EM1126B-P Linux6.1 IPC User Manual

V1.0

Boardcon Embedded Design

Overview

This document applies only to the EM1126B-P 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
Mini1126B-P V5 + EM1126B-P_V4	Y

Revision History

Version	Date	Author	Revision History
V1.0	2026-07-02	Liu Yuan	Initial version

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

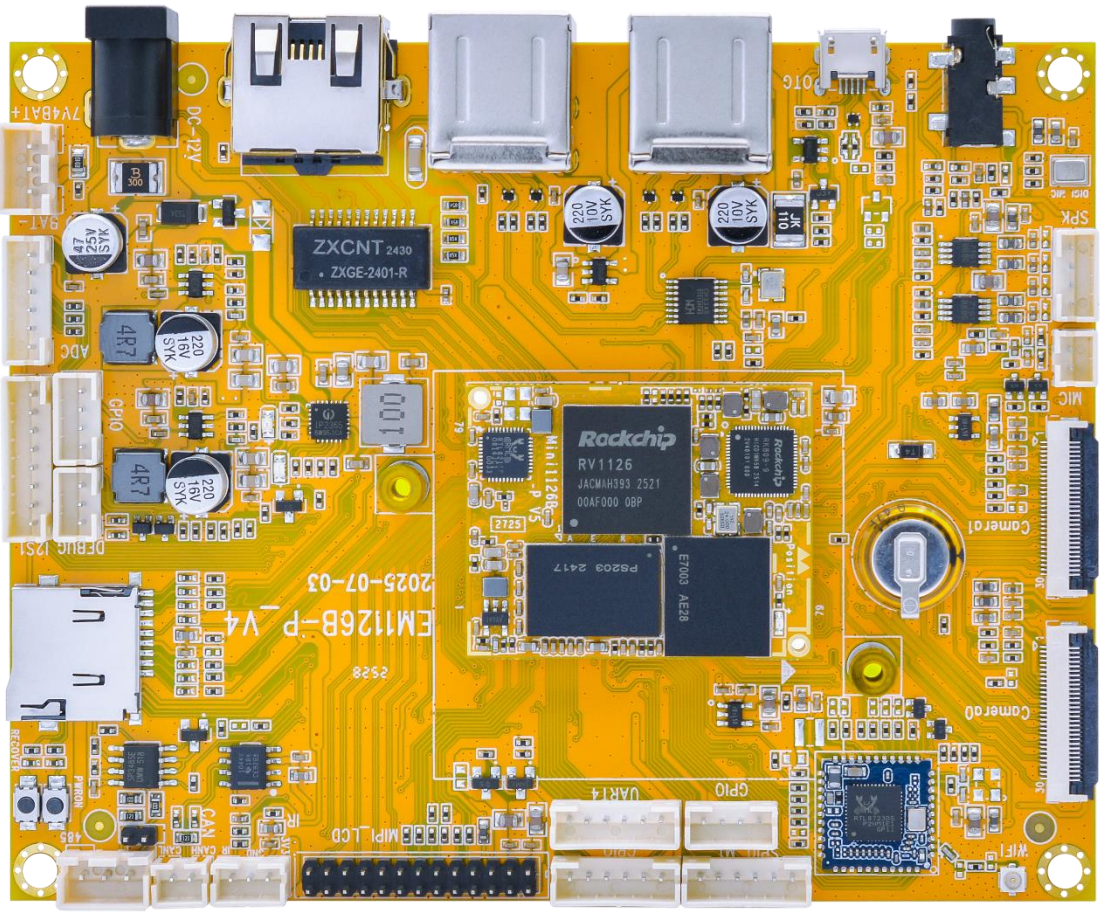
Content

1. Introduction

1.1 Overview

EM1126B-P is a high-performance intelligent vision development platform based on the Rockchip RV1126B-P AI SoC. It integrates a quad-core Cortex-A53 processor and a 3TOPS NPU, specifically designed for edge AI computing, video analytics, and image recognition applications.

With a built-in multi-channel AI-ISP image processing engine, H.264/H.265 codec, and a rich set of multimedia and peripheral interfaces, EM1126B-P supports MIPI camera inputs and 1080P display output. It is widely applicable to smart cameras, AI edge boxes, security surveillance, industrial vision, and more.

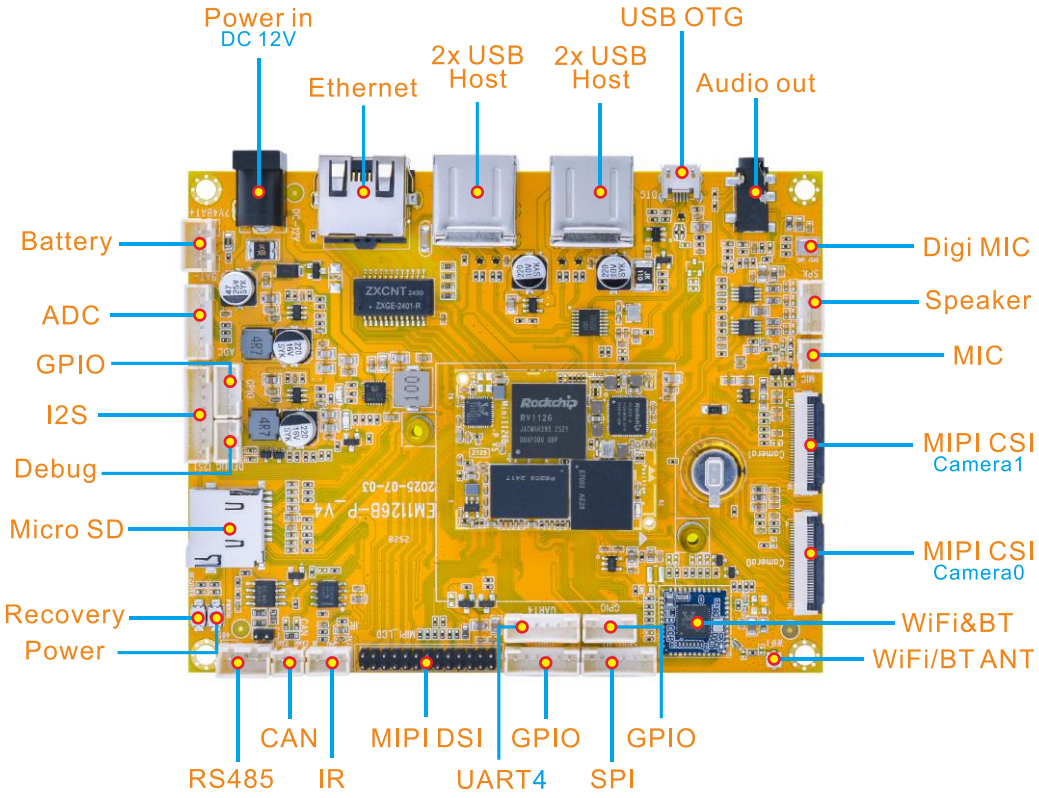


1.2 Product Parameters

Basic Parameters		
SOC		RV1126B-P
CPU		• Quad-core ARM Cortex-A53 • 32KB I-Cache / 32KB D-Cache per core • 512KB shared L2 Cache
NPU		• Up to 3 TOPS AI computing power • Supports INT4, INT8, INT16, FP16, BF16, TF32 • Optimized for CNN, Transformer, and other AI models
Video	Decoder	• H.265/H.264 up to 3840×2160@30fps • JPEG/MJPEG, VP8, VP9 decoding
	Encoder	• H.265/H.264 up to 3840×2160@30fps
AI-ISP		• Built-in 12MP AI Image Signal Processor • Supports multi-channel HDR, denoise, geometric correction
RAM		2GB LPDDR4
ROM		8GB eMMC
Support system		IPC
Hardware Parameters		
Extended Storage		• Support 1x MicroSD Card
Display		• Support 1x DSI MIPI
Audio		• Support 1x Speaker • Support 1x Headset out • Support 1x Differential MIC • Support 1x Digital MIC
USB		• Support 4x USB2.0
Network		• Support 1x Gigabit Ethernet

	• Support 1x WIFI/BT module
Camera	• Support 2x Camera
Peripheral communication	• Support 1x SPI • Support 1x RS485 • Support 1x RS232 • Support 1x CAN
Other parameters	Support 1xDebug UART, 1xOTG, 1xIR, 1xRTC, 1x Lion Battery connector
Electrical Parameters	
Power supply input voltage	12V/3A
RTC input voltage	3V/0.6uA
Operating temperature	0~70°
Storage temperature	-40~85°
Structural Parameters	
Core board dimensions	38mm x 30mm
Motherboard dimensions	120mm x 95mm

1.3 Hardware Interface Introduction



Interface parameters	
Power in DC 12V	12V DC power input interface
Ethernet	Gigabit Ethernet RJ45 interface
2x USB Host	Dual-layer USB2.0 HOST interface
USB OTG	USB OTG interface
Audio out	Headset out
Digi MIC	Digital MEMS MIC
Speaker	Speaker output
MIC	Differential MIC
MIPI CSI Camera1	MIPI CSI camera interface 1
MIPI CSI Camera0	MIPI CSI camera interface 0
WIFI&BT	WIFI&Bluetooth module

WIFI&BT ANT	Wi-Fi/Bluetooth antenna interface
GPIO	GPIO extension interface
SPI	SPI interface
UART4	UART4 serial communication interface
MIPI DIS	MIPI display interface
IR	IR interface
CAN	CAN communication interface
RS485	RS485 communication interface
Power	Power key
Recovery	Recovery key
Mirco SD	MicroSD card slot
Debug	Debug the serial port
ADC	ADC interface
Battery	Lion Battery interface

2. Install Drivers and Tools

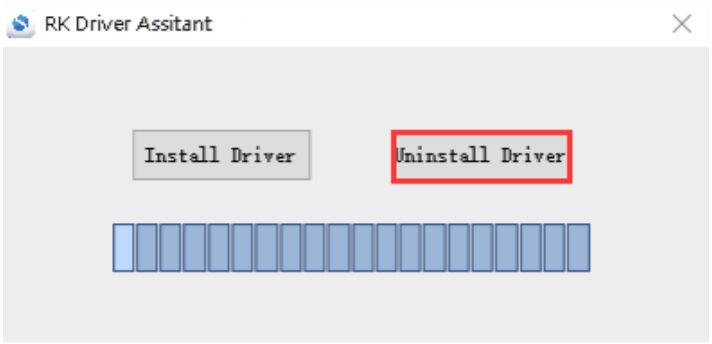
To download firmware and debug in the terminal, the following drivers and software need to be installed (for Windows computers):

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*.

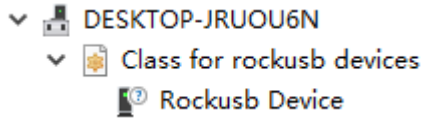
Step 2: To avoid driver conflicts, click **"Uninstall Driver"** to uninstall the driver.



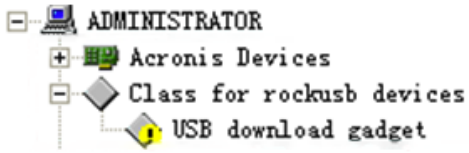
Step 3: Click button **"Install Driver"** to install.

Step 4: After the installation is complete, connect the board and PC with Micro USB 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.

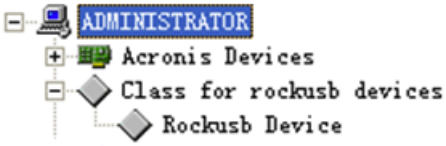


Step 5: If the following device information appears in the **Device Manager** after the operation in Step 4, user need to proceed to the next step.



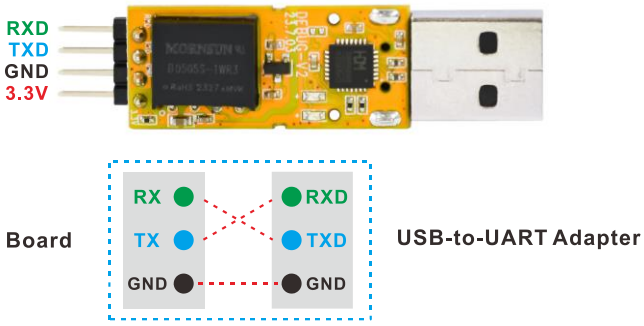
Step 6: The WINDOW will pop up found New Hardware Wizard dialog box, choose to install from the specified location, and then select: *DriverAssistant/ADBDriver*.

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



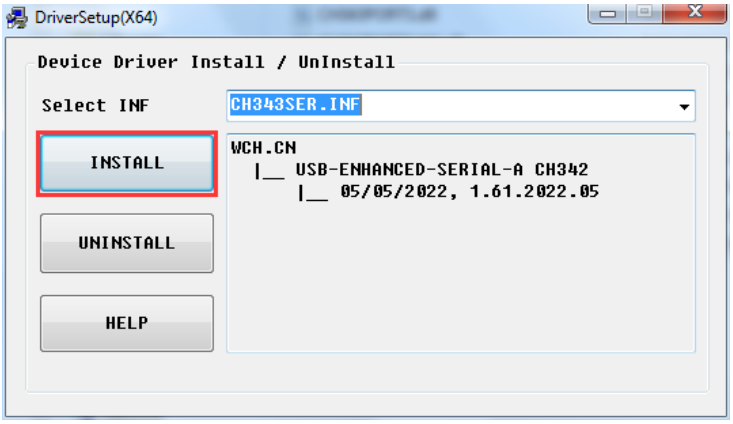
Pin	Connection Description
RXD	Receive, connect to TX pin of the board.
TXD	Transmit, connect to RX pin of the board.
GND	Ground, connect to GND pin of the board.
3V3	No need to connect.

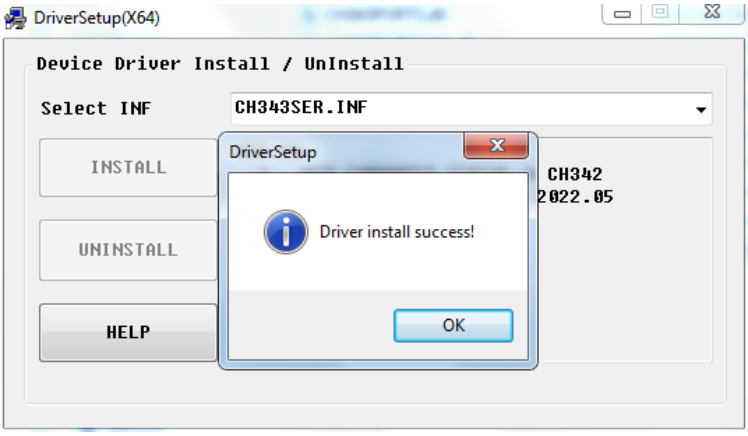
2.2.2 Install Driver

Step 1: Connect the CH9102X module to the PC.

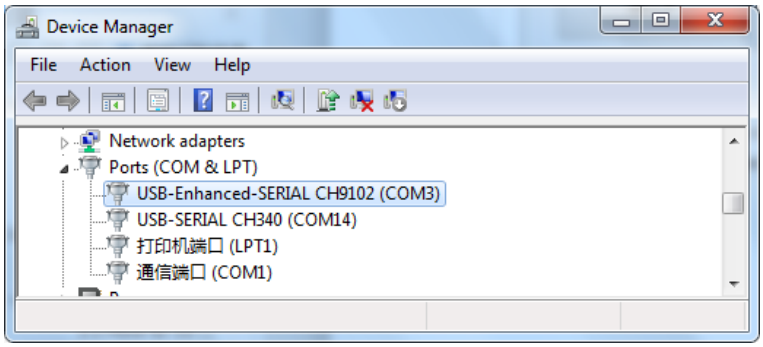
Step 2: Unzip *CH343SER.ZIP* on Windows.

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.

