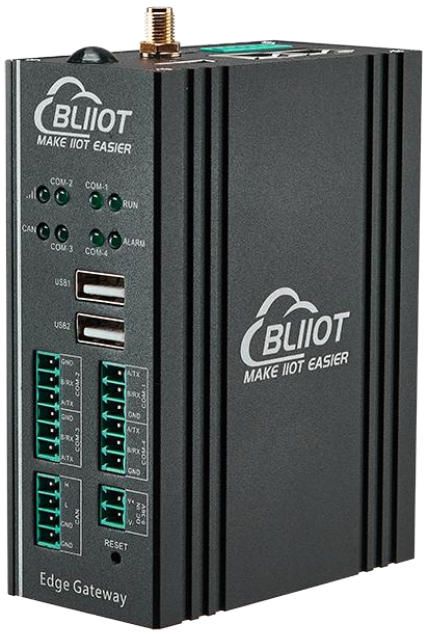
Embedded ARM Computer
BL303 BL304



BLIIOT®
MAKE IIOT EASIER

BL303/304 User Manual

Version: V1.0

Date: 2023-8-1

Shenzhen Beilai Technology Co.,Ltd

Website: https://www.bliiot.com

Preface

Thanks for choosing BLIIoT Embedded ARM Computer BL303 BL304. These operating instructions contain all the information you need for operation of a device in the EdgeCOM BL30 family.

Copyright

Disclaimer

This document is designed for assisting user to better understand the device. As the described device is under continuous improvement, this manual may be updated or revised from time to time without prior notice. Please follow the instructions in the manual. Any damages caused by wrong operation will be beyond warranty.

Revision History

Revision Date	Version	Description	Owner
2023/8/1	V1.0	Initial Release	LKY

Table of Contents

1 Introduction

1.1 Overview

The BL303/BL304 series Embedded ARM Computer use NXP i.MX8M Mini Quad-core 64-bit processor, running speed up to 1.8GHz, with advanced ARM 4-core Cortex-A53 architecture, 2GB DDR4 RAM, support one general-purpose Cortex®-M4 400MHz core processor, 8G eMMC.
BL304 comes with 4 RS485 or RS232, 1 CAN port, 2 Ethernet ports, 2 DI, 2 PWM output and 2 USB port, 1 power input/output port, 1 HDMI, 1 Mini PCIe expansion slot for a wireless module.
The computer supports LINUX, Ubuntu, Debian, Node-Red, QT, Python, C++; MySQL, InfluxDB, SQLite, Docker, Ignition, and Eclipse. This tiny embedded computer is widely applicable to a variety of industrial solutions.

1.2 Features

- NXP i.MX8M Mini processor, ARM Cortex-A53 architecture
- Comes with Ethernet ports, RS485 or RS232 serial ports
- 1 Mini PCIe expansion slot for 4G/WiFi module
- Supports LINUX, Ubuntu, Debian; Node-Red, QT, Python, C++; MySQL, InfluxDB, SQLite; Docker, Ignition, Eclipse.
- Automatic frequency reduction or restart when the chip is overheated
- Chip frequency can be adjusted manually
- Multiple sleep modes, with timing wake-up function
- IP30 protection; metal shell and system are safely isolated; DIN rail installation
- 110mmx43mmx83mm(LxWxH) tiny embedded computer

1.3 Application scenarios

BL303/BL304 series Embedded ARM Computer are widely applicable to IoT, Industrial IoT, digital factories, industrial automation, energy monitoring, smart security, rail transit, telecommunications, smart EV charging, human-computer interaction and other fields.

1.4 Technical Specifications

Item	Parameter	Description
System	Processor	i.MX8M Mini Quad-Core 64-bit, 1.8HGz
	RAM	2GB DDR4
	Flash	8GeMMC
Power	Input Voltage	DC 9～36V
	Power Consumption	Normal: 360mA@12V, MAX 550mA@12V
	Wiring	Anti- Inverse Connection Protection
Ethernet Port	Interface Spec	2 x RJ45, 1x10/100Mbps, 1x10/100/1000Mbps, adaptive MDI/MDIX
	Protection	ESD ±16kV (contact), ±18kV (air), EFT 40A (5/50ns), Lightening 6A (8/20μs)
Serial Port	QTY	4 x RS485/ RS232
	Baud Rate	300bps-115200bps
	Data Bit	7, 8
	Parity Bit	None, Even, Odd
	Stop Bit	1, 2
	Protection	ESD ±8kV (contact), ±15kV (air) EFT 2KV, 40A (5/50ns)
CAN Port	QTY	1
	MAX Speed	1Mbps
SIM Card	QTY	2 SIM Card Slot
	Spec	Drawer type slot, support 1.8V/3V SIM/UIM card (NANO)
	Protection	Built-in 15KV ESD Protection
Digital Input	QTY	2
	Input Type	Both Dry contact and Wet contact(NPN)
	Dry Contact	Close: Short circuit Open: Open circuit
	Wet Contact	Logic 0: 0-2.5VDC Logic 1: 10-30VDC
	Isolation protection	2KVrms
Digital Output	QTY	2
	Output Type	PWM

USB Port	QTY	1xmicro USB, 2x USB2.0
	Protection	Over Current Protection
SD Card Slot	QTY	1
	Spec	Supports SD, SDHC and SDXC (UHS-I) cards
HDMI	QTY	1
Antenna	QTY	1x Cellular antenna, 1xWiFi Antenna
	Type	SMA Hole Type
4G Module (Optional)	L-E version	GSM/EDGE:900,1800MHz WCDMA:B1,B5,B8 FDD-LTE:B1,B3,B5,B7,B8,B20 TDD-LTE:B38,B40,B41
	L-CE version	GSM/EDGE:900,1800MHz WCDMA:B1,B8 TD-SCDMA:B34,B39 FDD-LTE:B1,B3,B8 TDD-LTE:B38,B39,B40,B41
	L-A version	WCDMA:B2,B4,B5 FDD-LTE:B2,B4,B12
	L-AU version	GSM/EDGE:850,900,1800MHz WCDMA:B1,B2,B5,B8 FDD-LTE:B1,B3,B4,B5,B7,B8,B28 TDD-LTE:B40
	L-AF version	WCDMA:B2,B4,B5 FDD-LTE:B2,B4,B5,B12,B13,B14,B66,B71
	CAT-1 version	GSM:900,1800 FDD-LTE:B1,B3,B5,B8 TDD-LTE:B34,B38,B39,B40,B41
WiFi(Optional)	Interface	PCIe
	Protocol	IEEE 802.11b/g/n
	Mode	STA, AP
	Frequency	2.4GHz
	Channel	Ch1 ~ Ch13
	Security	Open, WPA, WPA2
	Encryption	AES, TKIP, TKIPAES
	Connection	8(Max)
	Speed Rate	150Mbps(Max)
	Transmission distance	Outdoor/Open area, up to 20 meters
	SSID	Support

