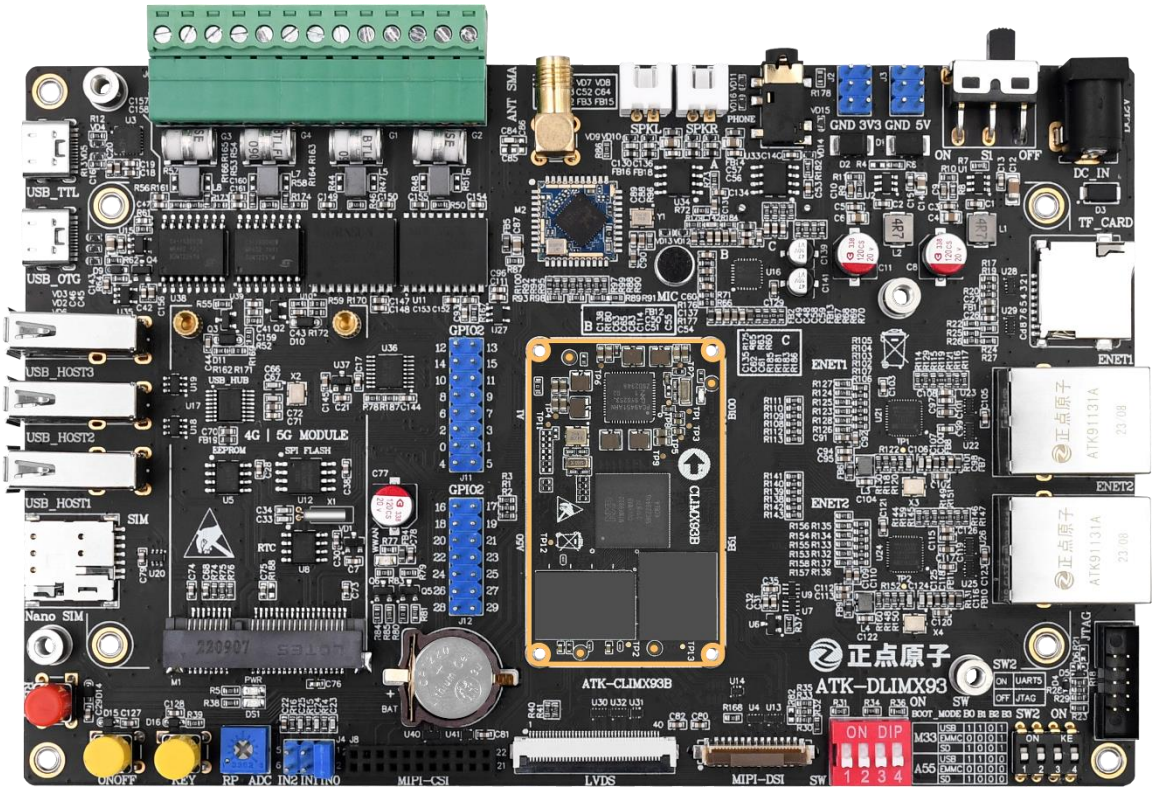
ATK-DLIMX93

Quick test manual

V1.0



ALIENTEK

1. Shopping:

TMALL: https://zhengdianyuanzi.tmall.com

TAOBAO: https://openedv.taobao.com

2. Download

Address: http://www.openedv.com/docs/index.html

3. FAE

Website : www.alientek.com

Forum : http://www.openedv.com/forum.php

Videos : www.yuanzige.com

Fax : +86 - 20 - 36773971

Phone : +86 - 20 - 38271790

Disclaimer

The product specifications and instructions mentioned in this document are for reference only and subject to update without prior notice; Unless otherwise agreed, this document is intended as a product guide only, and none of the representations made herein constitutes a warranty of any kind. The copyright of this document belongs to Guangzhou Xingyi Electronic Technology Co., LTD. Without the written permission of the company, any unit or individual shall not be used for profit-making purposes in any way of dissemination.

In order to get the latest version of product information, please regularly visit the download center or contact the customer service of Taobao ALIENTEK flagship store. Thank you for your tolerance and support.

Revision History:

Version	Version Update Notes	Responsible person	Proofreading	Date
V1.0	release officially	ALIENTEK	ALIENTEK	2024.04.9

Catalogue

Chapter 1. Introduces ATK-DLIMX93 hardware resources 1
1.1 Introduction to base plate resources 1
1.2 Introduction to Core board resources 3
Chapter 2. ATK-DLIMX93 burning system 4
2.1 Introduction to UUU 4
2.2 Using UUU to burn the system to the development board under Ubuntu20.04 4
2.2.1 Development board connection 4
2.2.2 Environment preparation 6
2.2.3 Write firmware 7
2.3 Using UUU to burn the system to TF card under Ubuntu20.04 8
2.3.1 Development board connection 8
2.3.2 Environment preparation 11
2.3.3 Write firmware 11
2.4 Use UUU to write the system to the development board under Windows 12
2.4.1 Development board connection 12
2.4.2 Environment preparation 13
2.4.3 Write firmware 13
2.5 Use UUU to write system to TF card under Window 16
2.5.1 Development board connection 16
2.5.2 Environment preparation 17
2.5.3 Write firmware 17
2.6 Using script to make TF system boot card under Ubuntu20.04 19
2.6.1 Environment preparation 19
2.6.2 Write firmware 20
Chapter 3. Preparation before use of ATK-DLIMX93 24
3.1 Precautions before power on 24
3.2 Serial port software installation 24
3.3 Setting of dial switch and landing development board 26
Chapter 4. ATK-DLIMX93 functional test 31
4.1 LED Testing 31
4.2 KEY Test 31
4.3 MIPI LCD test 33
4.3.1 Instructions for use 33
4.3.2 Touch test 36
4.3.3 Backlight test 36
4.4 LVDS LCD test 37
4.4.1 Instructions for use 37
4.4.2 Touch test 38
4.4.3 Backlight test 38
4.5 Network Testing 39
4.5.1 Network test 39

4.5.2 Gigabit network test 40
4.5.3 SSH login testing 40
4.6 Serial port test 43
4.6.1 RS485 serial port test 43
4.7 CAN Test 46
4.7.1 CAN test 47
4.7.2 CAN FD test 48
4.7.3 Onboard dual-channel CANFD test 50
4.8 USB Testing 51
4.8.1 USB HOST test 51
4.8.2 USB Mouse test 53
4.9 RTC clock test 54
4.10 EEPROM test 55
4.11 SPI FLASH test 55
4.12 Audio testing 56
4.12.1 Playing the audio test 57
4.12.2 Recording test 58
4.13 Onboard SDIO WIFI test 61
4.14 Onboard Bluetooth test 64
4.14.1 Bluetooth pairing 66
4.14.2 Bluetooth file transfer 69
4.15 4G module EC20 test 71
4.15.1 quectel-CM dial-up Internet access 72
4.15.2 pppd dial-up Internet access 74
4.15.3 GPS function test 78
4.16 4G module ME3630-W test 79
4.16.1 ECM goes online 79
4.16.2 pppd dial-up Internet access 81
4.16.3 GPS function test 84
4.17 TF (SD) card test 85
4.17.1 Reading speed test 86
4.17.2 Write Speed testing 86
4.18 OV5645 camera test 86
4.19 ADC test 89
4.20 M33 core test 91
4.21 NPU test 93
Chapter 5. Introduces the function of ATK-DLIMX93 file system 95
5.1 Introduction to filesystem directories 95
5.2 Examining the Qt environment variables 95
5.3 How can I disable Qt desktop startup 96
5.4 How do you create a self-starter 96
5.5 How do I set a static ip address 97

Chapter 1. Introduces ATK-DLIMX93 hardware resources

At present, ALIENTEK has a number of STM32, I.MXRT, Linux and FPGA development boards. These development boards have consistently ranked first in Taobao sales, and have shipped more than 10W sets. This ATK-DLIMX93 development board is a development board that can run Linux system launched by ALIENTEK, using the form of base board + core board. Next, we introduce the resources of the bottom board and core board of the ATK-DLIMX93 development board, respectively.

1.1 Introduction to base board resources

First, let's take a look at the backplane resource diagram of the ATK-DLIMX93 development board, as shown below:

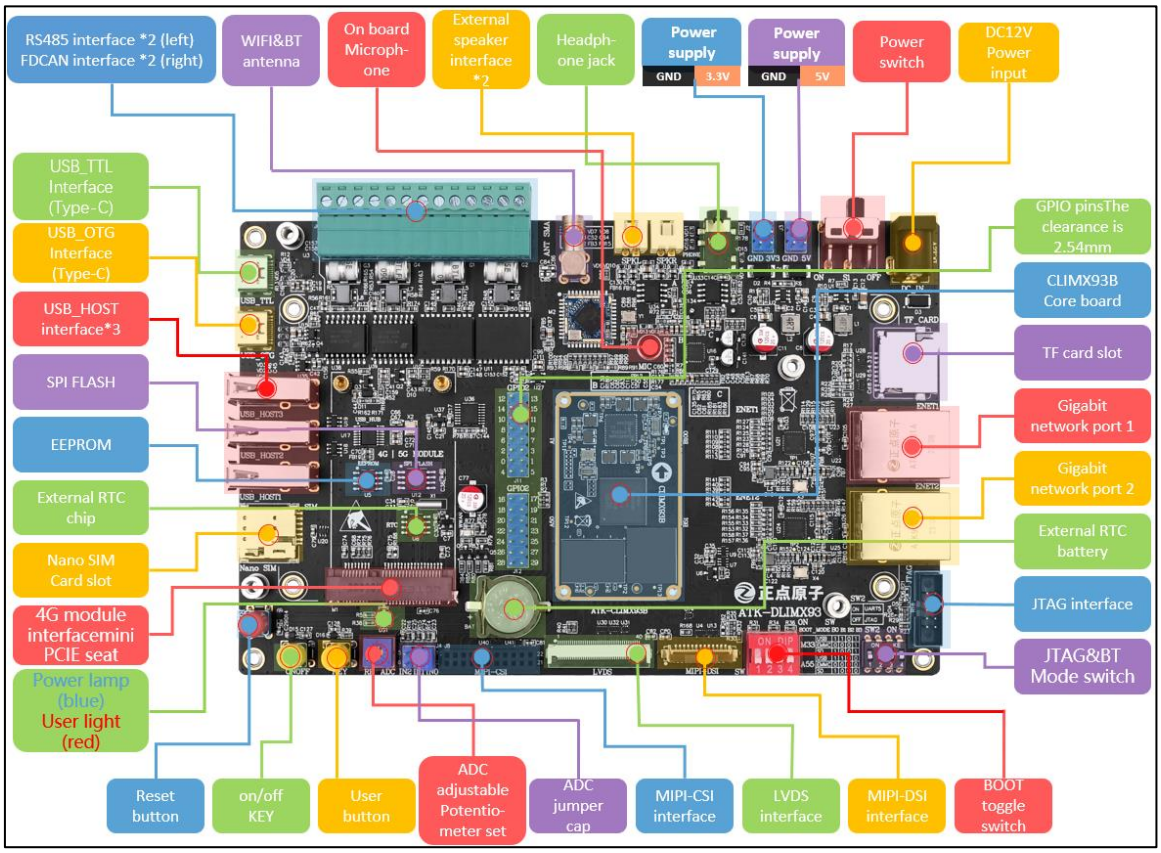


Figure 1.1-1 ATK-DLIMX93 development board base plate

As can be seen from the above figure, ATK-DLIMX93 development board is very rich in resources, expanded with a wealth of interfaces and functional modules. The external size of the development board is 170mm*110mm. The design of the board fully considers the humanized design, which is convenient for development and use.

The on-board resources of the ATK-DLIMX93 development board are as follows:

- 1 set of core board connector interface, supporting ATK-CLIMX93B core board
- 1 blue power indicator
- 1 red status indicator

- 1 NOR FLASH chip, 16MB
- 1 EEPROM chip, 8KB
- 1 RTC clock chip
- 1 high performance audio codec chip, ES8388
- 1 way headphone interface, support 4-segment headphone
- 2-way speaker external interface
- 1 electret microphone MIC
- 1 small speaker (back of board)
- 1 WIFI&BT antenna interface
- 1 SDIO WIFI&BT module
- 2-way CAN-FD isolation interface
- 2-way RS485 isolation interface
- 1 USB_TTL debug serial port, Type-C interface Type
- 1 channel USB_OTG interface, Type-C interface Type
- 3-way USB2.0 HOST interface
- 1 Mini PCIE 4G/5G module interface
- 1 way Nano SIM card interface
- One adjustable potentiometer for ADC test
- 1 way ADC connection interface
- 1 RTC button battery holder with battery
- 1 channel MIPI-CSI camera interface
- 1 MIPI-DSI screen interface
- 1 LVDS screen interface
- 1 start mode dial switch
- 1 JTAG& Bluetooth serial port switch
- 1 way JTAG debug interface
- Two 10M/100M/1000M Ethernet interface (RJ45)
- 1 TF card interface
- 1 channel DC12V power input interface (recommended input voltage range: DC6~16V)
- 1 power switch, control the whole board power supply
- 1 set of 5V power supply outlet
- 1 set of 3.3V power supply outlet
- 1 reset button
- 1 switch button
- 1 user button
- 1 group of 2×8P 2.54mm spacing row needles, leading to 16 IO
- 1 set of 2×7P 2.54mm spacing row needles, eliciting 14 IO

The features of the ATK-DLIMX93 development board include:

1) Rich interfaces. The development board provides more than ten kinds of standard interfaces, which can conveniently carry out the experiment and development of various peripherals.

2) Flexible design. Adopting the form of core board + bottom board, many resources on the board can be flexibly configured to meet the use of different conditions. The core board brings out all the

communication interfaces of i.MX93, including UART, I2C, SPI, CANFD, SDMMC, USB2.0, MIPI-DSI, MIPI-CSI, LVDS, ADC, Gigabit Ethernet, etc., which is greatly convenient for everyone to expand and use.

3) Abundant resources. The onboard high-performance audio codec chip, gigabit network card, EEPROM memory chip and various interface chips meet a variety of application requirements.

4) Humanized design. Each interface is marked with screen printing, which is clear at a glance; Some commonly used peripherals are marked with large screen printing for easy search; The interface position is designed reasonably and easy to use.

1.2 Introduction to Core board resources

Let's look at the ATK-CLIMX93B core board as shown below:

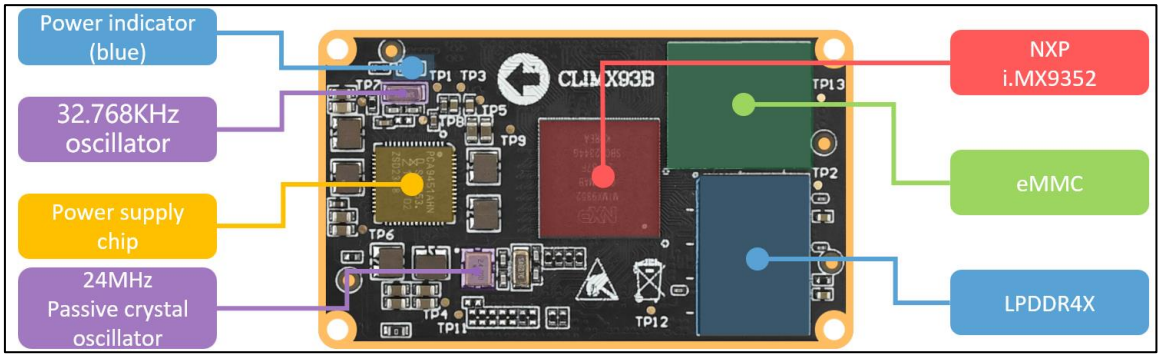


Figure 1.2-1 ATK-CLIMX93B core board

The overall size of the core board is 50mm*30mm, which is very small. It adopts 0.4mm board to board connector, and the height of the base board BTB connector is 1.5mm, which can be easily applied to various projects.

The on-board resources for the ATK-CLIMX93B core board are as follows:

- CPU: MIMX9352CVVXMAB, dual-core Cortex-A55(1.7GHz), single-core Cortex-M33 (250MHz), FCBGA306 package, 11x11mm.
- Memory LPDDR4: capacity of 1GB.
- Storage EMMC: capacity of 16GB.
- High stability power management PMIC.
- Two 2*50 BTB connector male seats, a total of 200PIN.

Features of the ATK-CLIMX93B core board include:

1) Small size. The core board is only 50mm*30mm in size, easy to use to a variety of projects.

2) Convenient integration. The core board uses two 2*50P BTB connectors, which can be very easily integrated into the customer PCB, easy replacement, convenient maintenance and testing.

3) Abundant resources. The core board has 1GB LPDDR4 memory and 16GB EMMC memory onboard, which can meet a variety of application requirements.

4) Stable performance. The core board adopts 8-layer board design, separate layer and power layer, and all key signals are wired with equal length to ensure stable and reliable operation.

5) Humanized design. According to the function of the partition of IO port, convenient wiring.

Chapter 2. ATK-DLIMX93 burning system

The development board or core board will be uniformly burned before delivery, and the system can be started after power on.

If you need to re-burn the factory firmware, you can download the latest factory system firmware from the ALIENTEK data download center, and use the UUU burn tool to burn to the development board or core board.

2.1 Introduction to UUU

Universal Update Utility (UUU) is an open source tool provided by NXP Corporation, also known as MFGTools v3, to perform firmware and software updates on NXP processor platforms. It has cross-platform capabilities and supports multiple operating systems such as Windows, Linux, and macOS. The UUU can be used with NXP i.MX series processors and other NXP chipsets and development boards. UUU has a flexible command-line interface that automates the upgrade process through scripts. It supports device connection and firmware burning through USB communication interface.

The version of UUU used in this document is 1.5.163.

The official download address https://github.com/nxp-imx/mfgtools/releases/tag/uuu_1.5.163

2.2 Using UUU to burn the system to the development board under Ubuntu20.04

2.2.1 Development board connection

Set the dial switch of the development board to USB burn mode (corresponding to code position 1100).

The USB_TTL interface of the development board is connected to the USB port of the computer through the Type-C cable, which is used to output print information when burning. If the virtual machine is enabled, switch the USB serial port to the Windows host. This item is only useful for checking information when writing errors.

The USB_OTG interface of the development board is connected to the USB3.0 port of the Ubuntu virtual machine through the Type-C cable. Please switch and fix the USB device to the Ubuntu virtual machine side for burning the system.

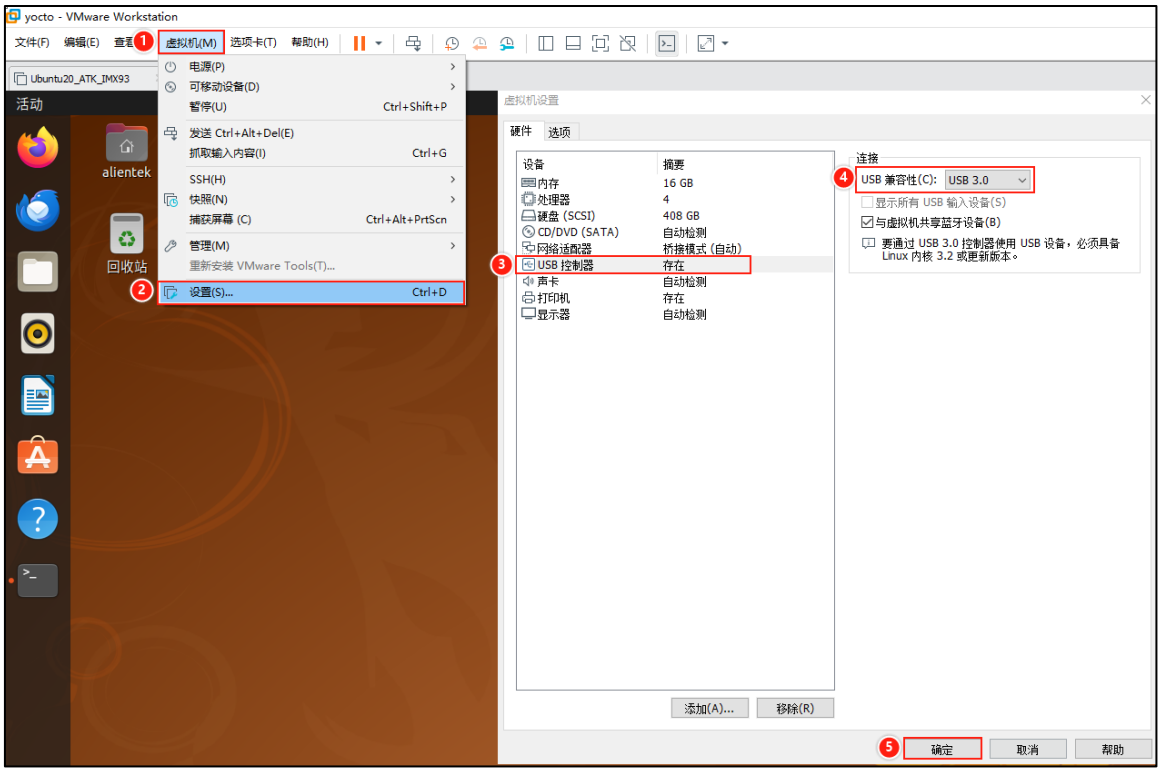


Figure 2.2-1 Set USB interface of virtual machine compatible with USB3.0

During the burning process, the development board will restart, and the computer may automatically connect the USB to the Windows or ask the USB access device. During this period, the UUU may prompt the burning failure. Therefore, it is recommended to connect the USB option of the virtual machine to the Ubuntu virtual machine by default before burning. This part of the function is provided by VMware. If the burning fails all the time, you can try the method in section 2.3 to close the virtual machine and burn under Windows.

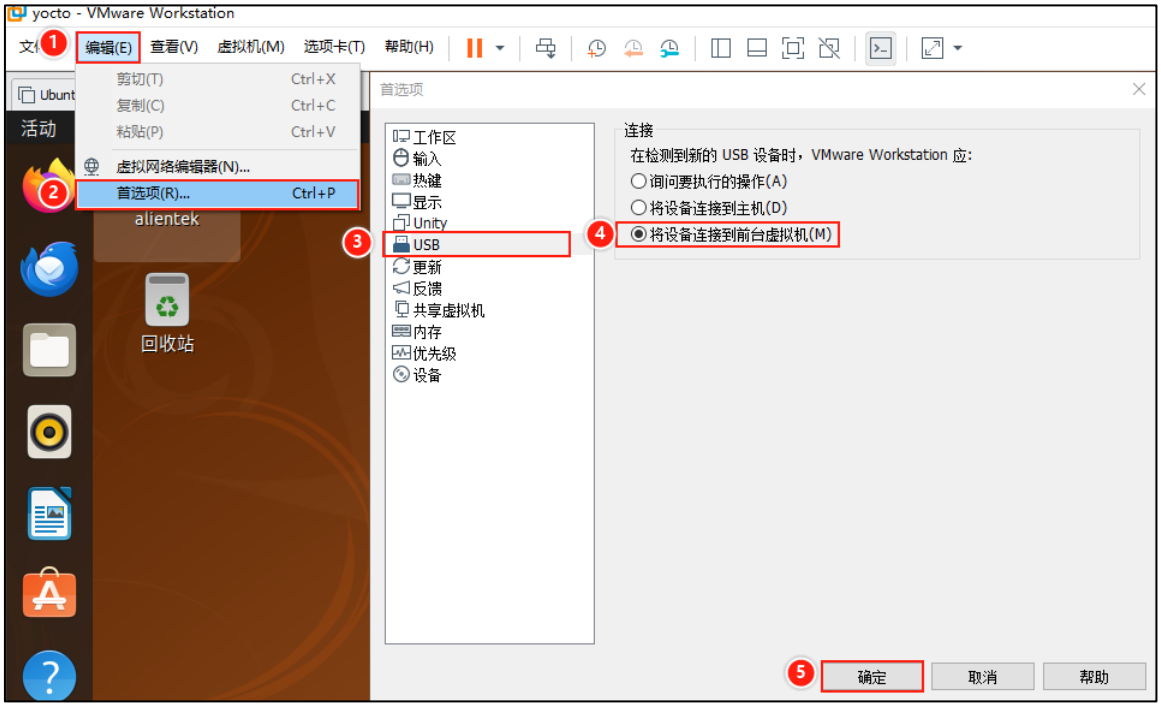


Figure 2.2-2 Set the default USB connection to Vmware

The connection status of the development board is shown as follows:

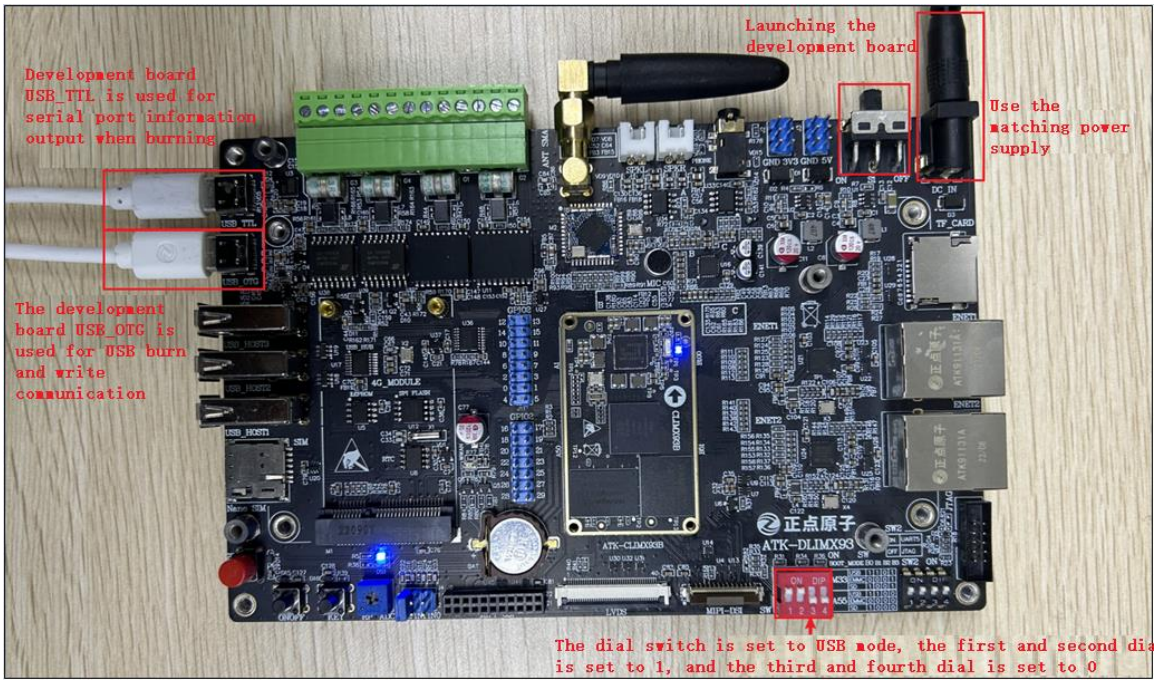


Figure 2.2-3 USB burning when the development board is connected

2.2.2 Environment preparation

In this section, the author uses Ubuntu20.04 system in VMware virtual machine on Windows computer for UUU burning. The environment version is as follows, for reference only.

Environment/software/interface	Version
Computer architecture	X86
Host computer system	Windows10
VMware software	15.5.0
Linux Distributions	Ubuntu20.04
USB interface	USB3.0

Packages that need to be installed in Ubuntu20.04:

```
sudo apt-get install libusb-1.0.0-dev libzip-dev libbz2-dev libtinyxml2-dev
```

2.2.3 Write firmware

Development board firmware package is located in: development board CD A - Basic information \8_system_image

Copy the board firmware package to Ubuntu and check the firmware package directory as shown below:

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ tree -L 3
.
├── 版本记录.txt
├── alientek-imx93-emmc.uuu
├── alientek-imx93-sd.uuu
├── boot
│   ├── extlinux-emmc
│   │   └── extlinux.conf
│   ├── extlinux-sd
│   │   └── extlinux.conf
│   ├── flash.bin
│   ├── Image
│   ├── imx93-11x11-atk.dtb
│   ├── imx93-11x11-atk-lvds-10.1-1280x800.dtb
│   ├── imx93-11x11-atk-mipi-dsi-10.1-800x1280.dtb
│   ├── imx93-11x11-atk-mipi-dsi-5.5-1080x1920.dtb
│   └── imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
├── filesystem
│   └── rootfs.tar.zst
├── fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot
├── imx93mksdboot.sh
├── modules
│   └── modules.tar.bz2
├── uuu
└── uuu.exe
```

Figure 2.2-4 Firmware package directory

We will use the uuu tool in Ubuntu20.04 to burn the firmware of the development board. Execute the following command in the terminal to burn the firmware. The uuu program can be copied to Ubuntu's /usr/bin directory for easier use. Use the chmod command to change the permissions of the burn script alientek-imx93-emmc.uuu, and run the burn script.

```
sudo cp uuu /usr/bin
sudo chmod +x alientek-imx93-emmc.uuu
sudo uuu alientek-imx93-emmc.uuu
```

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo cp uuu /usr/bin
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo chmod +x alientek-imx93-emmc.uuu
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo uuu alientek-imx93-emmc.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 0    Failure 0


3:21    24/31 [=>                6%                  ] FBK: ucp filesystem/rootfs.tar.zst t:-
3:22    15/15 [                                      ] acmd ${kboot} ${loadaddr} ${initrd_addr} ${fdt_addr}
```

Figure 2.2-5 Burnable firmware

After burning fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot, the development board will restart the system once. During this process, pay attention to keep the USB of the development board connected to the virtual machine all the time, otherwise the burning will report that the USB device cannot be opened and will be interrupted. The system needs to be re-burned, and the development board needs to be restarted or reset before re-burning.

Use the uuu program to run alientek-imx93-emmc.uuu burn script to burn the firmware to the EMMC storage of the development board. The burning speed of uuu is fast. After the final burning is completed, the command terminal is shown as follows:

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo uuu alientek-imx93-emmc.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 1    Failure 0