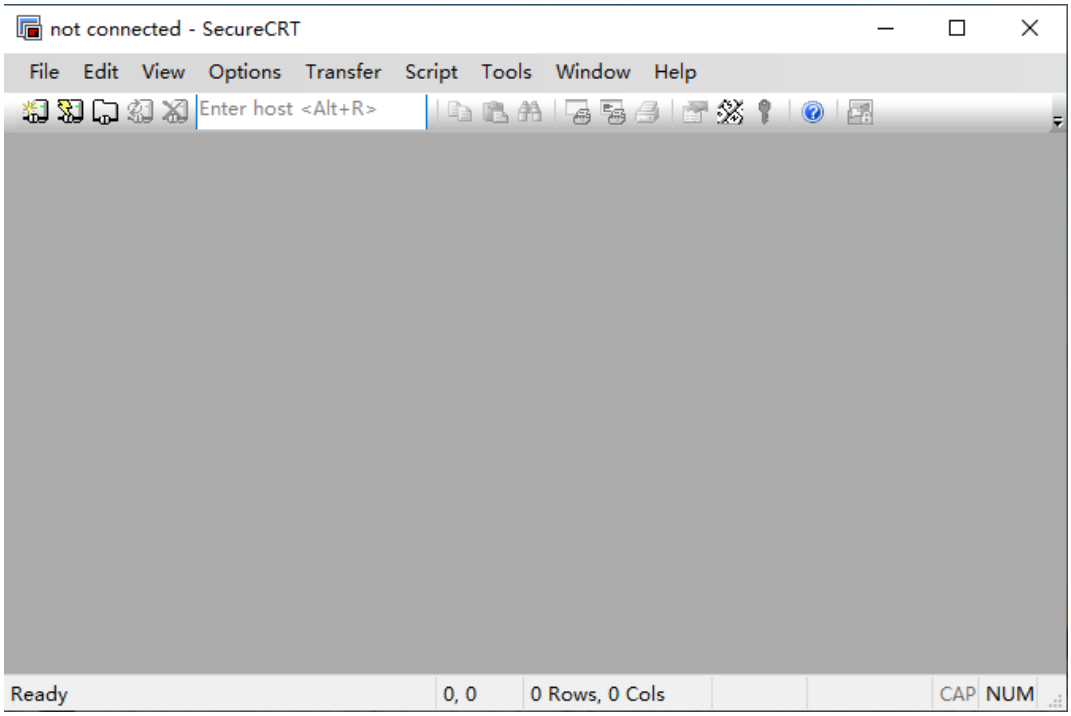


2.3 Install Serial Terminal Tool

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

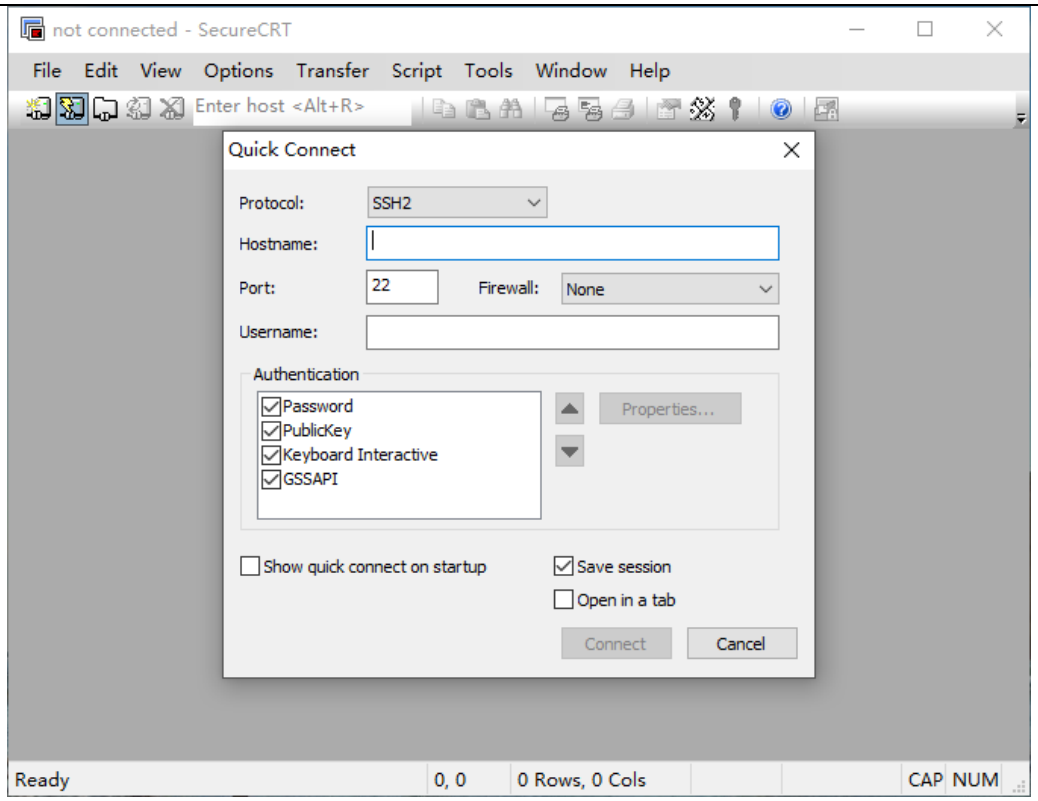
Step 1: Unzip *Platform/SecureCRT.rar* on PC.

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

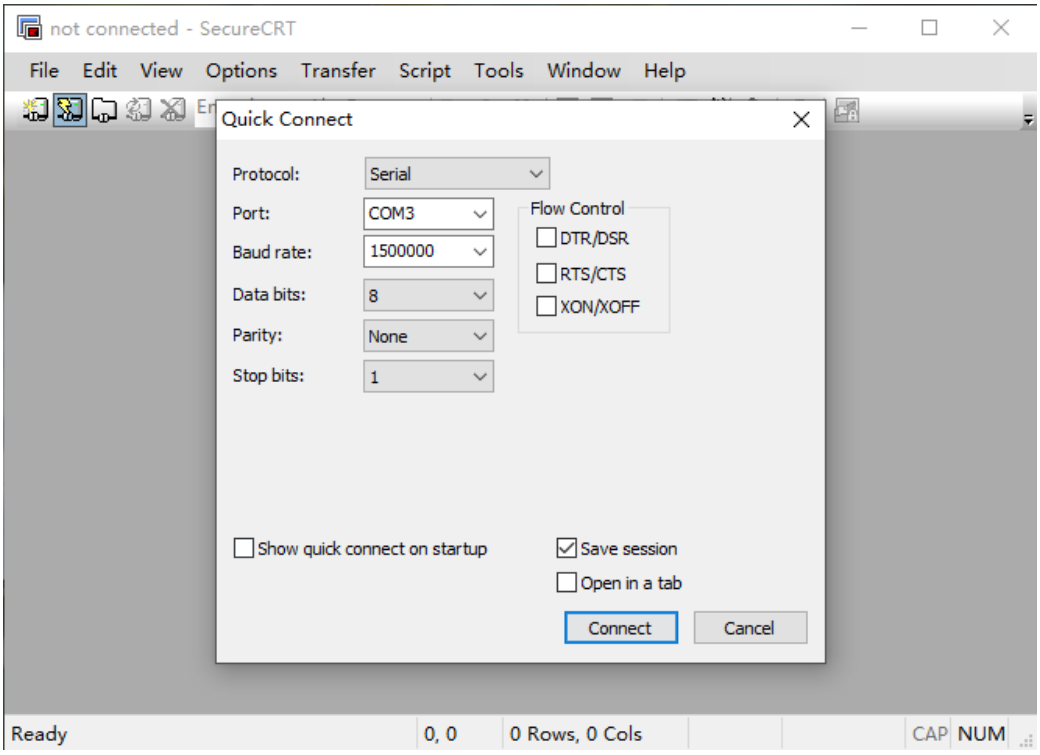


Step 3: Confirm that the CH9102X driver has been installed and the CH9102X module is connected to the PC.

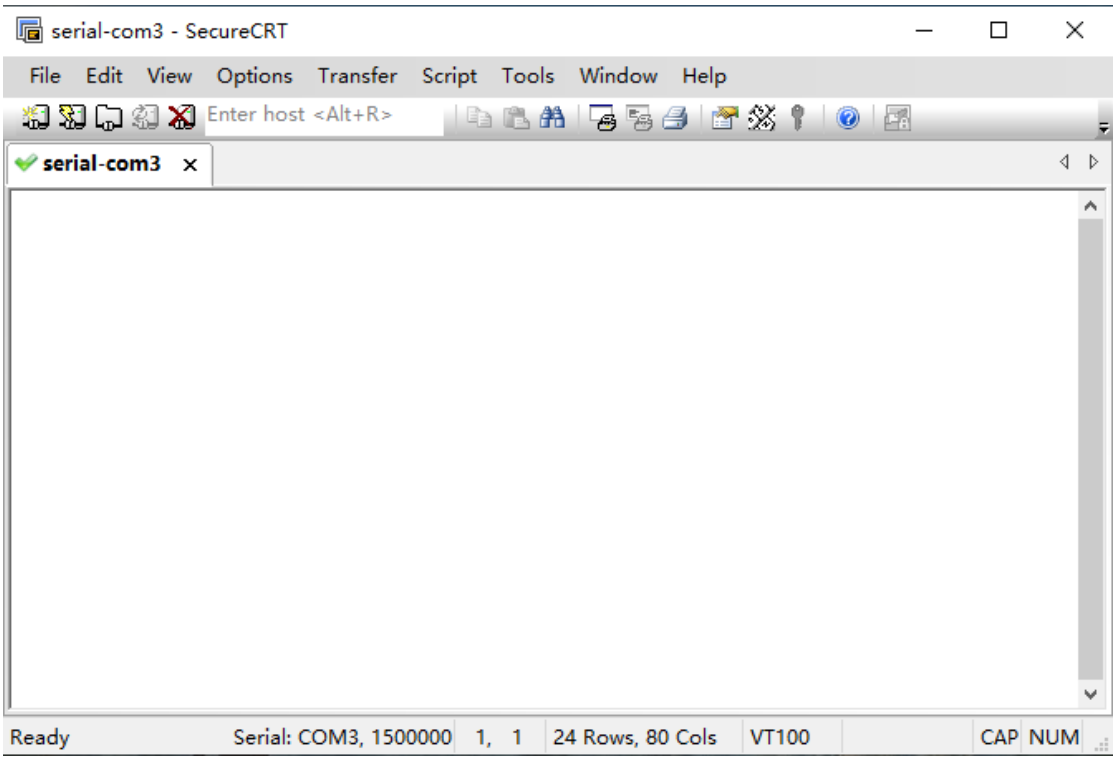
Step 4: Click the **"Quick Connect"** button to go to the Quick Connect configuration screen.



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.

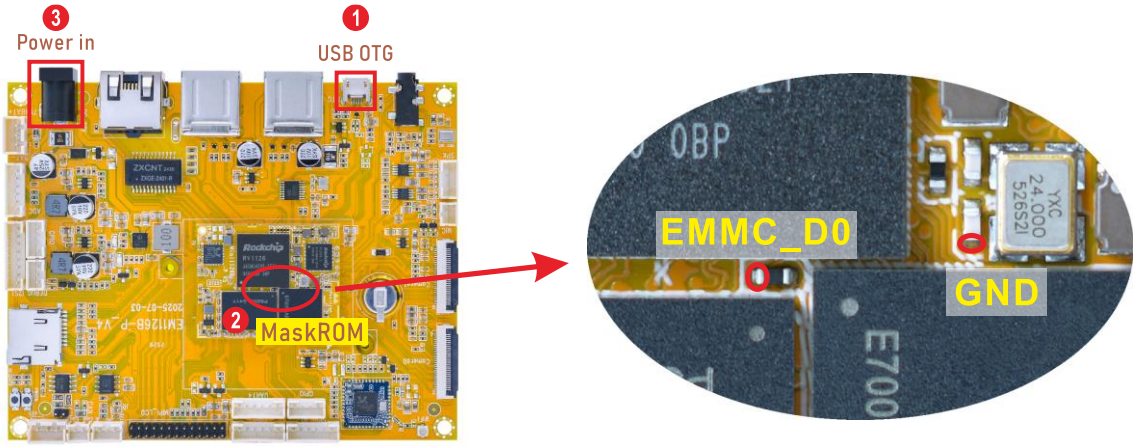
3.1.1 How to Enter MaskRom Mode

3.1.1.1 Hardware Method

Step 1: Disconnect the power adapter.

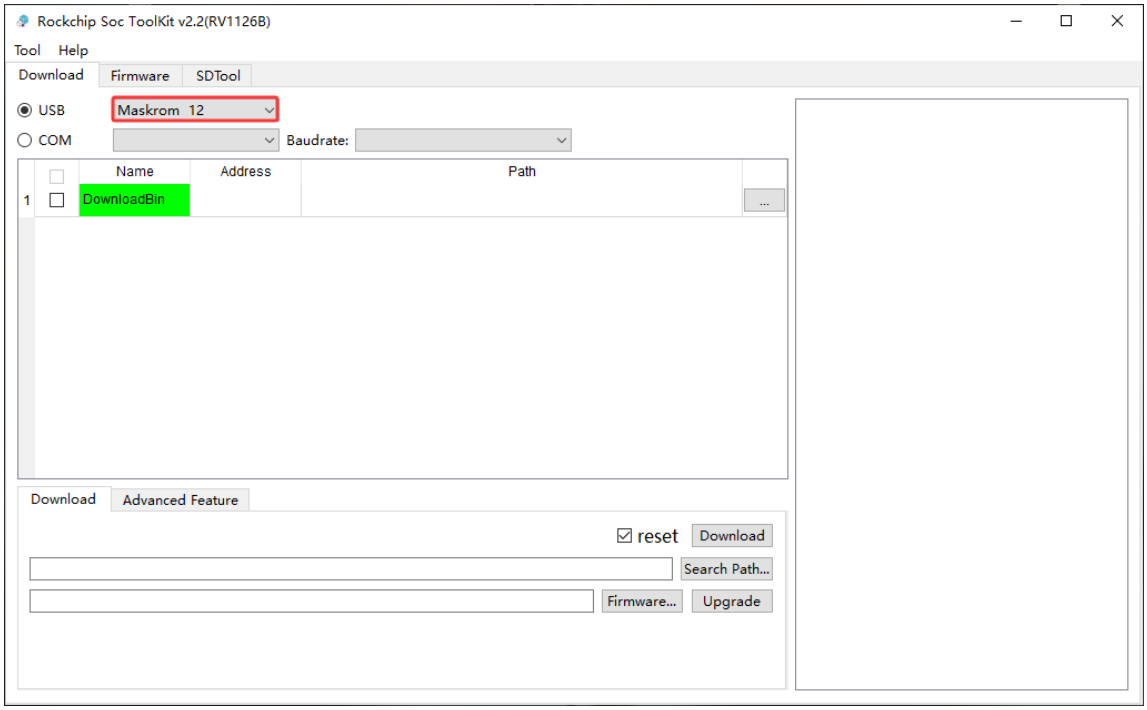
Step 2: Connect one end of the Micro cable to the host PC and the other end to the development board.

Step 3: Use tweezers to short the two MaskRom test points on the Mini1126B-P.



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

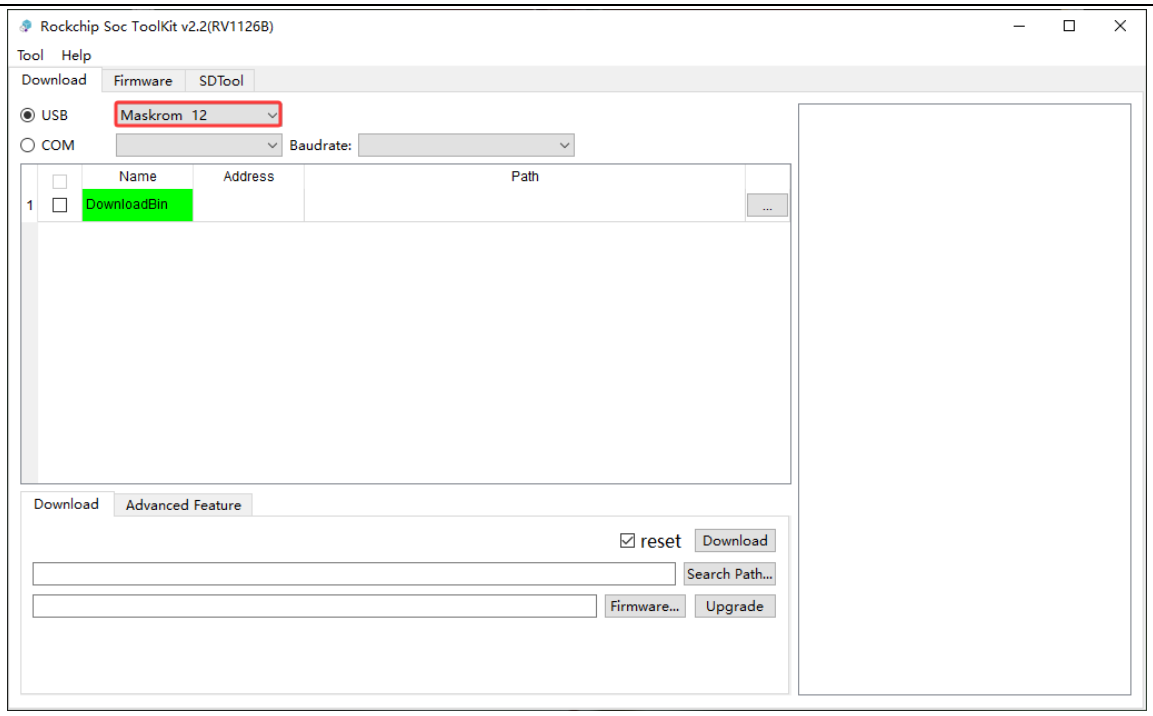
Step 5: After the USB device status changes to "**Maskrom**" in SocToolKit, release the tweezers. The device has now entered MaskRom mode.



3.1.1.2 Serial Terminal Method

Step 1: Connect one end of the Micro USB 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 After the USB device status changes to "**Maskrom**" in SocToolKit, release the tweezers. The device has now entered MaskRom mode.



