



NanoPi S2

[查看中文](#)

Contents

- [1 Introduction](#)
- [2 Hardware Spec](#)
- [3 Diagram, Layout and Dimension](#)
 - [3.1 Layout](#)
 - [3.2 Board Dimension](#)
- [4 Get Started](#)
 - [4.1 Essentials You Need](#)
 - [4.2 TF Card We Tested](#)
 - [4.3 Make an Installation MicroSD Card](#)
 - [4.3.1 Under Windows](#)
 - [4.3.2 Under Linux Desktop](#)
 - [4.3.3 Flash image to eMMC with eflasher](#)
 - [4.3.4 Extend NanoPi S2's TF Card Section](#)
 - [4.3.5 LCD/HDMI Resolution](#)
 - [4.4 Update Image Files in SD Card From PC Host](#)
 - [4.5 Run Android or Debian](#)
 - [4.6 Login to Debian via VNC & SSH](#)
- [5 Working with Debian](#)
 - [5.1 Wireless Connection](#)
 - [5.2 Install Debian Packages](#)
- [6 Make Your Own OS Image](#)
 - [6.1 Install Cross Compiler](#)
 - [6.2 Compile U-Boot](#)
 - [6.3 Prepare mkimage](#)
 - [6.4 Compile Linux Kernel](#)
 - [6.4.1 Compile Kernel](#)

[Getting Started](#)
[Learning](#)

[Products](#)

[CPU Boards](#)
[Carrier Boards](#)
[NanoPC Series](#)
[Mini Boards](#)
[Matrix](#)
[LCD Modules](#)
[3D Models](#)
[Accessories](#)

[FAQ](#)

[Linux](#)
[Ubuntu](#)
[Android](#)
[WindowsCE](#)
[Qt](#)
[Non-OS](#)

[Support](#)

[English WebSite](#)
[English Forum](#)
[中文官方网站](#)
[中文论坛](#)
[FriendlyARM Github](#)

[Tools](#)

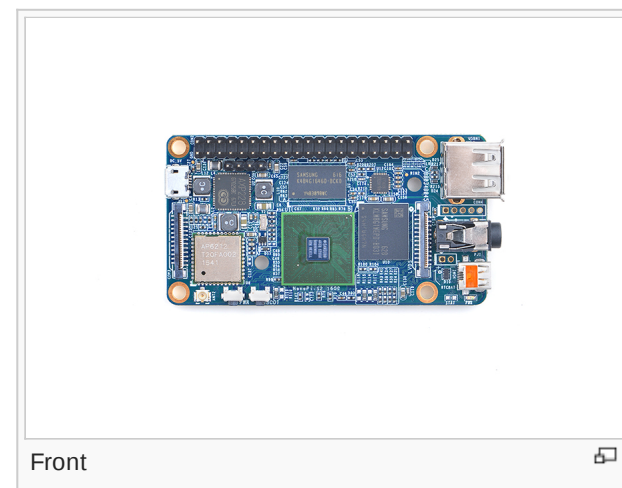
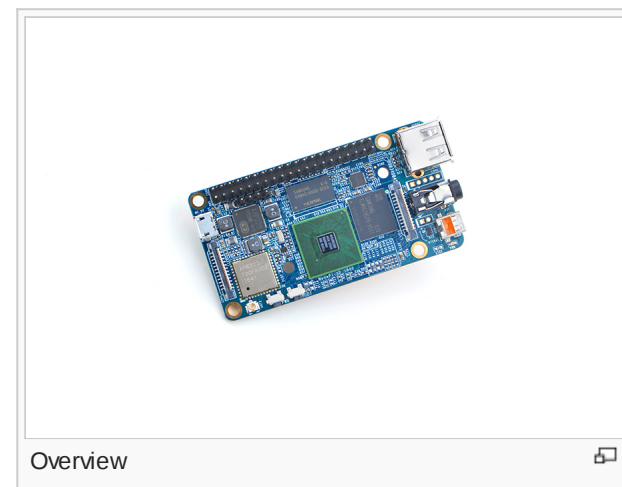
[What links here](#)
[Related changes](#)
[Special pages](#)

