SBC1126B Linux6.1 IPC User Manual

V2.0

Boardcon Embedded Design

Overview

This document applies only to the SBC1126B development board. It is intended to help users quickly understand the hardware interfaces of the board and provides guidance for environment setup, source code compilation, firmware flashing, and functional testing of onboard hardware interfaces.

System Support

Development Board	IPC
Tiny1126B-V2 + SBC1126B-V2	Y

Revision History

Version	Date	Author	Revision History
V1.0	2026-07-08	Liu Yuan	Update vision

Disclaimer

The information in this manual is provided for reference only. While Boardcon has made every effort to ensure the accuracy and reliability of the content, no guarantee is given regarding its completeness or correctness. All information in this manual is subject to change without prior notice.

Boardcon reserves the right to revise or update the contents of this manual at any time without prior notification. Boardcon shall not be held liable for any direct or indirect loss or damage arising from the use of this document or the products described herein.

Boardcon embedded design limited

2007-11 Haofang Tianji Plaza, 11008 Beihuan Avenue, Nanshan District, Shenzhen, Guangdong, China. 518051

URL: www.armdesigner.com | www.boardcon.com

Email: market@armdesigner.com

Technical Support Inquiries: support@armdesigner.com

Tel: +86-755-26481393 | +86-755-27571591

Content

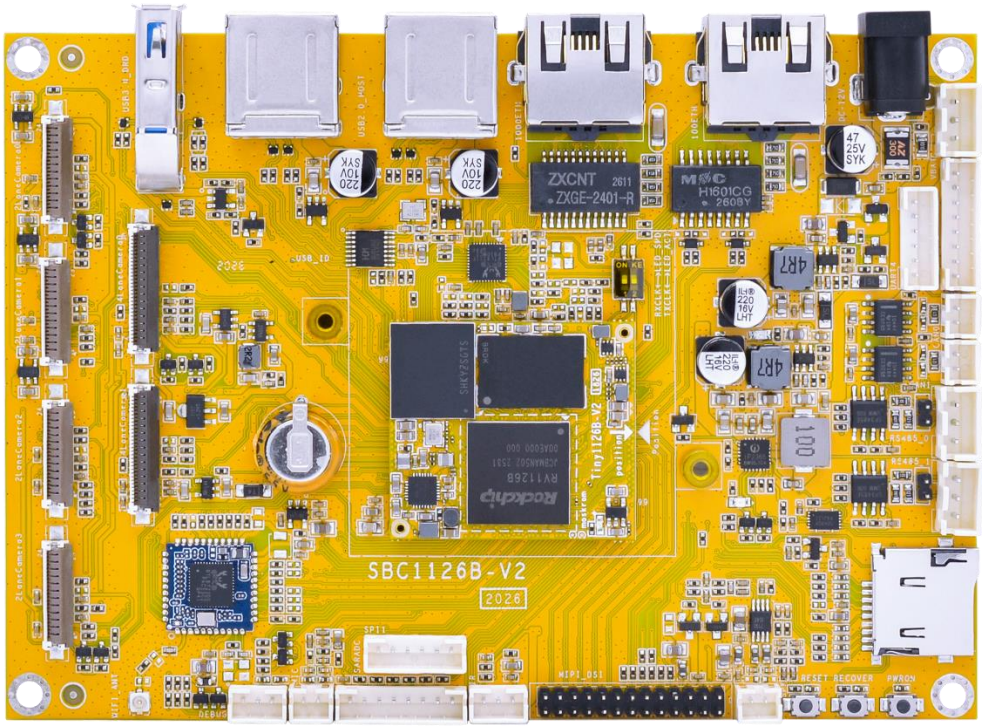
1. Introduction

1.1 Overview

SBC1126B is an embedded AI vision development board based on the Rockchip RV1126B processor. It uses the removable Tiny1126B core board and is suitable for AI vision, intelligent recognition, security monitoring, industrial control, and edge computing applications.

The board provides rich peripheral interfaces for product development and integration, including four camera inputs, MIPI DSI display output, USB3.0 DRD, dual USB2.0, 100M/1000M Ethernet, Wi-Fi/BT, CAN, RS485, UART, SPI, ADC, GPIO, IR, MicroSD, audio input/output, RTC battery, recovery, reset, and power control interfaces.

With its core-board and carrier-board design, SBC1126B offers good flexibility and maintainability. The removable Tiny1126B core board allows easier replacement, maintenance, and future upgrades, making it suitable for both development evaluation and product deployment.

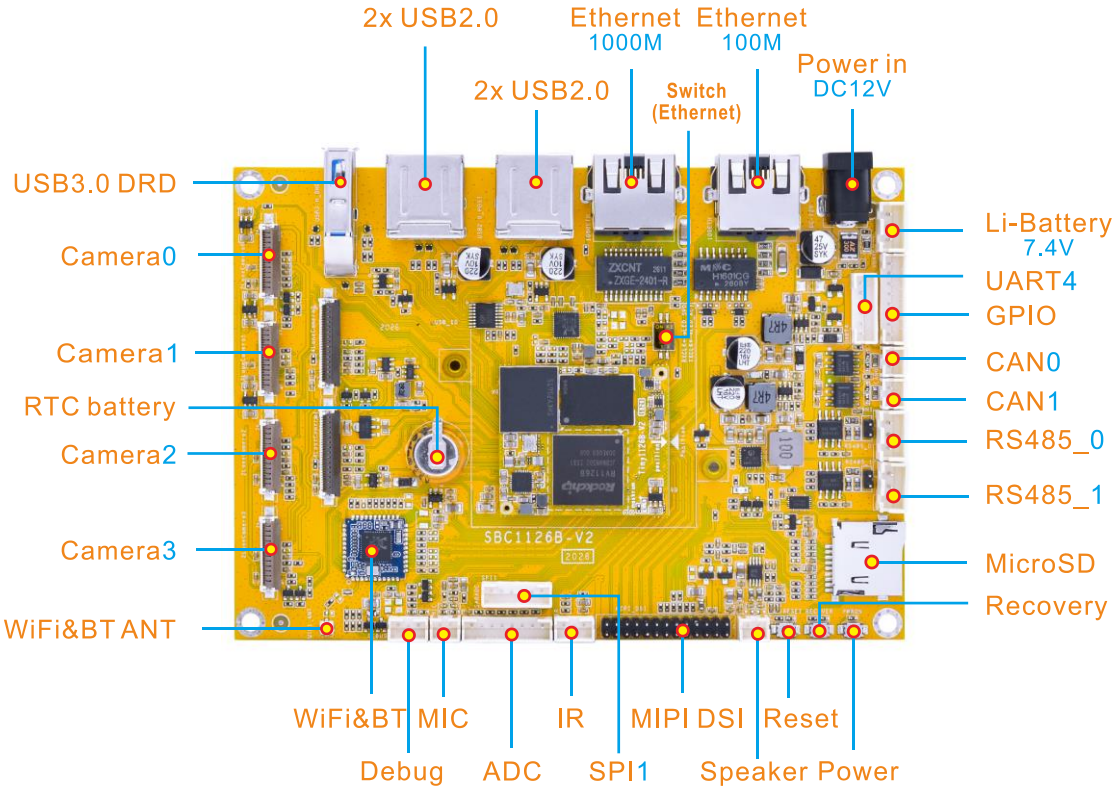


1.2 Product Parameters

Basic Parameters		
SOC		RV1126B
CPU		• Quad-core Arm Cortex-A53 64-bit processor • Neon SIMD and FPU support • 32 KB I-cache and 32 KB D-cache per core • 512 KB unified L2 cache
NPU		• Up to 3 TOPS AI computing power • Supports INT4, INT8, INT16, and FP16 operations • Supports mainstream AI frameworks, including TensorFlow, Caffe, TFLite, PyTorch, ONNX NN, and Android NN
Video	Decoder	• H.265/H.264 video decoding up to 3840×2160@30fps • JPEG decoding
	Encoder	• H.265/H.264 video encoding up to 3840×2160@30fps • JPEG encoding
AI-ISP		• Built-in AI-ISP engine • Supports up to 8MP@30fps • Supports AI-assisted image enhancement and spatial noise reduction
RAM		2GB LPDDR4x (up to 4GB)
ROM		8GB eMMC (up to 256GB)
Support system		Debian, Buildroot
Hardware Parameters		
Extended Storage		• Support 1x MicroSD Card
Display		• Support 1x DSI MIPI
Audio		• Support 1x Speaker

	• Support 1x MIC
USB	• Support 4x USB2.0 • Support 1x USB3.0 DRD
Network	• Support 1x 100M Ethernet • Support 1x 1000M Ethernet • Support 1x WIFI/BT module
Camera	• Support 4x Camera
Peripheral communication	• Support 1x SPI • Support 2x RS485 • Support 1x UART • Support 2x CAN
Other parameters	Support 1xDebug UART, 1xADC, 1xIR, 1xRTC, 1x Lion Battery connector
Electrical Parameters	
Power supply input voltage	12V/3A
RTC input voltage	3V/0.6uA
Operating temperature	0°C to 70°C
Storage temperature	-40°C to 85°C
Structural Parameters	
Core board dimensions	34.0mm x 30.0mm
Motherboard dimensions	135.0mm x 95.0mm

1.3 Hardware Interface Introduction



Interface parameters	
Power in DC 12V	12V DC power input interface
Ethernet 100M	100M Ethernet RJ45 interface
Switsh(Ethernet)	100Mbps/1000Mbps Ethernet Speed Switch
Ethernet 1000M	1000M Ethernet RJ45 interface
2x USB2.0	Dual-layer USB2.0 HOST interface
2x USB2.0	Dual-layer USB2.0 HOST interface
USB3.0 DRD	USB 3.0 DRD interface, supporting Host and Device modes
Camera0	Camera0 interface
Camera1	Camera1 interface
Camera2	Camera2 interface
Camera3	Camera3 interface

RTC battery	RTC battery interface
WIFI&BT	WIFI and Bluetooth module
WIFI&BT ANT	Wi-Fi/Bluetooth antenna interface
Debug	Debug serial port
MIC	Differential microphone input
ADC	ADC interface
IR	IR interface
SPI1	SPI interface
MIPI DIS	MIPI display interface
Speaker	Speaker output
Reset	Reset key
Power	Power key
Recovery	Recovery key
Mirco SD	MicroSD card slot
RS485_1	RS485 communication interface 1
RS485_0	RS485 communication interface 0
CAN1	CAN1 communication interface
CAN0	CAN0 communication interface
GPIO	GPIO expansion interface
UART4	UART4 serial communication interface
Li-Battery 7.4V	7.4V Lion Battery interface

2. Install Drivers and Tools

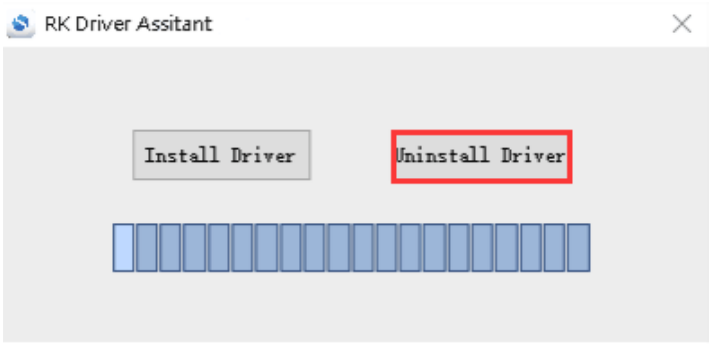
To download firmware and debug in the terminal, the following drivers and software need to be installed (for Windows computers):

Number	Driver name	Driver	Use
1	RK Driver Assistant	DriverInstall.exe	OTG USB driver installation assistant
2	CH9102x	SETUP.EXE	Serial port debugging driver
3	Serial Terminal Tool	SecureCRT.exe	Debugging tool

2.1 Install RK Driver Assistant

Step 1: Open *DriverAssistant/DriverInstall.exe*.

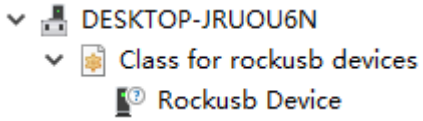
Step 2: To avoid driver conflicts, click **"Uninstall Driver"** to uninstall the driver.



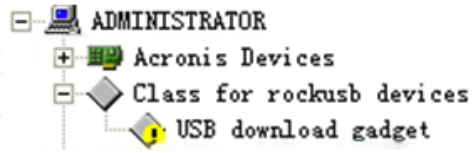
Step 3: Click button **"Install Driver"** to install.

Step 4: After the installation is complete, connect the board and PC with USB_A cable and press the **Recovery** key and hold then power the board, the following information is displayed in the Computer **Device Manager**, indicating that the USB driver was

successfully installed.

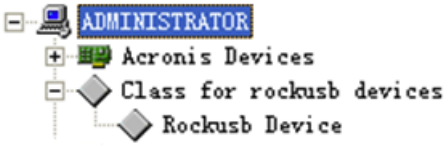


Step 5: If the following device information appears in the **Device Manager** after the operation in Step 4, user need to proceed to the next step.



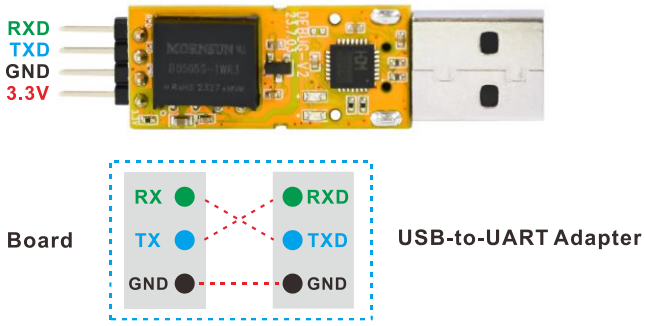
Step 6: The WINDOW will pop up found New Hardware Wizard dialog box, choose to install from the specified location, and then select: *DriverAssistant/ADBDriver*.

Step 7: After the installation is completed, the following device information can be seen in the Computer **Device Manager**.



2.2 Install CH9102X Driver

2.2.1 How to Connect the Serial Port Tool



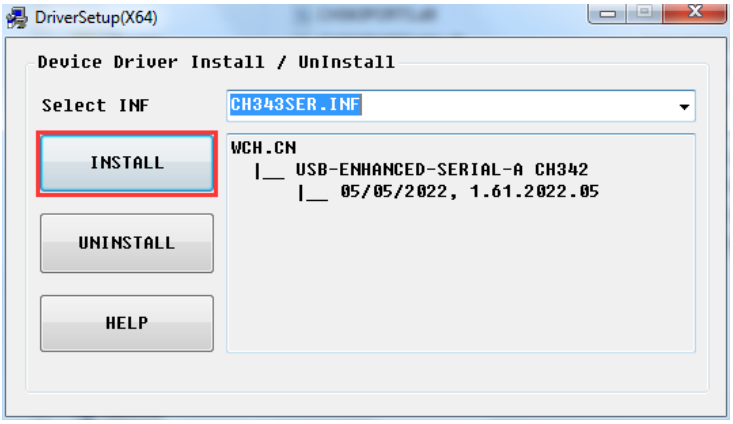
Pin	Connection Description
RXD	Receive, connect to TX pin of the board.
TXD	Transmit, connect to RX pin of the board.
GND	Ground, connect to GND pin of the board.
3V3	No need to connect.

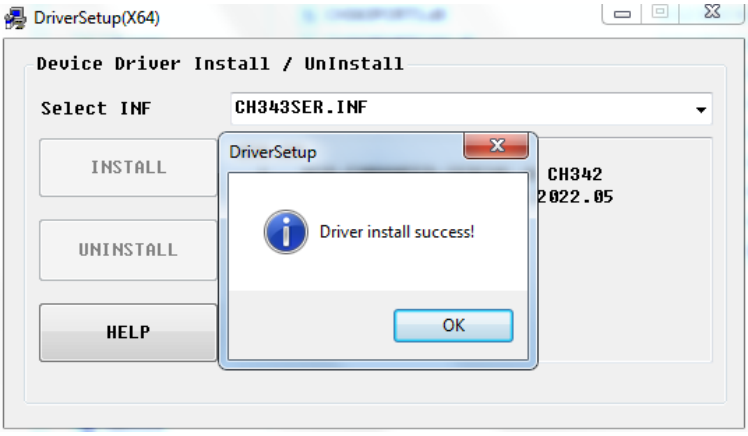
2.2.2 Install Driver

Step 1: Connect the CH9102X module to the PC.

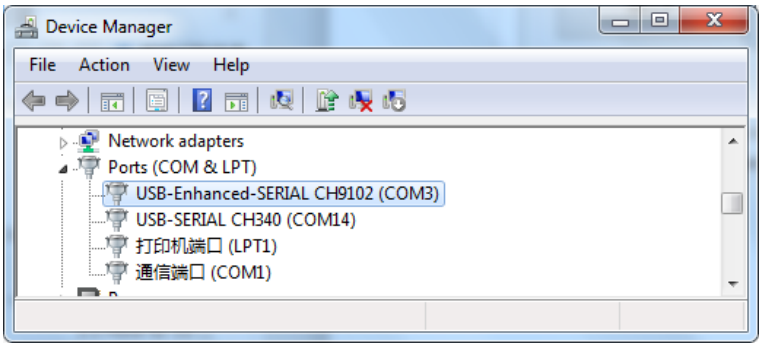
Step 2: Unzip *CH343SER.ZIP* on Windows.

Step 3: Select and install the corresponding *SETUP.EXE* according to the computer properties.





Step 4: After the installation is completed, the device will be listed under **Device Manager** ports with unique serial port assigned.