3:21    31/31 [Done                                  ] FBK: DONE
3:22    15/15 [                                      ] nitrd_addr} ${fdt_addr}

alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$
```

Figure 2.2-6 burning success

After burning, set the dial switch to EMMC mode (code bit 0000) to start the development board and enter the system.

2.3 Using UUU to burn the system to TF card under Ubuntu20.04

2.3.1 Development board connection

Set the dial switch of the development board to USB burn mode (corresponding to code position 1100).

The USB_TTL interface of the development board is connected to the USB port of the computer through the Type-C cable, which is used to output print information when burning. If the virtual machine is enabled, switch the USB serial port to the Windows host. This item is only useful for checking information when writing errors.

The USB_OTG interface of the development board is connected to the USB3.0 port of the Ubuntu virtual machine through the Type-C cable. Please switch and fix the USB device to the Ubuntu virtual machine side for burning the system.

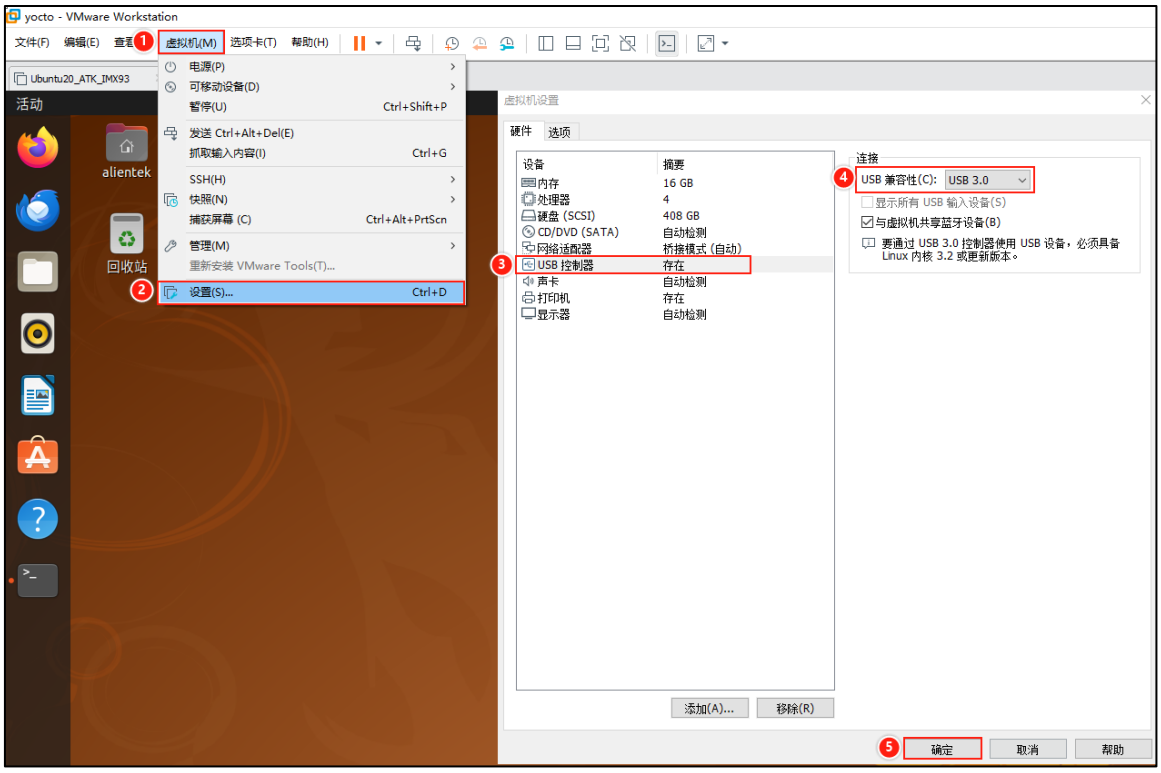


Figure 2.3-1 Set USB interface of virtual machine compatible with USB3.0

During the burning process, the development board will restart, and the computer may automatically connect the USB to the Windows or ask the USB access device. During this period, the UUU may prompt the burning failure. Therefore, it is recommended to connect the USB option of the virtual machine to the Ubuntu virtual machine by default before burning. This part of the function is provided by VMware. If the burning fails all the time, you can try the method in subsection 2.5 to close the virtual machine and burn under Windows.

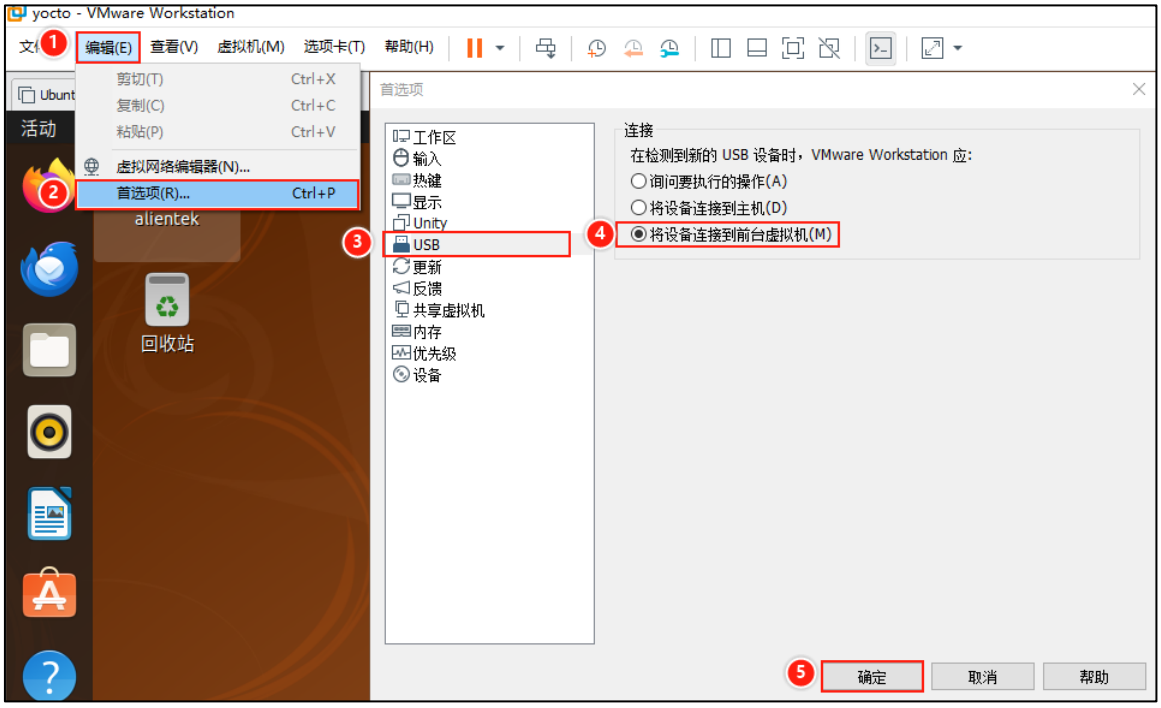


Figure 2.3-2 Set the default USB connection to Vmware

Prepare a high-speed TF card with a capacity of more than 8GB and format it into FAT32 format. Connect the TF card to the TF card slot of the development board, set the dial switch of the development board to USB burning mode 1100, and power on the development board. The connection status of the development board is shown in the following figure:

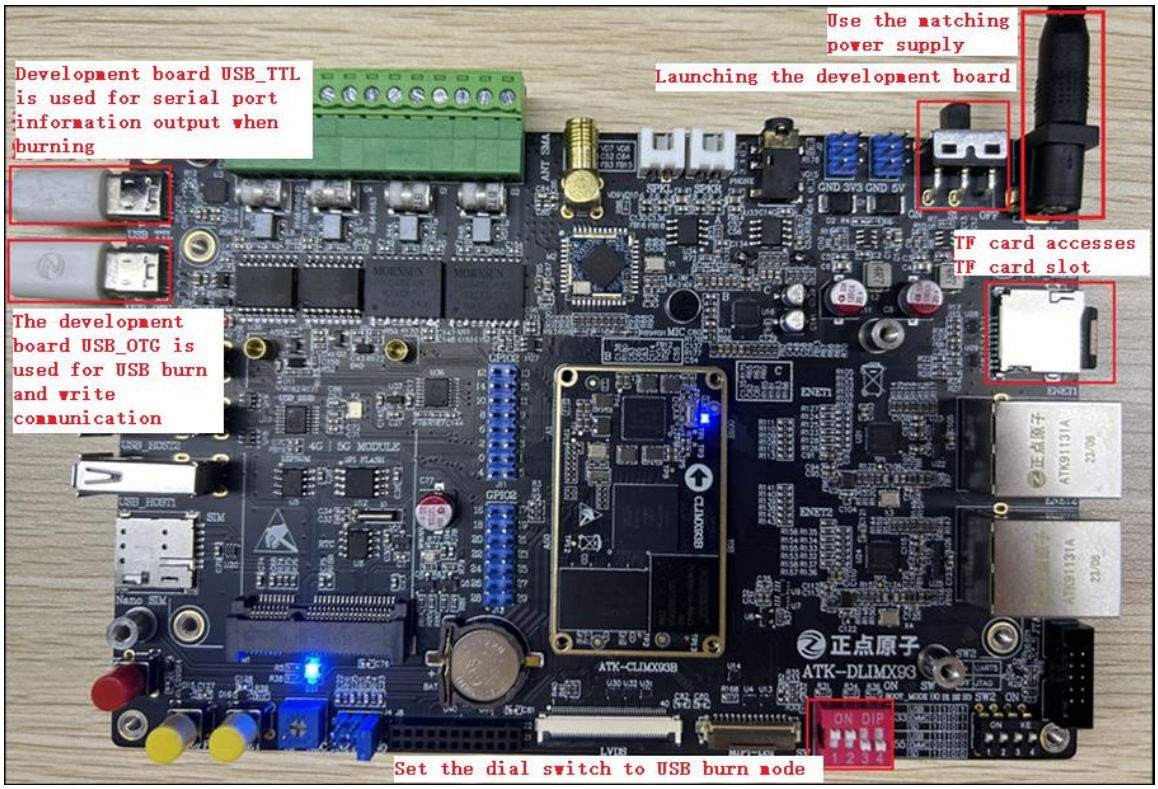


Figure 2.3-3 Hardware connection

2.3.2 Environment preparation

In this section, the author uses Ubuntu20.04 system in VMware virtual machine on Windows computer for UUU burning. The environment version is as follows, for reference only.

Environment/software/interface	Version
Computer architecture	X86
Host computer system	Windows10
VMware software	15.5.0
Linux Distributions	Ubuntu20.04
USB interface	USB3.0

Packages to install in Ubuntu20.04:

```
sudo apt-get install libusb-1.0.0-dev libzip-dev libbz2-dev libtinyxml2-dev
```

2.3.3 Write firmware

Development board firmware package is located in: development board CD A - Basic information \8_system_image

Copy the board firmware package to Ubuntu and check the firmware package directory as shown below:

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ tree -L 3
.
├── 版本记录.txt
├── alientek-imx93-emmc.uuu
├── alientek-imx93-sd.uuu
├── boot
│   ├── extlinux-emmc
│   │   └── extlinux.conf
│   ├── extlinux-sd
│   │   └── extlinux.conf
│   ├── flash.bin
│   ├── Image
│   ├── imx93-11x11-atk.dtb
│   ├── imx93-11x11-atk-lvds-10.1-1280x800.dtb
│   ├── imx93-11x11-atk-mipi-dsi-10.1-800x1280.dtb
│   ├── imx93-11x11-atk-mipi-dsi-5.5-1080x1920.dtb
│   └── imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
├── filesystem
│   └── rootfs.tar.zst
├── fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot
├── imx93mksdboot.sh
├── modules
│   └── modules.tar.bz2
├── uuu
└── uuu.exe
```

Figure 2.3-4 Firmware package directory

We will use the uuu tool in Ubuntu20.04 to burn the firmware of the development board. Execute the following command in the terminal to burn the firmware. The uuu program can be copied to Ubuntu's /usr/bin directory for easier use. Use the chmod command to change the permissions of the burn script alientek-imx93-sd.uuu, and run the burn script.

```
sudo cp uuu /usr/bin
sudo chmod +x alientek-imx93-sd.uuu
sudo uuu alientek-imx93-sd.uuu
```

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo cp uuu /usr/bin
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo chmod +x alientek-imx93-sd.uuu
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo uuu alientek-imx93-sd.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 0    Failure 0


3:21    23/30 [=>                7%                  ] FBK: ucp filesystem/rootfs.tar.zst t:-
3:22    10/10 [                                      ] itrd_addr} ${fdt_addr}
```

Figure 2.3-5 Burnable firmware

After burning fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot, the development board will restart the system once. During this process, pay attention to keep the USB of the development board connected to the virtual machine all the time, otherwise the burning will report that the USB device cannot be opened and will be interrupted. The system needs to be re-burned, and the development board needs to be restarted or reset before re-burning.

Use the uuu program to run the alientek-imx93-sd.uuu burn script to burn the firmware to the TF card of the development board. The writing speed of uuu is fast, but the writing speed is linked to the quality of TF card (compared with EMMC, TF card writing speed is slower). After the final writing is completed, the command terminal is shown as follows:

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo uuu alientek-imx93-sd.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 1    Failure 0


3:21    30/30 [Done                                  ] FBK: DONE
3:22    10/10 [                                      ] d ${kboot} ${loadaddr} ${initrd_addr} ${fdt_addr}
```

Figure 2.3-6 Complete burn

After burning, set the dial switch to SD start mode (code point 1000) to start the development board and enter the system.

2.4 Use UUU to write the system to the development board under Windows

2.4.1 Development board connection

Set the dial switch of the development board to USB burn mode (corresponding to code position 1100).

The USB_TTL interface of the development board is connected to the USB port of the computer through the Type-C cable, which is used to output print information when burning. If the virtual machine is enabled, switch the USB serial port to the Windows host. This item is only useful for checking information when writing errors.

The USB_OTG interface of the development board is connected to the USB3.0 port of the Windows computer through the Type-C cable for burning the system. If the virtual machine is enabled, fix the USB device switch to the Windows host.

During the burning process, the development board will be restarted, and the computer may automatically connect the USB to the virtual machine or ask the USB connected device. During this period, the UUU may prompt the burning failure. Therefore, it is recommended to set the USB option of the virtual machine to the Windows host by default before burning, or to shut down the virtual machine before burning.

The connection status of the development board is shown as follows:

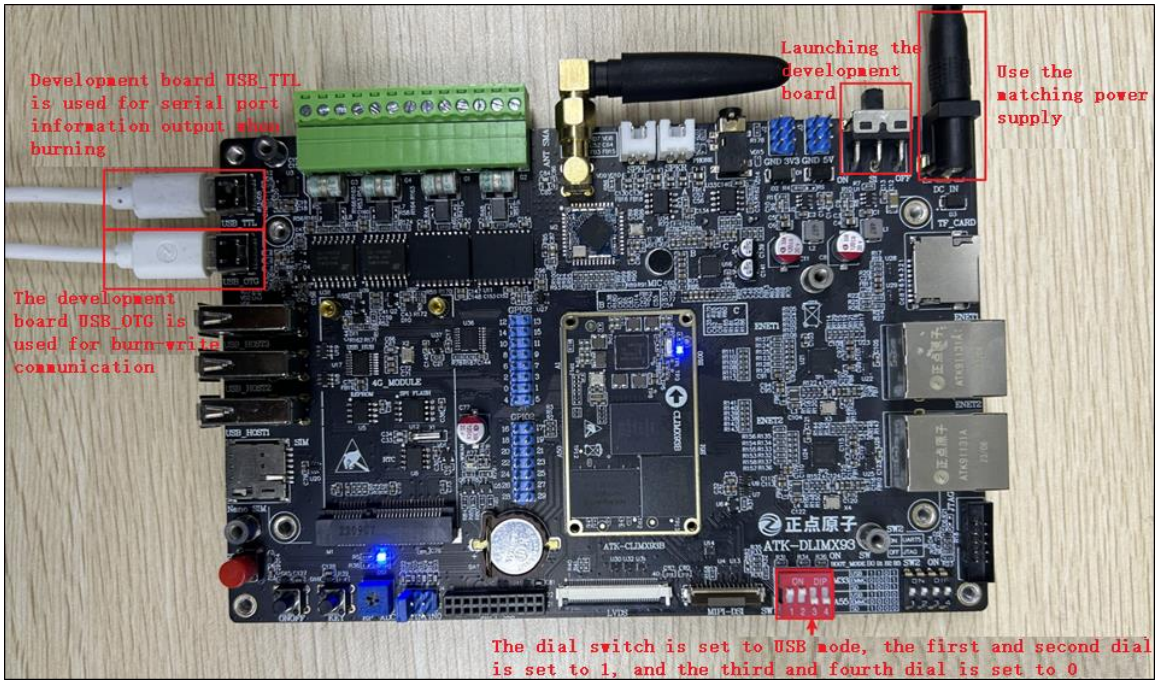


Figure 2.4-1 USB burning when the development board is connected

2.4.2 Environment preparation

In this section, the author carries out UUU burning on a Windows computer, the environment version is as follows, for reference only.

Environment/software/interface	Version
Computer architecture	X86
Host computer system	Windows10
USB interface	USB3.0

2.4.3 Write firmware

Development board firmware package is located in: development board CD A - Basic information \8_system_image

Copy the board firmware package to the Windows host and open the firmware package directory as shown below:

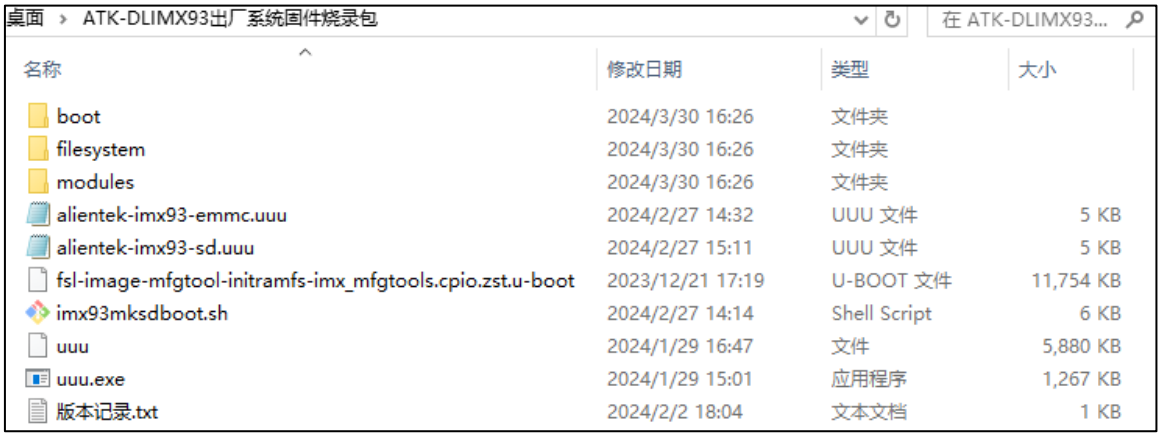


Figure 2.4-2 Firmware package directory

Type "cmd" into the address bar of File Explorer and press Enter to open a terminal window called Command Prompt.

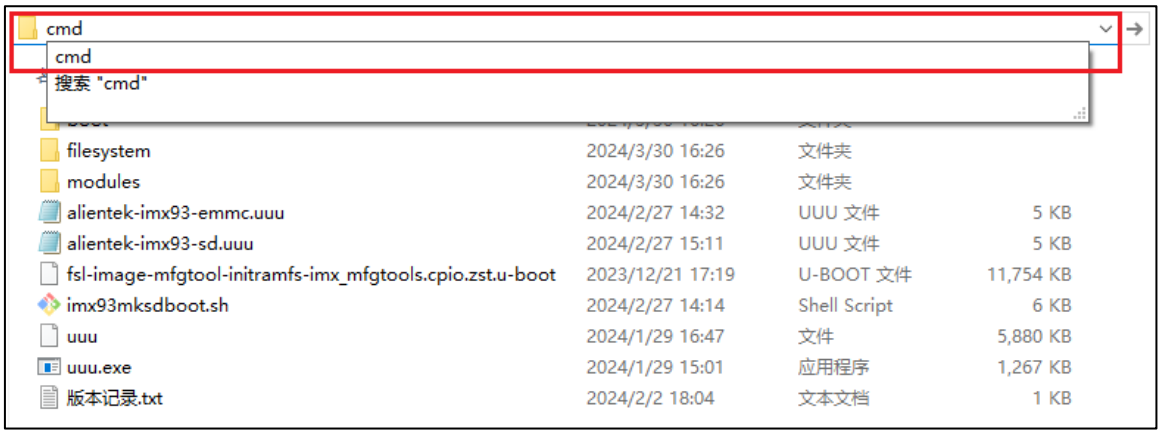


Figure 2.4-3 Enter cmd in the path bar

Through this window, the user can execute command line instructions and operating system functions. Note the differences between the commands in the Windows and Linux terminals. In the Windows terminal, the `tree /f /a` command displays files and folders in a tree structure with full path information in the current directory.

```
tree /f /a
```

/f: This parameter indicates that the file name should be included in the output.

/a: This parameter indicates that the full path information is displayed.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.19045.3930]
(c) Microsoft Corporation。保留所有权利。
                                   Current path
L:\desktop\ATK-DLIMX93出厂系统固件烧录包>tree /f /a      Enter the command
卷 研发资料 的文件夹 PATH 列表
卷序列号为 00000259 7CBE:CCCA
L:.
|   alientek-imx93-emmc.uuu
|   alientek-imx93-sd.uuu
|   fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot
|   imx93mksdboot.sh
|   uuu
|   uuu.exe
|   版本记录.txt
|
+---boot
|   |   flash.bin
|   |   Image
|   |   imx93-11x11-atk-lvds-10.1-1280x800.dtb
|   |   imx93-11x11-atk-mipi-dsi-10.1-800x1280.dtb
|   |   imx93-11x11-atk-mipi-dsi-5.5-1080x1920.dtb
|   |   imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
|   |   imx93-11x11-atk.dtb
|   |
|   +---extlinux-emmc
|   |       extlinux.conf
|   |
|   \---extlinux-sd
|           extlinux.conf
|
+---filesystem
|       rootfs.tar.zst
|
\---modules
        modules.tar.bz2

L:\desktop\ATK-DLIMX93出厂系统固件烧录包>
```

Figure 2.4-4 Windows Terminal

We will use the uuu.exe tool in the Windows system to burn the firmware of the development board, and execute the following instructions in the Windows terminal to burn the firmware.

```
uuu.exe alientek-imx93-emmc.uuu
```

```
L:\desktop\ATK-DLIMX93出厂系统固件烧录包>uuu.exe alientek-imx93-emmc.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 0    Failure 0


1:32    24/31 [===>            12%                  ] FBK: ucp filesystem/rootfs.tar.zst t:-
```

Figure 2.4-5 Burn system firmware

After burning fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot, the development board will restart the system once. During this process, pay attention to keep the development board USB connected to Windows. Otherwise, the writing will report an error that the USB device cannot be opened and will be interrupted, so the writing system needs to be re-burned. Before re-burning, the development board needs to be restarted or reset. You can shut down the virtual machine before burning.

Use the uuu.exe program to run the alientek-imx93-emmc.uuu burn script to burn the firmware to the EMMC storage of the development board. The burning speed of uuu is fast. After the final burning is completed, the command terminal is shown as follows:

```
L:\desktop\ATK-DLIMX93出厂系统固件烧录包>uuu.exe alientek-imx93-emmc.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 1    Failure 0


1:32    31/31 [Done                                  ] FBK: DONE
```

Figure 2.4-6 Complete burn

After burning, set the dial switch to EMMC mode (code bit 0000) to start the development board and enter the system.

2.5 Use UUU to write system to TF card under Window

2.5.1 Development board connection

Prepare a high-speed TF card with a capacity of more than 8GB and format it to FAT32 format, and connect the TF card to the TF card slot of the development board. Set the dial switch of the development board to USB burn mode (corresponding to code position 1100).

The USB_TTL interface of the development board is connected to the USB port of the computer through the Type-C cable, which is used to output print information when burning. If the virtual machine is enabled, switch the USB serial port to the Windows host. This item is only useful for checking information when writing errors.

The USB_OTG interface of the development board is connected to the USB3.0 port of the Windows computer through the Type-C cable for burning the system. If the virtual machine is enabled, fix the USB device switch to the Windows host.

During the burning process, the development board will be restarted, and the computer may automatically connect the USB to the virtual machine or ask the USB connected device. During this period, the UUU may prompt the burning failure. Therefore, it is recommended to set the USB option of the virtual machine to the Windows host by default before burning, or to shut down the virtual machine before burning.

The connection status of the development board is shown as follows:

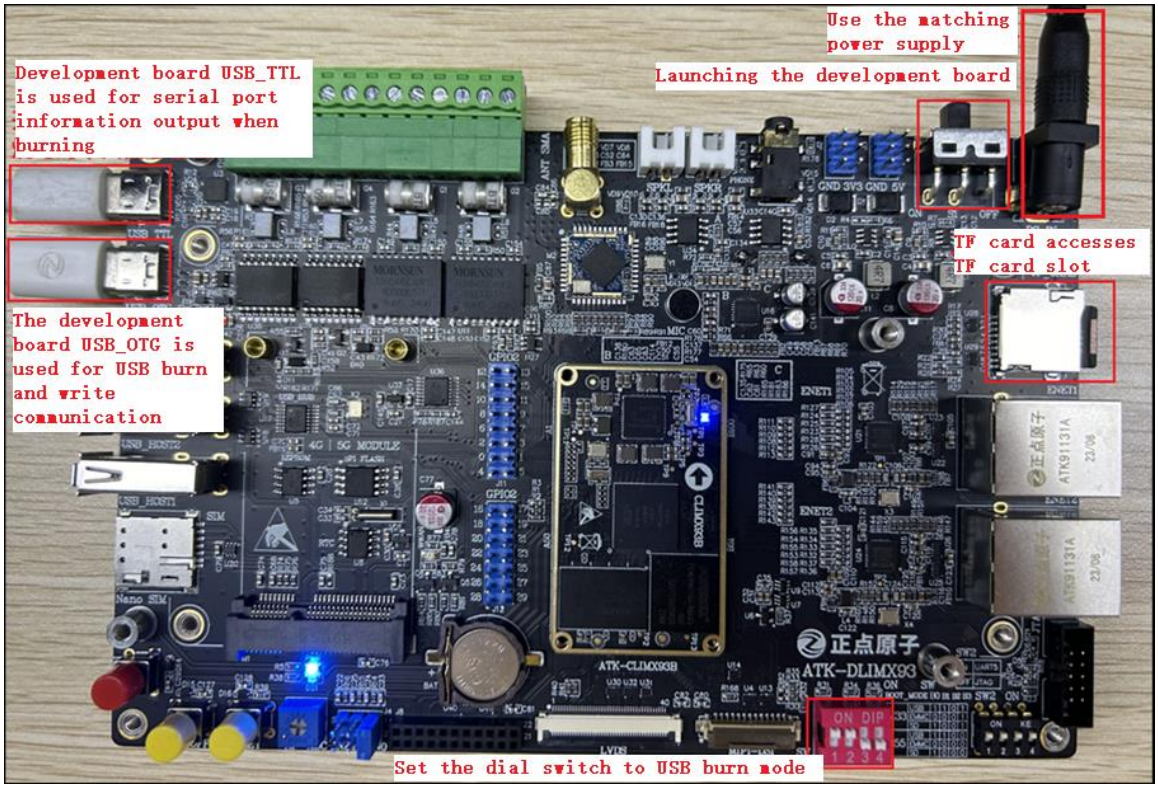


Figure 2.5-1 USB burning when the development board is connected

2.5.2 Environment preparation

In this section, the author carries out UUU burning on a Windows computer, the environment version is as follows, for reference only.

Environment/software/interface	Version
Computer architecture	X86
Computer host system	Windows10
USB interface	USB3.0

2.5.3 Write firmware

Development board firmware package is located in: development board CD A - Basic information \8_system_image

Copy the board firmware package to the Windows host and open the firmware package directory as shown below:

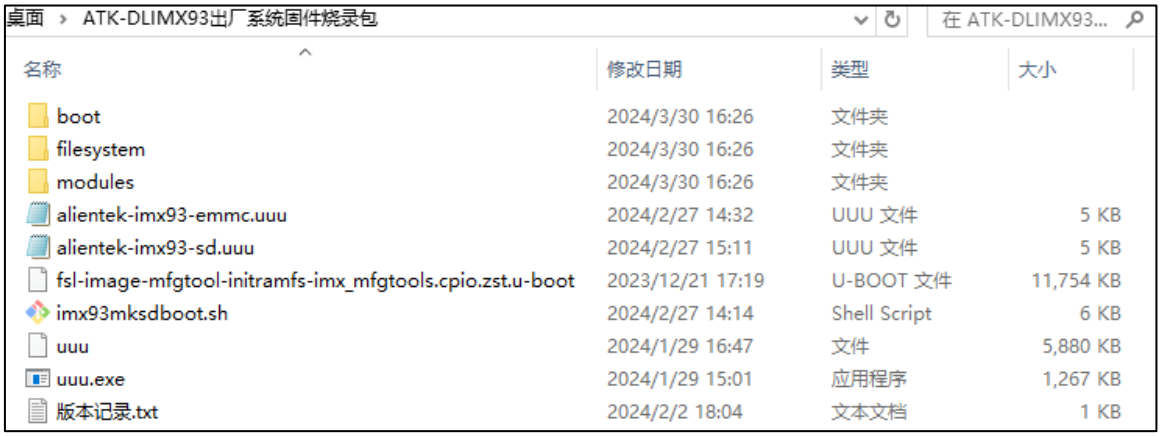


Figure 2.5-2 Firmware package directory

Type "cmd" into the address bar of File Explorer and press Enter to open a terminal window called Command Prompt.

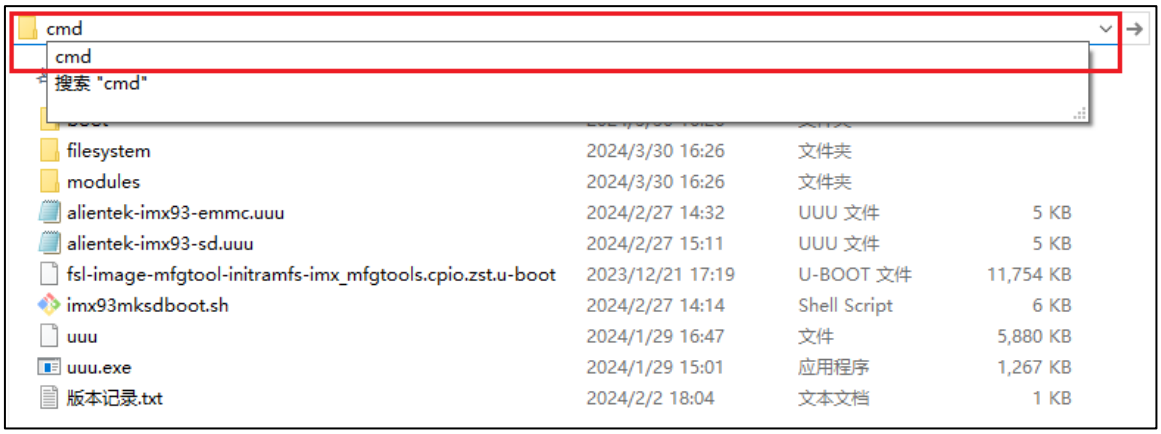


Figure 2.5-3 Enter cmd in the path bar

Through this window, the user can execute command line instructions and operating system functions. Note the differences between the commands in the Windows and Linux terminals. In the

Windows terminal, the tree /f /a command displays files and folders in a tree structure with full path information in the current directory.

```
tree /f /a
```

/f: This parameter indicates that the file name should be included in the output.

/a: This parameter indicates that the full path information is displayed.

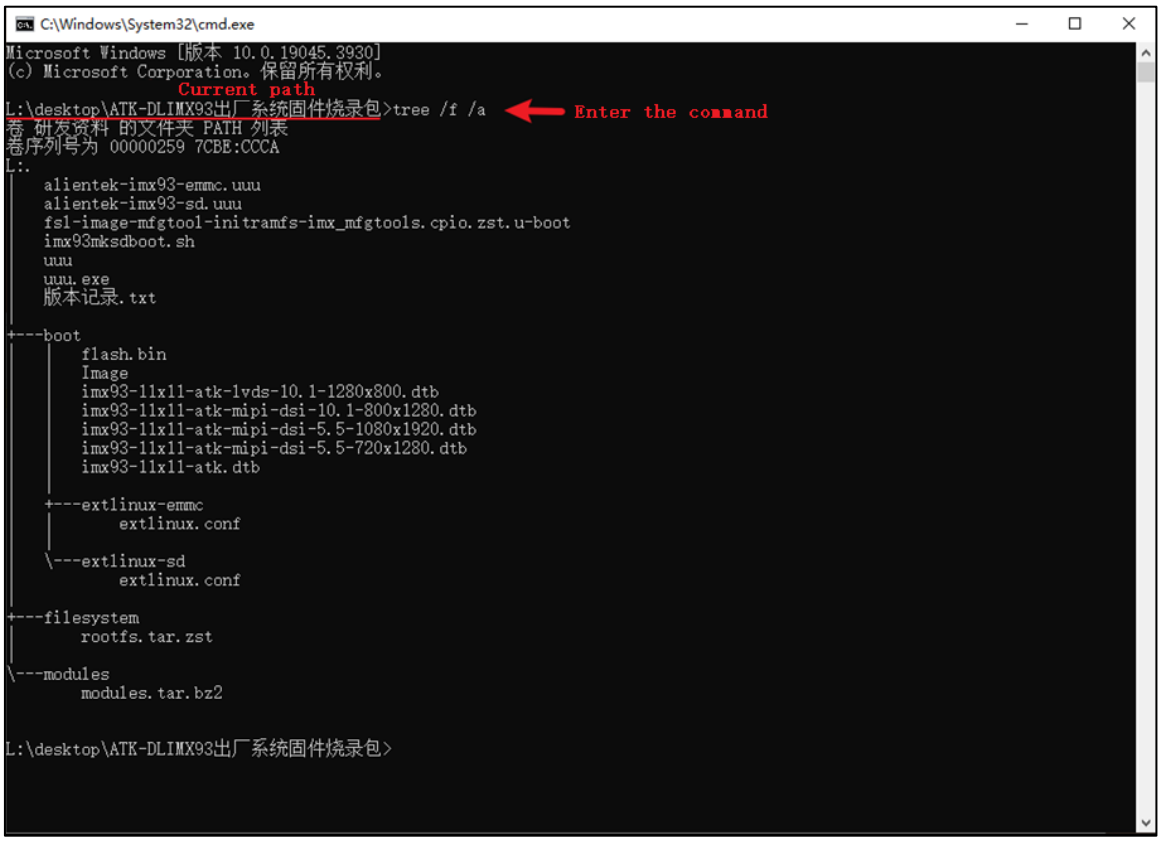


Figure 2.5-4 Windows Terminal

We will use the uuu.exe tool in the Windows system to burn the firmware of the development board, and execute the following instructions in the Windows terminal to burn the firmware.

```
uuu.exe alientek-imx93-sd.uuu
```

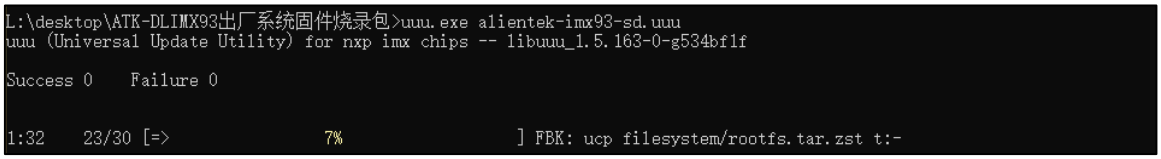


Figure 2.5-5 Burn system firmware

After burning fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot, the development board will restart the system once. During this process, pay attention to keep the development board USB connected to Windows. Otherwise, the writing will report an error that the USB device cannot be opened and will be interrupted, so the writing system needs to be re-burned. Before re-burning, the development board needs to be restarted or reset. You can shut down the virtual machine before burning.

Use the uuu program to run the alientek-imx93-sd.uuu burn script to burn the firmware to the TF card of the development board. The writing speed of uuu is fast, but the writing speed is linked to the

quality of TF card (compared with EMMC, TF card writing speed is slower). After the final writing is completed, the command terminal is shown as follows:

```
L:\desktop\ATK-DLIMX93出厂系统固件烧录包>uuu.exe alientek-imx93-sd.uuu
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.5.163-0-g534bf1f

Success 1    Failure 0


1:32    30/30 [Done                                ] FBK: DONE
```

Figure 2.5-6 Complete burn

After burning, set the dial switch to SD start mode (code point 1000) to start the development board and enter the system.

2.6 Using script to make TF system boot card under Ubuntu20.04

2.6.1 Environment preparation

Development board firmware package is located in: development board CD A - Basic information \8_system_image

Copy the board firmware package to Ubuntu and check the firmware package directory as shown below:

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ tree -L 3
.
├── 版本记录.txt
├── alientek-imx93-emmc.uuu
├── alientek-imx93-sd.uuu
├── boot
│   ├── extlinux-emmc
│   │   └── extlinux.conf
│   ├── extlinux-sd
│   │   └── extlinux.conf
│   ├── flash.bin
│   ├── Image
│   ├── imx93-11x11-atk.dtb
│   ├── imx93-11x11-atk-lvds-10.1-1280x800.dtb
│   ├── imx93-11x11-atk-mipi-dsi-10.1-800x1280.dtb
│   ├── imx93-11x11-atk-mipi-dsi-5.5-1080x1920.dtb
│   └── imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
├── filesystem
│   └── rootfs.tar.zst
├── fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot
├── imx93mksdboot.sh
├── modules
│   └── modules.tar.bz2
├── uuu
└── uuu.exe
```

Figure 2.6-1 Firmware package directory

Use the chmod command to modify the permissions of the system script imx93mksdboot.sh.

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ chmod +x imx93mksdboot.sh
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ ls
版本记录.txt            boot                                                    imx93mksdboot.sh  uuu.exe
alientek-imx93-emmc.uuu filesystem                                              modules
alientek-imx93-sd.uuu   fsl-image-mfgtool-initramfs-imx_mfgtools.cpio.zst.u-boot uuu
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$
```

Figure 2.6-2 Give the script executable permission

TF card is inserted into the Ubuntu virtual machine using the card reader. If Ubuntu does not prompt to connect the removable device to the virtual machine, follow the following steps to connect to the virtual machine. In addition, the virtual machine is set to USB3.0 in advance and the card reader is connected to the USB3.0 interface.

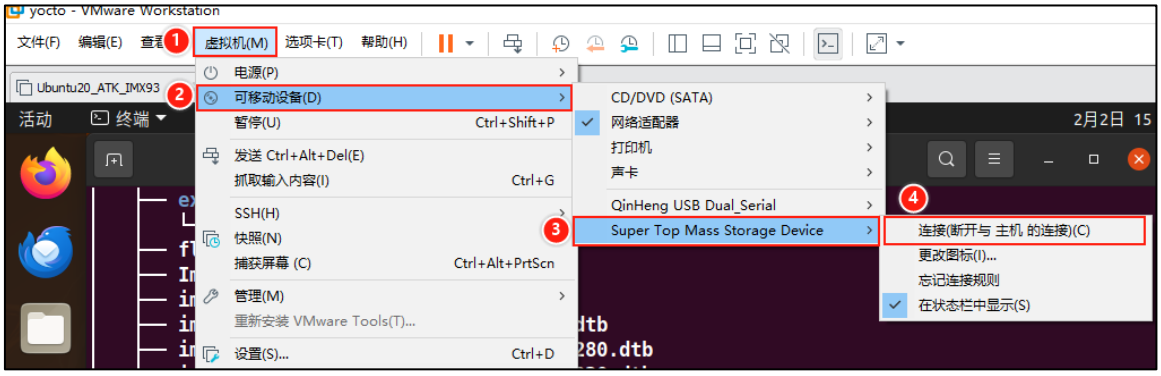


Figure 2.6-3 Steps to connect TF cards

Enter the following command to view the TF card mount node, as shown in the figure below, the capacity of my TF card is 28.9GB(32GB), and you can see that the node mounted is /dev/sdb.