Introduction

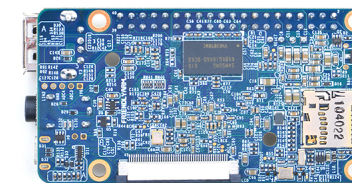
- The NanoPi S2(S2) is designed and developed by FriendlyElec for professionals, enterprise users, makers and hobbyists. It is a small board with WiFi, Bluetooth and eMMC.
- It uses the Samsung Quad Core Cortex-A9 S5P4418 SoC with dynamic frequency scaling up to 1.4GHz. It has 1G DDR3 RAM, 802.11 b/g/n WiFi & Bluetooth 4.0 module. Its in-built power management unit uses the AXP228 chip which supports software shutdown. It takes power over the MicroUSB port. It has video input/output interface, 3.5mm audio jack, USB port and MicroSD card slot, serial debug port and ADC pin-header.
- An Android and a Debian images are ready for the NanoPi S2. The Android OS supports HDMI and LCD output.
- The NanoPi S2 has various interfaces, ports such as LVDS, and GPIOs which are compatible with Raspberry Pi's GPIOs. Its PCB dimension is 40 * 75 mm. The NanoPi S2 works with most of the modules and OS images that are developed for FriendlyElec's S5P4418 based boards.

Hardware Spec

- CPU: S5P4418, dynamic frequency from 400Mhz to 1.4GHz
- PMU Power Management Unit: AXP228. It supports software shutdown and wake-up functions.
- DDR3 RAM: 1GB
- eMMC: 8GB
- Wireless : 802.11 b/g/n
- Bluetooth : 4.0 dual mode



- MicroSD Slot: 1 x MicroSD Slot
- Audio: 3.5mm jack/Via HDMI
- Microphone: 3.5mm jack
- USB Host: 1 x USB 2.0 Host
- Micro USB: 1 x MicroUSB, USB 2.0 for both data transmission and power input
- LCD Interface: 0.5 mm pitch 45-pin SMT FPC seat, for full-color LCD (RGB: 8-8-8)
- HDMI: microHDMI,1080P60 output
- DVP Camera Interface: 0.5mm pitch 24-pin FPC seat.
- LVDS : 0.5mm pitch 24-Pin FPC seat
- GPIO1: 2.54mm pitch 40pin, compatible with Raspberry Pi's GPIO. It includes UART, SPI, I2C, PWM, IO and etc
- ADC: onboard ADC pin header
- Serial Debug Port:2.54mm pitch 4-Pin header
- Antenna Interface: IPX
- User Key: 1 x Power , 1 x Boot Mode Switch
- LED: 1 x Power LED, 1 x System LED
- RTC: RTC Pins
- PCB Size(mm): 75 x 40, 8 layer , ENIG
- Power Supply: DC 5V/2A
- OS/Software: u-boot, Android5.1, Debian8