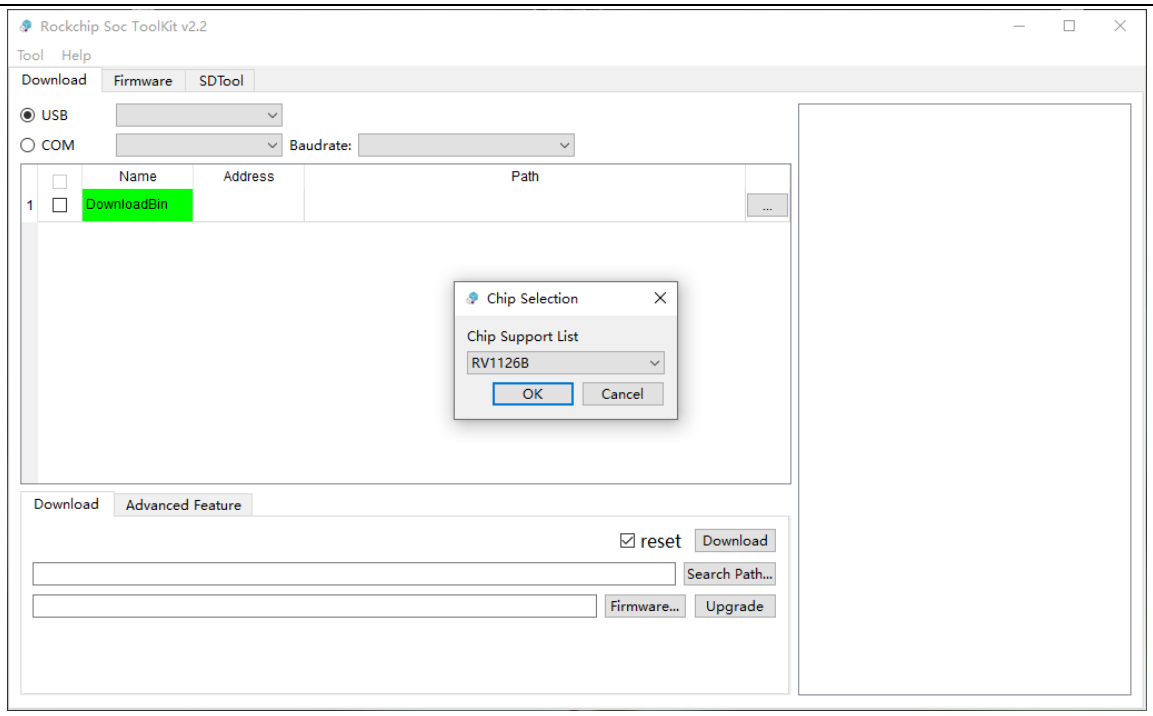
3.2 Firmware Flashing

Environment: Windows OS

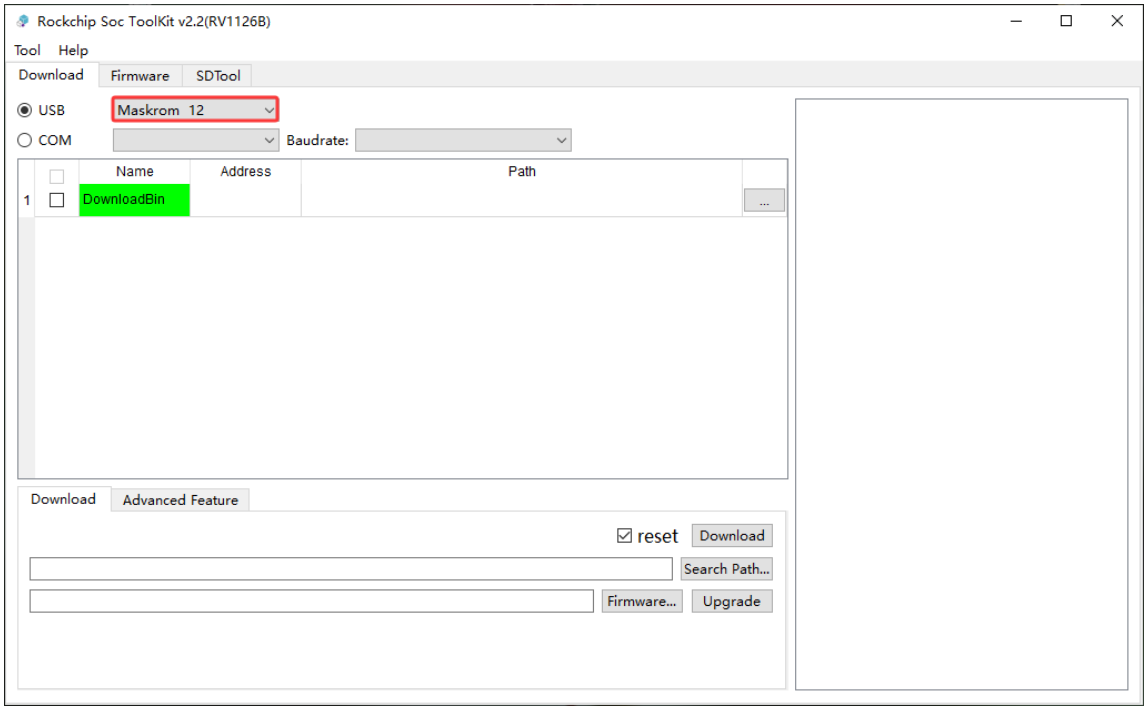
3.2.1 Flash Full Firmware Image

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

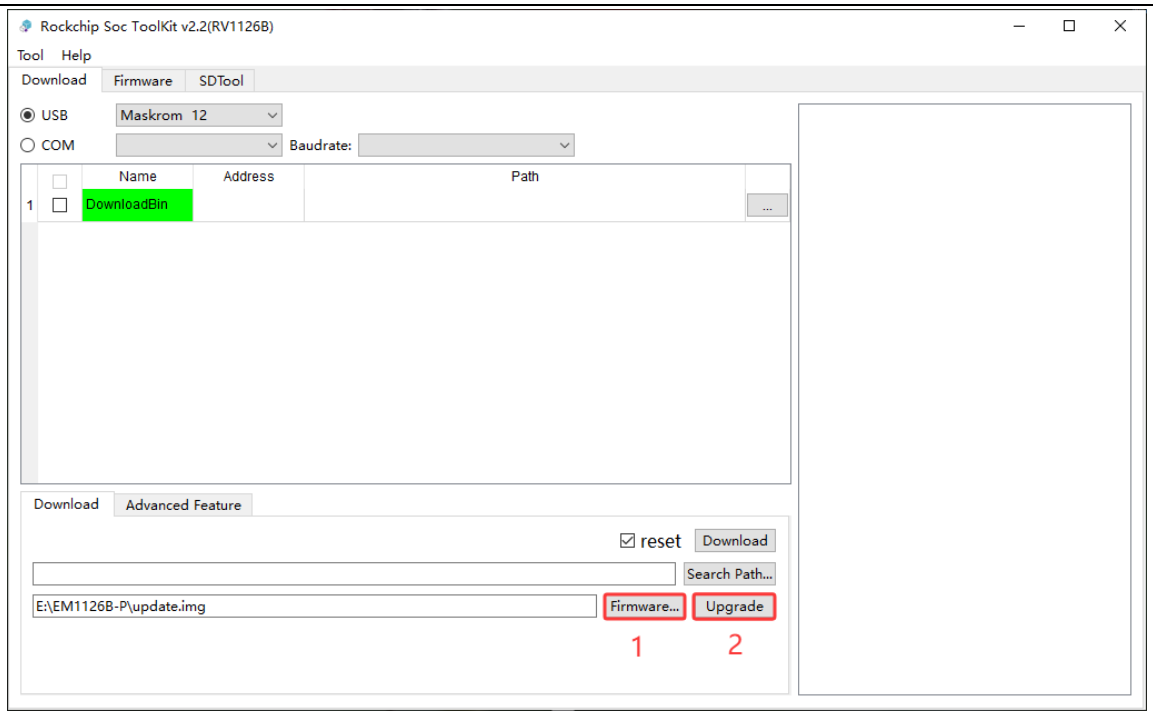
In the Chip Selection dialog box, keep the default chip option "**RV1126B**", and then click "**OK**".



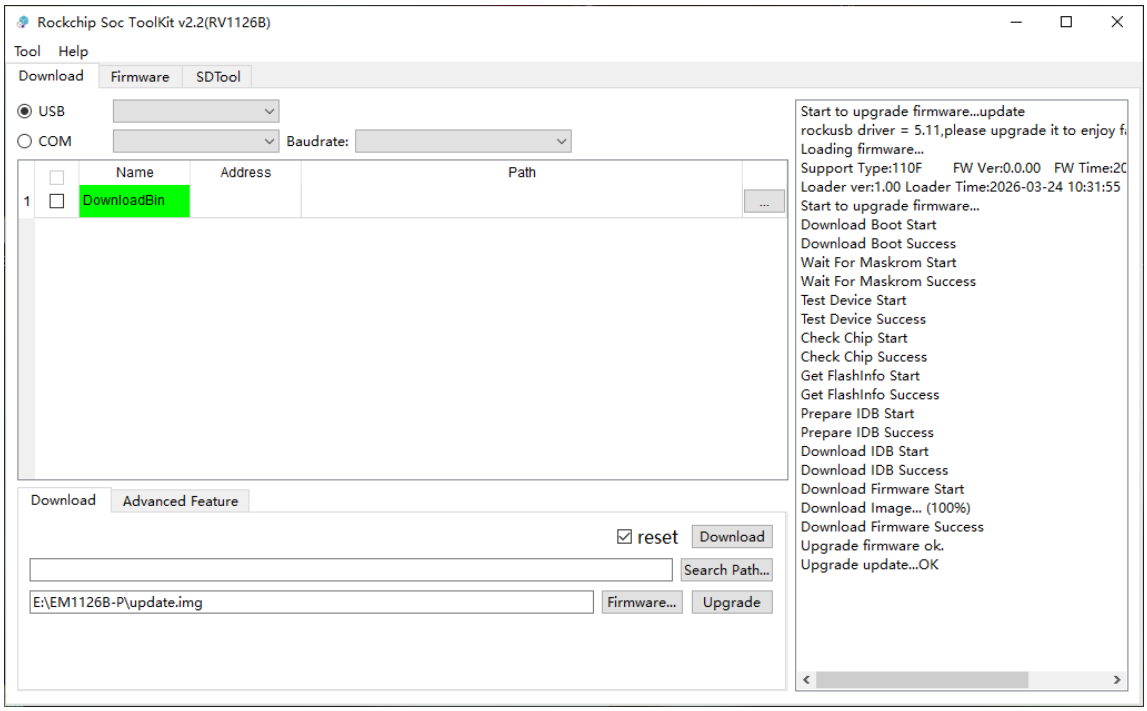
Step 2: Put the board into MaskRom mode. For details, refer to Section 3.1.2 How to Enter MaskRom Mode.



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

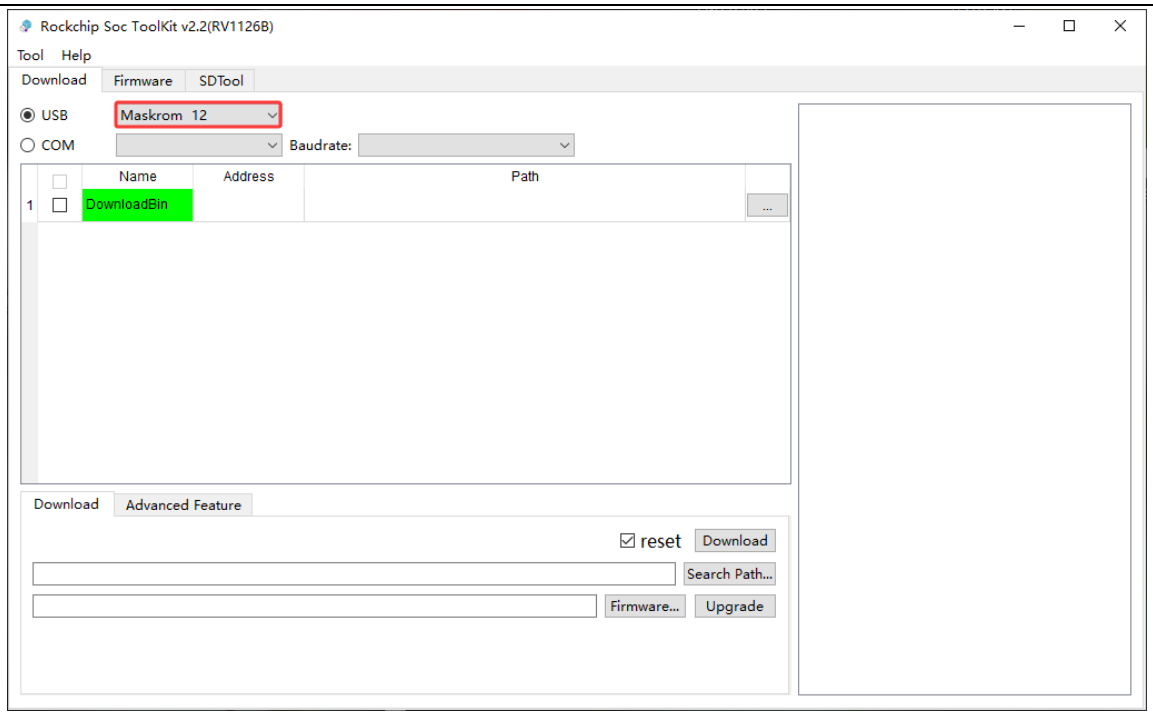


After the flashing is complete, the board will automatically reboot.

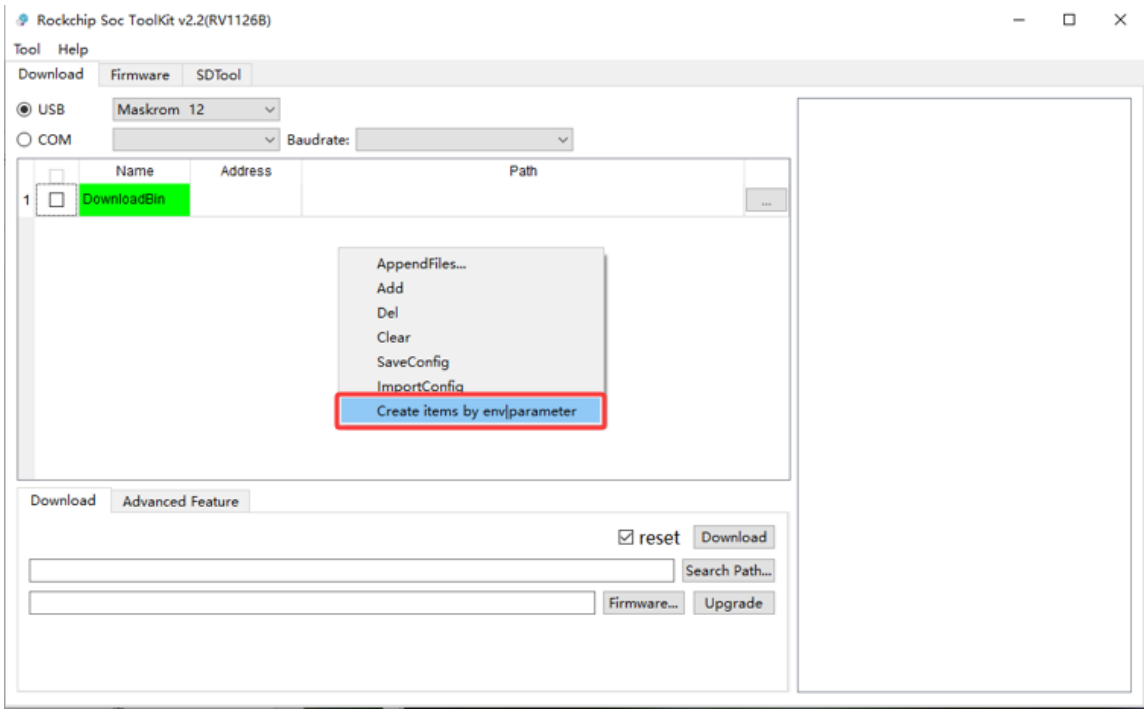


3.2.2 Flash Separate Partition Images

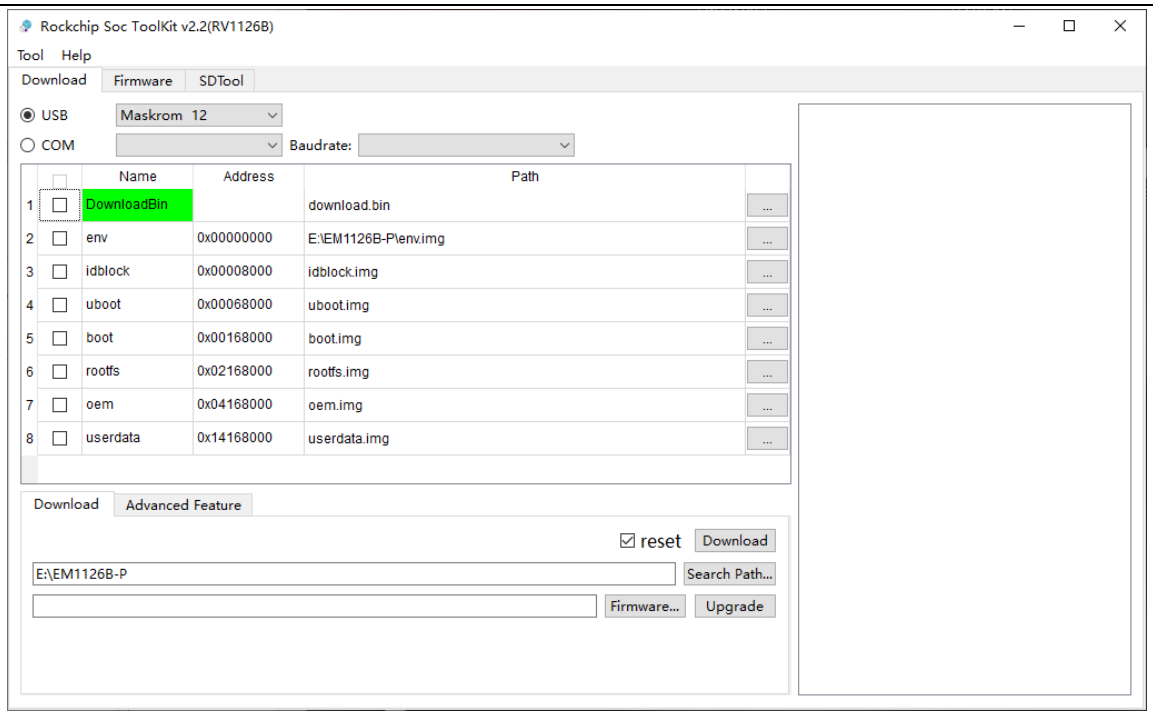
Step 1: Put the board into MaskRom mode. For details, refer to Section 3.1.2 How to Enter MaskRom Mode.