```
sudo fdisk -l
```

```
Disk /dev/sdb: 28.91 GiB, 31016878080 字节，60579840 个扇区
Disk model: Storage Device
单元：扇区 / 1 * 512 = 512 字节
扇区大小(逻辑/物理)：512 字节 / 512 字节
I/O 大小(最小/最佳)：512 字节 / 512 字节
磁盘标签类型：dos
磁盘标识符：0x9c13f1de

设备     启动  起点     末尾     扇区  大小 Id 类型
/dev/sdb1      8192 60579839 60571648 28.9G  c W95 FAT32 (LBA)
```

Figure 2.6-4 View the connection node of the TF card on Ubuntu

2.6.2 Write firmware

Run./imx93mksdboot.sh -help to see how the script is used.

```
./imx93mksdboot.sh –help
```

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ ./imx93mksdboot.sh –help


用法：imx93mksdboot.sh [选项] <(必选)-device>
用法示例：
sudo ./imx93mksdboot.sh -device /dev/sdx
命令选项：
  -device              SD卡块设备节点 (例如/dev/sdx)
可选选项：
  --version             打印版本信息.
  --help                打印帮助信息.

alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$
```

Figure 2.6-5 See the instructions for using imx93mksdboot.sh

Usage Note:

Usage: imx93mksdboot.sh [Options] <(required)-device>

-device: This specifies the device node (TF card mounted node such as /dev/sdx)

For example, the user TF card is now mounted on /dev/sdb. After executing the instruction, the script will ask whether the node mounted on the TF card corresponds to it again in Chinese, and all data

on the TF card will be cleared. Please pay attention to backup important data. Press Enter to confirm and proceed. It will take about a few minutes to solidify the TF card, which may take a lot of time depending on the PC and TF card used.

Execute the burn command:

```
sudo ./imx93mksdboot.sh -device /dev/sdb
```

```
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$ sudo ./imx93mksdboot.sh -device /dev/sdb
即将进行制作SD系统启动卡，大约花费几分钟时间,请耐心等待！
*************************************************************
*         注意：这将会清除/dev/sdb所有的数据                *
*         在脚本执行时请不要将/dev/sdb拔出                  *
*             请按<Enter>确认继续                           *
*************************************************************

卸载 device '/dev/sdb1'
记录了20480+0 的读入
记录了20480+0 的写出
20971520字节 (21 MB, 20 MiB) 已复制, 4.23911 s, 4.9 MB/s

欢迎使用 fdisk (util-linux 2.34)。
更改将停留在内存中，直到您决定将更改写入磁盘。
使用写入命令前请三思。

设备不包含可识别的分区表。
创建了一个磁盘标识符为 0xfe4dee63 的新 DOS 磁盘标签。

命令(输入 m 获取帮助): 分区类型
   p   主分区 (0个主分区，0个扩展分区，4空闲)
   e   扩展分区 (逻辑分区容器)
选择 (默认 p): 分区号 (1-4, 默认 1): 第一个扇区 (2048-61182975, 默认 2048): Last sector, +/-sectors or +/-size{K,M,G,T,P}
(20480-61182975, 默认 61182975):
创建了一个新分区 1，类型为“Linux”，大小为 500 MiB。                                  后续烧写信息省略.....
```

Figure 2.6-6 Execute burn system to TF card

In the process of curing, the TF card will be unmounted, and when the script is executed, the mouse should not leave the Ubuntu virtual machine. Otherwise, the TF card may be connected to the PC host when the script is unmounted, so that the script cannot find the TF card to execute, and it will prompt "mount: special device /dev/sdb1 does not exist ". The process of curing is suggested by combining Chinese and English, and the curing is completed as shown below.

```
正在分配组表: 完成
正在写入 inode表: 完成
创建日志 (32768 个块) : 完成
写入超级块和文件系统账户统计信息: 已完成

正在烧写flash.bin到/dev/sdb
记录了1926+0 的读入
记录了1926+0 的写出
1972224字节 (2.0 MB, 1.9 MiB) 已复制, 0.840987 s, 2.3 MB/s
烧写flash.bin到/dev/sdb完成!
正在准备复制...
正在复制设备树与内核到/dev/sdb1, 请稍候...
复制设备树与内核到/dev/sdb1完成!
卸载/dev/sdb1
正在解压文件系统到/dev/sdb2 , 请稍候...
解压文件系统到/dev/sdb2完成!
正在解压模块到/dev/sdb2/lib/modules/ , 请稍候...
解压模块到/dev/sdb2/lib/modules/完成!
卸载/dev/sdb2
SD卡启动系统烧写完成!
alientek@ubuntu:~/atk-imx93-6.1.55-sdk/ATK-DLIMX93出厂系统固件烧录包$
```

Figure 2.6-7 Complete burn

Follow the method of connecting TF card to Ubuntu, and then click "Disconnect" to exit TF card. After the writing is finished, connect the TF card to the TF card slot of the development board (note that the card reader is not connected to the USB of the development board), dial the code switch to TF

startup mode 1000, and start the system. You can connect the serial port in advance to see the specific startup information.

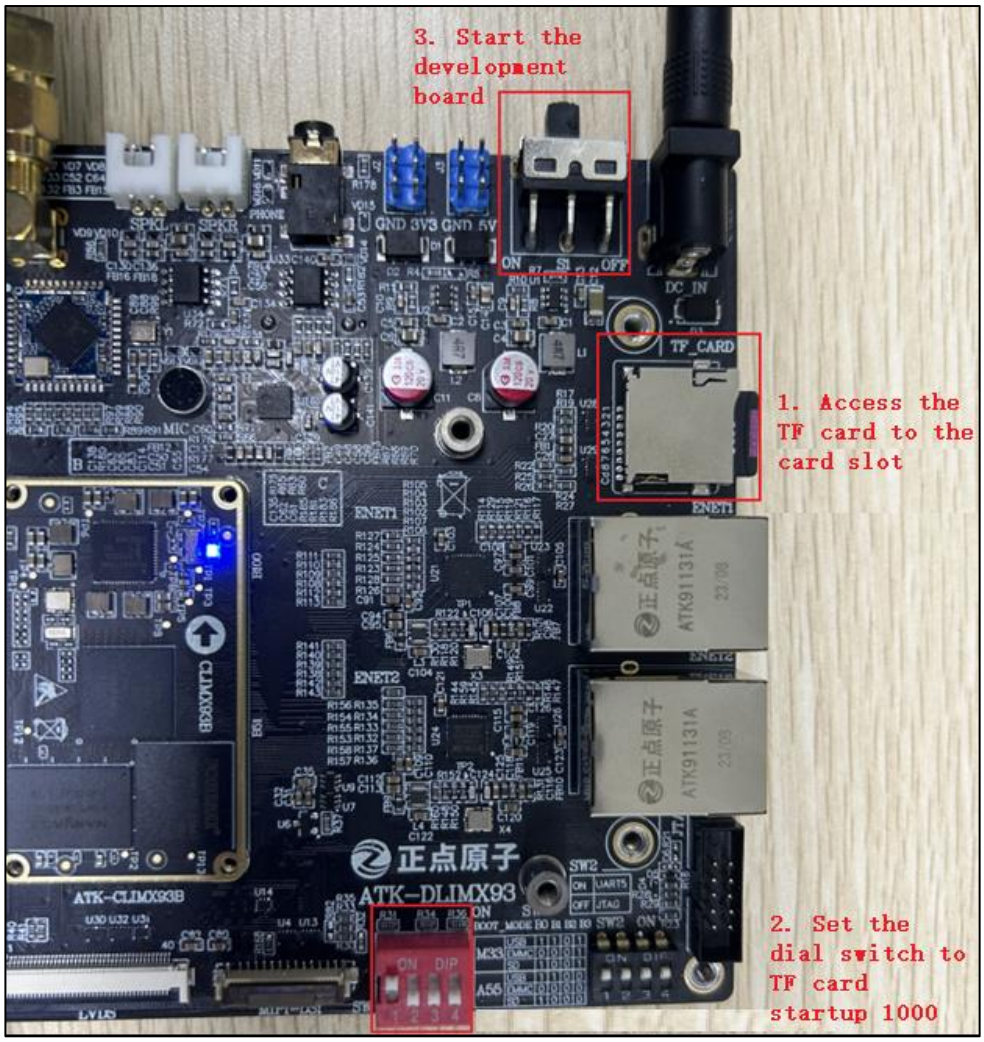


Figure 2.6-8 The TF card starts the development board

The specific boot information can be seen in the serial port terminal. For ATK-DLIMX93 development board, /dev/mmcblk1 is for TF card, /dev/mmcblk1p1 is for boot partition, and /dev/mmcblk1p2 is for rootfs partition.

```
eth0: ethernet@42890000, eth1: ethernet@428a0000 [PRIME]
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc1 is current device
Scanning mmc 1:1...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
ADC1_CH3=1799mV
Select the boot mode
1:      imx93-11x11-atk
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
3:      imx93-11x11-atk-mipi-dsi-5.5-1080x1920
4:      imx93-11x11-atk-mipi-dsi-10.1-800x1280
5:      imx93-11x11-atk-lvds-10.1-1280x800
Enter choice(For ATK-LVDS-LCD-10.1, enter number 5. Otherwise, auto select default, do not enter):
1:      imx93-11x11-atk
Retrieving file: /Image
append: root=/dev/mmcblk1p2 rootwait rw console=ttyLP0,115200
Retrieving file: /imx93-11x11-atk.dtb
## Flattened Device Tree blob at 83000000
   Booting using the fdt blob at 0x83000000
Working FDT set to 83000000
   Using Device Tree in place at 0000000083000000, end 000000008300dc31
Working FDT set to 83000000
fail to find output device
probe video device failed, ret -19

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
```

Figure 2.6-9 TF card startup information

Chapter 3. Preparation before use of ATK-DLIMX93

Here we will introduce the considerations for using the development board. If you don't pay attention to these things, your experiment may not be successful, or lead to damage to the development board, etc., so be sure to pay attention.

3.1 Precautions before power on

- First check whether the DC power supply is plugged in, the development board has removed the USB_TTL serial port power supply function, and the USB_TTL serial port power supply cannot be used.
- The entire board is protected by the acrylic plate before leaving the factory, but in some cases, the user may unload the acrylic plate. At this time, it is necessary to pay attention to observe that the bottom plate is not put on the debris table, and the protective film needs to be placed under the bottom plate, otherwise it is easy to contact with metal foreign bodies and short circuit some two contacts under the bottom plate. In the case of long use, we should also pay attention to observe whether there is a foreign body falling on the development board to prevent short circuit, pay attention to waterproof, moisture-proof, dustproof and so on.

3.2 Serial port software installation

Section Preface:

Here is a brief introduction to CH342 USB serial port driver installation and MobaXterm terminal installation.

- Install CH342 USB serial port driver (we need to install this driver for the computer to communicate with the development board by serial port)

In the development board CD A disk - basic data \3_softwares folder to find CH343SER.EXE (this driver is provided by Qin Heng, compatible with CH342), double click to run, and then pop up the window, directly click Install, wait for the installation success window to appear.

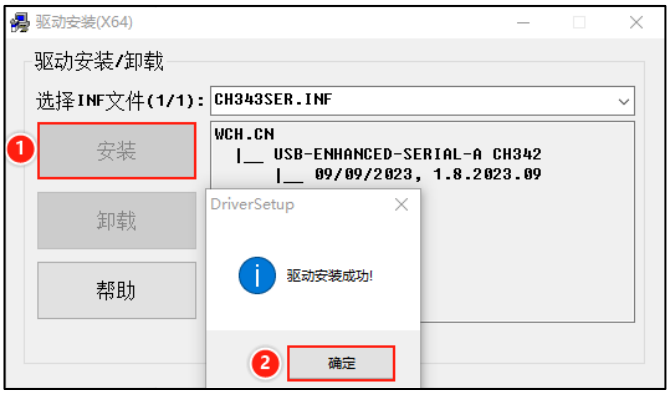


Figure 3.2-1 Install the serial driver

- Install MobaXterm terminal software (or install Xshell, SecureCRT and other terminal software)

Take, for example, MobaXterm. Network disk path development board CD A- Basic Information ->3_softwares ->MobaXterm_Installer_v12.3.zip. Double-click to open the package, wait for it to be

unpacked, and then double-click MobaXterm_installer_12.3.msi to install it. The MobaXterm installer will guide you through the installation.

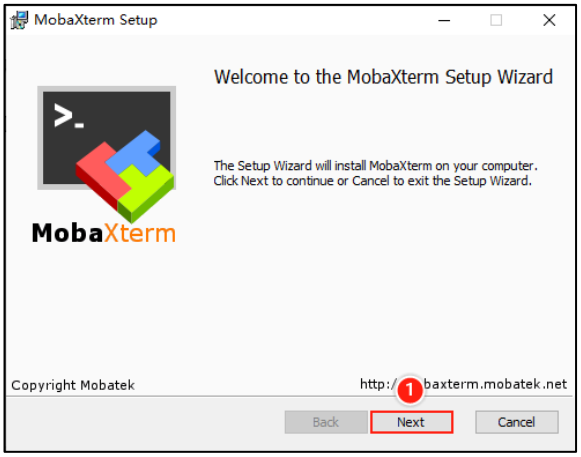


Figure 3.2-2 Starting the installation

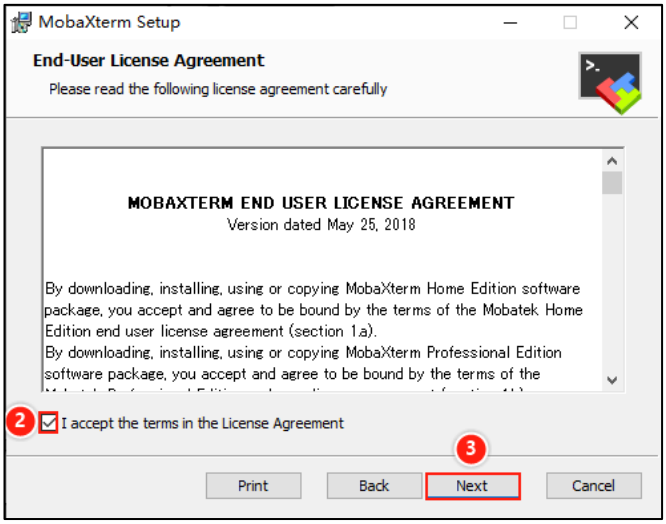


Figure 3.2-3 Check the Read protocol

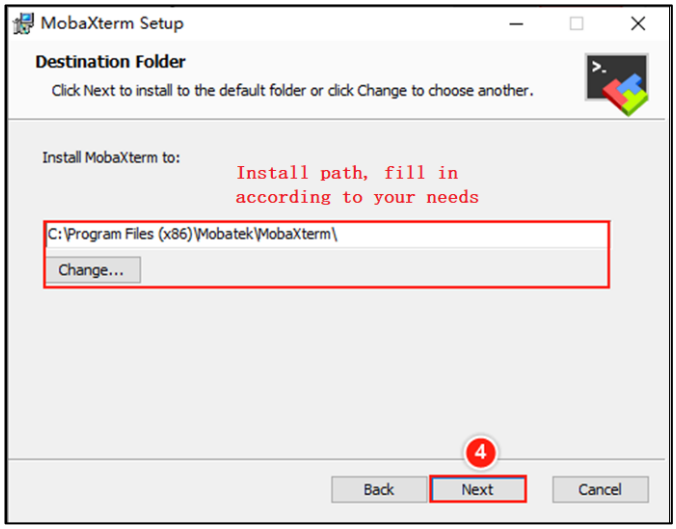


Figure 3.2-4 Select an installation path

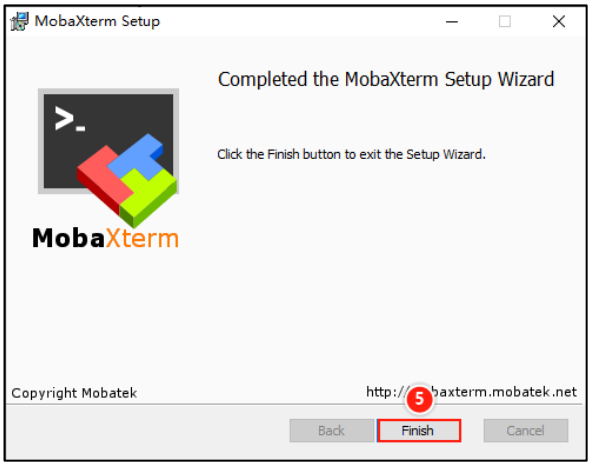
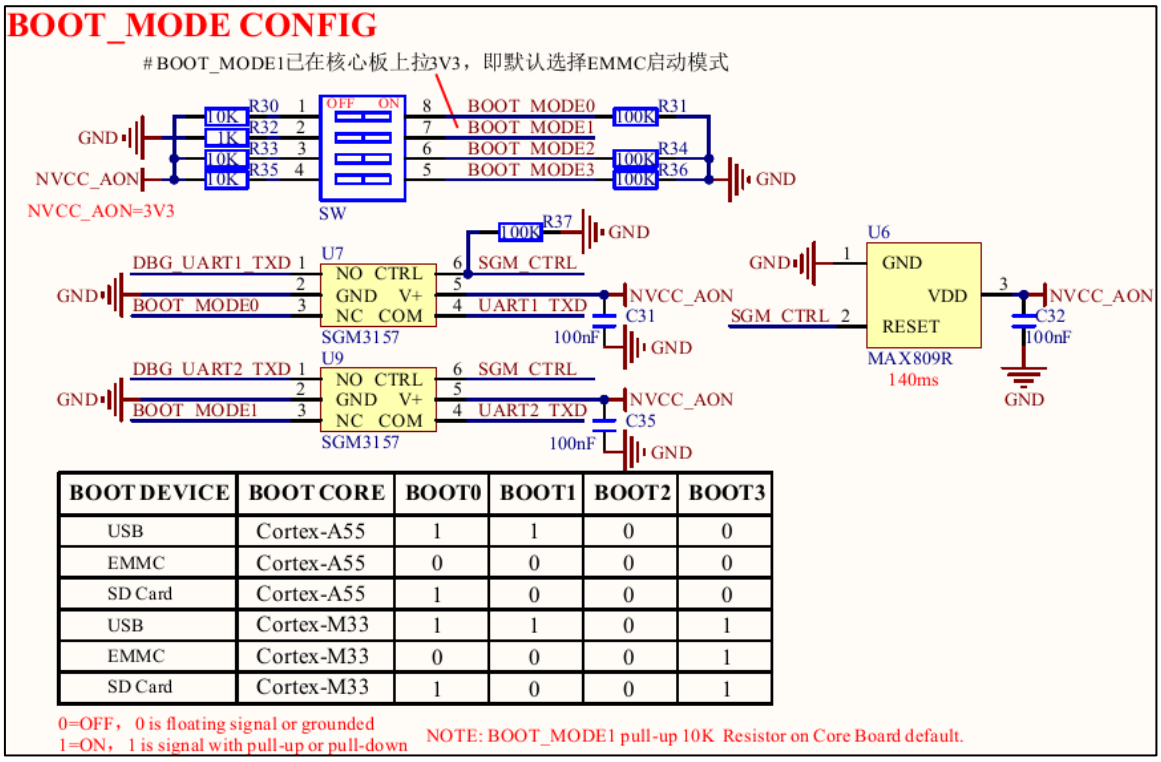


Figure 3.2-5 Installation Completed

3.3 Setting of dial switch and landing development board

The dip switch of ATK-DLIMX93 development board supports the following BOOT modes. The following figure is the boot schematic diagram of the development board bottom board.



BOOT DEVICE	BOOT CORE	BOOT0	BOOT1	BOOT2	BOOT3
USB	Cortex-A55	1	1	0	0
EMMC	Cortex-A55	0	0	0	0
SD Card	Cortex-A55	1	0	0	0
USB	Cortex-M33	1	1	0	1
EMMC	Cortex-M33	0	0	0	1
SD Card	Cortex-M33	1	0	0	1

0=OFF， 0 is floating signal or grounded
1=ON， 1 is signal with pull-up or pull-down
NOTE: BOOT_MODE1 pull-up 10K Resistor on Core Board default.

Figure 3.3-1 BOOT mode

The schematic diagram of the development board shows that:

(1) Dial 1100 for USB download mode of A core;

(2) The dial code for EMMC startup mode of A core is 0000;

(3) The dial code of the A-core SD Card start-up mode is 1000.

(4) Dial 1100 for M-core USB download mode;

(5) M core EMMC startup mode dial code is 0001;

(6) The dial code of M-core SD Card startup mode is 1001.

ATK-DLIMX93 development board uses a USB Type-C cable, connected to the USB_TTL interface on the bottom board, the other end is connected to the USB interface of the PC(computer), the factory has burned the Linux system to the eMMC storage on the core board. So we set the dial to 0000, which is represented as "1" (ON) up and "0" (OFF) down.

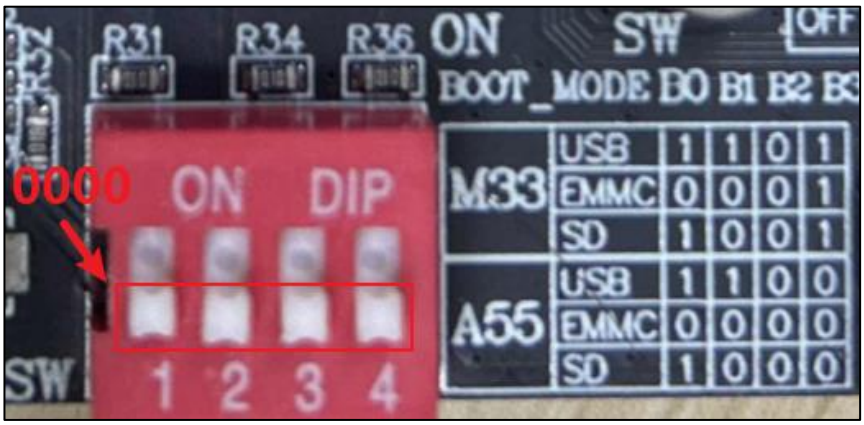


Figure 3.3-2 eMMC starts the dial mode

The development board is plugged into the DC12V DC power supply, and the MobaXterm is selected as the serial port type, as set as follows. Please select the corresponding COM port (COM port is PC (computer) identification development board USB_TTL serial port number. If the virtual machine is open at this time, please pay attention to whether the serial port is waiting to select connection or has been connected to the virtual machine, need to switch the serial port to connect to the Windows side). The development board uses CH342 chip to divide one serial port into two channels:

The A-core serial port uses USB-Enhanced-SERIAL-A-CH342,

The M-core serial port uses USB-Enhanced-SERIAL-B-CH342.

Here we describe the factory system booted based on A core, so the serial port is USB-Enhanced-SERIAL-A-CH342.

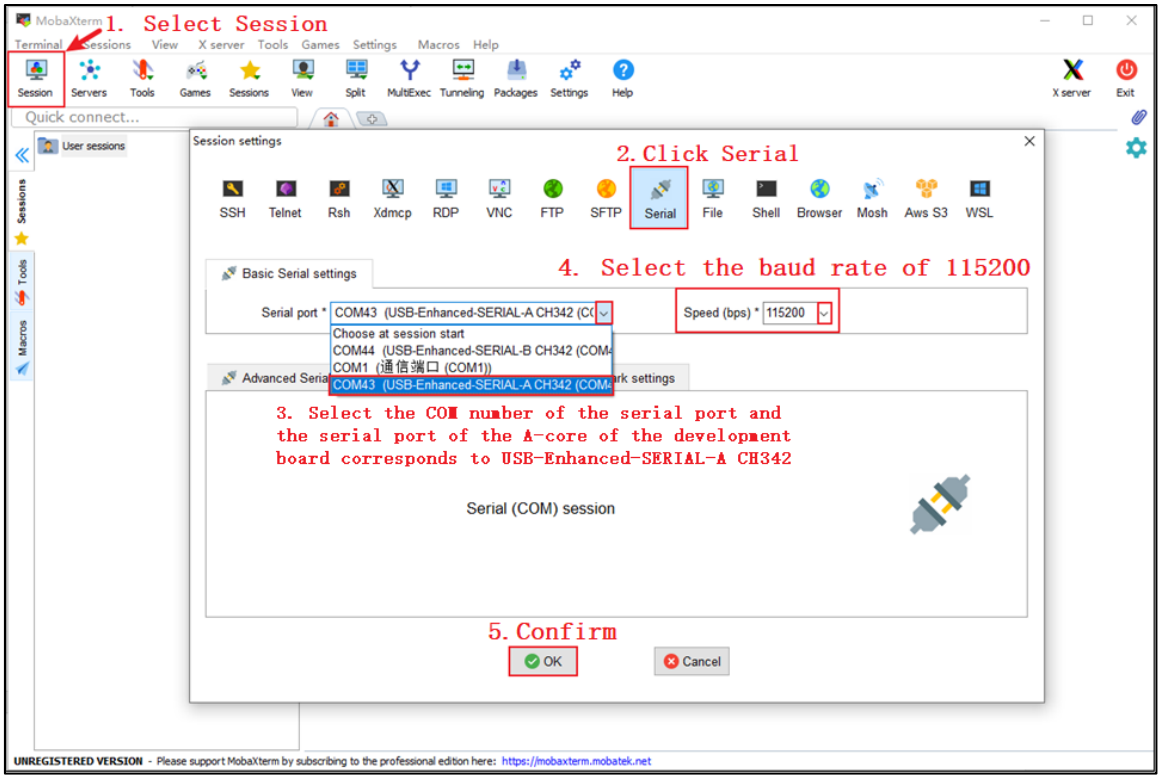


Figure 3.3-3 MobaXterm serial port Settings

After the development board is started from eMMC mode, the serial port terminal prints the operation information of U-Boot and kernel. We can start the development board (if the development board has been started, we can restart the development board), and observe that the serial port prints as follows.

```
U-Boot SPL 2023.04-g840fa172-dirty (Jan 12 2024 - 11:38:11 +0800)
SOC: 0xa1009300
LC: 0x2040010
PMIC: Over Drive Voltage Mode
DDR: 3733MTS
M33 prepare ok
Normal Boot
Trying to boot from BOOTROM
Boot Stage: Primary boot
image offset 0x0, pagesize 0x200, ivt offset 0x0
Load image from 0x47000 by ROM_API
NOTICE:  BL31: v2.8(release):lf-6.1.55-2.2.0-0-g08e9d4e
NOTICE:  BL31: Built : 06:43:30, Nov 21 2023


U-Boot 2023.04-g840fa172-dirty (Jan 12 2024 - 11:38:11 +0800)

CPU:   i.MX93(52) rev1.1 1700 MHz (running at 1692 MHz)
CPU:   Industrial temperature grade (-40C to 105C) at 39C
Reset cause: POR (0x1)
Model: NXP i.MX93 11X11 EVK board
DRAM:  992 MiB
Core:  204 devices, 30 uclasses, devicetree: separate
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
[*]-Video Link 0
fail to find output device
probe video device failed, ret -19

       [0] lcd-controller@4ae30000, video
fail to find output device
probe video device failed, ret -19
In:    serial
Out:   serial
Err:   serial

BuildInfo:
  - ELE firmware version 0.0.11-33cda99a

switch to partitions #0, OK
mmc0(part 0) is current device
panel=ATK-MIPI-LCD-5.5-720x1280
flash target is MMC:0
Net:   eth0: ethernet@42890000, eth1: ethernet@428a0000 [PRIME]
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0(part 0) is current device
Scanning mmc 0:1...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
ADC1_CH3=609mV
Select the boot mode
1:      imx93-11x11-atk
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
3:      imx93-11x11-atk-mipi-dsi-5.5-1080x1920
4:      imx93-11x11-atk-mipi-dsi-10.1-800x1280
5:      imx93-11x11-atk-lvds-10.1-1280x800
Enter choice(For ATK-LVDS-LCD-10.1, enter number 5. Otherwise, auto select default, do not enter):
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
Retrieving file: /image
append: root=/dev/mmcblk0p2 rootwait rw console=ttyLP0,115200
Retrieving file: /imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
## Flattened Device Tree blob at 83000000
```

Because the LCD function of uboot is not enabled, the uboot print cannot find the screen

uboot reads the positive atomic MIPI screen ID and automatically configures the environment variables

uboot will start the corresponding device tree based on the positive atomic MIPI screen ID without manual modification

The positive dot atom LVDS screen has no ID pin. If you use this screen, you need to manually select the number 5 to start the device tree

Figure 3.3-4 U-Boot operation information printed by the serial port

The factory file system is built by yocto. After starting the development board, the Qt interface will also be started.

By default, you do not need to enter the login account password to directly enter the root directory of the system. The serial terminal after login is shown below.

```
NXP i.MX Release Distro 6.1-mickledore atk-imx93 ttyLP0

atk-imx93 login: root (automatic login)

[    7.875542] CAN device driver interface
[    7.887019] imx8_media_dev: module is from the staging directory, the quality is unknown, you have been wa
rned.
[    7.898513] mxc-md 42800000.bus:camera: deferring cap_device registration
[    7.950608] mxc-md 42800000.bus:camera: deferring cap_device registration
[    7.972340] mxc-md 42800000.bus:camera: deferring cap_device registration
[    8.503498] imx-dwmac 428a0000.ethernet: IRQ eth_lpi not found
[    8.512477] imx-dwmac 428a0000.ethernet: User ID: 0x10, Synopsys ID: 0x52
[    8.521525] imx-dwmac 428a0000.ethernet:     DWMAC4/5
[    8.526411] imx-dwmac 428a0000.ethernet: DMA HW capability register supported
[    8.538939] imx-dwmac 428a0000.ethernet: RX Checksum Offload Engine supported
[    8.548923] fsl_mc_err_probe: No ECC DIMMs discovered
[    8.554021] imx-dwmac 428a0000.ethernet: TX Checksum insertion supported
[    8.560732] imx-dwmac 428a0000.ethernet: Wake-Up On Lan supported
[    8.569398] imx-dwmac 428a0000.ethernet: Enable RX Mitigation via HW Watchdog Timer
[    8.577080] imx-dwmac 428a0000.ethernet: Enabled L3L4 Flow TC (entries=8)
[    8.583881] imx-dwmac 428a0000.ethernet: Enabled RFS Flow TC (entries=10)
[    8.591051] imx-dwmac 428a0000.ethernet: Enabling HW TC (entries=256, max_off=256)
[    8.598701] imx-dwmac 428a0000.ethernet: Using 32/32 bits DMA host/device width
[    8.628048] isi-capture 4ae40000.isi:cap_device: deferring 4ae40000.isi:cap_device device registration
[    8.645016] imx-dwmac 428a0000.ethernet eth1: Register MEM_TYPE_PAGE_POOL RxQ-0
[    8.645853] mxc-isi 4ae40000.isi: mxc_isi.0 registered successfully
[    8.661397] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL
)
[    8.688971] mxc-md 42800000.bus:camera: deferring cap_device registration
[    8.721647] mx8-img-md: Registered mxc_isi.0.capture as /dev/video0
[    8.731005] unregister ISI channel: mxc_isi.0
[    8.778431] imx-dwmac 428a0000.ethernet eth1: No Safety Features support found
[    8.876234] imx-dwmac 428a0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp supported
[    8.976615] imx-dwmac 428a0000.ethernet eth1: registered PTP clock
[    9.072233] imx-dwmac 428a0000.ethernet eth1: FPE workqueue start
[    9.108958] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    9.302620] 8021q: adding VLAN 0 to HW filter on device eth1
[    9.534318] imx-dwmac 428a0000.ethernet eth1: FPE workqueue stop
[    9.582380] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL
)
[    9.641941] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    9.808359] imx-dwmac 428a0000.ethernet eth1: Register MEM_TYPE_PAGE_POOL RxQ-0
[    9.853854] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL
)
[    9.892301] imx-dwmac 428a0000.ethernet eth1: No Safety Features support found
[    9.907490] imx-dwmac 428a0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp supported
[    9.931643] imx-dwmac 428a0000.ethernet eth1: registered PTP clock
[    9.953546] imx-dwmac 428a0000.ethernet eth1: FPE workqueue start
[    9.960393] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    9.969356] 8021q: adding VLAN 0 to HW filter on device eth1
root@atk-imx93:~#
root@atk-imx93:~# [   19.684410] platform imx93-lpm: deferred probe pending

root@atk-imx93:~#
root@atk-imx93:~#
```

Figure 3.3-5 The serial port terminal prints the information of the login development board

Note: The factory file system supports SSH service by default. When users use SSH to login, they need to enter the account root and password root.

Chapter 4. ATK-DLIMX93 functional test

4.1 LED Testing

The ATK-DLIMX93 development board is equipped with a user LED light, which is configured to be on by default during the uboot phase and blinks after the kernel has been started.

The state of the LED can be controlled by entering the following command in the file system.

```
cat /sys/class/leds/sys-led/trigger                // See the current and supported LED trigger modes
echo none > /sys/class/leds/sys-led/trigger        // Change LED trigger to none
echo 1 > /sys/class/leds/sys-led/brightness        // Light up LED1
echo 0 > /sys/class/leds/sys-led/brightness        // Extinguish LED1
echo heartbeat > /sys/class/leds/sys-led/trigger // Change LED trigger mode to heartbeat
```

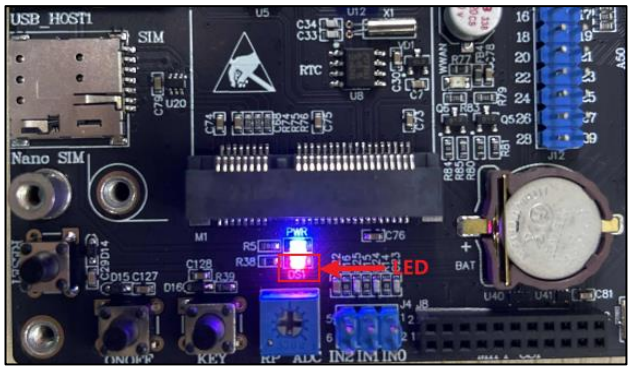


Figure 4.1-1 Development board LED position

4.2 KEY Test

The ATK-DLIMX93 development board is equipped with a user KEY, the key position is shown as follows:

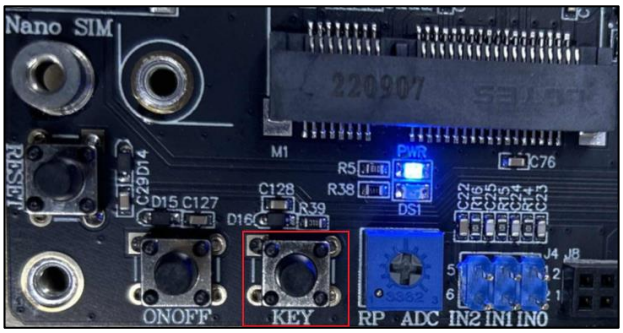


Figure 4.2-1 Key Position

Enter the following command in the factory file system to see the device corresponding to the input event.

```
cat /proc/bus/input/devices
```

```
root@atk-imx93:~# cat /proc/bus/input/devices
I: Bus=0019 Vendor=0000 Product=0000 Version=0000
N: Name="44440000.bbnsm:pwrkey"
P: Phys=bbnsm-pwrkey/input0
S: Sysfs=/devices/platform/soc@0/44000000.bus/44440000.bbnsm/44440000.bbnsm:pwrkey/input/input0
U: Uniq=
H: Handlers=kbd event0
B: PROP=0
B: EV=3
B: KEY=10000000000000 0

I: Bus=0018 Vendor=0416 Product=03a0 Version=6010
N: Name="Goodix Capacitive TouchScreen"
P: Phys=input/ts
S: Sysfs=/devices/platform/soc@0/42000000.bus/42540000.i2c/i2c-3/3-0014/input/input1
U: Uniq=
H: Handlers=kbd event1
B: PROP=2
B: EV=b
B: KEY=400 0 0 0 2000000000000001 f800000000000000
B: ABS=265800000000003

I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="gpio_keys@0"
P: Phys=gpio-keys/input0
S: Sysfs=/devices/platform/gpio_keys@0/input/input3
U: Uniq=
H: Handlers=kbd event2
B: PROP=0
B: EV=100003
B: KEY=4000000000000 0
```

Figure 4.2-2 KEY events

When the author was testing, the development board was connected to the LCD screen. From the above figure, we can know that event1 is the touch screen event (such as GT series touch chip), event2 is the KEY event. If the user is not connected to the touchscreen, event1 will be the KEY event. Therefore, event2 is not necessarily fixed as a key event, which is related to the device accessed on the user's board.

