IoT Gateway BPC-iMX6ULL-02 User Guide

Version: V1.0（2023-04）

Complied by: Polyhex Technology Company Limited（http://www.polyhex.net/）

IoT Gateway BPC-iMX6ULL-02 is a ruggedized and protected computer based on NXP i.MX 6UL processor. It is composed of a SOM iMX6ULL (core board), a BMB-07 board (carrier board) and a aluminum enclosure. It has the characteristics of ruggedness, dustproof, anti-vibration, anti-impact, wide temperature, portability and other indicators.

Figure 1

INDEX

Chapter 1 BPC-iMX6ULL-02 Introduction

IoT Gateway BPC-iMX6ULL-02 is a RISC architecture platform with high performance, wide temperature and flexible design. It serves as a gateway connecting inverters and remote monitoring center in power and energy application, which plays an important role.

Main features:

- Feature an advanced implementation of a single Arm® Cortex®-A7 core, which operates at speeds up to 900 MHz
- An integrated power management module that reduces the complexity of external power supply and simplifies power sequencing
- High security with support for secure encryption, tamper-proof monitoring, secure boot, and more
- Support Buildroot, Yocto

1.1. Overview of BPC-iMX6ULL-02

Figure 2

Figure 3

IoT Gateway BPC-iMX6ULL-02 uses SOM iMX6ULL and BMB-07 Board connection as the main board, which supports Ethernet, shock and vibration resistance, etc.. The specific specifications are as follows.

System	
Main Board	SOM iMX6ULL + BMB-07 Board
Type	BPC-iMX6ULL-02
CPU	NXP i.MX 6ULL/6UltraLite, 1 x Cortex-A7
Memory	512MB DDR3/DDR3L(256MB/1GB optional)
Storage	Onboard 16GB eMMC (8GB/32GB/64GB/128GB/256GB optional)
OS	Buildroot 2019.02.4, Yocto 2.5.2
Communication	
Ethernet	2 x 10Mbps/100Mbps RJ45 ports
Wi-Fi & Bluetooth	1 x 2.4GHz Wi-Fi, Bluetooth 5.0, external Wi-Fi and 4G/Lora/UWB(optional) SMA antenna interface
PCIe	1) Support 4G module, such as Quectel 4G module, built-in SIM card 2) Support LoRa module
UWB	1 x UWB module(optional)
I/O Interface	
USB 2.0	1 x USB 2.0 Host, the connector is Type-A interface (configurable as USB OTG interface)
SIM Slot	1 x Micro SIM pop-up card slot
Serial Ports	1) 2 x physically isolated RS485 2) 2 x physically isolated RS232 3) 2 x physically isolated CAN
DI & DO	1) 2 x physically isolated DI, supporting wet and dry nodes 2) 2 x physically isolated DO, support wet nodes, compatible with

	external relay dry nodes
LED & Key	1) 1 x Power LED 2) 2 x GPIO LED(functions can be customized) 3) 1 x Reset key
Power Supply	
Power Input	Default DC 12V power input, support DC 9V~24V wide voltage input 1) 1 x DC socket 2) 1 x 2Pin 3.81mm Phoenix
Mechanical & Environmental	
Dimension	107mm(D) x 107mm(W) x 35mm(H)
Enclosure Material	Aluminum alloy
Gross Weight	375g
Heat Dissipation	No fan, heat dissipation through the enclosure
CPU Temperature	0 °C to 70 °C
Operating Humidity	20%~95%(non-condensing)
Storage Humidity	5%~95%(non-condensing)

1.2. Composition of BPC-iMX6ULL-02

Figure 4

IoT Gateway BPC-iMX6ULL-02 assembly consists of these main components: SOM iMX6ULL + BMB-07 Board, enclosure and antenna.