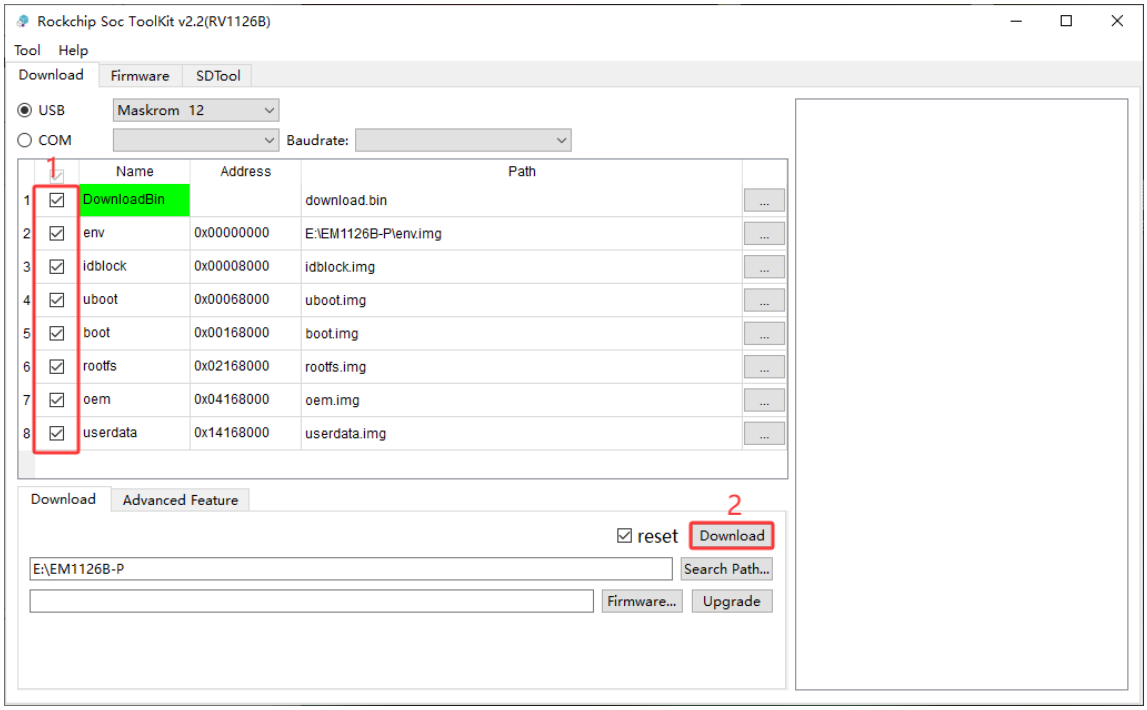
Step 2: Right-click the blank area in the image list and select **Create items by env|parameter**.



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.

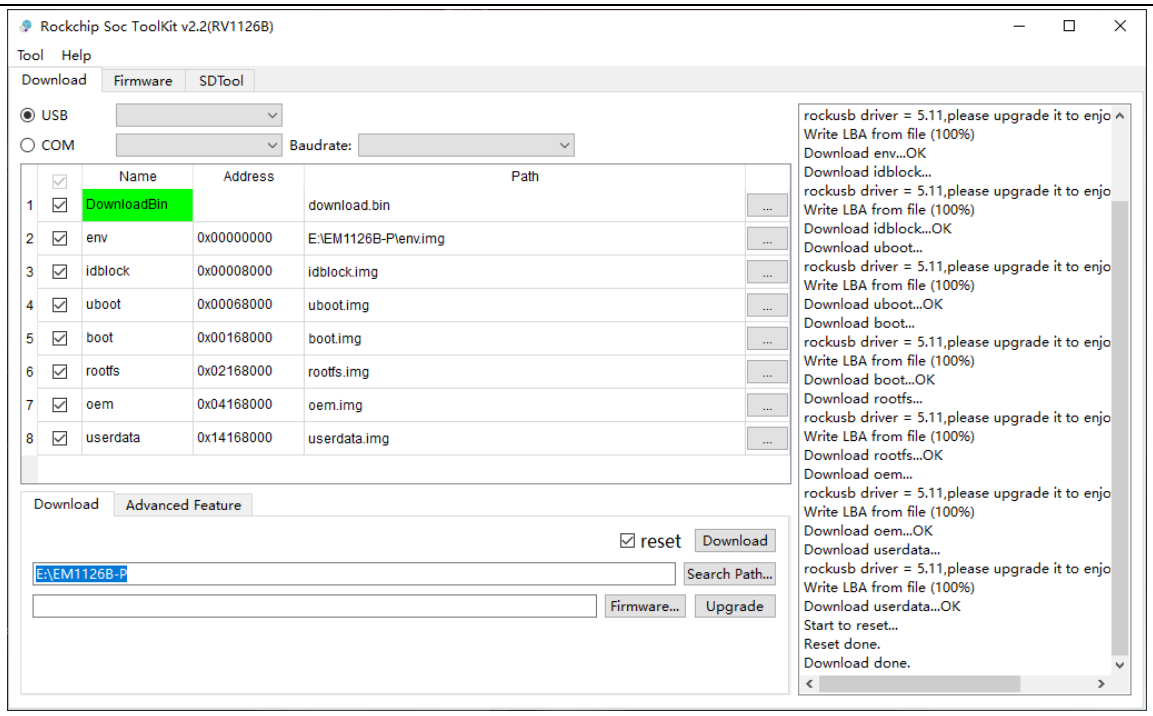


Step 3: Select the required image items to be flashed, and then click "**Download**" to start flashing the separate image firmware.



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 v2.2(RV1126B)
Tool
Help
Download
Firmware
SDTool
USB
COM
Baudrate:
Name
Address
Path
1 DownloadBin download.bin
2 env 0x00000000 E:\EM1126B-P\env.img
3 idblock 0x00008000 idblock.img
4 uboot 0x00068000 uboot.img
5 boot 0x00168000 boot.img
6 rootfs 0x02168000 rootfs.img
7 oem 0x04168000 oem.img
8 userdata 0x14168000 userdata.img
Download
Advanced Feature
reset
Download
E:\EM1126B-P
Search Path...
Firmware...
Upgrade
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download env...OK
Download idblock...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download idblock...OK
Download uboot...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download uboot...OK
Download boot...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download boot...OK
Download rootfs...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download rootfs...OK
Download oem...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download oem...OK
Download userdata...
rockusb driver = 5.11,please upgrade it to enjo
Write LBA from file (100%)
Download userdata...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_SDK_V2.tar.gz
$ 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 **0**:

0. BoardConfig_Boardcon/BoardConfig-EMMC-RK809-Boardcon_EM1126BP-IPC_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_Boardcon/BoardConfig-EMMC-RK809-Boardcon_EM1126BP-IPC_FASTBOOT.mk
                           boot medium(启动介质): EMMC
                         power solution(电源方案): RK809
                       hardware version(硬件版本): Boardcon_EM1126BP
                           application(应用场景): IPC_FASTBOOT
------------------------------------------------------------
------------------------------------------------------------
1. BoardConfig_IDEA1126/BoardConfig-EMMC-Boardcon-IDEA1126BP-IPC_FASTBOOT.mk
                           boot medium(启动介质): EMMC
                         power solution(电源方案): Boardcon
                       hardware version(硬件版本): IDEA1126BP
                           application(应用场景): IPC_FASTBOOT
------------------------------------------------------------
Which would you like? [0]:0
```

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.

Before testing other functions, stop the rkipc service first.

Stop the rkipc service:

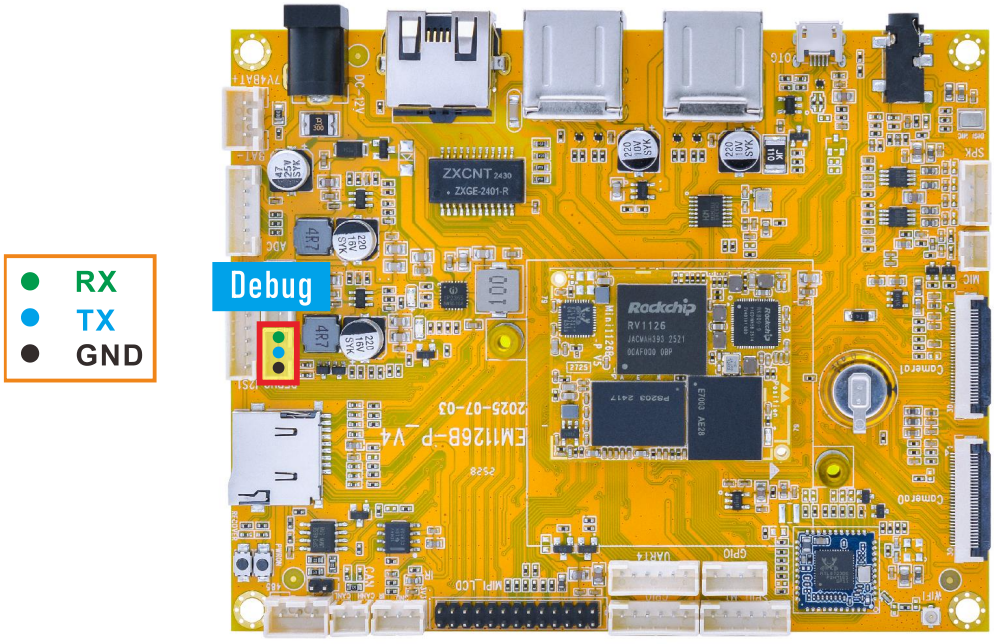
```
# killall -9 rkipc
```

restart the rkipc service:

```
# rkipc &
```

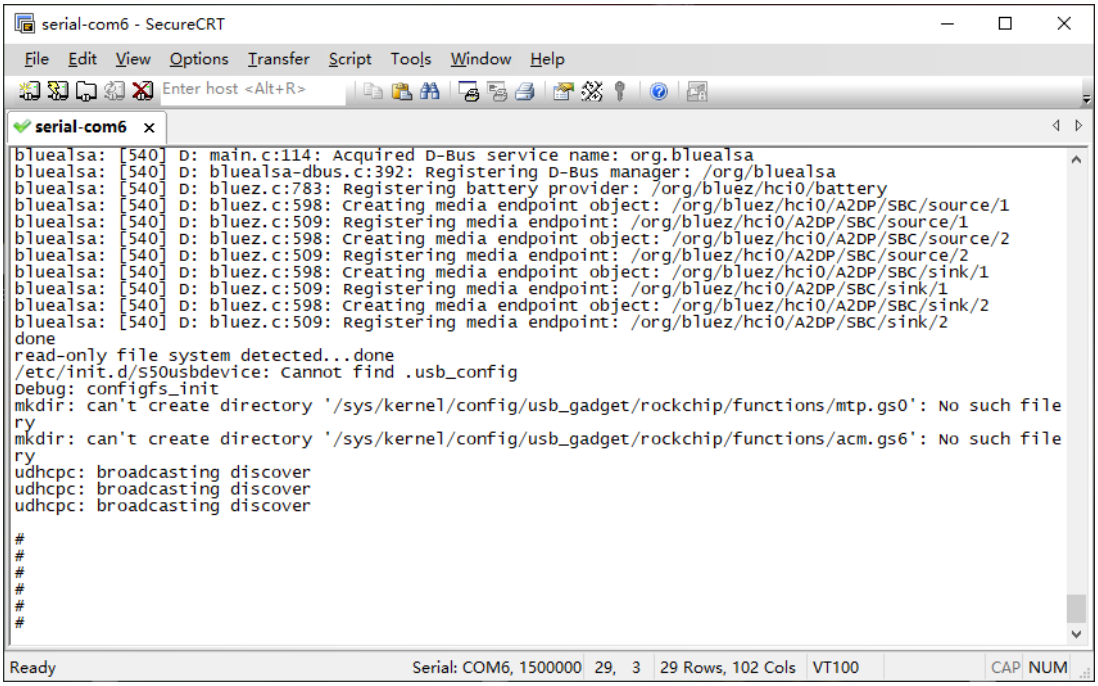
6.1 Serial Terminal

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.

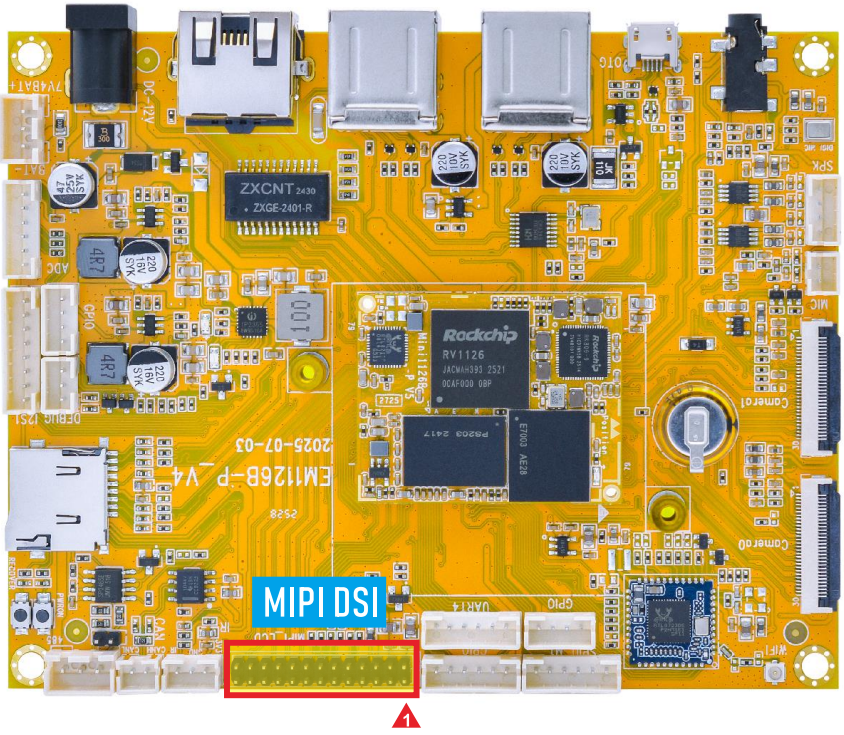
Step 3: Power on the board and check whether the boot log is displayed in the serial terminal.



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 EM1126B-P 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 camera preview after booting. The display output shows the preview image from Camera0.

6.3 USB

6.3.1 USB OTG

By default, the OTG port operates in **Device mode**, allowing ADB connections for debugging.

To enable ADB on a Windows host:

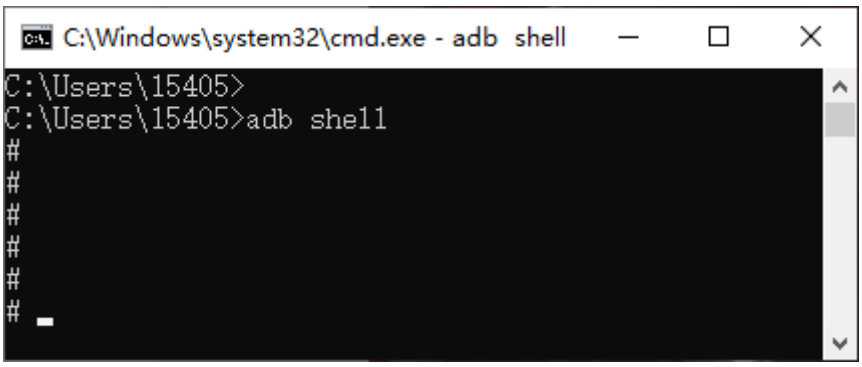
Step 1: Connect the board and PC host with Micro USB cable.

Step 2: Install ADB driver on Windows system.

Step 3: Press **Windows + R**, type cmd, and press **Enter** to open the command prompt

Step 4: Run the following command to check ADB connection.