If the screen is connected, you can use the od-x or hexdump commands to view the keypress input. Press CTRL-C to terminate the command.

```
    od -x /dev/input/event2
or hexdump /dev/input/event2
```

If the user is not connected to the LCD screen, the command to test the key is as follows:

```
    od -x /dev/input/event1
or hexdump /dev/input/event1
```

When the KEY is pressed, the following message appears, indicating that the key is normal. If there is no response, you can press the key several times to try.

```
root@atk-imx93:~# od -x /dev/input/event2
0000000 c685 6401 0000 0000 0212 000c 0000 0000
0000020 0001 0072 0001 0000 c685 6401 0000 0000
0000040 0212 000c 0000 0000 0000 0000 0000 0000
0000060 c685 6401 0000 0000 7a7b 000d 0000 0000
0000100 0001 0072 0000 0000 c685 6401 0000 0000
0000120 7a7b 000d 0000 0000 0000 0000 0000 0000
^C
root@atk-imx93:~# hexdump /dev/input/event2
0000000 c68b 6401 0000 0000 2d1a 0000 0000 0000
0000010 0001 0072 0001 0000 c68b 6401 0000 0000
0000020 2d1a 0000 0000 0000 0000 0000 0000 0000
0000030 c68b 6401 0000 0000 d62f 0001 0000 0000
0000040 0001 0072 0000 0000 c68b 6401 0000 0000
0000050 d62f 0001 0000 0000 0000 0000 0000 0000
^C
root@atk-imx93:~#
```

Figure 4.2-3 Print key event

4.3 MIPI LCD test

This section is only applicable to board users who have selected a ALIENTEK MIPI screen.

4.3.1 Instructions for use

In the factory system, because the MIPI screen cable of the ALIENTEK has an ADC pin connected to the CPU, the factory system will recognize the resolution of the different screens loaded. There are three MIPI screens used by the ALIENTEK, as shown in the following table:

Screen size	Touch chip
5.5 inch screen（720x1280）	gt911
5.5 inch screen（1080x1920）	gt911
10.1 inch screen（800x1280）	gt911

MIPI screen connection: use the matching 26pin FPC cable in the same direction of the screen, and the connection is as follows (this picture is only for convenient display of the interface, not the actual display effect). The FPC seat clip of the development board is opened upward, which is loosened, and flattened downward, which is tightened.

Figure 4.3-1 Development board and screen connection

After connecting the screen to the development board, power on and start the development board. By default, the system will select the corresponding device tree according to the screen ID value to start, without manually modifying the device tree. The specific screen information can be seen in the serial port terminal, as shown in the following figure:

```
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
[*]-Video Link 0
fail to find output device
probe video device failed, ret -19

       [0] lcd-controller@4ae30000, video
fail to find output device
probe video device failed, ret -19
In:    serial
Out:   serial
Err:   serial

BuildInfo:
  - ELE firmware version 0.0.11-33cda99a

switch to partitions #0, OK
mmc0(part 0) is current device
panel=ATK-MIPI-LCD-5.5-720x1280
flash target is MMC:0
Net:   eth0: ethernet@42890000, eth1: ethernet@428a0000 [PRIME]
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0(part 0) is current device
Scanning mmc 0:1...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
ADC1 CH3=609mV
Select the boot mode
1:      imx93-11x11-atk
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
3:      imx93-11x11-atk-mipi-dsi-5.5-1080x1920
4:      imx93-11x11-atk-mipi-dsi-10.1-800x1280
5:      imx93-11x11-atk-lvds-10.1-1280x800
Enter choice(For ATK-LVDS-LCD-10.1, enter number 5. Otherwise, auto select default, do not enter):
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
Retrieving file: /Image
append: root=/dev/mmcblk0p2 rootwait rw console=ttyLP0,115200
Retrieving file: /imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb
## Flattened Device Tree blob at 83000000
   Booting using the fdt blob at 0x83000000
Working FDT set to 83000000
   Using Device Tree in place at 0000000083000000, end 000000008300df6c
Working FDT set to 83000000
fail to find output device
probe video device failed, ret -19

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
```

uboot does not enable screens by default

Environment variables are automatically set based on screen ID values

ADC value corresponding to MIPI screen ID pin

Reserved for LVDS screen to manually select the device tree function, using MIPI screen or not using the screen, no manual selection

Start the device tree based on the MIPI screen ID obtained earlier

uboot does not enable screens by default

Figure 4.3-2 uboot prints the information

Due to the limited function of uboot, uboot multi-screen compatibility is not done here, and the screen function is not enabled by default.

By default, the system will identify the screen type according to the ID pin level value of the ALIENTEK MIPI screen. If you use the ALIENTEK MIPI screen or start without a screen, there is no need to manually change the device tree, and the corresponding device tree will be used to start in the Enter choice stage. The reason to keep the Enter choice function is that the ALIENTEK LVDS screen does not have ID pins, and if the ALIENTEK LVDS screen is used, the LVDS device tree can be selected at this stage.

If the user who does not use the screen or uses the ALIENTEK MIPI screen mistakenly touches the keyboard and stops at the Enter choice stage, he/she needs to manually input the corresponding number according to the screen type and then press Enter to start. If the selected device tree type is not correct, the screen display will be abnormal.

The numeric types correspond to the following:

Option	Corresponding device tree	Screen
1	imx93-11x11-atk.dtb	Instead of using the MIPI screen, the system self-starts the device tree based on the screen ID value
2	imx93-11x11-atk-mipi-dsi-5.5-720x1280.dtb	ALIENTEK 5.5 inch 720x1280 MIPI screen, the system according to the screen ID value to start the device tree
3	imx93-11x11-atk-mipi-dsi-5.5-1080x1920.dtb	ALIENTEK 5.5 inch 1080x1920 MIPI screen, the system automatically starts the device tree according to the screen ID value
4	imx93-11x11-atk-mipi-dsi-10.1-800x1280.dtb	ALIENTEK 10.1 inch 800x1280 MIPI screen, the system according to the screen ID value to start the device tree
5	imx93-11x11-atk-lvds-10.1-1280x800.dtb	ALIENTEK 10.1 inch 1280x800 LVDS screen needs to manually select the device tree to start

4.3.2 Touch test

Like the method in Section 4.2, after the development board is started, we can input the cat /proc/bus/input/devices instruction to check the corresponding touch event of the touch screen, or directly click and slide on the screen to test whether the screen is touched normally.

Note: The i.MX93 processor does not support the use of MIPI DSI and LVDS at the same time, so the ATK-DLIMX93 development board MIPI DSI and LVDS can not be used at the same time. In order to save pins, the development board MIPI DSI and LVDS screen touch use the same pin. Please ensure that the MIPI screen is not connected before using the LVDS screen touch function.

4.3.3 Backlight test

In the default factory system, the LCD screen backlight supports 100 levels of PWM adjustment, the brightness level is 0 ~ 100, the default is 100, the brightest state. When set to 0, the backlight is turned off. The higher the number, the brighter the screen.

```
cat /sys/class/backlight/backlight/max_brightness      //See the maximum level of backlight
cat /sys/class/backlight/backlight/brightness          //View the current brightness level
echo 50 > /sys/class/backlight/backlight/brightness
//Modify the current brightness level, viewing screen brightness changes
cat /sys/class/backlight/backlight/brightness          //To view the current brightness level
```

4.4 LVDS LCD test

This section is only applicable to the development board users who have selected the ALIENTEK 10.1inch LVDS screen.

4.4.1 Instructions for use

ATK-DLIMX93 is configured with one LVDS LCD interface, there is no ID identification pin, we need to manually select the device tree of LVDS to start. The ALIENTEK LVDS has a 10.1 inch screen size and 1280x800 resolution.

LVDS screen connection method: use the matching 40pin reverse soft cable for the screen cable, open the black clamp of the LVDS connection seat of the development board, connect the wire metal face down to the connection seat (pay attention to observe whether the wire metal falls off or is short-connected first), and close the clamp. The cable connection of the screen side is the same.

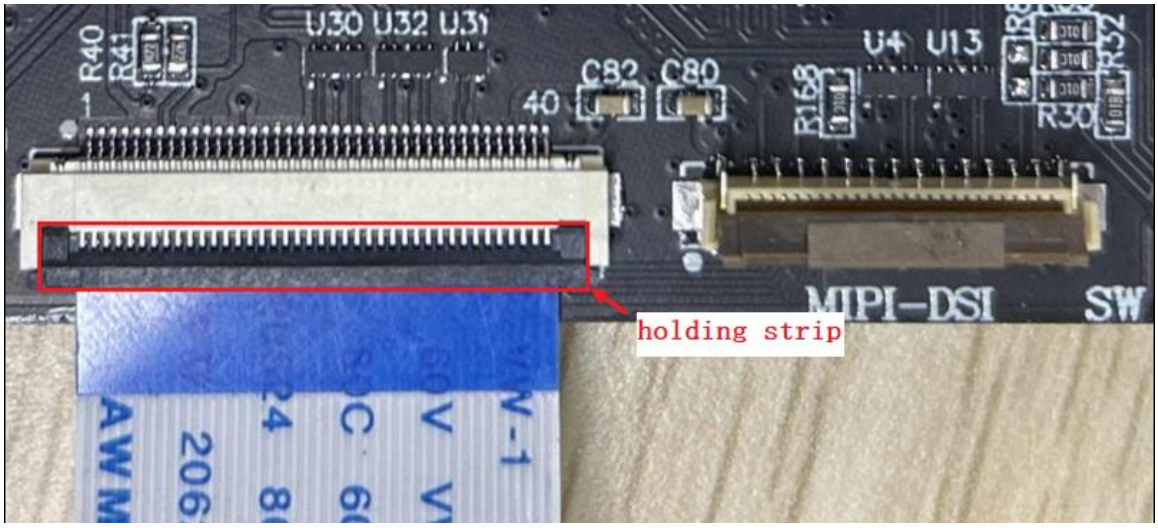


Figure 4.4-1 Development board LVDS connector

After the development board is connected to the LVDS screen and the USB_TTL interface, open the serial port terminal and start the development board to check the serial port information. When the development board runs to the Enter choice phase of uboot, press the Enter key of the keyboard to stay in the Enter choice phase, as shown in the following figure:

```
switch to partitions #0, OK
mmc0(part 0) is current device
panel=Without ATK-MIPI-LCD
flash target is MMC:0
Net:   eth0: ethernet@42890000, eth1: ethernet@428a0000 [PRIME]
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0(part 0) is current device
Scanning mmc 0:1...
Found /extlinux/extlinux.conf
Retrieving file: /extlinux/extlinux.conf
ADC1_CH3=1793mV
Select the boot mode
1:      imx93-11x11-atk
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
3:      imx93-11x11-atk-mipi-dsi-5.5-1080x1920
4:      imx93-11x11-atk-mipi-dsi-10.1-800x1280
5:      imx93-11x11-atk-lvds-10.1-1280x800
Enter choice(For ATK-LVDS-LCD-10.1, enter number 5. Otherwise, auto select default, do not enter):

 not found
```

When Enter choice appears, pressing the Enter key will stay in this phase

Figure 4.4-2 Enter choice phase

According to the information prompt, we need to enter the number 5 and press Enter to start with the corresponding lvds device tree.

```
Select the boot mode
1:      imx93-11x11-atk
2:      imx93-11x11-atk-mipi-dsi-5.5-720x1280
3:      imx93-11x11-atk-mipi-dsi-5.5-1080x1920
4:      imx93-11x11-atk-mipi-dsi-10.1-800x1280
5:      imx93-11x11-atk-lvds-10.1-1280x800
Enter choice(For ATK-LVDS-LCD-10.1, enter number 5. Otherwise, auto select default, do not enter):
5
5:      imx93-11x11-atk-lvds-10.1-1280x800
Retrieving file: /Image
append: root=/dev/mmcblk0p2 rootwait rw console=ttyLP0,115200
Retrieving file: /imx93-11x11-atk-lvds-10.1-1280x800.dtb
## Flattened Device Tree blob at 83000000
   Booting using the fdt blob at 0x83000000
Working FDT set to 83000000
   Using Device Tree in place at 0000000083000000, end 000000008300def5
Working FDT set to 83000000
fail to find output device
probe video device failed, ret -19

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
```

After entering the number 5, press Enter to start the lvds device tree

Figure 4.4-3 Start the lvds device tree

At this point observe the screen, you can see the LVDS screen started.

4.4.2 Touch test

Like the method in Section 4.2, after the development board is started, we can input the cat /proc/bus/input/devices instruction to check the corresponding touch event of the touch screen, or directly click and slide on the screen to test whether the screen is touched normally.

Note: The i.MX93 processor does not support the use of MIPI DSI and LVDS at the same time, so the ATK-DLIMX93 development board MIPI DSI and LVDS can not be used at the same time. In order to save pins, the development board MIPI DSI and LVDS screen touch use the same pin. Please ensure that the MIPI screen is not connected before using the LVDS screen touch function.

4.4.3 Backlight test

As with the methods in Section 4.3.2, we will not repeat them here.

4.5 Network Testing

4.5.1 Network test

ATK-DLIMX93 development board onboard 2-way Gigabit Ethernet, adaptive 10/100/1000M rate.

Firstly, insert the network cable into any network port to ensure that the network cable can access the Internet. The factory system of the development board will automatically obtain the Internet IP address according to the local network environment (the factory system supports DHCP), and input the ifconfig command to view the obtained network IP. As an example, the following access network port ENET1 corresponds to network node eth1. The network port ENET2 corresponds to the network node eth0, and the test method is the same.

```
ifconfig
```

```
root@atk-imx93:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.6.133  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::807c:5eff:fe13:cc0d  prefixlen 64  scopeid 0x20<link>
        ether 82:7c:5e:13:cc:0d  txqueuelen 1000  (Ethernet)
        RX packets 105  bytes 9856 (9.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 44  bytes 6897 (6.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28669<UP,BROADCAST,MULTICAST,DYNAMIC>  mtu 1500
        ether 7e:6e:ae:1a:af:d5  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 114

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 98  bytes 9176 (8.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 98  bytes 9176 (8.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.5-1 eth0 Gets the public ip address

Check whether the network can access the Internet, if there is a data reply, it means that the network function is normal. Press CTRL+C to terminate the command.

```
ping www.baidu.com
```

```
root@atk-imx93:~# ping www.baidu.com
PING www.baidu.com (183.2.172.42) 56(84) bytes of data.
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=1 ttl=51 time=5.40 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=2 ttl=51 time=5.15 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=3 ttl=51 time=5.05 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=4 ttl=51 time=5.19 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=5 ttl=51 time=5.30 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=6 ttl=51 time=5.17 ms
^C
--- www.baidu.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5006ms
rtt min/avg/max/mdev = 5.051/5.210/5.404/0.113 ms
root@atk-imx93:~#
```

Figure 4.5-2 ping Baidu

If the local network is not an Internet environment (LAN), test the network communication by configuring the static IP mode. For example, the development board configuration network port ENET2 network static IP: ifconfig eth0 192.168.6.100. PC (computer side) configuration static IP192.168.6.200. Both are tested for LAN ping communication. Note that the PC (computer side) needs to turn off the firewall.

4.5.2 Gigabit network test

Test whether it is a gigabit network, pay attention to the use of **gigabit network cable, gigabit router or gigabit switch**, PC (computer side) network card must be a gigabit network card, or the speed of the test or recognition may be 100 megabit.

In the figure below, when the system is identified as 1Gbps/Full, it is a gigabit network.

```
root@atk-imx93:~#
root@atk-imx93:~# [  173.251952] fec 42890000.ethernet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
[  173.259742] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
```

Figure 4.5-3 Identified as a gigabit network

Use the iperf3 command to test the gigabit network connection speed. Please ensure that the development board network can ping ubuntu properly before testing.

On ubuntu, set up ubuntu as the server by running the following command:

```
iperf3 -s
```

In the ATK-DLIMX93 development board command line, through the iperf3 command, enter the server IP.

Note that the server IP needs to be set according to the actual ubuntu IP of the user, not a uniform IP;

```
iperf3 -c 192.168.6.206 -i 10
```

Explanation:

(1) -c: Client.

(2) 192.168.6.206: The ubuntu ip address of the user, here is just an example.

(3) -i: The firing period.

The Bitrate I measured is around 900 Mbits/sec, which is in line with gigabit network speeds (please do not open other applications before testing, as this will not affect the test speed).

4.5.3 SSH login testing

After testing the network in the previous section, you can use SSH to log into the development board on the same LAN (provided that the current computer can ping the IP of the development board and enable SSH). Execute ifconfig to see the IP address of the current development board.

```
ifconfig
```

```
root@atk-imx93:~# ifconfig
eth0: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.204  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::8aec:faff:fe2f:ddae  prefixlen 64  scopeid 0x20<link>
        ether 88:ec:fa:2f:dd:ae  txqueuelen 1000  (Ethernet)
        RX packets 21  bytes 2240 (2.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 31  bytes 5002 (4.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28669<UP,BROADCAST,MULTICAST,DYNAMIC>  mtu 1500
        ether 88:73:03:9f:39:fc  txqueuelen 1000  (Ethernet)
        RX packets 937  bytes 87855 (85.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 79  bytes 10349 (10.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 115

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 132  bytes 11301 (11.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 132  bytes 11301 (11.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.5-4 Check the IP address of the development board

Use SSH command to log into the development board, example:

```
ssh root@192.168.6.204
```

root is the user name of the development board, the default is root. After @ is the IP address of the development board, please fill in according to the actual situation.

If a "WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! Prompt, please perform the prompt instruction SSH - the keygen -f "/ home/alientek /. SSH/known_hosts" -r "192.168.6.204" and then re-execute the SSH login instructions. Please enter yes and press Enter when prompted to connect. The default system is automatic login without password, if prompted to require a password, you can enter the password, the user name and password are root.

```
alientek@ubuntu:~$ ssh root@192.168.6.204
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
SHA256:Tir6YMW2+4t7EtK2pHU09CKs/62CUVoQoqjkSuCT3dY.
Please contact your system administrator.
Add correct host key in /home/alientek/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/alientek/.ssh/known_hosts:124
  remove with:
  ssh-keygen -f "/home/alientek/.ssh/known_hosts" -R "192.168.6.204"
ECDSA host key for 192.168.6.204 has changed and you have requested strict checking.
Host key verification failed.
alientek@ubuntu:~$   ssh-keygen -f "/home/alientek/.ssh/known_hosts" -R "192.168.6.204"
# Host 192.168.6.204 found: line 124
/home/alientek/.ssh/known_hosts updated.
Original contents retained as /home/alientek/.ssh/known_hosts.old
alientek@ubuntu:~$ ssh root@192.168.6.204
The authenticity of host '192.168.6.204 (192.168.6.204)' can't be established.
ECDSA key fingerprint is SHA256:Tir6YMW2+4t7EtK2pHU09CKs/62CUVoQoqjkSuCT3dY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.6.204' (ECDSA) to the list of known hosts.
Last login: Fri Feb  2 08:11:28 2024
root@atk-imx93:~#
root@atk-imx93:~# ls
minicom.log  shell
root@atk-imx93:~# exit
logout
Connection to 192.168.6.204 closed.
alientek@ubuntu:~$
```

SSH into the development board

ssh-keygen prompt

Confirm login

Login success

Log Out

Figure 4.5-5 SSH to the development board

The MobaXterm terminal can also have SSH terminal login function. The default SSH terminal is 22, as shown in the following figure:

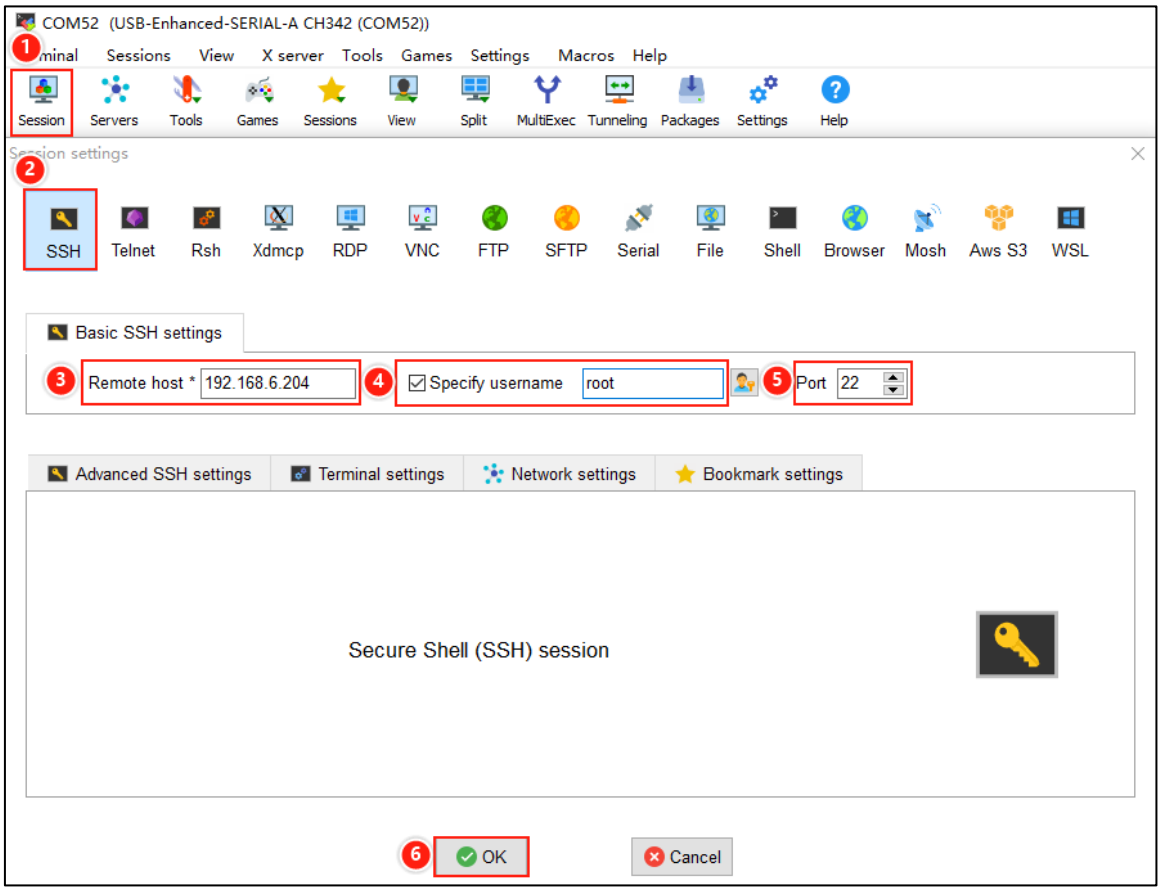


Figure 4.5-6 SSH in the MobaXterm terminal

4.6 Serial port test

4.6.1 RS485 serial port test

ATK-DLIMX93 development board onboard 2-channel RS485, before testing need to prepare the ALIENTEK USB converter to complete the test, as shown below:

Figure 4.6-1 ALIENTEK multi-in-one USB converter module

It is not necessary to use this module if you have other RS485related tests that you can test yourself. Since the development board has 2 RS485 channels, we can also use these 2 RS485 channels to communicate with each other to complete the test. The 2-way RS485 transceiver and transmitter test is demonstrated below.

The A end of RS485_1 is connected to the A end of RS485_2, and the B end of RS485_1 is connected to the B end of RS485_2 as shown below.

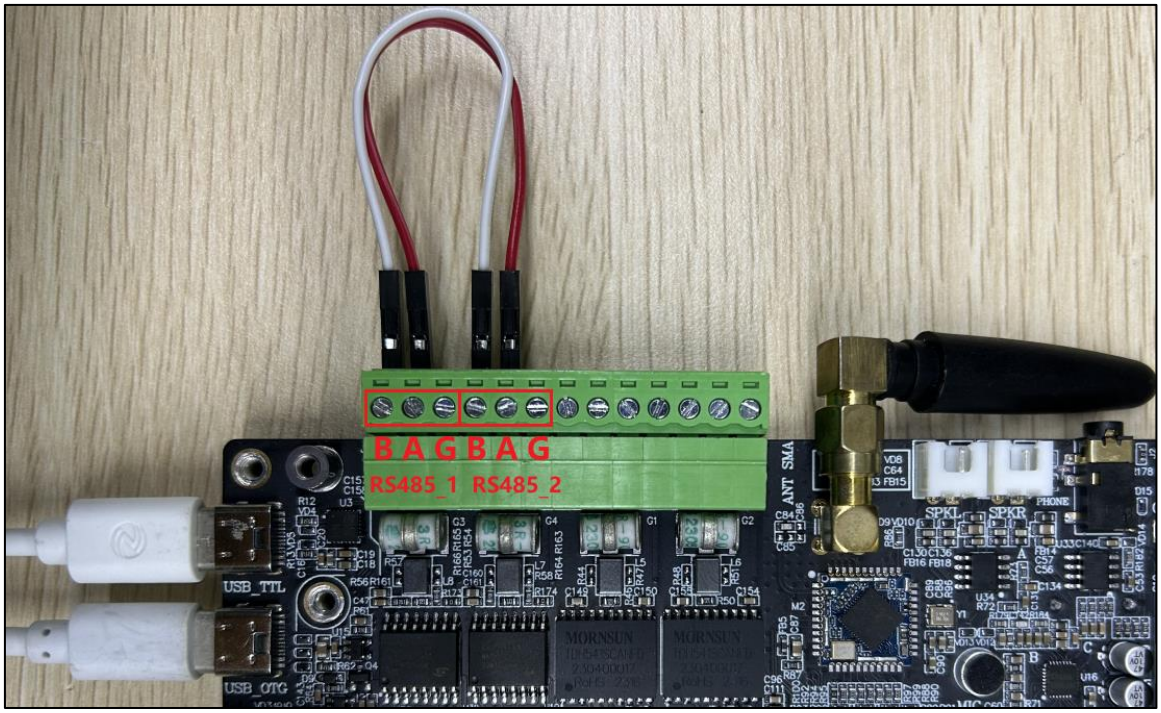


Figure 4.6-2 Two RS485 connection

Since we need to open two terminals to see the specific sending and receiving information, the one USB_TTL on the development board is not enough, so we need to view through SSH login. See Section 4.5.3 for SSH access to the development board.

To see which serial devices are enabled by default, execute the following command:

```
ls /dev/ttyLP*
```

```
root@atk-imx93:~# ls /dev/ttyLP*
/dev/ttyLP0  /dev/ttyLP3  /dev/ttyLP4  /dev/ttyLP7
root@atk-imx93:~#
```

Figure 4.6-3 Development board default serial port device

The corresponding peripheral information of serial port devices is shown in the following table:

Serial device	Development board peripherals
/dev/ttyLP0	USB_TTL（LPUART1）
/dev/ttyLP3	RS485_1（LPUART4）
/dev/ttyLP4	bluetooth（LPUART5）
/dev/ttyLP7	RS485_2（LPUART8）

Open RS485_1 using minicom-s in the serial terminal, as shown in the following screenshot:

```
A -    Serial Device      : /dev/ttyLP3
B - Lockfile Location     : /var/lock
C -   Callin Program      :
D -  Callout Program      :
E -    Bps/Par/Bits       : 115200 8N1
F - Hardware Flow Control : No
G - Software Flow Control : No
H -     RS485 Enable      : No
I -   RS485 Rts On Send   : No
J -  RS485 Rts After Send : No
K -  RS485 Rx During Tx   : No
L -  RS485 Terminate Bus  : No
M - RS485 Delay Rts Before: 0
N - RS485 Delay Rts After : 0

   Change which setting?
```

Figure 4.6-4 Configure RS485_1

Press crtl+A on the keyboard and then press Z to enter the minicom menu interface. Press E to enable input echo.

```
Welcome to minicom 2.8

OPTIONS: I18n
Compiled on Jan  1 2021, 17:4
Port /dev/ttyLP3, 10:46:12

Press CTRL-A Z for help on sp
```

```
                    Minicom Command Summary

          Commands can be called by CTRL-A <key>

           Main Functions                  Other Functions

Dialing directory..D  run script (Go)....G | Clear Screen.......C
Send files.........S  Receive files......R | cOnfigure Minicom..O
comm Parameters....P  Add linefeed.......A | Suspend minicom....J
Capture on/off.....L  Hangup.............H | eXit and reset.....X
send break.........F  initialize Modem...M | Quit with no reset.Q
Terminal settings..T  run Kermit.........K | Cursor key mode....I
lineWrap on/off....W  local Echo on/off..E | Help screen........Z
Paste file.........Y  Timestamp toggle...N | scroll Back........B
Add Carriage Ret...U

          Select function or press Enter for none.
```

Figure 4.6-5 RS485 1 Enable input echo

On the SSH side, use minicom-s and open RS485_2, as shown in the following figure:

```
A -    Serial Device      : /dev/ttyLP7
B - Lockfile Location     : /var/lock
C -   Callin Program      :
D -  Callout Program      :
E -    Bps/Par/Bits       : 115200 8N1
F - Hardware Flow Control : No
G - Software Flow Control : No
H -     RS485 Enable      : No
I -   RS485 Rts On Send   : No
J -  RS485 Rts After Send : No
K -  RS485 Rx During Tx   : No
L -  RS485 Terminate Bus  : No
M - RS485 Delay Rts Before: 0
N - RS485 Delay Rts After : 0

   Change which setting?
```

Figure 4.6-6 Configure RS485_2

Press crtl+A on the keyboard and then press Z to enter the minicom menu interface. Press E to enable input echo.

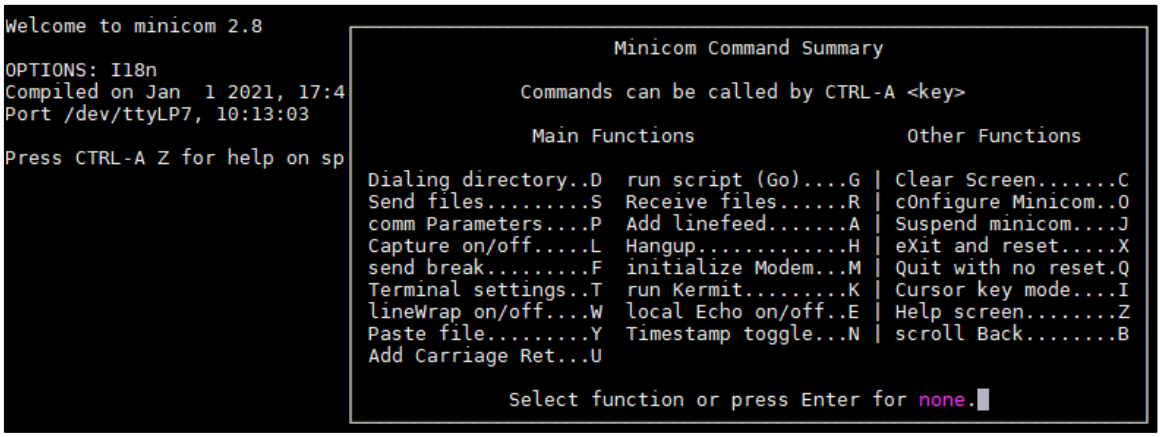


Figure 4.6-7 RS485_2 enables input echo

Type a character in RS485_1 and it will be displayed in RS485_2 (send in RS485_2 and receive in RS485_1) as shown below:

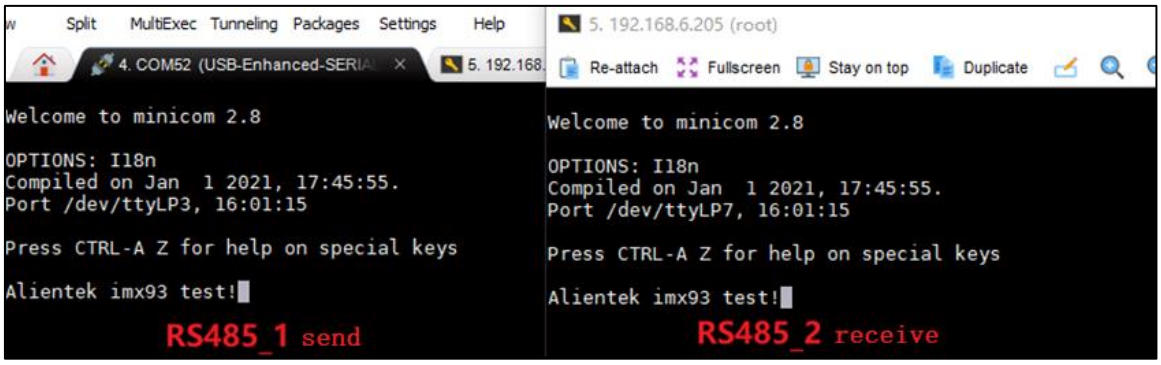


Figure 4.6-8 RS485 test

4.7 CAN Test

ATK-DLIMX93 development board onboard 2-way CAN (both support CAN FD). Before testing CAN, it is necessary to prepare the equipment for testing CAN, such as the CAN analyzer of Zhou Ligong, the CAN analyzer of Chuangxin Technology and the CAN analyzer of Guangcheng Technology, etc. Otherwise, it CAN only be tested with two different development boards or other CAN equipment. For the use of CAN instrument and CAN host computer, please refer to the operating instructions of each manufacturer's products.

The main features of the CAN used by the development board are as follows:

- Support CAN FD, which CAN be understood as an upgraded version of the CAN protocol. Only the protocol is upgraded, and the physical layer is not changed.
- Main differences between CAN and CAN FD: different transmission rate, different data length, different frame format, and different ID length.
- Data bit rate up to 5Mbps.

Before testing, please connect the CAN analyzer or the equipment for testing CAN to the CAN interface of the ATK-DLIMX93 development board. The CANH end is connected to the CANH of the CAN instrument, and the CANL end is connected to the CANL of the CAN instrument.

The following example uses FDCAN1 interface to carry out CAN function test, and FDCAN2 is the same. The CAN interface of the development board is shown below.