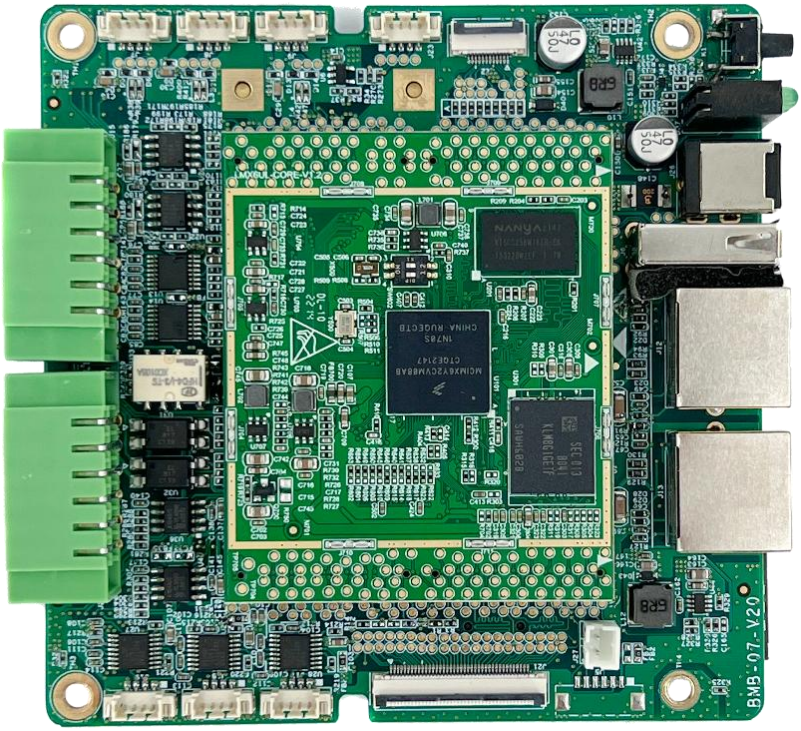


Figure 5 SOM iMX6ULL + BMB-07 Board

Figure 6 Enclosure

Figure 7 Antenna

1.3. Interface of BPC-iMX6ULL-02

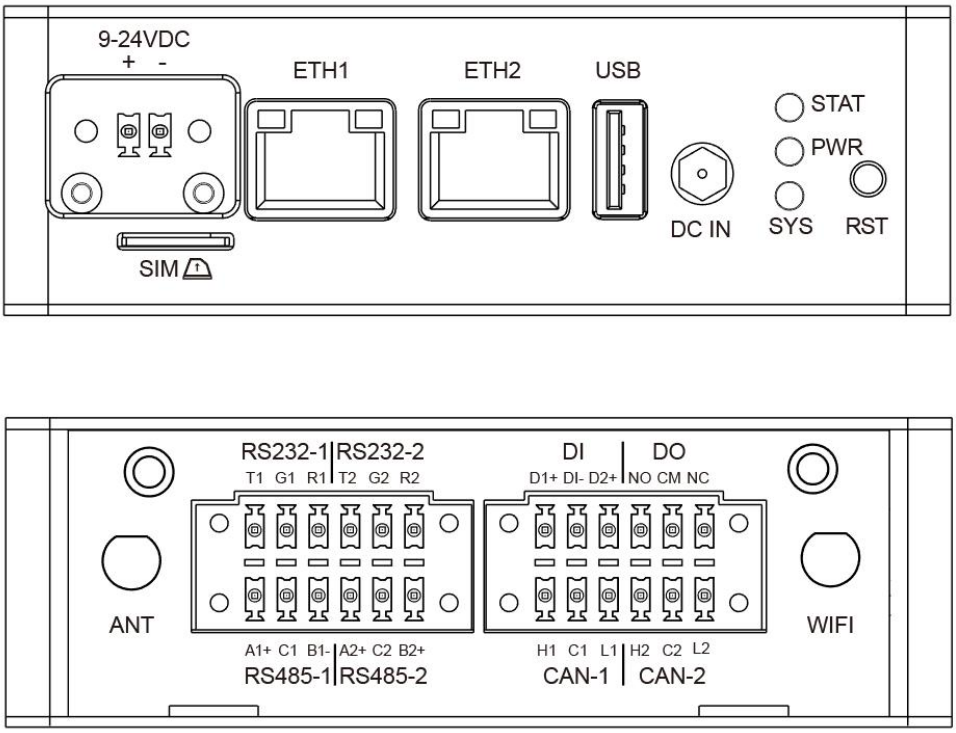


Figure 8 External I/O Interface

1.3.1. Power Interface

There are 2 power connectors, one DC socket connector (DC IN), one 2Pin 3.81mm Phoenix connector, with 9V~24V wide voltage input. As shown in the figure below.

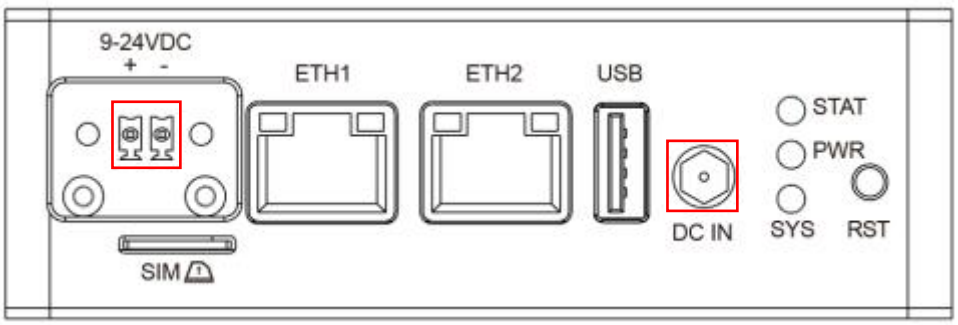


Figure 9

Function	Pin	Description
9-24VDC	+	DC power input positive pin

	-	DC Power input negative pin
DC IN	DC IN	DC Power input pin

1.3.2. Ethernet Interface

Two independent MAC RJ45 Ethernet ports on board (Network port 1: ETH1, Network port 2: ETH2), connect IoT Gateway to the network through the network cable of the RJ45 connector, and there is also a set of status indicators above the interface to display the signal, one is Link and the other is Active.

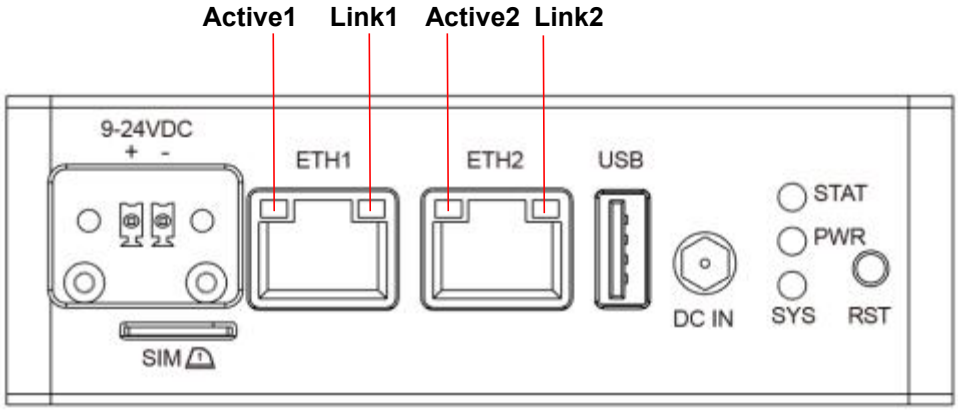


Figure 10

LED	Color	Description
Link	Green	Light, the network cable is plugged in, network connection status is good
Active	Yellow	Blinking, network data is being transmitted

1.3.3. USB Interface

IoT Gateway BPC-iMX6ULL-02 has 1 USB 2.0 host interface with Type-A connector.

1.3.4. RS485 Interface

IoT Gateway BPC-iMX6ULL-02 has 2 x RS485 interface with physical isolation, as shown in the figure below.