```
# adb shell
```



6.3.2 USB HOST

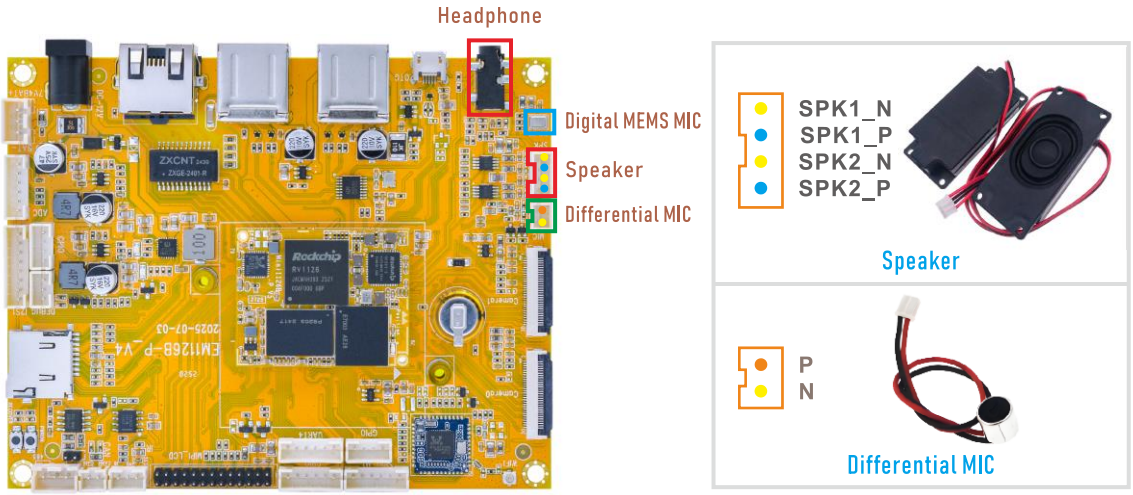
The USB2.0 Host interfaces can be used to connect USB peripherals, such as a USB mouse, USB keyboard, USB flash drive, and other USB devices.

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                33.9M     33.9M         0 100% /
devtmpfs                 60.1M         0     60.1M   0% /dev
tmpfs                   966.6M         0    966.6M   0% /dev/shm
tmpfs                   966.6M      8.0K    966.6M   0% /tmp
tmpfs                   966.6M    412.0K    966.2M   0% /run
/dev/block/by-name/oem
                        237.9M     86.1M    135.9M  39% /oem
/dev/block/by-name/userdata
                          5.8G    248.0K      5.7G   0% /userdata
/dev/sda1               117.2G    768.0K    117.2G   0% /tmp/sda1
/dev/sdb1                58.0G     37.5G     20.5G  65% /tmp/sdb1
```

6.4 Audio



7.4.1 Audio Input Test

There are two audio input devices: **Differential MIC** and **Digital MEMS MIC**.

Preparation

Before testing the audio input function, stop the rkipc service:

Execute the following command:

```
# killall -9 rkipc
```

```
# killall -9 rkipc
# Populating /dev using udev: bluealsa: [621] D: storage.c:84: Initializing persistent storage:
/var/lib/bluealsa
bluealsa: [621] W: storage.c:87: Couldn't create storage directory: Read-only file system
bluealsa: [621] D: main.c:604: Starting main dispatching loop
bluealsa: [621] D: main.c:114: Acquired D-Bus service name: org.bluealsa
bluealsa: [621] D: bluealsa-dbus.c:392: Registering D-Bus manager: /org/bluealsa
bluealsa: [621] D: bluez.c:783: Registering battery provider: /org/bluez/hci0/battery
bluealsa: [621] D: bluez.c:598: Creating media endpoint object: /org/bluez/hci0/A2DP/SBC/source/1
bluealsa: [621] D: bluez.c:509: Registering media endpoint: /org/bluez/hci0/A2DP/SBC/source/1
bluealsa: [621] D: bluez.c:598: Creating media endpoint object: /org/bluez/hci0/A2DP/SBC/source/2
bluealsa: [621] D: bluez.c:509: Registering media endpoint: /org/bluez/hci0/A2DP/SBC/source/2
bluealsa: [621] D: bluez.c:598: Creating media endpoint object: /org/bluez/hci0/A2DP/SBC/sink/1
bluealsa: [621] D: bluez.c:509: Registering media endpoint: /org/bluez/hci0/A2DP/SBC/sink/1
bluealsa: [621] D: bluez.c:598: Creating media endpoint object: /org/bluez/hci0/A2DP/SBC/sink/2
bluealsa: [621] D: bluez.c:509: Registering media endpoint: /org/bluez/hci0/A2DP/SBC/sink/2
done
read-only file system detected...done
/etc/init.d/S50usbdevice: Cannot find .usb_config
Debug: configfs_init
mkdir: can't create directory '/sys/kernel/config/usb_gadget/rockchip/functions/mtp.gs0': No such
file or directory
mkdir: can't create directory '/sys/kernel/config/usb_gadget/rockchip/functions/acm.gs6': No such
file or directory
```

- **Differential MIC**

Connect the Differential MIC, and then execute the following commands:

```
# rk_mpi_amix_test -C 0 --control 'Capture MIC Path' --value 'Main Mic'
# rk_mpi_ai_test --device_rate 44100 --device_ch 2 --out_rate 44100 --out_ch 2 -o
/tmp --sound_card_name hw:0,0 -q 320
```

```
# rk_mpi_amix_test -C 0 --control 'Capture MIC Path' --value 'Main Mic'
cmd parse result:
sound control id     : 0
control name         : Capture MIC Path
control value        : Main Mic
list controls        : 0
list contents        : 0
#
# rk_mpi_ai_test --device_rate 44100 --device_ch 2 --out_rate 44100 --out_ch 2 -
o /tmp --sound_card_name hw:0,0 -q 320udhcpc: broadcasting discover
udhcpc: broadcasting discover

cmd parse result:
input  file name     : (null)
output file name     : /tmp
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
```

• **Digital MEMS MIC**

Execute the following command to record audio from the Digital MEMS MIC:

```
# rk_mpi_ai_test --device_rate 16000 --device_ch 2 --out_rate 16000 --out_ch 2 --
sound_card_name hw:1,0 -o /tmp --set_track_mode 10 -q 320
```

```
# rk_mpi_ai_test --device_rate 16000 --device_ch 2 --out_rate 16000 --out_ch 2 -
-sound_card_name hw:1,0 -o /tmp --set_track_mode 10 -q 320
cmd parse result:
input  file name     : (null)
output file name     : /tmp
loop count          : 1
channel number       : 1
open sound rate      : 16000
record data rate     : 16000
sound card channel    : 2
output channel       : 2
bit_width           : 16
frame_number         : 4
frame_length         : 1024
sound card name      : hw:1,0
device id           : 0
set volume curve     : 0
set volume          : 100
set mute            : 0
set track_mode       : 10
```

7.4.2 Audio Output Test

There are two audio output devices: **Headphone** and **Speaker L/R**.

• **Headphone**

Connect the Headphone, and then execute the following commands:

```
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'HP'
# rk_mpi_ao_test -i /oem/usr/share/speaker_test.wav --input_ch 2 --input_rate
8000 --device_rate 8000 --device_ch 2 --sound_card_name hw:0,0
```

```
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'HP'
cmd parse result:
sound control id     : 0
control name         : Playback Path
control value        : HP
list controls        : 0
list contents        : 0
#
# udhcpc: broadcasting discover
udhcpc: broadcasting discover

# rk_mpi_ao_test -i /oem/usr/share/speaker_test.wav --input_ch 2 --input_rate 80
00 --device_rate 8000 --device_ch 2 --sound_card_name hw:0,0
cmd parse result:
input  file name     : /oem/usr/share/speaker_test.wav
output file name     : (null)
loop count           : 1
channel number       : 1
open sound rate      : 8000
open sound channel    : 2
input stream rate    : 8000
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)          11:11:11-909 {log_level_init    :209}
```

- **Speaker L/R**

Connect the Speaker, and then execute the following commands:

```
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'SPK'
# rk_mpi_ao_test -i /oem/usr/share/speaker_test.wav --input_ch 2 --input_rate
8000 --device_rate 8000 --device_ch 2 --sound_card_name hw:0,0
```

```
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'SPK'
cmd parse result:
sound control id     : 0
control name         : Playback Path
control value        : SPK
list controls        : 0
list contents        : 0
#
# udhcpc: broadcasting discover

# rk_mpi_ao_test -i /oem/usr/share/speaker_test.wav --input_ch 2 --input_rate 80
00 --device_rate 8000 --device_ch 2 --sound_card_name hw:0,0udhcpc: broadcasting discover

cmd parse result:
input  file name     : /oem/usr/share/speaker_test.wav
output file name     : (null)
loop count           : 1
channel number       : 1
open sound rate      : 8000
open sound channel   : 2
input stream rate    : 8000
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
```

6.5 Ethernet

Step 1: Connect the network cable to the Ethernet port.

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

```
# dmesg | grep -Ei "link|100Mbps"
[    0.327919] NET: Registered PF_NETLINK/PF_ROUTE protocol family
[    1.253812] mpp_rkvdec2 22140100.rkvdec: link mode probe finish
[    1.257018] can: netlink gateway - max_hops=1
[    2.322773] rk_gmac-dwmac 21c70000.ethernet eth0: configuring for phy/rgmii link mode
[ 2871.588024] rk_gmac-dwmac 21c70000.ethernet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
```

Step 2: Check the network interface information.

Execute the following command:

```
# ifconfig eth0
```

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 32:D8:C6:1C:4B:9F
          inet addr:192.168.0.182  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:375 errors:0 dropped:50 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32258 (31.5 KiB)  TX bytes:684 (684.0 B)
          Interrupt:108
```

Step 3: Test the network connection.

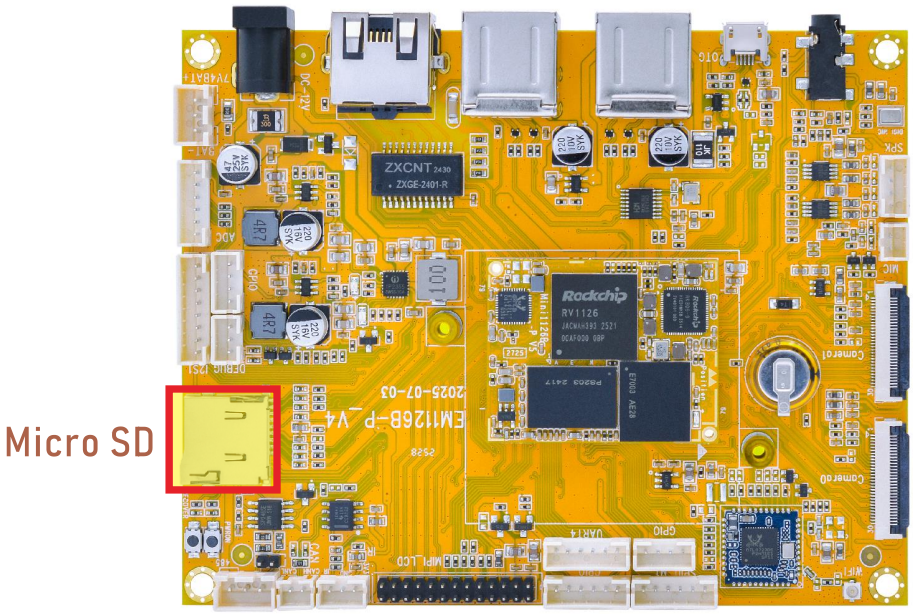
Execute the following command:

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

```
# ping -I eth0 www.armdesigner.com
PING www.armdesigner.com (108.138.246.80): 56 data bytes
64 bytes from 108.138.246.80: seq=0 ttl=245 time=161.349 ms
64 bytes from 108.138.246.80: seq=1 ttl=245 time=161.110 ms
64 bytes from 108.138.246.80: seq=2 ttl=245 time=161.513 ms
64 bytes from 108.138.246.80: seq=3 ttl=245 time=161.427 ms
64 bytes from 108.138.246.80: seq=4 ttl=245 time=161.518 ms
64 bytes from 108.138.246.80: seq=5 ttl=245 time=161.690 ms
^C
--- www.armdesigner.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 161.110/161.434/161.690 ms
```

6.6 SD Card

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



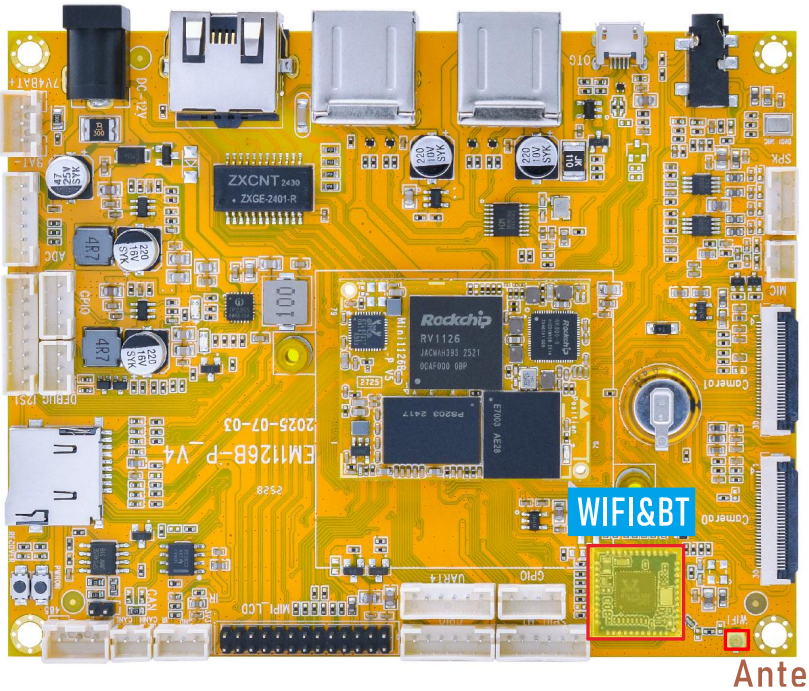
After the SD card is recognized, it will be mounted automatically. Run the following command to check the mount path of the SD card.

```
# df -h
```

```
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                33.9M     33.9M         0 100% /
devtmpfs                 60.1M         0     60.1M   0% /dev
tmpfs                   966.6M         0    966.6M   0% /dev/shm
tmpfs                   966.6M    336.0K    966.3M   0% /tmp
tmpfs                   966.6M    420.0K    966.2M   0% /run
/dev/block/by-name/oem
                        237.9M     86.1M    135.9M  39% /oem
/dev/block/by-name/userdata
                          5.8G    248.0K      5.7G   0% /userdata
/dev/sda1               117.2G    768.0K    117.2G   0% /tmp/sda1
/dev/sdb1                58.0G     37.5G     20.5G  65% /tmp/sdb1
/dev/mmcblk1p1           59.5G     12.1G     47.4G  20% /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 84:FC:14:8A:A6:29
          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    -49     [WPA2-PSK-CCMP][ESS]
b4:f1:8c:fd:d1:29       2462    -50     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    Boardcon_Wi-Fi5
b4:f1:8c:6d:d1:24       2462    -51     [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]    Boardcon
b2:22:7a:5a:b6:4a       2462    -45     [WPA2-PSK-CCMP][ESS]    DIRECT-4A-HP Laser 136w
d8:32:14:25:b7:a8       2437    -65     [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]      LYB-2.4G
44:6a:2e:20:83:60       2412    -69     [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]      ydsz
d8:fe:42:ce:ee:d3       2432    -80     [WPA2-PSK-CCMP][ESS]    DIRECT-zb-HUAWEI CV81-WDMa
b4:f1:8c:6d:d1:25       2462    -50     [ESS]
```

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
[1501846144:807223][src/Rk_wifi.c][wpa_wifi_connect1][l:1851],[wpa_wifi_connect1] ssid: Boardcon, psk:
Boardcon43435656

[1501846144:807276][src/Rk_wifi.c][_RK_wifi_running_getConnectionInfo][l:710],_RK_wifi_running_getConnecti
onInfo: enter

[1501846144:807363][src/Rk_wifi.c][_RK_wifi_running_getConnectionInfo][l:716],remove /tmp/wifi_status.tmp
failed!

# [1501846144:821049][src/Rk_wifi.c][_RK_wifi_running_getConnectionInfo][l:722],wifi_status.tmp:
wpa_state=DISCONNECTED
address=84:fc:14:8a:a6:29
```

Step 4: Check the network interface status.

