ARMxy Embedded Computer

BLIIOT®
MAKE IIOT EASIER

BL370
User Manual

Version: V1.0

Date: 2025-03-17

Shenzhen Beilai Technology Co.,Ltd

Website: https://www.bliiot.com

Preface

Thanks for choosing BLIIoT ARM based Embedded Computer. These operating instructions contain all the information you need for operation of BL370.

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
2025/3/17	V1.0	Initial Release	LKY

Table of Contents

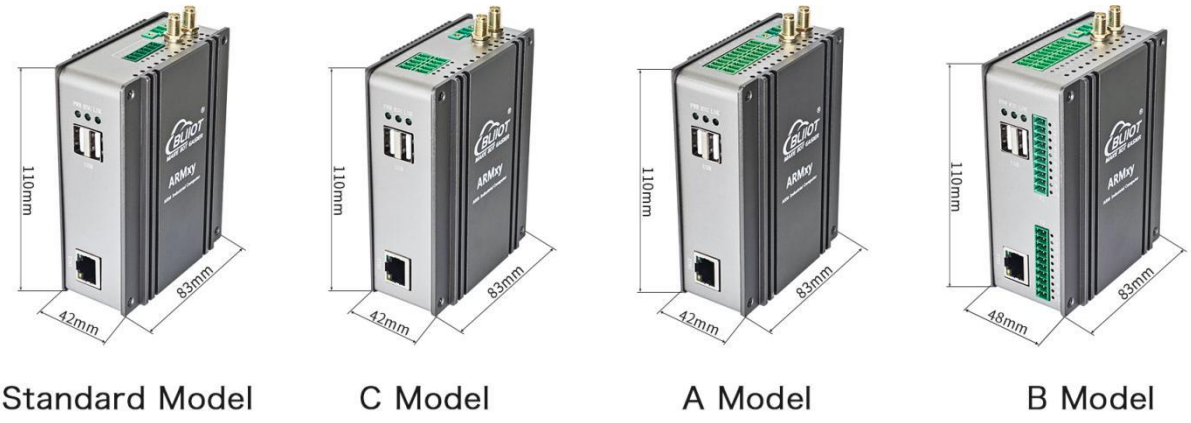
1 Introduction

1.1 Overview

BL370 Series ARM embedded computer from the ARMxy series is an industrial-grade ARM controller with flexible I/O configuration. Based on the Rockchip RK3562/RK3562J processor, it features a quad-core ARM Cortex-A53 + single-core ARM Cortex-M0 architecture, with clock speeds up to 1.8GHz/2.0GHz. It comes with various combinations of 8GB/16GB/32GB eMMC storage and 1GB/2GB/4GB LPDDR4X RAM, supports rich I/O interfaces, and includes a 1TOPS NPU for deep learning. It is suitable for use as a smart gateway, energy storage system EMS/BMS, motion control, edge computing, industrial control, and smart terminals.

BL370 offers abundant interface resources, including three Ethernet ports, two USB ports, multiple RS485 communication interfaces, and optional Wi-Fi or 4G modules. It also features 1 power interface and 1 HDMI video display interface. It can run Linux, Ubuntu, Debian, and is compatible with applications such as Node-Red, Qt, Python, and C++, supporting databases like MySQL, InfluxDB, and SQLite. The combination of versatile hardware interfaces and robust software compatibility makes the BL370 series ideal for a wide range of applications.

1.2 Appearance

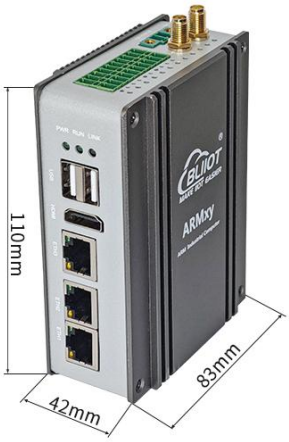


Standard Model

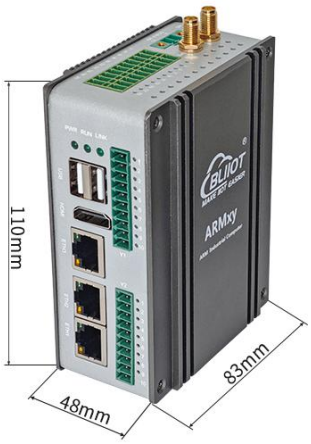
A Model

B Model

Standard Model



A Model



B Model

1.3 Technical Specifications

	Parameter	Description
System	CPU	Rockchip RK3562J/RK3562: 4x ARM Cortex-A53 (64-bit), 1x ARM Cortex-M0
	Clock Speed	RK3562J: Normal mode 1.2GHz, Overdrive mode 1.8GHz

		RK3562: 2.0GHz ARM Cortex-M0: 200MHz
	RAM	1/2/4GByte LPDDR4X
	Storage	8/16/32GByte eMMC
	NPU	1TOPS with support for INT4/INT8/INT16/FP16 Supports deep learning frameworks like TensorFlow, PyTorch, Caffe, and MXNet
	GPU	Mali-G52-2EE GPU, supporting OpenGL ES 1.1/2.0/3.2, OpenCL 2.0, and Vulkan 1.1
Power	Input Voltage	DC 9～36V
	Consumption	Normal: 290mA@12V(with 4G module), 220mA@12V (without 4G module) Maximum: 800mA@12V
	Reverse Polarity	Reverse Polarity Protection
Ethernet	Specification	1~3*RJ45, 3x10/100M, adaptive MDI/MDIX
	Protection	ESD ±2kV (contact), ±8kV (air);
SIM Card	Slot QTY	1
	Type	Drawer interface
USB	QTY	1*micro USB, 2*USB 2.0 HOST
SD Card	QTY	1
	Type	Support SD, SDHC and SDXC(UHS-I) card
HDMI	QTY	1
Antenna	Interface	1*WIFI/4G, 1*GPS antenna
	Type	SMA
X board Serial Port(optional)	QTY	2/4/RS232/RS485
	Baud Rate	300bps-115200bps
	Data Bit	7,8
	Parity Bit	None, Even, Odd
	Stop Bit	1, 2
X board Digital Input	Channel	2/4/8/16
	Input Type	Dry contact or Wet contact
	Dry Contact	Logic 1: Open circuit Logic 0: Short circuit
	Wet Contact	Logic 1: 0-3VDC Logic 0: 10-30VDC
	Isolation protection	2KVrms
X board Digital Output	Channel	2/4/8/16
	Output Type	SINK
	Single Channel Capacity	100mA

4G Module(Optional)	L-E	GSM/EDGE:900,1800MHz WCDMA:B1,B5,B8 FDD-LTE:B1,B3,B5,B7,B8,B20 TDD-LTE:B38,B40,B41
	L-CE	GSM/EDGE:900,1800MHz WCDMA:B1,B8 TD-SCDMA:B34,B39 FDD-LTE:B1,B3,B8 TDD-LTE:B38,B39,B40,B41
	L-A	WCDMA:B2,B4,B5 FDD-LTE:B2,B4,B12
	L-AU	GSM/EDGE:850,900,1800MHz WCDMA:B1,B2,B5,B8 FDD-LTE:B1,B3,B4,B5,B7,B8,B28 TDD-LTE:B40
	L-AF	WCDMA:B2,B4,B5 FDD-LTE:B2,B4,B5,B12,B13,B14,B66,B71
	CAT-1	GSM:900,1800 FDD-LTE:B1,B3,B5,B8 TDD-LTE:B34,B38,B39,B40,B41
5G(Optional)	Redcap version	5G NR：N1/N3/N5/N8/N28/N41/N78/N79 LTE-FDD：B1/B3/B5/B8 LTE-TD：B34/B38/B39/B40/B41
	FG652 version	NR：N1/28/41/78/79 LTE：FDD B1/3/5/8 LTE：TDD B34/38/39/40/41 WCDMA：B1/8
WiFi (Optional)	Interface	PCIE
	Protocol	IEEE 802.11a/b/g/n/ac
	Mode	STA, AP
	Frequency	2.4/5.8GHz
	Speed	433Mbps (Max)
LED	QTY	LED*3
Environment	Working	-40～85°C/0～70°C, 35～75% RH
	Storage	-40 to 85°C, 35 to 75% RH
Others	Housing	Aluminium housing + stainless steel
	Dimensions	110*83*42mm or 110*83*48mm
	Protection Level	IP30
	Installation	DIN35 rail mounted, wall mounting

	System	Linux-5.10.198

1.4 Model Selection

1.4.1 Main Model Selection

Model	ETH	USB	HDMI	X I/O Board	Y I/O Board	Dimension
BL370	1×100M	2	×	1×6PIN	×	42×83×110mm
BL370A	1×100M	2	×	1×20PIN	×	42×83×110mm
BL370B	1×100M	2	×	1×20PIN	2	48×83×110mm
BL370C	1×100M	2	×	1×10PIN	×	42×83×110mm
BL371	2×100M	2	×	1×6PIN	×	42×83×110mm
BL371A	2×100M	2	×	1×20PIN	×	42×83×110mm
BL371B	2×100M	2	×	1×20PIN	2	48×83×110mm
BL372	3×100M	2	1	1×6PIN	×	42×83×110mm
BL372A	3×100M	2	1	1×20PIN	×	42×83×110mm
BL372B	3×100M	2	1	1×20PIN	2	48×83×110mm

1.4.2 SOM Selection

ARMxy BL370 Series SOM Selection

Model	MCU	Clock Speed	Kernel	NPU	eMMC	DDR4	Temperature
SOM370	RK3562J	1.8GHz	4 x A53 +M0	1TOPS	8GByte	1GByte	Industrial grade -40~85℃
SOM371	RK3562J	1.8GHz	4 x A53 +M0	1TOPS	16GByte	2GByte	Industrial grade -40~85℃
SOM372	RK3562J	1.8GHz	4 x A53 +M0	1TOPS	32GByte	4GByte	Industrial grade -40~85℃
SOM373	RK3562	2.0GHz	4 x A53	1TOPS	8GByte	1GByte	Commercial

			+M0				grade 0~70℃
SOM374	RK3562	2.0GHz	4 x A53 +M0	1TOPS	16GByte	2GByte	Commercial grade 0~70℃
SOM375	RK3562	2.0GHz	4 x A53 +M0	1TOPS	32GByte	4GByte	Commercial grade 0~70℃

1.4.3 X Series I/O Board Selection

You can choose the X-series I/O board based on your needs. The number of pins on the X-series I/O board must match the housing.