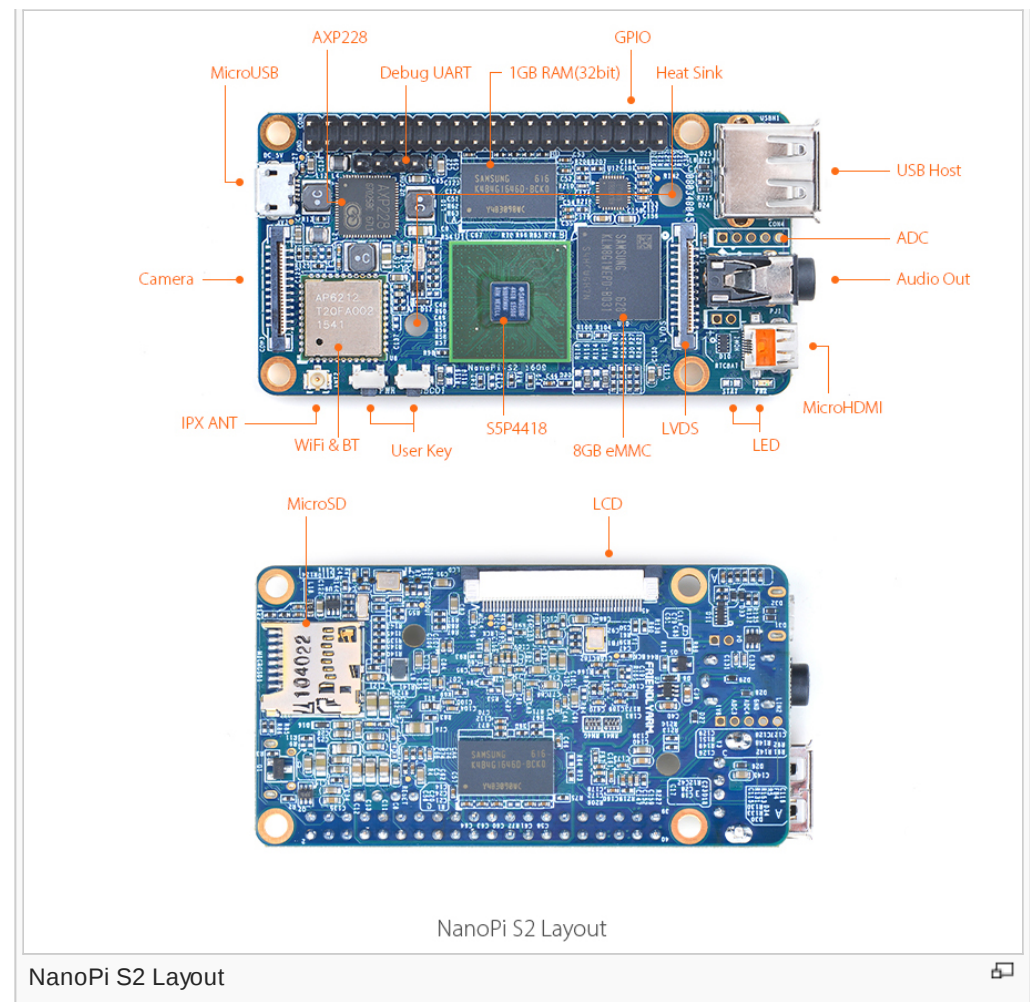
Back



Diagram, Layout and Dimension

Layout

- GPIO1 Pin Description



Pin#	Name	Pin#	Name
1	SYS_3.3V	2	VDD_5V
3	I2C0_SDA	4	VDD_5V
5	I2C0_SCL	6	DGND
7	GPIOD8/PPM	8	UART3_TXD/GPIOD21
9	DGND	10	UART3_RXD/GPIOD17
11	UART4_TX/GPIOB29	12	GPIOD1/PWM0
13	GPIOB30	14	DGND
15	GPIOB31	16	GPIOC14/PWM2

17	SYS_3.3V	18	GPIOB27
19	SPI0_MOSI/GPIOC31	20	DGND
21	SPI0_MISO/GPIOD0	22	UART4_RX/GPIOB28
23	SPI0_CLK/GPIOC29	24	SPI0_CS/GPIOC30
25	DGND	26	GPIOB26
27	I2C1_SDA	28	I2C1_SCL
29	GPIOC8	30	DGND
31	GPIOC7	32	GPIOC28
33	GPIOC13/PWM1	34	DGND
35	SPI2_MISO/GPIOC11	36	SPI2_CS/GPIOC10
37	AliveGPIO3	38	SPI2_MOSI/GPIOC12
39	DGND	40	SPI2_CLK/GPIOC9

The above pin description is different from that of the NanoPi 2. Here is a comparison table:[40 pins GPIO comparison table](#)