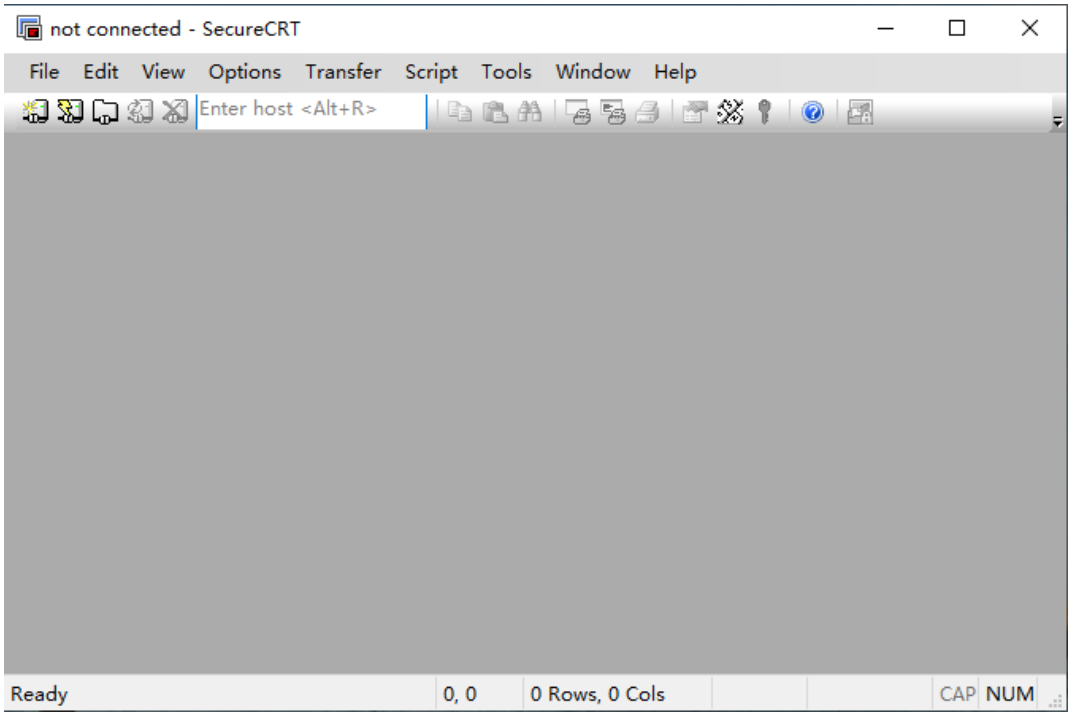


2.3 Install Serial Terminal Tool

The serial terminal SecureCRT is used for debugging in Windows. It can be used directly after decompression.

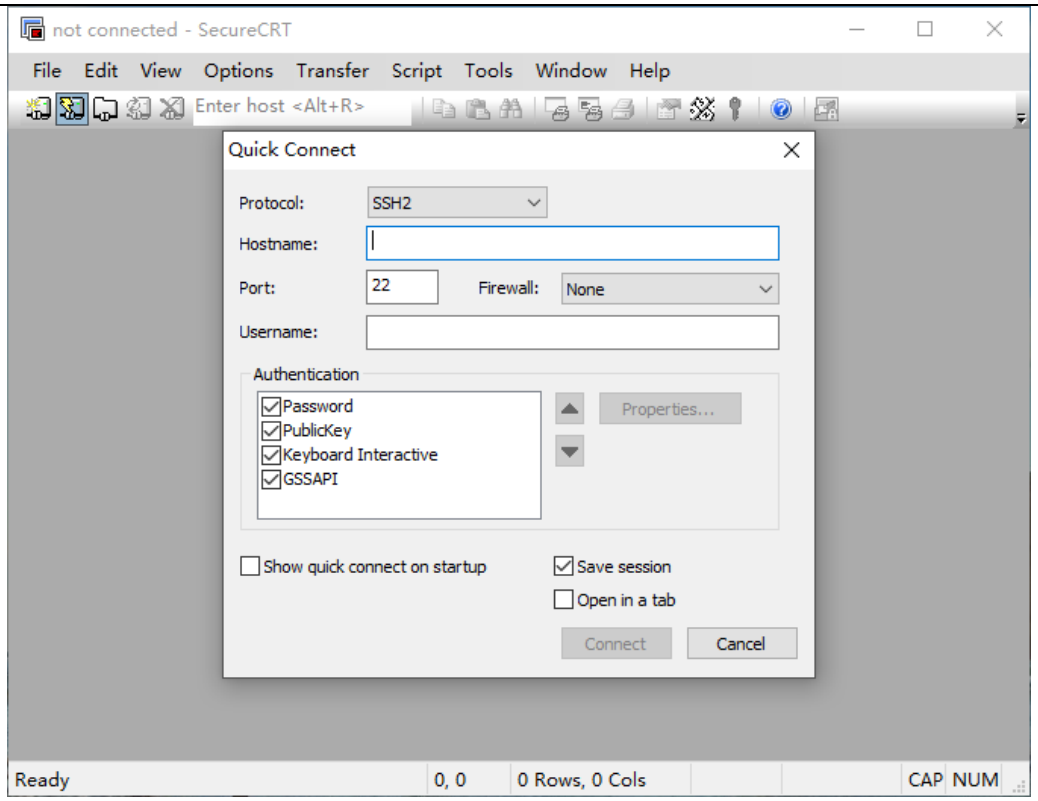
Step 1: Unzip *Platform/SecureCRT.rar* on PC.

Step 2: Click *SecureCRT/SecureCRT.exe* open to the SecureCRT.

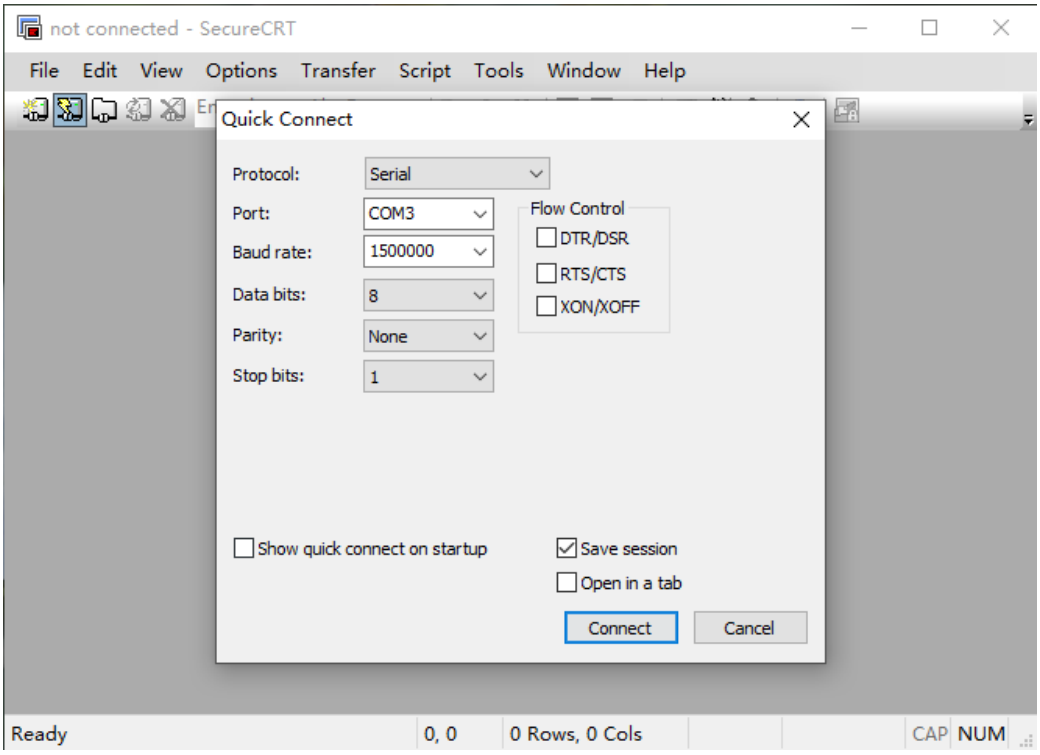


Step 3: Confirm that the CH9102X driver has been installed and the CH9102X module is connected to the PC.

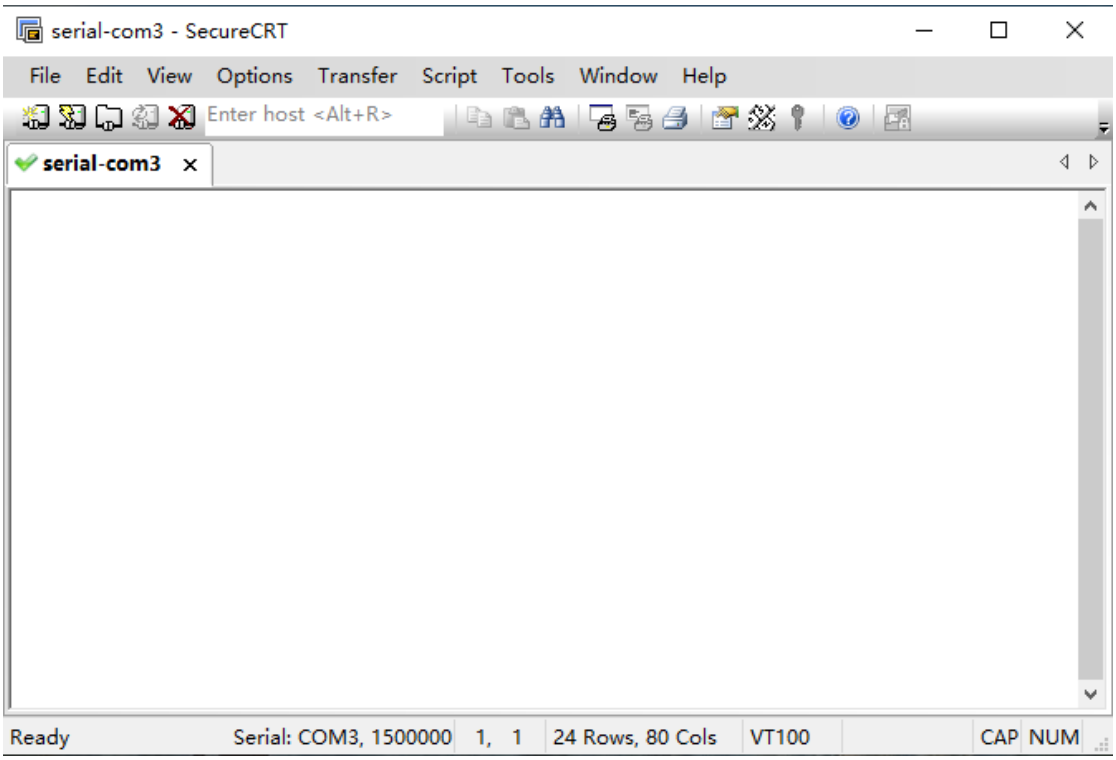
Step 4: Click the **"Quick Connect"** button to go to the Quick Connect configuration screen.



Step 5: Configure as shown in the following figure.



Step 6: After clicking the **"Connect"** button, the terminal serial interface will be successfully accessed.



3. Upgrade Introduction

3.1 Upgrade Mode

The firmware of this board must be upgraded in MaskRom Mode via USB cable.

Before upgrading, make sure the required USB driver has been installed on the PC. For driver installation instructions, refer to 2.1 Install RK Driver Assistant.

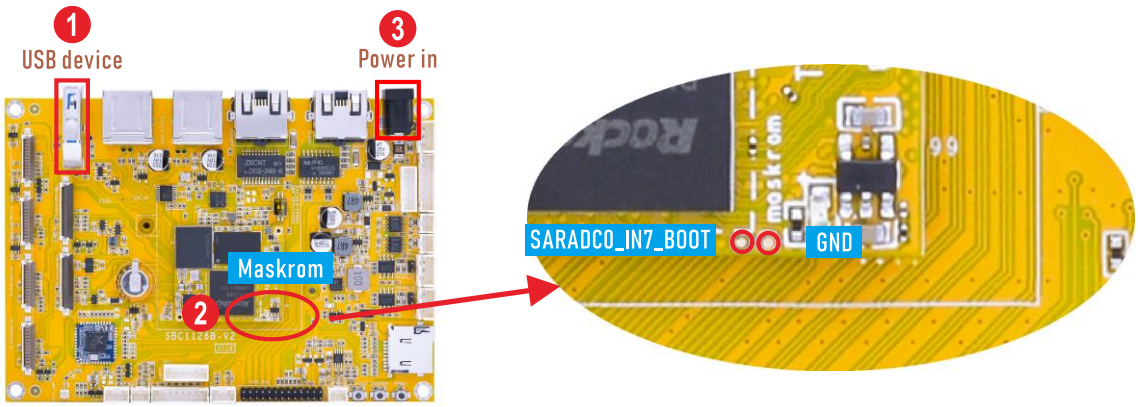
3.1.1 How to Enter MaskRom Mode

3.1.2.1 Hardware Method

Step 1: Disconnect the power adapter.

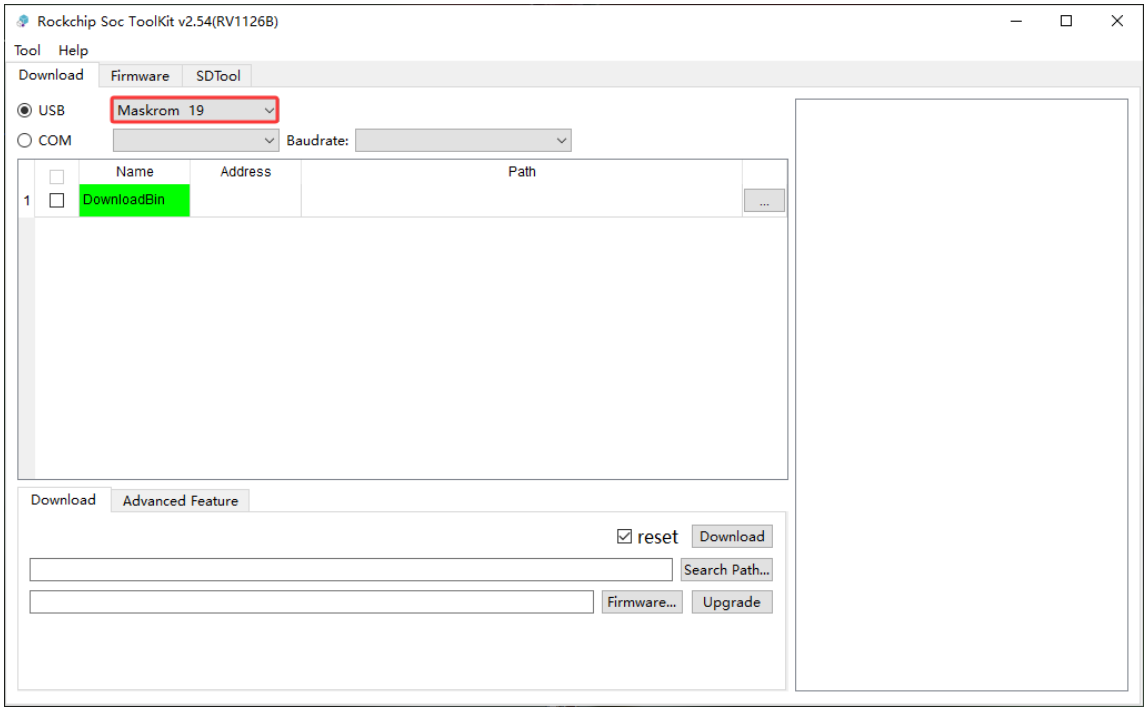
Step 2: Connect one end of the USB-A cable to the host PC and the other end to the development board.

Step 3: Use tweezers to short the two MaskRom test points on the Tiny1126B.



Step 4: While keeping the test points shorted, connect the power supply.

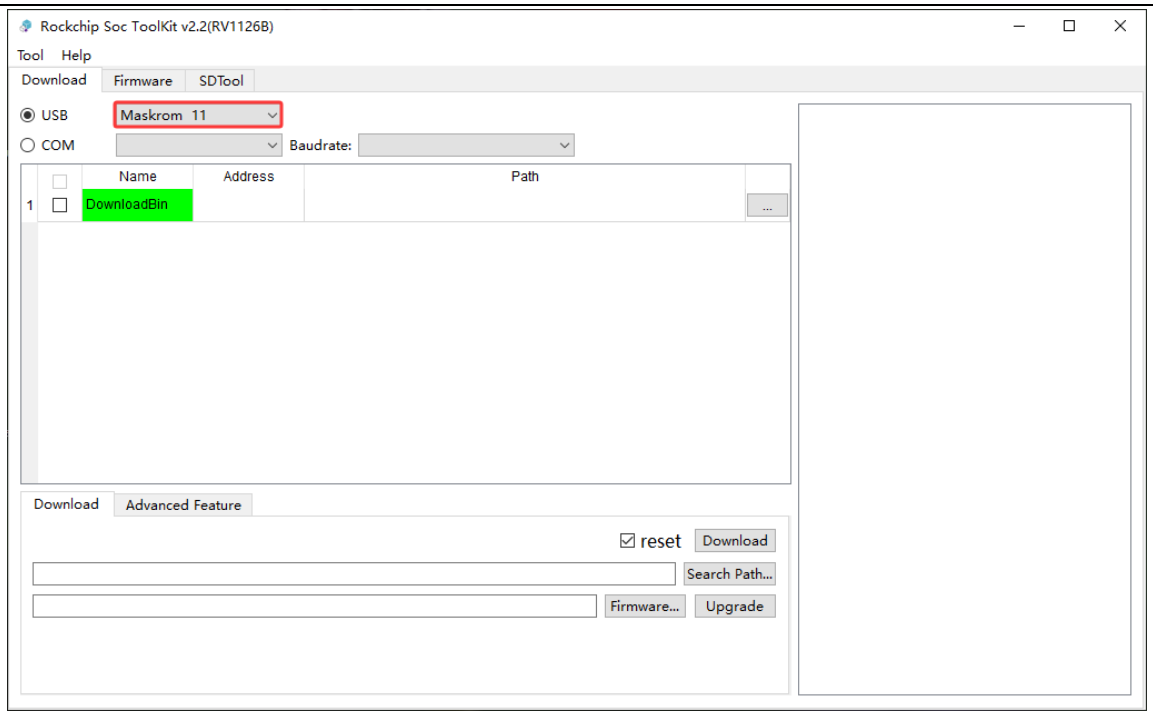
Step 5: After the USB device status changes to "**Maskrom**" in SocToolKit, release the tweezers. The device has now entered MaskRom mode.



3.1.2.2 Serial Terminal Method

Step 1: Connect one end of the USB_A cable to the host PC and the other end to the development board, then open the serial debug terminal.

Step 2: Press and hold **Ctrl + B** in the serial terminal, then power on the board. Release **Ctrl + B** when the flashing tool shows **"Maskrom"**.