Indicator	QTY	LEDx8
Environment	Working	0～70℃, 10～90% RH
Others	Case	Metal Case
	Size	110mmx43mmx83mm(LxWxH)
	Protection	IP30
	Mounting	DIN-Rail Mounting

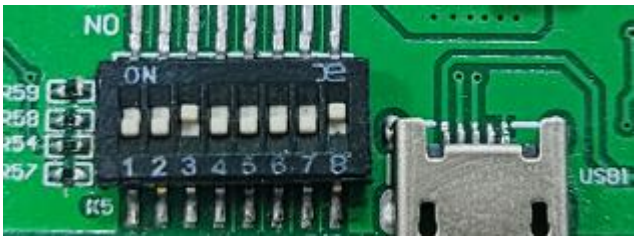
1.5 Model Selection

Model	BL303	BL304	BL303T	BL304T
Processor	iMX8M	iMX8M	iMX8M	iMX8M
CPU Frequency	MAX 1.8GHz	MAX 1.8GHz	MAX 1.6GHz	MAX 1.6GHz
RAM	2G DDR4	2G DDR4	2G DDR4	2G DDR4
Flash	8G eMMC	8G eMMC	8G eMMC	8G eMMC
ETH	1x100M, 1x1000M	1x100M, 1x1000M	1x100M, 1x1000M	1x100M, 1x1000M
USB	2	2	2	2
RS232/RS485	2	4	2	4
CAN	x	1	x	1
SD slot	1	1	1	1
MINI-PCIe	x	1	x	1
4G(GPS)/WIFI	x	√	x	√
SIM slot	x	2	x	2
HDMI	1	1	1	1
DI	x	2	x	2
DO	x	2	x	2
Temperature	0℃ ~ 70℃	0℃ ~ 70℃	-40℃ ~ 70℃	-40℃ ~ 70℃

2 System Programming

2.1 Settings

The computer supports USB OTG and TF card programming, supports eMMC and QSPI startup, DIP switche K5 is located on the side of the mini-usb port and K3 is located on the back of the core board where the network port is located. K3 is all OFF by default, and uses DIP switch (K5) to distinguish different operation methods (eMMC startup as shown in the figure below)



Switch / Mode	(K3) 1	(K3) 2	(K3) 3	(K3) 4	(K3) 5	(K3) 6	(K3) 7	(K3) 8	(K5) 1	(K5) 2	(K5) 3	(K5) 4
eMMC startup	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	**ON**	OFF
TF Programming	OFF	**ON**	OFF	OFF	OFF	**ON**	**ON**	**ON**	**ON**	OFF	**ON**	OFF
QSPI startup	**ON**	OFF	**ON**	OFF	**ON**	OFF	OFF	OFF	**ON**	**ON**	**ON**	OFF
OTG Programming	X	X	X	X	X	X	X	X	X	X	OFF	**ON**

2.2 Programming via TF Card

When using a TF card to programming, please use a genuine card with a capacity of 8G and above to test, and the format should be FAT32 format.

Creating a TF card

Copy the folder sdfuse to the VM TF card insert it into the VM and go to the folder.

```
ubuntu:~$ cd imx8mm/sdfuse/
ubuntu:~/imx8mm/sdfuse$ sudo ./mksdcard8mm.sh
[sudo] password for forlinx: //Need to enter forlinx No display of password：forlinx, Press enter to co
ntinue
[…]
```

```
Availible Drives to write images to:
# major minor size name
1: 8 16 15558144 sdb
Enter Device Number: 1 //Select TF card device, enter 1
sdb was selected1
Checking the device is unmounted
unmounting device '/dev/sdb1'
Would you like to re-partition the drive anyways [y/n] : y //Enter "y" to confirm
Now partitioning sdb ...
###############################################################################
Now making 1 partitions
###############################################################################
1+0 records in
1+0 records out
1024 bytes (1.0 kB, 1.0 KiB) copied, 0.00133858 s, 765 kB/s
DISK SIZE - 1024 bytes
[…]
###############################################################################
Partitioning Boot
###############################################################################
mkfs.fat 3.0.28 (2015-05-16)
mkfs.fat: warning - lowercase labels might not work properly with DOS or Windows
Buring th sdfuse.bin to sdcard
1359+1 records in
1359+1 records out
1392324 bytes (1.4 MB, 1.3 MiB) copied, 0.00231964 s, 600 MB/s
Syncing....
unmounting device '/dev/sdb1'
```

Print the information as shown above, indicating that the TF card has been successfully created. After making the card according to the script at this time, the TF card will be uninstalled automatically. If you want to make the card and still mount it on the development environment, you need to modify the card making script.

mksdcard8mm.sh：

```
ubuntu:~/imx8mm/sdfuse$ vi mksdcard8mm.sh
[…]
for i in `ls -1 $DRIVE?`; do
echo "unmounting device '$i'"
# umount $i 2>/dev/null                    //Comment out this command at the end of the script
done
```

Copy the image to the TF card, the specific options to be copied are listed in the table below

File	Description
config.ini	Programming configuration files, in the tools directory
update.itb	Programming tools, in the tools directory
flash_qspi.bin	U-Boot-QSPI image, when you need to programming the QSPI image in the development board, put this image into the TF card (optional)
env.ini	Write additional environment variables to uboot (optional)
m4_flash.bin	M4 Mirror Image (optional)
rootfs.sdcard	Image package, contains uboot and kernel, if you are using the original uboot, then just put this one into the TF card.

Mount the TF card to the computer, copy the rootfs.sdcard and update.itb under the image file in the windows system to the finished TF, and then copy the config.ini under the tool\sdfuse folder to the TF card. The content of the finished TF is shown below:

Insert the TF card into the device, turn switches 1 and 3 of K5 to ON, and 2, 6, 7 and 8 of the other switch (on the back) to ON. Turn on the power switch, and the device will automatically start to enter the programming program.

Other Notices

- If you are using A-core uboot, unmodified, just put rootfs.sdcard in the TF card, which contains the standard uboot image.
- If m4_flash.bin exists, it will be automatically programmed into qspi flash for QSPI boot of M-core, and m4_flash.bin will be burned into emmc with 5M offset at the same time. If m4_flash.bin does not exist, and flash_qspi.bin exists, it will be automatically burned into qspiflash for QSPI booting, and if it does not exist, it will not be burned by default.
- When using the TF card to burn, you can add a text named env.ini to the TF card, during the burning process, the burning program reads the environment variables by line, and the environment variables that meet the format will be appended to the default environment variables. env.ini has one environment variable for each line, and the format is: the name of the environment variable = the value of the environment variable.
- Users programming the already compiled image into the storage media during the TF card

programming process. The already compiled M4 image named m4_flash.bin can be put into the TF card, and during the programming process, the M4 image will be programmed into the emmc at an offset of 5M or the starting position of QSPI (QSPI Flash exists). Users can set the environment variable m4_run to start the M4 image. For example compile the image to run M4 image on QSPI, set m4_run=sf probe; bootaux 0x8000000.