Execute the following command:

```
# ifconfig wlan0
```

```
# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 84:FC:14:8A:A6:29
          inet addr:192.168.0.6  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:644 errors:0 dropped:127 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:70096 (68.4 KiB)  TX bytes:2878 (2.8 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 (108.138.246.87): 56 data bytes
64 bytes from 108.138.246.87: seq=0 ttl=245 time=219.358 ms
64 bytes from 108.138.246.87: seq=1 ttl=245 time=270.996 ms
64 bytes from 108.138.246.87: seq=2 ttl=245 time=302.633 ms
64 bytes from 108.138.246.87: seq=3 ttl=245 time=162.674 ms
64 bytes from 108.138.246.87: seq=4 ttl=245 time=194.119 ms
64 bytes from 108.138.246.87: seq=5 ttl=245 time=177.880 ms
^C
--- www.armdesigner.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 162.674/221.276/302.633 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: [346] D: storage.c:84: Initializing persistent storage: /var/lib/bluealsa
bluealsa: [346] W: storage.c:87: Couldn't create storage directory: Read-only file system
bluealsa: [346] D: main.c:604: Starting main dispatching loop
bluealsa: [346] E: main.c:133: Couldn't acquire D-Bus name. Please check D-Bus configuration. Requested
name: org.bluealsa
```

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 84:FC:14:8A:A6:2A Pairable: yes
[bluetooth]# [bluetooth]#
```

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

Execute the following commands in bluetoothctl:

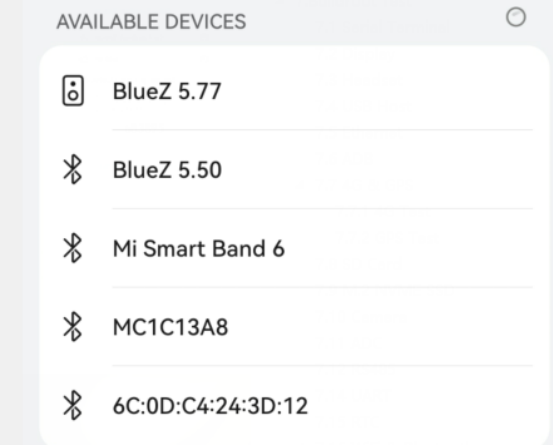
```
[bluetooth]# power on
[bluetooth]# discoverable on
```

```
[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 84:FC:14:8A:A6:2A Discoverable: yes
```

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]# [bluetooth]# [bluetooth]# [bluetooth]# [bluetooth]# [bluetooth]#
hci0 A8:35:12:9A:EB:4D type BR/EDR connected eir_len 11
[bluetooth]# [bluetooth]# bluealsa: [536] D: bluez.c:1199: Signal:
org.freedesktop.DBus.ObjectManager.InterfacesAdded()
                    [NEW] Device A8:35:12:9A:EB:4D liuy
[bluetooth]# [bluetooth]# [liuy]#                       Request confirmation
[liuy]# [liuy]# [agent] Confirm passkey 414906 (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 Bonded: yes
[liuy]# [liuy]# bluealsa: [536] D: bluez.c:1199: Signal:
org.freedesktop.DBus.ObjectManager.InterfacesAdded()
              [CHG] Device A8:35:12:9A:EB:4D Modalias: bluetooth:v010Fp107Ed1436
 [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]#                  hci0 A8:35:12:9A:EB:4D type BR/EDR disconnected with reason 2
[liuy]# [liuy]#                  [CHG] Device A8:35:12:9A:EB:4D ServicesResolved: no
[liuy]# [liuy]# [bluetooth]#                      [CHG] Device A8:35:12:9A:EB:4D Connected: no
[bluetooth]# [bluetooth]#                     hci0 A8:35:12:9A:EB:4D type BR/EDR connected eir_len 11
[bluetooth]# [bluetooth]# [liuy]#                       [CHG] Device A8:35:12:9A:EB:4D Connected: yes
[liuy]# [liuy]#                  Authorize service
[liuy]# [liuy]# [agent] Authorize service 0000110d-0000-1000-8000-00805f9b34fb (yes/no): yes
 [liuy]# [liuy]# bluealsa: [536] D: bluez.c:1199: Signal:
org.freedesktop.DBus.ObjectManager.InterfacesAdded()
bluealsa: [536] D: dbus.c:47: Called: org.bluez.MediaEndpoint1.SetConfiguration() on
```

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:

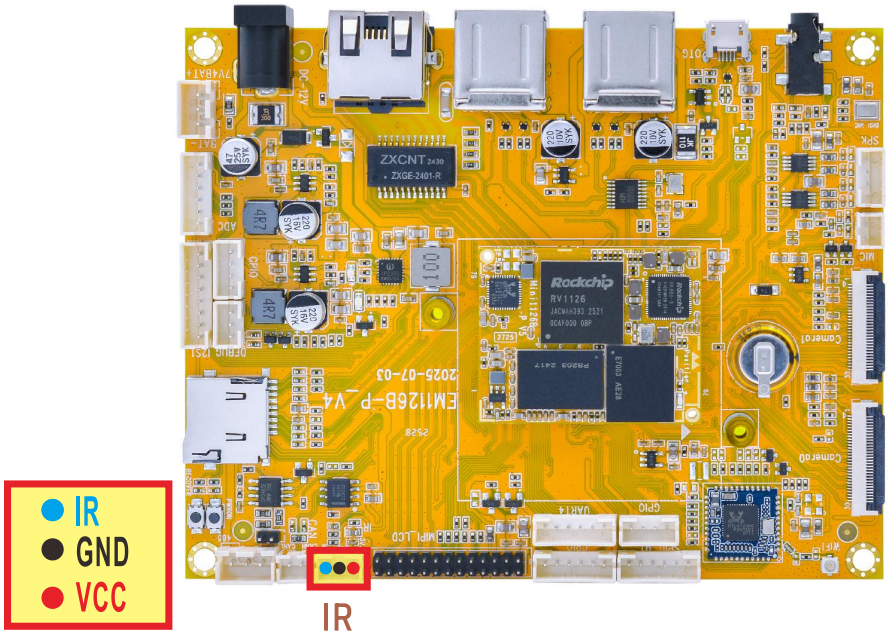
```
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'SPK'
# bluealsa-aplay A8:35:12:9A:EB:4D &
```

```
[liuy]# [liuy]# [liuy]# [liuy]# exit
[liuy]# [liuy]#                  #
#
# rk_mpi_amix_test -C 0 --control 'Playback Path' --value 'SPK'
cmd parse result:
sound control id      : 0
control name         : Playback Path
control value        : SPK
list controls        : 0
list contents        : 0
#
# bluealsa-aplay A8:35:12:9A:EB:4D &
# bluealsa-aplay: [3261] D: aplay.c:846: Creating IO worker A8:35:12:9A:EB:4D
bluealsa-aplay: [3261] D: aplay.c:1194: Starting main loop
bluealsa-aplay: [3262] D: aplay.c:514: Opening BlueALSA source PCM:
/org/bluealsa/hci0/dev_A8_35_12_9A_EB_4D/a2dpsnk/source
bluealsa: [3263] D: dbus.c:47: Called: org.bluealsa.PCM1.Open() on
/org/bluealsa/hci0/dev_A8_35_12_9A_EB_4D/a2dpsnk/source
bluealsa-aplay: [3262] D: aplay.c:540: Starting IO loop
```

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.

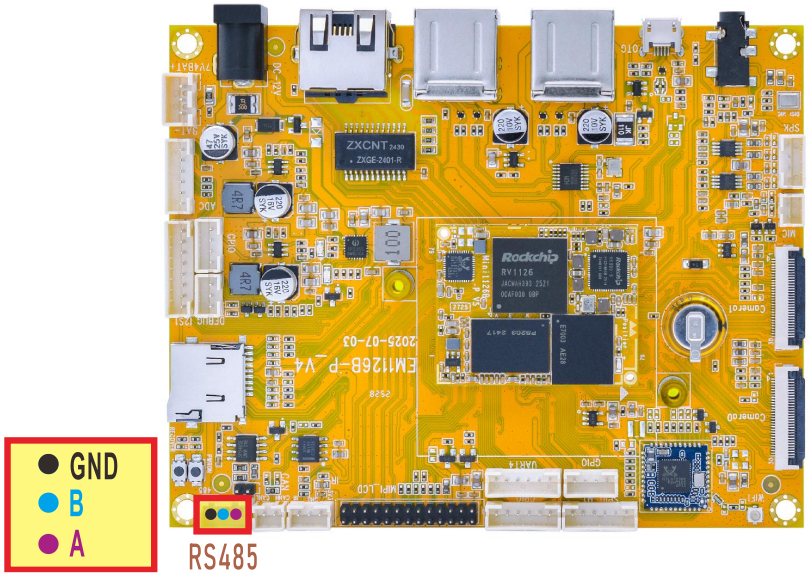
Execute the following command:

```
# 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.

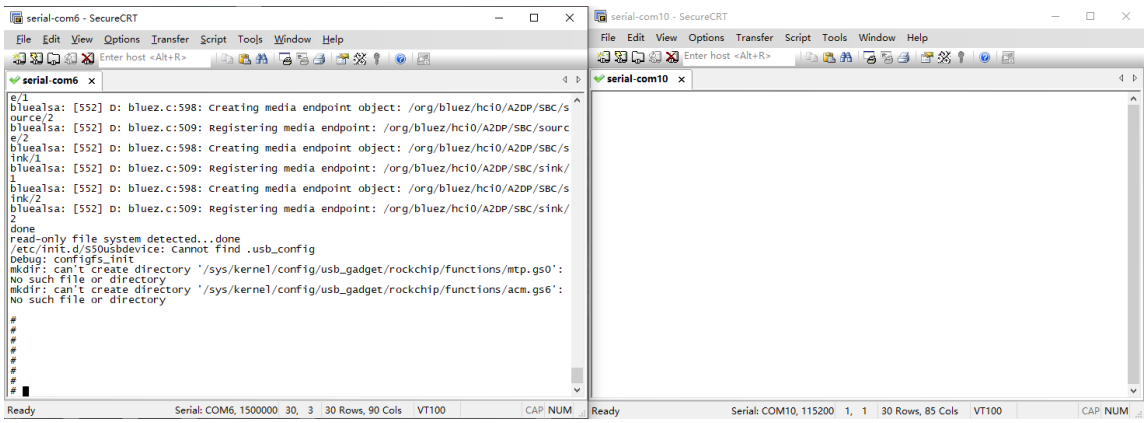
```
# echo 1 > /sys/module/rockchip_pwm_remotectl/parameters/code_print
#
# dmesg | grep -Ei "USERCODE|RMC_GETDATA"
[  925.345525] USERCODE=0x1818
[  925.372634] RMC_GETDATA=98
[  925.745122] USERCODE=0x1818
[  925.772253] RMC_GETDATA=99
[  926.604101] USERCODE=0x1818
[  926.631277] RMC_GETDATA=9a
[  927.023002] USERCODE=0x1818
[  927.050134] RMC_GETDATA=9b
[  927.562497] USERCODE=0x1818
[  927.589573] RMC_GETDATA=99
[  927.862661] USERCODE=0x1818
[  927.889769] RMC_GETDATA=98
[  928.179190] USERCODE=0x1818
[  928.206300] RMC_GETDATA=99
```

6.9 RS485



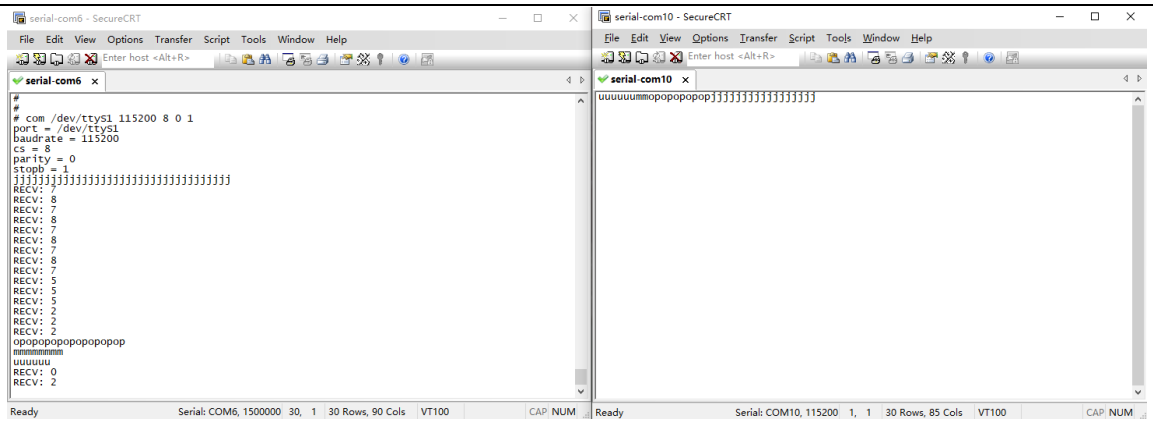
Step 1: Connect the RS485 test tool to the development board as shown in the diagram.

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.



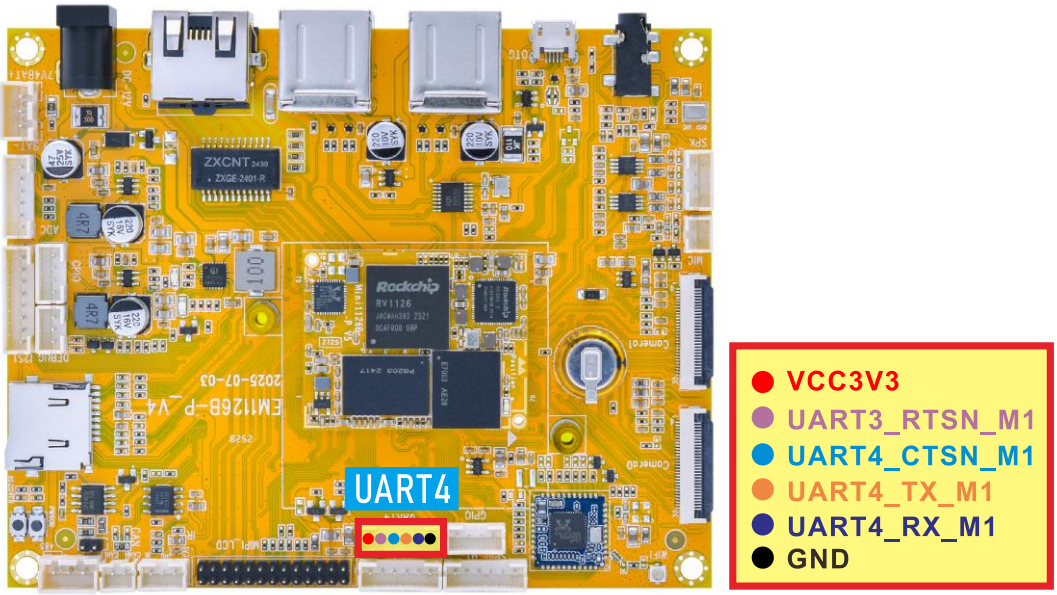
Step 3: Execute the following command on the board to test the RS485 data transmission and reception.

```
# com /dev/ttyS1 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.



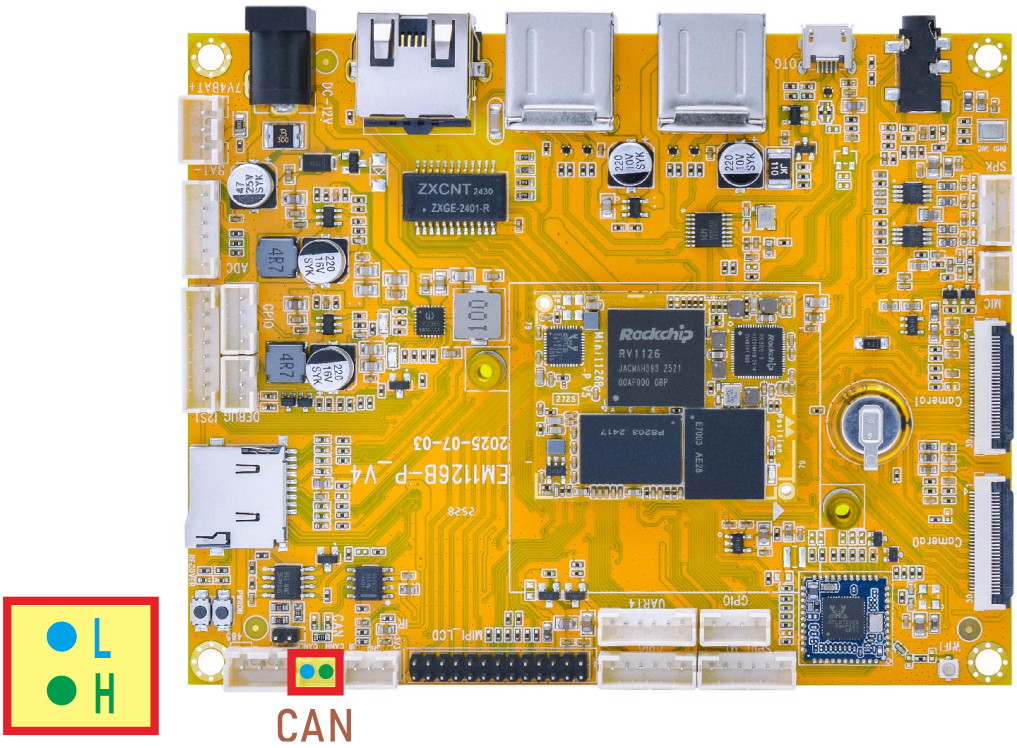
Step 2: Test UART loopback communication.