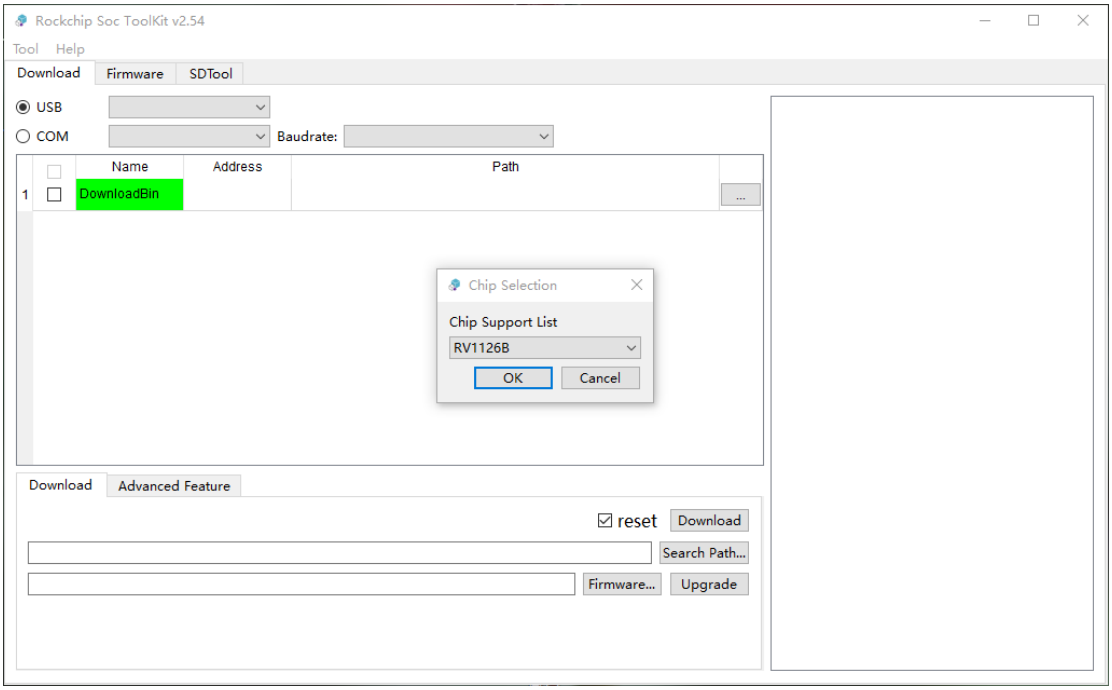
3.2 Firmware Flashing

Environment: Windows OS

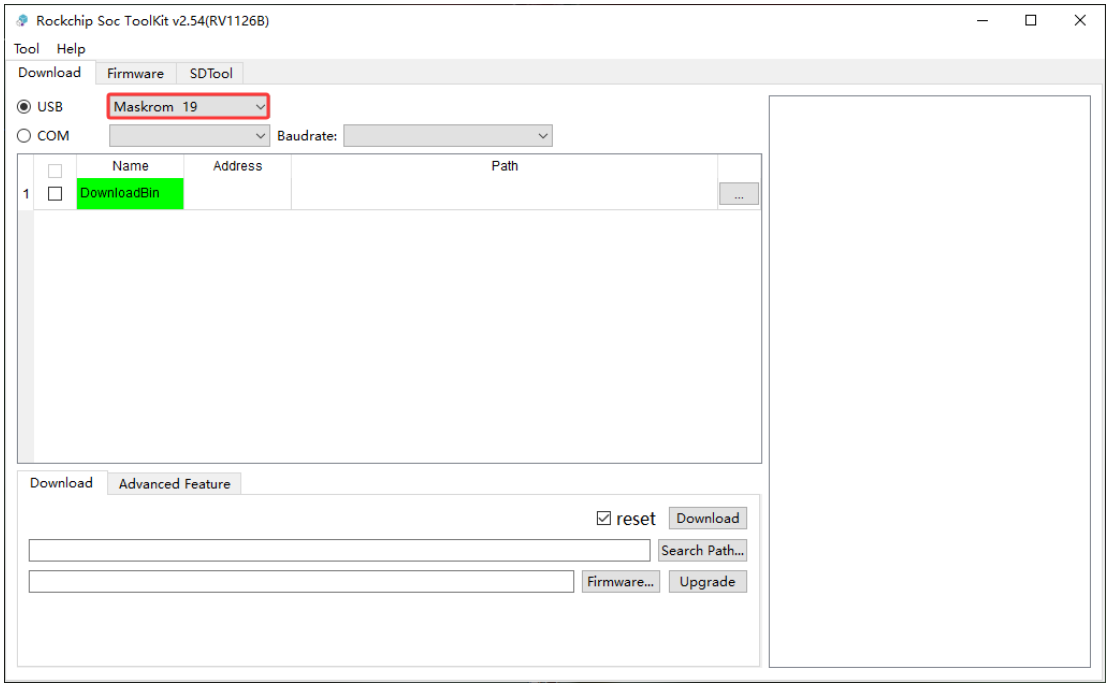
3.2.1 Flash Full Firmware Image

Step 1: Open *SocToolKit\SocToolKit.exe*.

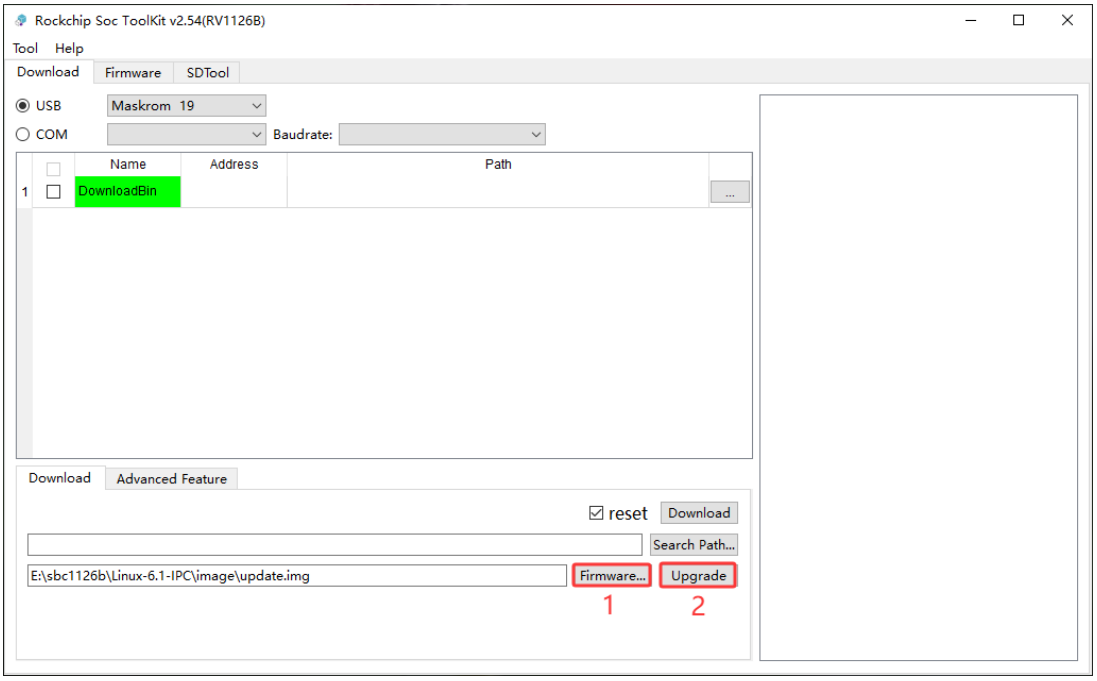
In the Chip Selection dialog box, keep the default chip option "**RV1126B**", and then click "**OK**".



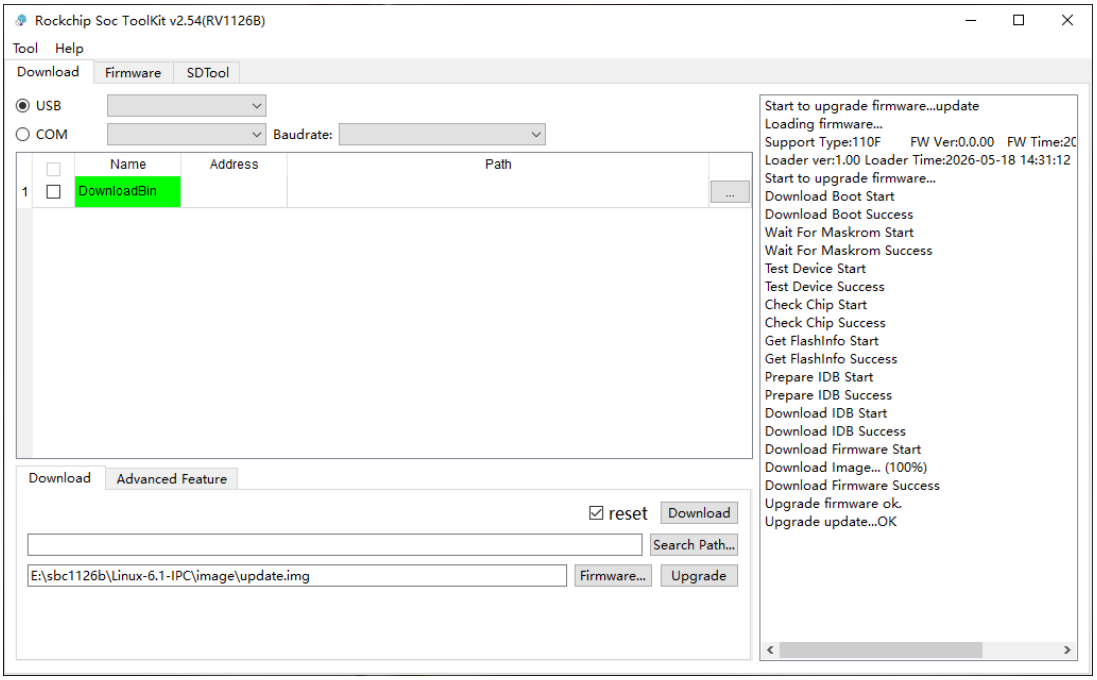
Step 2: Put the board into MaskRom mode. For details, refer to Section 3.1.2 How to Enter MaskRom Mode.



Step 3: Click **Firmware**, select **update.img**, and then click **Upgrade** to start flashing.

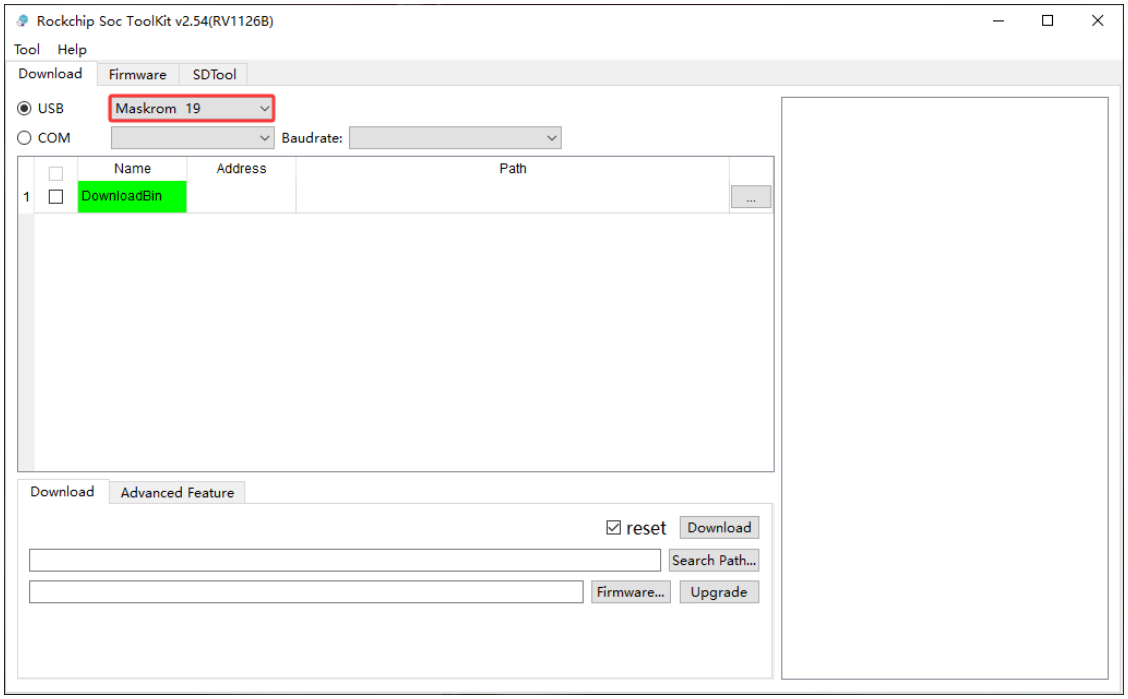


After the flashing is complete, the board will automatically reboot.

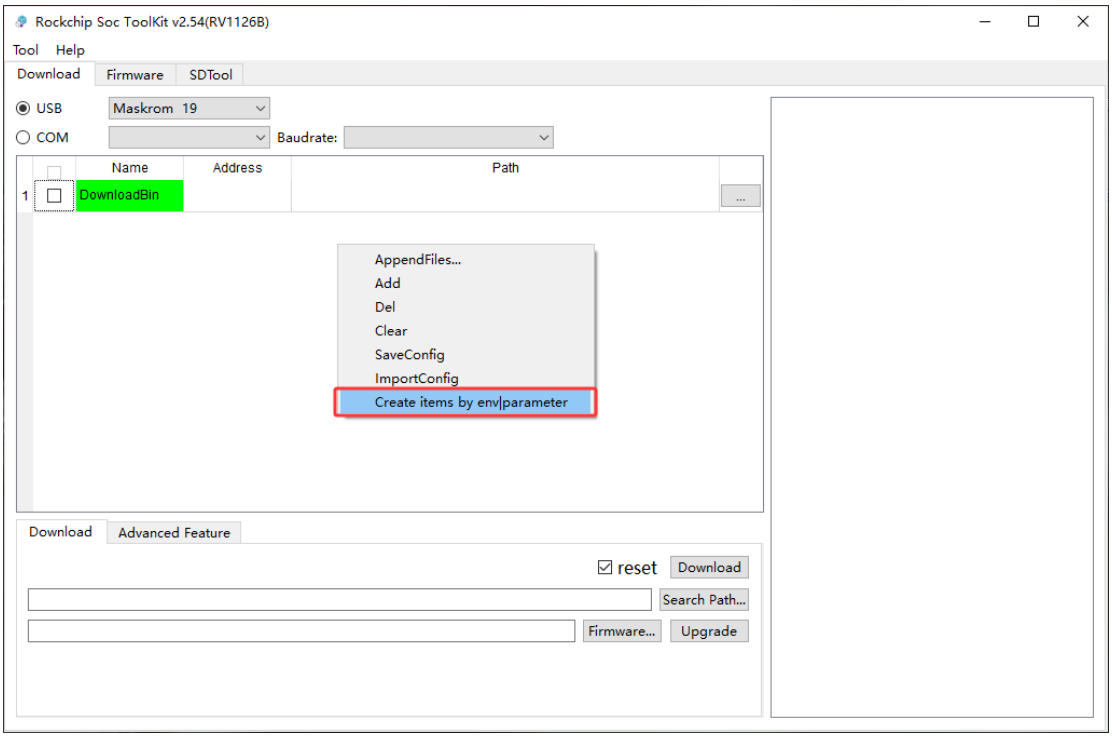


3.2.2 Flash Separate Partition Images

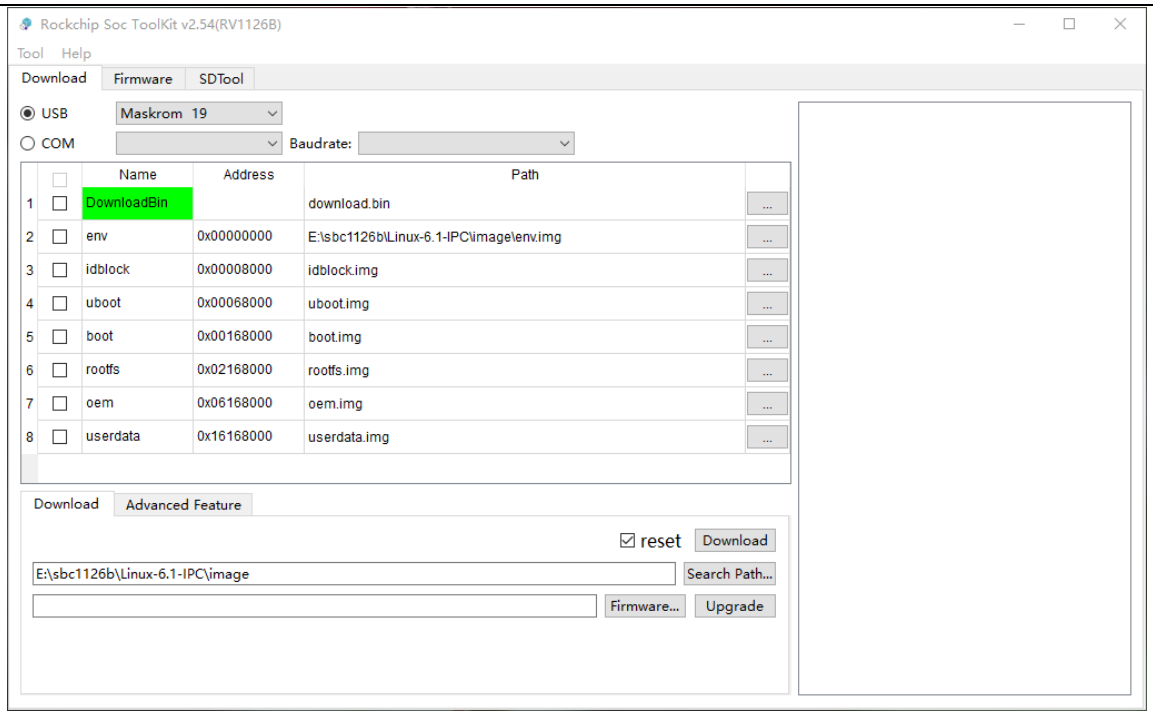
Step 1: Put the board into MaskRom mode. For details, refer to Section 3.1.2 How to Enter MaskRom Mode.



