD_PATH="$PWD/out-modules" modules -j$(nproc)
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 INSTALL_MOD_PATH="$PWD/out-modules" modules_install
KERNEL_VER=$(make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 kernelrelease)
[ ! -f "$PWD/out-modules/lib/modules/${KERNEL_VER}/modules.dep" ] && depmod -b $PWD/out-modules -E Module.symvers -F System.map -w ${KERNEL_VER}
(cd $PWD/out-modules && find . -name \*.ko | xargs aarch64-linux-strip --strip-unneeded)
```

Pack the kernel.img and resource.img:

```
wget https://raw.githubusercontent.com/friendlyarm/sd-fuse_rk3328/kernel-6.1.y/tools/mkkrnlimg && chmod 755 mkkrnlimg
wget https://raw.githubusercontent.com/friendlyarm/sd-fuse_rk3328/kernel-6.1.y/tools/resource_tool && chmod 755 resource_tool
wget https://raw.githubusercontent.com/friendlyarm/sd-fuse_rk3328/kernel-6.1.y/prebuilt/boot/logo.bmp
wget https://raw.githubusercontent.com/friendlyarm/sd-fuse_rk3328/kernel-6.1.y/prebuilt/boot/logo_kernel.bmp
./mkkrnlimg arch/arm64/boot/Image kernel.img
./resource_tool --dtbname arch/arm64/boot/dts/rockchip/rk3328-nanopi*-rev*.dtb logo.bmp logo_kernel.bmp
```

After the compilation, the following files will be generated:

| | | |
|---|---|---|
| kernel.img | resource.img | The kernel modules are located in the out-modules directory |

Run your build:
Please refre to #Running the build

### 9.4.4 Build u-boot v2017.09

This section applies to the following operating systems:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| buildroot | friendlycore-focal-arm64 | friendlywrt21 | friendlywrt21-docker | friendlywrt23 | friendlywrt23-docker | ubuntu-noble-core-arm64 | openmediavault-arm64 | debian-bookworm-core-arm64 |

Clone the repository to your local drive then build:

```
git clone https://github.com/friendlyarm/rkbin --single-branch --depth 1 -b friendlyelec
git clone https://github.com/friendlyarm/uboot-rockchip --single-branch --depth 1 -b nanopi4-v2017.09
export PATH=/opt/FriendlyARM/toolchain/11.3-aarch64/bin/:$PATH
cd uboot-rockchip/
./make.sh nanopi_r2
```

After the compilation, the following files will be generated:

| | | |
|---|---|---|
| uboot.img | trust.img | rk3328_loader_v1.16.250.bin (aka MiniLoaderAll.bin) |

Run your build:
Please refre to #Running the build

### 9.4.5 Running the build

#### 9.4.5.1 Install to target board

This section applies to the following operating systems:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| buildroot | friendlycore-focal-arm64 | friendlywrt21 | friendlywrt21-docker | friendlywrt23 | friendlywrt23-docker | ubuntu-noble-core-arm64 | openmediavault-arm64 | debian-bookworm-core-arm64 |

RK3328 uses GPT partitions by default, you can use the dd command, but be careful to choose the right output device:

- The SD/TF Card device node: /dev/mmcblk0
- The eMMC device node: /dev/mmcblk2

The following is an example of how to update the kernel to eMMC:
Use the 'parted' command to view the partition layout:

```
parted /dev/mmcblk2 print
```

Sample outputs:

```
Model: MMC BJTD4R (sd/mmc)
Disk /dev/mmcblk2: 31.3GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name      Flags
 1      8389kB  12.6MB  4194kB               uboot
 2      12.6MB  16.8MB  4194kB               trust
 3      16.8MB  21.0MB  4194kB               misc
 4      21.0MB  25.2MB  4194kB               dtbo
 5      25.2MB  41.9MB  16.8MB               resource
 6      41.9MB  83.9MB  41.9MB               kernel
 7      83.9MB  134MB   50.3MB               boot
 8      134MB   2500MB  2366MB  ext4         rootfs
 9      2500MB  31.3GB  28.8GB  ext4         userdata
```

as shown above, the resource partition is located at 5 and the kernel partition is located at 6. Use the dd command to write the resource.img and kernel.img files to these partitions, the commands are as follows:

```
dd if=resource.img of=/dev/mmcblk2p5 bs=1M
dd if=kernel.img of=/dev/mmcblk2p6 bs=1M
```

If you want to update u-boot:

```
dd if=uboot.img of=/dev/mmcblk2p1 bs=1M
```

To update new driver modules, copy the newly compiled driver modules to the appropriate directory under /lib/modules.

##### 9.4.5.2 Packaging and creating an SD image

To create a new OS image file, you need to use the "sd-fuse" packaging tool.

"sd-fuse" is a collection of scripts that can be used to create bootable SD card images for FriendlyElec boards. Its main features include:

- Creation of root filesystem images from a directory
- Building of bootable SD card images
- Simple compilation of kernel, U-Boot, and third-party drivers

Please click on the following link to find out more:

| Kernel version | Packaging Tool |
|---|---|
| linux v4.19.y | sd-fuse_rk3328/kernel-4.19 (https://github.com/friendlyarm/sd-fuse_rk3328/tree/kernel-4.19) |
| linux v6.1.y | sd-fuse_rk3328/kernel-6.1.y (https://github.com/friendlyarm/sd-fuse_rk3328/tree/kernel-6.1.y) |

##### 9.4.5.3 USB flashing

###### 9.4.5.3.1 Linux

Reboot the board and enter loader mode with the following command:

```
sudo reboot loader
```

To flash U-Boot and kernel using the "upgrade_tool_v2.17_for_linux" tool, please use the following command:

```
sudo upgrade_tool di -k kernel.img
sudo upgrade_tool di -re resource.img
sudo upgrade_tool di -u uboot.img
sudo upgrade_tool RD
```

Note: "upgrade_tool" is a command-line tool provided by Rockchip for Linux operating systems (Linux_Upgrade_Tool).

## 9.5 Build the code using scripts

### 9.5.1 Download scripts and image files

```
git clone https://github.com/friendlyarm/sd-fuse_rk3328.git -b kernel-4.19
cd sd-fuse_rk3328
wget http://112.124.9.243/dvdfiles/rk3328/images-for-eflasher/friendlycore-focal-arm64-images.tgz
tar xvzf friendlycore-focal-arm64-images.tgz
```

### 9.5.2 Compile the kernel

Download the kernel source code and compile it. the relevant image files in the friendlycore-focal-arm64 directory will be automatically updated, including the kernel modules in the file system:

```
git clone https://github.com/friendlyarm/kernel-rockchip --depth 1 -b nanopi4-v4.19.y kernel-rk3328
KERNEL_SRC=$PWD/kernel-rk3328 ./build-kernel.sh friendlycore-focal-arm64
```

### 9.5.3 Compile the kernel headers

```
git clone https://github.com/friendlyarm/kernel-rockchip --depth 1 -b nanopi4-v4.19.y kernel-rk3328
MK_HEADERS_DEB=1 BUILD_THIRD_PARTY_DRIVER=0 KERNEL_SRC=$PWD/kernel-rk3328 ./build-kernel.sh friendlycore-focal-arm64
```

### 9.5.4 Compile the uboot

Download the uboot source code and compile it. the relevant image files in the friendlycore-focal-arm64 directory will be automatically updated:

```
git clone https://github.com/friendlyarm/uboot-rockchip --depth 1 -b nanopi4-v2017.09
UBOOT_SRC=$PWD/uboot-rockchip ./build-uboot.sh friendlycore-focal-arm64
```

### 9.5.5 Generate new image

Repackage the image file in the friendlycore-focal-arm64 directory into sd card image:

```
./mk-sd-image.sh friendlycore-focal-arm64
```

After the command is completed, the image is in the out directory, you can use the dd command to make the SD boot card, for example:

```
dd if=out/rk3328-sd-friendlycore-focal-4.19-arm64-YYYYMMDD.img of=/dev/sdX bs=1M
```

# 10 Backup rootfs and create custom SD image (to burn your application into other boards)

## 10.1 Backup rootfs

Run the following commands on your target board. These commands will back up the entire root partition:

```
sudo passwd root
su root
cd /
tar --warning=no-file-changed -cvpzf /rootfs.tar.gz \
    --exclude=/rootfs.tar.gz --exclude=/var/lib/docker/runtimes \
    --exclude=/etc/firstuser --exclude=/etc/friendlyelec-release \
    --exclude=/usr/local/first_boot_flag --one-file-system /
```

Note: if there is a mounted directory on the system, an error message will appear at the end, which can be ignored.

## 10.2 Making a bootable SD card from a root filesystem

Run the following script on your Linux PC host, we'll only mention "ubuntu-noble-core-arm64 os" for brevity, but you can apply the same process for every linux OS.