Note: The default port for this device is RS485. If you need RS232, please specify this to the sales team.

X Series I/O Board Selection

Model	RS232/RS485	CAN	DI	DO	GPIO	PIN
X10	2	×	×	×	×	6PIN
X11	×	2	×	×	×	6PIN
X12	1	1	×	×	×	6PIN
X13	×	×	2	2	×	6PIN
X14	×	×	4	×	×	6PIN
X15	×	×	×	4	×	6PIN
X16	×	×	×	×	4	6PIN
X20	4	×	×	×	×	10PIN
X21	3	1	×	×	×	10PIN
X22	2	2	×	×	×	10PIN
X23	4	×	4	4	×	20PIN
X24	3	1	4	4	×	20PIN
X25	2	2	4	4	×	20PIN
X26	2	×	8	4	×	20PIN

X27	1	1	8	4	×	20PIN
X28	2	×	12	×	×	20PIN
X29	1	1	12	×	×	20PIN
X30	×	×	×	×	16	20PIN

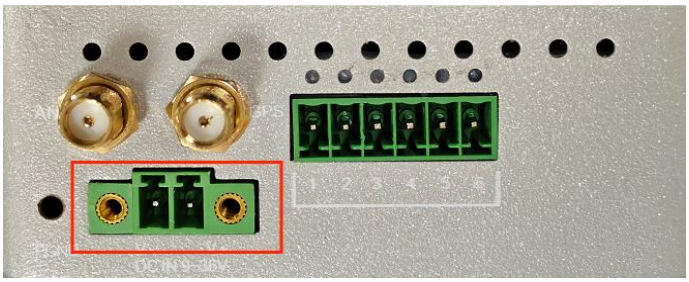
1.4.4 Y Series I/O Board Selection

You can select the Y-series I/O board based on your needs. Y-series I/O modules are compatible with all Y slots. When the Y63 is selected, you can not choose second Y-series IO board.

Model	Description		Model	Description
Y01	4DI+4DO, NPN		Y41	4AO, 0/4~20mA
Y02	4DI+4DO, PNP		Y43	4AO, 0~5/10V
Y11	8DI, NPN		Y46	4AO, ±5V/±10V
Y12	8DI, PNP		Y51	2RTD, 3-Wire PT100
Y21	8DO, PNP		Y52	2RTD, 3-Wire PT1000
Y22	8DO, NPN		Y53	2RTD, 4-Wire PT100
Y24	4DO, Relay		Y54	2RTD, 4-Wire PT1000
Y31	4AI, single-ended, 0/4~20mA		Y56	Resistance measurement
Y33	4AI, single-ended, 0~5/10V		Y57	Voltage measurement
Y34	4AI, differential, 0~5/10V		Y58	4TC
Y36	4AI, differential, ±5V/±10V		Y63	4 RS485 or RS232
Y37	4 IEPE Measurement		Y95	4 PWM Output + 4 Pulse Counter (1 High-Speed, 3 Low-Speed), NPN
/	/		Y96	4 PWM Output + 4 Pulse Counter (1 High-Speed, 3 Low-Speed), PNP

2 Hardware

2.1 Power Interface



Supports 1CH DC9~36V input, with reverse polarity protection.

2.2 I/O Module Port Description

Different X/Y boards offer various serial port options. The currently available board types are as follows.

2.2.1 RS232&RS485 Module

X10 Module

1	2	3	4	5	6
ttyS2-A	ttyS2-B	GND	ttyS3-A	ttyS3-B	GND

Y63 Module

1	2	3	4	5	6	7	8	9	10
ttyWCH0-A	ttyWCH0-B	ttyWCH1-A	ttyWCH1-B	GND	GND	ttyWCH2-A	ttyWCH2-B	ttyWCH3-A	ttyWCH3-B

2.2.2 DI Module

X14 Module

1	2	3	4	5	6
DI1	DI2	GND	DI3	DI4	COM

Y11/12 Module

1	2	3	4	5	6	7	8	9	10

DI1	DI2	DI3	DI4	COM	COM	DI5	DI6	DI7	DI8

2.2.3 DO Module

X15 Module

1	2	3	4	5	6
DO1	DO2	GND	DO3	DO4	GND

Y21/22 Module

1	2	3	4	5	6	7	8	9	10
DO1	DO2	DO3	DO4	GND	GND	DO5	DO6	DO7	DO8

2.2.4 AO Module

Y41 Module

1	2	3	4	5	6	7	8	9	10
AO1	GND	AO2	GND	/	/	AO3	GND	AO4	GND

Note: Each AO module requires a fixed current of approximately 100mA.

2.2.5 AI Module

Y31/Y34 Module

1	2	3	4	5	6	7	8	9	10
AI1	GND	AI2	GND	/	/	AI3	GND	AI4	GND

2.2.6 RTD Module

Y51 Module

1	2	3	4	5	6	7	8	9	10
/	PT2+	PT2-	PT2-	/	/	/	PT1+	PT1-	PT1-

2.2.7 Multiple Port Combinations

X23 Module

1	3	5	7	9	11	13	15	17	19
DI4	DI3	DI2	DI1	GND	COM	ttyS9-A	ttyS4-A	ttyS3-A	ttyS2-A

DO4	DO3	DO2	DO1	POWER	GND	ttyS9-B	ttyS4-B	ttyS3-B	ttyS2-B
2	4	6	8	10	12	14	16	18	20

X25 Module

1	3	5	7	9	11	13	15	17	19
DI4	DI3	DI2	DI1	GND	COM	ttyS9-A	ttyS4-A	can1-H	can0-H
DO4	DO3	DO2	DO1	POWER	GND	ttyS9-B	ttyS4-B	can1-L	can0-L
2	4	6	8	10	12	14	16	18	20

X26 Module

1	3	5	7	9	11	13	15	17	19
DI6	DI5	DI4	DI3	GND	COM	DI2	DI1	ttyS3-A	ttyS2-A
DO4	DO3	DO2	DO1	POWER	GND	DI8	DI7	ttyS3-B	ttyS2-B
2	4	6	8	10	12	14	16	18	20

X27 Module

1	3	5	7	9	11	13	15	17	19
DI6	DI5	DI4	DI3	GND	COM	DI2	DI1	ttyS3-A	can0-H
DO4	DO3	DO2	DO1	POWER	GND	DI8	DI7	ttyS3-B	can0-L
2	4	6	8	10	12	14	16	18	20

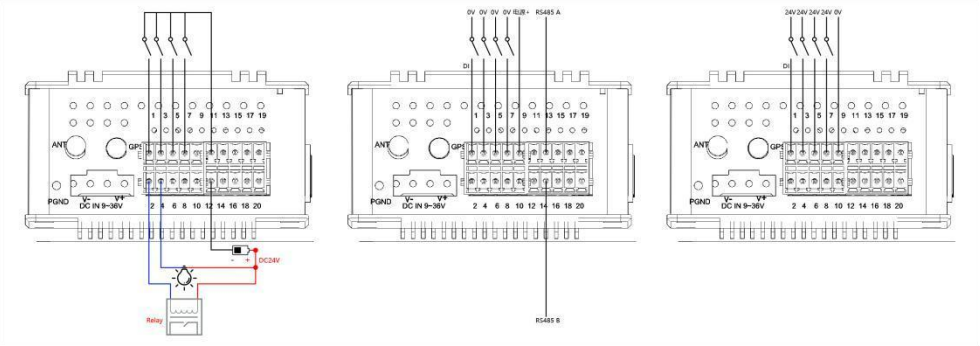
X28 Module

1	3	5	7	9	11	13	15	17	19
DI6	DI5	DI4	DI3	GND	COM	DI2	DI1	ttyS3-A	ttyS2-A
DI12	DI11	DI10	DI9	POWER	GND	DI8	DI7	ttyS3-B	ttyS2-B
2	4	6	8	10	12	14	16	18	20

The COM port is used for dry contact DI, while the GND is used for wet contact DI.

2.2.8 X board Serial Port Usage

Taking X23 as an example, the wiring example is as follows.



10: The POWER port can be connected to the positive power supply to prevent reverse transient current from damaging the circuit when the DO is disconnected.

Enter the /sys/class/beilai/ directory and type ls to view the controllable I/O.

```
root@bliiot:/sys/class/beilai# ls
DI1    DI2    DI3    DI4    DO1    DO2    DO3    DO4
```

Taking the DI1 (dry contact) in the X22 module as an example, the DI status is as follows:

1 means open (0~3V),

0 means close (10~30V).

```
root@bliiot:/sys/class/beilai/DI1# ls
cfg    data    device    drv    power    pull    subsystem    ueven
```

By checking the value of the data, it can be determined that the DI1 status is open.

```
root@bliiot:/sys/class/beilai/DI1# cat data
1
```

For DO, the state of the DO can be changed by modifying the value of the data. For example, when writing 0, DO1 will be connected to GND.

```
root@bliiot:/sys/class/beilai/DO1# echo    0    > data
```

For CAN port transmission, taking CAN0 as an example, configure the CAN bus bitrate to 1Mbps and then start the CAN bus.

```
root@bliiot:~# ifconfig can0 down
root@bliiot:~# ip link set can0 type can bitrate 1000000
root@bliiot:~# ifconfig can0 up
```

To send data via CAN0

```
root@bliiot:~# cansend can0 123#1122334455667788
```

CAN0 is waiting to receive data.

```
root@bliiot:~# candump can0
```

If CAN0 is receiving normally, it should receive the data when sent from another device.

```
root@bliiot:~# candump can0
  can0    123      [8]    11 22 33 44 55 66 77 88
```

For RS232/485 transmission, when using the RS232/485 serial port, connect the RS232/485 cable to the port, such as the RS485-1 port, whose device file is /dev/ttyS2. Set the baud rate to 115200, 8N1, and no parity bit.

```
stty -F /dev/ttyS2 ispeed 115200 ospeed 115200 cs8
echo 12345 > /dev/ttyS2          //To send data through the RS485-1 port
cat /dev/ttyS2                   //To wait and view the received data
```

Press "Ctrl+C" to stop.

