NanoPi NEO Air

From FriendlyARM WiKi

查看中文

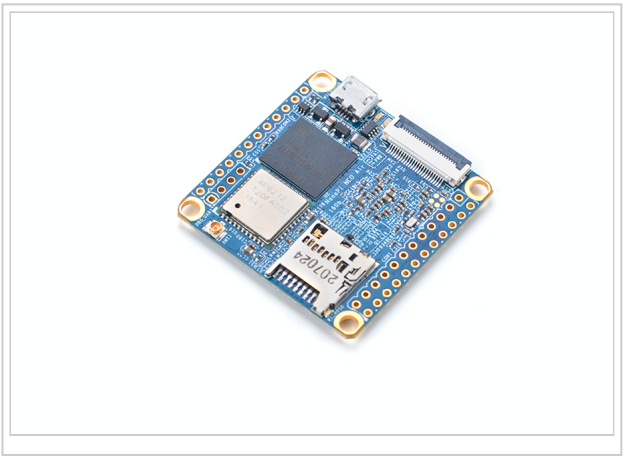
Contents

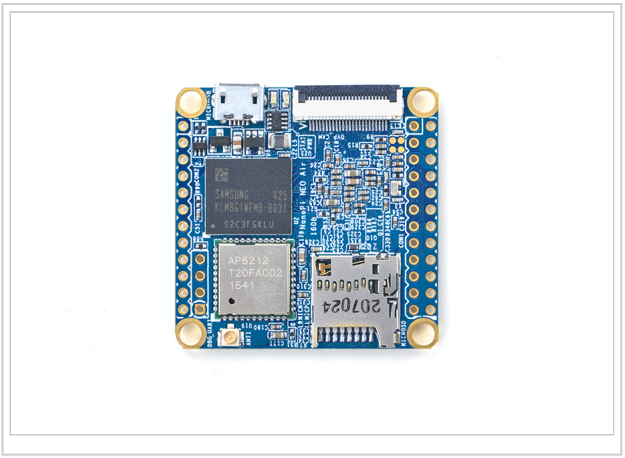
Introduction

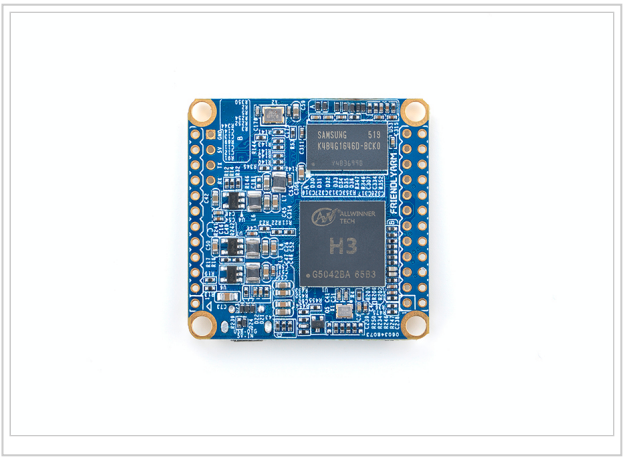
Specifications

- CPU: Allwinner H3, Quad-core Cortex-A7 Up to 1.2GHz
- RAM: 512MB DDR3 RAM
- Storage: 8GB eMMC

- WiFi: 802.11b/g/n
- Bluetooth: 4.0 dual mode
- MicroUSB: OTG and power input
- MicroSD Slot x 1
- Debug Serial Port: 4Pin,2.54mm pitch pin header
- GPIO1: 2.54mm spacing 24pin,It includes UART,SPI,I2C,GPIO
- GPIO2: 2.54mm spacing 12pin,It includes USBx2,IR,SPDIF,I2S
- PCB Size: 40 x 40mm
- PCB layer: 6
- Power Supply: DC 5V/2A
- OS/Software: u-boot, UbuntuCore
- Weight: 7.5g(WITHOUT Pin-headers); 9.7g(WITH Pin-headers)