```
su root
git clone https://github.com/friendlyarm/sd-fuse_rk3328 --single-branch -b kernel-6.1.y
cd sd-fuse_rk3328
tar xvzf /path/to/netdrive/03_Partition\ image\ files/ubuntu-noble-core-arm64-images.tgz
tar xvzf /path/to/netdrive/03_Partition\ image\ files/emmc-eflasher-images.tgz
scp pi@BOARDIP:/rootfs.tar.gz /rootfs.tar.gz
mkdir rootfs
tar xvzfp rootfs.tar.gz -C rootfs --numeric-owner --same-owner
./build-rootfs-img.sh rootfs ubuntu-noble-core-arm64
./mk-sd-image.sh ubuntu-noble-core-arm64
./mk-emmc-image.sh ubuntu-noble-core-arm64 autostart=yes
```

# 11 Common Linux-based operating system operations

## 11.1 Using ADB on Linux Systems

### 11.1.1 Enabling ADB in Buildroot System

Enable on Startup

```
mv /etc/init.d/K50usbdevice.sh /etc/init.d/S50usbdevice.sh
reboot
```

Enable Temporarily

```
usbdevice-wrapper start
```

### 11.1.2 Enabling ADB in Ubuntu and Debian Systems

Enable on Startup

```
sudo systemctl enable usbdevice
sudo reboot
```

Enable Temporarily

```
usbdevice-wrapper start
```

### 11.1.3 How to Connect

When using ADB, the port connected to the computer is the same as the USB flashing port.

## 11.2 Install Kernel Headers

To install the .deb file located in the /opt/archives directory:

```
sudo dpkg -i /opt/archives/linux-headers-*.deb
```

To download and update the kernel header files online:

```
wget http://112.124.9.243/archives/rk3328/linux-headers-$(uname -r)-latest.deb
sudo dpkg -i ./linux-headers-latest.deb
```

You can visit http://112.124.9.243/archives/rk3328 to see which kernel deb packages are available.

## 11.3 Setting Kernel Boot Parameters (eMMC/UFS Only)

Flash the firmware file XXXX-eflasher-multiple-os-YYYYMMDD-30g.img.gz to a TF card, then insert the TF card into your computer. Windows will usually recognize the TF card partition automatically (formatted as exFAT). For Linux or Mac users, manually mount the first partition of the TF card. Assuming the TF card's device name is /dev/sdX, mount /dev/sdX1.

Edit the info.conf configuration file in the OS directory on the TF card, adding the bootargs-ext parameter. For example:

```
bootargs-ext=rockchipdrm.fb_max_sz=2048
```

To remove a specified parameter, set it to empty. For example, to remove the userdata parameter:

```
bootargs-ext=userdata=
```

After editing, use this TF card to flash the system to eMMC/UFS Only.

To set kernel boot parameters during the creation of a mass production card, refer to the following script (example for RK3588): https://github.com/friendlyarm/sd-fuse_rk3588/blob/kernel-6.1.y/test/test-custom-bootargs.sh

# 12 Unbricking Method

If the ROM is not installed correctly, causing the development board to become bricked, and you might not have the opportunity to reinstall the ROM via an SD card, you need to enter Maskrom mode to unbrick it by erasing the storage device.

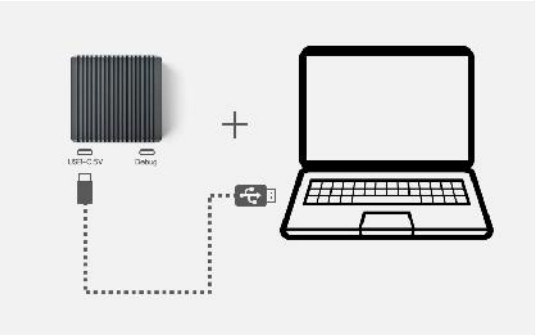
## 12.1 Windows Users

### 12.1.1 Download Required Files

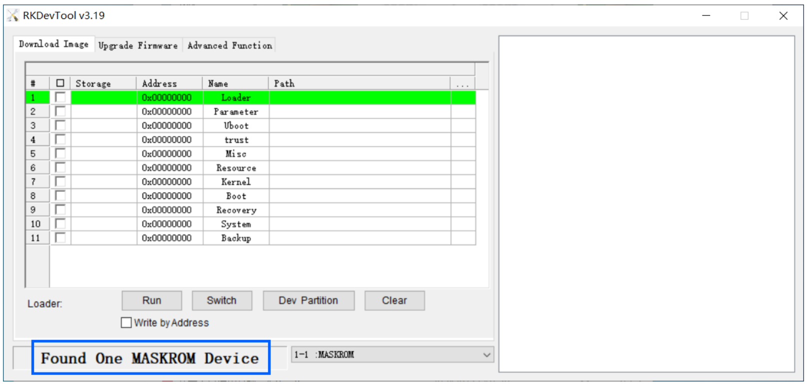
- **Get the necessary tools**: Visit here (https://dl.friendlyelec.com/NanoPi-R2S-Plus), find **RKDevTool_v3.19_for_window.zip** and **DriverAssitant_v5.12.zip** in the **05_Tools** directory, and download them to your local machine.
- **Install Rockchip USB driver and RKDevTool**: Extract **DriverAssitant_v5.12.zip** to install the Rockchip USB driver, and extract **RKDevTool_v3.19_for_window.zip** to obtain the Rockchip flashing tool **RKDevTool**.
- **Get the loader**: Visit here (http://112.124.9.243/dvdfiles/), enter the tools directory corresponding to your CPU model, and download **MiniLoaderAll.bin**.

### 12.1.2 Enter Maskrom Mode to Erase the Storage Device

- Remove the SD card, USB device, and other peripherals from the development board
- Start **RKDevTool** on your computer.
- Press and hold the "Mask" key, Use a USB C-to-A cable, connect NanoPi-R2S-Plus to a PC， After the status LED has been on for at least 3 seconds, release the Mask key



- You will see **Found One MASKROM Device** displayed at the bottom of the **RKDevTool** interface, as shown below:

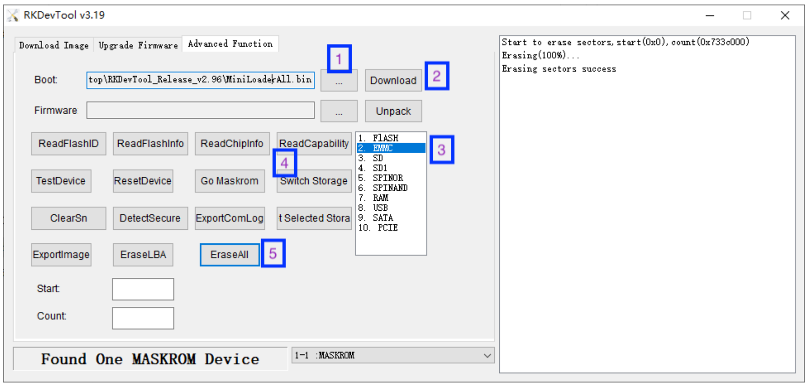


| # | □ | Storage | Address | Name | Path | ... |
|---|---|---|---|---|---|---|
| 1 | □ |  | 0x00000000 | Loader |  |  |
| 2 | □ |  | 0x00000000 | Parameter |  |  |
| 3 | □ |  | 0x00000000 | Uboot |  |  |
| 4 | □ |  | 0x00000000 | trust |  |  |
| 5 | □ |  | 0x00000000 | Misc |  |  |
| 6 | □ |  | 0x00000000 | Resource |  |  |
| 7 | □ |  | 0x00000000 | Kernel |  |  |
| 8 | □ |  | 0x00000000 | Boot |  |  |
| 9 | □ |  | 0x00000000 | Recovery |  |  |
| 10 | □ |  | 0x00000000 | System |  |  |
| 11 | □ |  | 0x00000000 | Backup |  |  |

Found One MASKROM Device

1-1 :MASKROM

- Click the **Advanced Function** tab in the **RKDevTool** interface.
- In the **Boot** text box, select **MiniLoaderAll.bin**, then click the **Download** button.
- Select **EMMC**, click **Switch Storage**, then click the **EraseAll** button to erase the eMMC.



Boot: top\RKDevTool_Release_v2.96\MiniLoaderAll.bin

1. FlASH
2. EMMC
3. SD
4. SD1
5. SPINOR
6. SPINAND
7. RAM
8. USB
9. SATA
10. PCIE