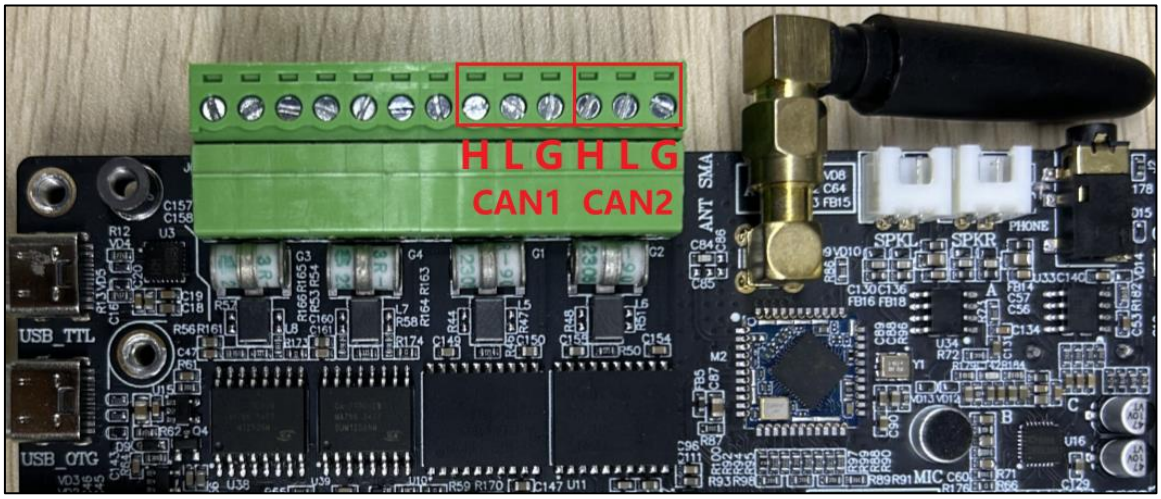


Figure 4.7-1 CAN interface location

The corresponding CAN devices in the system are as follows:

Peripheral device interface	System interface
CAN1	can0
CAN2	can1

4.7.1 CAN test

The minimum data bit rate of the CAN is 10kBit/s, and the maximum data bit rate is 1000kBit/s.

Before testing, note that the following example instructions are based on the FDCAN1 interface test, and the corresponding system device name is can0.

If the user needs to test FDCAN2 interface, the corresponding system device name is can1, pay attention to modify the instruction.

Example FDCAN1 interface, configured with can0 data bit rate of 500kBit/s.

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

Data is sent using the cansend command.

```
cansend can0 123#01.02.03.04.05.06.07.08
```

Explanation:

(1) can0: can0 represents FDCAN1 interface device, can1 represents FDCAN2 interface device.

(2) 2.123: Frame ID

(3) 01.02.03.04.05.06.07.08: Frame data

Receive data using the candump command. Press CTRL-C to terminate the command.

```
candump -ta can0
```

Explanation:

(1) -ta: t for print time, a for ASCII output.

The CAN analyzer of Guangcheng Technology is used in the following, and the rate of CAN is set to 500000. The test results are shown as follows.

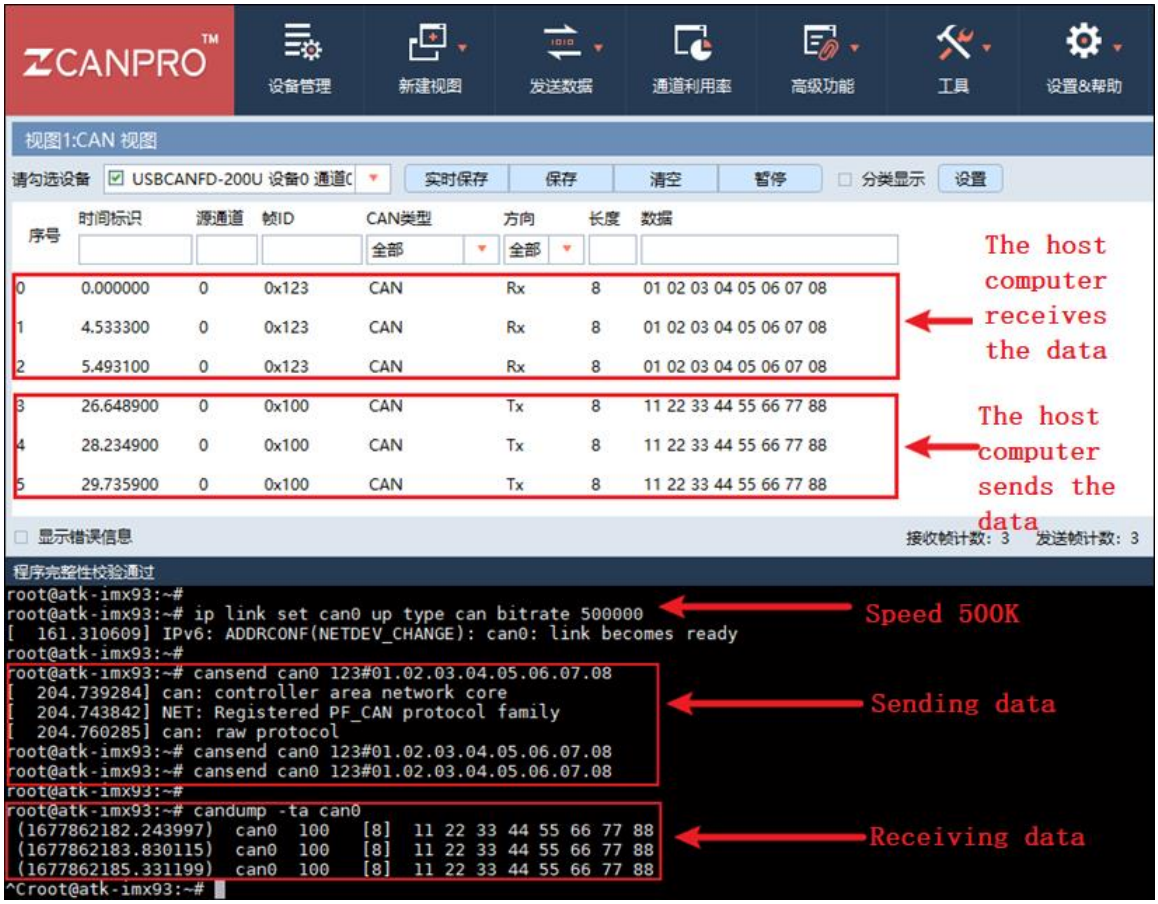


Figure 4.7-2 The host computer is used to send and receive CAN communication with the development board

4.7.2 CAN FD test

In the following, the CAN test equipment (supporting CAN FD) of Guangcheng Technology is used to conduct the CAN FD test. If the CAN test device does not support the CAN FD function, it cannot be used to test the CAN FD.

The development board has a maximum CAN FD baud rate of 1000 kBit/s and a data baud rate of 5000 kBit/s. The available rates are as follows: If you need to use a lower baud rate, use CAN instead of CAN FD.

Before testing, note that the following example instructions are based on the FDCAN1 interface test, and the corresponding system device name is can0. If the user needs to test the CAN2 interface, the corresponding system device name is can1, and pay attention to modify the instruction.

When setting different rates, it is necessary to turn off the CAN device first and then set the rate of CAN. If the setting is not successful, please restart the board or the host computer software.

```
ifconfig can0 down
```

An example rate of configured CAN FD is provided here.

```
ip link set can0 up type can bitrate 1000000 dbitrate 5000000 fd on
```

Here, the baud rate is set to 1000 kBit/s and the data baud rate is 5000 kBit/s for testing.

```
root@atk-imx93:~# ifconfig can0 down
root@atk-imx93:~# ip link set can0 up type can bitrate 1000000 dbitrate 5000000 fd on
[   61.498524] IPv6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready
root@atk-imx93:~#
```

Figure 4.7-3 The development board sets the CAN FD rate

Figure 4.7-4 Host computer configuration

Using the cansend command to send data, the following is a complete line of instructions, please copy and combine!

```
cansend can0 123##300.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff
```

Explanation:

(1) can0: can0 represents FDCAN1 interface device, can1 represents FDCAN2 interface device.

(2) 123: Frame ID

(3) 3: flags

(4) 00.11.22... : Frame data

Receive data using the candump command. Press CTRL-C to terminate the command.

```
candump -ta can0
```

Explanation:

(1) -ta: t for print time, a for ASCII output

The development board CAN FD test results are shown below.

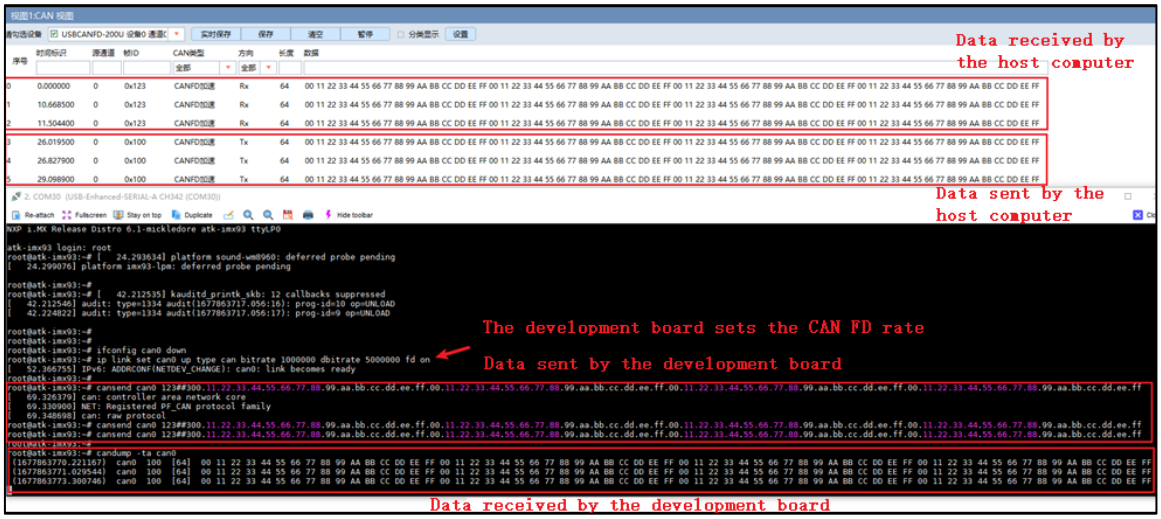


Figure 4.7-5 The host computer and the development board are used for CAN FD transceiver communication

4.7.3 Onboard dual-channel CANFD test

Since the development board has 2 CANFDs, we can also use these 2 CANFDs to communicate with each other to complete the test. The 2-way CANFD transceiver and transmitter test is demonstrated in the following.

The H end of CAN1 is connected to the H end of CAN2, and the L end of CAN1 is connected to the L end of CAN2, as shown below.

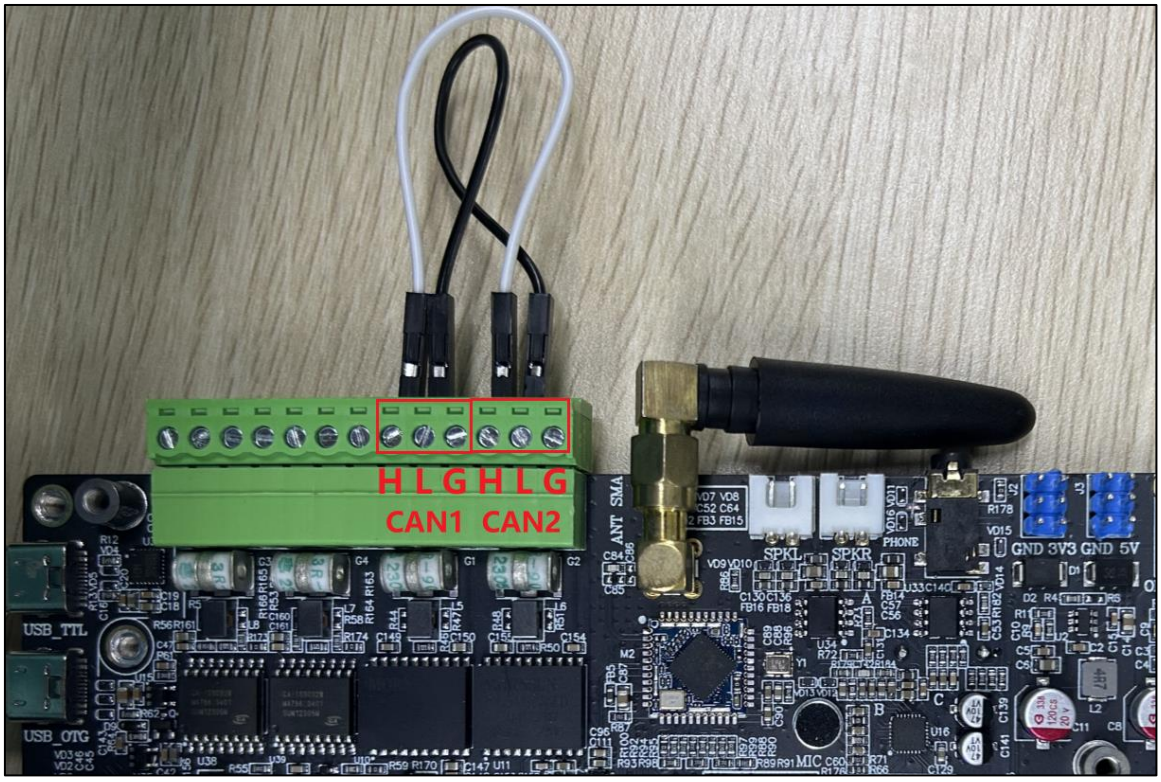


Figure 4.7-6 The on-board CAN interconnect

Since we need to open two terminals to see the specific sending and receiving information, the one USB_TTL on the development board is not enough, so we need to view through SSH login. See Section 4.5.3 for SSH access to the development board.

To enable and configure can0 and can1, run the following command:

```
ip link set can0 up type can bitrate 1000000 dbitrate 5000000 fd on
ip link set can1 up type can bitrate 1000000 dbitrate 5000000 fd on
```

Here, the baud rate is set to 1000 kBit/s and the data baud rate is 5000 kBit/s for testing.

```
root@atk-imx93:~# ip link set can0 up type can bitrate 1000000 dbitrate 5000000 fd on
[   69.302396] IPv6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready
root@atk-imx93:~# ip link set can1 up type can bitrate 1000000 dbitrate 5000000 fd on
[   77.600462] IPv6: ADDRCONF(NETDEV_CHANGE): can1: link becomes ready
root@atk-imx93:~#
```

Figure 4.7-7 Enable can0 and can1

Log on the development board at the SSH side and execute the following instructions to let can1 use candump instruction to receive can0 data. Press CTRL-C to terminate the command.

```
candump -ta can1
```

Explanation:

(1) -ta: t for print time, a for ASCII output

The serial port executes the following command to let can0 use the cansend command to send data. The following is a complete line of instructions, please copy and combine!

```
cansend can0 123##300.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff.00.11.22.33.44.55.66.77.88.99.aa.bb.cc.dd.ee.ff
```

Explanation:

(1) can0: can0 represents FDCAN1 interface device.

(2) 123: Frame ID

(3) 3: flags

(4) 00.11.22... : Frame data

The results are as follows:

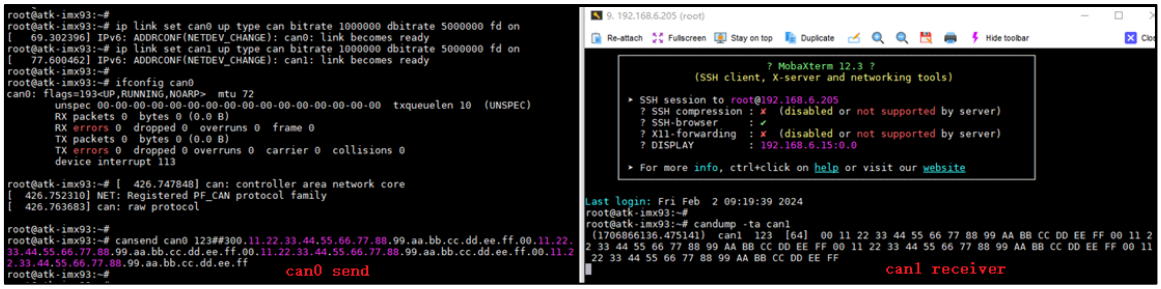


Figure 4.7-8 can test

4.8 USB Testing

4.8.1 USB HOST test

Get a FAT32 USB flash drive or a TF card reader. The author uses TF card + card reader and inserts the device into the USB interface of USB_HOST1 or USB_HOST 2 or USB_HOST 3 of the ATK-DLIMX93 development board, as shown in the following figure:

Figure 4.8-1 TF card + card reader access to the development board USB HOST interface

The factory system will pop up the following identification information.

```
root@atk-imx93:~# [  197.995770] usb 1-1.3: new high-speed USB device number 3 using ci_hdrc
[  198.120097] usb-storage 1-1.3:1.0: USB Mass Storage device detected
[  198.127484] scsi host0: usb-storage 1-1.3:1.0
[  199.140670] scsi 0:0:0:0: Direct-Access     Mass     Storage Device   1.00 PQ: 0 ANSI: 0 CCS
[  199.281799] sd 0:0:0:0: [sda] 30277632 512-byte logical blocks: (15.5 GB/14.4 GiB)
[  199.289829] sd 0:0:0:0: [sda] Write Protect is off                                  capacity
[  199.295163] sd 0:0:0:0: [sda] No Caching mode page found
[  199.300500] sd 0:0:0:0: [sda] Assuming drive cache: write through
[  199.311433]  sda: sda1 Node name
[  199.319827] sd 0:0:0:0: [sda] Attached SCSI removable disk

root@atk-imx93:~#
```

Figure 4.8-2 U disk identification information

Type the df -h command to see the current mount path of the USB key.

```
df -h
```

```
root@atk-imx93:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        14G  3.3G  9.8G  25% /
devtmpfs        341M  4.0K  341M   1% /dev
tmpfs           471M     0  471M   0% /dev/shm
tmpfs           189M  9.6M  179M   6% /run
tmpfs           4.0M     0  4.0M   0% /sys/fs/cgroup
tmpfs           471M  8.0K  471M   1% /tmp
tmpfs           471M  156K  471M   1% /var/volatile
tmpfs            95M     0   95M   0% /run/user/0
/dev/mmcblk0p1  500M   33M  468M   7% /run/media/mmcblk0p1 U disk usage information
/dev/sda1        15G  128K   15G   1% /run/media/sda1       and mount path
root@atk-imx93:~#
```

Figure 4.8-3 Check the U disk mount path

As can be seen from the above figure, the current U disk mount path is /media/sda1, and the U disk read and write speed test is carried out below.

Tips, the reader itself interface read and write speed and TF card type, U disk type will also affect the speed value of fast or slow, the user

The test values may be different from the editor test values, and the speed values are for reference only.

Write speed test:

To write a test file to the USB key, enter the following command:

```
time dd if=/dev/zero of=/run/media/sda1/test bs=1024k count=100 conv=fdatasync
```

```
root@atk-imx93:~# time dd if=/dev/zero of=/run/media/sda1/test bs=1024k count=100 conv=fdatasync
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 8.91817 s, 11.8 MB/s

real    0m8.925s
user    0m0.005s
sys     0m0.526s
root@atk-imx93:~#
```

The above means writing a 100MiB test file with a write speed of 11.8MB/s. The larger the actual file, the closer the average write speed is to the actual value. Before you unplug the USB flash drive, run the sync command to sync it to the storage disk.

Reading speed test:

Type the following to clear the run cache:

```
echo 3 > /proc/sys/vm/drop_caches
```

Note: Because of the LINUX kernel mechanism, it is not necessary to release a used cache. These cache contents can increase the speed of reading and writing files. In this case, to test the read and write speed, we need to perform a release.

Type the following to read the test file you wrote earlier using the dd instruction:

```
time dd if=/run/media/sda1/test of=/dev/null bs=1024k
```

```
root@atk-imx93:~# echo 3 > /proc/sys/vm/drop_caches
[  520.720730] sh (494): drop_caches: 3
root@atk-imx93:~# time dd if=/run/media/sda1/test of=/dev/null bs=1024k
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 5.6851 s, 18.4 MB/s

real    0m5.692s
user    0m0.000s
sys     0m0.231s
root@atk-imx93:~#
```

Figure 4.8-4 Read speed test

This means reading 100MiB of data at 18.4MB/s. The reading and writing speed of the card reader will also affect this speed value, which is for reference only.

4.8.2 USB Mouse test

Note: Use factory file system, plug in the ALIENTEK MIPI or LVDS screen before boot.

After the development board enters the system, the following information will be printed when the mouse is plugged in.

```
root@atk-imx93:~# [   40.001362] usb 1-1.3: new full-speed USB device number 3 using ci_hdrc
[   40.009615] systemd-journald[180]: Oldest entry in /run/log/journal/740664299ec147289546f11b53f7d122/system.journal i
s older than the configured file retention duration (1month), suggesting rotation.
[   40.027381] systemd-journald[180]: /run/log/journal/740664299ec147289546f11b53f7d122/system.journal: Journal header l
imits reached or header out-of-date, rotating.
[   40.118168] input: MOSART Semi. 2.4G INPUT DEVICE as /devices/platform/soc@0/4c200000.usb/ci_hdrc.1/usb1/1-1/1-1.3/1-
1.3:1.0/0003:062A:4180.0001/input/input6
[   40.198035] hid-generic 0003:062A:4180.0001: input: USB HID v1.10 Keyboard [MOSART Semi. 2.4G INPUT DEVICE] on usb-ci
_hdrc.1-1.3/input0
[   40.220545] input: MOSART Semi. 2.4G INPUT DEVICE Mouse as /devices/platform/soc@0/4c200000.usb/ci_hdrc.1/usb1/1-1/1-
1.3/1-1.3:1.1/0003:062A:4180.0002/input/input7
[   40.236356] input: MOSART Semi. 2.4G INPUT DEVICE Consumer Control as /devices/platform/soc@0/4c200000.usb/ci_hdrc.1/
usb1/1-1/1-1.3/1-1.3:1.1/0003:062A:4180.0002/input/input8
[   40.310518] input: MOSART Semi. 2.4G INPUT DEVICE System Control as /devices/platform/soc@0/4c200000.usb/ci_hdrc.1/us
b1/1-1/1-1.3/1-1.3:1.1/0003:062A:4180.0002/input/input9
[   40.327053] input: MOSART Semi. 2.4G INPUT DEVICE as /devices/platform/soc@0/4c200000.usb/ci_hdrc.1/usb1/1-1/1-1.3/1-
1.3:1.1/0003:062A:4180.0002/input/input10
[   40.342293] hid-generic 0003:062A:4180.0002: input: USB HID v1.10 Mouse [MOSART Semi. 2.4G INPUT DEVICE] on usb-ci_hd
rc.1-1.3/input1
```

Figure 4.8-5 Information printed when the mouse is plugged in

At the same time, the factory Qt desktop also shows the mouse, you can use the mouse to click the operation.

4.9 RTC clock test

ATK-DLIMX93 development board has two hardware clocks, one is the core board chip internal RTC clock (/sys/class/rtc/rtc0), development board power failure does not save time data and timing. The other is the RTC clock chip PCF8563 on the base board, which belongs to the external RTC clock (/sys/class/rtc/rtc1) of the chip. It is powered by the RTC button battery, and the time and timing are still saved when the power is off. If you need to improve the accuracy of the clock, you need to use high precision crystal.

Please check whether the RTC button battery is installed on the development board. You can also use a multimeter to check whether the RTC battery has power, and it is normal to measure about 3.3v. Prevent the RTC battery power can not save time. (Note that the battery is a consumable, if no power, please replace the button battery)

The Linux system is divided into two clocks, one is the system time (software clock), the other is the hardware clock (hardware clock). The date and hwclock commands can be used to view and set the system and hardware time, respectively. The system clock will disappear when the power is off, and the RTC clock will run for a long time when there is a battery. The system clock is synchronized with the RTC clock when the system is restarted.

To check the system clock, use the date command. Note that the system clock of the factory system uses ntpd clock synchronization service, so if you have an Internet cable and access to the Internet when you boot up, your system clock will be synchronized with the clock on the Internet. However, this synchronized clock is not written to the hardware clock. So you might see two different times using date and hwclock.

```
date
```

Write time to PCF8563:

```
date -s "2024-01-26 10:08:00"
hwclock --systohc --rtc=/dev/rtc1 --localtime
```

Query PCF8563 time:

```
cat /sys/class/rtc/rtc1/time
cat /sys/class/rtc/rtc1/date
date
```

```
root@atk-imx93:~# date
Fri Jan 26 11:11:40 AM CST 2024
root@atk-imx93:~# date -s "2024-01-26 10:08:00"
Fri Jan 26 10:08:00 AM CST 2024
root@atk-imx93:~# hwclock --systohc --rtc=/dev/rtc1 --localtime
root@atk-imx93:~# cat /sys/class/rtc/rtc1/time
10:08:16
root@atk-imx93:~# cat /sys/class/rtc/rtc1/date
2024-01-26
root@atk-imx93:~# date
Fri Jan 26 10:08:26 AM CST 2024
```

Figure 4.9-1 Set the RTC time

It can be seen that the system time and the RTC time agree. Because the system time is in CST, the 12-hour clock is used, so if 17:00:00 is written, then 05:00:00 PM is displayed.

If the development board is networked, the system can obtain the time through the network, so that the time is more accurate, the default system will automatically obtain the network time when starting. If we want to regain the network time, we can run the following command:

```
systemctl restart ntpd
```

After a few moments, you can check the system time, which is the same as the network time.

```
date
```

The obtained network time is not automatically written to the RTC time, and the RTC time needs to be rewritten according to the previous instructions to modify it.

4.10 EEPROM test

ATK-DLIMX93 development board onboard EEPROM chip, model is AT24C64, it reads and writes through the I2C bus access, with 8KB storage space.

The factory system provides a simple read and write test tool eeprom-rw to carry out the read and write test of AT24C64. Execute the following instructions.

```
cd shell/eeprom
./eeprom-rw
```

The following message is returned to indicate that the read and write operation was normal:

```
root@atk-imx93:~# cd shell/eeprom/
root@atk-imx93:~/shell/eeprom# ls
eeprom-rw
root@atk-imx93:~/shell/eeprom# ./eeprom-rw
len = 32
writing...

===write ok===
reading...
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!Hello,this is an eeprom test!!!!
```

Figure 4.10-1 eeprom read and write test

4.11 SPI FLASH test

ATK-DLIMX93 development board on-board SPI NOR FLASH chip, model is W25Q128, it reads and writes through SPI bus access, with 16MB storage space.

The W25Q128 driver will be loaded automatically when the kernel is booted, and the kernel will drive the loading information when the chip is working normally. The factory system registers the SPI Flash as an MTD device, so we can read and write files to it through the MTD subsystem.

```
ls /dev/mtd*
```

```
root@atk-imx93:~# ls /dev/mtd*
/dev/mtd0  /dev/mtd0ro  /dev/mtdblock0
root@atk-imx93:~#
```

Figure 4.11-1 Look at the mtd device

The file read and write tests are performed as follows:

Format the /dev/mtdblock0 partition once for the first time. Note that if you are not using the format command for the first time, the data written to the file will be erased and will need to be written again.

```
mkfs.vfat /dev/mtdblock0
```

1. Create a custom mount directory and mount the /dev/mtdblock0 partition to this directory in VFAT format.

```
mkdir /home/root/w25q128
mount -t vfat /dev/mtdblock0 /home/root/w25q128
```

2. Enter the mount directory, write file test;

```
cd /home/root/w25q128
echo "W25Q128 TEST!" > file.txt
sync
```

3. Read the file and confirm whether it was written successfully.

```
cat file.txt
```

4. Finally exit the mount directory, and cancel the /dev/mtdblock0 partition mount;

```
cd ../
umount /dev/mtdblock0
```

At this time, the file data written has been stored in the SPI FLASH chip, and the file can be read and written by mounting the directory again or restarting the board.

```
root@atk-imx93:~# mkfs.vfat /dev/mtdblock0
mkfs.fat 4.2 (2021-01-31)
root@atk-imx93:~# mkdir /home/root/w25q128
root@atk-imx93:~# mount -t vfat /dev/mtdblock0 /home/root/w25q128
root@atk-imx93:~# cd /home/root/w25q128
root@atk-imx93:~/w25q128# echo "W25Q128 TEST!" > file.txt
root@atk-imx93:~/w25q128# sync
root@atk-imx93:~/w25q128# cat file.txt
W25Q128 TEST!
root@atk-imx93:~/w25q128# cd ../
root@atk-imx93:~# umount /dev/mtdblock0
root@atk-imx93:~#
```

Figure 4.11-2 SPI FLASH read and write file test process

4.12 Audio testing

ATK-DLIMX93 development board has a high performance audio codec chip ES8388 on board. The on-board microphone can record, and a small power speaker is connected to the back of the bottom plate to facilitate the user to conduct audio testing.

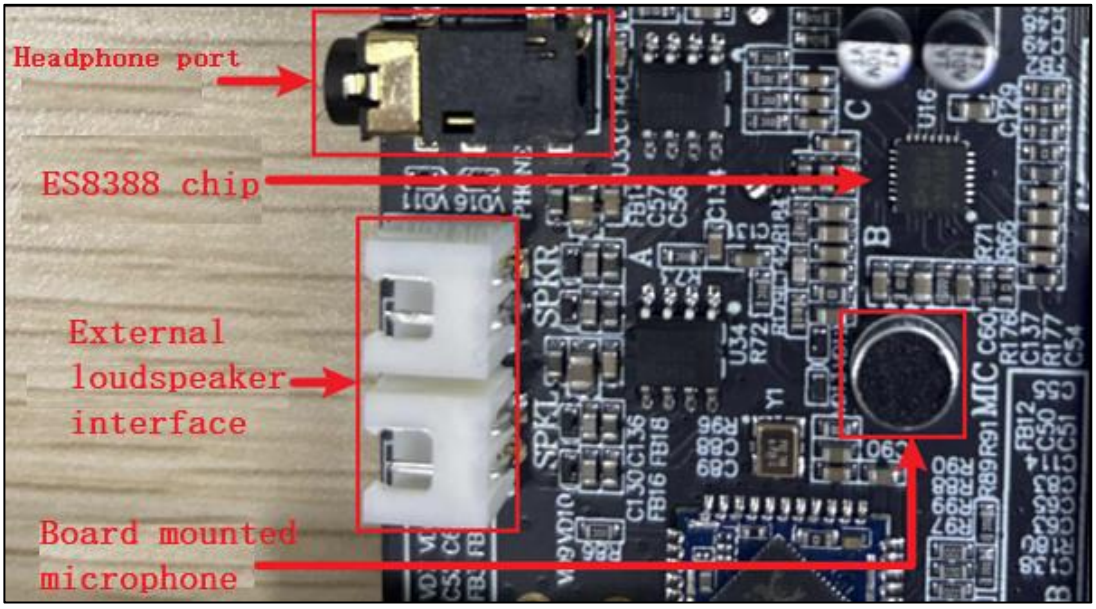


Figure 4.12-1 Development board front audio section

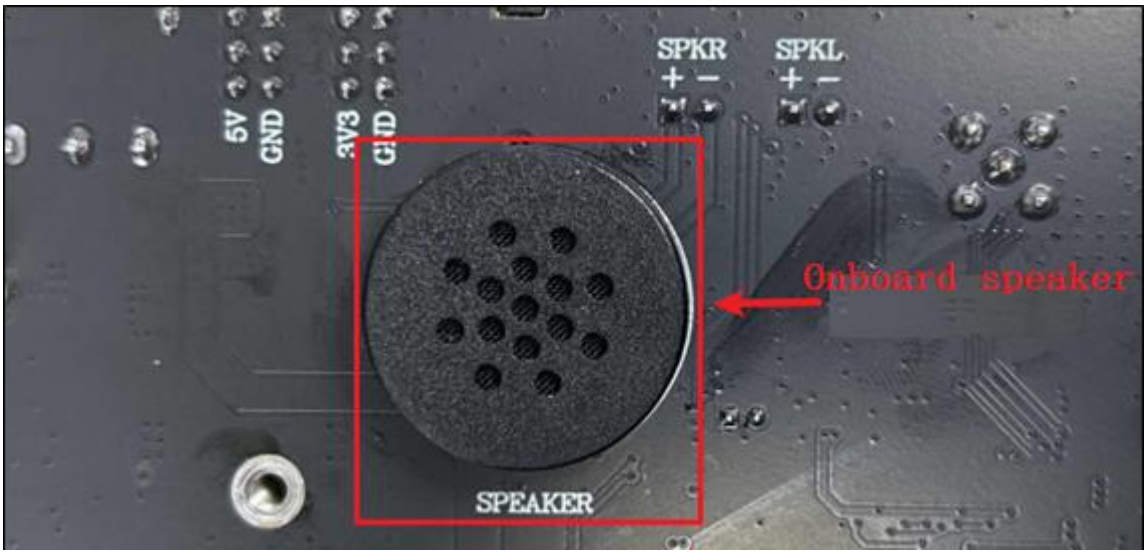


Figure 4.12-2 the audio section on the back of the development board

4.12.1 Playing the audio test

There are audio configuration and test files in the factory system of the development board. Execute the audio test script according to the following instructions.

```
cd shell/audio
./atk_audioTest.sh
```

The first time this script is run, the initialization of the audio device will be printed, and the subsequent execution of this script will not print the initialization information. Press crtl+c to exit the script.

After initializing the audio device, enter the number 2 and confirm to play the audio test. The playback information is as follows. Audio will be played from the onboard speakers, or on the headset side if headphones are used.

```
root@atk-imx93:~/shell/audio# ./atk_audioTest.sh

可选ES8388测试项目:
1. 录音测试
2. 播音测试
输入您需要测试的项目编号（1或2）: 2

开始播音，按 Ctrl+C 可退出播音
Press 'k' to see a list of keyboard shortcuts.
Now playing /home/root/shell/audio/short.mp3
Prerolling...
====== BEEP: 4.8.2 build on Oct 10 2023 02:31:27. ======
        Core: MP3 decoder Wrapper  build on Sep  8 2023 20:21:10
 file: /usr/lib/imx-mm/audio-codec/wrap/lib_mp3d_wrap_arm_elinux.so.3
CODEC: BLN_MAD-MMCODECS_MP3D_ARM_02.13.01_ARMV8  build on Sep  8 2023 17:49:07.
Redistribute latency...
Redistribute latency...
0:00:38.4 / 0:03:10.0
清理并退出...
root@atk-imx93:~/shell/audio#
```

Enter the number 2 and press Enter to confirm

Audio playback information

Press CTRL-C to exit and print this message

Figure 4.12-3 Audio Playback test

4.12.2 Recording test

There are audio configuration and test files in the factory system of the development board. Execute the audio test script according to the following instructions.

```
cd shell/audio
./atk_audioTest.sh
```

The first time this script is run, the initialization of the audio device will be printed, and the subsequent execution of this script will not print the initialization information. Press crtl+c to exit the script.