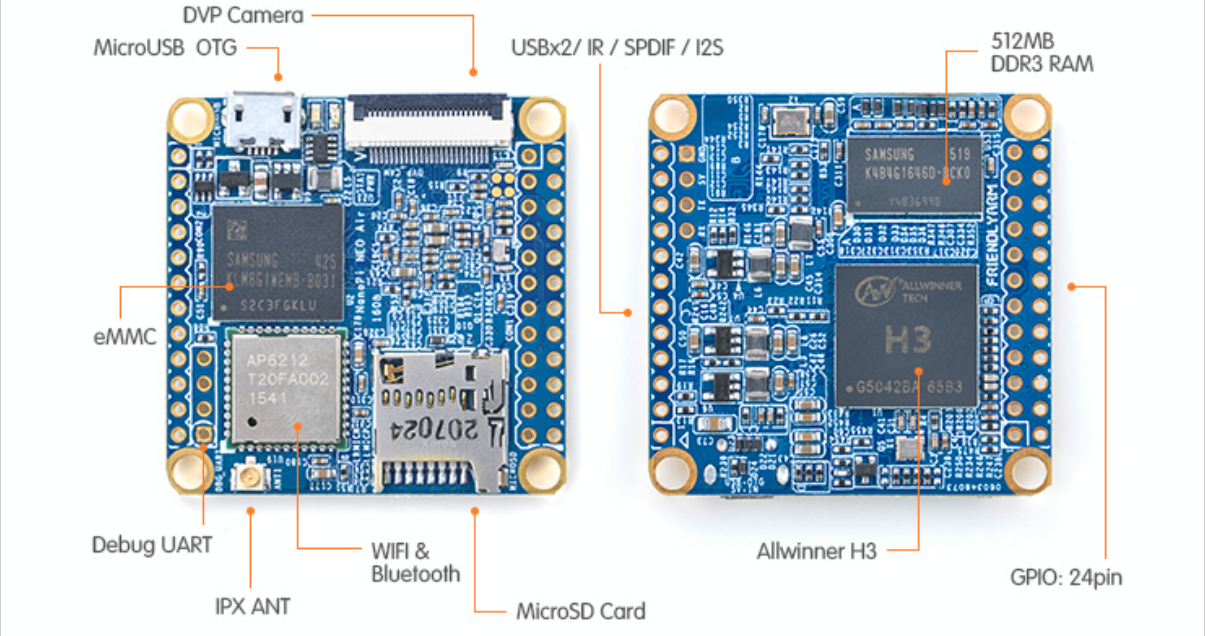




Diagram, Layout and Dimension

Layout

- **GPIO Pin Description**



NanoPi NEO-AIR Layout

Pin#	Name	Linux gpio	Pin#	Name	Linux gpio
1	SYS_3.3V		2	VDD_5V	
3	I2C0_SDA		4	VDD_5V	
5	I2C0_SCL		6	GND	
7	GPIOG11	203	8	UART1_TX/GPIOG6	198
9	GND		10	UART1_RX/GPIOG7	199
11	UART2_TX/GPIOA0	0	12	PWM1/GPIOA6	6
13	UART2_RTS/GPIOA2	2	14	GND	
15	UART2_CTS/GPIOA3	3	16	UART1_RTS/GPIOG8	200
17	SYS_3.3V		18	UART1_CTS/GPIOG9	201
19	SPI0_MOSI/GPIOC0	64	20	GND	
21	SPI0_MISO/GPIOC1	65	22	UART2_RX/GPIOA1	1
23	SPI0_CLK/GPIOC2	93	24	SPI0_CS/GPIOC3	67

- **USB/Audio/IR Pin Description**

Pin#	Name	Description
1	VDD_5V	5V Power Out
2	USB-DP1	USB1 DP Signal
3	USB-DM1	USB1 DM Signal
4	USB-DP2	USB2 DP Signal
5	USB-DM2	USB2 DM Signal
6	GPIOL11/IR-RX	GPIOL11 or IR Receive
7	SPDIF-OUT/GPIOA17	GPIOA17 or SPDIF-OUT
8	PCM0_SYNC/I2S0_LRC	I2S/PCM Sample Rate Clock/Sync
9	PCM0_CLK/I2S0_BCK	I2S/PCM Sample Rate Clock
10	PCM0_DOUT/I2S0_SDOUT	I2S/PCM Serial Bata Output
11	PCM0_DIN/I2S0_SDIN	I2S/PCM Serial Data Input
12	GND	0V