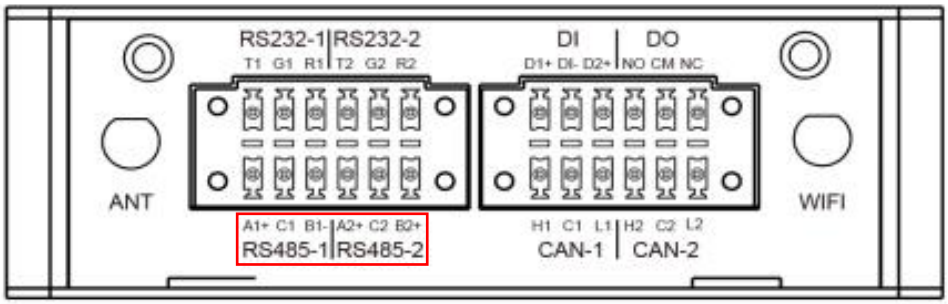


Figure 11

The 2 x RS485 interface is defined as follows:

Function	Pin	Description	Device node
RS485-1	A1+	Noninverting receiver input1 and Noninverting driver output1	/dev/ttymxc4
	C1	GND1	
	B1-	Inverting receiver input1 and inverting driver output1	
RS485-2	A2+	Noninverting receiver input2 and Noninverting driver output2	/dev/ttymxc5
	C2	GND2	
	B2+	Inverting receiver input2 and inverting driver output2	

1.3.5. RS232 Interface

There is 2 x RS232 interface with physical isolation, as shown in the figure below.

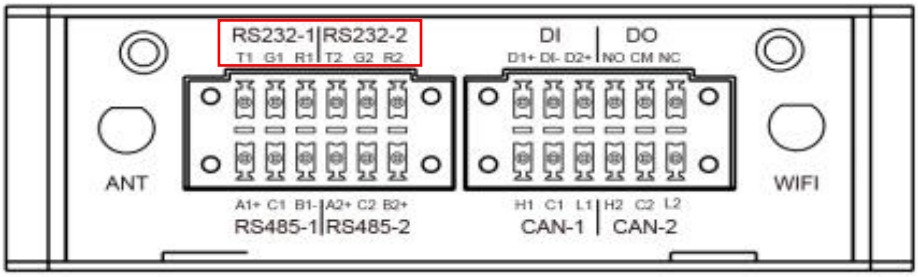


Figure 12

The 2 x RS232 interface is defined as follows:

Function	Pin	Description	Device node
RS232-1	T1	Transfer data1	/dev/ttymxc2
	G1	GND1	

	R1	Receive data1	
RS232-2	T2	Transfer data2	/dev/ttymxc3
	G2	GND2	
	R2	Receive data2	

1.3.6. CAN Interface

Near the bottom of WiFi antenna side there is 2 x CAN interface with physical isolation, as shown in the figure below.

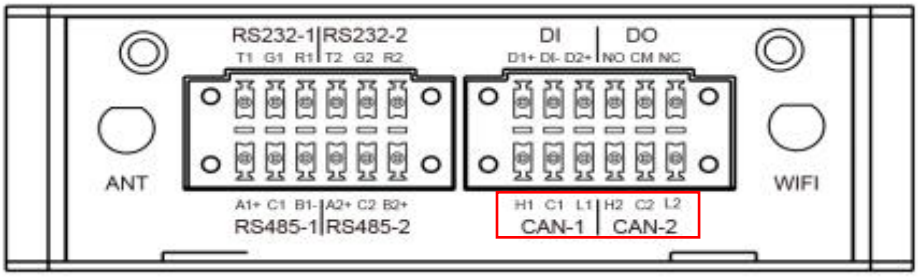


Figure 13

The 2 x CAN interface is defined as follows:

Function	Pin	Description	Device node
CAN-1	H1	High-level CAN bus line1	can0
	C1	GND1	
	L1	Low-level CAN bus line1	
CAN-2	H2	High-level CAN bus line2	can1
	C2	GND2	
	L2	Low-level CAN bus line2	

1.3.7. DI/DO Interface

Near the top of WiFi antenna side there is a group of DI and DO interfaces with physical isolation, as shown in the figure below.

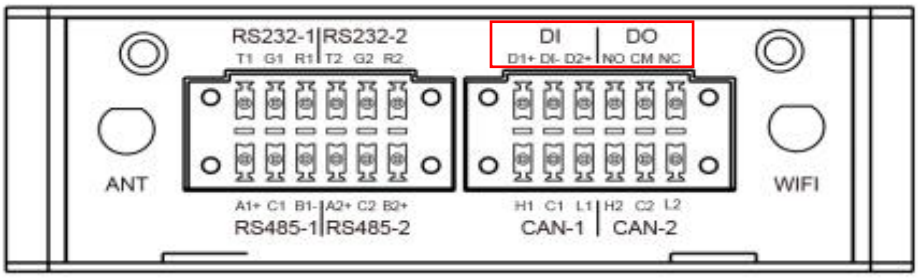


Figure 14

The interface is defined as follows:

Function	Pin	Description	Device node
DI	D1+	Digital input1 positive	/dev/input/event1
	DI-	Digital input negative	
	D2+	Digital input2 positive	/dev/input/event2
DO	NO	Normal open	
	CM	Common	
	NC	Normal connected	

1.3.8. LED and Key

There are 3 LED status indicators and 1 Reset button at the rear of IoT Gateway BPC-iMX6ULL-02, as shown in the figure below:

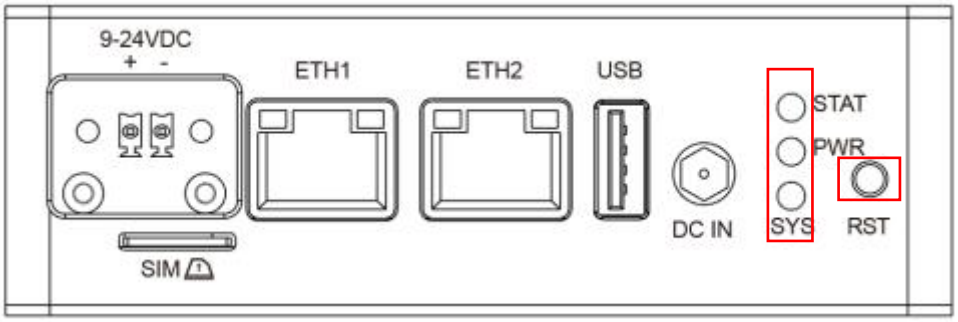


Figure 15

LED and Key	Status	Description
STAT (controllable)	Blinking	System works normally
	off	System works abnormally

PWR (not controllable)	Lighting	Power is on
	off	Power is off
SYS (not controllable)	Lighting	Power is on
	off	Power is off
RST	Long press 6s	System reset

1.3.9. Micro SIM Slot

There is 1 Micro SIM card slot below the 9-24VDC power connector of the IoT Gateway BPC-iMX6ULL-02, as shown in the figure below:

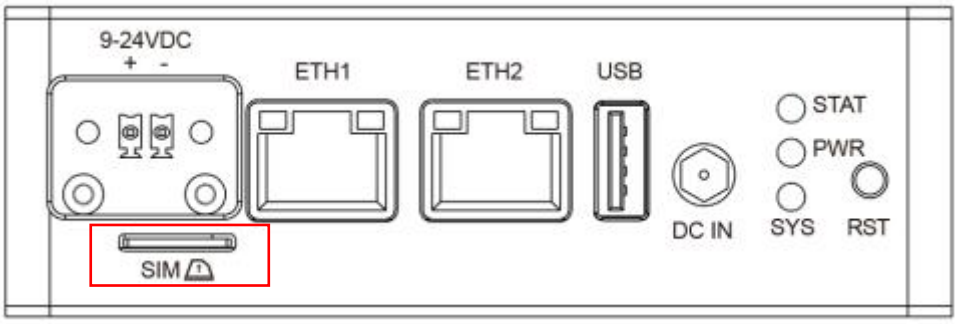


Figure 16

When inserting the Micro SIM card into the Micro SIM card slot, you need to pay attention to the insertion and removal direction (the direction position has been marked on the device).

1.4. Safety Precaution and Instruction

The following messages inform how to make each cable connection. In most cases, you will simply need to connect a standard cable.

Warning!

Always disconnect the power cord from the chassis whenever there is no workload required on it. Do not connect the power cable while the power is on. A sudden rush of power can damage sensitive electronic components. Only experienced electricians should open the chassis.

Caution!

Always ground yourself to remove any static electric charge before touching BPC-iMX6ULL-02. Modern electronic devices are very sensitive to electric charges. Use a grounding wrist strap at all times. Place all electronic components on a static-dissipative surface or in a static-shielded bag.

Safety Instruction

To avoid malfunction or damage to this product please observe the following:

1．Disconnect the device from the DC power supply before cleaning. Use a damp cloth. Do not use liquid detergents or spray-on detergents.

2．Keep the device away from moisture.

3．During installation, set the device down on a reliable surface. Drops and bumps will lead to damage.

4．Before connecting the power supply, ensure that the voltage is in the required range, and the way of wiring is correct.

5．Carefully put the power cable in place to avoid stepping on it.

6．If the device is not used for a long time, power it off to avoid damage caused by sudden overvoltage.

7．Do not pour liquid into the venting holes of the enclosure, as this could cause fire or electric shock.

8．For safety reasons, the device can only be disassembled by professional personnel.

9．If one of the following situations occur, get the equipment checked by service personnel:

- The power cord or plug is damaged.
- Liquid has penetrated into the equipment.
- The equipment has been exposed to moisture.
- The equipment does not work well, or you cannot get it to work according to the user's manual.
- The equipment has been dropped and damaged.
- The equipment has obvious signs of breakage.

10．Do not place the device in a place where the ambient temperature is below 0°C (0°F) or above 70°C (158°F). This will damage the machine. It needs to be kept in an environment at controlled temperature.

11．Due to the sensitive nature of the equipment, it must be stored in a restricted access location, only accessible by qualified engineer.

DISCLAIMER: Polyhex disclaims all responsibility for the accuracy of any statement of this instructional document.

Declaration of conformity

CE: This device has passed the CE environmental indicator test.

FCC: This equipment has been tested and found to comply with the FCC rules.

RoHS: This device has passed the RoHS test.

CCC: This equipment has passed the CCC test.

Technical support and assistance

1. Visit Polyhex website http://www.polyhex.net/ where you can find the latest information about the product.
2. Contact your distributor, sales representative or Polyhex's customer service center (https://discord.com/invite/adaHHaDkH2) for technical support if you need additional assistance. Please have the following info ready before you call:
    - Product name
    - Description of your peripheral attachments
    - Description of your software(operating system, version, application software, etc.)
    - A complete description of the problem
    - The exact wording of any error messages

Chapter 2 Getting started with IoT Gateway BPC-iMX6ULL-02

IoT Gateway BPC-iMX6ULL-02 product list:

- 1 x WiFi antenna
- 1 x Power adapter
- 1 x BPC-iMX6ULL-02 box

After receiving the product, install the accessories as follows.

2.1. Installation

1. Install the WiFi antenna to the WiFi antenna connection port as shown in the following figure.

Figure 17

2. Connect the power adapter to the DC connector of enclosure. When the SYS and PWR

LED are on, and the STAT LED is blinking, it proves that the IoT Gateway is powered on.

Figure 18 Power adapter

2.2. Power on

Note: The factory default configuration of IoT Gateway BPC-iMX6ULL-02 is WiFi antenna. If you need to configure other antennas, please contact our engineer for modification before leaving the factory, and do not disassemble the machine by yourself.

Chapter 3 Software Application Examples

3.1. Remote login SSH

Plug the cable into IoT Gateway network port, so that the device is connected to the LAN, enter the router background, query the IP address obtained by the device according to the MAC address, SSH to the device background through "putty" or other tools, access the account: root, the password is empty by default, as shown below:

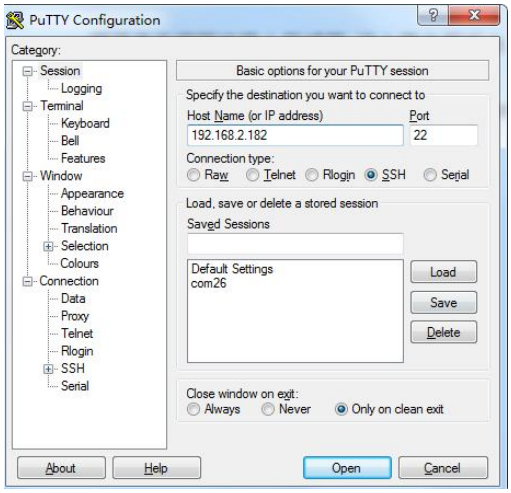


Figure 19

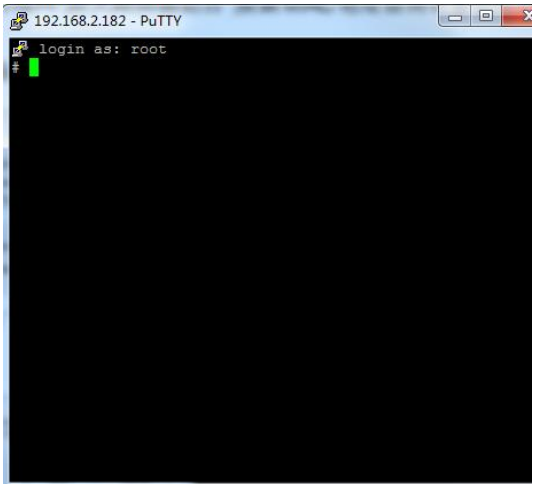


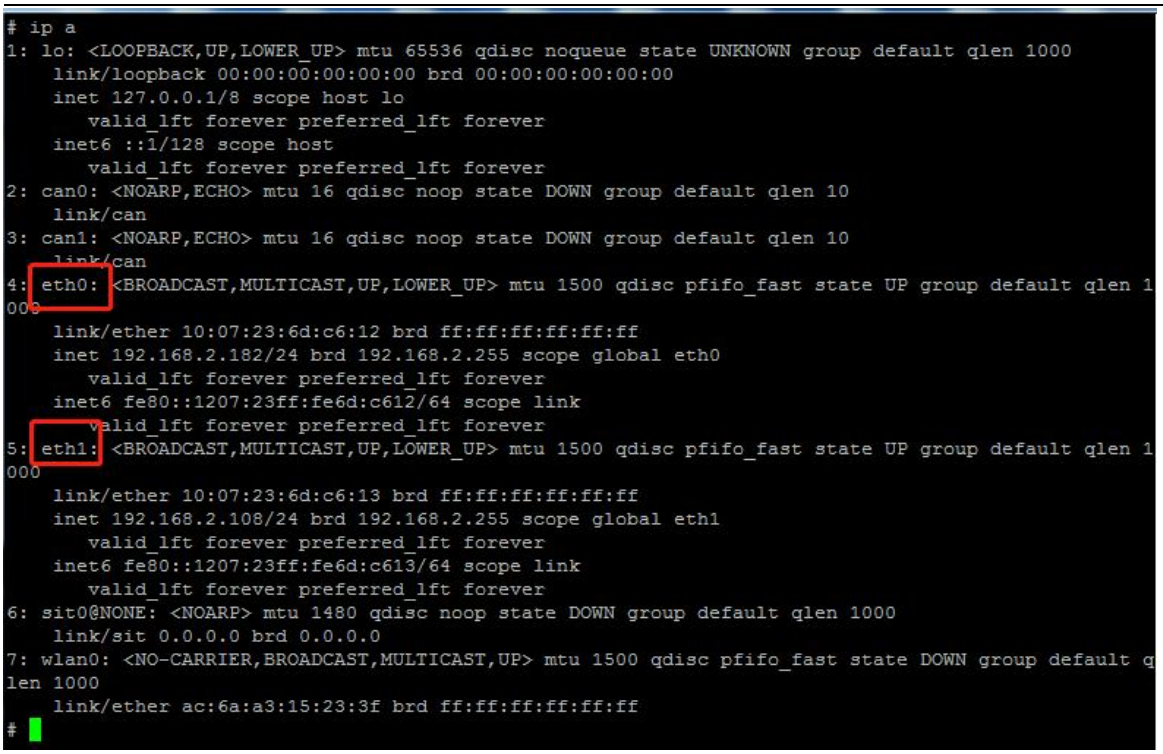
Figure 20

Change root password via passwd root, and enter the new password twice in a row.

3.2. Usage of Ethernet

1. Query ip command.

ip a



As shown above: eth0 network card corresponds to the network port of the device silkscreen "ETH1" (Figure 10, left side);

eth1 network card corresponds to the network port of the device silkscreen "ETH2" (Figure 10, right side).

2. Apply ping command.

```
ping -i eth0 192.168.2.254
```

```
# ping -i eth0 192.168.2.254
ping: option argument contains garbage: eth0
ping: this will become fatal error in the future
PING 192.168.2.254 (192.168.2.254) 56(84) bytes of data.
64 bytes from 192.168.2.254: icmp_seq=1 ttl=254 time=1.27 ms
64 bytes from 192.168.2.254: icmp_seq=2 ttl=254 time=0.733 ms
64 bytes from 192.168.2.254: icmp_seq=3 ttl=254 time=1.34 ms
64 bytes from 192.168.2.254: icmp_seq=4 ttl=254 time=1.19 ms
64 bytes from 192.168.2.254: icmp_seq=5 ttl=254 time=1.19 ms
64 bytes from 192.168.2.254: icmp_seq=6 ttl=254 time=1.17 ms
64 bytes from 192.168.2.254: icmp_seq=7 ttl=254 time=1.20 ms
64 bytes from 192.168.2.254: icmp_seq=8 ttl=254 time=1.17 ms
64 bytes from 192.168.2.254: icmp_seq=9 ttl=254 time=1.16 ms
64 bytes from 192.168.2.254: icmp_seq=10 ttl=254 time=1.19 ms
64 bytes from 192.168.2.254: icmp_seq=11 ttl=254 time=1.21 ms
64 bytes from 192.168.2.254: icmp_seq=12 ttl=254 time=1.23 ms
64 bytes from 192.168.2.254: icmp_seq=13 ttl=254 time=1.20 ms
64 bytes from 192.168.2.254: icmp_seq=14 ttl=254 time=1.23 ms
64 bytes from 192.168.2.254: icmp_seq=15 ttl=254 time=1.22 ms
64 bytes from 192.168.2.254: icmp_seq=16 ttl=254 time=1.23 ms
64 bytes from 192.168.2.254: icmp_seq=17 ttl=254 time=1.20 ms
64 bytes from 192.168.2.254: icmp_seq=18 ttl=254 time=1.23 ms
64 bytes from 192.168.2.254: icmp_seq=19 ttl=254 time=1.19 ms
```

3.3. Usage of WiFi

- Edit the configuration file and set the "SSID" and connection password of the connected router.