```
Start to erase sectors,start(0x0),count(0x733c000)
Erasing(100%)...
Erasing sectors success
```

Found One MASKROM Device

1-1 :MASKROM

- At this point, NanoPi-R2S-Plus is restored to its initial state and can be normally booted via SD card or eMMC.

## 12.2 Linux Users

### 12.2.1 Download the Required Files

- **Get the necessary tools**: Visit here (https://dl.friendlyelec.com/NanoPi-R2S-Plus) and find **upgrade_tool_v2.30_for_linux.tgz** in the **05_Tools** directory and download it locally.
- **Get the loader**: Visit here (http://112.124.9.243/dvdfiles/), enter the tools directory corresponding to your CPU model, and download **MiniLoaderAll.bin**.

### 12.2.2 Installation for upgrade_tool

Using the following commands:

```
tar xzf upgrade_tool_v2.30_for_linux.tgz
cd upgrade_tool_v2.30_for_linux
sudo cp upgrade_tool /usr/local/sbin/
sudo chmod 755 /usr/local/sbin/upgrade_tool
```

### 12.2.3 Enter Maskrom Mode to Erase the Storage Device

- Connect NanoPi-R2S-Plus to the computer using a USB data cable.
- Disconnect the power from NanoPi-R2S-Plus, hold down the **MASK** button, connect the power, and release the button after 4 seconds.
- Check the connection with the following command:

```
upgrade_tool LD
```

A result similar to "DevNo=1 Vid=0x2207,Pid=0x350b,LocationID=13 Mode=Maskrom SerialNo=" indicates that the device has been detected.

- Erase the eMMC with the following command:

```
upgrade_tool EF MiniLoaderAll.bin
```

- At this point, NanoPi-R2S-Plus has been restored to its initial state and can boot the system normally via SD card or eMMC.

## 12.3 Mac Users

Our tests found that upgrade_tool_v2.25 does not work properly on macOS. Therefore, we recommend using Windows or Linux unless an updated version of upgrade_tool becomes available.

# 13 More OS Support

## 13.1 DietPi

DietPi is a highly optimised & minimal Debian-based Linux distribution. DietPi is extremely lightweight at its core, and also extremely easy to install and use.
Setting up a single board computer (SBC) or even a computer, for both regular or server use, takes time and skill. DietPi provides an easy way to install and run favourite software you choose.
For more information, please visit this link https://dietpi.com/docs/.

DietPi supports many of the NanoPi board series, you may download the image file from here:

- https://dietpi.com/docs/hardware/#nanopi-series-friendlyarm

# 14 Resources

## 14.1 Datasheets and Schematics

- Schematics
  - https://wiki.friendlyelec.com/wiki/images/d/da/NanoPi_R2S_Plus_2309_SCH.PDF

- PCB Dimensional Diagram
  - https://wiki.friendlyelec.com/wiki/images/0/0c/Nanopi_r2s_plus_2309_dxf.zip
- Datasheet
  - RK3328 Datasheet Rockchip_RK3328_Datasheet.pdf (http://wiki.friendlyelec.com/wiki/images/d/d7/Rockchip_RK3328_Datasheet_V1.1-20170309.pdf)

# 15 Update Logs

Initial Release

Retrieved from "https://wiki.friendlyelec.com/wiki/index.php?title=NanoPi_R2S_Plus&oldid=26526"

---

- This page was last modified on 29 October 2024, at 11:03.