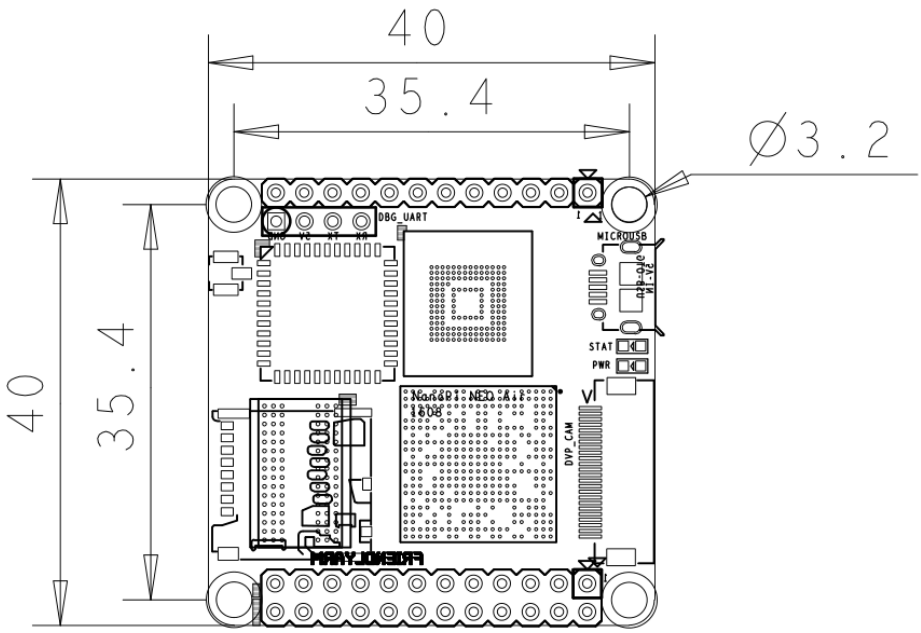
- **Debug Port（UART0）**

Pin#	Name
1	GND
2	VDD_5V
3	UART_TXD0
4	UART_RXD0

Note:

1. SYS_3.3V: 3.3V power output
2. VVDD_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 ~ 5.6V
3. All pins are 3.3V, output current is 5mA
4. For more details refer to the document:NanoPi-NEO-Air-1608-Schematic.pdf (http://wiki.friendlyarm.com/wiki/images/9/98/NanoPi-NEO-Air-1608-Schematic.pdf)

Dimensional Diagram



Get Started

Essentials You Need

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

- NanoPi NEO AIR
- microSD Card/TFCard: Class 10 or Above, minimum 8GB SDHC
- microUSB power. A 5V/2A power is a must
- A Host computer running Ubuntu 14.04 64 bit system

TF Cards We Tested

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

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

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

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

Make an Installation TF Card

Boot NanoPi NEO AIR from TF Card

- Get the following files from here [1] (https://www.mediafire.com/folder/sr5d0qpz774cs/NanoPi-NEO_Air) to download image files and the flashing utility:

Image Files	
nanopi-air-core-qte-sd4g.img.zip	Ubuntu core with Qt Embedded
nanopi-air-eflasher-sd8g.img.zip	eflasher image which can be used to Flash image files to eMMC
Flash Utility	
win32diskimager.rar	Windows utility. Under Linux users can use "dd"

- Uncompress these files. Insert a TF card(at least 4G) into a Windows PC and run the win32diskimager utility as administrator. On the utility's main window select your TF card's drive, the wanted image file and click on "write" to start flashing the SD card till it is done.
- Insert this card into your AIR's MicroSD card slot and power on (with a 5V/2A power source). If the blue LED is blinking this indicates your AIR has successfully booted.