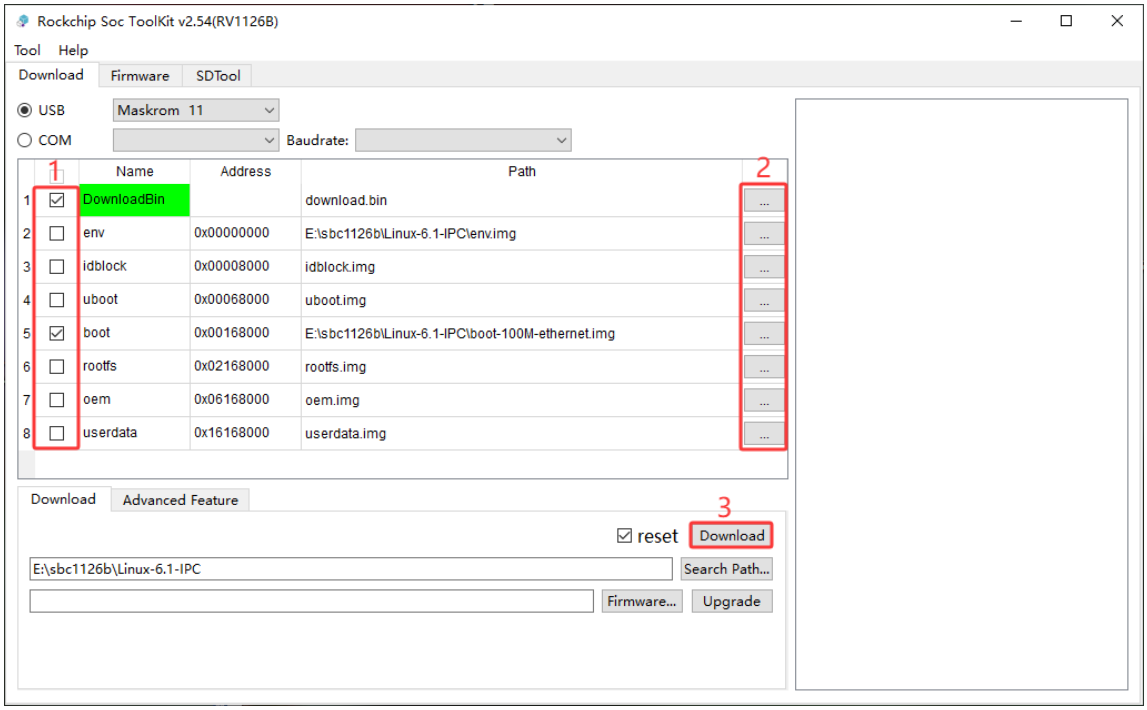
Step 2: Right-click the blank area in the image list and select **Create items by env|parameter**.



In the file selection window, select the **env.img** file from the separate image firmware directory. SocToolKit will automatically generate the flashing items and load the corresponding partition names, addresses, and image paths.

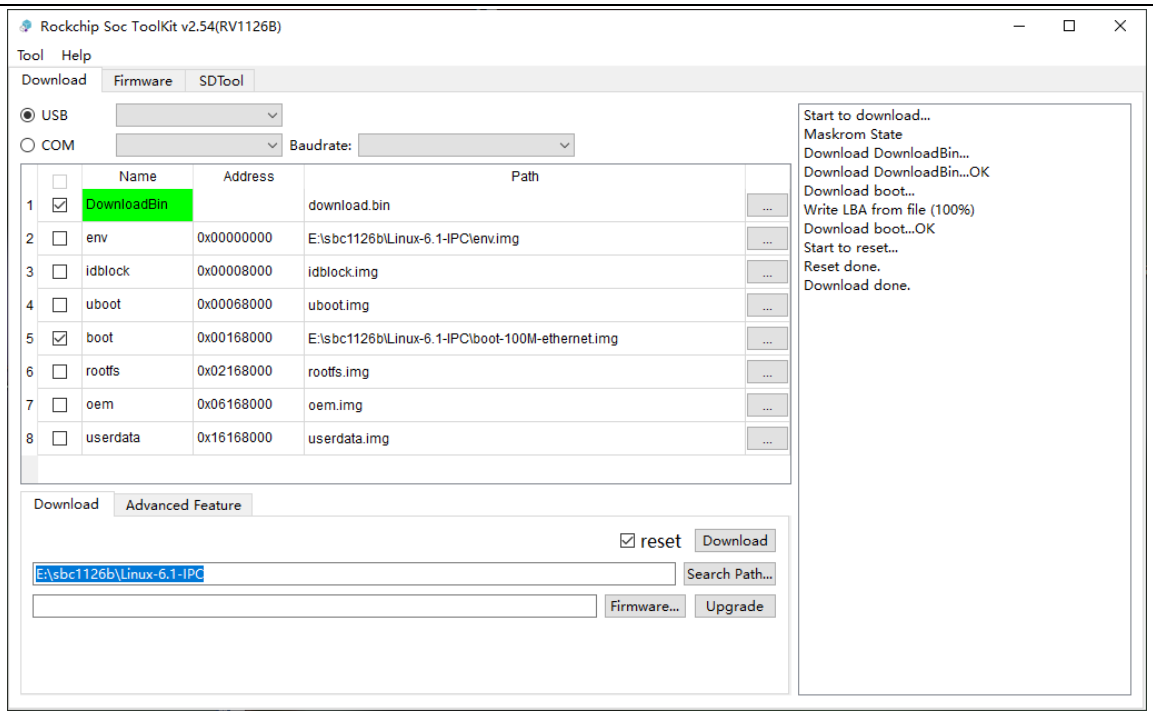


Step 3: Select the required image items in the list, click "**...**" to load the corresponding image files, and then click "**Download**" to start flashing.



Note: When flashing separate images, make sure that "DownloadBin" is selected. Otherwise, the flashing process may fail.

After flashing is complete, the board will automatically reboot.


Rockchip Soc ToolKit v2.54(RV1126B)
Tool
Help
Download
Firmware
SDTool
USB
COM
Baudrate:
Name
Address
Path
1 DownloadBin download.bin
2 env 0x00000000 E:\sbc1126b\Linux-6.1-IPC\env.img
3 idblock 0x00008000 idblock.img
4 uboot 0x00068000 uboot.img
5 boot 0x00168000 E:\sbc1126b\Linux-6.1-IPC\boot-100M-ethernet.img
6 rootfs 0x02168000 rootfs.img
7 oem 0x06168000 oem.img
8 userdata 0x16168000 userdata.img
Download
Advanced Feature
reset
Download
E:\sbc1126b\Linux-6.1-IPC
Search Path...
Firmware...
Upgrade
Start to download...
Maskrom State
Download DownloadBin...
Download DownloadBin...OK
Download boot...
Write LBA from file (100%)
Download boot...OK
Start to reset...
Reset done.
Download done.

4. Development Environment

4.1 Preparing the Development Environment

Ubuntu 22.04 or later is recommended for SDK compilation. If an error occurs during compilation, check the error message and install the required software packages accordingly. For other Linux distributions, the package names or installation commands may need to be adjusted.

In addition to the operating system requirements, the following hardware and software requirements should be met:

Hardware requirements	Software requirements
64-bit system, hard disk space should be greater than 200G. If you do multiple builds, you will need more hard drive space.	Ubuntu 22.04

4.2 Installing Libraries and Toolkits

This section provides the commands for installing the software packages required to build the SDK compilation environment. Other tools, such as Samba and SSH, should be installed as needed.

PC OS	Network	Permission
Ubuntu 22.04	online	root

To install the required tools, execute the following commands:

```
$ sudo apt-get install git ssh make gcc libssl-dev liblz4-tool libmpc-dev
$ sudo apt-get install expect g++ patchelf chrpath gawk texinfo chrpath diffstat
$ sudo apt-get install binfmt-support live-build bison flex fakeroot libgmp-dev
$ sudo apt-get install cmake gcc-multilib g++-multilib unzip device-tree-compiler
$ sudo apt-get install ncurses-dev libgucharmap-2-90-dev bzip2 expat gpgv2
$ sudo apt-get install cpp-aarch64-linux-gnu g++-aarch64-linux-gnu
$ sudo apt install python2 python-is-python3
```

5. Compile Source Code

5.1 Extract Source Code

To extract the source code package, execute the following commands:

```
$ tar xvf RV1126B_Linux_IPC*.tar.bz2
$ cd RV1126B_Linux_IPC_SDK
```

5.2 Configure the Target Board

To configure the target board, execute the following command:

```
$ ./build.sh lunch
```

When the BoardConfig list is displayed, select option **3**:

3.BoardConfig_IPC/BoardConfig-EMMC-RK801-Boardcon-SBC1126B_FASTBOOT.mk

```
You're building on Linux
Lunch menu...pick a combo:
BoardConfig-*.mk naming rules:
BoardConfig-"启动介质"-"电源方案"-"硬件版本"-"应用场景".mk
BoardConfig-"boot medium"-"power solution"-"hardware version"-"application".mk
----------------------------------------------------------------
0. BoardConfig_BatteryIPC/BoardConfig-EMMC-NONE-RV1126B_EVB2_V12-BAT_IPC.mk
                            boot medium(启动介质): EMMC
                          power solution(电源方案): NONE
                        hardware version(硬件版本): RV1126B_EVB2_V12
                            application(应用场景): BAT_IPC
----------------------------------------------------------------
----------------------------------------------------------------
1. BoardConfig_CVR/BoardConfig-EMMC-RK801-RV1126B_EVB1_V10-CVR.mk
                            boot medium(启动介质): EMMC
                          power solution(电源方案): RK801
                        hardware version(硬件版本): RV1126B_EVB1_V10
                            application(应用场景): CVR
----------------------------------------------------------------
----------------------------------------------------------------
2. BoardConfig_DV/BoardConfig-EMMC-RK801-RV1126B_EVB1_V12-DV_64bit.mk
                            boot medium(启动介质): EMMC
                          power solution(电源方案): RK801
                        hardware version(硬件版本): RV1126B_EVB1_V12
                            application(应用场景): DV_64bit
----------------------------------------------------------------
----------------------------------------------------------------
3. BoardConfig_IPC/BoardConfig-EMMC-RK801-Boardcon-SBC1126B_FASTBOOT.mk
                            boot medium(启动介质): EMMC
                          power solution(电源方案): RK801
                        hardware version(硬件版本): Boardcon
                            application(应用场景): SBC1126B_FASTBOOT
----------------------------------------------------------------
Which would you like? [0]: 3
[build.sh:info] switching to board:
/home/qinxueqin/1126b/sbc1126b/RV1126B_Linux_IPC_SDK/project/cfg/BoardConfig_IPC/BoardConfig-EMMC-RK801-
Boardcon-SBC1126B_FASTBOOT.mk
[build.sh:info] Running build_select_board succeeded.
```

5.3 Build the Firmware

The firmware can be built in either of the following ways:

- Build all components with one command.
- Build each component step by step.

5.3.1 Build All Components

To build all components with one command, execute:

```
$ ./build.sh
```

After the command is completed, the image files will be generated in the *output/image/* directory.

5.3.2 Build Components Step by Step

To build each component separately, execute the following commands as needed.

Step 1: Compile U-Boot

To compile U-Boot, execute the following command:

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

Step 2: Compile the Kernel

To compile the kernel, execute the following command:

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

Step 3: Compile the Sysdrv

To compile the sysdrv, execute the following command:

```
$ ./build.sh sysdrv
```

Step 4: Compile the Root Filesystem

To compile the root filesystem, execute the following command:

```
$ ./build.sh rootfs
```

Step 5: Compile Media

To compile media, execute the following command:

```
$ ./build.sh media
```

Step 6: Compile App

To compile app, execute the following command:

```
$ ./build.sh app
```

Step 7: Generate Firmware Images

After all required components are compiled, execute the following command to generate the firmware images:

```
$ ./build.sh firmware
```

After the command is completed, the image files will be generated in the *output/image/* directory.

5.3.3 Package Firmware Images

To package the firmware images into update.img, execute:

```
$ ./build.sh updateimg
```

After the command is completed, **update.img** will be generated in the *output/image/* directory.

6. Test

In the IPC version, the rkipc service starts automatically after system boot.

To avoid conflicts during function tests, stop the rkipc service before testing other functions.

Stop the rkipc service:

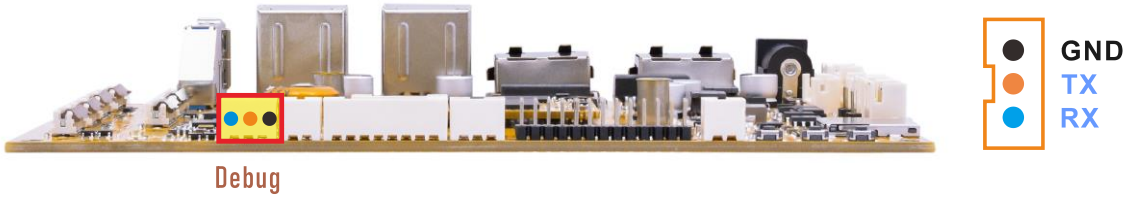
```
# RkLunch-stop.sh
```

Restart the rkipc service after testing:

```
# RkLunch.sh
```

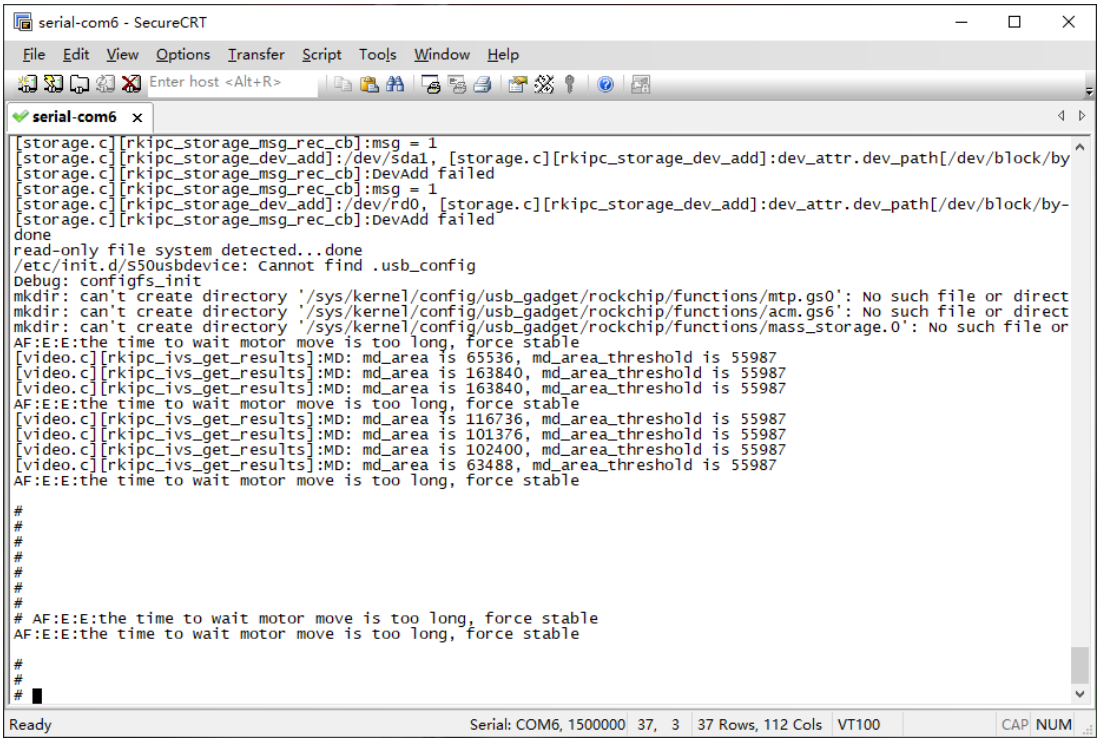
6.1 Serial Terminal

Step 1: Connect the USB-to-serial module to the PC and connect the other end to the Debug UART interface on the board.



Step 2: Open a serial terminal tool on the PC, select the corresponding serial port, and set the baud rate to 1500000.

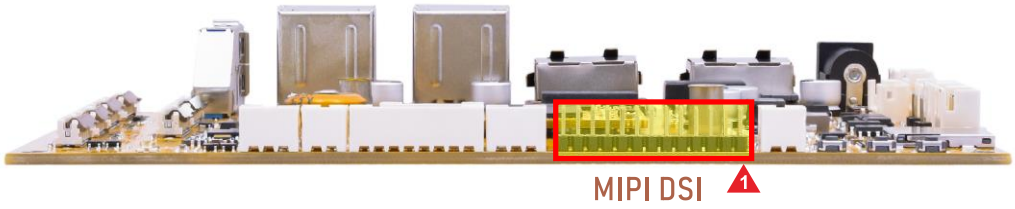
Step 3: Power on the board and check whether the boot log is displayed in the serial terminal.



For details about installing and using the serial terminal tool, please refer to Section [2.2 Install CH9102X Driver](#) and [2.3 Install Serial Terminal Tool](#).

6.2 Display

The default MIPI display resolution of the SBC1126B is 800×1280@60 Hz.



In the IPC version, the system does not display a desktop by default after startup.

If two cameras are connected before power-on, the system will automatically start the dual-camera preview after booting. The display output shows the stitched preview image from the two camera channels.

6.3 USB

6.3.1 USB OTG

The USB OTG port of the SBC1126B is configured as Device mode by default after system startup.

• To switch the USB OTG port to Host mode, execute the following command:

```
# echo host > /sys/kernel/debug/usb/21500000.usb/mode
```

```
# echo host > /sys/kernel/debug/usb/21500000.usb/mode
# cat /sys/kernel/debug/usb/21500000.usb/mode
host
```

• To switch the USB OTG port back to Device mode, execute the following command:

```
# /etc/init.d/S50usbdevice restart
# echo device > /sys/kernel/debug/usb/21500000.usb/mode
```

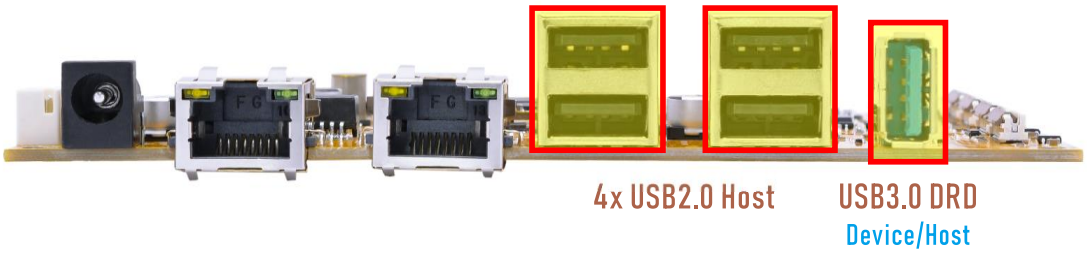
```
# /etc/init.d/S50usbdevice restart
mkdir: can't create directory '/dev/usb-ffs/adb': File exists
mount: mounting adb on /dev/usb-ffs/adb failed: Device or resource busy
# echo device > /sys/kernel/debug/usb/21500000.usb/mode
# cat /sys/kernel/debug/usb/21500000.usb/mode
device
```

6.3.2 USB HOST

The SBC1126B provides USB 2.0 Host interfaces and one USB 3.0 OTG interface. The USB 2.0 Host interfaces can be used to connect USB peripherals, such as a USB mouse, USB keyboard, USB flash drive, and other USB devices.

Note:

The USB 3.0 interface is an OTG interface. To use it as a Host interface, switch it to Host mode first. For details, refer to Section 6.3.1 USB OTG.



The current USB connection speed can be checked from the debug log.