- After Programming, if emmc mode startup report an error: [end kernel panic -not syncing:Attempted to kill init!], it means that there is a problem with programming, not successful, need to re-program.

2.3 Programming via OTG

Please use USB2

2.3.1 Programming with Linux

1. The command to burn uboot does not clear the environment variables; to restore the default environment variables enter them on the uboot command line:

```
u-boot=> env default -a -f //Restore default environment variables
u-boot=> saveenv //Save Environment Variables
```

2. QSPI flash can only store one of the M4 image or the uboot image, not both.
3. m4_flash.bin is the M4 image compiled by the user.

Copy uuu from the tool file to the /usr/bin/ directory under the development environment and add executable permissions; connect the device to the computer and turn dip switch 4 to ON.

```
ubuntu:~$ cd /home/forlinx/imx8mm/OK8MM-linux-sdk/images
```

(1) Programming system (u-boot, kernel and filesystem) to emmc:

```
ubuntu:~/imx8mm/OK8MM-linux-sdk/images$ sudo uuu -b emmc_all flash_sd_em\ mc.bin rootfs.sdcard
```

(2) Programming u-boot to emmc

```
ubuntu:~/imx8mm/OK8MM-linux-sdk/images$ sudo uuu -b emmc flash_sd_emmc.bin
```

(3) Programming u-boot to QSPI flash:

```
ubuntu:~/imx8mm/OK8MM-linux-sdk/images$ sudo uuu -b qspi flash_qspi.bin
```

(4) Programming M4 image to QSPI flash:

```
ubuntu:~/imx8mm/OK8MM-linux-sdk/images$ sudo uuu -b qspi flash_qspi.bin \
m4_flash.bin
```

2.3.2 Programming with Windows

Copy the uuu.exe file to the C:\Windows\System32 directory. Then extract the platform-tools_r28.0.3-windows.zip file to C:\Windows\System32, or C:\Windows\SysWOW64 for

64-bit windows systems.

Use OTG cable to connect the device to the computer, turn the 4 of the dip switch to ON to start the device.

Create the uuu directory in D drive and copy the image into it, run the cmd program under Windows to enter the directory where the image file is located, there are four kinds of programming commands, corresponding to different ways of programming and different ways of programming the image, if you are not clear about the difference, we suggest that you choose the first programming command directly.

(1) Programming u-boot, kernel, and filesystem to emmc:

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Administrator>D:                                    //Go to disk D
D:\>cd uuu                                                   //Go to the uuu folder
D:\uuu> uuu.exe -b emmc_all flash_sd_emmc.bin rootfs.sdcard
```

(2) Programming u-boot to emmc:

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Administrator>D:
D:\>cd uuu
D:\uuu> uuu.exe -b emmc flash_sd_emmc.bin
```

(3) Programming u-boot to QSPI flash:

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Administrator>D:
D:\>cd uuu
D:\uuu> uuu.exe -b qspi flash_qspi.bin
```

(4) Programming M4 image to QSPI flash:

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Administrator>D:
D:\>cd uuu
D:\uuu> uuu.exe -b qspi flash_qspi.bin m4_flash.bin
```

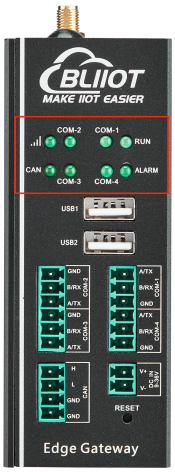
3 Hardware Specifications

3.1 Power Interface

BL303/BL304 comes with 1 power input. Support DC 9~36V input, anti-reverse connection protection.

3.2 LED Indicators

The following figures shows the LED indicators, and the order from left to right and from top to bottom is LED6, LED5, LED1, LED2, LED8, LED7, LED3, LED4, correspondence with the LEDs in the /sys/class/leds directory.



View trigger conditions:

```
root@okmx8mm:~# cat /sys/class/leds/led1/trigger
[none] rc-feedback nand-disk mmc0 timer oneshot heartbeat backlight gpio
```

Where [none] means the current trigger condition of led1 is none. Write the above string to trigger to

modify the trigger condition.

When the led trigger condition is set to none, user can control the led light on or off by command.

```
root@okmx8mm:~# echo none > /sys/class/leds/led1/trigger
```

Control LED1 lights ON

```
root@okmx8mm:~# echo 1 > /sys/class/leds/led1/brightness
```

Controls LED1 lights OFF

```
root@okmx8mm:~# echo 0 > /sys/class/leds/led1/brightness
```

The other 7 led lights are similar, just change under /sys/class/leds to the one corresponding to the corresponding ledx.

3.3 RS485&RS232 Serial Port

BL303/BL304 comes with RS485 or RS232. COM1, COM2, COM3 and COM4 are corresponding to /dev/ttymxc0, /dev/ttymxc1, /dev/ttymxc3 and /dev/ttymxc2 respectively. The R485 serial port supports a maximum baud rate of 115200 with a cable length of 200 meters.

For debugging, enter the following command:

```
busybox microcom -s 115200 /dev/ttymxc0
```

microcom: Command to test the serial port;

-s: Indicates the transmission speed, the baud rate is 115200;

/dev/ttymxc0: Refers to the serial port name of the specific device.

COM2 is the debugging serial port, if you need to use it, you need to convert it to a normal serial port.

3.4 CAN Interface

The CAN interface is as shown in the figure, enter the following command:

ifconfig -a //View all network cards

If the FlexCAN driver works well, you will see the network card interface corresponding to CAN, as shown in the figure, there is a network card named "can0", which is the CAN network card corresponding to the CAN interface on the BL304 board.

```
root@i.MX8MM:~# ifconfig -a
can0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0      Link encap:Ethernet  HWaddr b6:97:b2:51:25:f6
          UP BROADCAST MULTICAST DYNAMIC  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr 00:0e:c6:87:72:01
          UP BROADCAST MULTICAST DYNAMIC  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth2      Link encap:Ethernet  HWaddr 96:f8:1e:f7:9c:f4
          UP BROADCAST NOARP MULTICAST DYNAMIC  MTU:1500  Metric:1
          RX packets:0 errors:1 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2113 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2113 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:463323 (452.4 KiB)  TX bytes:463323 (452.4 KiB)
```

CAN port usage example:

Connect the CAN ports of the two devices, note that CAN_H connects to CAN_H and CAN_L connects to CAN_L.