vi /etc/wpa_supplicant.conf

wpa_supplicant -Dnl80211 -iwlan0 -c/etc/wpa_supplicant.conf &

```
# wpa_supplicant -Dnl80211 -iwlan0 -c/etc/wpa_supplicant.conf &
# Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
wlan0: Trying to associate with SSID 'BH123'
wlan0: Associated with 0c:d8:6c:a9:cc:08
wlan0: CTRL-EVENT-CONNECTED - Connection to 0c:d8:6c:a9:cc:08 completed [id=0 id_str=]
wlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0

#
```

- Obtain the IP address assigned by the router.

udhcpc -i wlan0 –n

```
#  udhcpc -i wlan0 -n
udhcpc: started, v1.34.1
udhcpc: broadcasting discover
udhcpc: broadcasting select for 192.168.10.8, server 192.168.10.254
udhcpc: lease of 192.168.10.8 obtained from 192.168.10.254, lease time 86400
deleting routers
adding dns 202.96.134.133
adding dns 114.114.114.114
#
```

3.4. Usage of Bluetooth

Start bluetooth and match bluetooth:

hciconfig hci0 up

bluetoothctl

power on

agent on

default-agent

scan on

pair yourDeviceMAC #Match the Bluetooth MAC address

```
# bluetoothctl
Agent registered
[CHG] Controller AC:6A:A3:15:23:40 Pairable: yes
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# agent on
Agent is already registered
[bluetooth]# default-agent
Default agent request successful
[bluetooth]# scan on
Discovery started
[CHG] Controller AC:6A:A3:15:23:40 Discovering: yes
[NEW] Device 6F:77:E4:55:30:6B 6F-77-E4-55-30-6B
[NEW] Device 58:1F:3E:7C:17:CE 58-1F-3E-7C-17-CE
[NEW] Device 61:8D:F0:19:75:3E 61-8D-F0-19-75-3E
[NEW] Device 68:7A:15:E7:AD:CA 68-7A-15-E7-AD-CA
[NEW] Device 78:21:08:79:5C:85 78-21-08-79-5C-85
[NEW] Device 6F:66:07:AC:13:D7 6F-66-07-AC-13-D7
[NEW] Device 68:E4:6A:8E:99:74 68-E4-6A-8E-99-74
[NEW] Device 54:AF:B7:03:4D:69 54-AF-B7-03-4D-69
[NEW] Device 74:5F:D2:47:FC:43 74-5F-D2-47-FC-43
```

```
[bluetooth]# pair 4C:02:20:3C:2A:6C
Attempting to pair with 4C:02:20:3C:2A:6C
[CHG] Device 4C:02:20:3C:2A:6C Connected: yes
Request confirmation
[agent] Confirm passkey 381184 (yes/no): yes
[CHG] Device 4C:02:20:3C:2A:6C Modalias: bluetooth:v038Fp1200d1436
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001105-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001112-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001115-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001132-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001800-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 0000fdaa-0000-1000-8000-00805f9b34fb
[CHG] Device 4C:02:20:3C:2A:6C UUIDs: 98b97136-36a2-11ea-8467-484d7e99a198
[CHG] Device 4C:02:20:3C:2A:6C ServicesResolved: yes
[CHG] Device 4C:02:20:3C:2A:6C Paired: yes
Pairing successful
```

3.5. Usage of USB

1. Access the U disk in FAT32 format, the system will automatically mount it to the /mnt path.

```
df -h
```

```
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 4.8G     60.7M      4.4G   1% /
devtmpfs                163.9M         0    163.9M   0% /dev
tmpfs                   244.4M         0    244.4M   0% /dev/shm
tmpfs                   244.4M     68.0K    244.4M   0% /tmp
tmpfs                   244.4M     40.0K    244.4M   0% /run
/dev/mmcblk1p3            1.7G     60.0K      1.6G   0% /recovery
/dev/sda1                28.7G    544.0K     28.7G   0% /mnt
#
```

- If the U disk is not mounted, you can mount the U disk with the following command:
  - Query the U disk letter: fdisk -l

```
# fdisk -l
Disk /dev/mmcblk1: 7456 MB, 7818182656 bytes, 15269888 sectors
238592 cylinders, 4 heads, 16 sectors/track
Units: sectors of 1 * 512 = 512 bytes

Device       Boot StartCHS    EndCHS        StartLBA     EndLBA    Sectors  Size Id Type
/dev/mmcblk1p1    320,0,1     959,3,16         20480    1044479    1024000  500M  c Win95 FAT32 (L
BA)
/dev/mmcblk1p2    768,0,1     639,3,16       1228800   11509759   10280960 5020M 83 Linux
/dev/mmcblk1p3    640,0,1     1023,3,16     11509760   15269887    3760128 1836M 83 Linux
Disk /dev/sda: 29 GB, 30784094208 bytes, 60125184 sectors
3742 cylinders, 255 heads, 63 sectors/track
Units: sectors of 1 * 512 = 512 bytes

Device  Boot StartCHS    EndCHS        StartLBA     EndLBA    Sectors  Size Id Type
/dev/sda1    0,0,33      1023,254,63         32   60125183   60125152 28.6G  c Win95 FAT32 (LBA)
#
```

  - Mounting the U disk: mount /dev/sda1 /mnt

```
# mount /dev/sda1 /mnt
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 4.8G     60.8M      4.4G   1% /
devtmpfs                163.9M         0    163.9M   0% /dev
tmpfs                   244.4M         0    244.4M   0% /dev/shm
tmpfs                   244.4M     52.0K    244.4M   0% /tmp
tmpfs                   244.4M     36.0K    244.4M   0% /run
/dev/mmcblk1p3            1.7G     60.0K      1.6G   0% /recovery
/dev/sda1                28.7G    544.0K     28.7G   0% /mnt
#
```

2. Enter the U disk directory: cd /mnt

```
# cd /mnt/
# ls
BMB07-factorytest           BMB07-factorytest-V1.0.4.rar  System Volume Information
# cp BMB07-factorytest-V1.0.4.rar /root/
```

3.6. Verification of RS232

Connect T1 of RS232-1 to the receiving end R2 of RS232-2, R1 to the sending end T2 of RS232-2, and G1 to the ground terminal G2 of RS232-2. The wiring is shown in the following

figure.