• When the USB device operates in USB 2.0 mode, the log usually shows “high-speed”:

```
# dmesg | grep -Ei "high-speed|usb"
[    2.690102] usbcore: registered new interface driver usbfs
[    2.690174] usbcore: registered new interface driver hub
[    2.690248] usbcore: registered new device driver usb
[    2.709289] rockchip-usb2phy 21400000.usb2-phy: error -ENXIO: IRQ index 0 not found
[    2.751832] ohci-platform 214c0000.usb: Generic Platform OHCI controller
[    2.751876] ohci-platform 214c0000.usb: new USB bus registered, assigned bus number 1
[    2.752030] ohci-platform 214c0000.usb: irq 126, io mem 0x214c0000
[    2.813611] hub 1-0:1.0: USB hub found
[    2.843384] usbcore: registered new interface driver usbhid
[    2.843399] usbhid: USB HID core driver
[    2.852768] ehci-platform 21480000.usb: EHCI Host Controller
[    2.852798] ehci-platform 21480000.usb: new USB bus registered, assigned bus number 2
[    2.852903] ehci-platform 21480000.usb: irq 128, io mem 0x21480000
[    2.868527] ehci-platform 21480000.usb: USB 2.0 started, EHCI 1.00
[    2.869233] hub 2-0:1.0: USB hub found
[    2.927195] usbcore: registered new interface driver usb-storage
[    3.128522] usb 2-1: new high-speed USB device number 2 using ehci-platform
[    3.286767] hub 2-1:1.0: USB hub found
[   44.024581] usb 2-1.4: new high-speed USB device number 3 using ehci-platform
[   44.135239] usb-storage 2-1.4:1.0: USB Mass Storage device detected
[   44.136330] scsi host0: usb-storage 2-1.4:1.0
[   45.158039] scsi 0:0:0:0: Direct-Access     Lexar    USB Flash Drive  1.00 PQ: 0 ANSI: 4
```

• When the USB device operates in USB 3.0 mode, the log usually shows “SuperSpeed”:

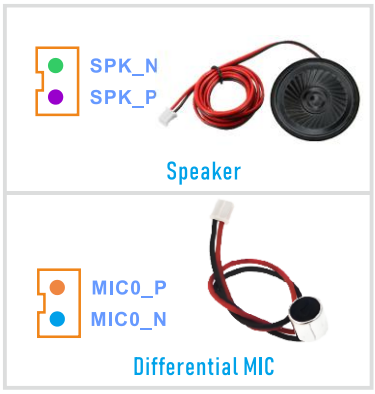
```
# dmesg | grep -Ei "SuperSpeed|xhci"
[  323.198029] xhci-hcd xhci-hcd.2.auto: xHCI Host Controller
[  323.198061] xhci-hcd xhci-hcd.2.auto: new USB bus registered, assigned bus number 3
[  323.198252] xhci-hcd xhci-hcd.2.auto: hcc params 0x0220fe64 hci version 0x110 quirks 0x0004008022010010
[  323.198306] xhci-hcd xhci-hcd.2.auto: irq 127, io mem 0x21500000
[  323.198493] xhci-hcd xhci-hcd.2.auto: xHCI Host Controller
[  323.198521] xhci-hcd xhci-hcd.2.auto: new USB bus registered, assigned bus number 4
[  323.198545] xhci-hcd xhci-hcd.2.auto: Host supports USB 3.0 SuperSpeed
[  323.585737] usb 4-1: new SuperSpeed USB device number 2 using xhci-hcd
[  374.896474] xhci-hcd xhci-hcd.2.auto: remove, state 1
[  374.984611] xhci-hcd xhci-hcd.2.auto: USB bus 4 deregistered
[  374.984677] xhci-hcd xhci-hcd.2.auto: remove, state 4
[  374.986066] xhci-hcd xhci-hcd.2.auto: USB bus 3 deregistered
[  433.981994] xhci-hcd xhci-hcd.2.auto: xHCI Host Controller
[  433.982032] xhci-hcd xhci-hcd.2.auto: new USB bus registered, assigned bus number 3
[  433.982210] xhci-hcd xhci-hcd.2.auto: hcc params 0x0220fe64 hci version 0x110 quirks 0x0004008022010010
[  433.982265] xhci-hcd xhci-hcd.2.auto: irq 127, io mem 0x21500000
[  433.982454] xhci-hcd xhci-hcd.2.auto: xHCI Host Controller
[  433.982478] xhci-hcd xhci-hcd.2.auto: new USB bus registered, assigned bus number 4
[  433.982501] xhci-hcd xhci-hcd.2.auto: Host supports USB 3.0 SuperSpeed
[  434.369765] usb 4-1: new SuperSpeed USB device number 2 using xhci-hcd
```

After a USB flash drive is connected, it will be mounted automatically. Run the following command to check the mount path of the USB flash drive:

```
# df -h
```

```
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                49.5M     49.5M         0 100% /
devtmpfs                 60.4M         0     60.4M   0% /dev
tmpfs                   966.9M         0    966.9M   0% /dev/shm
tmpfs                   966.9M      8.0K    966.9M   0% /tmp
tmpfs                   966.9M    384.0K    966.5M   0% /run
/dev/block/by-name/oem
                        237.9M     58.1M    164.3M  26% /oem
/dev/block/by-name/userdata
                          5.8G      9.3M      5.7G   0% /userdata
/dev/sda1                32.0G      5.4G     26.6G  17% /tmp/sda1
```

6.4 Audio I/O



6.4.1 Audio Output Test

Step 1: Set the DAC digital volume.

Execute the following command:

```
# rk_mpi_amix_test -C 0 --control 'DAC Digital Volume' --value '250,250'
```

Step 2: Play the audio file.

Execute the following command according to the actual audio file path:

```
# rk_mpi_ao_test -i /tmp/sda1/video/music/2.wav --input_ch 2 --input_rate 44100 --device_rate 44100 --device_ch 2 --sound_card_name hw:0,0
```

```
# rk_mpi_amix_test -C 0 --control 'DAC Digital Volume' --value '250,250'
cmd parse result:
sound control id     : 0
control name         : DAC Digital Volume
control value        : 250,250
list controls        : 0
list contents        : 0
#
# rk_mpi_ao_test -i /tmp/sda1/video/music/2.wav --input_ch 2 --input_rate 44100
--device_rate 44100 --device_ch 2 --sound_card_name hw:0,0
cmd parse result:
input  file name     : /tmp/sda1/video/music/2.wav
output file name     : (null)
loop count           : 1
channel number       : 1
open sound rate      : 44100
open sound channel    : 2
input stream rate    : 44100
input channel        : 2
bit_width            : 16
period_count         : 4
period_size          : 4096
sound card name      : hw:0,0
device id            : 0
set volume curve     : 0
set volume           : 100
set mute             : 0
set track_mode       : 0
get volume           : 0
get mute             : 0
get track_mode       : 0
query stat           : 0
pause and resume chn  : 0
save file            : 0
query save file stat  : 0
clear buf            : 0
get attribute        : 0
clear attribute      : 0
set loopback mode    : 0
vqe enable           : 0
rockit_load start
v4l2_tx probe successv4l2_rx_probe success
rockit_load end
rockit default level 4, can use export rt_log_level=x, x=0,1,2,3,4,5,6 change
rockit log path (null), log_size = 0
log_file = (nil)
(null)          09:40:21-804 {read_log_level    :083} text is all=4
(null)          09:40:21-804 {read_log_level    :085} module is all, log_level is 4
(null)          09:40:21-804 {dump_version      :060} ----------------------------------------
-------------
(null)            09:40:21-804 {dump_version      :061} rockit version: git-b85f3c523 Mon Jan 26
09:47:22 2026 +0800
(null)              09:40:21-804 {dump_version       :062} rockit building: built-Chu 2026-01-27
09:45:41
(null)          09:40:21-804 {dump_version      :063} ----------------------------------------
-------------
(null)          09:40:21-804 {monitor_log_level :141} #Start monitor_log_level thread, arg:(nil)
vsys dev open 4
load library(librga.so) in releative path
cmpi            09:40:21-823 {rk_mpi_ao_test_ent:717} start running loop count  = 0
cmpi            09:40:21-829 {test_init_mpi_ao  :235} Set volume curve type: 0
cmpi            09:40:21-829 {sendDataThread    :318} params->s32ChnIndex : 0
cmpi            09:40:21-829 {commandThread     :397} test info : mute = 0, volume = 100
```

If the audio file is stored in another directory, replace **/tmp/sda1/video/music/2.wav** with the actual file path.

6.4.2 Audio Input Test

Step 1: Stop the rkipc service.

Execute the following command:

```
# RkLunch-stop.sh
```

```
# RkLunch-stop.sh
Stop Application ...
[rkipc.c][sig_proc]:received signo 15
killall: udhcpc: no process killed
  349 root     2827m S<   rkipc -a /oem/usr/share/iqfiles
rkipc active
[server.c][rkipc_server_deinit]:rkipc_server_deinit failed
[osd.c][osd_time_server]:exit
[video.c][rkipc_osd_bmp_destroy]:RK_MPI_RGN_Destroy [2] failed with 0xa0038005
[video.c][rkipc_osd_bmp_destroy]:RK_MPI_RGN_Destroy [3] failed with 0xa0038005
[video.c][rkipc_osd_cover_destroy]:RK_MPI_RGN_Destroy [4] failed with 0xa0038005
```

Step 2: Select MIC as the recording input source.

Execute the following command:

```
# rk_mpi_amix_test -C 0 --control 'SAI2 Receive PATH0 Source Select' --value
'From SDI0'
```

```
# rk_mpi_amix_test -C 0 --control 'SAI2 Receive PATH0 Source Select' --value  'F
rom SDI0'
cmd parse result:
sound control id     : 0
control name         : SAI2 Receive PATH0 Source Select
control value        : From SDI0
list controls        : 0
list contents        : 0
#
```

Step 3: Set the audio codec digital gain.

Execute the following command:

```
# rk_mpi_amix_test -C 0 --control 'ACodec Digital Gain Volume' --value '100'
```

```
# rk_mpi_amix_test -C 0 --control 'ACodec Digital Gain Volume' --value '100'
cmd parse result:
sound control id     : 0
control name         : ACodec Digital Gain Volume
control value        : 100
list controls        : 0
list contents        : 0
```

Step 4: Record audio from MIC.

Execute the following command:

```
# rk_mpi_ai_test --device_rate 44100 --device_ch 2 --out_rate 44100 --out_ch 2 -o
/oem --sound_card_name hw:0,0
```

```
# rk_mpi_ai_test --device_rate 44100 --device_ch 2 --out_rate 44100 --out_ch 2 -
o /oem --sound_card_name hw:0,0
cmd parse result:
input  file name     : (null)
output file name     : /oem
loop count          : 1
channel number       : 1
open sound rate      : 44100
record data rate     : 44100
sound card channel    : 2
output channel       : 2
bit_width           : 16
frame_number         : 4
frame_length         : 1024
sound card name      : hw:0,0
device id           : 0
set volume curve     : 0
set volume          : 100
set mute            : 0
set track_mode       : 0
get volume          : 0
get mute            : 0
get track_mode       : 0
data read enable     : 0
aed enable          : 0
bcd enable          : 0
buz enable          : 0
vqe gap duration (ms) : 16
vqe enable          : 0
get vqe result       : 0
bcd NN model path    : (null)
vqe config file      : (null)
dump algo pcm data    : 0
dev queue len        : 0
sed queue len        : 0
fd get enable        : 1
rockit_load start
v412_tx probe successv412_rx_probe success
rockit_load end
rockit log path (null), log_size = 0
log_file = (nil)
(null)         10:46:52-197 {log_level_init    :209}

 please use echo name=level > /tmp/rt_log_level set log level
       name: all cmpi mb sys vdec venc rgn vpss vgs tde avs wbc vo vi ai ao aenc adec
       log_level: 0 1 2 3 4 5 6
```

Step 5: Play the recorded audio through the speaker.

Execute the following command:

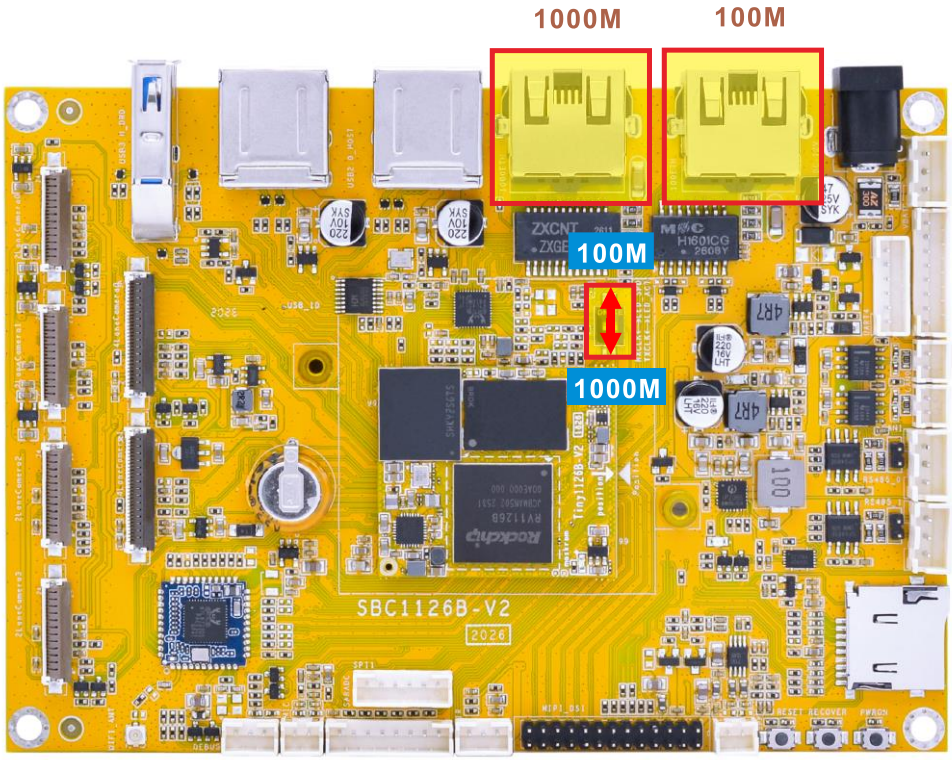
```
# rk_mpi_ao_test -i /oem/cap_out.pcm --input_ch 2 --input_rate 44100 --
device rate 44100 --device ch 2 --sound card name hw:0,0
```

```
# rk_mpi_ao_test -i /oem/cap_out.pcm --input_ch 2 --input_rate 44100 --device_ra
te 44100 --device_ch 2 --sound_card_name hw:0,0
cmd parse result:
input  file name     : /oem/cap_out.pcm
output file name     : (null)
loop count          : 1
channel number       : 1
open sound rate      : 44100
open sound channel    : 2
input stream rate     : 44100
input channel        : 2
bit_width           : 16
period_count         : 4
period_size         : 4096
sound card name      : hw:0,0
device id           : 0
set volume curve     : 0
set volume          : 100
set mute            : 0
set track_mode       : 0
get volume          : 0
get mute            : 0
get track_mode       : 0
query stat          : 0
pause and resume chn  : 0
save file           : 0
query save file stat  : 0
clear buf           : 0
get attribute        : 0
clear attribute      : 0
set loopback mode     : 0
vqe enable          : 0
rockit_load start
v4l2_tx probe successv4l2_rx_probe success
rockit_load end
rockit log path (null), log_size = 0
log_file = (nil)
(null)          10:49:40-968 {log_level_init    :209}

 please use echo name=level > /tmp/rt_log_level set log level
       name: all cmpi mb sys vdec venc rgn vpss vgs tde avs wbc vo vi ai ao aenc adec
       log_level: 0 1 2 3 4 5 6
```

6.5 Ethernet

The 100M Ethernet and 1000M Ethernet interfaces on the SBC1126B are multiplexed and cannot be used at the same time. When switching between them, the corresponding firmware image must be updated in software, and the hardware selection must also be changed using the DIP switch on the development board.



DTS Configuration Switching Instructions

File path: *sysdrv\source\kernel\arch\arm64\boot\dts\rockchip\boardcon-sbc1126b.dtsi*

When using 100M Ethernet:

```
// #include "boardcon-sbc1126b-gephy.dtsi" //1000M
#include "boardcon-sbc1126b-fephy.dtsi"   //100M
```

When using 1000M Ethernet:

```
#include "boardcon-sbc1126b-gephy.dtsi"    //1000M
//#include "boardcon-sbc1126b-fephy.dtsi"  //100M
```

6.5.1 100M Ethernet

Step 1: Connect the network cable to the 100M Ethernet port, and set the DIP switch to the 100M position.

According to the log, it can be seen that the 100M Ethernet recognition is successful.

```
# dmesg | grep -Ei "link|100Mbps"
[    0.268284] NET: Registered PF_NETLINK/PF_ROUTE protocol family
[    0.712381] mpp_rkvdec2 22140100.rkvdec: link mode probe finish
[    0.719821] can: netlink gateway - max_hops=1
[    1.368230] rk_gmac-dwmac 21c70000.ethernet eth0: configuring for phy/rmii link mode
[    2.726968] dw-mipi-dsi-rockchip 22120000.dsi: [drm:dw_mipi_dsi_bridge_atomic_enable] final DSI-Link
bandwidth: 468 x 4 Mbps
[   12.969534] rk_gmac-dwmac 21c70000.ethernet eth0: configuring for phy/rmii link mode
[  162.477091] rk_gmac-dwmac 21c70000.ethernet eth0: Link is Up - 100Mbps/Full - flow control rx/tx
[  162.477120] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
```

Step 2: Check the network interface information.

Execute the following command:

```
# ifconfig eth0
```

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 0A:E7:14:87:42:44
          inet addr:192.168.0.221  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::8e7:14ff:fe87:4244/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:759 errors:0 dropped:92 overruns:0 frame:0
          TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:63001 (61.5 KiB)  TX bytes:2428 (2.3 KiB)
          Interrupt:91
```

Step 3: Test the network connection.

Execute the following command:

```
# ping -I eth0 www.armdesigner.com
```

```
# ping -I eth0 www.armdesigner.com
PING www.armdesigner.com (13.249.126.87): 56 data bytes
64 bytes from 13.249.126.87: seq=0 ttl=246 time=169.120 ms
64 bytes from 13.249.126.87: seq=1 ttl=246 time=169.190 ms
64 bytes from 13.249.126.87: seq=2 ttl=246 time=168.847 ms
64 bytes from 13.249.126.87: seq=3 ttl=246 time=170.724 ms
64 bytes from 13.249.126.87: seq=4 ttl=246 time=169.255 ms
64 bytes from 13.249.126.87: seq=5 ttl=246 time=169.033 ms
^C
--- www.armdesigner.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 168.847/169.361/170.724 ms
```

6.5.2 1000M Ethernet

Step 1: Connect the network cable to the 1000M Ethernet port, and set the DIP switch to the 1000M position.

According to the log, it can be seen that the 1000M Ethernet recognition is successful.