First you need to turn CAN0 off

ifconfig can0 down

Use the IP command to set the CAN interfaces of the two devices, first set the baud rate of the CAN interface to 12500 and enter the command shown below:

```
ip link set can0 up type can bitrate 125000 triple-sampling on
```

The speed of the upper two CAN devices should be set to the same! After the speed is set open the can0 card with the following command:

```
ifconfig can0 up //open can0
```

One device is used to receive data and one is used to send data. The device that receives data uses the candump command. Enter the following command:

```
candump can0 //receive data
```

The device sending data uses the cansend command to send 8 bytes of data to the receiving unit: 0X11, 0X22, 0X33, 0X44, 0X55, 0X66, 0X77, 0X88. Enter the following command:

```
cansend can0 5A1#11.22.33.44.55.66.77.88
```

cansend command is used to send CAN data, "5A1" is the frame ID, "11.22.33.44.55.66.77.88" after "#" is the data to be sent, hexadecimal. CAN2.0 can send up to 8 bytes of data at a time, and the 8 bytes of data are separated by a "."

If CAN is working correctly the receiver will receive the 8 bytes of data sent above.

```
root@okmx8mm:~# candump can0 //Receive data from can0
interface = can0, family = 29, type = 3, proto = 1
<0x010> [8] 11 22 33 44 55 66 77 88
```

From the above, it can be seen that the can0 interface on the receiver side received 8 bytes of data with a frame ID of 5A1, indicating that the CAN driver is working properly.

If you want to close can0 enter the following command:

```
ifconfig can0 down
```

If you want to perform a CAN loopback test on a board, set up the CAN according to the following command:

```
ifconfig can0 down //If can0 is already on, close it first
ip link set can0 type can bitrate 500000 loopback on //Open Loopback Test
ifconfig can0 up //Reopen can0
candump can0 & //candump backend receives data
cansend can0 5A1#11.22.33.44.55.66.77.88 //cansend send data
```

If the loopback test is successful then the device receives the data sent to itself as shown in the figure:

```
/ # cansend can0 5A1#11.22.33.44.55.66.77.88
can0  5A1  [8] 11 22 33 44 55 66 77 88
```

3.5 PWM Interface

The PWM port is shown in the figure. PWM devices are under the directory /sys/class/pwm, where PWM1 and PWM2 are applied to pwmchip0 and pwmchip1.

Taking PWM1 as an example, first you need to call out the pwm0 directory under pwmchip0, and enter the following command:

echo 0 > /sys/class/pwm/pwmchip0/export

After the execution is completed, a subdirectory named "pwm0" will be generated under the pwmchip0 directory, as shown below

```
root@okmx8mm:/sys/class/pwm/pwmchip0# ls
device    export    npwm    power    subsystem    uevent    unexport
root@okmx8mm:/sys/class/pwm/pwmchip0# echo 0 > export
root@okmx8mm:/sys/class/pwm/pwmchip0# ls
device    export    npwm    power    pwm0    subsystem    uevent    unexport
```

Enable PWM1: Enter the following command to enable PWM1

echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable

Set the frequency of PWM1: Note that the period value is set here, and the unit is ns. For example, the period of 20KHz frequency is 50000ns. Enter the following command:

echo 50000 > /sys/class/pwm/pwmchip0/pwm0/period

Set the duty cycle of PWM1: You cannot set the duty cycle directly, please set the ON time of a cycle, that is, the high-level time, for example, the ON time of 20% duty cycle at 20KHz frequency is 10000, enter the following command

echo 10000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle

If you need to adjust the frequency or duty cycle, pay attention to the high-level time when adjusting.

3.6 DI

The devices corresponding to DI1 and DI2 are gpio_input_0 and gpio_input_1 respectively. Select the dry/wet contact mode by the switch below.

Take DI1 as an example, when debugging the DI interface, enter the command:

./inputapp /dev/gpio_input_0

When the dry contact mode is selected, the port is shorted; when the wet contact mode is selected, the input is greater than 3V; the console output is as follows:

```
root@okmx8mm: /opt# ./input64 /dev/gpio_input_0
di value = 0x1
```

3.7 LAN

There are eth0 and eth1 network cards on the bottom board of BL304. When the power is just turned on and the startup is complete, if the network cable is not inserted, you can see that the network port has no IP address with ifconfig. This is because dhcp dynamically allocates ip, when network cable is inserted, the console will print the corresponding Ethernet, and you can check the corresponding IP address with ifconfig.

Software equipment	Supported Speed
ETH1	10/100Mbps
ETH0	10/100/1000Mbps

Note: eth1 and eth0 cannot be used in the same LAN.

Take eth0 as an example.

Under the Linux system, use the ifconfig command to display or configure network devices, and use ethtool to query and set network card parameters.

Set the IP address and view the details of the current network card:

```
root@okmx8mm:~# ifconfig eth0 192.168.1.120 //set ip
root@okmx8mm:~# ifconfig eth0 //Check the network status after setting
eth0 Link encap:Ethernet HWaddr 3A:D9:93:8E:A8:A4
inet addr:192.168.1.120 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::38d9:93ff:fe8e:a8a4%2124311408/64 Scope:Link
inet6 addr: fec0::38d9:93ff:fe8e:a8a4%2124311408/64 Scope:Site
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:28 errors:0 dropped:0 overruns:0 frame:0
TX packets:63 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:11550 (11.2 KiB) TX bytes:11579 (11.3 KiB)
```

inet addr:192.168.1.120 indicates IP setting successful

If your device is connected to a router, and the router supports DHCP automatic IP address assignment, you can enter the command in the HyperTerminal:

```
root@fl-imx6ull:~# udhcpc -i eth0
udhcpc (v1.24.1) started
Sending discover...
Sending select for 192.168.20.101...
Lease of 192.168.20.101 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 222.222.222.222
```

It is used to dynamically obtain the IP address. The "-i" parameter is used to specify the name of the network card. The name of the network card of the wired network is eth0.

The dns server information in the /etc/resolv.conf file will be added automatically.