Execute the following command:

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

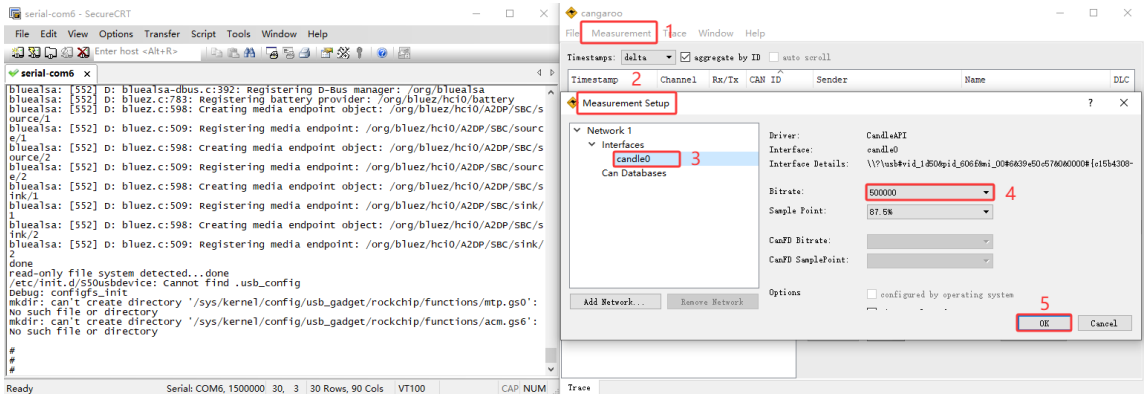
```
# com /dev/ttyS4 115200 8 0 1
port = /dev/ttyS4
baudrate = 115200
cs = 8
parity = 0
stopb = 1
klklklklklklkl
RECV: klklklklklklkl
98989898989
RECV: 98989898989
uiuiuiu
RECV: uiuiuiu
gghghghg232323
RECV: gghghghg232323
jjj
RECV: jjj
```

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.



Step 3: Configure and enable the CAN0 interface on the board with a bitrate of 500000.

Execute the following command:

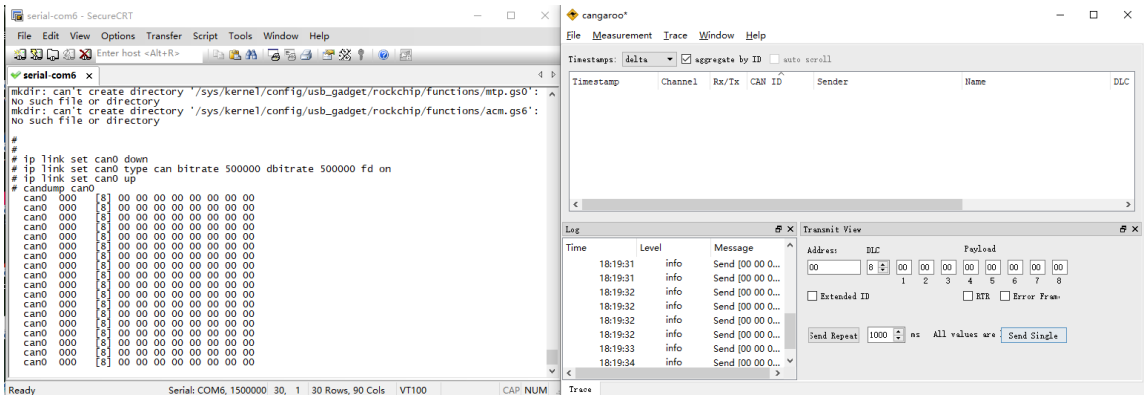
```
# ip link set can0 down
# ip link set can0 type can bitrate 500000 dbitrate 500000 fd on
# ip link set can0 up
```

Step 4: 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
```

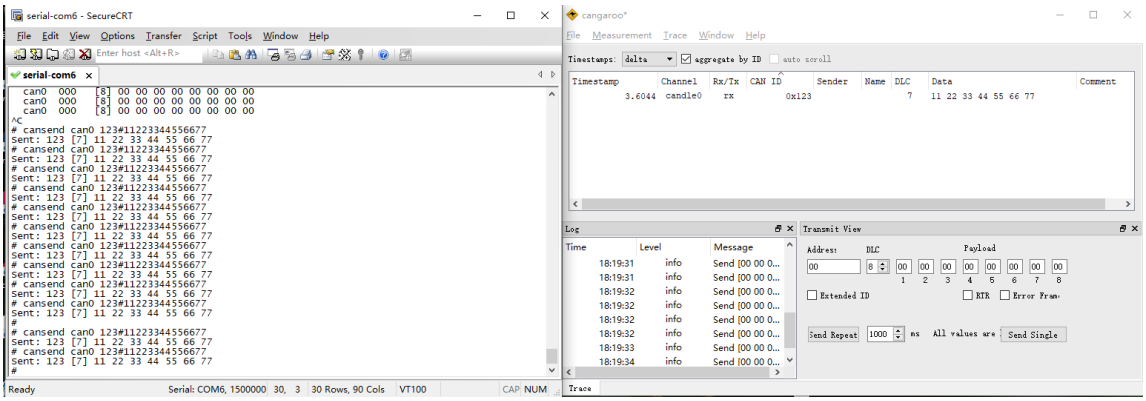


Step 5: Test CAN0 sending.

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

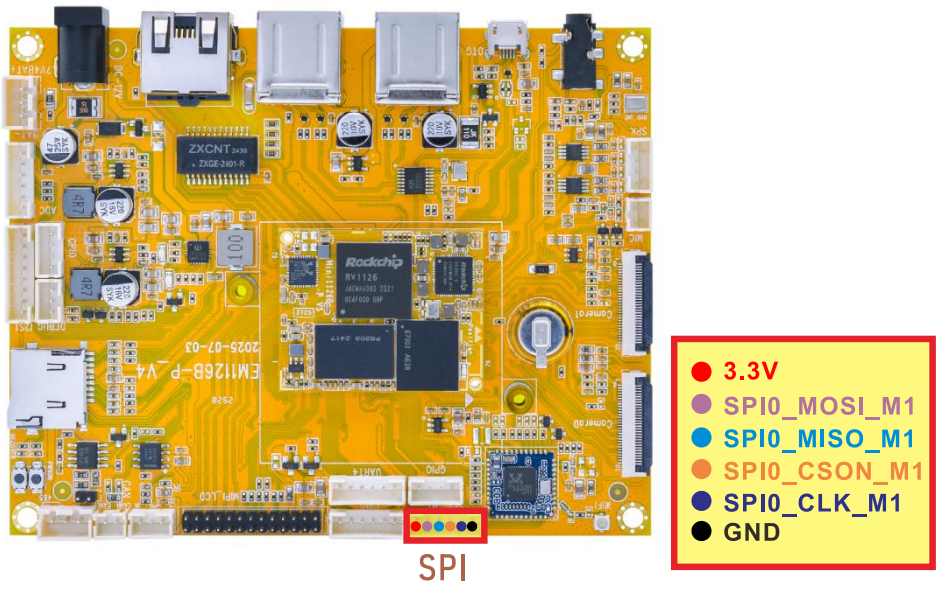
Execute the following command:

```
# cansend can0 123#11223344556677
```



6.12 SPI

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



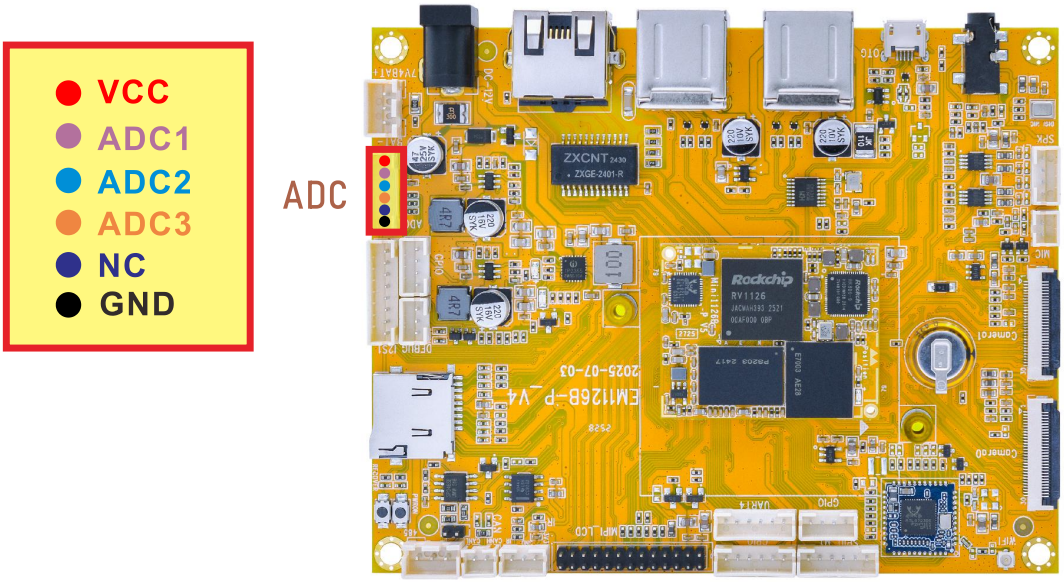
Step 2: Execute the following command to perform the SPI loopback test:

```
# spidev-test -D /dev/spidev0.0 -v
```

```
# spidev-test -D /dev/spidev0.0 -v
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 kHz)
TX | 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 F0 0D
|......@.........................|
RX | 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 F0 0D
|......@.........................|
```

6.13 ADC

The board provides six ADC input interfaces: ADC1, ADC2, ADC3.



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
8104
# cat /sys/bus/iio/devices/iio:\device0/in_voltage1_raw
32
```

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

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

6.14 RTC

The RTC on the EM1126B-P is powered by a supercapacitor.

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

```
# date -s "2026-07-02 17:28:00"
```

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 "2026-07-02 17:28:00"
[ntp.c][rkipc_ntp_update]:connect error
Thu Jul  2 17:28:00 UTC 2026
[audio.c][save_aenc_thread]:fail to get aenc frame
# hwclock -w
# hwclock
Thu Jul  2 17:28:06 2026  0.000000 seconds
# hwclock
Thu Jul  2 17:28:20 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.

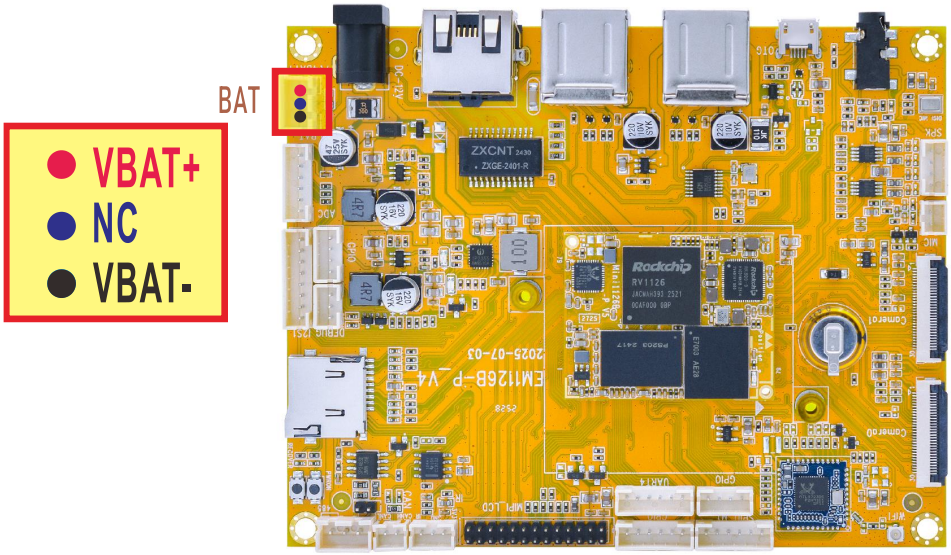
Step 5: Read the hardware clock time again:

```
# hwclock
Thu Jul  2 17:50:08 2026  0.000000 seconds
# hwclock
Thu Jul  2 17:50:20 2026  0.000000 seconds
# hwclock
Thu Jul  2 17:51:09 2026  0.000000 seconds
# hwclock
Thu Jul  2 17:51:11 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.



Step 2: Read the current battery voltage.

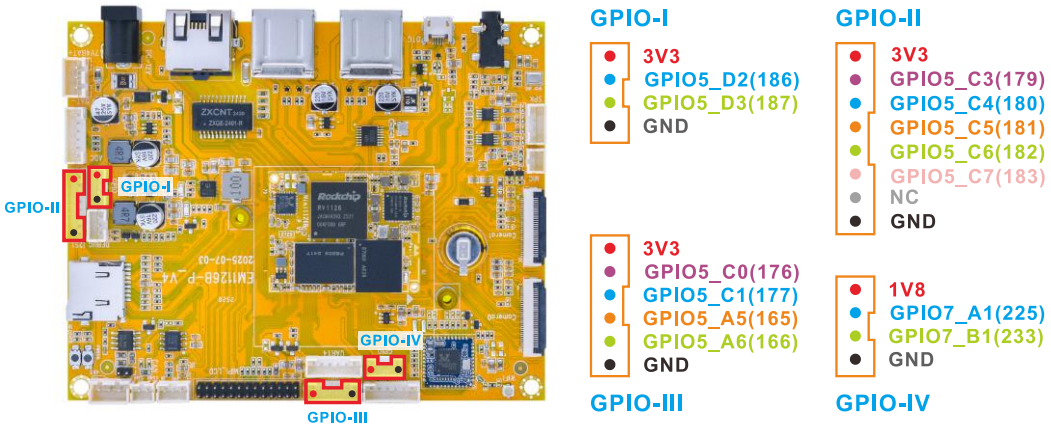
```
# cat /sys/class/power_supply/battery/voltage_now
```

```
# cat /sys/class/power_supply/battery/voltage_now
8424000
```

Note:

Currently, only basic battery status reading is supported. Since the battery capacity curve has not been calibrated, the capacity percentage is for reference only.

6.16 GPIO



General GPIO Control Command Guide：

```
# Export the GPIO and set it as output
gpio_ctrl.sh x export

# Set output to high level
gpio_ctrl.sh x set 1

# Read the current GPIO level
gpio_ctrl.sh x get

# Set output to low level (should be 1)
gpio_ctrl.sh x set 0

# Read the GPIO level again (should be 0)
gpio_ctrl.sh x get

# Unexport GPIO
gpio_ctrl.sh x unexport
```

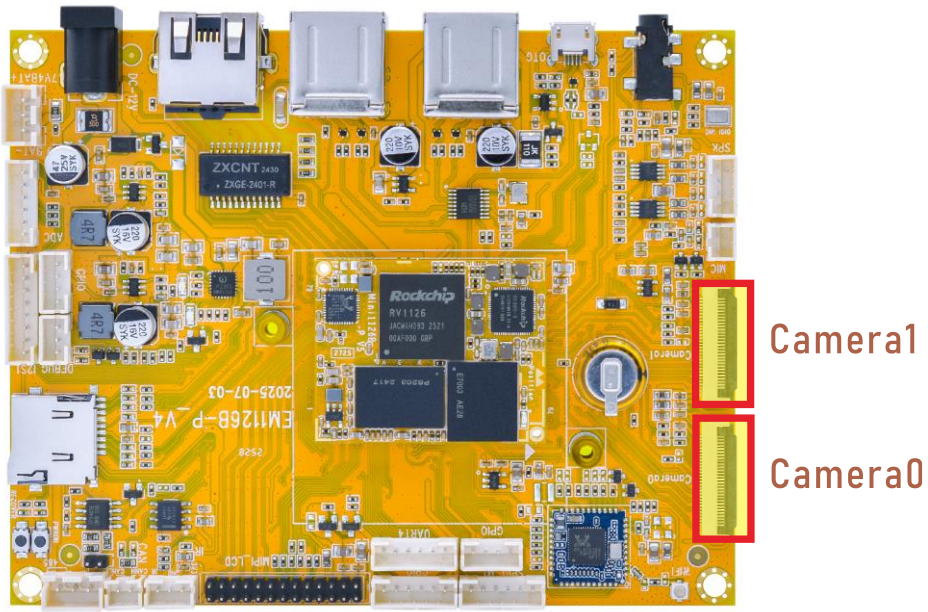
For example, using GPIO5_D2:

```
# gpio_ctrl.sh 186 export
# gpio_ctrl.sh 186 set 1
# gpio_ctrl.sh 186 get
# gpio_ctrl.sh 186 set 0
# gpio_ctrl.sh 186 get
# gpio_ctrl.sh 186 unexport
```

```
# gpio_ctrl.sh 186 export
GPIO186 exported as out
#
# gpio_ctrl.sh 186 set 1
GPIO186 set to 1
#
# gpio_ctrl.sh 186 get
GPIO186 value: 1
#
# gpio_ctrl.sh 186 set 0
GPIO186 set to 0
#
# gpio_ctrl.sh 186 get
GPIO186 value: 0
#
# gpio_ctrl.sh 186 unexport
GPIO186 unexported
```

6.17 Camera

Step 1: Connect the camera modules(IM415), then power on the board.



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 32:D8:C6:1C:4B:9F
          inet addr:192.168.0.182  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:76 errors:0 dropped:12 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8094 (7.9 KiB)  TX bytes:684 (684.0 B)
          Interrupt:108
```

Step 3: Stop the default rkipc service.