- **Debug Port (UART0)**

Pin#	Name
1	DGND
2	VDD_5V
3	UART_TXD0
4	UART_RXD0

- **DVP Camera Interface Pin Description**

Pin#	Name
1, 2	SYS_3.3V
7,9,13,15,24	DGND
3	I2C0_SCL
4	I2C0_SDA
5	GPIOB14

6	GPIOB16
8,10	NC
11	VSYNC
12	HREF
14	PCLK
16-23	Data bit7-0

- **RGB LCD Interface Pin Description**

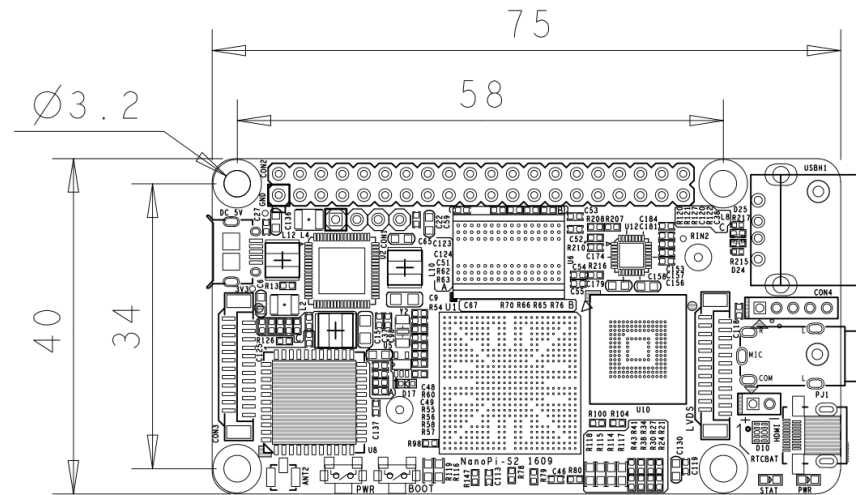
Pin#	Name	Description
1, 2	VDD_5V	5V Output, it can be used to power LCD modules
11 , 20 , 29, 37 , 38 , 39 , 40, 45	DGND	Ground
3-10	Blue LSB to MSB	RGB blue
12-19	Green LSB to MSB	RGB green
21-28	Red LSB to MSB	RGB red
30	GPIOB25	available for users
31	GPIOC15	occupied by FriendlyARM one wire technology to recognize LCD models and control backlight and implement resistive touch, not applicable for users
32	XnRSTOUT Form CPU	low when system is reset
33	VDEN	signal the external LCD that data is valid on the data bus
34	VSYNC	vertical synchronization
35	HSYNC	horizontal synchronization
36	LCDCLK	LCD clock, Pixel frequency
41	I2C2_SCL	I2C2 clock signal, for capacitive touch data transmission
42	I2C2_SDA	I2C2 data signal, for capacitive touch data transmission

43	GPIOC16	interrupt pin for capacitive touch, used with I2C2
44	NC	Not connected

Notes

1. SYS_3.3V: 3.3V power output
2. VDD_5V: 5V power output 5V. 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. GPIO1' pin description is different from that of the NanoPi 2. Here is a comparison table [40 pins GPIO comparison table](#)
5. For more details refer to the document:[]

Board Dimension



For more details please refer to the document: [NanoPi-S2-1609-dimensions](#)

Get Started

Essentials You Need

Before starting to use your NanoPi S2 get the following items ready

- NanoPi S2

- MicroSD Card/TF Card: Class 10 or Above, minimum 8GB SDHC
- A DC 5V/2A power is a must
- HDMI monitor or LCD
- USB keyboard, mouse and possible a USB hub(or a TTL to serial board)
- A host computer running Ubuntu 14.04 64 bit system

TF Card We Tested

To make your NanoPi S2 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:



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



Make an Installation MicroSD Card

Under Windows

Get the following files from here [download link](#):

- Get a 4G SDHC card and backup its data if necessary

FriendlyARM migrated both Android 5.1 and Android 4.4 to the NanoPi S2. Android 4.4 includes features that professional users usually need: 4G, Ethernet configuration and etc.

For LCD or HDMI output please use the following files:	
nanopi2-debian-sd4g.img.zip	Debian image file
nanopi2-android-sd4g.img.zip	Android5.1 image file
s5p4418-kitkat-sd4g-20160803.img.zip	Android4.4 image file
Flash Utility:	
win32diskimager.rar	Windows utility. Under Linux users can use "dd"

- Uncompress these files. Insert an SD card(at least 4G) into a Windows PC and run the win32diskimager utility 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.
- Insert this card into your NanoPi S2's boot slot, press and hold the boot key and power on (with a 5V/2A power source). If the green LED is on and the blue LED is blinking this indicates your NanoPi S2 has successfully booted.