2.2.9 Y Board Serial Port Usage

(1) Software Installation

The corresponding file location is /io. The actual name should be verified based on the file. Execute chmod +x iolib_v1.1.0_install.bin on BEILAI_IOy_ARMV8_V1.0.bin. then install the software.

```
root@bliiot:/# chmod +x BEILAI_IOy_ARMV8_V1.0.bin
root@bliiot:/# ./BEILAI_IOy_ARMV8_V1.0.bin
Md5 verify pass!
tar: ./iolib: time stamp 2024-05-29 09:16:25 is 1716973791.042311023 s in the future
tar: ./ioy: time stamp 2024-06-25 03:05:22 is 1719284328.036465231 s in the future
tar: ./S90iolib: time stamp 2024-05-27 07:47:16 is 1716795642.036139981 s in the future
Install complete!
Restarting iolib:
Stopping iolib: stopped iolib (pid 1568)
OK
Starting iolib: OK
```

(2) Port Usage

Type ioy help to see command help:

The corresponding file location is under the /io folder. Please refer to the file name for the actual name.

Use the command ioy show to view the IO board information.

Use the command ioy help to view the command help.

```
Usage: ioy <command> [<arguments>]

Commands:
        show
        get             <address>|<slot>.<channel>
        set             <address>|<slot>.<channel> <value>
        config          <address>|<slot>.<channel> mode <mode>,
                        <address>|<slot>.<channel> min <min-value> max <max-value>

config mode:
        ai|ao           4t20(4~20mA),0t20(0~20mA),0t5(0~5V),0t10(0~10V),
                        -5t5(-5~5V),-10t10(-10~10V)
        rtd             pt100-3(pt100 3wire),pt100-4(pt100 4wire),
                        pt1000-3(pt1000 3wire),pt1000-4(pt1000 4wire)
        tc              k,j,e,t,s,r,b,n
```

Taking the DI module as an example. Short-circuit DI2, then input ioy show to view the information.

Distinguish the Y1 and Y2 positions based on the slot.

slot	name	channel	address	mode	value	min	max
2	Y12	1	2000	*	0	0.0000	0.0000
2	Y12	2	2001	*	1	0.0000	0.0000
2	Y12	3	2002	*	0	0.0000	0.0000
2	Y12	4	2003	*	0	0.0000	0.0000
2	Y12	5	2004	*	0	0.0000	0.0000
2	Y12	6	2005	*	0	0.0000	0.0000
2	Y12	7	2006	*	0	0.0000	0.0000
2	Y12	8	2007	*	0	0.0000	0.0000

You can also use the get command to retrieve the channel value.

```
root@bliiot:/# ioy get 2004      //View by address
address 2004 value 1
root@bliiot:/# ioy get 2.5       //View by <slot>.<channel>.
slot 2 channel 5 value 1
```

Taking the AO module as an example, enter the ioy show command to view the information.

slot	name	channel	address	mode	value	min	max
2	Y41	1	4000	4t20	4.0000	4.0000	20.0000
2	Y41	2	4002	4t20	4.0000	4.0000	20.0000
2	Y41	3	4004	4t20	4.0000	4.0000	20.0000
2	Y41	4	4006	4t20	4.0000	4.0000	20.0000
2	Y41	5	4008	4t20	4.0000	4.0000	20.0000
2	Y41	6	4010	4t20	4.0000	4.0000	20.0000
2	Y41	7	4012	4t20	4.0000	4.0000	20.0000
2	Y41	8	4014	4t20	4.0000	4.0000	20.0000

The mode type is displayed as 4t20, which corresponds to the 4~20mA current output in the config mode of ioy help.

Use the set command to set the channel value:

```
root@bliiot:/# ioy set 4000 10      //Output 10mA by address setting
root@bliiot:/# ioy set 2.1 10       //Setting via <slot>. <channel>
root@bliiot:~# ioy get 4000
address 4000 value 10.000000
```

2.3 LED

LED	Description
PWR	Power LED: It remains constantly on when the power is connected. This LED light cannot be programmed by the user.
RUN	Default Settings: The LED blinks when the CPU usage is below 90% and remains on continuously when the CPU usage exceeds 90% This LED light can be programmed by the user.
LINK	Default Settings: The LED remains on when there is an internet connection and turns off when there is no internet connection. This LED light can be programmed by the user.

The LED indicators are as follows:

LED2 (POWER): Stays on when the power is normal after startup.

LED1 (RUN): Blinks when the system is running normally.

LED0 (LINK): Remains on when connected to the internet via a wired network; blinks when using 4G or WiFi.

The configuration file for these settings is /etc/beilai_led.sh

To view the trigger conditions: cat /sys/class/leds/user-led1/trigger

```
root@bliiot:~# cat /sys/class/leds/user-led1/trigger
[none] rfkill-any rfkill-none kbd-scrolllock kbd-numlock kbd-capslock kbd-kanalock kbd-shiftlock kbd-altgrlock kbd-ctrllock kbd-altlock kbd-shiftllock kbd-shiftrlock kbd-ctrllock kbd-ctrlrlock mmc1 disk-activity disk-read disk-write ide-disk heartbeat mmc0
```

Where [none] means the current trigger condition of led0 is none. Write the above string to trigger to modify the trigger condition.

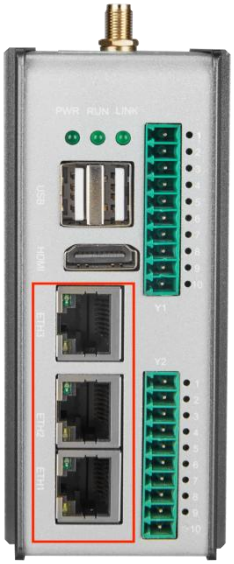
When the led trigger condition is set to none, the user can control the led light on or off by commands

To control LED1 to be on: echo 1 >/sys/class/leds/user-led1/brightness

```
root@bliiot:~# echo none >/sys/class/leds/user-led1/brightness
root@bliiot:~# echo 1 >/sys/class/leds/user-led1/brightness
```

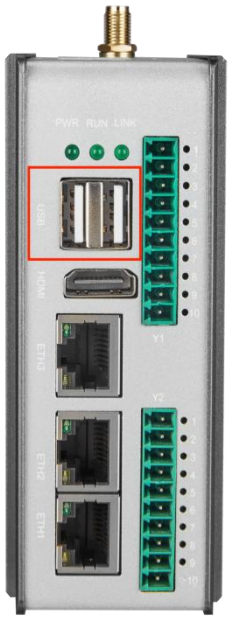
To control LED1 to be off: echo 0 >/sys/class/leds/user-led1/brightness

2.4 Ethernet Port



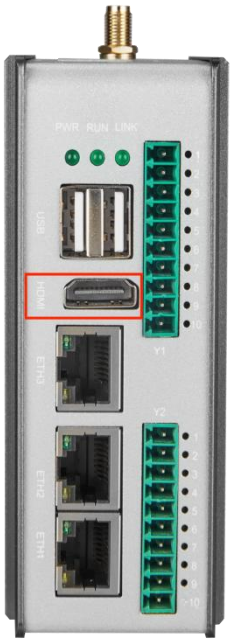
The device is equipped with 3x100M Ethernet ports

2.5 USB Port



The device has 2 USB 2.0 HOST interfaces, supporting FAT32 formatted USB drives. When reading or writing data to the USB drive, use the sync command to ensure data is properly saved and prevent data loss.

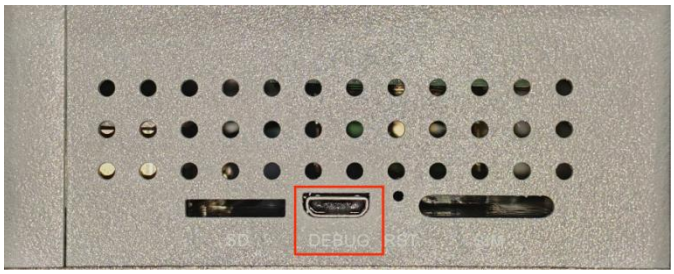
2.6 HDMI



The HDMI as shown in the image, HDMI 1.4 and HDMI 2.0 standards are supported.

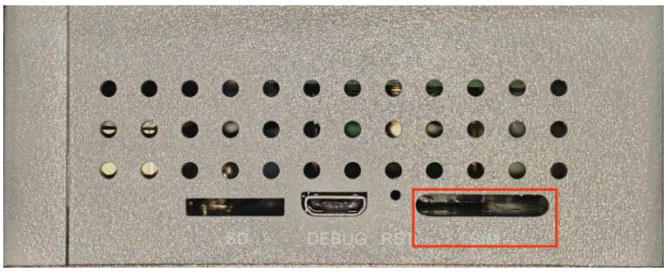
The maximum supported resolution of the system is 4K@30fps.

2.7 Debugging Serial Port

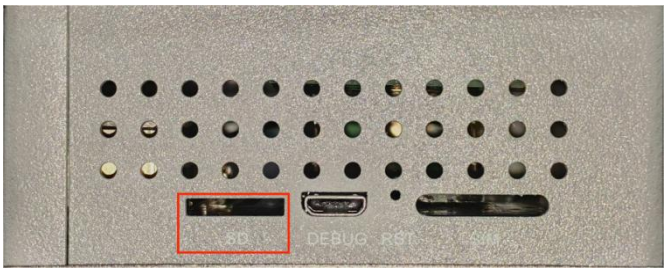


The debugging interface is as shown in the image. You can access the device's system through this port.