Modify mac address:

```
root@okmx8mm:~# ifconfig eth0 hw ether 00:00:00:00:00:01
root@okmx8mm:~# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:00:00:00:00:01
inet addr:192.168.20.101 Bcast:192.168.20.255 Mask:255.255.255.0
inet6 addr: fec0::38d9:93ff:fe8e:a8a4%2128292720/64 Scope:Site
inet6 addr: fec0::200:ff:fe00:1%2128292720/64 Scope:Site
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:85 errors:0 dropped:0 overruns:0 frame:0
TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:22942 (22.4 KiB) TX bytes:22259 (21.7 KiB)
```

Set subnet mask:

```
root@okmx8mm:~# ifconfig eth0 netmask 255.255.255.0 //set eth0 subnet mask255.255.255.0
root@okmx8mm:~# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:00:00:00:00:01
inet addr:192.168.20.101 Bcast:192.168.20.255 Mask:255.255.255.0
inet6 addr: fec0::38d9:93ff:fe8e:a8a4%2128915312/64 Scope:Site
inet6 addr: fec0::200:ff:fe00:1%2128915312/64 Scope:Site
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:107 errors:0 dropped:0 overruns:0 frame:0
TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
```

RX bytes:25700 (25.0 KiB) TX bytes:22259 (21.7 KiB)

Set broadcast address

root@okmx8mm:~# ifconfig eth0 broadcast 192.168.1.255//eth0 broadcast address192.168.1.255

root@okmx8mm:~# ifconfig eth0

The print information is as follows:

eth0 Link encap:Ethernet HWaddr 00:00:00:00:00:01

inet addr:192.168.20.101 **Bcast:192.168.1.255** Mask:255.255.255.0

inet6 addr: fec0::38d9:93ff:fe8e:a8a4%2123332464/64 Scope:Site

inet6 addr: fec0::200:ff:fe00:1%2123332464/64 Scope:Site

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:111 errors:0 dropped:0 overruns:0 frame:0

TX packets:132 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:26130 (25.5 KiB) TX bytes:25947 (25.3 KiB)

Bcast:192.168.1.255 indicates broadcast address setting is successful

Add default gateway:

root@okmx8mm:~# route add default gw 192.168.20.1

Delete the default gateway:

root@okmx8mm:~# route del default gw 192.168.20.1

Close the eth0 network card:

root@fl-imx6ull:~# ifconfig eth0 down

Open the eth0 network card:

root@okmx8mm:~# ifconfig eth0 up

fec 20b4000.ethernet eth0: Freescale FEC PHY driver [Micrel KSZ8081 or KSZ8091]

(mii_bus:phy_addr=20b4000.ethernet:01, irq=-1)

root@okmx8mm:~# fec 20b4000.ethernet eth0: Link is Up - 100Mbps/Full - flow control rx/tx

Ping test

First, make sure network cable works well, connect the device and the host or virtual machine with a

network cable, and set the device and the host or virtual machine on the same network segment. For example. The IP address of my device is 192.168.1.174, and the IP address of my virtual machine is 192.168.1.141, and the ping command can be used to ping.

3.8 WiFi Module

The WiFi module is PCIe interface, supports 2.4G frequency, it is compatible with 8821cu, 8723du and 8822BU three kinds of WiFi drivers, and the default router adopts wpa encryption.
After connecting the module and powering on the device, enter the command line. You can check the USB status through the lsusb command as follows

```
root@okmx8mm:/opt# ./app /dev/gpiopci 1          //Power on the WIFI module
root@okmx8mm:~# lsmod
Module Size Used by
mx6s_capture 14876 0
8723du 1313893 0          //WiFi is automatically loaded, and 8723du has been loaded successfully
ov9650_camera 12446 0
```

3.8.1 STA Mode

STA mode is to connect to the wireless network as a station, the operation method is as follows:
-i indicates the WiFi model; -s indicates the name of the WiFi hotspot; -p indicates the password, if there is no password, enter -p NONE; the router uses wpa encryption, and the specific operation instructions can be found in the wifi.sh script

```
root@okmx8mm:~# wifi.sh -i wlan0 -s beilai -p xxx //Execute the test script
```

Print information as below

```
wifi 8723du
ssid beilai
pasw xxx
usbcore: deregistering interface driver rtl8723du
usbcore: registered new interface driver rtl8723du
IPv6: ADDRCONF(NETDEV_UP): wlan0: link is not ready
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
udhcpc (v1.24.1) started
Sending discover...
wlan0: CTRL-EVENT-REGDOM-CHANGE init=BEACON_HINT type=UNKNOWN
```

wlan0: Trying to associate with 04:d7:a5:f9:26:1d (SSID='beilai' freq=2427 MHz)

wlan0: Associated with 04:d7:a5:f9:26:1d

IPv6: ADDRCONF(NETDEV_CHANGE): wlan0: link becomes ready

wlan0: WPA: Key negotiation completed with 04:d7:a5:f9:26:1d [PTK=CCMP GTK=TKIP]

wlan0: CTRL-EVENT-CONNECTED - Connection to 04:d7:a5:f9:26:1d completed [id=0 id_str=]

nf_conntrack: automatic helper assignment is deprecated and it will be removed soon. Use the iptables CT target to attach helpers instead.

Sending discover...

Sending select for 192.168.5.186...

Lease of 192.168.5.186 obtained, lease time 1800

/etc/udhcpc.d/50default: Adding DNS 222.222.202.202

/etc/udhcpc.d/50default: Adding DNS 222.222.222.222

WLAN Finshed!

After the script runs, it can automatically assign IP and generate DNS, and the WiFi connection is successful.

To ping IP or domain name, the command is as follows:

```
root@okmx8mm:~# ping -I 192.168.1.118 www.baidu.com
```

Print information as below

```
ping -I 192.168.1.118 www.baidu.com : 56 data bytes
64 bytes from 192.168.1.118: seq=0 ttl=128 time=39.783 ms
64 bytes from 192.168.1.118: seq=1 ttl=128 time=81.529 ms
64 bytes from 192.168.1.118: seq=2 ttl=128 time=15.236 ms
64 bytes from 192.168.1.118: seq=3 ttl=128 time=12.076 ms
64 bytes from 192.168.1.118: seq=4 ttl=128 time=16.300 ms
--- 192.168.1.118 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 12.076/32.984/81.529 ms
```

Method of checking WiFi signal:

```
root@okmx8mm:~# cat /proc/net/wireless | grep wlan0 | awk '{print $3}' //Get signal strength
root@okmx8mm:~# cat /proc/net/wireless | grep wlan0 | awk '{print $4}' //Get the signal quality, in dBm
root@okmx8mm:~# cat /proc/net/wireless | grep wlan0 | awk '{print $5}' //Network port background noise, in dBm
```

3.8.2 AP Mode

In AP mode, the device can connect with maximum 8 users theoretically

For example, Ethernet eth0 connecting to the router. After configuring the Ethernet, you need to test whether eth0 can connect to the external network. If you can connect to the external network (refer to the "LAN" chapter for the method), please follow the steps. If not, please check whether the Ethernet or router connection is good.

Working in AP mode, mobile phones and other devices can directly connect to the module.