Under Linux Desktop

- 1) Insert your microSD card to your host running Ubuntu and check your SD card's device name

```
dmesg | tail
```

Search the messages output by "dmesg" for similar words like "sd: sdc1 sdc2". If you can find them it means your SD card is recognized as "/dev/sdc". Or you can check that by commanding "cat /proc/partitions".

- 2) Download Flashing Script

```
git clone https://github.com/friendlyarm/sd-fuse_nanopi2.git
cd sd-fuse_nanopi2
```

- 3) Flash Android Firmware to MicroSD Card

```
su
./fusing.sh /dev/sdx
```

(Note: you need to replace "/dev/sdx" with the device name in your system) When you do "git clone" you have to hit "Y" within 10 seconds after

it prompts you to download image files otherwise you will miss the download.

- 4) Flash Debian Firmware to MicroSD Card

```
./fusing.sh /dev/sdx debian
```

Flash image to eMMC with eflasher

- Download eflasher

Get the eflasher utility s5p4418-eflasher-sd8g-xxx-full.img.7z

This package includes a Ubuntu Core, Debian, Android 5 and Android 4.4 image files;

Get the Windows utility: win32diskimager.rar;

- Flash eflasher Image

Extract the .7z package and you will get .img files. Insert an SD card (at least 4G) into a Windows PC and run the win32diskimager utility 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. If your PC runs Linux you can use the dd command to flash a .img file to the SD card;

- Flash image to eMMC

Insert this card into your NanoPi S2, connect the board to an HDMI monitor or an LCD, press and hold the boot key and power on (with a 5V/2A power source) the board. After your board is powered on you will see multiple OS options and you can select an OS to start installation.

Extend NanoPi S2's TF Card Section

- When Debian/Ubuntu is loaded the SD card's section will be automatically extended.
- When Android is loaded you need to run the following commands on your host PC to extend your SD card's section:

```
sudo umount /dev/sdx?  
sudo parted /dev/sdx unit % resizepart 4 100 resizepart 7 100 unit MB print  
sudo resize2fs -f /dev/sdx7
```

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

LCD/HDMI Resolution

When the system boots our uboot will check whether it is connected to an LCD or to an HDMI monitor. If it recognizes an LCD it will configure its resolution. Our uboot defaults to the HDMI 720P configuration.

If you want to modify the LCD resolution you can modify file "arch/arm/plat-s5p4418/nanopi2/lcds.c" in the kernel and recompile it.

If your NanoPi S2 is connected to an HDMI monitor and it runs Android it will automatically set the resolution to an appropriate HDMI mode by checking the "EDID". If your NanoPi S2 is connected to an HDMI monitor and it runs Debian by default it will set the resolution to the HDMI

720P configuration. If you want to modify the HDMI resolution to 1080P modify your kernel's configuration as explained above.

Update Image Files in SD Card From PC Host

If you want to make some changes to the image files in your SD card follow the steps below otherwise you can skip this section. Insert your SD card into a host PC running Linux, mount the boot and rootfs sections of the SD card and follow the steps below:

1) If you want to change your kernel command line parameters you can do it via the fw_setenv utility under "sd-fuse_nanopi2/tools".

Check the current Command Line:

```
cd sd-fuse_nanopi2/tools
./fw_printenv /dev/sdc | grep bootargs
```

Android 5.1.1_r6 starts SELinux. By default it is enforcing. You can change it this way:

```
./fw_setenv /dev/sdc bootargs XXX androidboot.selinux=permissive
```

This sets it to "permissive". The "XXX" stands for the original bootargs' value.

2) Update Kernel

Our customized uboot will check the LCD type when it boots.

For a non-Android OS if it recognizes that an LCD is connected to the NanoPi S2 it will load "ulmage" from "boot" otherwise it will load "ulmage.hdmi".

For Android it doesn't make any difference which display device is detected. You can use your generated ulmage to replace the existing one under "boot".

For Debian if your generated kernel is for an LCD you need to replace the existing ulmage or if your kernel is for an HDMI monitor you need to replace the existing ulmage.hdmi.