After initializing the audio device, enter the number 1 to confirm, and then select the microphone test item in the next step. If the headset with microphone is connected to the PHONE interface of the development board, use 1. Headphone microphone; If the headset is not connected, directly use the on-board microphone MIC of the development board, then use 2. On-board microphone. Here, the author uses the on-board microphone of the development board, and chooses the second item.

```
可选ES8388测试项目:
1. 录音测试
2. 播音测试
输入您需要测试的项目编号（1或2）: 1

可选麦克风测试项目:
1. 耳机麦克风
2. 板载麦克风
请输入您的选择: 2

应用麦克风配置项 2
numid=32,iface=MIXER,name='Left Mixer Left Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on
numid=34,iface=MIXER,name='Right Mixer Right Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on
numid=31,iface=MIXER,name='Left Mixer Left Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=33,iface=MIXER,name='Right Mixer Right Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=27,iface=MIXER,name='Differential Mux'
  ; type=ENUMERATED,access=rw------,values=1,items=2
  ; Item #0 'Line 1'
  ; Item #1 'Line 2'
  : values=1
numid=29,iface=MIXER,name='Left Line Mux'
  ; type=ENUMERATED,access=rw------,values=1,items=4
  ; Item #0 'Line 1L'
  ; Item #1 'Line 2L'
  ; Item #2 'NC'
  ; Item #3 'MicL'
  : values=1
numid=30,iface=MIXER,name='Right Line Mux'
  ; type=ENUMERATED,access=rw------,values=1,items=4
  ; Item #0 'Line 1R'
  ; Item #1 'Line 2R'
  ; Item #2 'NC'
  ; Item #3 'MicR'
  : values=1
```

Enter the number 1 and press Enter to confirm

Record using the onboard microphone by entering the number 2 and pressing Enter to confirm

Microphone configuration information

Figure 4.12-4 An on-board microphone configuration is used

After selecting the corresponding microphone configuration, the script will automatically record. Please approach the microphone for recording test.

```
开始录音...
Recording WAVE '/home/root/shell/audio/record.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
numid=32,iface=MIXER,name='Left Mixer Left Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=34,iface=MIXER,name='Right Mixer Right Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=31,iface=MIXER,name='Left Mixer Left Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on
numid=33,iface=MIXER,name='Right Mixer Right Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on

播放录音...
Playing WAVE '/home/root/shell/audio/record.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
numid=32,iface=MIXER,name='Left Mixer Left Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=34,iface=MIXER,name='Right Mixer Right Bypass Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=31,iface=MIXER,name='Left Mixer Left Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=33,iface=MIXER,name='Right Mixer Right Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=off
numid=29,iface=MIXER,name='Left Line Mux'
  ; type=ENUMERATED,access=rw------,values=1,items=4
  ; Item #0 'Line 1L'
  ; Item #1 'Line 2L'
  ; Item #2 'NC'
  ; Item #3 'MicL'
  : values=2
numid=30,iface=MIXER,name='Right Line Mux'
  ; type=ENUMERATED,access=rw------,values=1,items=4
  ; Item #0 'Line 1R'
  ; Item #1 'Line 2R'
  ; Item #2 'NC'
  ; Item #3 'MicR'
  : values=2
```

Automatic recording, please speak close to the on-board microphone

When the recording is finished, the recorded audio file will be played automatically

Figure 4.12-5 Recording test

After the recording is completed, the record.wav file will be generated in the current directory, which is the audio file that the author just recorded. After the script finishes playing, the audio configuration will be turned off by default. You need to manually execute the command to enable it, and then you can use aplay to play the recorded audio file again.

```
amixer cset name='Left Mixer Left Playback Switch' 'on'
amixer cset name='Right Mixer Right Playback Switch' 'on'
aplay record.wav
```

```
root@atk-imx93:~/shell/audio# ls
atk_audioTest.sh  record.wav  short.mp3
 'on'atk-imx93:~/shell/audio# amixer cset name='Left Mixer Left Playback Switch'
numid=31,iface=MIXER,name='Left Mixer Left Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on
root@atk-imx93:~/shell/audio# amixer cset name='Right Mixer Right Playback Switch' 'on'
numid=33,iface=MIXER,name='Right Mixer Right Playback Switch'
  ; type=BOOLEAN,access=rw------,values=1
  : values=on
root@atk-imx93:~/shell/audio# aplay record.wav
Playing WAVE 'record.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
root@atk-imx93:~/shell/audio#
```

Figure 4.12-6 aplay plays the recorded audio file

When the test is complete, turn off the audio configuration.

```
amixer cset name='Left Mixer Left Playback Switch' 'off'
amixer cset name='Right Mixer Right Playback Switch' 'off'
```

4.13 Onboard SDIO WIFI test

ATK-DLIMX93 development board uses WIFI& Bluetooth two-in-one module, the chip is RTL8723DS.

Before the WIFI test, please make sure that the antenna is installed on the development board, otherwise the WIFI cannot be connected.

Once in the filesystem, enter the following command to test the WIFI.

```
connmanctl
connmanctl> enable wifi
connmanctl> scan wifi
connmanctl> agent on
connmanctl> services
```

Here's the result:

```
root@atk-imx93:~# connmanctl
connmanctl> enable wifi
wifi is already enabled
connmanctl> scan wifi
Scan completed for wifi
connmanctl> agent on
Agent registered
connmanctl> services
*AR Wired                ethernet_2aa450a53aef_cable
    ALIENTEK-YY          wifi_94a4082f8dfa_414c49454e54454b2d5959_managed_psk
    wifi6_5g             wifi_94a4082f8dfa_77696669365f3567_managed_psk
    ChinaNet-ymyk        wifi_94a4082f8dfa_4368696e614e65742d796d796b_managed_psk
                         wifi_94a4082f8dfa_hidden_managed_psk
    RICH901              wifi_94a4082f8dfa_52494348393031_managed_psk
    ALIENTEK-7机房     wifi_94a4082f8dfa_414c49454e54454b2d37e69cbae688bf_managed_psk
    ALIENTEK-WX          wifi_94a4082f8dfa_414c49454e54454b2d5758_managed_psk
    DIRECT-HCDESKTOP-7A885MVmsJN wifi_94a4082f8dfa_4449524543542d48434445534b544f502d37413838354d566d734a4e_ma
naged_psk
    ATK-HUAWEI           wifi_94a4082f8dfa_41544b2d485541574549_managed_psk
    ALIENTEK-SYS         wifi_94a4082f8dfa_414c49454e54454b2d535953_managed_psk
    ALIENTEK-XHYS        wifi_94a4082f8dfa_414c49454e54454b2d58485953_managed_psk
    ALIENTEK-SC          wifi_94a4082f8dfa_414c49454e54454b2d5343_managed_psk
    ALIENTEK-7办公     wifi_94a4082f8dfa_414c49454e54454b2d37e58a9ee585ac_managed_psk
    ALIENTEK-YF-SYS      wifi_94a4082f8dfa_414c49454e54454b2d59462d535953_managed_psk
    SSID-2.4G            wifi_94a4082f8dfa_535349442d322e3447_managed_none
    606-2                wifi_94a4082f8dfa_3630362d32_managed_psk
connmanctl>
```

Figure 4.13-1 Scanned hot spots

From the list of scanned WIFI hotspots, use the command connect XXXX to connect, then enter the password and press Enter to confirm. For example, if we wanted to connect to ALIENTEK-YY, it would look like this:

```
connmanctl> services
*AR Wired                ethernet_2aa450a53aef_cable
    ALIENTEK-YY          wifi_94a4082f8dfa_414c49454e54454b2d5959_managed_psk
    wifi6_5g             wifi_94a4082f8dfa_77696669365f3567_managed_psk
    ChinaNet-ymyk        wifi_94a4082f8dfa_4368696e614e65742d796d796b_managed_psk
                         wifi_94a4082f8dfa_hidden_managed_psk
    RICH901              wifi_94a4082f8dfa_52494348393031_managed_psk
    ALIENTEK-7机房     wifi_94a4082f8dfa_414c49454e54454b2d37e69cbae688bf_managed_psk
    ALIENTEK-WX          wifi_94a4082f8dfa_414c49454e54454b2d5758_managed_psk
    DIRECT-HCDESKTOP-7A885MVmsJN wifi_94a4082f8dfa_4449524543542d48434445534b544f502d37413838354d566d734a4e_ma
naged_psk
    ATK-HUAWEI           wifi_94a4082f8dfa_41544b2d485541574549_managed_psk
    ALIENTEK-SYS         wifi_94a4082f8dfa_414c49454e54454b2d535953_managed_psk
    ALIENTEK-XHYS        wifi_94a4082f8dfa_414c49454e54454b2d58485953_managed_psk
    ALIENTEK-SC          wifi_94a4082f8dfa_414c49454e54454b2d5343_managed_psk
    ALIENTEK-7办公     wifi_94a4082f8dfa_414c49454e54454b2d37e58a9ee585ac_managed_psk
    ALIENTEK-YF-SYS      wifi_94a4082f8dfa_414c49454e54454b2d59462d535953_managed_psk
    SSID-2.4G            wifi_94a4082f8dfa_535349442d322e3447_managed_none
    606-2                wifi_94a4082f8dfa_3630362d32_managed_psk
connmanctl> connect wifi_94a4082f8dfa_414c49454e54454b2d5959_managed_psk   <- Connect to WiFi
Agent RequestInput wifi_94a4082f8dfa_414c49454e54454b2d5959_managed_psk
  Passphrase = [ Type=psk, Requirement=mandatory, Alternates=[ WPS ] ]
  WPS = [ Type=wpspin, Requirement=alternate ]
Passphrase? [illegible]   <- Enter the password for WiFi
connmanctl> [ 2776.374921] IPv6: ADDRCONF(NETDEV_CHANGE): wlan0: link becomes ready
Connected wifi_94a4082f8dfa_414c49454e54454b2d5959_managed_psk   Connection successful
connmanctl> exit   <- Exit configuration
root@atk-imx93:~#
```

Figure 4.13-2 Connecting to a hotspot

As a result, we are connected to the WIFI hotspot, and we can use the ifconfig command to see the ip address obtained by our wlan0.

```
root@atk-imx93:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 88:c6:a3:51:dd:6e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28669<UP,BROADCAST,MULTICAST,DYNAMIC>  mtu 1500
        ether 88:95:a3:23:89:49  txqueuelen 1000  (Ethernet)
        RX packets 904  bytes 166488 (162.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 116  bytes 12925 (12.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 159  bytes 14448 (14.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 159  bytes 14448 (14.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.3.196  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::96a4:8ff:fe2f:8dfa  prefixlen 64  scopeid 0x20<link>
        ether 94:a4:08:2f:8d:fa  txqueuelen 1000  (Ethernet)
        RX packets 280  bytes 48232 (47.1 KiB)
        RX errors 0  dropped 64  overruns 0  frame 0
        TX packets 66  bytes 9453 (9.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@atk-imx93:~#
```

Figure 4.13-3 Check the obtained ip address

ping Baidu to test connectivity, or ping gateway to test WIFI connectivity. Note that if the ping is not Baidu, please do not access the network cable or other network equipment in the development board, because the system will only allow one network card to connect to the outside network by default.

```
ping www.baidu.com -I wlan0
```

```
root@atk-imx93:~# ping www.baidu.com -I wlan0
PING www.baidu.com (183.2.172.42) from 192.168.3.196 wlan0: 56(84) bytes of data.
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=1 ttl=50 time=14.3 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=2 ttl=50 time=7.05 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=3 ttl=50 time=12.4 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=4 ttl=50 time=5.90 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=5 ttl=50 time=10.9 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=6 ttl=50 time=6.23 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=7 ttl=50 time=5.54 ms
^C
--- www.baidu.com ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6007ms
rtt min/avg/max/mdev = 5.538/8.896/14.254/3.290 ms
root@atk-imx93:~#
```

Figure 4.13-4 ping Baidu test

If you have access to a cable or other network device, you can add a wifi gateway by deleting the default gateway address and then using the route command. After I access the network line here, I can not connect to the external network using wifi. The ifconfig information is as follows:

```
root@atk-imx93:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 88:01:4a:7a:87:15  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.144  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::8ada:c8ff:febf:66b8  prefixlen 64  scopeid 0x20<link>
        ether 88:da:c8:bf:66:b8  txqueuelen 1000  (Ethernet)
        RX packets 1692  bytes 237758 (232.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 113  bytes 13613 (13.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 139  bytes 13181 (12.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 139  bytes 13181 (12.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.3.196  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::96a4:8ff:fe2f:8dfa  prefixlen 64  scopeid 0x20<link>
        ether 94:a4:08:2f:8d:fa  txqueuelen 1000  (Ethernet)
        RX packets 133  bytes 26347 (25.7 KiB)
        RX errors 0  dropped 26  overruns 0  frame 0
        TX packets 52  bytes 8768 (8.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@atk-imx93:~#
root@atk-imx93:~# ping www.baidu.com -I wlan0
PING www.baidu.com (183.2.172.185) from 192.168.3.196 wlan0: 56(84) bytes of data.
From atk-imx93 (192.168.3.196) icmp_seq=1 Destination Host Unreachable
From atk-imx93 (192.168.3.196) icmp_seq=2 Destination Host Unreachable
From atk-imx93 (192.168.3.196) icmp_seq=3 Destination Host Unreachable
^C
--- www.baidu.com ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3078ms
pipe 4
```

Figure 4.13-5 Connect to Internet cable and wifi

We need to add the wifi gateway address first, and then use the route command to check if the addition was successful.

```
route                                   // View route table
route add default gw 192.168.3.1
// Add wifi gateway, fill in the address according to actual information
route add default gw 192.168.3.1
// Add wifi gateway again, fill in the address according to actual information
route
// Check the routing table again to make sure only wifi gateways are reserved
```

You can see that the default option below has been changed to use wifi.

```
root@atk-imx93:~# route   View routing table
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    0      0        0 eth1
default         _gateway        0.0.0.0         UG    10     0        0 eth1
192.168.3.0     0.0.0.0         255.255.255.0   U     0      0        0 wlan0
192.168.3.1     0.0.0.0         255.255.255.255 UH    0      0        0 wlan0
192.168.6.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
_gateway        0.0.0.0         255.255.255.255 UH    0      0        0 eth1
_gateway        0.0.0.0         255.255.255.255 UH    10     0        0 eth1
root@atk-imx93:~# route add default gw 192.168.3.1
root@atk-imx93:~# route add default gw 192.168.3.1   Add wlan gateway information
root@atk-imx93:~# route
Kernel IP routing table   Check the routing table again to make sure that the wlan0 gateway has been added
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    0      0        0 wlan0
default         _gateway        0.0.0.0         UG    0      0        0 eth1
default         _gateway        0.0.0.0         UG    10     0        0 eth1
192.168.3.0     0.0.0.0         255.255.255.0   U     0      0        0 wlan0
_gateway        0.0.0.0         255.255.255.255 UH    0      0        0 wlan0
192.168.6.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
_gateway        0.0.0.0         255.255.255.255 UH    0      0        0 eth1
_gateway        0.0.0.0         255.255.255.255 UH    10     0        0 eth1
root@atk-imx93:~# ping www.baidu.com -I wlan0
PING www.baidu.com (183.2.172.185) from 192.168.3.216 wlan0: 56(84) bytes of data.   WiFi test
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=1 ttl=50 time=39.3 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=2 ttl=50 time=9.88 ms
```

Figure 4.13-6 Modify the routing table for wlan0

4.14 Onboard Bluetooth test

ATK-DLIMX93 development board uses WIFI& Bluetooth two-in-one module, the chip is RTL8723DS, and the Bluetooth serial port is connected to the UART5 of the core board by default. There is a situation of pin sharing between Bluetooth and JTAG ON the development board. Before using Bluetooth, we need to set the SW2 dial switch in the bottom right corner of the development board to all ON (dial code up), and switch the pin to Bluetooth UART5, as shown in the following figure:

Figure 4.14-1 SW2 dial switch

In order to prevent Bluetooth from not being fully reset, we request a gpio from the kernel (this gpio hardware is connected to the Bluetooth power on enable pin), set it to the output direction, first give a low level, and then give a high level, so that Bluetooth can be reset.

Bluetooth initialization script editor's atk - bluetooth - init. Sh, in/home/root/shell/bluetooth. We can enter/home/root/shell/bluetooth directory and run the initialization scripts.

```
cd shell/bluetooth/
./atk-bluetooth-init.sh
```

```
root@atk-imx93:~/shell/bluetooth# ./atk-bluetooth-init.sh
Realtek Bluetooth :Realtek Bluetooth init uart with init speed:115200, type:HCI UART H5
Realtek Bluetooth :Realtek hciattach version 3.1.dced3af.20210423-153942

Realtek Bluetooth :Use epoll
Realtek Bluetooth :[SYNC] Get SYNC Resp Pkt
Realtek Bluetooth :[CONFIG] Get SYNC pkt
Realtek Bluetooth :[CONFIG] Get SYNC pkt
Realtek Bluetooth :[CONFIG] Get CONFG pkt
Realtek Bluetooth WARN: OP_H5_CONFIG Transmission timeout
Realtek Bluetooth :[CONFIG] Get CONFG resp pkt
Realtek Bluetooth :dic is 1, cfg field 0x14
Realtek Bluetooth :H5 init finished

Realtek Bluetooth :Realtek H5 IC
Realtek Bluetooth :Receive cmd complete event of command: 1001
Realtek Bluetooth :HCI Version 0x08
Realtek Bluetooth :HCI Revision 0x000d
Realtek Bluetooth :LMP Subversion 0x8723
Realtek Bluetooth :Receive cmd complete event of command: fc6d
Realtek Bluetooth :Read ROM version 02
Realtek Bluetooth :LMP Subversion 0x8723
Realtek Bluetooth :EVersion 2
Realtek Bluetooth :IC: RTL8723DS
Realtek Bluetooth :Firmware/config: rtl8723d_fw, rtl8723d_config
Realtek Bluetooth :Couldnt open extra config /opt/rtk_btconfig.txt, No such file or directory
Realtek Bluetooth :Couldnt access customer BT MAC file /opt/bdaddr
Realtek Bluetooth :Origin cfg len 47
Realtek Bluetooth :55 ab 23 87 29 00 f4 00 01 01 f6 00 02 81 00 fa
Realtek Bluetooth :00 02 12 80 0c 00 10 02 80 92 04 50 c5 ea 19 e1
Realtek Bluetooth :1b fd af 5f 01 a4 0b d9 00 01 0f e4 00 01 08
Realtek Bluetooth :Config baudrate: 04928002
Realtek Bluetooth :uart flow ctrl: 1
Realtek Bluetooth :Vendor baud from Config file: 04928002
Realtek Bluetooth :New cfg len 47
Realtek Bluetooth :55 ab 23 87 29 00 f4 00 01 01 f6 00 02 81 00 fa
Realtek Bluetooth :00 02 12 80 0c 00 10 02 80 92 04 50 c5 ea 19 e1
Realtek Bluetooth :1b fd af 5f 01 a4 0b d9 00 01 0f e4 00 01 08
Realtek Bluetooth :Load FW /lib/firmware/rtlbt/rtl8723d_fw OK, size 51980
Realtek Bluetooth :rtb_get_fw_project_id: opcode 0, len 1, data 9
Realtek Bluetooth :FW version 0xaa7a8cb2, Patch num 3
Realtek Bluetooth :Chip id 0x0001
Realtek Bluetooth :Chip id 0x0002
Realtek Bluetooth :Chip id 0x0003
Realtek Bluetooth :Patch length 0x8144
Realtek Bluetooth :Start offset 0x00004980
Realtek Bluetooth :Svn version:    20130
Realtek Bluetooth :Coexistence: BTCOEX_20180125-2323

Realtek Bluetooth :FW  exists, Config file  exists
Realtek Bluetooth :Total len 33139 for fwc
Realtek Bluetooth :baudrate in change speed command: 0x02 0x80 0x92 0x04
Realtek Bluetooth :Receive cmd complete event of command: fc17
Realtek Bluetooth :Received cc of vendor change baud
Realtek Bluetooth :Final speed 1500000
Realtek Bluetooth :end_idx: 131, lp_len: 127, additional pkts: 0

Realtek Bluetooth :Start downloading...
Realtek Bluetooth :Send last pkt
Realtek Bluetooth :Enable host hw flow control
Realtek Bluetooth :Received event 0xff during download patch
Realtek Bluetooth :h5_hci_reset: Issue hci reset cmd
Realtek Bluetooth :Receive cmd complete event of command: 0c03
Realtek Bluetooth :Received cc of hci reset cmd
Realtek Bluetooth :Init Process finished
Realtek Bluetooth :Realtek Bluetooth post process
Realtek Bluetooth :Device setup complete
[ 1225.225343] Bluetooth: : Non-link packet received in non-active state
[ 1225.376015] Bluetooth: MGMT ver 1.22
[ 1225.385771] NET: Registered PF_ALG protocol family
root@atk-imx93:~/shell/bluetooth#
```

Figure 4.14-2 Running Bluetooth scripts

Run the following command to see the Bluetooth device

```
hciconfig -a
```

```
root@atk-imx93:~/shell/bluetooth# hciconfig -a
hci0:   Type: Primary  Bus: UART
        BD Address: 94:A4:08:2F:8D:FB  ACL MTU: 1021:8  SCO MTU: 255:12
        UP RUNNING
        RX bytes:1216 acl:0 sco:0 events:42 errors:0
        TX bytes:2022 acl:0 sco:0 commands:41 errors:0
        Features: 0xff 0xff 0xff 0xfe 0xdb 0xff 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH HOLD SNIFF PARK
        Link mode: PERIPHERAL ACCEPT
        Name: 'atk-imx93'
        Class: 0x2c0000
        Service Classes: Rendering, Capturing, Audio
        Device Class: Miscellaneous,
        HCI Version: 4.1 (0x7)  Revision: 0xaa7a
        LMP Version: 4.1 (0x7)  Subversion: 0x8cb2
        Manufacturer: Realtek Semiconductor Corporation (93)

root@atk-imx93:~/shell/bluetooth#
```

Figure 4.14-3 View Bluetooth devices

4.14.1 Bluetooth pairing

Bluetooth pairing requires the bluetoothd service, which runs in the background as a system service and is responsible for managing the Bluetooth adapter and handling the connection, pairing, and communication of Bluetooth devices. It is one of the core components related to Bluetooth in Linux system. bluetoothd is stored in /usr/libexec/bluetoothing and is enabled by default when the system is running. If you need to enable it manually, run the following command to enable it in the background.

```
/usr/libexec/bluetooth/bluetoothd -C &
```

Enter the following command to enter the bluetoothctl interactive mode.

```
bluetoothctl
```

Execute the following commands in turn:

```
power on          # Power on
agent on          # Starts the agent
discoverable on   # discoverable on
scan on           # Enable the scanning device
devices           # Currently scanned for devices
```

```
root@atk-imx93:~/shell/bluetooth# bluetoothctl
Agent registered
[CHG] Controller 94:A4:08:2F:8D:FB Pairable: yes
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# agent on
Agent is already registered
[bluetooth]# discoverable on
Changing discoverable on succeeded
[CHG] Controller 94:A4:08:2F:8D:FB Discoverable: yes
[bluetooth]# scan on
Discovery started
[CHG] Controller 94:A4:08:2F:8D:FB Discovering: yes
[NEW] Device 5B:19:27:12:F2:7F 5B-19-27-12-F2-7F
[NEW] Device 40:44:7C:BA:DB:BB 40-44-7C-BA-DB-BB
[NEW] Device 61:30:4F:AF:67:92 61-30-4F-AF-67-92
[NEW] Device A6:D7:3C:42:4A:D3 L3250 Series
[NEW] Device 1C:52:16:AC:BB:6A QCY-APP
[NEW] Device 43:A8:C5:C4:AC:A2 43-A8-C5-C4-AC-A2
```

Figure 4.14-4 Configure and run Bluetooth

In the following picture, the author scans the Bluetooth MAC address of his Android phone, and can pair after confirmation. Different mobile phone systems may have different interception or data

protection policies for Bluetooth (such as iOS system), please ensure that mobile phone Bluetooth is not blocked.

```
[bluetooth]# devices
Device 61:ED:FE:48:4E:[redacted] 61-ED-FE-48-4E-0B
Device 44:7C:B7:56:E6:[redacted] 44-7C-B7-56-E6-F7
Device 54:4F:F6:E1:49:[redacted] 54-4F-F6-E1-49-60
Device 72:FD:39:4D:91:[redacted] 72-FD-39-4D-91-95
Device 61:34:E7:5D:02:[redacted] 61-34-E7-5D-02-BA
Device 1C:52:16:AC:BB:[redacted] QCY-APP
Device 5F:31:3F:31:78:[redacted] lock_5F313F31789D
Device 60:24:13:A7:C8:[redacted] 60-24-13-A7-C8-18
Device 4B:CA:29:A9:C1:[redacted] 4B-CA-29-A9-C1-5A
Device 4A:06:A3:79:8A:[redacted] 4A-06-A3-79-8A-20
Device 6E:FF:94:BB:ED:[redacted] 6E-FF-94-BB-ED-CB
Device 51:8A:F4:82:37:[redacted] 51-8A-F4-82-37-8B
Device 6A:D9:26:16:B2:[redacted] 6A-D9-26-16-B2-9E
Device 48:61:69:5D:59:[redacted] 48-61-69-5D-59-6E
Device A6:D7:3C:42:4A:[redacted] L3250 Series
Device 5A:73:95:F1:6B:[redacted] 5A-73-95-F1-6B-1C
Device 75:45:6B:B7:4B:[redacted] 75-45-6B-B7-4B-B8
Device 57:DF:A5:13:AC:[redacted] 57-DF-A5-13-AC-62
Device 34:21:8A:1C:79:[redacted] 34-21-8A-1C-79-09
Device C0:68:2C:61:A1:[redacted] Mi Smart Band 4
Device 50:C3:A2:07:62:[redacted] 欧尚Z6
Device 48:C4:61:DC:5C:[redacted] 正点原子蓝牙测试
Device 00:13:EF:A0:04:[redacted] DX3 Pro
```

Figure 4.14-5 Scan to Bluetooth device

Enter the following to pair.

pair 48:C4:61:DC:5C:78 # Please fill in your own Bluetooth MAC

```
[bluetooth]# pair 48:C4:61:DC:5C:[redacted]

Attempting to pair with 48:C4:61:DC:5C:[redacted]
[CHG] Device 48:C4:61:DC:5C:[redacted] Connected: yes        Enter yes
Request confirmation
[agent] Confirm passkey 503079 (yes/no): [DEL] Device 75:45:6B:B7:4B:B8 75-45-6B-B7-4B-B8
[agent] Confirm passkey 503079 (yes/no): yes

[DEL] Device 7D:B6:54:D8:EA:[redacted] 7D-B6-54-D8-EA-52          The information
[DEL] Device 46:42:0F:F9:63:[redacted] 46-42-0F-F9-63-15          output during
[DEL] Device 50:C3:A2:07:62:[redacted] 欧尚Z6                      scan on is not
[DEL] Device 90:F0:52:BC:8D:[redacted] MEIZU 20                   needed
[DEL] Device 48:95:07:54:ED:[redacted] OPPO A5
[CHG] Device 48:C4:61:DC:5C:[redacted] UUIDs: 00001855-0000-1000-8000-00805f9b34fb
[CHG] Device 48:C4:61:DC:5C:[redacted] UUIDs: 0000aa15-0000-1000-8000-00805f9b34fb
[CHG] Device 48:C4:61:DC:5C:[redacted] UUIDs: 0000afaf-0000-1000-8000-00805f9b34fb
[CHG] Device 48:C4:61:DC:5C:[redacted] UUIDs: a49eaa15-cb06-495c-9f4f-bb80a90cdf00
[CHG] Device 48:C4:61:DC:5C:[redacted] UUIDs: a49ecc15-cb06-495c-9f4f-bb80a90cdf00
[CHG] Device 48:C4:61:DC:5C:[redacted] ServicesResolved: yes
[CHG] Device 48:C4:61:DC:5C:[redacted] Paired: yes
Pairing successful                                   Successful match
```

Figure 4.14-6 Confirm the Bluetooth pairing request

At the same time, the mobile terminal also chooses to pair.

Figure 4.14-7 Bluetooth pairing request on mobile

After successful matching, we can manually connect the development board Bluetooth on the mobile phone.

Figure 4.14-8 The mobile phone is connected to the development board Bluetooth

Similarly, the development board will also prompt the Bluetooth connection and input yes to confirm the connection.

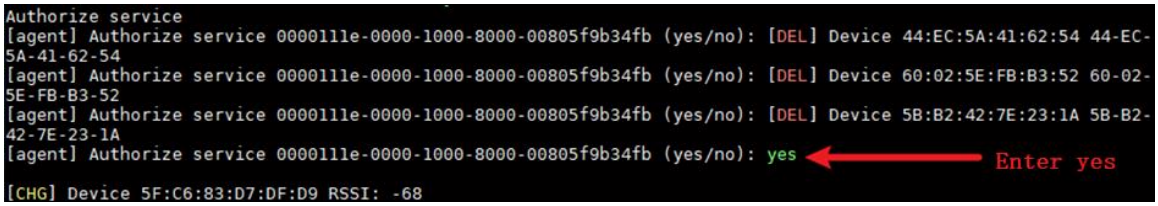


Figure 4.14-9 The development board confirms that the mobile phone Bluetooth is connected

Wait until you are connected to your Bluetooth device, type scan off to stop scanning, and then type exit to exit interactive mode.

```
scan off
exit
```

```
[正点原子蓝牙测试]# scan off

[CHG] Device 30:BB:7D:91:52:[redacted] RSSI is nil
[CHG] Device 40:46:73:77:95:[redacted] RSSI is nil
[CHG] Device 5B:B2:42:7E:23:[redacted] TxPower is nil
[CHG] Device 5B:B2:42:7E:23:[redacted] RSSI is nil
[CHG] Device 6D:98:27:81:D9:[redacted] RSSI is nil
[CHG] Device 4F:95:CE:44:12:[redacted] RSSI is nil
[CHG] Device 82:77:16:E3:54:[redacted] RSSI is nil
[CHG] Device 43:ED:95:B5:7D:[redacted] TxPower is nil
[CHG] Device 5E:8F:4E:F7:5A:[redacted] TxPower is nil
[CHG] Device 5E:8F:4E:F7:5A:[redacted] RSSI is nil
[CHG] Device 48:C4:61:DC:5C:[redacted] RSSI is nil
[CHG] Controller 94:A4:08:2F:8D:[redacted] Discovering: no
Discovery stopped
[正点原子蓝牙测试]# exit


root@atk-imx93:~/shell/bluetooth#
```

Figure 4.14-10 Exit Bluetooth Tool

Test the Bluetooth connection using the l2ping tool. This tool requires the target device to support the L2CAP echo function and enable the relevant Bluetooth permissions, at a moderate distance (usually within 10 meters) and without obvious physical obstacles and electromagnetic interference sources. The target device used by this writer is an Android phone.

```
l2ping 48:C4:61:DC:5C:78            # Please fill in your own Bluetooth MAC
```

```
root@atk-imx93:~/shell/bluetooth# l2ping 48:C4:61:DC:5C:78
Ping: 48:C4:61:DC:5C:78 from 94:A4:08:2F:8D:FB (data size 44) ...
44 bytes from 48:C4:61:DC:5C:78 id 0 time 472.92ms
44 bytes from 48:C4:61:DC:5C:78 id 1 time 24.84ms
44 bytes from 48:C4:61:DC:5C:78 id 2 time 24.87ms
44 bytes from 48:C4:61:DC:5C:78 id 3 time 23.60ms
44 bytes from 48:C4:61:DC:5C:78 id 4 time 23.63ms
44 bytes from 48:C4:61:DC:5C:78 id 5 time 27.38ms
44 bytes from 48:C4:61:DC:5C:78 id 6 time 33.68ms
44 bytes from 48:C4:61:DC:5C:78 id 7 time 476.16ms
44 bytes from 48:C4:61:DC:5C:78 id 8 time 22.45ms
^C9 sent, 9 received, 0% loss
root@atk-imx93:~/shell/bluetooth# _
```

Figure 4.14-11 Bluetooth l2ping test

4.14.2 Bluetooth file transfer

After successful initialization of Bluetooth. Use the following command to enable Bluetooth.

```
hciconfig hci0 up
hciconfig hci0 piscan      // Enable Bluetooth to be scanned
```

Use the following command to scan the Bluetooth device. The user first opens the Bluetooth of the mobile phone, and then use the following command to scan the Bluetooth device.

```
hcitool scan
```

```
root@atk-imx93:~/shell/bluetooth# hcitool scan
Scanning ...
        50:C3:A2:07:62:[redacted]        n/a
        48:C4:61:DC:5C:[redacted]        正点原子蓝牙测试
        00:13:EF:A0:04:[redacted]        DX3 Pro
        90:F0:52:BC:8D:[redacted]        n/a.Pro
        E4:40:97:4A:8B:[redacted]        一加 Ace 2 Pro
        30:BB:7D:65:33:[redacted]        OnePlus Ace Pro
        30:BB:7D:91:52:[redacted]        n/a
        48:95:07:54:ED:[redacted]        OPPO A5.
        88:68:4B:80:D8:[redacted]        一加 Ace 2V
root@atk-imx93:~/shell/bluetooth#
```

Figure 4.14-12 Scanning Bluetooth devices

Start the obexd daemon and set the receiving directory to /home/root/

```
/usr/libexec/bluetooth/obexd -r /home/root -a -d &
```

Start obexctl, the command-line testing tool

```
obexctl
```

Execute the following instructions to connect the mobile phone. We have scanned the Bluetooth of the mobile phone in the previous section. Enter the following instructions to connect the mobile phone in the format connect [MAC].

```
connect XX:XX:XX:XX:XX:XX // Please enter your Bluetooth MAC address
```

```
root@atk-imx93:~/shell/bluetooth# obexctl
[NEW] Client /org/bluez/obex
[obex]# connect 48:C4:61:DC:5C:78
Attempting to connect to 48:C4:61:DC:5C:78
[NEW] Session /org/bluez/obex/client/session0 [default]
[NEW] ObjectPush /org/bluez/obex/client/session0
Connection successful
[48:C4:61:DC:5C:78]#
```

Figure 4.14-13 Connect to Bluetooth

For example, there is a short.mp3 file in the /home/root/shell/audio directory. (Pay attention to the name or type of the transferred file, the file with the same name may be renamed by the phone, and some types of files may not be received by the phone) we send it to the phone as follows:

```
send /home/root/shell/audio/short.mp3
```

```
[48:C4:61:DC:5C:78]# send /home/root/shell/audio/short.mp3
Attempting to send /home/root/shell/audio/short.mp3 to /org/bluez/obex/client/session0
[NEW] Transfer /org/bluez/obex/client/session0/transfer0
Transfer /org/bluez/obex/client/session0/transfer0
        Status: queued
        Name: short.mp3
        Size: 3041198
        Filename: /home/root/shell/audio/short.mp3
        Session: /org/bluez/obex/client/session0
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Status: active
```

Figure 4.14-14 Perform Bluetooth file transfer

The mobile phone pop-up window shows whether to accept the file from the development board. After clicking accept, you can check the Bluetooth transmission progress on the mobile phone task bar.