2.8 SIM Card Slot

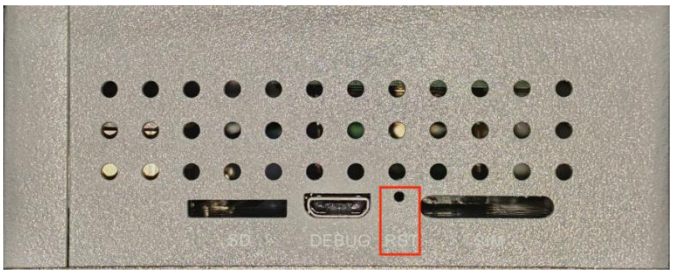


2.9 SD Card Slot



The SD card slot, as shown in the image, supports FAT32 formatted SD cards. After reading or writing data using this slot, use the sync command to ensure data is properly saved and prevent data loss.

2.10 Reset Button



Press the restart button and release it to reboot the device.

2.11 PCIE

The PCIe interface supports both 4G and WiFi.

2.11.1 4G Module

Using the Quectel EC20 module as an example, place the SIM card into the module and connect the

antenna. The test program can be found in the /usr/demo/4G directory.

(1) Network Function

Disable other network connections and keep only the 4G module network active.

```
ifconfig eth1 down
ifconfig eth2 down
ifconfig eth3 down
udhcpc -i usb0
ifconfig
```

At this point, a network node usb0 should be generated. If this node is not present, the module may not have network functionality enabled by default. Try configuring the 4G module with the following commands:

The EC200 series module uses /dev/ttyUSB1.

```
echo -e "AT+QCFG=\"USBNET\",1\r" > /dev/ttyUSB2
```

If using the EC200 module, an additional command may be required to establish a network connection.

```
echo -e "AT+QNETDEVCTL=3,1,1\r" > /dev/ttyUSB1
```

After executing the command, if the device returns “OK”, it indicates that the configuration is successful and this setup needs to be done only once. After rebooting the device, the usb0 node should be created. Then, you can re-run the network disable and enable commands.

Once the usb0 node is generated, use the following commands to test the network.

```
ping www.baidu.com -I usb0
```

(2) SMS Function

To test the SMS functionality, execute the test command in the program directory.

```
./send_sms <device> <phonenumber> <text>
```

Command Description:

<device>: The 4G module device node.

<phonenumber>: The target phone number for sending the SMS.

<text>: The content of the SMS. There should be no spaces between characters in the SMS content; otherwise, an error will be prompted.

For example: ./send_sms /dev/ttyUSB2 152******** test

At this time, the corresponding number should receive a text message with the content of "test".

(3) Call Function

To test the dialing function, execute the test command in the program directory:

```
./phone_call <device> <phonenumber>
```

Command Description: <device> is the 4G module device node. <phonenumber> is the dialling target mobile phone number.

For example: ./phone_call /dev/ttyUSB2 152********

At this point the corresponding number should receive an incoming call from the device.

(4) GPS Function

The GPS function can be tested by executing the test command in the test program directory:

```
./get_location <device> <timeout>
```

Command Description:

<device>: The device node, determined by the result of the command ls /dev/ttyUSB*. This may change after rebooting the device.

<timeout>: The time to wait for the return of latitude and longitude information (in seconds).

For example: ./get_location /dev/ttyUSB2 1

Obtaining latitude and longitude information may take a few minutes. If the retrieval fails or times out, check if the antenna is properly connected and ensure you are testing in an open area.

2.11.2 WiFi Module

The test program and driver are located under the /usr/demo/wifi path. Please ensure the antenna is properly connected.

(1) STA Function

Enter the test program directory, disable other networks, keep only the WiFi network, and load the WiFi driver.

```
ifconfig eth1 down
ifconfig eth2 down
ifconfig eth3 down
insmod -f 8188eu.ko                    //Load WiFi driver
```

```
ifconfig wlan0 up                    //Based on the name of the network card shown in ifconfig
```

Execute the following command to connect the device to the specified WiFi network. Use -i followed by the WiFi name and -p followed by the WiFi password.

```
./wifi_setup.sh -i bliiot -p bebetter
```

You can check the obtained IP address using ifconfig. Then, execute the following command to test if the network functionality is working correctly.

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

(2) AP Function

After restarting the system, enter the directory where the test program is located, disable other networks, keep only the WiFi network, and load the WiFi driver.

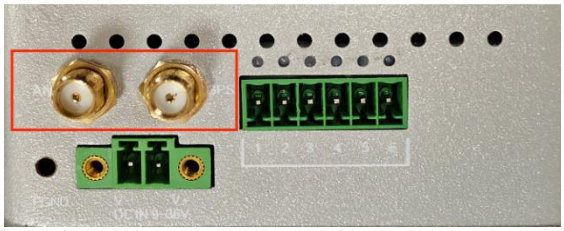
```
ifconfig eth1 down
ifconfig eth2 down
ifconfig eth3 down
insmod -f 8188eu.ko
Ifconfig wlan0 up
```

Execute the following command to set the WiFi module to AP mode:

```
./ap_setup.sh
```

The default WiFi name is rtl8188eu and the password is 88888888. You can modify these settings in the rtl_hostapd_2G.conf configuration file.

2.12 Antenna Interface



The antenna interface includes one WiFi/mobility network antenna interface and one GPS antenna interface.

2.13 Hardware Watchdog

Note: The hardware watchdog is disabled by default.

Watchdog control pin: pin144 (disables the hardware watchdog when set to 1).

Feed pin: pin15.

The hardware watchdog timeout is 30 ms.

2.14 Encryption Chip

The encryption chip model RJGT102 is based on the SHA-256 encryption and authentication algorithm. It also provides a configurable watchdog timer and external reset functionality. The chip communicates with the MCU through an I²C-5 serial interface and supports a low-power mode.
The device uses the encryption chip's demo by writing /proc/sys/kernel/random/uuid to the encryption chip and saving the UUID to /usr/rjgt_unique.json. When used, the data from the encryption chip is compared with the external data. If the external data matches the internal data on the encryption chip, the encryption validation is successful.

Please modify the cross-compiler path in the Makefile and then run make to compile. Alternatively, refer to the "RJGT102 Data Manual" for more details.

Run the sample programme rigt102 and if the uuid is correct, the following reply will appear.

```
root@bliiot: ./rigt102
open unique file failed, create unique file!
random uuid would write rjgt102 : b6275e22-4928-4828-88fb-54a6fd8!
Contrast success
root@bliiot:./rigt102
Contrast success
```

2.15 External RTC

This device includes an external RTC clock.
To view the external RTC device node:

```
root@bliiot:~# ls /dev/rtc*
/dev/rtc     /dev/rtc0
root@bliiot:~# dmesg | grep rtc0
[       4.319167] rtc-isl1208 5-006f: rtc core: registered rtc-isl1208 as rtc0
```

To view the system clock:

```
root@bliiot:~# date
Tue Jul 05 01:22:15 UTC 2024
```

To set the system time:

```
root@bliiot:~# date -s "2024-7-05 09:24:00" && hwclock -w -f /dev/rtc0
root@bliiot:~# hwclock -f /dev/rtc0
```

To synchronize the system clock to the RTC:

```
root@bliiot:~# hwclock --systohc -u
root@bliiot:~# hwclock -u
```

To synchronize both the system clock and the RTC

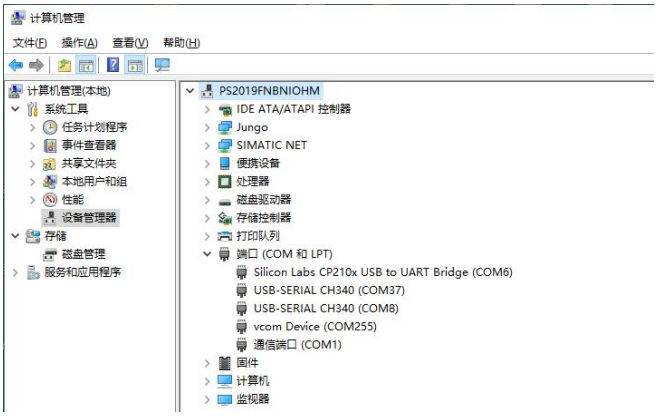
```
root@bliiot:~# hwclock --hctosys -u
```

After executing the command, the system will synchronize the RTC clock to be the system clock.

3 Device Login

3.1 USB Login

To access this on a computer, navigate to "This PC" → "Manage" → "Device Manager". Open the Ports section, then insert the USB cable. The refreshed port indicates the connected device port.



Here's an example using SecureCRT:

1. Open SecureCRT and create a new connection.
2. Choose "Serial" for the connection type.
3. Select the corresponding port.
4. Set the following parameters:
    1. Baud rate: 115200
    2. Data bits: 8
    3. Parity: None

4. Stop bits: 1

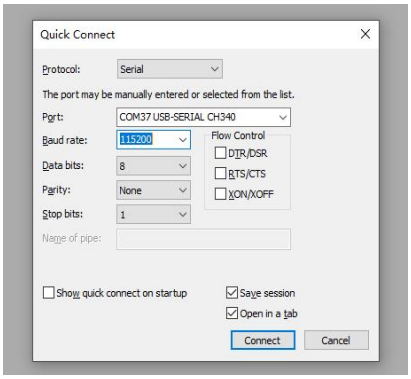
5. Click "Connect" to access the device.

Linux systems do not have a default login password set.