Run Android or Debian

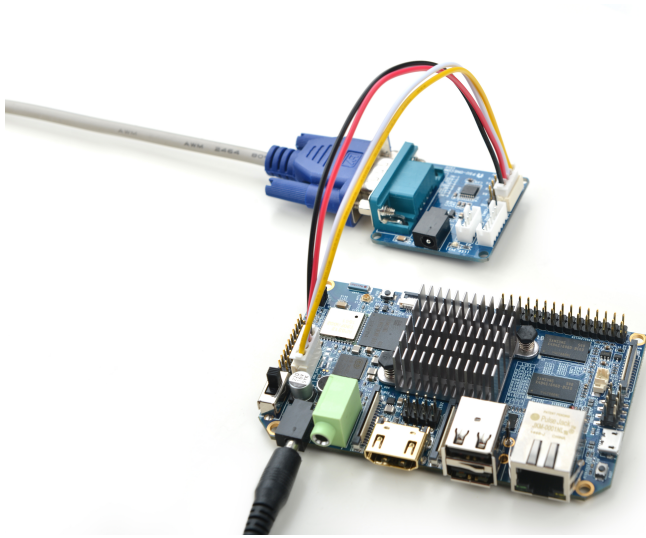
- Insert a MicroSD card with Android/Debian image file into your NanoPi S2, connect the board to an HDMI monitor, press and hold the boot key, power on the board the NanoPi S2 will boot from the SD card. If you can see the blue LED flashing it means your board is working and you will see Android/Debian being loaded on the HDMI monitor.

1) If you connect the NanoPi S2 to an HDMI monitor you need to use a USB mouse and a USB keyboard to operate. If you connect it to an LCD with capacitive touch you can operate directly on the LCD.

2) 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 S2 to a PC running Ubuntu and Minicom via a serial cable you will see system messages

output to the PC's minicom terminal:



- Under Debian the password for "root" is "fa"

Login to Debian via VNC & SSH

If your NanoPi S2 is not connected to a display device you can download and install a "VNC Viewer" from [here](#) on a mobile phone and login to the NanoPi S2 via VNC. Its default password is "fa123456". Here is a screenshot which shows how it looks like when users login to the NanoPi S2 from an iPhone via VNC:



You can login via "SSH -l root 192.168.8.1" the default password for "root" is "fa"

Working with Debian

Wireless Connection

Open the file `"/etc/wpa_supplicant/wpa_supplicant.conf"` with vi or gedit and append the following lines:

```
network={
    ssid="YourWiFiESSID"
    psk="YourWiFiPassword"
}
```

The `"YourWiFiESSID"` and `"YourWiFiPassword"` need to be replaced with your actual ESSID and password.

Save, exit and run the following commands your board will be connected to your specified WiFi:

```
ifdown wlan0
ifup wlan0
```

If your WiFi password has special characters or you don't want your password saved as plain text you can use `"wpa_passphrase"` to generate a psk for your WiFi password. Here is how you can do it:

```
wpa_passphrase YourWiFiESSID
```

Following the prompt type in your password. If you open the file `"/etc/wpa_supplicant/wpa_supplicant.conf"` you will find that your password has been updated and you can delete your clear-text password.

If the system's WiFi AP mode is on it cannot search and connect to a wireless router. You need to turn off the WiFi AP mode by following the instructions below:

```
su
turn-wifi-into-apmode no
```

Install Debian Packages

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

```
apt-get update
```

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

```
apt-get install vsftpd
```

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

Make Your Own OS Image

Install Cross Compiler

Download the compiler package:

```
git clone https://github.com/friendlyarm/prebuilts.git
sudo mkdir -p /opt/FriendlyARM/toolchain
sudo tar xf prebuilts/gcc-x64/arm-cortexa9-linux-gnueabihf-4.9.3.tar.xz -C /opt/FriendlyARM/toolchain/
```

Then add the compiler's directory to "PATH" by appending the following lines in "~/.bashrc":

```
export PATH=/opt/FriendlyARM/toolchain/4.9.3/bin:$PATH
export GCC_COLORS=auto
```

Execute "~/.bashrc" to make the changes take effect. Note that there is a space after the first ".":

```
. ~/.bashrc
```