```
# dmesg | grep -Ei "link|1Gbps"
[    0.266194] NET: Registered PF_NETLINK/PF_ROUTE protocol family
[    0.806565] mpp_rkvdec2 22140100.rkvdec: link mode probe finish
[    0.814165] can: netlink gateway - max_hops=1
[    1.376796] rk_gmac-dwmac 21c70000.ethernet eth0: configuring for phy/rgmii link mode
[    2.772943] dw-mipi-dsi-rockchip 22120000.dsi: [drm:dw_mipi_dsi_bridge_atomic_enable] final DSI-Link
bandwidth: 468 x 4 Mbps
[   13.856116] rk_gmac-dwmac 21c70000.ethernet eth0: configuring for phy/rgmii link mode
[  158.253401] rk_gmac-dwmac 21c70000.ethernet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
[  158.253500] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
```

Step 2: Check the network interface information.

Execute the following command:

```
# ifconfig eth0
```

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 0A:E7:14:87:42:44
          inet addr:192.168.0.221  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::8e7:14ff:fe87:4244/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3627 errors:0 dropped:431 overruns:0 frame:0
          TX packets:15 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:354016 (345.7 KiB)  TX bytes:1690 (1.6 KiB)
          Interrupt:91
```

Step 3: Test the network connection.

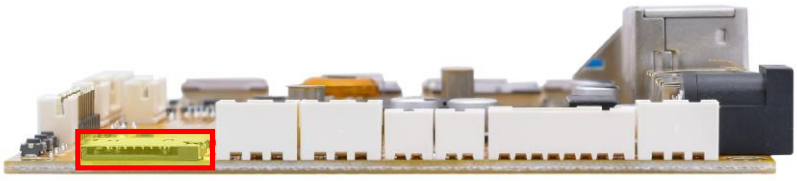
Execute the following command:

```
# ping -I eth0 www.armdesigner.com
```

```
# ping -I eth0 www.armdesigner.com
PING www.armdesigner.com (13.249.126.17): 56 data bytes
64 bytes from 13.249.126.17: seq=0 ttl=246 time=159.443 ms
64 bytes from 13.249.126.17: seq=1 ttl=246 time=159.864 ms
64 bytes from 13.249.126.17: seq=2 ttl=246 time=159.315 ms
64 bytes from 13.249.126.17: seq=3 ttl=246 time=159.512 ms
64 bytes from 13.249.126.17: seq=4 ttl=246 time=159.779 ms
64 bytes from 13.249.126.17: seq=5 ttl=246 time=159.436 ms
^C
--- www.armdesigner.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 159.315/159.558/159.864 ms
```

6.6 SD Card

Step 1: Insert the micro SD card into the card slot.



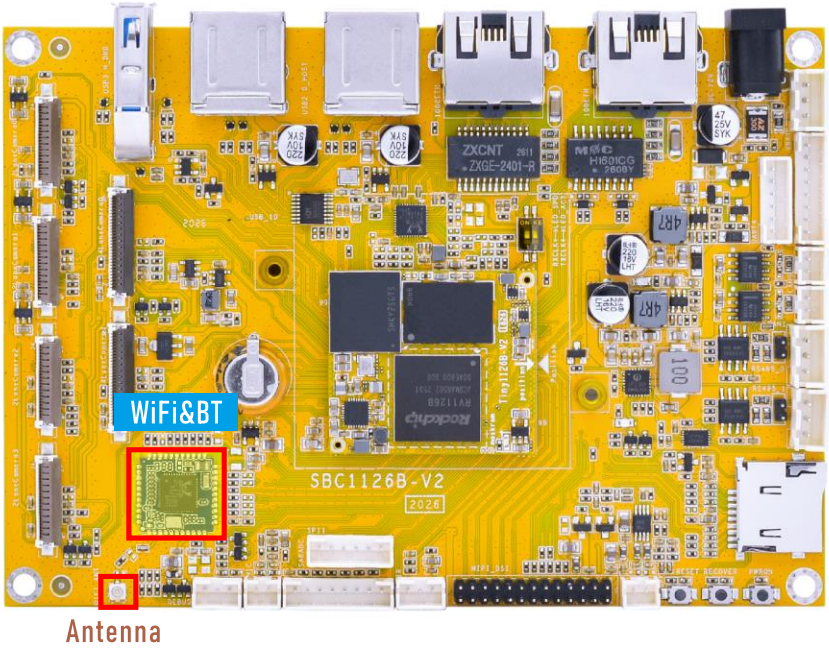
After the SD card is recognized, it will be mounted automatically. Run the following command to check the mount path of the SD card.

```
# df -h
```

```
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                47.5M     47.5M         0 100% /
devtmpfs                 60.4M         0     60.4M   0% /dev
tmpfs                   966.9M         0    966.9M   0% /dev/shm
tmpfs                   966.9M     16.0K    966.9M   0% /tmp
tmpfs                   966.9M    664.0K    966.3M   0% /run
/dev/block/by-name/oem
                        237.9M     55.3M    167.2M  25% /oem
/dev/block/by-name/userdata
                          5.8G     49.5M      5.6G   1% /userdata
/dev/sda1                58.0G     37.5G     20.4G  65% /tmp/sda1
/dev/mmcblk1p1           30.0G      4.5G     25.4G  15% /mnt/sdcard
```

6.7 WiFi & Bluetooth

To use the WiFi and Bluetooth functions properly, make sure the antenna is connected.



6.7.1 WiFi

Step 1: Check the Wi-Fi device information.

Execute the following command:

```
# ifconfig wlan0
```

```
# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 6C:D5:52:A2:3A:35
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Step 2: Scan for available WiFi hotspots.

Execute the following command:

```
# wpa_cli -i wlan0 scan
# wpa_cli -i wlan0 scan_results
```

```
# wpa_cli -i wlan0 scan
OK
# [network.c][rk_network_get_cable_state]:read Cable state
[network.c][rk_network_get_cable_state]:
[wlan0] link down

# wpa_cli -i wlan0 scan_results
bssid / frequency / signal level / flags / ssid
b4:f1:8c:6d:d1:29       2462    -91     [WPA2-PSK-CCMP][ESS]
b4:f1:8c:fd:d1:29       2462    -59     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    Boardcon_Wi-Fi5
b4:f1:8c:6d:d1:25       2462    -58     [ESS]
c8:3a:35:51:1e:98       2442    -45     [WPA-PSK-CCMP][ESS]     Tenda_511E98
b4:f1:8c:6d:d1:24       2462    -58     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    Boardcon
9c:a6:15:10:da:c5       2462    -65     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    TP
44:6a:2e:20:83:60       2462    -61     [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]      ydsz
7c:7d:21:2c:95:34       2442    -69     [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]      Ly-2.4g
9c:a6:15:10:e5:57       2437    -74     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    TP
38:20:28:50:0f:81       2412    -70     [WPA2-PSK-CCMP][ESS]    Casvi
38:20:28:50:1e:a1       2437    -72     [WPA2-PSK-CCMP][ESS]    Casvi
b2:22:7a:5a:b6:4a       2462    -72     [WPA2-PSK-CCMP][ESS]    DIRECT-4A-HP Laser 136w
42:bd:f6:cf:b7:70       2412    -81     [WPA2-PSK-CCMP][ESS]    HUAWEI-FK1E1D
a0:10:77:12:57:1e       2412    -83     [WPA2-PSK-CCMP][ESS]    ChinaNet-dEUc
```

Step 3: Connect to the WiFi hotspot.

Execute the following command:

```
# rkwifi_server connect ssid password
```

```
# rkwifi_server connect Boardcon Boardcon43435656
rkwifi cmd: connect Boardcon Boardcon43435656
OK
cmd_line: connect Boardcon Boardcon43435656 [33]
rk_cmd_parse connect Boardcon Boardcon43435656
rk_tokenize_cmd connect Boardcon Boardcon43435656
[connect 3]: connect Boardcon
rk_wifi_connect: ssid: Boardcon, psk: Boardcon43435656
[1783521888:843183][src/Rk_wifi.c][wpa_wifi_connect1][l:1851],[wpa_wifi_connect1] ssid: Boardcon, psk:
Boardcon43435656

[1783521888:843229][src/Rk_wifi.c][_RK_wifi_running_getConnectionInfo][l:710],_RK_wifi_running_getConnecti
onInfo: enter
```

Step 4: Check the network interface status.

Execute the following command:

```
# ifconfig wlan0
```

```
# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 6C:D5:52:A2:3A:35
          inet addr:192.168.0.158  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::6ed5:52ff:fea2:3a35/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:510 errors:0 dropped:71 overruns:0 frame:0
          TX packets:39 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:58562 (57.1 KiB)  TX bytes:8462 (8.2 KiB)
```

Step 5: Test the WiFi network connection.

Execute the following command:

```
# ping -I wlan0 www.armdesigner.com
```

```
# ping -I wlan0 www.armdesigner.com
PING www.armdesigner.com (13.249.126.124): 56 data bytes
64 bytes from 13.249.126.124: seq=0 ttl=246 time=179.223 ms
64 bytes from 13.249.126.124: seq=1 ttl=246 time=174.994 ms
64 bytes from 13.249.126.124: seq=2 ttl=246 time=351.503 ms
64 bytes from 13.249.126.124: seq=3 ttl=246 time=180.463 ms
64 bytes from 13.249.126.124: seq=4 ttl=246 time=178.491 ms
64 bytes from 13.249.126.124: seq=5 ttl=246 time=188.076 ms
^C
--- www.armdesigner.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 174.994/208.791/351.503 ms
```

6.7.2 Bluetooth

Bluetooth is configured to work as a Bluetooth speaker by default.

Step 1: Start the BlueALSA A2DP sink service.

Execute the following command:

```
# bluealsa -p a2dp-sink &
```

```
# bluealsa -p a2dp-sink &
# bluealsa: W: Couldn't create storage directory: No such file or directory
bluealsa: E: Couldn't acquire D-Bus name. Please check D-Bus configuration. Requested name: org.bluealsa

[1]+  Done(1)                 bluealsa -p a2dp-sink
```

Step 2: Enter the Bluetooth control tool.

Execute the following command:

```
# bluetoothctl
```

```
# bluetoothctl
Waiting to connect to bluetoothd...Waiting to connect to bluetoothd...[bluetooth]#
hci0 new_settings: powered bondable ssp br/edr le secure-conn
[bluetooth]# [bluetooth]#                     Agent registered
[bluetooth]# [bluetooth]#                     [CHG] Controller 6C:D5:52:A2:3A:36 Pairable: yes
[bluetooth]# [bluetooth]#
```

Step 3: Enable Bluetooth and set it to discoverable mode.

Execute the following commands in bluetoothctl:

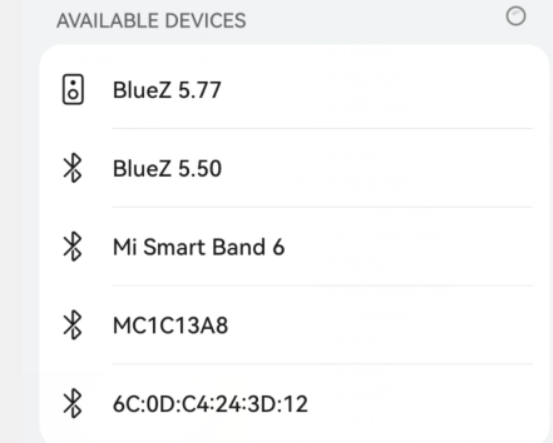
```
[bluetooth]# power on
[bluetooth]# discoverable on
```

```
[bluetooth]# [bluetooth]# power on
[bluetooth]# [bluetooth]#                          Changing power on succeeded
[bluetooth]# [bluetooth]# discoverable on
[bluetooth]# [bluetooth]#                          hci0 new_settings: powered connectable bondable ssp br/edr
le secure-conn
[bluetooth]# [bluetooth]#                          hci0 new_settings: powered connectable discoverable
bondable ssp br/edr le secure-conn
[bluetooth]# [bluetooth]#                          Changing discoverable on succeeded
[bluetooth]# [bluetooth]#                          [CHG] Controller 6C:D5:52:A2:3A:36 Discoverable: yes
[bluetooth]# [bluetooth]#
```

After this step, the Bluetooth device can be discovered by the phone.

Step 4: Pair and connect the Bluetooth device.

On the phone, find the Bluetooth device named "**BlueZ 5.77**" in the Bluetooth device list, then tap it to pair and connect.



If a pairing confirmation message appears, confirm it on both the phone and the board.

```
[bluetooth]# [bluetooth]#                          hci0 new_settings: powered connectable bondable ssp br/edr
le secure-conn
[bluetooth]# [bluetooth]#                          hci0 new_settings: powered connectable discoverable
bondable ssp br/edr le secure-conn
[bluetooth]# [bluetooth]#                          Changing discoverable on succeeded
[bluetooth]# [bluetooth]#                          [CHG] Controller 6C:D5:52:A2:3A:36 Discoverable: yes
[bluetooth]# [bluetooth]#                          hci0 A8:35:12:9A:EB:4D type BR/EDR connected eir_len 11
[bluetooth]# [bluetooth]#                          [NEW] Device A8:35:12:9A:EB:4D liuy
[bluetooth]# [bluetooth]# [liuy]#                                Request confirmation
[liuy]# [liuy]# [agent] Confirm passkey 303999 (yes/no): yes
[liuy]# [liuy]#                  hci0 new_link_key A8:35:12:9A:EB:4D type 0x05 pin_len 0 store_hint 1
[liuy]# [liuy]#                  hci0 device_flags_changed: A8:35:12:9A:EB:4D (BR/EDR)
[liuy]# [liuy]#                       supp: 0x00000000  curr: 0x00000000
[liuy]# [liuy]#                  [CHG] Device A8:35:12:9A:EB:4D ServicesResolved: yes
[liuy]# [liuy]#                  [CHG] Device A8:35:12:9A:EB:4D Paired: yes
[liuy]# [liuy]#                  Authorize service
[liuy]# [liuy]# [agent] Authorize service 0000110d-0000-1000-8000-00805f9b34fb (yes/no): yes
               [CHG] Device A8:35:12:9A:EB:4D UUIDs: 8ce255c0-200a-11e0-ac64-0800200c9a66
[liuy]# [liuy]#
```

Check the log output to confirm the MAC address. In this example, the MAC address of the mobile phone is **A8:35:12:9A:EB:4D**.

Step 5: Exit bluetoothctl.

Execute the following command:

```
[bluetooth]# exit
```

Step 6: Start Bluetooth audio playback.

Execute the following command:

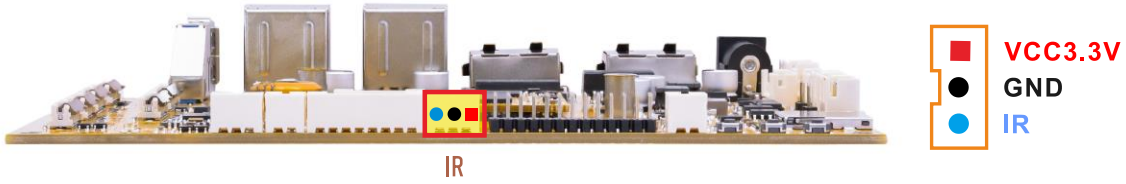
```
# bluealsa-aplay A8:35:12:9A:EB:4D &
```

```
[liuy]# exit
[liuy]# [liuy]#                 #
# bluealsa-aplay A8:35:12:9A:EB:4D &
# bluealsa-aplay: W: Couldn't open ALSA mixer: Mixer element not found
#
```

After the Bluetooth connection is established, play audio on the phone. The audio should be output from the board.

6.8 IR

Step 1: Connect the IR receiver.



Step 2: Enable IR debug logs.

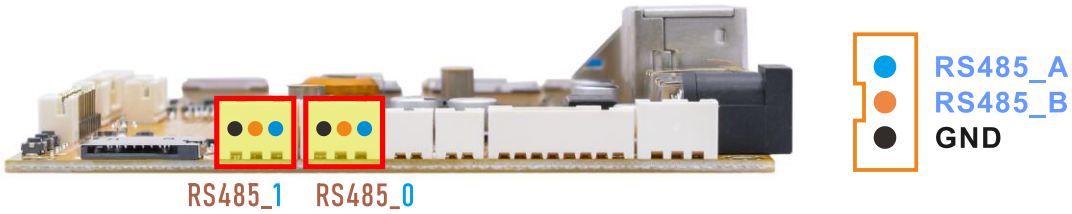
Execute the following command:

```
# echo 1 > /sys/module/rockchip_pwm_remotectl/parameters/code_print
```

Step 3: Point the remote control at the IR receiver and press a button. The corresponding key value will be printed in the log.

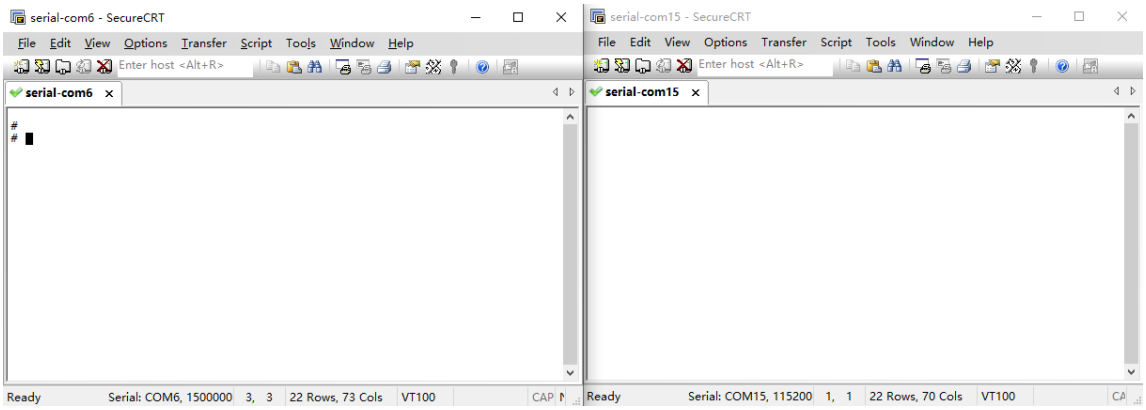
```
# echo 1 > /sys/module/rockchip_pwm_remotectl/parameters/code_print
# dmesg | grep -Ei "USERCODE|RMC_GETDATA"
[   46.215271] USERCODE=0x1818
[   46.242508] RMC_GETDATA=e6
[   46.629794] USERCODE=0x1818
[   46.656951] RMC_GETDATA=9a
[   47.177519] USERCODE=0x1818
[   47.204731] RMC_GETDATA=9b
[   48.986199] USERCODE=0x1818
[   49.013275] RMC_GETDATA=99
[   49.326217] USERCODE=0x1818
[   49.353353] RMC_GETDATA=99
[   49.667350] USERCODE=0x1818
[   49.694497] RMC_GETDATA=9a
[   49.961815] USERCODE=0x1818
[   49.988972] RMC_GETDATA=98
[   51.210911] USERCODE=0x1818
```

6.9 RS485



Step 1: Connect the RS485 test tool to the development board as shown in the diagram.

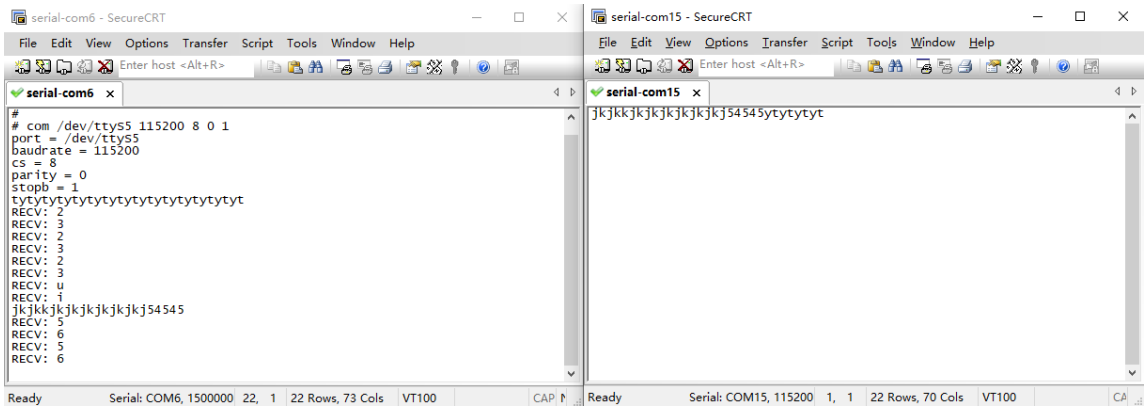
Step 2: Open two serial terminal windows. Set the debug serial port baud rate to 1500000, and set the RS485 serial port baud rate to 115200.



Step 3: Execute the following command on the board to test the RS485 data transmission and reception.

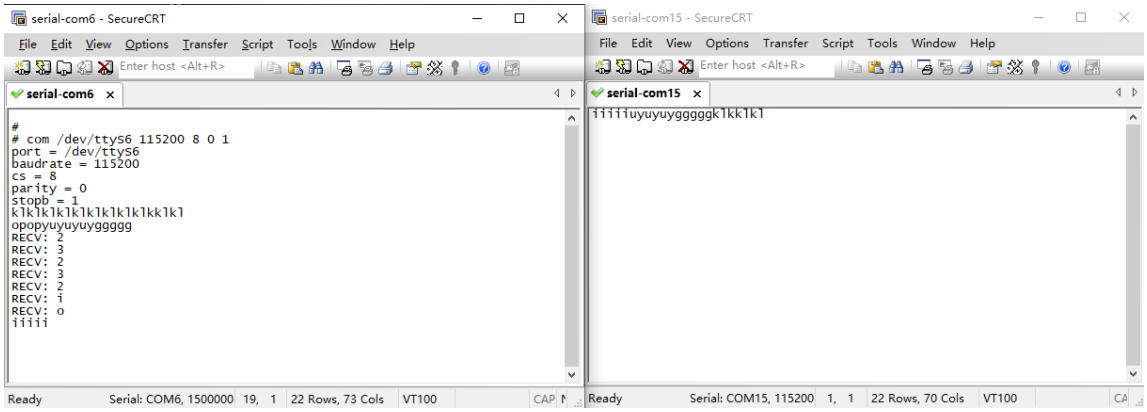
RS485_0:

```
# com /dev/ttyS5 115200 8 0 1
```



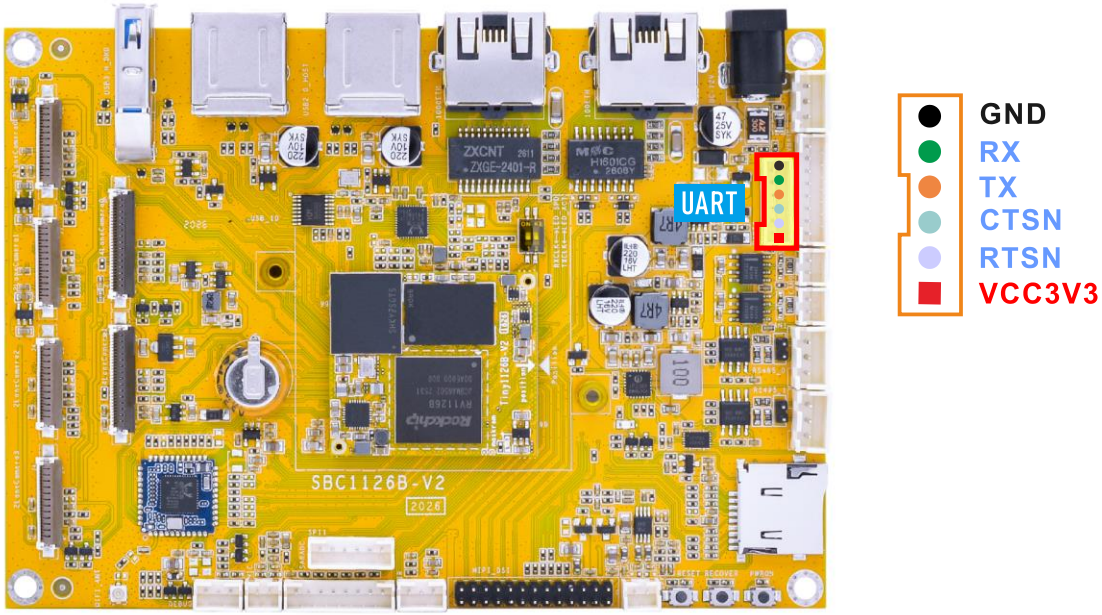
RS485_1:

```
# com /dev/ttyS6 115200 8 0 1
```



6.10 UART

Step 1: Connect the RX and TX pins of the UART interface together to create a loopback connection.



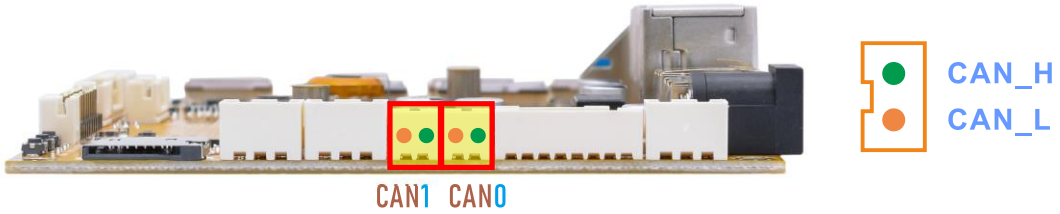
Step 2: Test UART loopback communication.

Execute the following command:

```
# com /dev/ttyS4 115200 8 0 1
```

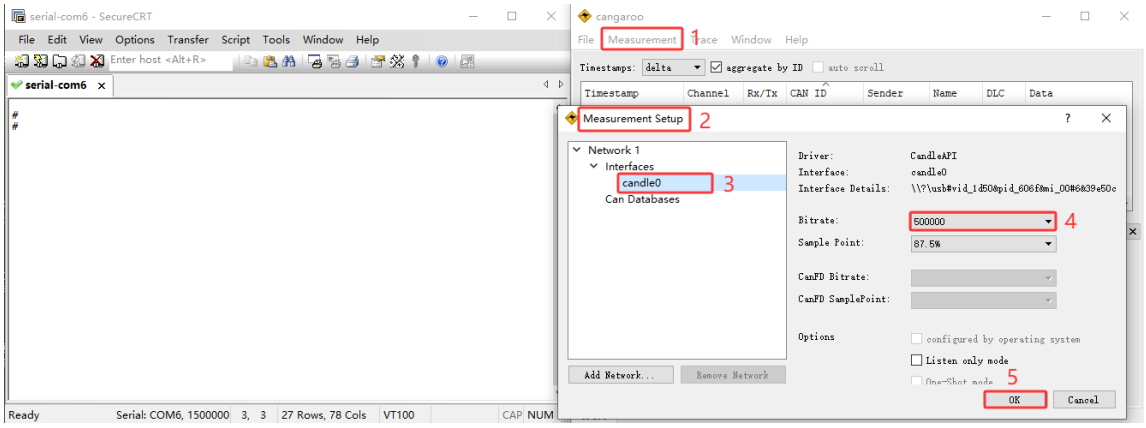
```
# com /dev/ttyS4 115200 8 0 1
port = /dev/ttyS4
baudrate = 115200
cs = 8
parity = 0
stopb = 1
1212121212121
RECV: 1212121212121
opopopopopo
RECV: opopopopopo
yuyuyuyuyu
RECV: yuyuyuyuyu
kklklklkl
RECV: kklklklkl
```

6.11 CAN



Step 1: Connect the CAN test tool to the board as shown in the diagram below.

Step 2: Open the CAN test software and set the baud rate to 500000.



6.11.1 CAN0 Test

Step 1: Configure and enable the CAN0 interface on the board with a bitrate of 500000.

Execute the following command:

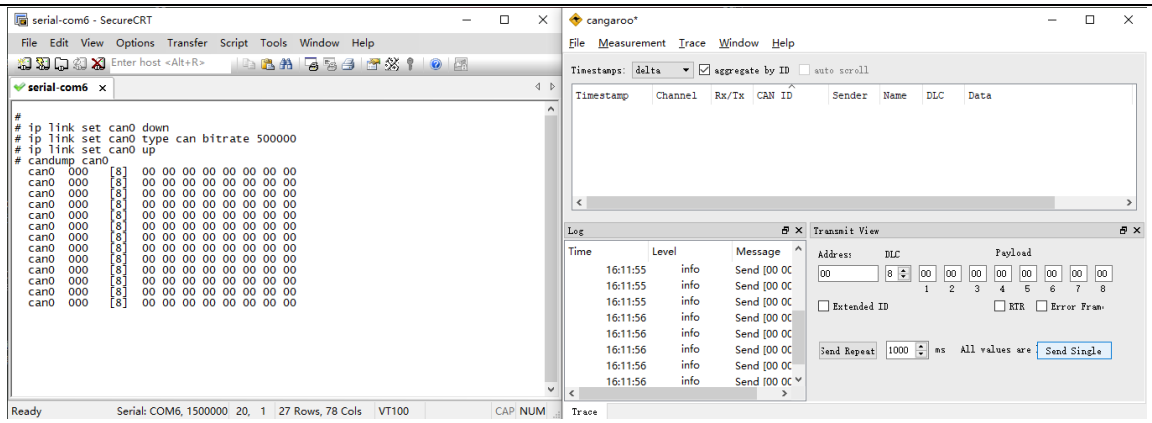
```
# ip link set can0 down
# ip link set can0 type can bitrate 500000
# ip link set can0 up
```

Step 2: Test CAN0 receiving.

Configure the CAN test tool as the sender, and configure CAN0 on the board as the receiver.

Execute the following command:

```
# candump can0
```

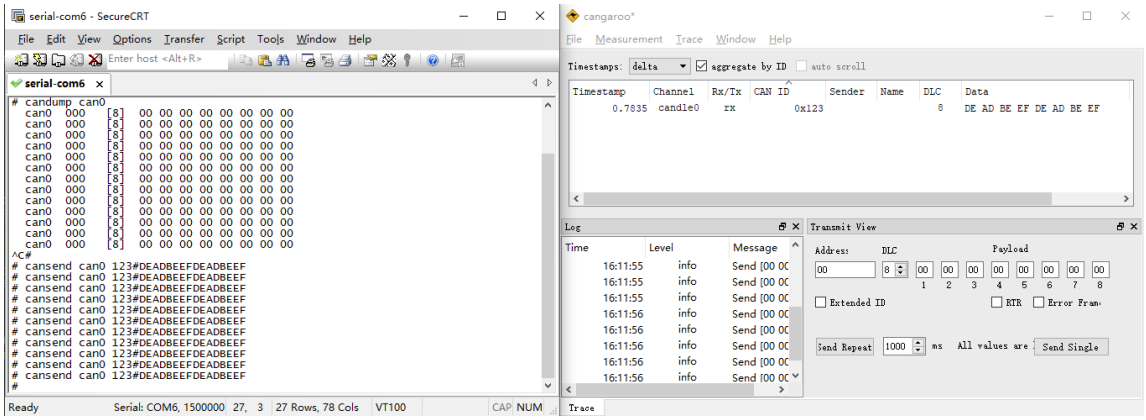


Step 3: Test CAN0 sending.

Configure the CAN test tool as the receiver, and configure CAN0 on the board as the sender.

Execute the following command:

```
# cansend can0 123#DEADBEEFDEADBEEF
```



6.11.2 CAN1 Test

Step 1: Configure and enable CAN1 with a bitrate of 500000.

Execute the following command:

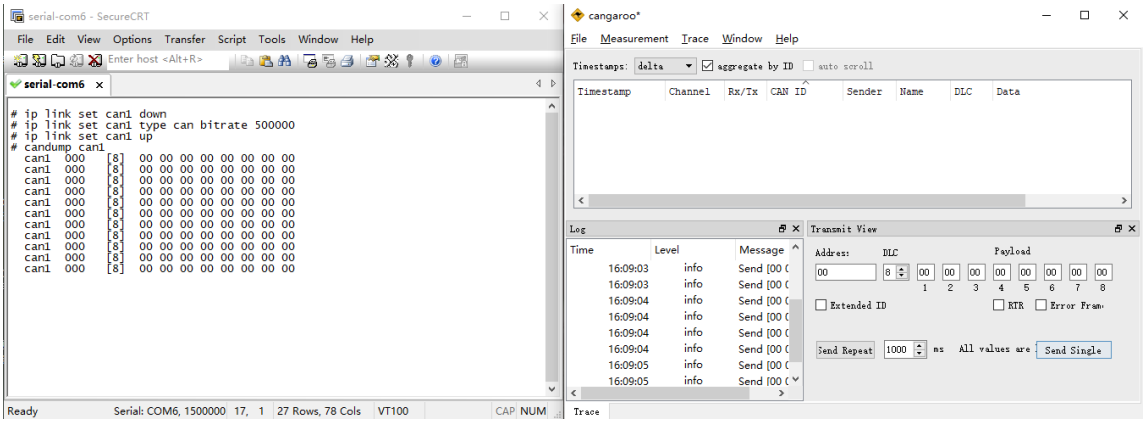
```
# ip link set can1 down
# ip link set can1 type can bitrate 500000
# ip link set can1 up
```

Step 2: Test CAN1 receiving.

Configure the CAN test tool as the sender, and configure CAN1 on the board as the receiver.

Execute the following command:

```
# candump can1
```

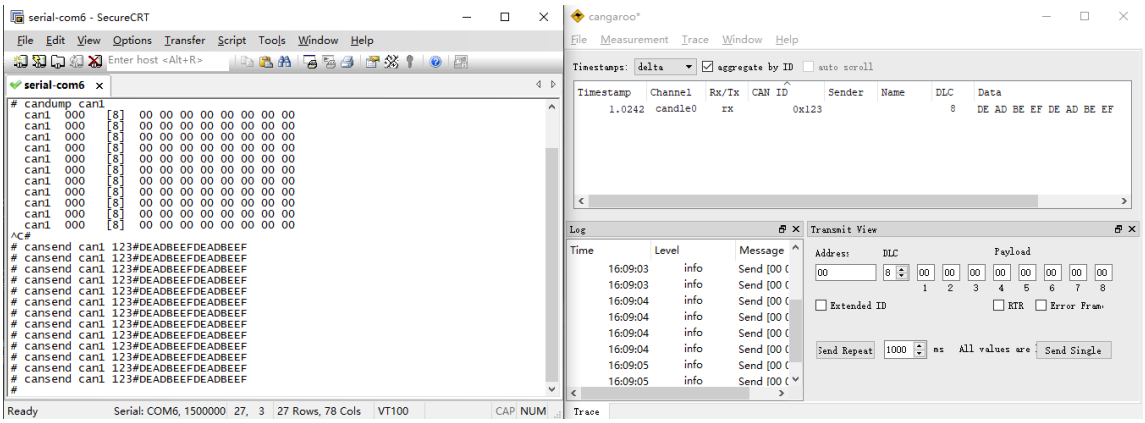


Step 3: Test CAN1 sending.

Configure the CAN test tool as the receiver, and configure CAN1 on the board as the sender.

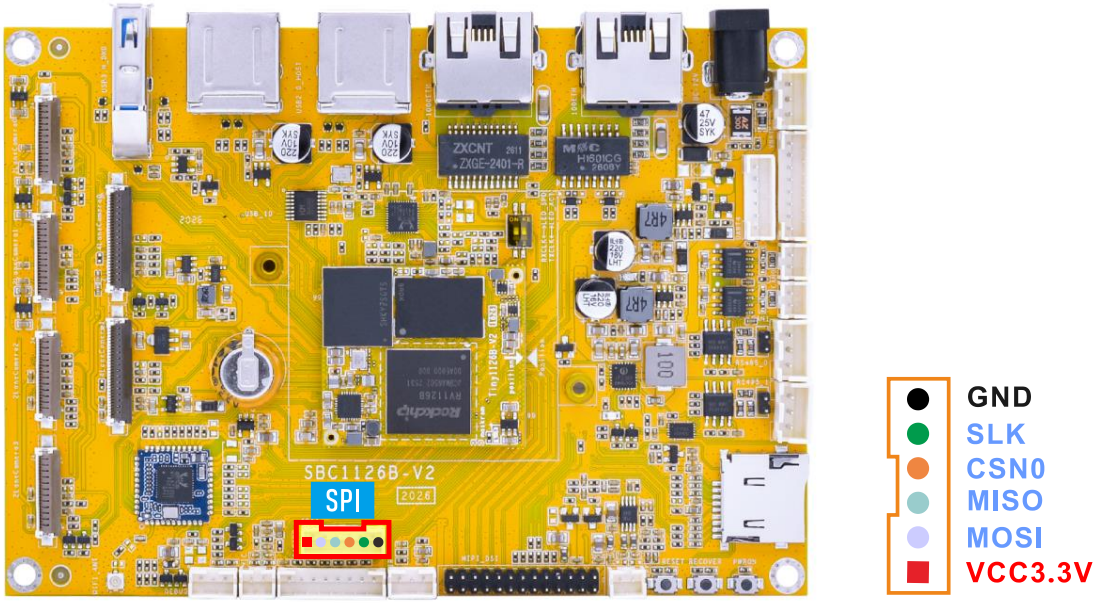
Execute the following command:

```
# cansend can1 123#DEADBEEFDEADBEEF
```



6.12 SPI

Step 1: Short the **MOSI** and **MISO** pins of the SPI interface to create a loopback connection.



Step 2: Execute the following command to perform the SPI loopback test:

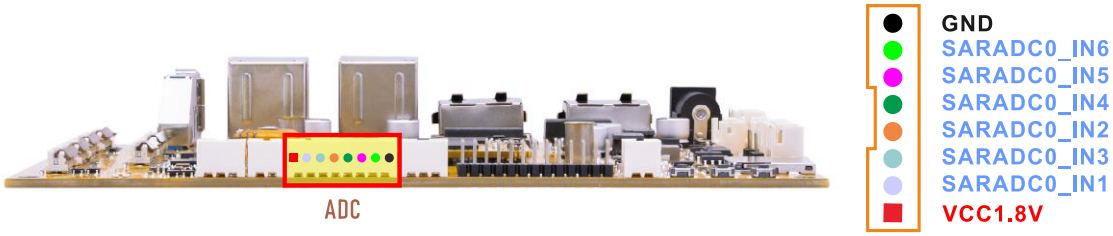
```
# test_spi 1 0
```

```
# test_spi 1 0
device= /dev/spidev1.0
spi mode: 0
bits per word: 8
max speed: 5000000 Hz (5000 KHz)

FF FF FF FF FF FF
40 00 00 00 00 95
FF FF FF FF FF FF
FF FF FF FF FF FF
FF FF FF FF FF FF
DE AD BE EF BA AD
F0 0D
```

6.13 ADC

The board provides six ADC input interfaces: ADC1, ADC2, ADC3, ADC4, ADC5, and ADC6.



This test is used to check whether the ADC raw value changes according to the input

voltage.

Step 1: Apply a voltage level to the ADC input pin.

Connect the ADC pin to GND for a low level, or connect it to 1.8 V for a high level.

Step 2: Read the ADC raw value.

Example for **ADC1**:

```
# cat /sys/bus/iio/devices/iio:\device0/in_voltage1_raw
```

```
# cat /sys/bus/iio/devices/iio:\device0/in_voltage1_raw
8
# cat /sys/bus/iio/devices/iio:\device0/in_voltage1_raw
8191
```

For other ADC read commands, refer to the ADC read command description below:

```
Single-channel read commands:
ADC1 : cat /sys/bus/iio/devices/iio:\device0/in_voltage1_raw
ADC2 : cat /sys/bus/iio/devices/iio:\device0/in_voltage2_raw
ADC3 : cat /sys/bus/iio/devices/iio:\device0/in_voltage3_raw
ADC4 : cat /sys/bus/iio/devices/iio:\device0/in_voltage4_raw
ADC5 : cat /sys/bus/iio/devices/iio:\device0/in_voltage5_raw
ADC6 : cat /sys/bus/iio/devices/iio:\device0/in_voltage6_raw

To read all ADC1–ADC6 raw values, execute the following command:
for i in 1 2 3 4 5 6; do
    echo -n "ADC$i: "
    cat /sys/bus/iio/devices/iio:\device0/in_voltage${i}_raw
done
```

6.14 RTC

The RTC on the SBC1126B is powered by a supercapacitor.

Step 1: Set the system time manually, for example:

```
# date -s "2026-07-08 16:22:00"
```

Note: If the network is connected, the system time may be synchronized automatically to the current UTC time.

Step 2: Write the system time to the hardware clock:

```
# hwclock -w
```

Step 3: Read the current hardware clock time:

```
# hwclock
```

```
# date -s "2026-07-08 16:22:00"
Wed Jul  8 16:22:00 UTC 2026
# hwclock -w
# hwclock
Wed Jul  8 16:22:06 2026  0.000000 seconds
# hwclock
Wed Jul  8 16:22:41 2026  0.000000 seconds
# hwclock
Wed Jul  8 16:23:04 2026  0.000000 seconds
```

Step 4: Power off the board and disconnect the main power supply. Wait for a period of time, then power on the board again.

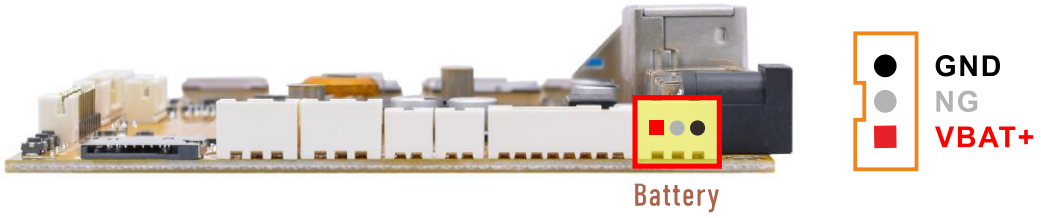
Step 5: Read the hardware clock time again:

```
# hwclock
Wed Jul  8 16:31:18 2026  0.000000 seconds
# hwclock
Wed Jul  8 16:31:33 2026  0.000000 seconds
# hwclock
Wed Jul  8 16:32:01 2026  0.000000 seconds
```

If the RTC is working properly, the hardware clock time should be retained and continue running after power-off.

6.15 Battery supply

Step 1: Connect the 7.4V Li-ion battery, then press and hold the power button to turn on the board.



Step 2: Read the current battery capacity percentage.

```
# cat /sys/class/power_supply/cw221X-bat/capacity
```

```
# cat /sys/class/power_supply/cw221X-bat/capacity
36
```

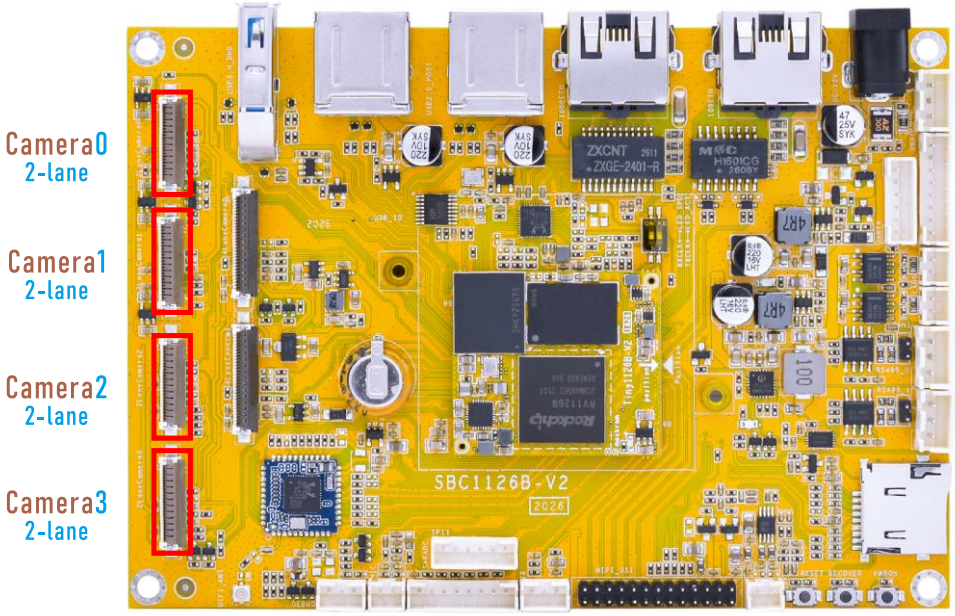
Step 3: Read the current battery voltage.

```
# cat /sys/class/power_supply/cw221X-bat/voltage_now
```

```
# cat /sys/class/power_supply/cw221X-bat/voltage_now
3427000
```

6.16 Camera

Step 1: Connect the camera modules, then power on the board.



After the system boots, the stitched preview image from all four camera channels is displayed by default.

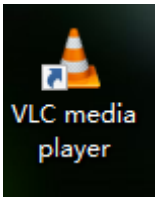
Step 2: Check the network status and obtain the Ethernet IP address of the board.

Execute the following command:

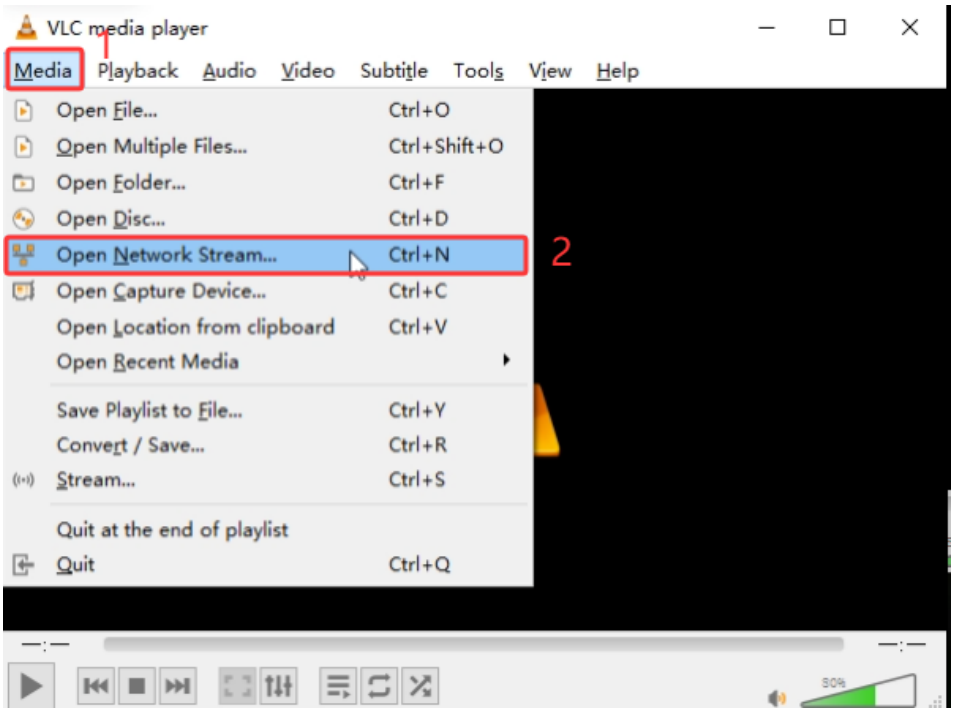
```
# ifconfig eth0
```

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr C6:72:C0:7F:A8:3A
          inet addr:192.168.0.66  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::c472:c0ff:fe7f:a83a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:294 errors:0 dropped:37 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:26761 (26.1 KiB)  TX bytes:3462 (3.3 KiB)
          Interrupt:91
```

Step 3: Open VLC media player on the PC.

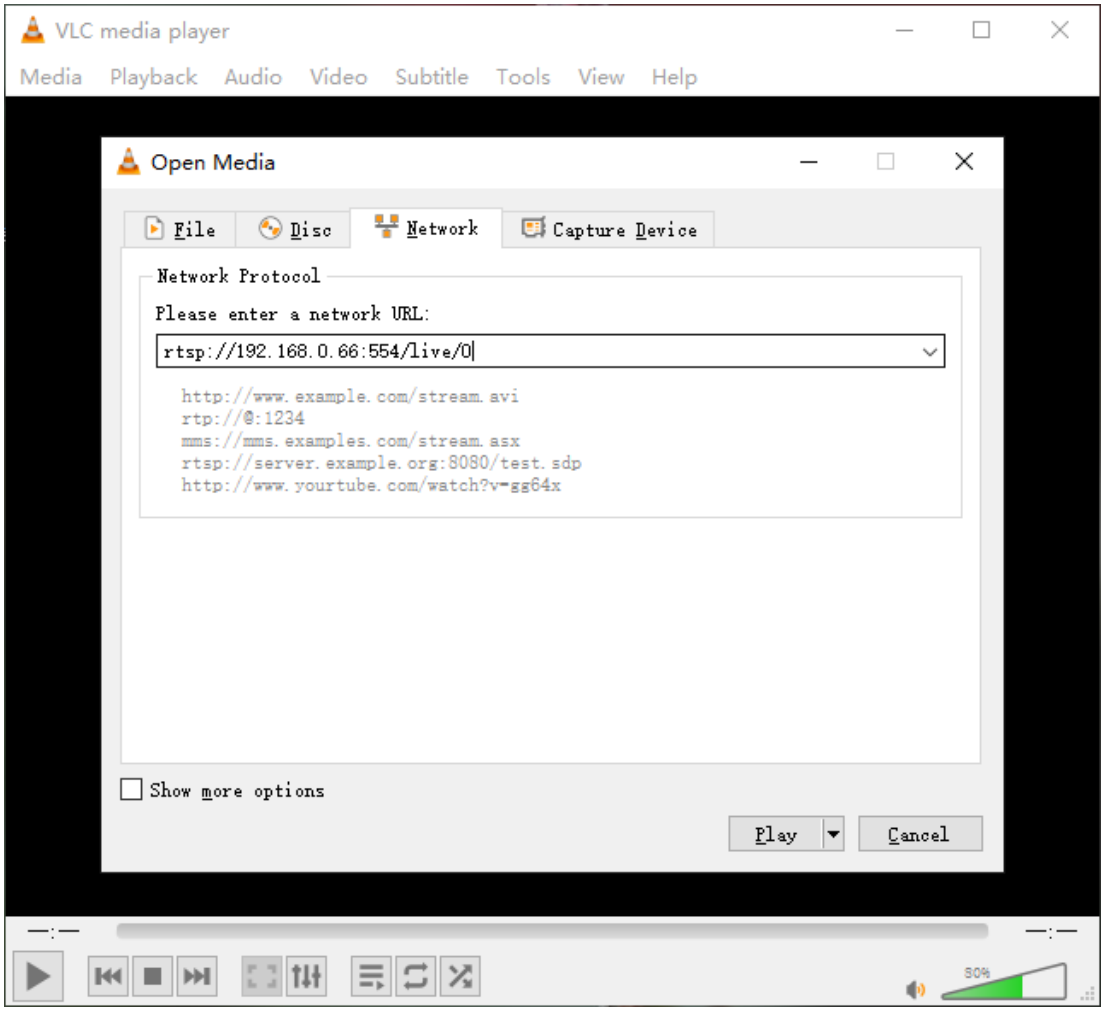


Step 4: Select **Media** -> **Open Network Stream...**.



Step 5: Enter the following RTSP URL to preview the camera stream:

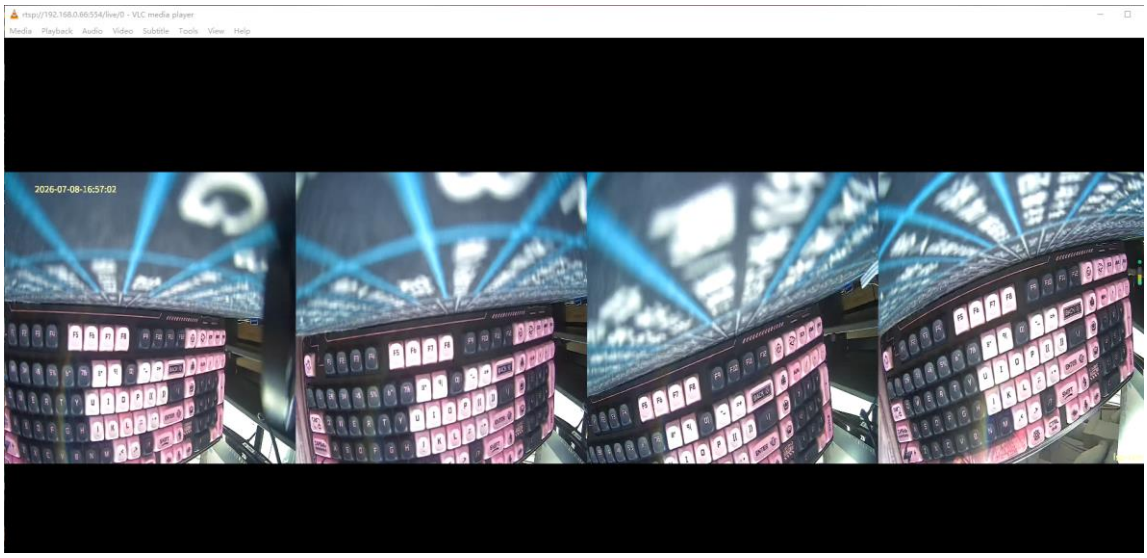
```
rtsp://192.168.0.66:554/live/0
```



Note:

Replace 192.168.0.66 with the actual Ethernet IP address of the board.

Step 6: Check the four-camera preview result.



7.17 Video Playback

Video playback can be tested using the simple_vdec_bind_vo command.

Execute the following command:

```
# simple_vdec_bind_vo -i /tmp/sda1/testh264.h264
```

```
# simple_vdec_bind_vo -i /tmp/sda1/testh264.h264
param list: w = 1920, h = 1080, i = /tmp/sda1/testh264.h264, b = 1
rockit_load start
v4l2_tx probe successv4l2_rx_probe success
rockit_load end
rockit log path (null), log_size = 0
log_file = (nil)
(null)          17:33:44-604 {log_level_init    :209}

 please use echo name=level > /tmp/rt_log_level set log level
       name: all cmpi mb sys vdec venc rgn vpss vgs tde avs wbc vo vi ai ao aenc adec
       log_level: 0 1 2 3 4 5 6

rockit default level 4, can use export rt_log_level=x, x=0,1,2,3,4,5,6 change
(null)          17:33:44-605 {read_log_level    :083} text is all=4
(null)          17:33:44-605 {read_log_level    :085} module is all, log_level is 4
(null)          17:33:44-605 {dump_version      :060} ----------------------------------------------------
-----
(null)           17:33:44-605 {dump_version      :061} rockit version: git-b85f3c523 Mon Jan 26 09:47:22
2026 +0800
(null)          17:33:44-605 {dump_version      :062} rockit building: built-Chu 2026-01-27 09:45:41
(null)          17:33:44-605 {dump_version      :063} ----------------------------------------------------
-----
(null)          17:33:44-605 {monitor_log_level :141} #Start monitor_log_level thread, arg:(nil)
vsys dev open 4
load library(librga.so) in releative path
mpp[1041]: mpp_info: mpp version: d7dd00e4f author: Herman Chen  2026-02-13 fix[h265d]: Fix invalid hvcc
data crash on seek
mpp[1041]: hal_264d_com: control info: fmt 0, w 1920, h 1080
mpp[1041]: mpp_dec: mpp_dec_proc_cfg found MPP_DEC_SET_FRAME_INFO fmt 0
mpp[1041]: mpp_buf_slot: mismatch h_stride_by_pixel 1984 - 1920
```

Command explanation:

- simple_vdec_bind_vo: Video decoding and display test tool.
- -i: Specifies the input video file.
- /tmp/sda1/testh264.h264: Input H.264 file path.

Note:

This command supports H.264 elementary stream files, such as .h264 files. Standard MP4 files are not supported.