For Ubuntu systems:

Default Login Account: root

Password: root

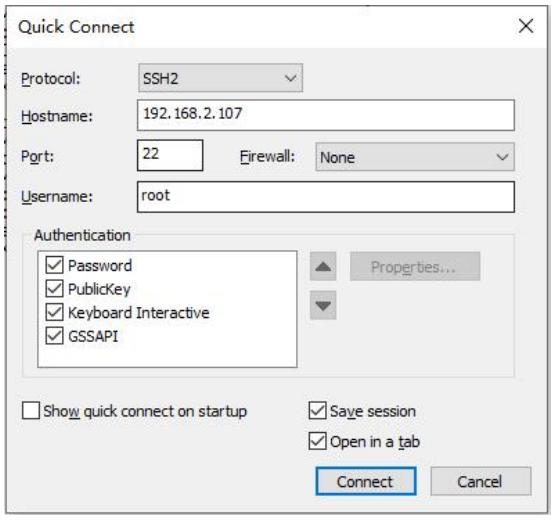


3.2 SSH Login

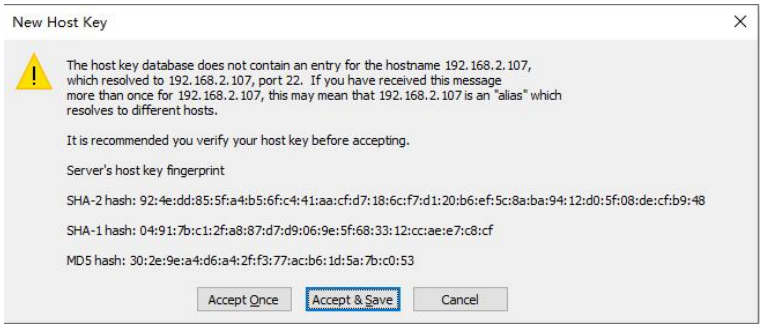
Before logging in via the network port, you need to set the IP address for the corresponding port. For example, ETH2 is connected to the router, and the obtained IP address is 192.168.2.107. The computer IP is on network segment 2.

```
docker0   Link encap:Ethernet  HWaddr 02:42:3D:E2:6F:88
          inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth2      Link encap:Ethernet  HWaddr 00:E0:99:CD:55:B9
          inet addr:192.168.2.107  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fd5f:4184:3ad4:4:66ee:75b9:2b9:476d/64 Scope:Global
          inet6 addr: fe80::5d2c:48eb:826c:7f6/64 Scope:Link
          inet6 addr: fd5f:4184:3ad4:4::74e/128 Scope:Global
          inet6 addr: fd2f:fd7:7cda::74e/128 Scope:Global
          inet6 addr: fd2f:fd7:7cda:0:15d3:ffef:f05a:3ebf/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:240 errors:0 dropped:18 overruns:0 frame:0
          TX packets:80 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24541 (23.9 KiB)  TX bytes:8407 (8.2 KiB)

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:146 errors:0 dropped:0 overruns:0 frame:0
          TX packets:146 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:10796 (10.5 KiB)  TX bytes:10796 (10.5 KiB)
```

Click Create Connection, select the protocol as SSH2, enter the hostname as the device IP: 192.168.2.107, set the port to 22, and use the username root. Then, click Connect to establish the connection.



Select Accept for a successful connection.



4 System Programming

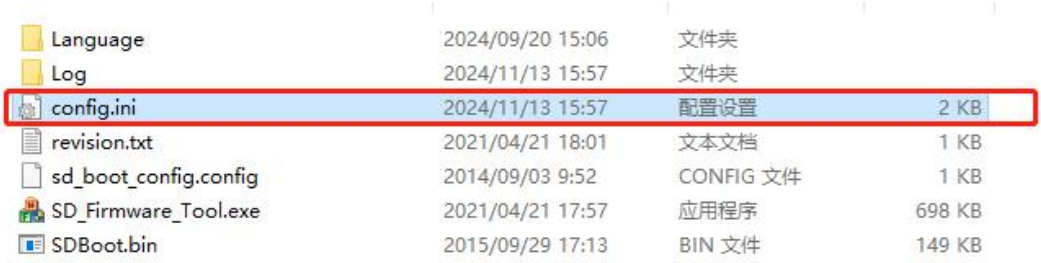
The baud rate of the debugging serial port during programming is 115200.

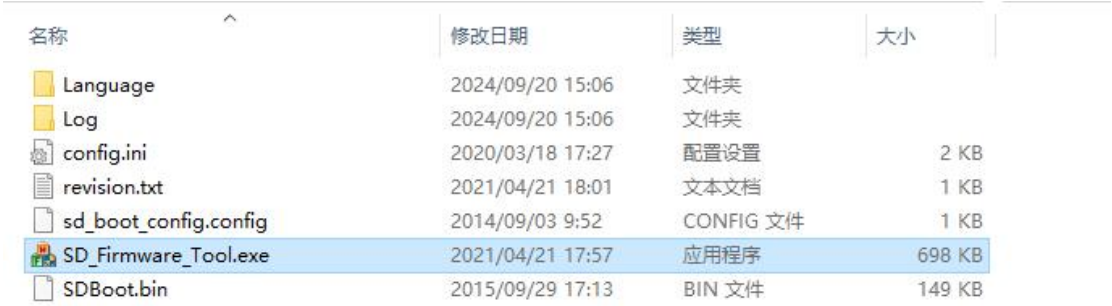
4.1 Micro SD Card Boot

4.1.1 Boot Card Creation

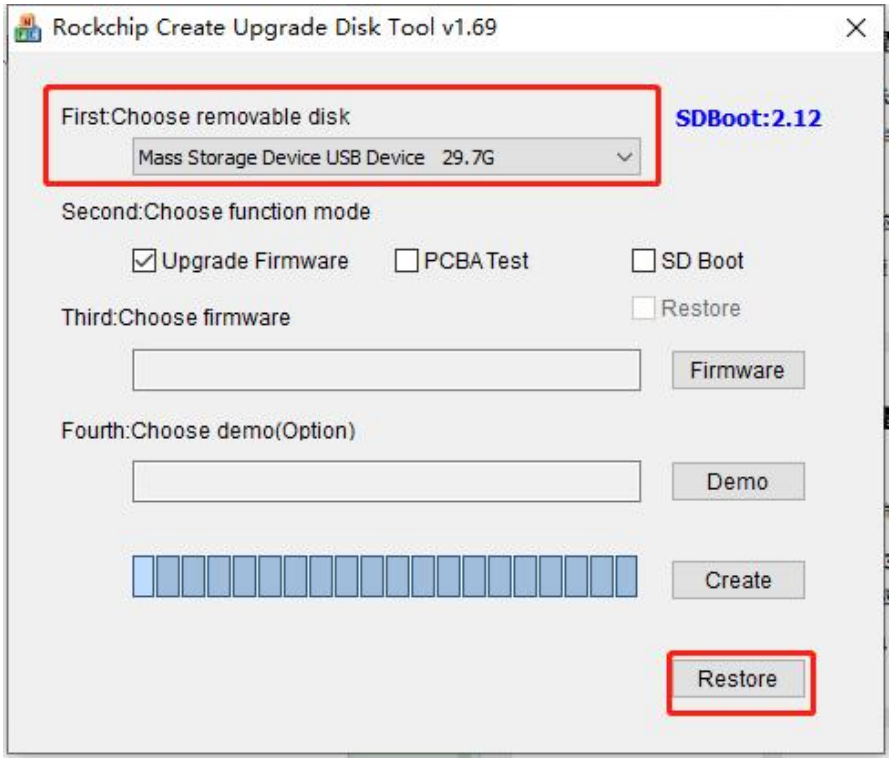
Connect the blank Micro SD card to your computer, open the “SDDiskTool_v1.69” folder, right-click on "SD_Firmware_Tool.exe," and select "Run as administrator (A)."

Switch the language to English before running the software. Open the config.ini file. The setting Selected=2 means the language switched to English.

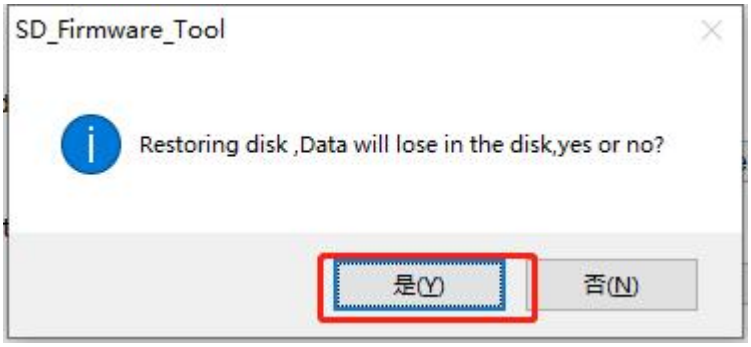


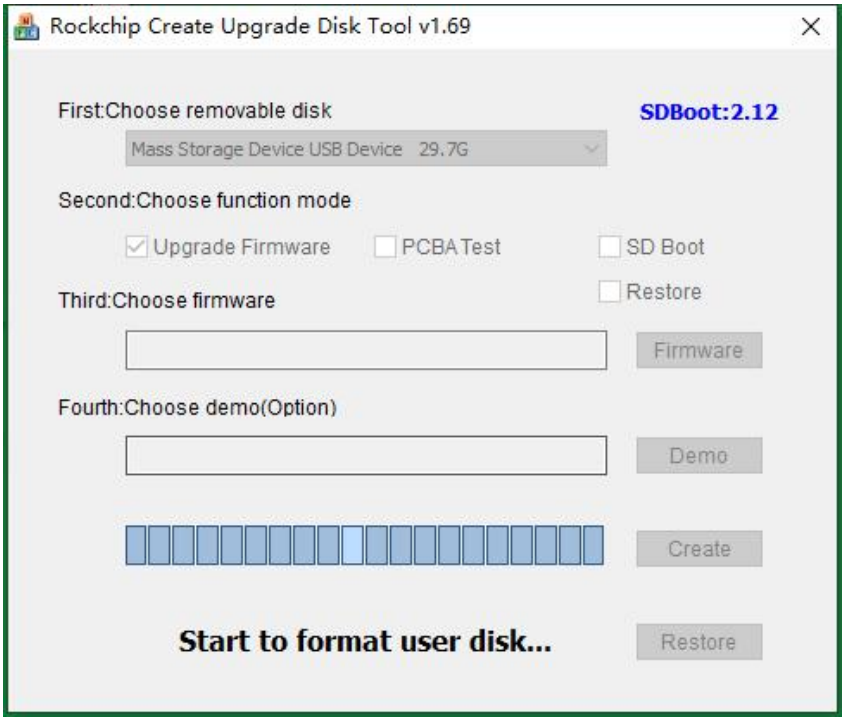


In "First: Choose Removable Disk," choose the removable disk device, then click "Restore" to format it, as shown in the image below.