This compiler is a 64-bit one therefore it cannot be run on a 32-bit Linux machine. After the compiler is installed you can verify it by running the following commands:

```
arm-linux-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gcc
COLLECT_LTO_WRAPPER=/opt/FriendlyARM/toolchain/4.9.3/libexec/gcc/arm-cortexa9-linux-gnueabihf/4.9.3/lto-wrapper
Target: arm-cortexa9-linux-gnueabihf
Configured with: /work/toolchain/build/src/gcc-4.9.3/configure --build=x86_64-build_pc-linux-gnu
--host=x86_64-build_pc-linux-gnu --target=arm-cortexa9-linux-gnueabihf --prefix=/opt/FriendlyARM/toolchain/4.9.3
--with-sysroot=/opt/FriendlyARM/toolchain/4.9.3/arm-cortexa9-linux-gnueabihf/sys-root --enable-languages=c,c++
--with-arch=armv7-a --with-tune=cortex-a9 --with-fpu=vfpv3 --with-float=hard
...
Thread model: posix
gcc version 4.9.3 (ctng-1.21.0-229g-FA)
```

Compile U-Boot

Download the U-Boot source code and compile it. Note that the github's branch is nanopi2-lollipop-mr1:

```
git clone https://github.com/friendlyarm/uboot_nanopi2.git
cd uboot_nanopi2
git checkout nanopi2-lollipop-mr1
make s5p4418_nanopi2_config
make CROSS_COMPILE=arm-linux-
```

After your compilation succeeds a u-boot.bin will be generated. If you want to test it flash it to your installation SD card via fastboot. Here is how you can do it:

- 1) On your host PC run "sudo apt-get install android-tools-fastboot" to install the fastboot utility;
- 2) Connect your NanoPi S2 to your host PC via a serial cable (e.g. PSU-ONECOME). Press the enter key within two seconds right after you power on your NanoPi S2 and you will enter uboot's command line mode;
- 3) After type in "fastboot" and press "enter" you will enter the fastboot mode;
- 4) Connect your NanoPi S2 to this host PC via a microUSB cable and type in the following command to flash u-boot.bin:

```
fastboot flash bootloader u-boot.bin
```

Warning: you cannot update this SD card by commanding "dd". This command will cause trouble when booting the NanoPi S2.

Prepare mkimage

You need the mkimage utility to compile a U-Boot source code package. Make sure this utility works well on your host before you start compiling a ulmage.

You can install this utility by either commanding "sudo apt-get install u-boot-tools" or following the commands below:

```
cd uboot_nanopi2
make CROSS_COMPILE=arm-linux- tools
sudo mkdir -p /usr/local/sbin && sudo cp -v tools/mkimage /usr/local/sbin
```

Compile Linux Kernel

Compile Kernel

- [Download Kernel Source Code](#)

```
git clone https://github.com/friendlyarm/linux-3.4.y.git
cd linux-3.4.y
git checkout nanopi2-lollipop-mr1
```

The NanoPi S2's kernel source code lies in the "nanopi2-lollipop-mr1" branch.

- Compile Android Kernel

```
make nanopi2_android_defconfig
touch .scmversion
make uImage
```

- Compile Debian Kernel

```
make nanopi2_linux_defconfig
touch .scmversion
make uImage
```

After your compilation succeeds a uImage will be generated in the "arch/arm/boot/uImage" directory. This kernel is for HDMI 720P. You can use it to replace the existing uImage.hdmi.

If you want to generate a kernel for HDMI 1080P you can do it this way:

```
touch .scmversion
make nanopi2_linux_defconfig
make menuconfig
  Device Drivers -->
    Graphics support -->
      Nexell Graphics -->
        [ ] LCD
        [*] HDMI
          (0) Display In [0=Display 0, 1=Display 1]
              Resolution (1920 * 1080p) ---->
make uImage
```

After your compilation succeeds a uImage will be generated for HDMI 1080P. You can use it to replace the existing uImage.

If you want to generate a kernel for an LCD you can do it this way:

```
touch .scmversion
make nanopi2_linux_defconfig
make menuconfig
  Device Drivers -->
    Graphics support -->
      Nexell Graphics -->
```



```
[*] LCD
[ ] HDMI
make uImage
```

After your compilation succeeds a ulmage will be generated for LCDs. You can use it to replace the existing ulmage.