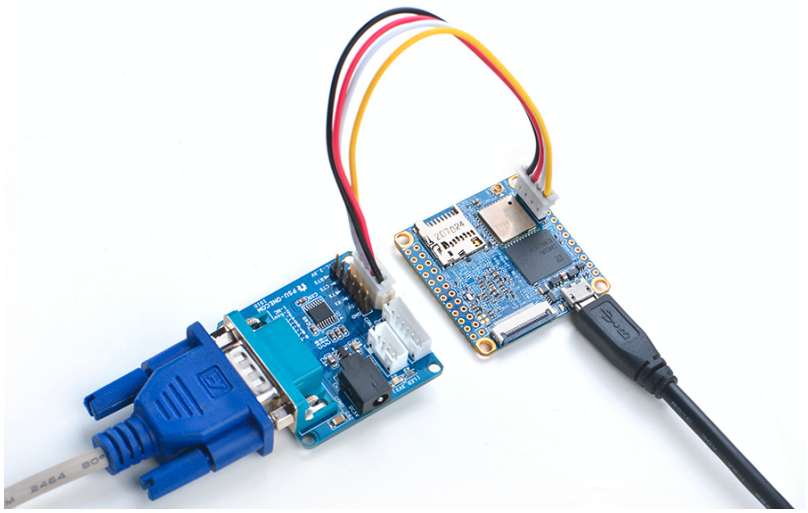
Flash image to eMMC with eflasher

- The eflasher is a utility FriendlyARM developed based on UbuntuCore. It can be used to flash OS image files to eMMC.
- Extract the nanopi-air-eflasher-sd8g.img.zip package and win32diskimager.rar. Insert a TF card(at least 8G) into a Windows PC and run the win32diskimager utility as administrator. On the utility's main window select your TF card's drive, the wanted image file and click on "write" to start flashing the TF card.
- Insert this card into your AIR and power on (with a 5V/2A power source) the board. If the blue LED is blinking it indicates your eflasher has started installation.

eflasher Functions

System Login via Serial Port

- If you want to do kernel development you need to use a serial communication board, ie a PSU-ONECOM board, which will allow you to operate the board via a serial terminal.Here is a setup where we connect a NanoPi NEO AIR to a PC via a serial cable you will see system messages output to the PC's serial terminal:



- The password for both "root" and "fa" is "fa"
- Update software packages

```
apt-get update
```

WiFi Connection

Insert a TF card with eflasher to a PC running Ubuntu and make the following changes in the etc/wpa_supplicant/wpa_supplicant.conf file under the rootfs section:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
	ssid="YOUR-WIFI-ESSID"
	psk="YOUR-WIFI-PASSWORD"
}
```

Note: the "YOUR-WIFI-ESSID" and "YOUR-WIFI-PASSWORD" need to be replaced with your actual ESSID and password.
Save, exit and insert the TF card to your AIR. After power on your AIR and the blue LED is blinking for one minute you will be able to check AIR's IP address.

SSH Login

The NanoPi NEO AIR has no display interface. If you don't have a serial communication board to connect your AIR to a PC you can log in your AIR via SSH. In our example the AIR's IP address was 192.168.1.230 and we logged in AIR by using the following command:

```
ssh root@192.168.1.230
```

The password is fa。

Flash Image to eMMC

The eflasher utility has a Ubuntu-Core system. After you run eflasher you can flash Ubuntu-core to eMMC by using the following command:

```
flash_eMMC.sh -d /mnt/sdcard/Ubuntu-Core-qte/
```

This command flashes the Ubuntu-Core image to eMMC, copies the wifi configuration file to eMMC and extends eMMC's file system. After it is done you will see the following message:

```
INFO: flash system to eMMC success
```

Power off the board, take out the TF card, power on the board again and it will boot from eMMC. If a TF card is inserted the board will boot from the TF card first. If booting from TF card fails it will boot from eMMC instead.

Work with Ubuntu-Core

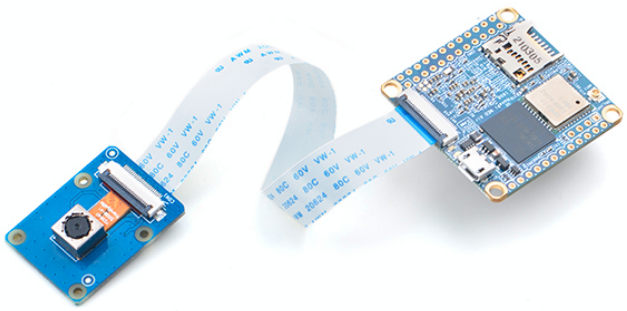
Check CPU's Working Temperature

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