Figure 4.14-15 The mobile terminal confirms the reception

The serial port terminal on the development board is printed as follows.

```
[48:C4:61:DC:5C:78]# send /home/root/shell/audio/short.mp3
Attempting to send /home/root/shell/audio/short.mp3 to /org/bluez/obex/client/session0
[NEW] Transfer /org/bluez/obex/client/session0/transfer0
Transfer /org/bluez/obex/client/session0/transfer0
        Status: queued
        Name: short.mp3
        Size: 3041198
        Filename: /home/root/shell/audio/short.mp3
        Session: /org/bluez/obex/client/session0
                                                        transmission speed
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Status: active
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Transferred: 8044 (@8KB/s 06:17)
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Transferred: 2698351 (@2690KB/s 00:00)
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Transferred: 2803378 (@105KB/s 00:02)
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Transferred: 2916484 (@113KB/s 00:01)
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Transferred: 3029590 (@113KB/s 00:00)
[CHG] Transfer /org/bluez/obex/client/session0/transfer0 Status: complete
[DEL] Transfer /org/bluez/obex/client/session0/transfer0
[48:C4:61:DC:5C:78]#
```

Figure 4.14-16 Bluetooth transmission progress

After the transfer, you can view the transferred file on your mobile phone.

Figure 4.14-17 Mobile terminal to view the received file

4.15 4G module EC20 test

The 4G module interface (MINI PCIE interface) is reserved on the base board of the ATK-DLIMX93, which supports the installation of 4G modules such as ME3630-W and EC20 sold in the ALIENTEK store.

Prepare the EC20 module, please buy it from the ALIENTEK store. Pay attention to the need to buy an antenna when purchasing, and the module alone will not work properly! (Remarks: EC20 has many types of modules, the current test is EC20-CE module, which EC20-CE series has a variety of modules, different module functions are not the same, such as the support of the operator is not the same, please consult the seller for details), other EC20 series please test by yourself, theoretically driven, there is a need to find far technical support).

Insert the EC20 module into the interface of the 4G module and screw it on. Ensure that the 4G module is connected to the seat interface. Please use the original antenna and connect the antenna to the MAIN interface of the 4G module.

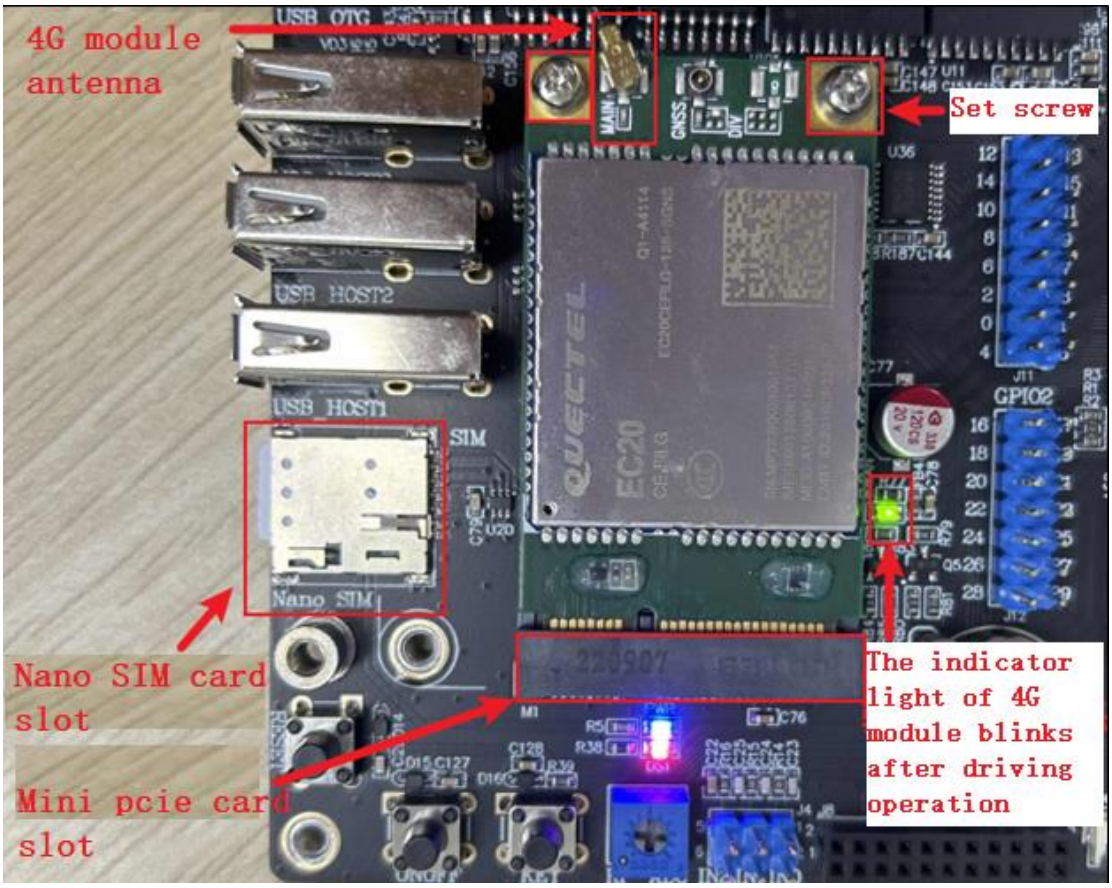


Figure 4.15-1 Development board 4G related interface

Insert the 4G card correctly (support the operator, please consult the seller of the corresponding module, note that some modules may not support the Internet of things card, please use the ordinary 4G card test) and plug in the module. After the development board is started, the WWAN LED on the bottom board will light green.

If the green light of the WWAN LED does not light up, please check whether the module is correctly connected and inserted, whether the 4G card is inserted, whether the antenna is connected, and whether the development board must be connected to the 12V power supply.

After the 4G module is loaded normally, you can see 4 ttyUSB* generated in /dev/.

```
ls /dev/ttyUSB*
```

```
root@atk-imx93:~# ls /dev/ttyUSB*
/dev/ttyUSB0  /dev/ttyUSB1  /dev/ttyUSB2  /dev/ttyUSB3
root@atk-imx93:~#
```

Figure 4.15-2 Check the interface of the 4G module

4.15.1 quectel-CM dial-up Internet access

Use the quectel-CM dialer tool (which we cross-compiled and placed in the /usr/sbin directory of the filesystem) for your convenience. This driver is provided by Quectel. It is recommended to use this dialing method.

Type quectel-CM -h to see how it works.

```
quectel-CM -h
```

```
root@atk-imx93:~#
root@atk-imx93:~# quectel-CM -h
[03-04_07:32:59:456] QConnectManager_Linux_V1.6.4
[03-04_07:32:59:457] Usage: quectel-CM [options]
[03-04_07:32:59:457] -s [apn [user password auth]]          Set apn/user/password/auth get from your network provider. auth: 1~pap, 2~chap, 3~MsChapV2
[03-04_07:32:59:457] -p pincode                             Verify sim card pin if sim card is locked
[03-04_07:32:59:457] -p [quectel-][qmi|mbim]-proxy          Request to use proxy
[03-04_07:32:59:457] -f logfilename                         Save log message of this program to file
[03-04_07:32:59:457] -u usbmonlog filename                  Save usbmon log to file
[03-04_07:32:59:457] -i interface                           Specify which network interface to setup data call when multi-modems exits
[03-04_07:32:59:457] -4                                     Setup IPv4 data call (default)
[03-04_07:32:59:457] -6                                     Setup IPv6 data call
[03-04_07:32:59:457] -n pdn                                 Specify which pdn to setup data call (default 1 for QMI, 0 for MBIM)
[03-04_07:32:59:457] -k pdn                                 Specify which pdn to hangup data call (by send SIGINT to 'quectel-CM -n pdn')
[03-04_07:32:59:457] -m iface-idx                           Bind QMI data call to wwan0_<iface idx> when QMAP used. E.g '-n 7 -m 1' bind pdn-7 data call to wwan0_1
[03-04_07:32:59:457] -b                                     Enable network interface bridge function (default 0)
[03-04_07:32:59:457] -v                                     Verbose log mode, for debug purpose.
[03-04_07:32:59:457] [Examples]
[03-04_07:32:59:457] Example 1: quectel-CM
[03-04_07:32:59:457] Example 2: quectel-CM -s 3gnet
[03-04_07:32:59:457] Example 3: quectel-CM -s 3gnet carl 1234 1 -p 1234 -f gobinet_log.txt
root@atk-imx93:~# _
```

Figure 4.15-3 See the quectel-CM tool instructions

It can be seen that the -s parameter specifies the apn type. The mobile card apn is generally cmnet, the Unicom card apn is generally 3gnet, and the telecom card is generally ctnet. APN refers to a network access technology. It is usually a parameter that must be configured when the mobile phone accesses the Internet. It determines which access method the mobile phone uses to access the network.

quectel-CM & // If it's unclear, just type quectel-CM, and & will run in the background.

```
root@atk-imx93:~# quectel-CM &
[1] 643
[03-04_07:33:31:237] QConnectManager_Linux_V1.6.4
[03-04_07:33:31:238] Find /sys/bus/usb/devices/1-1.1 idVendor=0x2c7c idProduct=0x125, bus=0x001, dev=0x003
[03-04_07:33:31:239] Auto find qmichannel = /dev/cdc-wdm0
[03-04_07:33:31:239] Auto find usbnet_adapter = wwan0
[03-04_07:33:31:239] netcard driver = qmi_wwan_q, driver version = V1.2.6
[03-04_07:33:31:239] Modem works in QMI mode
[03-04_07:33:31:247] cdc_wdm_fd = 7
root@atk-imx93:~# [03-04_07:33:31:339] Get clientWDS = 5
[03-04_07:33:31:371] Get clientDMS = 1
[03-04_07:33:31:403] Get clientNAS = 2
[03-04_07:33:31:436] Get clientUIM = 1
[03-04_07:33:31:467] Get clientWDA = 1
[03-04_07:33:31:499] requestBaseBandVersion EC20CEFILGR08A01M1G
[03-04_07:33:31:627] requestGetSIMStatus SIMStatus: SIM_READY
[03-04_07:33:31:659] requestGetProfile[1] CMNET///0/IPV4
[03-04_07:33:31:691] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE
[03-04_07:33:31:724] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[03-04_07:33:31:724] ifconfig wwan0 0.0.0.0
[03-04_07:33:31:739] ifconfig wwan0 down
[03-04_07:33:32:044] requestSetupDataCall WdsConnectionIPv4Handle: 0x86ce64e0
[03-04_07:33:32:172] ifconfig wwan0 up
[03-04_07:33:32:191] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.36.1
Dropped protocol specifier '.udhcpc' from 'wwan0.udhcpc'. Using 'wwan0' (ifindex=3).
[  893.824212] imx-dwmac 428a0000.ethernet eth1: Link is Down
[  893.831232] imx-dwmac 428a0000.ethernet eth1: FPE workqueue stop
[  893.853828] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL)
[  893.873120] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
udhcpc: broadcasting discover
udhcpc: broadcasting select for 10.21.253.126, server 10.21.253.125
udhcpc: lease of 10.21.253.126 obtained from 10.21.253.125, lease time 7200
[03-04_07:33:32:793] /etc/udhcpc.d/50default: Adding DNS 221.179.38.7   <-- DNS
[03-04_07:33:32:793] /etc/udhcpc.d/50default: Adding DNS 120.196.165.7
Dropped protocol specifier '.udhcpc' from 'wwan0.udhcpc'. Using 'wwan0' (ifindex=3).
```

Figure 4.15-4 The Internet was accessed through the quectel-CM tool

Press Enter to return to the command line and enter the ifconfig command to see the network node name of the EC20 4G module. wwan0 is the network node name of EC20 4G module.

```
root@atk-imx93:~# ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 181  bytes 16131 (15.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 181  bytes 16131 (15.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=-28669<UP,BROADCAST,MULTICAST,DYNAMIC>  mtu 1500
        inet6 fe80::96a4:8ff:fe2f:8dfa  prefixlen 64  scopeid 0x20<link>
        ether 94:a4:08:2f:8d:fa  txqueuelen 1000  (Ethernet)
        RX packets 648  bytes 127712 (124.7 KiB)
        RX errors 0  dropped 123  overruns 0  frame 0
        TX packets 50  bytes 8193 (8.0 KiB)
        TX errors 0  dropped 4 overruns 0  carrier 0  collisions 0

wwan0: flags=193<UP,RUNNING,NOARP>  mtu 1500
        inet 10.21.253.126  netmask 255.255.255.252
        inet6 fe80::a876:a6ff:fe39:68cb  prefixlen 64  scopeid 0x20<link>
        ether aa:76:a6:39:68:cb  txqueuelen 1000  (Ethernet)
        RX packets 2  bytes 612 (612.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@atk-imx93:~#
```

EC20 network node name and obtained IP

Figure 4.15-5 Retrieve the IP address

ping Baidu test connectivity, -I is the specified network card, there is data reply to indicate connectivity.

```
ping www.baidu.com -I wwan0
```

If you are prompted with "bad address 'www.baidu.com", then check that the DNS server address is set correctly, open /etc/resolv.conf, and add the DNS address obtained above, for example, "221.179.38.7" for me.

```
root@atk-imx93:~# ping www.baidu.com -I wwan0
ping: www.baidu.com: Temporary failure in name resolution
root@atk-imx93:~#
```

Figure 4.15-6 Missing DNS

```
echo nameserver 221.179.38.7 >> /etc/resolv.conf
```

```
root@atk-imx93:~# echo nameserver 221.179.38.7 >> /etc/resolv.conf
root@atk-imx93:~# ping www.baidu.com -I wwan0
PING www.baidu.com (120.232.145.185) from 10.21.253.126 wwan0: 56(84) bytes of data.
64 bytes from 120.232.145.185 (120.232.145.185): icmp_seq=1 ttl=53 time=29.0 ms
64 bytes from 120.232.145.185 (120.232.145.185): icmp_seq=2 ttl=53 time=24.2 ms
64 bytes from 120.232.145.185 (120.232.145.185): icmp_seq=3 ttl=53 time=22.8 ms
64 bytes from 120.232.145.185 (120.232.145.185): icmp_seq=4 ttl=53 time=26.0 ms
^C
--- www.baidu.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 22.806/25.509/29.034/2.323 ms
```

Figure 4.15-7 Add dns to /etc/resolv.conf and test

4.15.2 pppd dial-up Internet access

The factory file system has stored the relevant scripts for testing 4G module in the /home/root/shell/4G directory. If you used the quectel-CM dial, restart your system.

Go to /home/root/shell/4G/.

```
cd /home/root/shell/4G/
```

Use the ls command to see that there are multiple script files under this path. If there is no such file, please update the system firmware in time.

```
root@atk-imx93:~# cd /home/root/shell/4G/
root@atk-imx93:~/shell/4G# ls
disconnect  ECM_DEMO_AUTO    gosuncn_ppp_dialer_10000  gosuncn_ppp_dialer_10086  ppp-on-10010
ECM_DEMO    gosuncn_options  gosuncn_ppp_dialer_10010  ppp-on-10000              ppp-on-10086
root@atk-imx93:~/shell/4G#
```

Figure 4.15-8 View script

The suffix 10000 indicates that it is suitable for telecom cards, 10010 indicates that it is suitable for Unicom cards, and 10086 indicates that it is suitable for mobile cards.

Take mobile cards as an example here. Execute the following ppp-on script to configure the 4G module. & is used to run in the background.

```
./ppp-on-10086 &
```

```
root@atk-imx93:~/shell/4G# ./ppp-on-10086 &
[1] 620
root@atk-imx93:~/shell/4G#
pppd options in effect:
debug           # (from gosuncn_options)
nodetach                # (from gosuncn_options)
persist         # (from gosuncn_options)
dump            # (from gosuncn_options)
noauth          # (from gosuncn_options)
user Anyname            # (from gosuncn_options)
password ??????         # (from gosuncn_options)
/dev/ttyUSB2            # (from gosuncn_options)
115200          # (from gosuncn_options)
lock            # (from gosuncn_options)
connect chat -v -f /home/root/shell/4G/gosuncn_ppp_dialer_10086         # (from command line)
crtscts         # (from gosuncn_options)
modem           # (from gosuncn_options)
novj            # (from gosuncn_options)
ipcp-accept-local               # (from gosuncn_options)
ipcp-accept-remote              # (from gosuncn_options)
noipdefault             # (from gosuncn_options)
defaultroute            # (from gosuncn_options)
usepeerdns              # (from gosuncn_options)
noccp           # (from gosuncn_options)
nobsdcomp               # (from gosuncn_options)
ATE
OK
ATH
OK
ATP
OK
AT+CGDCONT=1,"IP","CMNET"
OK
ATD*99#
CONNECT
Script chat -v -f /home/root/shell/4G/gosuncn_ppp_dialer_10086 finished (pid 624), status = 0x0
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB2
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0xbbe118e> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x516130ea> <pcomp> <accomp>]
sent [LCP ConfAck id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x516130ea> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0xbbe118e> <pcomp> <accomp>]
rcvd [LCP DiscReq id=0x1 magic=0x516130ea]
rcvd [CHAP Challenge id=0x1 <7ad44ffd73a1d9ffe6ee0d63b1864a38>, name = "UMTS_CHAP_SRVR"]
sent [CHAP Response id=0x1 <f96483929b1958a6522f4ff1ee5cf36a>, name = "Anyname"]
rcvd [CHAP Success id=0x1 ""]
CHAP authentication succeeded
CHAP authentication succeeded
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
sent [IPV6CP ConfReq id=0x1 <addr fe80::9d79:7e5b:319a:1c15>]
rcvd [IPCP ConfReq id=0x0]
sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]
rcvd [IPCP ConfNak id=0x1 <addr 10.22.55.186> <ms-dns1 221.179.38.7> <ms-dns2 120.196.165.7>]
sent [IPCP ConfReq id=0x2 <addr 10.22.55.186> <ms-dns1 221.179.38.7> <ms-dns2 120.196.165.7>]
rcvd [IPCP ConfReq id=0x1]
sent [IPCP ConfAck id=0x1]
rcvd [IPCP ConfAck id=0x2 <addr 10.22.55.186> <ms-dns1 221.179.38.7> <ms-dns2 120.196.165.7>]
Could not determine remote IP address: defaulting to 10.64.64.64
Failed to create /etc/ppp/resolv.conf: No such file or directory
not replacing default route to eth1 [192.168.6.1]
local  IP address 10.22.55.186
remote IP address 10.64.64.64
primary   DNS address 221.179.38.7
secondary DNS address 120.196.165.7
sent [IPV6CP ConfReq id=0x1 <addr fe80::9d79:7e5b:319a:1c15>]

root@atk-imx93:~/shell/4G# sent [IPV6CP ConfReq id=0x1 <addr fe80::9d79:7e5b:319a:1c15>]
```

Get the IP and DNS

Figure 4.15-9 ppp dial-up Internet access to obtain IP and DNS

Press Enter to return to the command line, and use ifconfig to view the dial-up node name of the 4G module.

```
root@atk-imx93:~/shell/4G# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 88:c6:a3:51:dd:6e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.196  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::8a95:a3ff:fe23:8949  prefixlen 64  scopeid 0x20<link>
        ether 88:95:a3:23:89:49  txqueuelen 1000  (Ethernet)
        RX packets 2065  bytes 252641 (246.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 125  bytes 13616 (13.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 132  bytes 11596 (11.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 132  bytes 11596 (11.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ppp0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.22.55.186  netmask 255.255.255.255  destination 10.64.64.64
        ppp  txqueuelen 3  (Point-to-Point Protocol)
        RX packets 5  bytes 92 (92.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14  bytes 198 (198.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

ppp0 is the name of the dial-up node of the 4G module, and the IP is also obtained

Figure 4.15-10 Check the 4G network node name

ping Baidu test connectivity, -I is the specified network card, there is data reply to indicate connectivity. Press CTRL-C to terminate the command.

```
ping www.baidu.com -I ppp0
```

If you are prompted with "bad address 'www.baidu.com", then check that the DNS server address is set correctly, open /etc/resolv.conf, and add the DNS address obtained above, for example, "221.179.38.7" for me.

```
root@atk-imx93:~/shell/4G# ping www.baidu.com -I ppp0
ping: www.baidu.com: Temporary failure in name resolution
root@atk-imx93:~/shell/4G#
```

Figure 4.15-11 Missing DNS

```
root@atk-imx93:~/shell/4G# echo nameserver 221.179.38.7 >> /etc/resolv.conf
root@atk-imx93:~/shell/4G# cat /etc/resolv.conf
# Generated by Connection Manager
nameserver ::1
nameserver 127.0.0.1
nameserver 221.179.38.7
root@atk-imx93:~/shell/4G#
```

Figure 4.15-12 Add DNS to /etc/resolv.conf

```
root@atk-imx93:~/shell/4G# ping www.baidu.com -I ppp0
PING www.baidu.com (183.2.172.42) from 10.22.55.186 ppp0: 56(84) bytes of data.
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=1 ttl=47 time=148 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=2 ttl=47 time=31.3 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=3 ttl=47 time=62.8 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=4 ttl=47 time=29.0 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=5 ttl=47 time=26.5 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=6 ttl=47 time=30.1 ms
^C
--- www.baidu.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5006ms
rtt min/avg/max/mdev = 26.501/54.663/148.323/43.671 ms
```

Figure 4.15-13 ping Baidu test 4G network connection

4.15.3 GPS function test

EC20 also has a GPS function, through the AT command to set to open the GPS, this section only teaches how to open and view the original GPS data, For other AT instructions, please see the development board CD A - Basic Information \4_reference_data \ Quectel_EC2x&EG2x&EG9x&EM05 series _GNSS_ Application Guidelin.pdf.

Note: There is a GPS antenna interface in the picture. You can buy EC20 with GPS antenna version. Passive antenna needs to be connected to GPS interface.

The 4G module is normally initialized with the following printing information:

```
[    4.629569] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB0
         Starting Flush Journal to Persistent Storage...
[    4.652028] systemd-journald[173]: Received client request to flush runtime journal.
[    4.660215] idVendor=2c7c, idProduct=125, bInterfaceNumber =1
         Startin[    4.669736] option 1-1.1:1.1: GSM modem (1-port) converter detected
g Apply Kernel Variables...
[    4.681520] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB1
[    4.693471] idVendor=2c7c, idProduct=125, bInterfaceNumber =2
[    4.699568] option 1-1.1:1.2: GSM modem (1-port) converter detected
[    4.721206] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB2
         Starting Create Static Device Nodes in /dev...
[    4.749627] idVendor=2c7c, idProduct=125, bInterfaceNumber =3
[  OK      4.758763] option 1-1.1:1.3: GSM modem (1-port) converter detected
m] Mounted FUSE Control [    4.767989] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB3
File System.
[  OK  ] Mounted Kernel Configuration File System.
[    4.787103] qmi_wwan_q 1-1.1:1.4: cdc-wdm0: USB WDM device
[    4.795133] qmi_wwan_q 1-1.1:1.4: Quectel EC20-CE-FILG work on RawIP mode
[    4.808124] qmi_wwan_q 1-1.1:1.4: rx_urb_size = 1520
[  OK  ] Finished Flush Journal to Persistent[    4.818012] qmi_wwan_q 1-1.1:1.4 wwan0: register 'qmi_wwan_q' at
 usb-ci_hdrc.1-1.1, WWAN/QMI Raw IP device, 7e:55:5f:76:08:cd
 Storage.
```

Figure 4.15-14 Module driver information

Run the following command to use the GPS function, and send the AT+QGPS=1 data to USB2 through the echo command.

```
echo -e "AT+QGPS=1\r\n" >/dev/ttyUSB2
cat /dev/ttyUSB1
```

Waiting for the star search to end, you can see the original GPS data, refer to the following:

```
root@atk-imx93:~# cat /dev/ttyUSB1
$GPGSV,2,1,06,08,29,037,42,14,22,313,40,17,25,260,25,21,57,054,37,1*66

$GPGSV,2,2,06,30,43,306,42,16,03,102,,1*62

$GPGGA,034619.00,2318.098189,N,11319.723998,E,1,04,2.4,5.6,M,-5.0,M,,*70

$GPVTG,0.0,T,2.4,M,0.0,N,0.0,K,A*25

$GPRMC,034619.00,A,2318.098189,N,11319.723998,E,0.0,0.0,030424,2.4,W,A,V*5F

$GPGSA,A,3,08,14,21,30,,,,,,,,,2.6,2.4,0.9,1*29
```

Figure 4.15-15 GPS raw data

Press "Ctrl + c" to terminate the cat command and turn off the GPS function to run the following code.

```
echo -e "AT+QGPSEND\r\n" >/dev/ttyUSB2
```

Wait a few seconds for the GPS function to turn off, at which point we can see that the ttyUSB1 serial port no longer prints the raw GPS data.

```
cat /dev/ttyUSB1
```

4.16 4G module ME3630-W test

ATK-DLIMX93 development board reserves a 4G module interface (MINI PCIE interface) to support the installation of ME3630-W, EC20 and other 4G modules. Prepare the ME3630-W 4G module, insert the ME3630-W 4G module into the interface of the 4G module, and screw on the screws. Ensure that the 4G module is connected to the seat interface. Please use the original antenna and connect the antenna to the MAIN interface of the 4G module. (Warm tip: after the 4G module is unsealed, the gold finger is easy to oxidiz.After a long time, there will be a layer of oxide film on it. If many experiments are not successful, please check whether the 4G module is inserted or whether there is an oxide film. Please save the 4G module after use!)

Insert the 4G card correctly (support mobile, China Unicong, Telecom 4G card) and insert the module, and the WWAN LED on the bottom board will light green after the development board is started. If the green light of the WWAN LED does not light up, please check whether the module is properly connected and inserted, and whether the 4G card is inserted

In, whether the antenna is connected properly, the development board should be connected to the 12V power supply.

After the 4G module is loaded normally, you can see 3 ttyUSB* generated in /dev/. The 4G module communicates with /dev/ttyUSB2 via the AT command.

```
ls /dev/ttyUSB*
```

```
root@atk-imx93:~# ls /dev/ttyUSB*
/dev/ttyUSB0  /dev/ttyUSB1  /dev/ttyUSB2
root@atk-imx93:~#
```

Figure 4.16-1 Look at the 4G module node

4.16.1 ECM goes online

In the view of the operating system, the CDC ECM device is a virtual Ethernet card that contains the MAC and IP addresses required by standard network cards.

ECM_DEMO_AUTO and ECM_DEMO are high emerging ECM online programs (we do not need to know the source code, because it is just a tool), detailed parameters can be executed ecm_demo-h to see.

ECM_DEMO_AUTO and ECM_DEMO parameters are the same, the difference is that ECM_DEMO executes the process will exit, ECM_DEMO_AUTO will run and every once in a while will check whether the network is disconnected, the network will automatically reconnect. ECM_DEMO_AUTO executes the dialing process automatically by default, so it is suitable for booting programs.

To configure the ECM connection, if pppd is running, run the disconnect script to disconnect the pppd dial-up connection, and then run the following command to configure the connection to ECM mode.

```
cd shell/4G
```

./disconnect

./ECM_DEMO -t up

```
root@atk-imx93:~/shell/4G# ./ECM_DEMO -t up
[  151.105172] IPv6: ADDRCONF(NETDEV_CHANGE): usb0: link becomes ready
[03_04_08:32:23] ECM_DEMO success
root@atk-imx93:~/shell/4G#
```

Figure 4.16-2 First disconnect the ppp-on network, and then execute the ecm-on network

Use ifconfig command to check the ip address obtained by usb0, if it is not obtained, use udhcpc -i usb0 to obtain the ip address.

```
root@atk-imx93:~/shell/4G# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 88:c6:a3:51:dd:6e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.211  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::28a4:50ff:fea5:3aef  prefixlen 64  scopeid 0x20<link>
        ether 2a:a4:50:a5:3a:ef  txqueuelen 1000  (Ethernet)
        RX packets 1258  bytes 176724 (172.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 126  bytes 14083 (13.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 143  bytes 13553 (13.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 143  bytes 13553 (13.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

usb0: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 10.1.188.40  netmask 255.255.255.240  broadcast 10.1.188.47
        inet6 fe80::6c74:bfff:febd:63b7  prefixlen 64  scopeid 0x20<link>
        ether 6e:74:bf:bd:63:b7  txqueuelen 1000  (Ethernet)
        RX packets 27  bytes 2834 (2.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 42  bytes 7810 (7.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4.16-3 Checking ip addresses

ping Baidu test connectivity, -I is the specified network card, there is data reply to indicate connectivity. Press CTRL-C to terminate the command.

ping www.baidu.com -I usb0 // The -I argument specifies the card name usb0 (4G network).

```
root@atk-imx93:~/shell/4G# ping www.baidu.com -I usb0
PING www.baidu.com (183.2.172.42) from 10.1.188.40 usb0: 56(84) bytes of data.
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=1 ttl=46 time=133 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=2 ttl=46 time=31.3 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=3 ttl=46 time=55.0 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=4 ttl=46 time=33.3 ms
64 bytes from 183.2.172.42 (183.2.172.42): icmp_seq=5 ttl=46 time=32.3 ms
^C
--- www.baidu.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 31.285/56.998/133.068/39.043 ms
root@atk-imx93:~/shell/4G#
```

Figure 4.16-4 ping Baidu test Internet access

If you need to check if 4G is online (disconnect and reconnect), it is recommended to use the./ECM_DEMO_AUTO & command and run it in the background. If you do not get the ip address after connecting, use udhcpc -i usb0 to get it.

```
root@atk-imx93:~/shell/4G# ./ECM_DEMO_AUTO &
[1] 593
[03_04_08:36:15] ECM_DEMO_AUTO VERSION:ECM_CALLV1.0.1B05 start...
root@atk-imx93:~/shell/4G# [03_04_08:36:19] ECM_DEMO_AUTO GET IMEI=[864863043372842]
[03_04_08:36:19] ECM_DEMO_AUTO GET ICCID=[89860001192007171680]
[03_04_08:36:19] ECM_DEMO_AUTO GET SIM_STATUS=[SIM_READY]
[   29.104999] IPv6: ADDRCONF(NETDEV_CHANGE): usb0: link becomes ready
[03_04_08:36:25] ECM_DEMO_AUTO Network Connected,NetType[LTE],SignalStrength[86]
```

Figure 4.16-5 ECM_DEMO_AUTO networking

4.16.2 pppd dial-up Internet access

The factory file system has stored the relevant scripts for testing 4G module in the /home/root/shell/4G directory. Reboot the system if ECM dialing was used earlier.

Go to /home/root/shell/4G/.

```
cd /home/root/shell/4G/
```

Use the ls command to see that there are multiple script files under this path. If there is no such file, please update the system firmware in time.

```
root@atk-imx93:~# cd /home/root/shell/4G/
root@atk-imx93:~/shell/4G# ls
disconnect  ECM_DEMO_AUTO    gosuncn_ppp_dialer_10000  gosuncn_ppp_dialer_10086  ppp-on-10010
ECM_DEMO    gosuncn_options  gosuncn_ppp_dialer_10010  ppp-on-10000              ppp-on-10086
root@atk-imx93:~/shell/4G#
```

Figure 4.16-6 View script

The suffix 10000 indicates that it is suitable for telecom cards, 10010 indicates that it is suitable for Unicom cards, and 10086 indicates that it is suitable for mobile cards.

Take mobile cards as an example here. Execute the following ppp-on script to configure the 4G module. & is used to run in the background.

```
./ppp-on-10086 &
```

```
root@atk-imx93:~/shell/4G#
pppd options in effect:
debug           # (from gosuncn_options)
nodetach                # (from gosuncn_options)
persist         # (from gosuncn_options)
dump            # (from gosuncn_options)
noauth          # (from gosuncn_options)
user Anyname            # (from gosuncn_options)
password ??????         # (from gosuncn_options)
/dev/ttyUSB2            # (from gosuncn_options)
115200          # (from gosuncn_options)
lock            # (from gosuncn_options)
connect chat -v -f /home/root/shell/4G/gosuncn_ppp_dialer_10086         # (from command line)
crtscts         # (from gosuncn_options)
modem           # (from gosuncn_options)
novj            # (from gosuncn_options)
ipcp-accept-local               # (from gosuncn_options)
ipcp-accept-remote              # (from gosuncn_options)
noipdefault             # (from gosuncn_options)
defaultroute            # (from gosuncn_options)
usepeerdns              # (from gosuncn_options)
noccp           # (from gosuncn_options)
nobsdcomp               # (from gosuncn_options)
ATE
OK
ATH
OK
ATP
OK
AT+CGDCONT=1,"IP","CMNET"
OK
ATD*99#
CONNECT
Script chat -v -f /home/root/shell/4G/gosuncn_ppp_dialer_10086 finished (pid 614), status = 0x0
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB2
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x4e07c239> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x74938c00> <pcomp> <accomp>]
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x74938c00> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x4e07c239> <pcomp> <accomp>]
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
sent [IPV6CP ConfReq id=0x1 <addr fe80::6c54:3cf1:54be:5ca2>]
rcvd [IPCP ConfReq id=0x1 <addr 10.1.188.41>]
sent [IPCP ConfAck id=0x1 <addr 10.1.188.41>]
rcvd [IPV6CP ConfReq id=0x1 <addr fe80::2508:7e87:f346:347e>]
sent [IPV6CP ConfAck id=0x1 <addr fe80::2508:7e87:f346:347e>]
rcvd [IPCP ConfNak id=0x1 <addr 10.1.188.40> <ms-dns1 221.179.38.7> <ms-dns2 221.179.38.7>]
sent [IPCP ConfReq id=0x2 <addr 10.1.188.40> <ms-dns1 221.179.38.7> <ms-dns2 221.179.38.7>]
rcvd [IPV6CP ConfAck id=0x1 <addr fe80::6c54:3cf1:54be:5ca2>]
local  LL address fe80::6c54:3cf1:54be:5ca2
remote LL address fe80::2508:7e87:f346:347e
rcvd [IPCP ConfAck id=0x2 <addr 10.1.188.40> <ms-dns1 221.179.38.7> <ms-dns2 221.179.38.7>]
Failed to create /etc/ppp/resolv.conf: No such file or directory
not replacing default route to eth1 [192.168.6.1]
local  IP address 10.1.188.40
remote IP address 10.1.188.41
primary   DNS address 221.179.38.7
secondary DNS address 221.179.38.7

root@atk-imx93:~/shell/4G#
```

Retrieve the IP and DNS

Figure 4.16-7 ppp dial-up Internet access to obtain IP and DNS

Press Enter to return to the command line, and use ifconfig to view the dial-up node name of the 4G module.

```
root@atk-imx93:~/shell/4G# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 88:c6:a3:51:dd:6e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.211  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::8a95:a3ff:fe23:8949  prefixlen 64  scopeid 0x20<link>
        ether 88:95:a3:23:89:49  txqueuelen 1000  (Ethernet)
        RX packets 1604  bytes 227399 (222.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 129  bytes 15253 (14.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 163  bytes 15413 (15.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 163  bytes 15413 (15.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ppp0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.1.188.40  netmask 255.255.255.255  destination 10.1.188.41
        inet6 fe80::6c54:3cf1:54be:5ca2  prefixlen 128  scopeid 0x20<link>
        ppp  txqueuelen 3  (Point-to-Point Protocol)
        RX packets 8  bytes 226 (226.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 322 (322.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

ppp0 is the name of the dial-up node of the 4G network module, and the IP is also obtained

Figure 4.16-8 Check the 4G network node name

ping Baidu test connectivity, -I is the specified network card, there is data reply to indicate connectivity. Press CTRL-C to terminate the command.