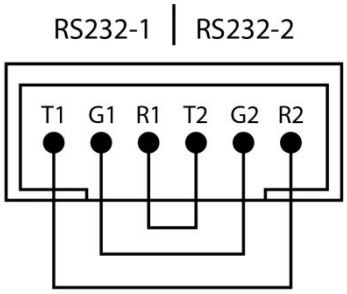


Figure 21

- Set RS232-1 to write data and RS232-2 to read data

uart_read /dev/ttymxc3 115200 & #(RS232-2 backend read data)

uart_write /dev/ttymxc2 115200 123 #(RS232-1 write data)

uart_write /dev/ttymxc2 115200 123456 #(RS232-1 write data)

killall uart_read

- Switch to RS232-2 to write data, RS232-1 to read data

uart_read /dev/ttymxc2 115200 & #(RS232-1 backend read data)

uart_write /dev/ttymxc3 115200 456 #(RS232-2 write data)

uart_write /dev/ttymxc3 115200 456789 #(RS232-2 write data)

killall uart_read

```
# uart_read /dev/ttymxc3 115200 &
# open /dev/ttymxc3 speed 115200 8n1

# uart_write /dev/ttymxc2 115200 123
open /dev/ttymxc2 speed 115200 8n1
read [3] [123]
# uart_write /dev/ttymxc2 115200 123456
open /dev/ttymxc2 speed 115200 8n1
read [6] [123456]
# killall uart_read
[3]+  Terminated              uart_read /dev/ttymxc3 115200
# uart_read /dev/ttymxc2 115200 &
# open /dev/ttymxc2 speed 115200 8n1

# uart_write /dev/ttymxc3 115200 456
open /dev/ttymxc3 speed 115200 8n1
read [3] [456]
# uart_write /dev/ttymxc3 115200 456789
open /dev/ttymxc3 speed 115200 8n1
read [6] [456789]
# killall uart_read
[3]+  Terminated              uart_read /dev/ttymxc2 115200
#
```

3.7. Verification of RS485

Connect A1 to A2, and connect B1 to B2, and connect C1 to C2. The wiring is shown in the following figure.

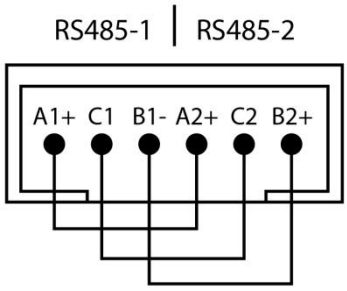


Figure 22

- Set RS485-2 to write data and RS485-1 to read data

```
uart_read /dev/ttymxc4 115200 &   #(RS485-1 backend read data)
uart_write /dev/ttymxc5 115200 123   #(RS485-2 write data)
uart_write /dev/ttymxc5 115200 123456   #(RS485-2 write data)
killall uart_read
```

- Switch to RS485-1 to write data, RS485-2 to read data

```
uart_read /dev/ttymxc5 115200 &   #(RS485-2 backend read data)
uart_write /dev/ttymxc4 115200 456   #(RS485-1 write data)
uart_write /dev/ttymxc4 115200 456789   #(RS485-1 write data)
killall uart_read
```

```
# uart_read /dev/ttymxc4 115200 &
# open /dev/ttymxc4 speed 115200 8n1
# uart_write /dev/ttymxc5 115200 123
open /dev/ttymxc5 speed 115200 8n1
read [3] [123]
# uart_write /dev/ttymxc5 115200 123456
open /dev/ttymxc5 speed 115200 8n1
read [5] [12345]
read [1] [6]
# killall uart_read
[3]+  Terminated                 uart_read /dev/ttymxc4 115200
# uart_read /dev/ttymxc5 115200 &
# open /dev/ttymxc5 speed 115200 8n1
# uart_write /dev/ttymxc4 115200 456
open /dev/ttymxc4 speed 115200 8n1
read [3] [456]
# uart_write /dev/ttymxc4 115200 456789
open /dev/ttymxc4 speed 115200 8n1
read [5] [45678]
read [1] [9]
#
```

3.8. Verification of CAN

Connect H1 to H2, and connect L1 to L2 (that is, H to H, and L to L). The wiring is shown in the following figure.

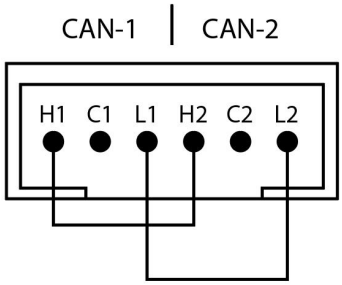


Figure 23

```
# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN group default qlen 10
    link/can
3: can1: <NOARP,ECHO> mtu 16 qdisc noop state DOWN group default qlen 10
    link/can
4: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 10:07:23:6d:c6:12 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.182/24 brd 192.168.2.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::1207:23ff:fe6d:c612/64 scope link
       valid_lft forever preferred_lft forever
5: eth1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group default ql
en 1000
    link/ether 10:07:23:6d:c6:13 brd ff:ff:ff:ff:ff:ff
6: sit0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000
    link/sit 0.0.0.0 brd 0.0.0.0
7: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether ac:6a:a3:15:23:3f brd ff:ff:ff:ff:ff:ff
#
```

As shown above: can0 corresponds to the port of the device silkscreen "CAN-1" (Figure 13, left side);

can1 corresponds to the port of the device silkscreen "CAN-2" (Figure 13, right side).