Set Ethernet IP, configure network firewall:

```
root@okmx8mm:~# udhcpc -i eth0 //Automatically assign IP, if the testing of eth0 network works, this step is not required
root@okmx8mm:~# echo 1 > /proc/sys/net/ipv4/ip_forward //Turn on IP forwarding
root@okmx8mm:~# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE//Set forwarding rules
```

Set WiFi mode and IP

```
root@okmx8mm:~# ifconfig wlan0 up //open WiFi
root@okmx8mm:~# ifconfig wlan0 192.168.0.10 netmask 255.255.255.0 //Set IP and subnet mask
root@okmx8mm:~# ifconfig wlan0 promisc //Set wlan0 to promiscuous mode
```

Enable AP

```
root@okmx8mm:~# udhcpd /etc/udhcpd.conf & //Configuration information such as WiFi address and gateway
root@okmx8mm:~# hostapd -d /etc/hostapd.conf & //Encryption method, user name, password and other settings
```

In the hostapd.conf file: ssid is user name, wpa_passphrase is password; mobile phones can connect to the AP hotspot of the device through WiFi, and the device uses the following user name and password by default: Hotspot name: beilaitest Password: 1234567890

Unload modules that have been added to the kernel:

```
root@okmx8mm:~# rmmod 8723du
usbcore: deregistering interface driver rtl8723du
wlan0: CTRL-EVENT-DISCONNECTED bssid=04:d7:a5:f9:26:1d reason=0
```

3.9 4G

The 4G module is PCIE interface. Taking the 4G module as an example, please confirm the firmware

version of the module when using IoT card for testing. Low version firmware does not support it, the firmware needs to upgrade to EC20. After connecting the module and powering on the board, enter the command line, and you can check the USB status through the lsusb command as follows

```
root@okmx8mm:/opt# ./app /dev/gpiopci 1          //Power on the 4G module, and look at the 4 virtual
devices. The device is powered on by default, and writing 0 means power off.
root@okmx8mm:/opt# random: nonblocking pool is initialized
usb 2-1: new high-speed USB device number 2 using ci_hdrc
option 2-1:1.0: GSM modem (1-port) converter detected
usb 2-1: GSM modem (1-port) converter now attached to ttyUSB0
option 2-1:1.1: GSM modem (1-port) converter detected
usb 2-1: GSM modem (1-port) converter now attached to ttyUSB1
option 2-1:1.2: GSM modem (1-port) converter detected
usb 2-1: GSM modem (1-port) converter now attached to ttyUSB2
option 2-1:1.3: GSM modem (1-port) converter detected
usb 2-1: GSM modem (1-port) converter now attached to ttyUSB3
GobiNet 2-1:1.4 eth2: kevent 12 may have been dropped
GobiNet 2-1:1.4 eth2: register 'GobiNet' at usb-ci_hdrc.1-1, GobiNet Ethernet Device, 9e:33:27:a1:5a:2c
creating qcqmi2
GobiNet 2-1:1.4 eth2: kevent 12 may have been dropped
IPv6: ADDRCONF(NETDEV_UP): eth2: link is not ready
root@imx6ulevk:~# lsusb
Bus 001 Device 004: ID 0bda:b720
Bus 001 Device 005: ID 2c7c:0125                  //EC20 VID and PID
Bus 001 Device 002: ID 0424:2514
Bus 001 Device 001: ID 1d6b:0002
```

/dev Check the device node status

```
root@okmx8mm:/opt# ls /dev/ttyUSB*
/dev/ttyUSB0    /dev/ttyUSB1    /dev/ttyUSB2    /dev/ttyUSB3
```

EC20 dial-up: Sometimes Sending discover... will appear several times, which is caused by poor signal.

```
root@okmx8mm:~# ec20.sh &
[1] 598
root@okmx8mm:/forlinx/cmdbin#[04-26_19:16:06:781]
WCDMA&LTE_QConnectManager_Linux&Android_V1.1.34
[04-26_19:16:06:783] ./quectel-CM profile[1] = (null)/(null)/(null)/0, pincode = (null)
[04-26_19:16:06:790] Find /sys/bus/usb/devices/1-1.1 idVendor=2c7c idProduct=0125
```

[04-26_19:16:06:791] Find /sys/bus/usb/devices/1-1.1:1.4/net/eth2

[04-26_19:16:06:791] Find usbnet_adapter = eth2

[04-26_19:16:06:792] Find /sys/bus/usb/devices/1-1.1:1.4/GobiQMI/qcqmi2

[04-26_19:16:06:792] Find qmichannel = /dev/qcqmi2

[04-26_19:16:06:851] Get clientWDS = 7

[04-26_19:16:06:882] Get clientDMS = 8

[04-26_19:16:06:914] Get clientNAS = 9

[04-26_19:16:06:946] Get clientUIM = 10

[04-26_19:16:06:978] Get clientWDA = 11

[04-26_19:16:07:011] requestBaseBandVersion EC20CEHCLGR06A0**5M1G**

//If the version number in the printed information is 5Mxx, that means it supports the IoT card, if it is 2Mxx, then it does not support IoT card

[04-26_19:16:07:106] requestGetSIMStatus SIMStatus: SIM_READY

[04-26_19:16:07:138] requestGetProfile[1] ctnet///0

[04-26_19:16:07:171] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached, DataCap: LTE

[04-26_19:16:07:202] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED

[04-26_19:16:07:266] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached, DataCap: LTE

[04-26_19:16:07:300] requestSetupDataCall WdsConnectionIPv4Handle: 0xe1645ec0

[04-26_19:16:07:394] requestQueryDataCall IPv4ConnectionStatus: CONNECTED

[04-26_19:16:07:427] ifconfig eth2 up

[04-26_19:16:07:471] busybox udhcpc -f -n -q -t 5 -i eth2

[04-26_19:16:07:506] udhcpc (v1.24.1) started

[04-26_19:16:07:631] Sending discover...

[04-26_19:16:07:691] Sending select for 172.29.86.131...

[04-26_19:16:07:751] Lease of 172.29.86.131 obtained, lease time 7200

[04-26_19:16:07:869] /etc/udhcpc.d/50default: Adding DNS 222.222.222.222

[04-26_19:16:07:869] /etc/udhcpc.d/50default: Adding DNS 222.222.202.202

After the connection is successful, ping Baidu to test:

root@okmx8mm:~# ping www.baidu.com

PING www.baidu.com (220.181.38.150): 56 data bytes

64 bytes from 220.181.38.150: seq=0 ttl=53 time=137.243 ms

64 bytes from 220.181.38.150: seq=1 ttl=53 time=51.239 ms

64 bytes from 220.181.38.150: seq=2 ttl=53 time=94.440 ms

The 4G module has two SIM card slots, select SIM1 or SIM2 by setting /dev/gpiosgm, after the system starts, the default is SIM1, write 1 to the /dev/gpiosgm device, and select SIM2. After replacing the card slot, please power on the module again.

```
root@okmx8mm:/opt# ./app /dev/gpiosgm 1
```

If both wired network and 4G network are used at the same time, one of them will be unavailable due to gateway priority. If you need to change it, you can enter the following command to view the current default gateway information.

```
ip route show
```

Change the gateway information.

```
sudo route add default gw 172.29.86.131    //add 4G module gateway, gw followed by IP address
sudo route del default gw 172.29.86.0    //delete 4G module gateway, gw followed by IP address
```

3.10 USB Port

Support hot swapping of USB mouse, USB keyboard, and U disk devices.