```
ping www.baidu.com -I ppp0
```

If you are prompted with "bad address 'www.baidu.com", then check that the DNS server address is set correctly, open /etc/resolv.conf, and add the DNS address obtained above, for example, 221.179.38.7 in my case.

```
echo nameserver 221.179.38.7 >> /etc/resolv.conf
```

```
root@atk-imx93:~/shell/4G# ping www.baidu.com -I ppp0
ping: www.baidu.com: Temporary failure in name resolution
root@atk-imx93:~/shell/4G#
```

Figure 4.16-9 Missing DNS

```
root@atk-imx93:~/shell/4G# echo nameserver 221.179.38.7 >> /etc/resolv.conf
root@atk-imx93:~/shell/4G# cat /etc/resolv.conf
# Generated by Connection Manager
nameserver ::1
nameserver 127.0.0.1
nameserver 221.179.38.7
root@atk-imx93:~/shell/4G#
```

Figure 4.16-10 Add DNS to /etc/resolv.conf

```
root@atk-imx93:~/shell/4G# ping www.baidu.com -I ppp0
PING www.baidu.com (120.232.145.144) from 10.227.81.11 ppp0: 56(84) bytes of data.
64 bytes from 120.232.145.144 (120.232.145.144): icmp_seq=1 ttl=52 time=38.1 ms
64 bytes from 120.232.145.144 (120.232.145.144): icmp_seq=2 ttl=52 time=49.6 ms
64 bytes from 120.232.145.144 (120.232.145.144): icmp_seq=3 ttl=52 time=29.4 ms
64 bytes from 120.232.145.144 (120.232.145.144): icmp_seq=4 ttl=52 time=29.6 ms
^C64 bytes from 120.232.145.144: icmp_seq=5 ttl=52 time=26.5 ms

--- www.baidu.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 32362ms
rtt min/avg/max/mdev = 26.549/34.670/49.610/8.410 ms
root@atk-imx93:~/shell/4G#
```

Figure 4.16-11 ping Baidu test 4G network connection

4.16.3 GPS function test

This section only teaches how to open and view the original GPS data. For other AT instructions, please refer to the ME3630 module information "ME3630&ME3630-W Module AT Instruction Manual.pdf".

Note: The ME3630-C3C with GPS antenna version is required. The GPS antenna uses a passive antenna, which needs to be connected to the GPS interface of the module. The antenna must be placed outside. Note that the ME3630-C3B version is not GPS capable and cannot be located using an active antenna.

The 4G module is normally initialized with the following printing information:

```
[    3.015834] usb 1-1.1: new high-speed USB device number 3 using ci_hdrc
[    3.149524] idVendor=19d2, idProduct=1476, bInterfaceNumber =0
[    3.155524] option 1-1.1:1.0: GSM modem (1-port) converter detected
[    3.162109] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB0
[    3.169420] idVendor=19d2, idProduct=1476, bInterfaceNumber =1
[    3.175435] option 1-1.1:1.1: GSM modem (1-port) converter detected
[    3.182024] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB1
[    3.189357] idVendor=19d2, idProduct=1476, bInterfaceNumber =2
[    3.195359] option 1-1.1:1.2: GSM modem (1-port) converter detected
[    3.201944] usb 1-1.1: GSM modem (1-port) converter now attached to ttyUSB2
[    3.210576] idVendor=19d2, idProduct=1476, bInterfaceNumber =3
[    3.216802] idVendor=19d2, idProduct=1476, bInterfaceNumber =4
```

Figure 4.16-12 Module driver information

Run the following command to use the GPS function, and send the AT+QGPS=1 data to USB2 through the echo command.

```
echo -e "AT+ZGINIT\r\n" >/dev/ttyUSB1          // Initialize GPS
echo -e "AT+ZGMODE=3\r\n" >/dev/ttyUSB1        // Set the positioning mode
echo -e "AT+ZGPORT=0\r\n" >/dev/ttyUSB1
// Location information is reported from AT, MODEM and UART
echo -e "AT+ZGNMEA=31\r\n" >/dev/ttyUSB1       // Set GPS data to NMEA format
echo -e "AT+ZGPSR=1\r\n" >/dev/ttyUSB1         // Enable ZGPSR data
echo -e "AT+ZGRUN=2\r\n" >/dev/ttyUSB1         // Continuous positioning mode
```

After the configuration is completed, wait for the star search to end, you can see the original GPS data, refer to the following:

```
cat /dev/ttyUSB1
```

```
root@atk-imx93:~# cat /dev/ttyUSB1
$GLGSV,2,1,05,82,40,015,30,73,28,334,34,79,30,129,,83,73,293,*67

$GLGSV,2,2,05,81,,,*69

$GPGSV,4,1,16,08,20,049,34,14,32,323,40,17,,,33,21,45,040,38*42

$GPGSV,4,2,16,22,15,313,40,30,52,286,33,01,72,070,,03,20,135,*74

$GPGSV,4,3,16,07,58,199,,09,,,,10,,,,13,,,*48

$GPGSV,4,4,16,19,03,251,,26,01,118,,27,,,,28,,,*71

$BDGSV,2,1,05,07,,,39,10,,,39,26,,,44,16,,,,0,4*6F

$BDGSV,2,2,05,19,00,132,,0,4*51

$GPGGA,041909.00,2318.124131,N,11319.718160,E,1,05,2.4,40.5,M,-5.0,M,,*4D

$GPVTG,0.0,T,2.4,M,0.0,N,0.0,K,A*25

$GPRMC,041909.00,A,2318.124131,N,11319.718160,E,0.0,0.0,030424,2.4,W,A*2B

$GPGSA,A,2,08,14,21,22,30,,,,,,,,2.6,2.4,1.0*3D
```

Figure 4.16-13 GPS raw data

Press "Ctrl + c" to terminate the cat command and turn off the GPS function to run the following code.

```
echo -e "AT+ZGRUN=0\r\n" >/dev/ttyUSB1          // Initializing GPS
```

Wait a few seconds for the GPS function to turn off, at which point we can see that the ttyUSB1 serial port no longer prints the raw GPS data.

```
cat /dev/ttyUSB1
```

4.17 TF (SD) card test

Instruction Prompt:

The time command is often used to measure the running time of a command, dd is used to copy, read from the if(input file) file and write to the of(output file) specified file, bs is the block size of each write, and count is the number of read and write blocks. if=/dev/zero" does not produce IO, that is, it can continuously output data, so it can be used to test pure write speed.

The author starts from eMMC and can see that /dev/mmcblk1 is the device node of the TF card, and /dev/mmcblk1p1 is the default partition of the TF card. Note that the FAT32 format TF card is required, not NTFS.

```
root@atk-imx93:~#
root@atk-imx93:~#                    System information when accessing TF card
root@atk-imx93:~#
root@atk-imx93:~# [  169.740984] mmc1: host does not support reading read-only switch, assuming write-enabl
[  169.773754] mmc1: new ultra high speed SDR104 SDHC card at address 5048
[  169.781190] mmcblk1: mmc1:5048 SD32G 29.7 GiB    TF card capacity
[  169.792937]  mmcblk1: p1

root@atk-imx93:~# ls /dev/mmcblk*    View TF card device nodes and partitions
/dev/mmcblk0       /dev/mmcblk0boot1  /dev/mmcblk0p2    /dev/mmcblk1
/dev/mmcblk0boot0  /dev/mmcblk0p1     /dev/mmcblk0rpmb   /dev/mmcblk1p1
root@atk-imx93:~#
```

Figure 4.17-1 View TF card device nodes and partitions

Use the df -h command to view the TF card mount path, the following figure shows that the TF card mount directory is /run/media/mmcblk1p1.

```
df -h
```

```
root@atk-imx93:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       6.6G  3.3G  3.0G  52% /
devtmpfs        341M  4.0K  341M   1% /dev
tmpfs           471M     0  471M   0% /dev/shm
tmpfs           189M  9.0M  180M   5% /run
tmpfs           4.0M     0  4.0M   0% /sys/fs/cgroup
tmpfs           471M  8.0K  471M   1% /tmp
tmpfs           471M  168K  471M   1% /var/volatile
tmpfs            95M     0   95M   0% /run/user/0
/dev/mmcblk0p1  500M   33M  468M   7% /run/media/mmcblk0p1 View the TF card mount path
/dev/mmcblk1p1   30G  128K   30G   1% /run/media/mmcblk1p1
root@atk-imx93:~#
```

Figure 4.17-2 View the TF card mount path

4.17.1 Reading speed test

The instructions hdparm and dd are commonly used to test hard disk read speed; only hdparm read TF card speed is demonstrated here. (Read and write speed depends on TF card type used)

```
hdparm -t /dev/mmcblk1
```

```
root@atk-imx93:~# hdparm -t /dev/mmcblk1

/dev/mmcblk1:
 Timing buffered disk reads: 260 MB in  3.02 seconds =  86.07 MB/sec
root@atk-imx93:~#
```

Figure 4.17-3 Reading speed test

4.17.2 Write Speed testing

Write 100MB of data to the mount directory of the TF card using the dd instruction, that is, /run/media/mmcblk1p1. The larger the write, the closer it is to the actual value.

```
time dd if=/dev/zero of=/run/media/mmcblk1p1/test bs=1024k count=100 conv=fdatasync
```

```
root@atk-imx93:~# time dd if=/dev/zero of=/run/media/mmcblk1p1/test bs=1024k count=100 conv=fdatasync
100+0 records in
100+0 records out
104857600 bytes (105 MB, 100 MiB) copied, 5.35894 s, 19.6 MB/s

real    0m5.366s
user    0m0.001s
sys     0m0.489s
root@atk-imx93:~# ls /run/media/mmcblk1p1/
'System Volume Information'   test
root@atk-imx93:~#
```

Figure 4.17-4 Write speed test

After the test is completed, the test file "test" under the mount directory /run/media/mmcblk1p1 can be deleted to complete the write speed test of the TF card.

4.18 OV5645 camera test

Before the experiment, please prepare ov5645 camera module, our MIPI LCD capacitor screen of any resolution.

Camera insertion method: the camera lens is directly inserted into the MIPI CSI socket to the outside of the development board. When inserting the camera, it is necessary to pay attention to the screen printing of the bottom plate, and insert it according to the pin number, that is, the lens is toward the outside of the board. As shown below:

Figure 4.18-1 Physical connection of camera

Before starting, plug in the camera and LCD screen (neither support hotplug), and then start the development board after connecting the device.

```
root@atk-imx93:~# dmesg | grep ov5645
[    5.482537] ov5645 0-003c: OV5645 Chip id = 5645
[   13.022647] mx8-img-md: Registered sensor subdevice: ov5645 0-003c (1)
[   13.056097] mx8-img-md: created link [ov5645 0-003c] => [mxc-mipi-csi2.0]
root@atk-imx93:~#
```

Figure 4.18-2 Information printed by the ov5645 driver at startup

Look at the node generated by ov5645 and confirm that the node is video0.

```
ls /dev/video0
```

```
root@atk-imx93:~# ls /dev/video0
/dev/video0
root@atk-imx93:~#
```

Figure 4.18-3 ov5645 nodes

Open the "Camera" application of the QT interface to open the OV5645 camera for image acquisition. This is shown in the figure below.

Figure 4.18-4 Qt interface camera function

Qt interface camera function uses 720x480 pixels by default, and the display interface is made into a full horizontal screen display. This program is only for test reference.

Figure 4.18-5 Camera interface Preview

If you don't have a screen, you can use the take a picture command to take a picture, and then upload the picture to the computer to open the view. The instructions for taking photos are as follows (the following is a full line of instructions, please copy and combine!). :

gst-launch-1.0 v4l2src device=/dev/video0 io-mode=4 num-buffers=1 ! video/x-raw,format=YUY2,width=1920,height=1080,framerate=30/1 ! jpegenc ! filesink location=pic.jpeg

```
root@atk-imx93:~#
root@atk-imx93:~# gst-launch-1.0 v4l2src device=/dev/video0 io-mode=4 num-buffers=1 ! video/x-raw,format=YUY2
,width=1920,height=1080,framerate=30/1 ! jpegenc ! filesink location=pic.jpeg
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
[  207.658200] mxc-mipi-csi2.0: format: 0x2008
[  207.667424] bypass csc
[  207.669811] input fmt YUV4
[  207.672511] output fmt YUYV
[  208.269194] dwc-mipi-csi2-host 4ae00000.csi: enter enable=1
Redistribute latency...
Got EOS from element "pipeline0".
Execution ended after 0:00:04.332159461
Setting pipeline to NULL ...
[  209.037188] dwc-mipi-csi2-host 4ae00000.csi: enter enable=0
Freeing pipeline ...
root@atk-imx93:~#
root@atk-imx93:~# ls
pic.jpeg  shell
root@atk-imx93:~#
```

Figure 4.18-6 Camera photo command

Through SSH or in the form of file copy will be taken pictures to the computer to open the access.

4.19 ADC test

Note: The maximum absolute value of ADC acquisition voltage is 1.8V, please do not exceed 1.8V, otherwise it may cause damage to the chip.

There is one ADC at J4 on the ATK-DLIMX93 base plate for user testing. The schematic diagram of the base plate is as follows.

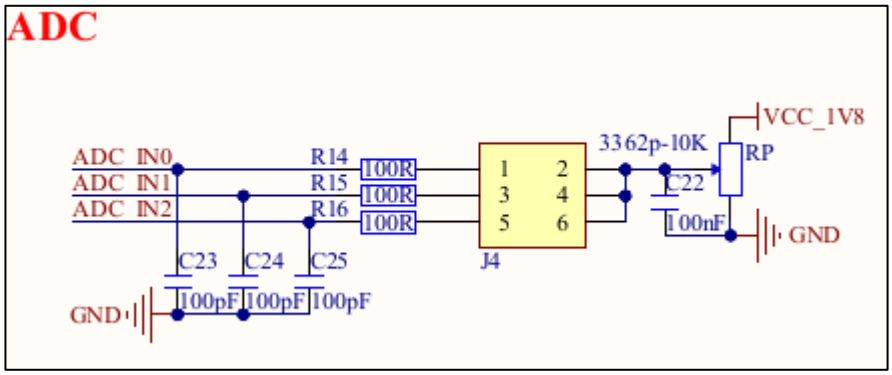


Figure 4.19-1 Schematic diagram of the ADC base board

The following figure shows the ADC location of the ATK-DLIMX93 development board.

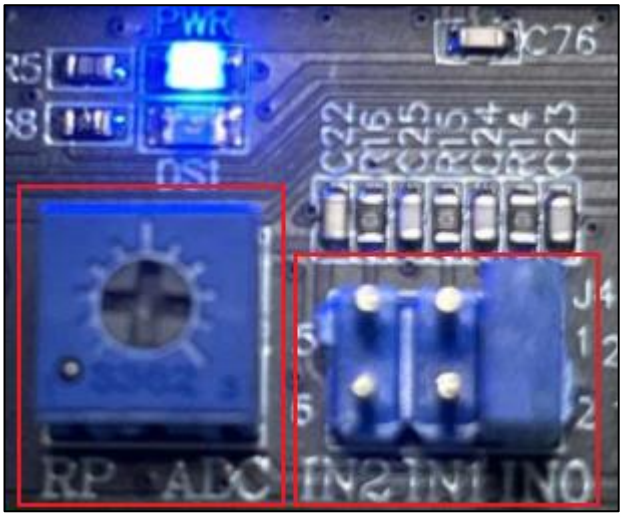


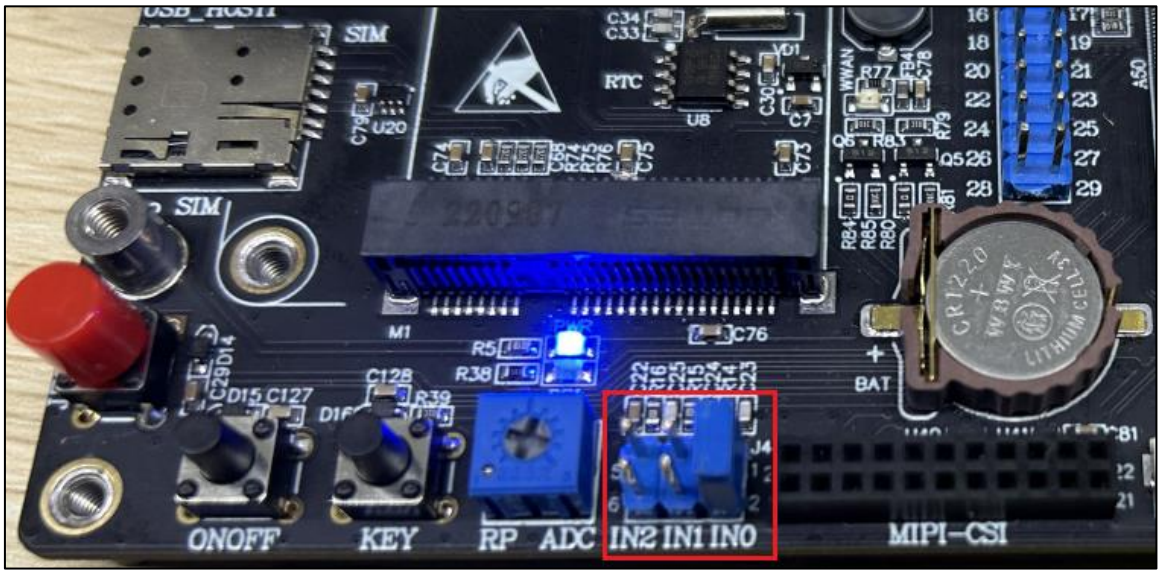
Figure 4.19-2 Development board ADC location

The main features of ADC are:

- The IMX93 supports ADC functionality and provides an ADC controller, which is an 8-channel 12-bit 1 MS/s ADC with 4 channels connected to pins in the processor.

In Linux system, ADC sampling belongs to the IIO subsystem, and the ADC controller can be accessed through the interface provided by sysfs, using 12-bit precision.

Make sure the IN0 jumper cap is connected in hardware, as shown in the following figure:



Use the following command to get the data sampled by ADC1, where the value read out is the raw raw_value.

```
cat /sys/bus/iio/devices/iio\:device0/in_voltage0_raw
```

```
root@atk-imx93:~# cat /sys/bus/iio/devices/iio\:device0/in_voltage0_raw
2105
root@atk-imx93:~#
```

Figure 4.19-3 Check the value of raw

Actual voltage value (mV)=in_voltage1_raw * in_voltage_scale.

Where in_voltage_scale is the ADC1 scale file (resolution) in mV, which can be retrieved by the following command and is related to the reference voltage.

cat /sys/bus/iio/devices/iio\:device0/in_voltage_scale

```
root@atk-imx93:~# cat /sys/bus/iio/devices/iio\:device0/in_voltage_scale
0.439453125
root@atk-imx93:~#
```

Figure 4.19-4 Viewing the scale value

Then the actual voltage value of ADC1 is 2105* 0.439453125≈925mV. The ADC1 voltage can be adjusted by adjusting the potentiometer RP_ADC.

The value of other channels of ADC1 can be tested by switching the jumper cap. The channel data read must match the actual connected channel jumper cap, otherwise the test is meaningless.

ADC1 channel original value	Base plate jumper cap connection
/sys/bus/iio/devices/iio\:device0/in_voltage0_raw	IN0
/sys/bus/iio/devices/iio\:device0/in_voltage1_raw	IN1
/sys/bus/iio/devices/iio\:device0/in_voltage2_raw	IN2

4.20 M33 core test

The development board uses CH342 chip to divide one serial port into two channels:

The A-core serial port uses USB-Enhanced-SERIAL-A-CH342,

The M-core serial port uses USB-Enhanced-SERIAL-B-CH342.

Before we explained that we always use the factory system based on the A-core boot, so we choose USB-Enhanced-SERIAL-A-CH342 as the serial port. A serial port printing example of M33 is integrated on the factory system. We open a new serial port terminal and select USB-enenhanced -SERIAL-B-CH342 as the serial port, which is used to display the printing information of M core with a baud rate of 115200. The configuration is as follows:

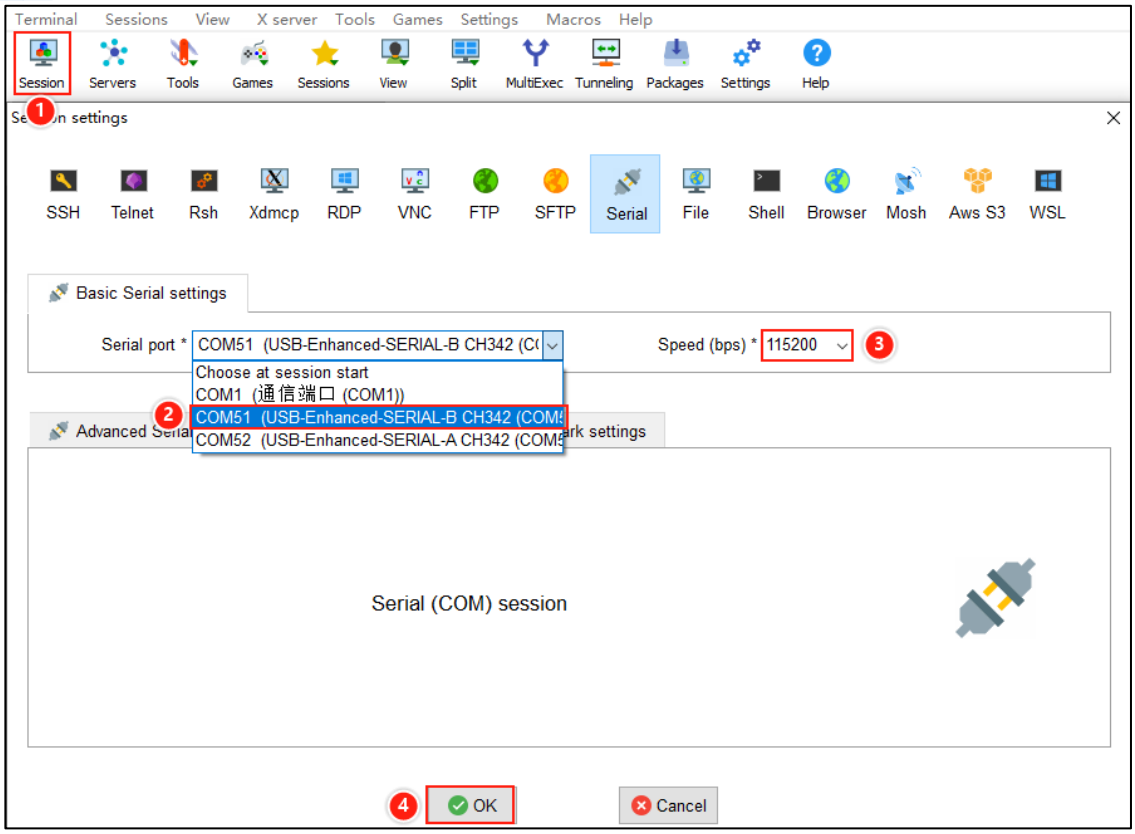


Figure 4.20-1 M33 serial port configuration

Return to the serial port terminal of the factory system (that is, the A-core serial port terminal), switch to the M33 test program directory in the factory system and execute the M33 routine.

```
cd /lib/firmware/
./load_remoteproc.sh hello_world.elf
```

```
root@atk-imx93:~# cd /lib/firmware/
root@atk-imx93:/lib/firmware# ls
ethosu_firmware  load_remoteproc.sh  regulatory.db.p7s  tee.elf              tee-pager_v2.bin
hello_world.elf  nxp                 rtlbt              tee-header_v2.bin    tee-raw.bin
jailhouse.bin    regulatory.db       tee.bin            tee-pageable_v2.bin  unload_remoteproc.sh
root@atk-imx93:/lib/firmware# ./load_remoteproc.sh hello_world.elf
[   44.209310] remoteproc remoteproc0: powering up imx-rproc
[   44.216905] remoteproc remoteproc0: Booting fw image hello_world.elf, size 247344
[   44.224470] remoteproc remoteproc0: header-less resource table
[   44.230319] remoteproc remoteproc0: No resource table in elf
[   44.741247] remoteproc remoteproc0: remote processor imx-rproc is now up
Remote Processor started.
root@atk-imx93:/lib/firmware#
```

Figure 4.20-2 Run the M33 test program

After running the program, the M-core serial port terminal will print a hello world.

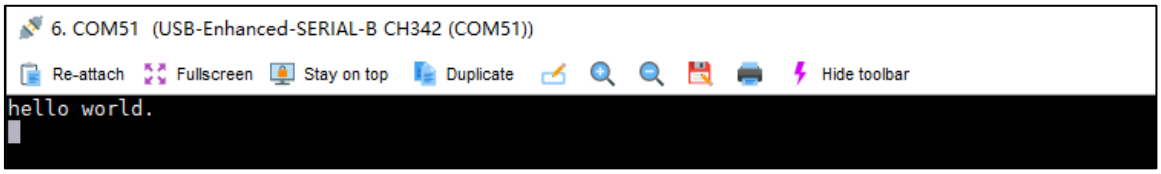


Figure 4.20-3 M33 Print information

When the test is complete, execute the following command to close the program.

```
./unload_remoteproc.sh
```

```
root@atk-imx93:/lib/firmware# ./unload_remoteproc.sh
[  170.002905] remoteproc remoteproc0: stopped remote processor imx-rproc
Remote Processor stoped.
root@atk-imx93:/lib/firmware#
```

Figure 4.20-4 Close the M33 test program

4.21 NPU test

ATK-DLIMX93 development board factory system QT interface integrates a routine of AI camera, this program can be used to test the NPU function of the development board. Before using the AI camera of QT interface, the OV5645 camera and screen need to be connected to the development board.

Figure 4.21-1 Qt Interface -AI Camera

Open the AI camera in the main screen, this routine uses object detection, the effect is as follows:

Figure 4.21-2 AI Camera effects

For more AI routines, please see the development board CD-ROM A - Basic Information \9_tutorials \ "[ALIENTEK] ATK-DLIMX93 Embedded AI Development Manual.pdf

Chapter 5. Introduces the function of ATK-DLIMX93 file system

5.1 Introduction to filesystem directories

ATK-DLIMX93 factory Qt file system directory Introduction:

Directory	The content of a directory function or placement
/bin	The system places a number of execution files, such as mv, cp, date instructions, which are generally used in single-user maintenance mode and can also be used by the root user
/boot	The boot directory usually holds boot files, but i.MX93 already has a separate boot partition for storing boot files (Image and device tree dtb files).
/dev	Any device and interface used by Linux system are stored in the form of files in this directory, in Linux everything below is a file
/etc	This is where almost all system configuration files, such as startup scripts and all service files, are placed
/home	The user's home folder, which contains the root user by default. This is where the user home folder will be generated for new users
/lib	System libraries, and kernel modules
/lost+found	Used to store file fragments or missing files found during file system repair
/media	A mount point for mounting removable storage devices such as USB flash drives, optical disks, and so on
/mnt	Generally used to temporarily mount directories
/opt	Third party software placed in the directory, so the factory Qt applications and used song lyrics are placed in this directory
/proc	This directory is a virtual filesystem that holds data such as the kernel, processes, peripherals, and network status in memory and therefore does not occupy disk space
/root	This directory is the home directory of the superuser, where they can store and manage their own files, configurations, and scripts, usually in /home/root
/run	Places files needed by the system to run, formerly in /var/run but later in a separate /run directory. After the restart, the corresponding directory data is regenerated
/sbin	The instructions placed in the sbin directory are system instructions that can only be executed by the root user
/srv	It can be used to store system operation services or service related data
/sys	Like the /proc directory, it is a virtual filesystem that records hardware information about the core system
/tmp	Directory to store temporary files
/unit_tests	It can be used to store code and files related to unit tests
/usr	Linux kernel directory, the directory contains a lot of files, such as shared files, some executable files and so on
/var	Stores files that change frequently during system execution

5.2 Examining the Qt environment variables

On the factory filesystem, the Qt environment variable is stored in /etc/profile.d/qt.sh, as shown in the following figure:

```
root@atk-imx93:~# cat /etc/profile.d/qt.sh
export LANG=en_US.UTF-8
export QT_QPA_FONTDIR=/usr/share/fonts
export QT_QPA_PLATFORM=linuxfb
root@atk-imx93:~#
```

Figure 5.2-1 Look at the Qt environment variables for the filesystem

5.3 How can I disable Qt desktop startup

In older kernels and Yocto, the filesystem used already uses the Systemd way to launch daemons, which has become the standard configuration for most distributions. If you have learned Systemd, then you will easily understand the Qt desktop service startup configured by ALIENTEK.

In the file system directory/etc/systemd/system/systemui. Service this service, we can view what Qt desktop services. How to disable the ALIENTEK Qt desktop boot? The factory default is to start systemui.service. If we don't need to start Qt desktop. Then we execute the following command.

```
systemctl disable systemui.service
// Perform this to close Qt desktop service, the next start will not be effective desktop
systemctl enable systemui.service
// Do this to start Qt desktop service, the next time start the effective desktop
systemctl stop systemui.service
// Temporarily stop Qt desktop service, this command is invalid after restart
systemctl restart systemui.service
// Temporarily start Qt desktop service, this command is invalid after restart
```

The above "service" can be relatively complex for beginners. It takes time to get used to it. The functionality of the file system is far more than that, and individuals need to explore and explore by themselves.

5.4 How do you create a self-starter

/etc/profile is executed automatically by the boot system and is executed just before the system is logged in, so we can write the boot to this file.

```
vi /etc/profile
```

Type echo "www.openedv.com" in the following position.

```
                fi
        fi
        # Use the EDITOR not being set as a trigger to call resize
        # and only do this for /dev/tty[A-z] which are typically
        # serial ports
        if [ -z "$EDITOR" -a "$SHLVL" = 1 ] ; then
                case $(tty 2>/dev/null) in
                        /dev/tty[A-z]*) resize >/dev/null;;
                esac
        fi
fi

EDITOR="vi"                     # needed for packages like cron, git-commit
export PATH PS1 OPIEDIR QPEDIR QTDIR EDITOR TERM
alias gst-play-1.0="gst-play-1.0 --audiosink=alsasink"

echo "www.openedv.com"

umask 022
I /etc/profile [Modified] 69/69 100%
```

Figure 5.4-1 Add printing to /etc/profile

After saving the file and restarting the development board, print the following in the serial port terminal. Print out "www.openedv.com."

```
[  OK  ] Mounted /run/med[    8.763770] imx-dwmac 428a0000.ethernet eth1: No Safety Features support found
[    8.780119] imx-dwmac 428a0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp supported

NXP i.MX Release Distro 6.1-mickledore atk-imx93 ttyLP0

atk-i[    8.794292] imx-dwmac 428a0000.ethernet eth1: registered PTP clock
mx93 login: root (automatic login)

[    8.808327] imx-dwmac 428a0000.ethernet eth1: FPE workqueue start
[    8.814654] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    8.826217] 8021q: adding VLAN 0 to HW filter on device eth1
[    8.915131] imx-dwmac 428a0000.ethernet eth1: FPE workqueue stop
[    8.929679] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL)
[    8.947781] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    8.967667] imx-dwmac 428a0000.ethernet eth1: Register MEM_TYPE_PAGE_POOL RxQ-0
[    8.979166] imx-dwmac 428a0000.ethernet eth1: PHY [stmmac-1:01] driver [YT8531 Gigabit Ethernet] (irq=POLL)
[    8.999789] imx-dwmac 428a0000.ethernet eth1: No Safety Features support found
[    9.011813] imx-dwmac 428a0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp supported
[    9.027596] imx-dwmac 428a0000.ethernet eth1: registered PTP clock
[    9.039932] imx-dwmac 428a0000.ethernet eth1: FPE workqueue start
[    9.048223] imx-dwmac 428a0000.ethernet eth1: configuring for phy/rgmii-id link mode
[    9.067214] 8021q: adding VLAN 0 to HW filter on device eth1
www.openedv.com
root@atk-imx93:~#
root@atk-imx93:~# [   19.428224] platform imx93-lpm: deferred probe pending
```

Figure 5.4-2 Self starting effect

If you want to run your own Qt program, you can refer to the following format, for example, I have a Qt executable file named demo at the /home/root/path

```
systemctl disable systemui.service
systemctl stop systemui.service
/home/root/demo >/dev/null 2>&1 &
```

5.5 How do I set a static ip address

Add the following code to the end of the /etc/profile file in the factory filesystem to ensure that the specified network interface starts with a static IP address at boot time. Here is an example:

```
ifconfig eth0 192.168.6.200 netmask 255.255.255.0 broadcast 192.168.6.255
route add default gw 192.168.6.1

ifconfig eth1 192.168.6.210 netmask 255.255.255.0 broadcast 192.168.6.255
route add default gw 192.168.6.1
```

```
        # Use the EDITOR not being set as a trigger to call resize
        # and only do this for /dev/tty[A-z] which are typically
        # serial ports
        if [ -z "$EDITOR" -a "$SHLVL" = 1 ] ; then
                case $(tty 2>/dev/null) in
                        /dev/tty[A-z]*) resize >/dev/null;;
                esac
        fi
fi

EDITOR="vi"                     # needed for packages like cron, git-commit
export PATH PS1 OPIEDIR QPEDIR QTDIR EDITOR TERM
alias gst-play-1.0="gst-play-1.0 --audiosink=alsasink"

ifconfig eth0 192.168.6.200 netmask 255.255.255.0 broadcast 192.168.6.255
route add default gw 192.168.6.1

ifconfig eth1 192.168.6.210 netmask 255.255.255.0 broadcast 192.168.6.255
route add default gw 192.168.6.1

umask 022
~
I /etc/profile [Modified] 35/75 46%
```

Figure 5.5-1 Add a static IP address example

This code uses the ifconfig command to set the "eth0" and "eth1" network interfaces to IP addresses 192.168.6.200 and 192.168.6.210, and the subnet mask to 255.255.255.0. The broadcast address is 192.168.6.255 and both interfaces are enabled. Then use the route command to add a default gateway that routes all external traffic to 192.168.6.1.

Make sure to add this code to the end of /etc/profile and save /etc/profile. When the system is restarted, it will automatically execute these configuration commands, so that the "eth0" and "eth1" network interfaces are configured according to the command content, and the network port is enabled by default, but the network cable needs to be connected in the actual test. In a real-world environment, you should use the correct IP address assigned by your network and the default gateway, and ensure that the IP addresses do not conflict.

```
root@atk-imx93:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.6.200  netmask 255.255.255.0  broadcast 192.168.6.255
        ether 88:ba:c3:91:ce:3c  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=-28605<UP,BROADCAST,RUNNING,MULTICAST,DYNAMIC>  mtu 1500
        inet 192.168.6.210  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::8a4f:b4ff:fe43:b7eb  prefixlen 64  scopeid 0x20<link>
        ether 88:4f:b4:43:b7:eb  txqueuelen 1000  (Ethernet)
        RX packets 5392  bytes 448269 (437.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 74  bytes 9836 (9.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 117

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 222  bytes 19970 (19.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 222  bytes 19970 (19.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 5.5-2 Static IP effect