```
cpu_freq
```

Connect NanoPi NEO AIR to DVP Camera(CAM500B)

The CAM500B camera module is a 5M-pixel camera with DVP interface. For more tech details about it you can refer to Matrix - CAM500B.



Boot Debian, connect your NEO AIR to a network, log into the board as root and run "mjpg-streamer":

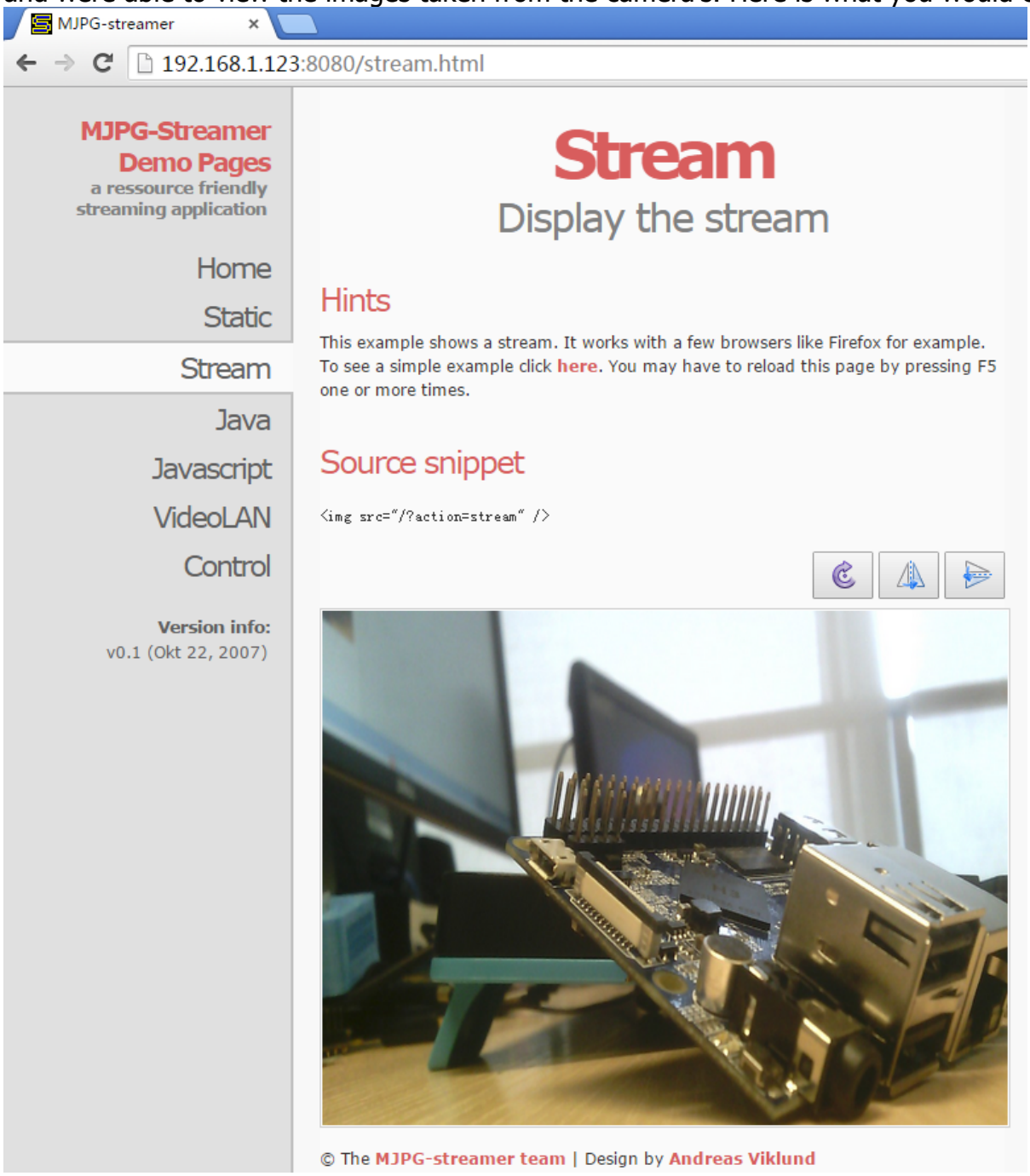
```
cd /root/mjpg-streamer
make
```