When using a USB flash drive, it is recommended to use a formatting tool to format it into a FAT32 format that can be recognized by the linux system. The mounted directory of the U disk is /run/media, insert the U disk, and the following information is displayed:

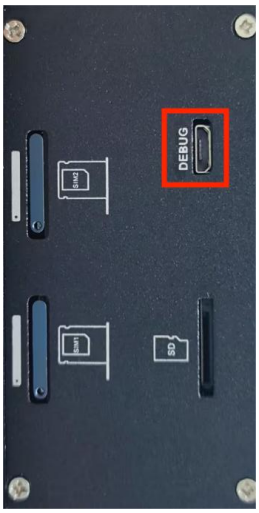
```
root@okmx8mm:~# usb 1-1.3: new high-speed USB device number 5 using ci_hdrc
usb-storage 1-1.3:1.0: USB Mass Storage device detected
scsi host1: usb-storage 1-1.3:1.0
scsi 1:0:0:0: Direct-Access Generic MassStorageClass 1536 PQ: 0 ANSI: 6
sd 1:0:0:0: [sda] 31116288 512-byte logical blocks: (7.94 GB/7.40 GiB)
sd 1:0:0:0: [sda] Write Protect is off
sd 1:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
sda: sda1 //the mount device name is sda1
sd 1:0:0:0: [sda] Attached SCSI removable disk
FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
```

Check the usb storage device, /run/media is the mounting directory of the U disk, and the device name after the U disk is mounted is sda1.

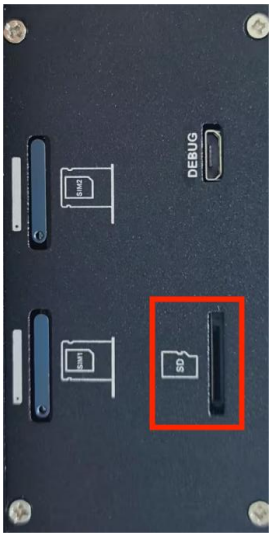
If you don't need to use the U disk anymore, please use umount to uninstall the U disk before unplug it.

```
root@okmx8mm:~# umount /run/media/sda1
```

3.11 Debug



3.12 SD Card slot



This device does not support NTFS and exFAT format file systems. If you do not know the SD card format, please format it into FAT32 format before use.

The SD card mount directory is /run/media, supports hot swapping, and the terminal will print information about the SD card. Different SD cards may display different information. After the SD card

is inserted into the SD card slot of the device, the system will automatically check and mount the SD card. After the mount is successful, the SD card can be read and written.

Plug in the 32G SD card, after mounting, you can see the device name after the SD card is mounted from the print information. The print information is as follows:

```
root@okmx8mm:/# mmc0: host does not support reading read-only switch, assuming write-enable
mmc0: new high speed SDHC card at address 59b4
mmcblk0: mmc0:59b4 SD32G 29.1 GiB
Mmcblk0: p1
FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
```

/run/media is the mounting directory of the SD card, and view the files in this directory

```
root@okmx8mm:~# ls /run/media                    //list file under /run/media directory
```

The printed information is as follows, mmcblk0p1 is the file name after the SD card is mounted

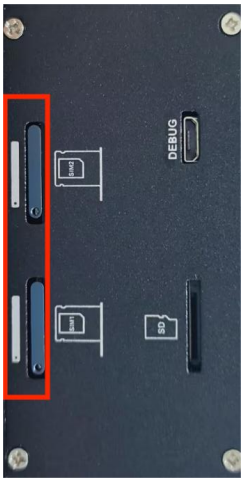
```
mmcblk0p1 mmcblk1p1
```

If you don't need to use the SD card anymore, please use umount to uninstall the SD card before removing the SD card.

```
root@okmx8mm:~# umount /run/media/mmcblk0p1
```

Note: First exiting the SD card mounting path, then insert and remove the SD card.

3.13 SIM Card Slot



When inserting/removing the SIM card, please make sure the device is turned off

Note: Please place the device flat when inserting/removing the SIM card.

3.14 Antenna Interface

The top is the cellular network antenna interface and the bottom is the WiFi antenna interface. Only one of cellular network or WiFi can be selected.

3.15 Reset Button

After the device is running normally, press the reset button and the device will automatically restart.

3.16 HDMI

Maximum support 1080p 60fps. Please connect to HDMI first and then power on.

The first time you use it, please press any button during the countdown firstly.

```
U-Boot 2018.03-00007-g8bc4bc5 (Mar 29 2022 - 03:52:58 +0000)

CPU:   Freescale i.MX8MMQ rev1.0 1600 MHz (running at 1200 MHz)
CPU:   Industrial temperature grade (-40C to 105C) at 53C
Reset cause: POR
Model: FSL i.MX8MM EVK board
DRAM:  2 GiB
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... OK
Display: MIPI7 (1024x600)
Video: 1024x600x24
615478 bytes read in 15 ms (39.1 MiB/s)
Logo:
Image size    : 1024 x 600
Bits per pixel: 8
Compression   : 0
In:    serial
Out:   serial
Err:   serial

 BuildInfo:
  - ATF d6451cc
  - U-Boot 2018.03-00007-g8bc4bc5

switch to partitions #0, OK
mmc1(part 0) is current device
flash target is MMC:1
Net:   eth0: ethernet@30be0000
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  2
```

The following interface appears, press the corresponding number to select the resolution, and then select 0 to restart the device

```
Compression   : 0
In:    serial
Out:   serial
Err:   serial

 BuildInfo:
  - ATF d6451cc
  - U-Boot 2018.03-00007-g8bc4bc5

switch to partitions #0, OK
mmc1(part 0) is current device
flash target is MMC:1
Net:   eth0: ethernet@30be0000
Fastboot: Normal
Normal Boot
Hit any key to stop autoboot:  0
## forlinx params set menu ...
--------------------------
0:reboot
1:exit to shell
2:MIPI7
3:MIPI2HDMI 1080P
4:MIPI2HDMI 720P
5:MIPI2HDMI 480P
6:MIPI2LVDS 1280x800
7:MIPI2HDMI custom
--------------------------
```

4 Software

4.1 Login

Serial Port Login method:

The device can be logged in via micro-USB, and the default system login name is: root. Take SecureCRT as an example, connect the power supply and USB cable of the device, open SecureCRT, and click the quick login button in the upper left corner.

Select Serial protocol, select the port, and set the baud rate to 115200.

Enter the login name root to log in.