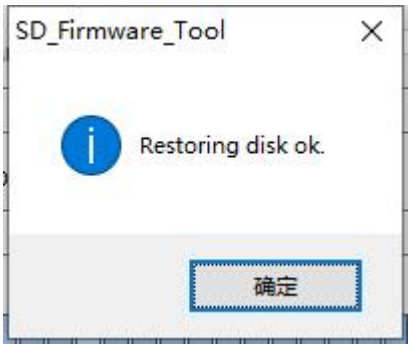


Please confirm that the selected removable disk device is correct, then click “Yes (Y)” in the pop-up window to proceed with formatting.

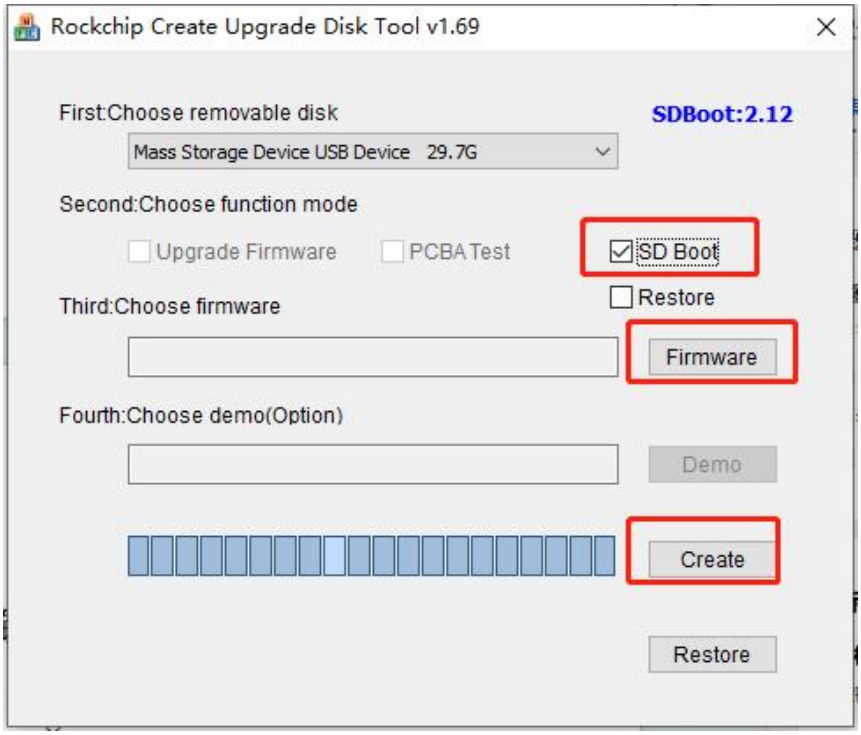


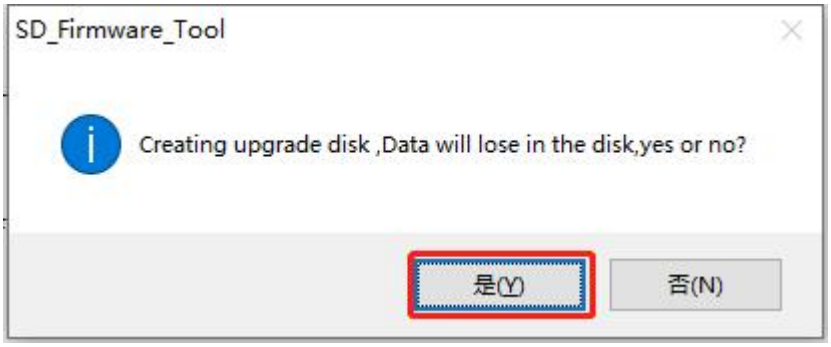


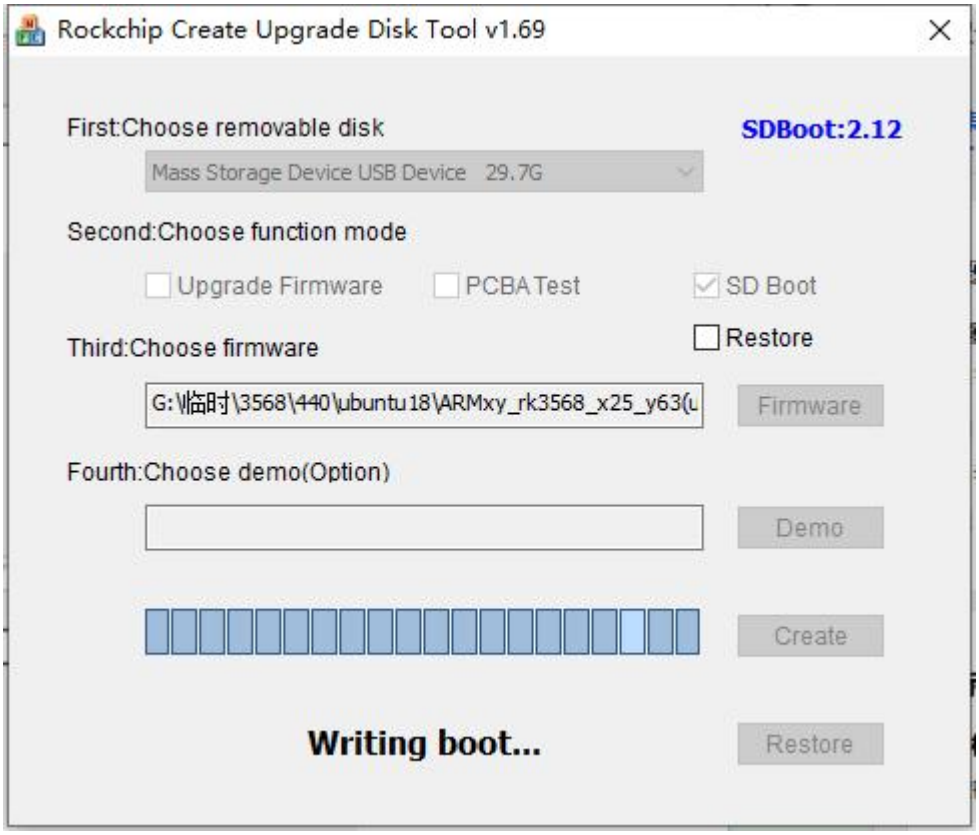
After the formatting is complete, click “OK” in the pop-up window.



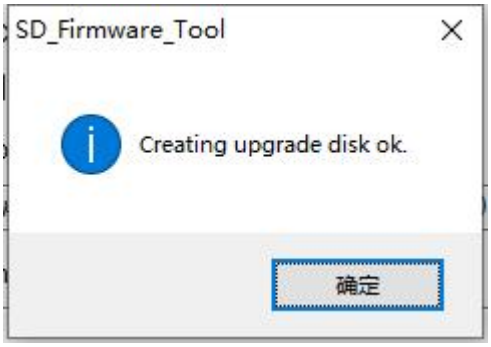
Check the "SD Boot" option, click "Choose Firmware" to choose the target system image file, then click "Start Create." In the pop-up window, click "Yes (Y)" to create the SD boot card.


Rockchip Create Upgrade Disk Tool v1.69
First:Choose removable disk
SDBoot:2.12
Mass Storage Device USB Device 29.7G
Second:Choose function mode
Upgrade Firmware
PCBA Test
SD Boot
Restore
Third:Choose firmware
Firmware
Fourth:Choose demo(Option)
Demo
Create
Restore


SD_Firmware_Tool
Creating upgrade disk ,Data will lose in the disk,yes or no?
是(Y)
否(N)



In the pop-up window, click "OK." The SD boot card create is now complete.



4.1.2 Boot from the Boot Card

After inserting the boot card into the device's Micro SD card slot and powering on the device, the system will automatically boot from the boot card and log in as the root user. The serial debug terminal will print similar startup information.

"Bootdev(atags): mmc 1" indicates that the system is booting from the Micro SD card.