```
./start.sh
```

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

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

In our case the NEO AIR'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 soft-encodes data with libjpeg and you can hard-encode its data with ffmpeg which will greatly increase CPU's efficiency and speed up data 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.

Make Your Own OS(Compile BSP)

Visit download link (http://pan.baidu.com/s/1miMwKoK) and go to the sources directory and download nanopi-H3-bsp.
Use the 7-zip utility to uncompress it and a lihee directory and an Android directory will be generated.
Or you can get it from our github:

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

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

Compile lichee Source Code

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

```
sudo apt-get install gawk git gnupg flex bison gperf build-essential \
zip curl libc6-dev libncurses5-dev:i386 x11proto-core-dev \
libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 \
libgl1-mesa-dev g++-multilib mingw32 tofrodos \
python-markdown libxml2-utils xsltproc zlib1g-dev:i386 u-boot-tools
```

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

```
cd lichee
./build.sh -p sun8iw7p1 -b nanopi-h3
```

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

Package System Modules

```
./gen_script.sh nanopi-air
```

This command copies the generated executables including u-boot and Linux kernel and configuration files to the "lichee/tools/pack/out/" directory and generates a script.bin file.
The script.bin file is designed by Allwinner for its CPUs. For more details refer to script.bin (http://linux-sunxi.org/Script.bin)。

You can use the following commands to update the u-boot on your TF card:

```
./fuse_uboot.sh /dev/sdx
```

You need to replace /dev/sdx with the real device name in your system.
The uImage and kernel modules are under linux-3.4/output. Copy the uImage to your TF card's boot section and your TF card will boot your new image file.

Compile U-boot

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

```
./build.sh -p sun8iw7p1 -b nanopi-h3 -m uboot
```

After a u-boot executable is generated some extra patches need to be patched to it. Run "./build.sh pack" to patch this executable.
If you want to manually patch the executable refer to H3_Manual_build_howto (http://linux-sunxi.org/H3_Manual_build_howto) and run the following commands to update the u-boot in the TF card:

```
./fuse_uboot.sh /dev/sdx
```

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

Compile Linux Kernel

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

```
./build.sh -p sun8iw7p1 -b nanopi-h3 -m kernel
```

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

Clean Source Code

```
./build.sh -p sun8iw7p1_linux -b nanopi-h3 -m clean
```

3D Printing Files

NanoPi NEO-AIR 3D printed housing
3D Printing Files (http://www.thingiverse.com/thing:1698298)

Resources

- Schematics

    - NanoPi-NEO-Air-1608-Schematic.pdf (http://wiki.friendlyarm.com/wiki/images/9/98/NanoPi-NEO-Air-1608-Schematic.pdf)
- Dimensional Diagram
    - pcb file in dxf format (http://wiki.friendlyarm.com/wiki/images/7/78/NanoPi-NEO-Air-1608-dimensions%28dxf%29.zip)
- H3 datasheet Allwinner_H3_Datasheet_V1.2.pdf (http://wiki.friendlyarm.com/wiki/images/4/4b/Allwinner_H3_Datasheet_V1.2.pdf)

Update Log

Sep-28-2016

- Released English Version

Retrieved from "http://wiki.friendlyarm.com/wiki/index.php?title=NanoPi_NEO_Air&oldid=7178"

Category: Pages with broken file links

- This page was last modified on 29 September 2016, at 09:32.