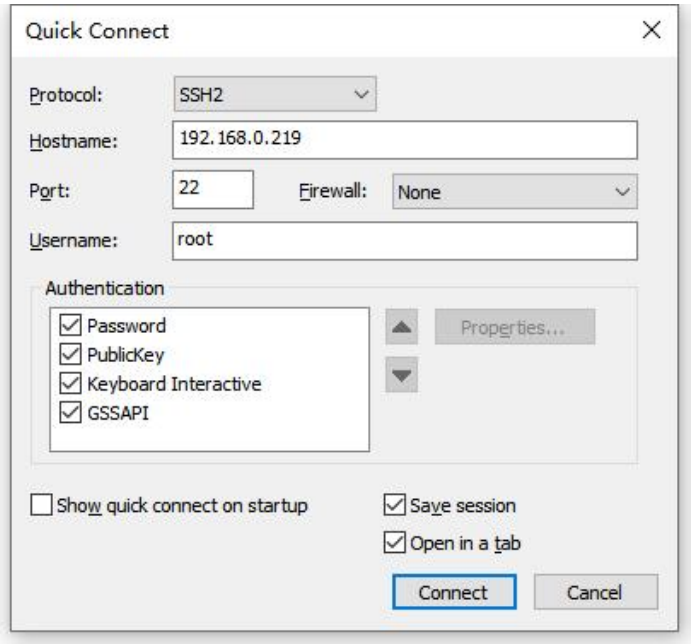
```
NXP i.MX Release Distro 4.14-sumo okmx8mm ttymxc1

okmx8mm login: root
Last login: Thu Jun 20 10:00:15 UTC 2019 on tty7
root@okmx8mm:~#
```

Ethernet Login Method

Make sure the network is working before using this method.

```
root@okmx8mm:~# udhcpc -i eth0
udhcpc: started, v1.27.2
udhcpc: sending discover
udhcpc: sending select for 192.168.0.219
udhcpc: lease of 192.168.0.219 obtained, lease time 43200
/etc/udhcpc.d/50default: Adding DNS 192.168.0.1
root@okmx8mm:~#
```

Hostname fills in the device IP, Username is root.



Click Connect to enter the device.

4.2 Time Setting

By using the date and hwclock tools to set the software and hardware time, test whether the software clock reads the RTC clock synchronously when the board is powered off and on again.

root@okmx8mm:~# date -s "2023-08-05 10:00:00" //**Set software time**

Sat Aug 5 10:00:00 UTC 2023

root@okmx8mm:~# hwclock --show //**Show hardware time**

Fri May 3 17:50:51 2019 0.000000 seconds

```
root@okmx8mm:~# hwclock -w //Synchronize software time to hardware time
```

Power off the board and power it on again. After entering the system, use the command date to read the system time, and the time has been synchronized.

4.3 MCU Frequency Modulation

When the user needs to modify the MCU frequency, BL303/BL304 supports adjusting the MCU frequency by command. All cpufreq governor types supported in the current kernel:

```
root@okmx8mm:~# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
interactive conservative userspace powersave ondemand performance schedutil
```

userspace represents the user mode, in which other user programs are allowed to adjust the CPU frequency. View the frequency gear supported by the current CPU:

```
root@okmx8mm:~# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
1200000 1600000
```

Modify it to user mode, modify the frequency to 1.6GHz:

```
root@okmx8mm:~# echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
root@okmx8mm:~# echo 1600000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
```

Check the current frequency:

```
root@okmx8mm:~# cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq
1600000
```

4.4 Temperature Control

In the default setting of the kernel, the CPU junction temperature, if it exceeds 85 degrees, the CPU will reduce the frequency; if it exceeds 105 degrees, the CPU will restart;
View the current CPU temperature value:

```
root@okmx8mm:~# cat /sys/class/thermal/thermal_zone0/temp
 51890 //temperature is 51.890℃  (51890/1000)
```

View the CPU frequency reduction temperature value in the kernel

```
root@okmx8mm:~# cat /sys/devices/virtual/thermal/thermal_zone0/trip_point_0_temp
```

85000 //temperature is 85℃

View the CPU restart temperature value in the kernel

root@okmx8mm:~# cat /sys/devices/virtual/thermal/thermal_zone0/trip_point_1_temp

105000 //temperature is 105℃

4.5 Wake From Sleep

Set the wake-up source and put the system to sleep:

root@okmx8mm:~# echo enabled > /sys/class/tty/ttymxc1/power/wakeup

root@okmx8mm:~# echo mem > /sys/power/state

[275.147348] PM: suspend entry (deep)

[275.150934] PM: Syncing filesystems ... done.

[275.160355] rtk_btusb: rtkbt_pm_notify: pm_event 3

[275.165233] rtk_btusb: rtkbt_pm_notify: suspend prepare

[275.170546] rtk_btusb: Remote wakeup not support, no needs binding

[275.176785] Freezing user space processes ... (elapsed 0.001 seconds) done.

[275.185298] OOM killer disabled.

[275.188536] Freezing remaining freezable tasks ... (elapsed 0.001 seconds) done.

[275.197334] Suspending console(s) (use no_console_suspend to debug)

[275.205555] rtc-rx8010 1-0032: Frequency stop detected

[275.253984] rtk_btusb: btusb_suspend message.event 0x2, data->suspend_count 0

[275.253988] rtk_btusb: btusb_suspend: hdev is not HCI_RUNNING

[275.253990] rtk_btusb: btusb_suspend: suspending...

[275.918696] PM: suspend devices took 0.712 seconds

[275.922500] Disabling non-boot CPUs ...

[275.935549] CPU1: shutdown

[275.935555] psci: CPU1 killed.

[275.959412] CPU2: shutdown

[275.959417] psci: CPU2 killed.

[275.983234] IRQ 6: no longer affine to CPU3

[275.983352] CPU3: shutdown

[276.002988] psci: Retrying again to check for CPU kill

.....

[277.623078] rtk_btusb: rtkbt_pm_notify: pm_event 4

[277.627965] PM: suspend exit

Tap the command line to wake up.

4.6 Node-Red

Support node-v16.14, node-v19.6.1 and other versions.

4.7 SQLite

Support sqliteV3.11~sqliteV3.40 and other versions.

4.8 Python

Support PythonV3.6~V3.10 and other versions.

4.9 QT

Support qtV5.9 and other versions.

4.10 MySQL

Support MysqlV5.1.51~5.1.73 and other versions.

4.11 Ignition

Support Ignition version V8.1.25.

4.12 Docker

Support DockerV18 and other versions.

4.13 Eclipse

Support eclipse

4.14 Debian

Support DebianV9~V11 and other versions.

4.15 Ubuntu

Support UbuntuV16 and other versions.

5 Firmware update

Please contact BLIIoT if you need to upgrade firmware.

6 Warranty Terms

1) This equipment will be repaired free of charge for any material or quality problems within one year from the date of purchase.
2) This one-year warranty does not cover any product failure caused by man-made damage, improper operation, etc

7 Technical Support

Shenzhen Beilai Technology Co., Ltd
Website: https://www.bliiot.com