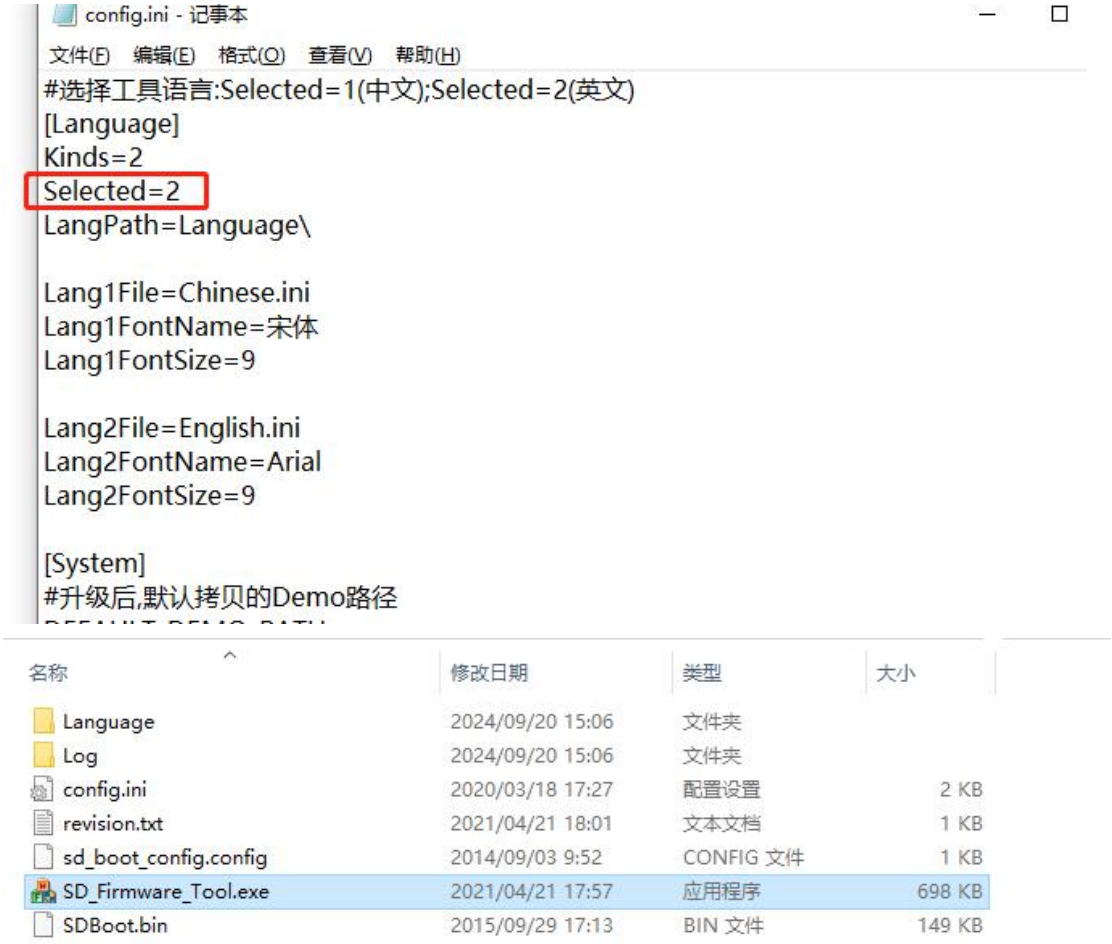
```
U-Boot 2017.09-g54ac980 #root (Sep 18 2024 - 10:27:19 +0800)

Model: Rockchip RK3568 Evaluation Board
MPIDR: 0x81000000
PreSerial: 2, raw, 0xfe660000
DRAM:  2 GiB
Sysmem: init
Relocation Offset: 7d21c000
Relocation fdt: 7b7f8d68 - 7b7fece0
CR: M/C/I
DM: v1
no mmc device at slot 1
dwmmc@fe2b0000: 1 (SD), dwmmc@fe2c0000: 2, sdhci@fe310000: 0
Bootdev(atags): mmc 1
MMC1: Legacy, 52MHz
PartType: EFI
boot mode: None
RESC: 'boot', blk@0x00014884
resource: sha256+
FIT: no signed, no conf required
DTB: rk-kernel.dtb
HASH(c): OK
I2c0 speed: 100000Hz
PMIC:  RK8090 (on=0x40, off=0x00)
vdd_logic init 900000 uV
vdd_gpu init 900000 uV
vdd_npu init 900000 uV
io-domain: OK
INFO:    ddr dmc_fsp already initialized in loader.
```

4.2 EMMC Boot

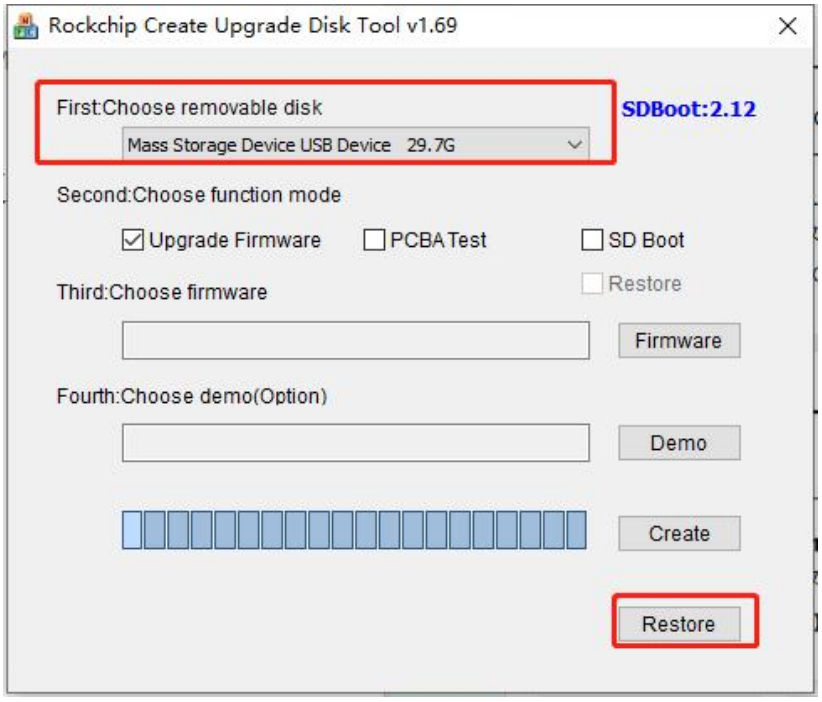
4.2.1 Programming Card Creation

Connect the blank Micro SD card to your computer, open the “SDDiskTool_v1.69” folder, right-click on "SD_Firmware_Tool.exe," and select "Run as Administrator (A)."

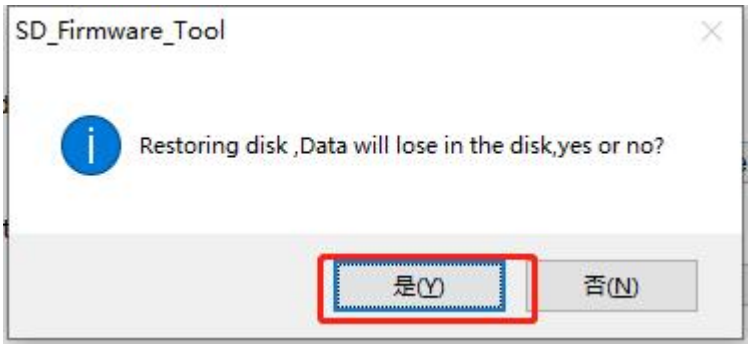
Switch the language to English before running the software. Open the config.ini file. The setting Selected=2 means the language switched to English.

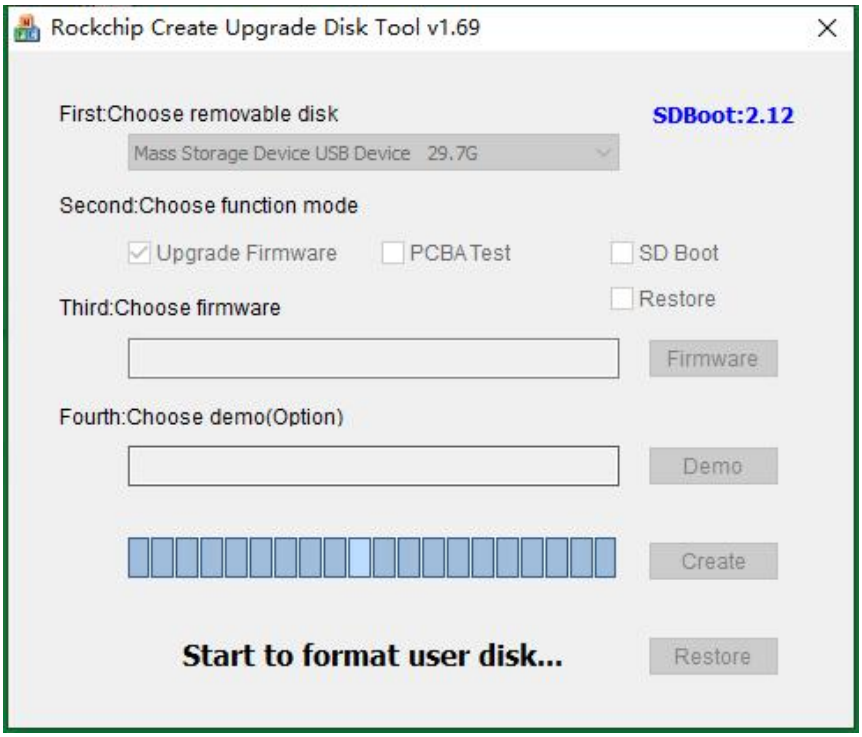


In "First: Choose Removable Disk," choose the removable disk device, then click "Restore" to format it, as shown in the image below.

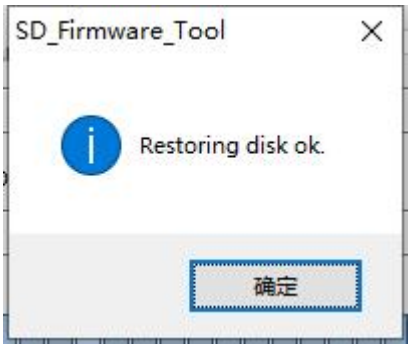


Please confirm that the selected removable disk device is correct, then click "Yes (Y)" in the pop-up window to proceed with formatting.