Compile Kernel Modules

Android contains kernel modules which are in the "/lib/modules" directory in the system section. If you want to add your own modules to the kernel or you changed your kernel configurations you need to recompile these new modules.

Compile Original Kernel Modules:

```
cd linux-3.4.y
make CROSS_COMPILE=arm-linux- modules
```

Here we have two new modules and we can compile them by following the commands below:

```
cd /opt/FriendlyARM/s5p4418/android
./vendor/friendly-arm/build/common/build-modules.sh
```

The "/opt/FriendlyARM/s5p4418/android" directory points to the top directory of Android source code. You can get more details by specifying option "-h".

After your compilation succeeds new modules will be generated.

Compile Android

- Install Cross Compiler

Install 64 bit Ubuntu 14.04 on your host PC.

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

For more details refer to <https://source.android.com/source/initializing.html> ◦

- Download Android 5.1's Source Code

You need to use repo to get the Android source code. Refer to <https://source.android.com/source/downloading.html>

```
mkdir android && cd android
repo init -u https://github.com/friendlyarm/android_manifest.git -b nanopi2-lollipop-mr1
```

repo **sync**

The "android" directory is the working directory.

- Compile System Package

```
source build/envsetup.sh  
lunch aosp_nanopi2-userdebug  
make -j8
```



After your compilation succeeds the following files will be generated in the "out/target/product/nanopi2/" directory.

filename	partition	Description
boot.img	boot	-
cache.img	cache	-
userdata.img	userdata	-
system.img	system	-
partmap.txt	-	partition description file







- Copy Image to SD Card

Copy the image file to your installation SD card's sd-fuse_nanopi2/android/ directory and you can use this SD card to flash Android to eMMC. For more details please refer to [#Under Linux Desktop](#).

Source Code and Image Files Download Links

- Image File: [\[2\]](#) 
- Source Code: [\[3\]](#) 

Resources

- 《创客秘籍》 [Hacker's Book in Chinese by FriendlyARM](#) 
- 《创客秘籍-02》 [Hacker's Book-02 in Chinese by FriendlyARM](#) 
- 《创客秘籍-03》 [Hacker's Book-03 in Chinese by FriendlyARM](#) 
- SEC_Users_Manual_S5P4418_Users_Manual_Preliminary[4] 
- AXP228_Users_Manual_AXP228_V1.1_20130106 
- eMMC [eMMC5.0_1xnm_based_e_MMC](#) 

- [Schematic \(NanoPi-S2-1609-Schematic\)](#)
- Matrix Modules & Wiki Sites:
 - [Button](#)
 - [LED](#)
 - [A/D Converter](#)
 - [Relay](#)
 - [3-Axis Digital Accelerometer](#)
 - [3-Axis Digital Compass](#)
 - [Temperature Sensor](#)
 - [Temperature & Humidity Sensor](#)
 - [Buzzer](#)
 - [Joystick](#)
 - [I2C\(PCF8574\)+LCD1602](#)
 - [Sound Sensor](#)
 - [Ultrasonic Ranger](#)
 - [GPS](#)
 - [Matrix - Compact Kit](#)
 - [Fire Sensor](#)
 - [CAM500A Camera](#)
 - [BALL Rolling Switch](#)
 - [2'8 SPI Key TFT 2.8" SPI LCD](#)
 - [IR Counter](#)
 - [IR Receiver](#)
 - [L298N Motor Driver](#)
 - [MQ-2 Gas Sensor](#)
 - [MQ-3 Gas Sensor](#)
 - [One_Touch_Sensor](#)
 - [_Photoresistor](#)
 - [_Potentiometer](#)
 - [Pressure & Temperature Sensor](#)
 - [RGB LED](#)
 - [RTC](#)
 - [Rotary Encoder](#)

- [Soil Moisture Sensor](#)
- [Thermistor](#)
- [USB WiFi](#)
- [Water Sensor](#)

Update Log

Oct-25-2016

- Released English version

Nov-2-2016

- Added section 5.1
- Updated section 9

This page was last modified on 2 November 2016, at 14:57.

[Privacy policy](#) [About FriendlyARM Wiki](#) [Disclaimers](#)