Execute the following command:

```
# killall rkipc
```

```
# killall rkipc
[rkipc.c][sig_proc]:received signo 15
# [server.c][rkipc_server_deinit]:rkipc_server_deinit failed
[audio.c][save_aenc_thread]:exit
[video.c][rk_video_deinit]:rk_video_deinit
cmpi            20:12:52-538 {ivs_chn_getResults:1051} get result timeout

[video.c][rkipc_ivs_get_results]:get chn 0 fail -1609203698
[osd.c][osd_time_server]:exit
[video.c][rkipc_osd_bmp_destroy]:rkipc_osd_bmp_destroy
[video.c][rkipc_osd_bmp_destroy]:Destory handle:0 success
[video.c][rkipc_osd_bmp_destroy]:rkipc_osd_bmp_destroy
[video.c][rkipc_osd_bmp_destroy]:Destory handle:1 success
[video.c][rkipc_osd_bmp_destroy]:rkipc_osd_bmp_destroy
[video.c][rkipc_osd_bmp_destroy]:RK_MPI_RGN_Destroy [2] failed with 0xa0038005
[video.c][rkipc_osd_bmp_destroy]:Destory handle:2 success
[video.c][rkipc_osd_bmp_destroy]:rkipc_osd_bmp_destroy
[video.c][rkipc_osd_bmp_destroy]:RK_MPI_RGN_Destroy [3] failed with 0xa0038005
[video.c][rkipc_osd_bmp_destroy]:Destory handle:3 success
[video.c][rkipc_osd_cover_destroy]:rkipc_osd_cover_destroy
[video.c][rkipc_osd_cover_destroy]:RK_MPI_RGN_DetachFrmChn (4) to vi/vpss failed with 0xa0038005
[video.c][rkipc_osd_cover_destroy]:RK_MPI_RGN_DetachFromChn to vi/vpss success
[video.c][rkipc_osd_cover_destroy]:RK_MPI_RGN_Destroy [4] failed with 0xa0038005
```

Step 4: Start the camera demo.

Execute the following command:

```
# sample_demo_dual_camera -s 0 -W 1920 -H 1080 -w 720 -h 576 -f 30 -r 0 -s 1 -W
1920 -H 1080 -w 720 -h 576 -f 30 -r 0 -n 0 -b 1
```

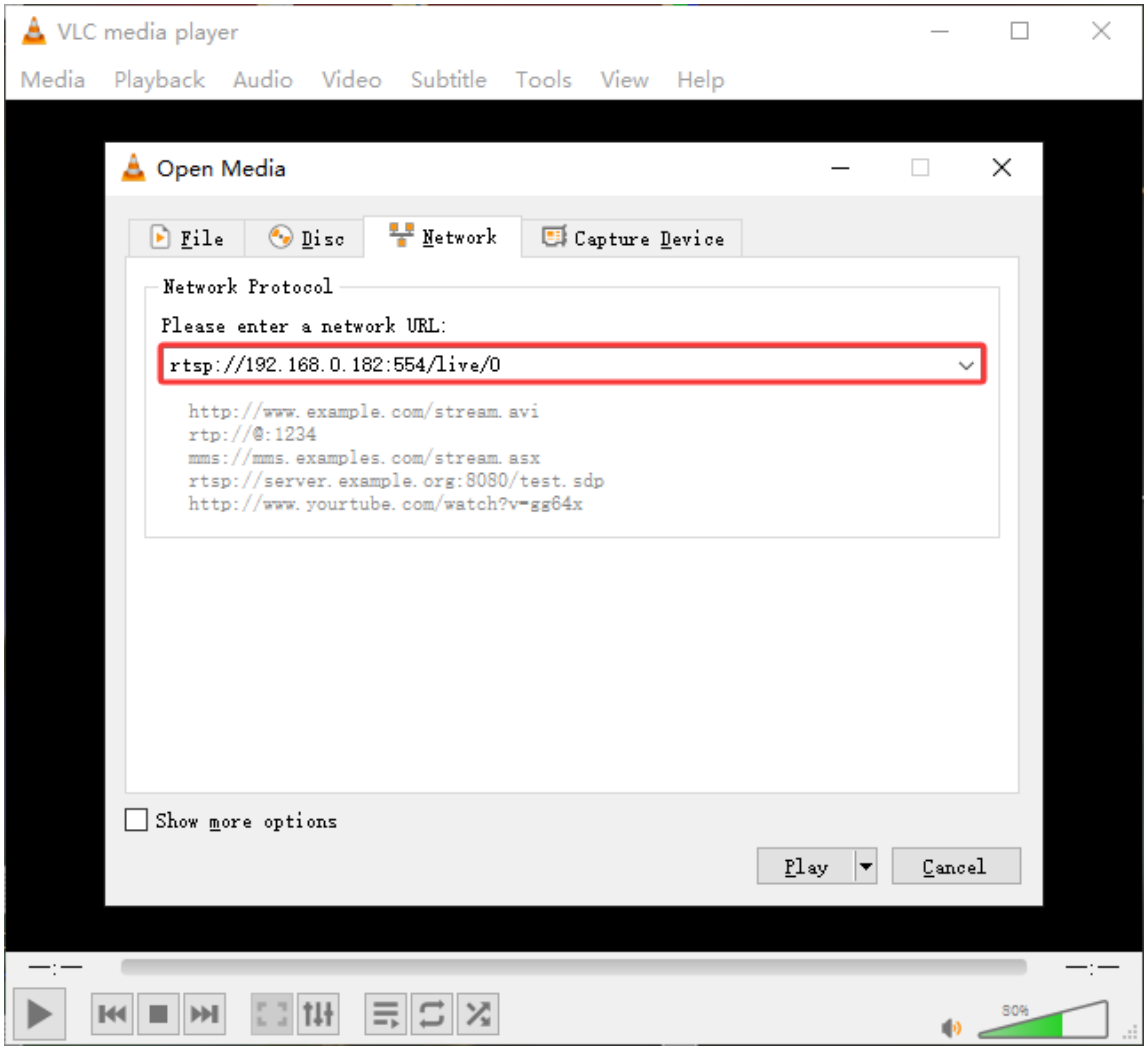
```
# sample_demo_dual_camera -s 0 -W 1920 -H 1080 -w 720 -h 576 -f 30 -r 0 -s 1 -W
1920 -H 1080 -w 720 -h 576 -f 30 -r 0 -n 0 -b 1
sensor 0: main:1920*1080, sub:720*576
sensor 1: main:1920*1080, sub:720*576
#CodecName:H265
#IQ Path: /oem/usr/share/iqfiles
#Rkaiq XML DirPath: /oem/usr/share/iqfiles
#bMultictx: 1

rkaiq log level ff1
XCORE:K:rk_aiq_init_lib, ISP HW ver: 35

************************ VERSION INFOS ************************
version release date: 2026-01-09
        AIQ:      AIQ v6.0x32.1
git logs:
7ed63f87 uapi2_c: fix aiq crash of tuingtool in camgroup mode
9502ee46 iqfiles: isp35: common: fix wrong aibnr en in imx464
b47b562b iqfiles: add imx766 iqfile for 1126B UVC
52163955 iqfiles: add imx464 iqfile for RV1126B AVS
b54d0b87 hwi: fix enh max row for isp33
```

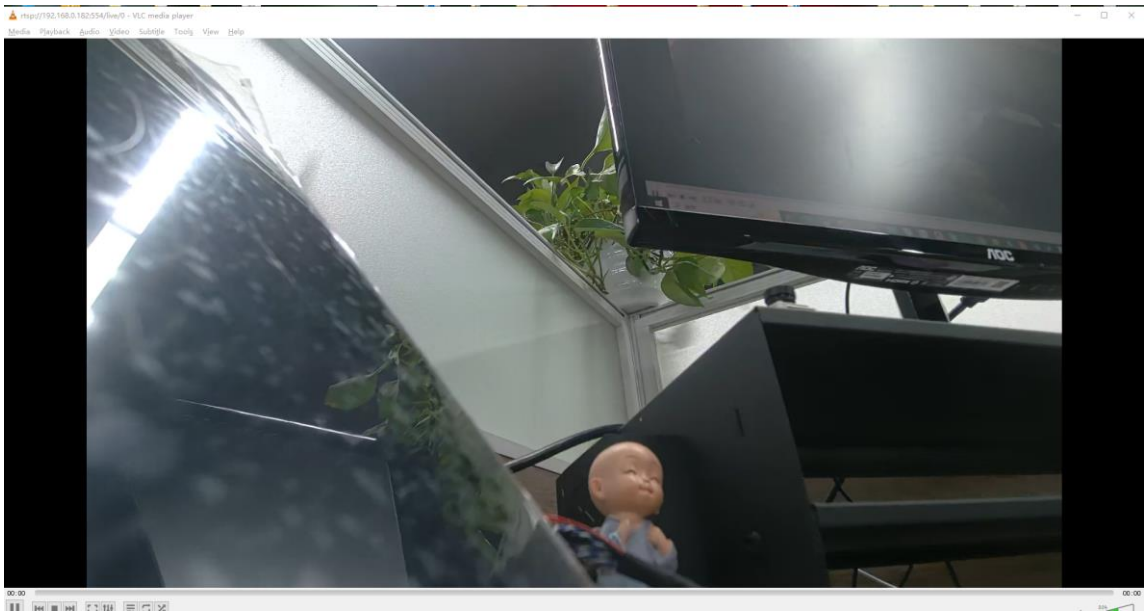
Step 5: Preview the camera stream on the PC.

Open VLC media player on the PC, select **Media** -> **Open Network Stream...**, and enter one of the following RTSP URLs.

Camera0：

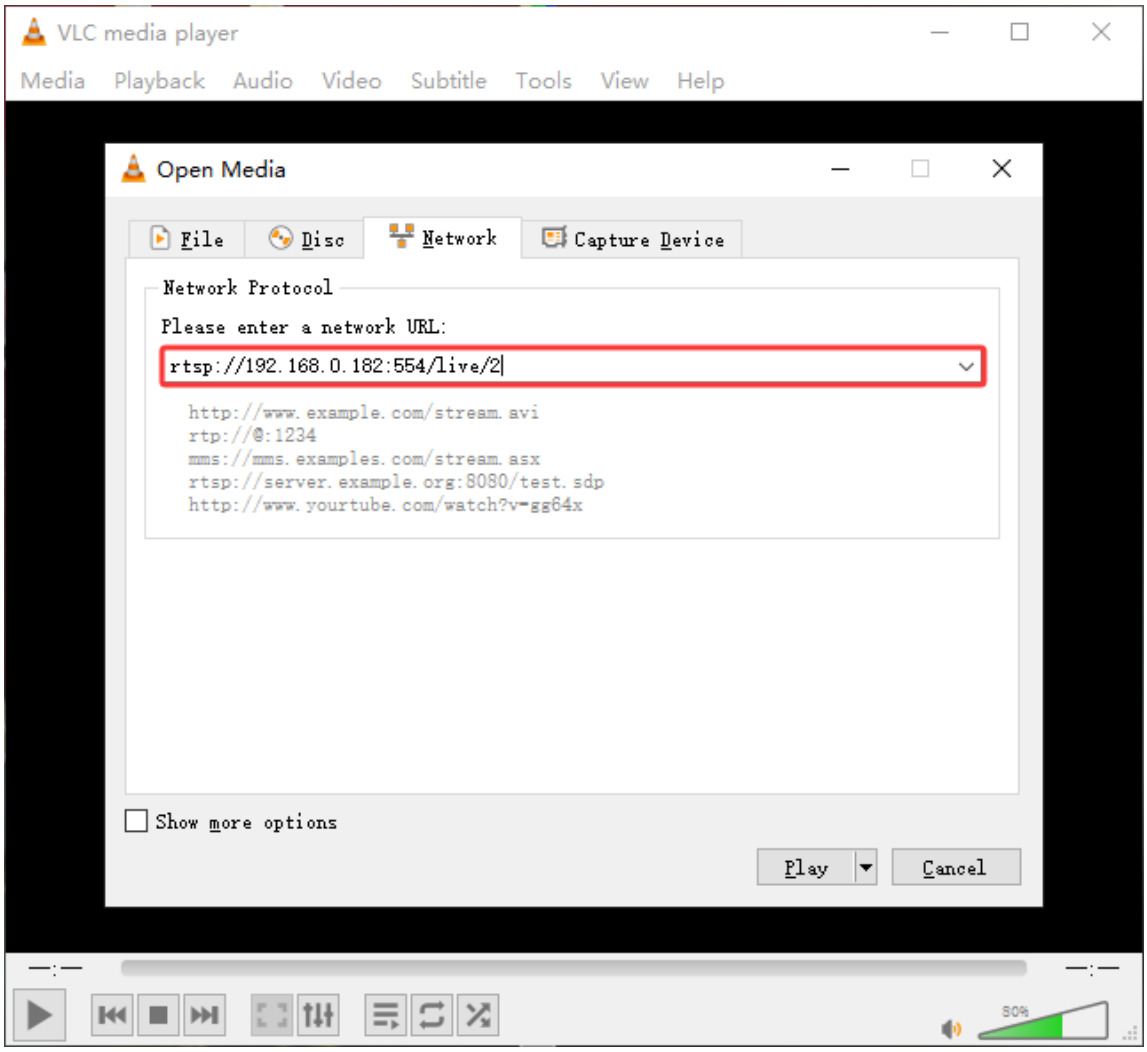
```
rtsp://192.168.0.182:554/live/0
```

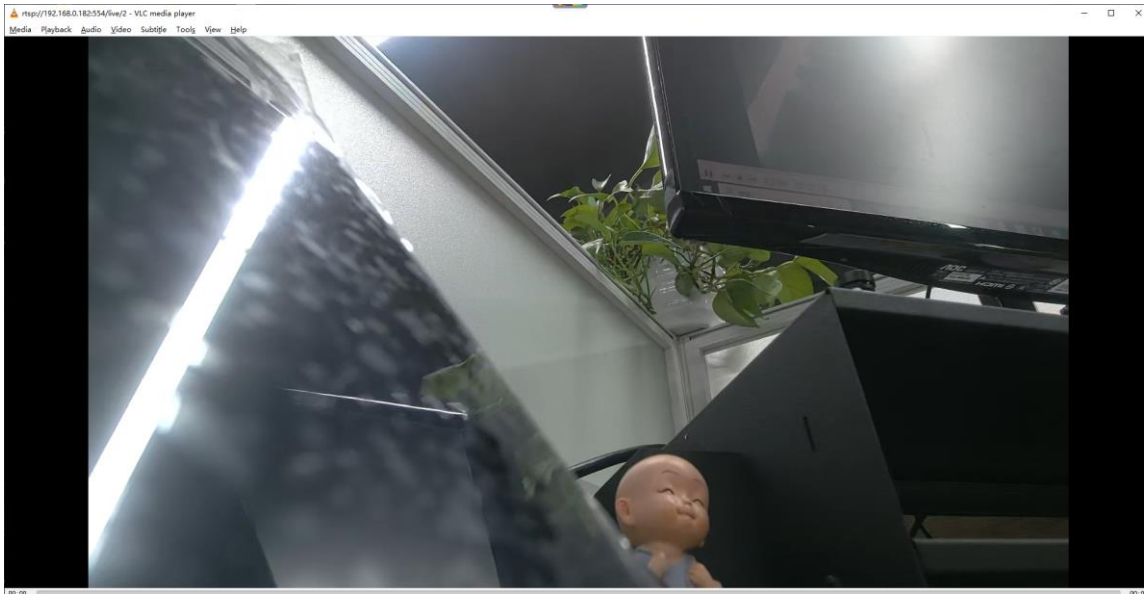




Camera1：

```
rtsp://192.168.0.182:554/live/2
```





Note:

Replace 192.168.0.182 with the actual Ethernet IP address of the board.

7.18 Video Playback

Step 1: Stop the default rkipc service.

Execute the following command:

```
# killall rkipc
```

Step 2: 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)          10:18:11-068 {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)          10:18:11-068 {read_log_level    :083} text is all=4
(null)          10:18:11-068 {read_log_level    :085} module is all, log_level is 4
(null)          10:18:11-069 {dump_version      :060} ----------------------------------------------------
-----
(null)           10:18:11-069 {dump_version      :061} rockit version: git-b85f3c523 Mon Jan 26 09:47:22
2026 +0800
(null)          10:18:11-069 {dump_version      :062} rockit building: built-Chu 2026-01-27 09:45:41
(null)          10:18:11-069 {dump_version      :063} ----------------------------------------------------
-----
(null)          10:18:11-069 {monitor_log_level :141} #Start monitor_log_level thread, arg:(nil)
vsys dev open 4
load library(librga.so) in releative path
mpp[3428]: mpp_info: mpp version: d7dd00e4 author: Herman Chen   2026-02-13 fix[h265d]: Fix invalid hvcc
data crash on seek
mpp[3428]: hal_264d_com: control info: fmt 0, w 1920, h 1080
mpp[3428]: mpp_dec: mpp_dec_proc_cfg found MPP_DEC_SET_FRAME_INFO fmt 0
mpp[3428]: mpp_buf_slot: mismatch h_stride_by_pixel 1984 - 1920
mpp[3428]: mpp_buf_slot: mismatch h_stride_by_byte 1984 - 1920
mpp[3428]: mpp_buf_slot: mismatch size_total 3237888 - 4177920
mpp[3428]: mpp_buf_slot: set frame info: w 1920 h 1080 hor 1920 ver 1088
mpp[3428]: mpp_dec: setting default w 1920 h 1080 h_str 1920 v_str 1088
Layer 0 is successfully bound to device 0.
get a hdl = 0x37e27890
mpp[3428]: mpp_buf_slot: mismatch h_stride_by_pixel 960 - 800
mpp[3428]: mpp_buf_slot: mismatch h_stride_by_byte 960 - 800
mpp[3428]: mpp_buf_slot: mismatch size_total 1843200 - 2048000
get a hdl = 0x37e3a1e0
get a hdl = 0x37e3aa30
load library(librga.so) in releative path
rga_api version 1.10.5_[11]
```

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.