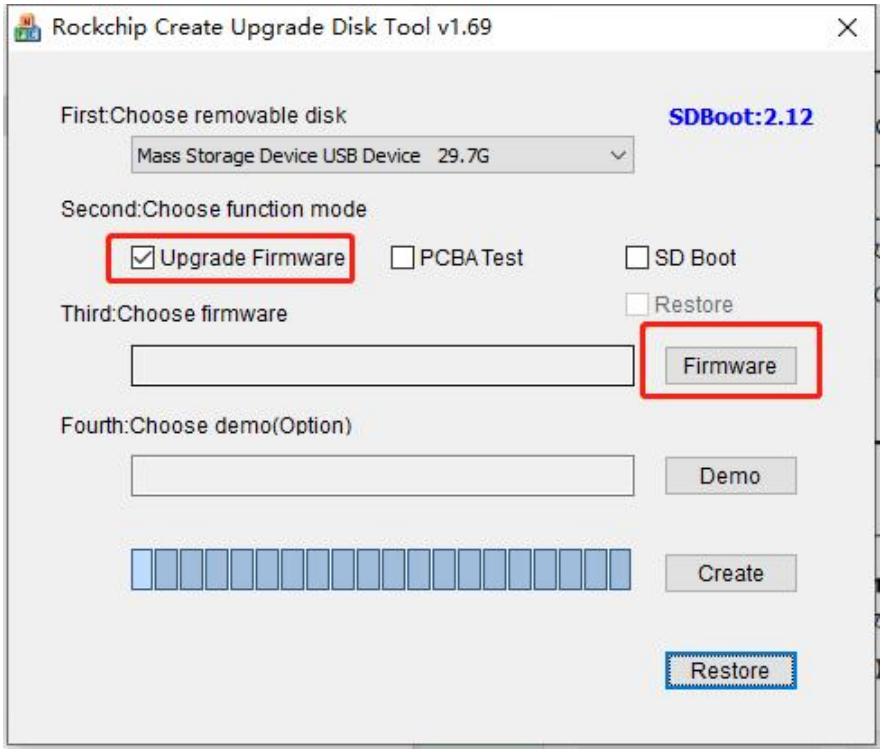


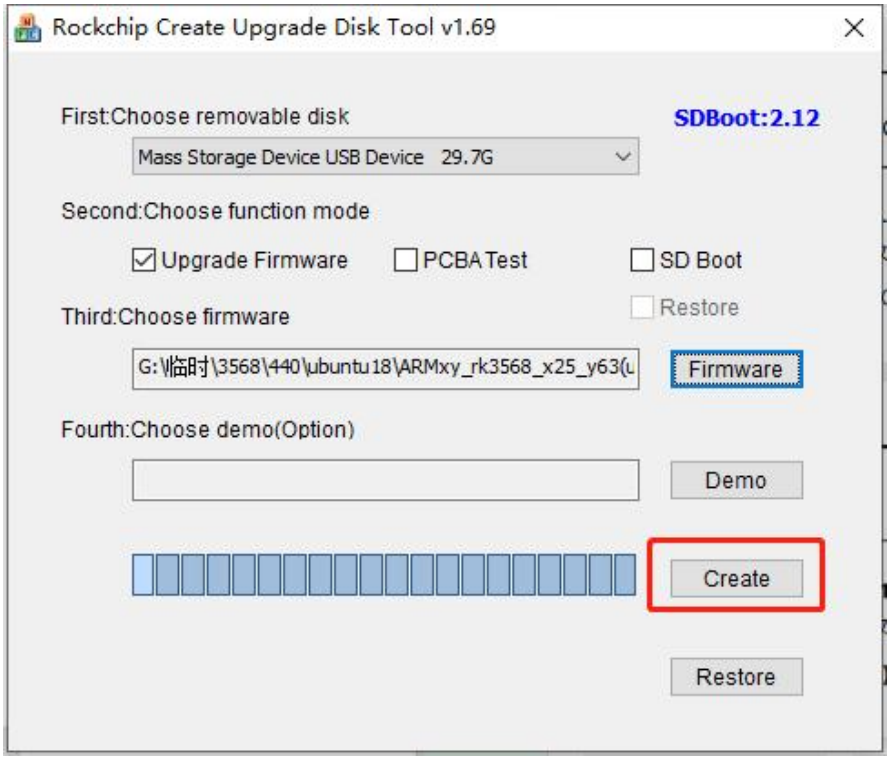


Wait for the formatting to complete, then click "OK" in the pop-up window.

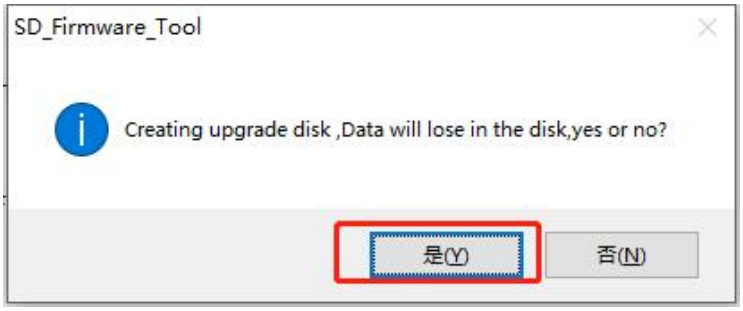


Check the "Upgrade Firmware" option, click "Choose Firmware" to choose the target system image file, then click "Start Create." In the pop-up window, click "Yes (Y)" to create the SD boot card.

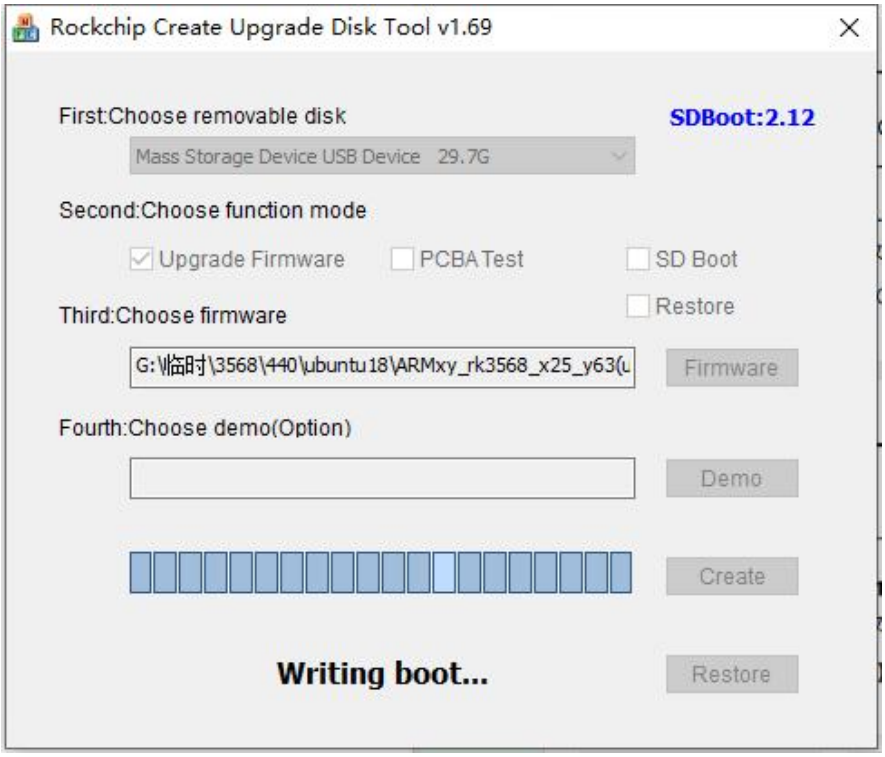

Rockchip Create Upgrade Disk Tool v1.69
First:Choose removable disk
SDBoot:2.12
Mass Storage Device USB Device 29.7G
Second:Choose function mode
Upgrade Firmware
PCBA Test
SD Boot
Restore
Third:Choose firmware
Firmware
Fourth:Choose demo(Option)
Demo
Create
Restore


Rockchip Create Upgrade Disk Tool v1.69
First:Choose removable disk
SDBoot:2.12
Mass Storage Device USB Device 29.7G
Second:Choose function mode
Upgrade Firmware
PCBA Test
SD Boot
Restore
Third:Choose firmware
G:\临时\3568\440\ubuntu18\ARMxy_rk3568_x25_y63(u
Firmware
Fourth:Choose demo(Option)
Demo
Create
Restore

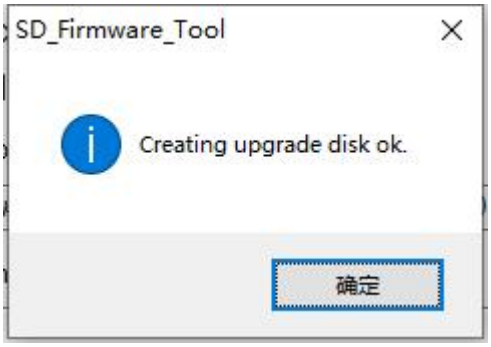
Click "OK"



Start flashing the system.



A prompt will appear indicating that the creation was successful.



4.2.2 System Programming

After inserting the prepared SD card into the device's Micro SD card slot and powering it on, the device will boot from the SD card and automatically flash the system to the eMMC. Wait for about 5 minutes. Once the system flashing is complete, the device will shut down automatically. Serial port output will be as follows.

```
[    6.617363] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    6.644025] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    6.670639] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    6.697305] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    6.723973] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    6.750596] rk-pcie fe000000.pcie: PCIe Linking... LTSSM is 0x3
[    7.172892]  mmcblk0: p1 p2 p3 p4 p5 p6 p7 p8 p9
[    8.057280] rk-pcie fe000000.pcie: PCIe Link Fail, LTSSM is 0x3, hw_retries=1
[    9.077303] rk-pcie fe000000.pcie: failed to initialize host
[   35.637286] vcc3v3_pcie20: disabling
[   35.637302] vcc3v3_clk: disabling

root@rk3562:/# [  373.246612] Alternate GPT is invalid, using primary GPT.
[  373.246649]  mmcblk0: p1 p2 p3 p4 p5 p6 p7 p8 p9
LOG_INFO:
Please remove SD CARD!!!, wait for reboot.
```

Remove the SD card, set the baud rate to 115200, and then power it on again.The system will boot from eMMC. Once the system starts, it will automatically log in as the root user, and the serial debug terminal will print startup information similar to the following:
"Bootdev(atags): mmc 0" indicates that the system is booting from eMMC.

```
U-Boot 2017.09-g54ac980 #root (Sep 18 2024 - 10:27:19 +0800)

Model: Rockchip RK3568 Evaluation Board
MPIDR: 0x81000000
PreSerial: 2, raw, 0xfe660000
DRAM:  2 GiB
Sysmem: init
Relocation Offset: 7d21c000
Relocation fdt: 7b7f8d68 - 7b7fece0
CR: M/C/I
DM: v1
dwmmc@fe2b0000: 1, dwmmc@fe2c0000: 2, sdhci@fe310000: 0
Bootdev(atags): mmc 0
MMC0: HS200, 200Mhz
PartType: EFI
boot mode: None
RESC: 'boot', blk@0x00014884
resource: sha256+
FIT: no signed, no conf required
DTB: rk-kernel.dtb
HASH(c): OK
```

5 Software

- BLIoTLink
- Node-red
- Docker
- QT

- Codesys
- IgnitionSCADA
- Ubuntu 20.04

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