- CAN1 sends data, CAN0 receives data

Open a Terminal, configure CAN0 to receive, and CAN1 to send.

```
ifconfig can0 down
ifconfig can1 down
ip link set can0 type can bitrate 200000
ip link set can1 type can bitrate 200000
ifconfig can0 up
ifconfig can1 up
candump can0 &     #(can0 backend receive data)
cansend can1 123#1122334455667788     #(can1 send data)
```

- CAN1 receives data, CAN0 sends data

In the Terminal, switch can1 to receive and can0 to send.

```
candump can1 &     #(can1 backend receive data)
cansend can0 123#1122334455667788     #(can0 send data)
```

```
# ifconfig can0 down
# ifconfig can1 down
# ip link set can0 type can bitrate 20000
# ip link set can1 type can bitrate 20000
# ifconfig can0 up
# ifconfig can1 up
# candump can0 &
# canse
cansend      cansequence
# cansend can1 123#1122334455667788
#   can0  123   [8]  11 22 33 44 55 66 77 88

# candump can1 &
# cansend can0 123#1122334455667788
#   can0  123   [8]  11 22 33 44 55 66 77 88
  can1  123   [8]  11 22 33 44 55 66 77 88
```

3.9. Verification of DI

```
# ls /dev/input/event* -l
crw-------    1 root     root       13,  64 Jan  1  1970 /dev/input/event0
crw-------    1 root     root       13,  65 Nov 10 07:22 /dev/input/event1
crw-------    1 root     root       13,  66 Nov 10 07:22 /dev/input/event2
#
```

As shown above: DIN1 corresponds to the port of the device silkscreen "D1+ & DI-" (Figure 14, left side);

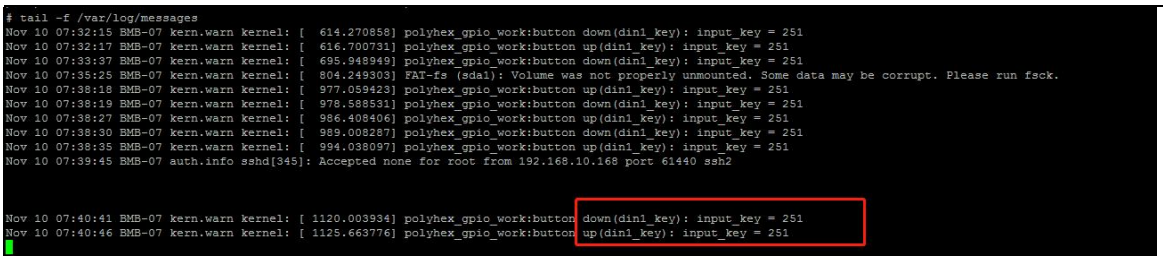
DIN2 corresponds to the port of the device silkscreen "D2+ & DI-" (Figure 14, right side).

- Take DIN1 as an example, lead to a type-C female connector, as shown in the following figure, input/disconnect a 5V power.

Figure 24

- Query message event.

tail –f /var/log/message



3.10. Verification of DO

The default state of DO is "NC", and the connectivity between CM and NC can be measured using a multimeter.

Switch the DO state:

ph_ctl_gpio kv_coil_en_on (Switch status to "NO")

ph_ctl_gpio kv_coil_en_off (Switch status to "NC")

3.11. Usage of 4G Module

Insert the Micro SIM card, connect 4G module (EC21ECGA-128-SSNS for example), connect the antenna adapter cable and the matching 4G external antenna in the IoT Gateway power-off state.

The 4G module is identified as /dev/ttyUSB2 under the system and can be verified by the relevant instructions of the serial port debugging tool microcom.

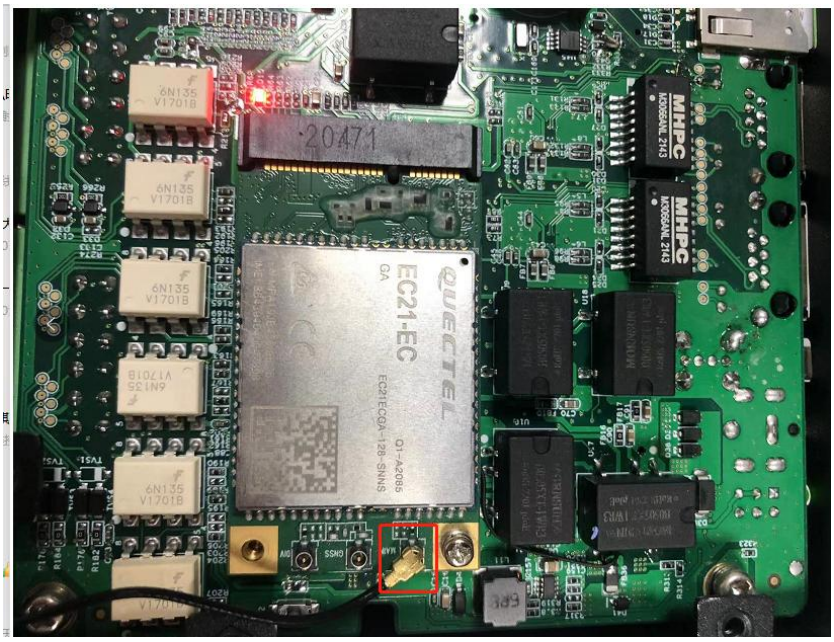


Figure 25

- Query 4G module command:

ifdown ppp0

ifup ppp0

ip a

```
# ifdown ppp0
# ifup ppp0
# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN group default qlen 10
    link/can
3: can1: <NOARP,ECHO> mtu 16 qdisc noop state DOWN group default qlen 10
    link/can
4: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 10:07:23:6d:c6:12 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.182/24 brd 192.168.2.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::1207:23ff:fe6d:c612/64 scope link
       valid_lft forever preferred_lft forever
5: eth1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group default qlen 1000
    link/ether 10:07:23:6d:c6:13 brd ff:ff:ff:ff:ff:ff
6: sit0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000
    link/sit 0.0.0.0 brd 0.0.0.0
7: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether ac:6a:a3:15:23:3f brd ff:ff:ff:ff:ff:ff
8: wwan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 26:43:fb:f3:15:66 brd ff:ff:ff:ff:ff:ff
10: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default qlen 3
    link/ppp
    inet 10.214.138.254 peer 10.64.64.64/32 scope global ppp0
       valid_lft forever preferred_lft forever
#
```

- Verify the function of 4G module:

1. Apply ping command to check the network.

ping -I ppp0 www.baidu.com

```
# ping -I ppp0 www.baidu.com
PING www.a.shifen.com (112.80.248.76) from 10.214.138.254 ppp0: 56(84) bytes of data.
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=1 ttl=55 time=1206 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=2 ttl=55 time=201 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=3 ttl=55 time=110 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=4 ttl=55 time=107 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=5 ttl=55 time=105 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=6 ttl=55 time=104 ms
64 bytes from 112.80.248.76 (112.80.248.76): icmp_seq=7 ttl=55 time=106 ms
```

2. Query and verify the 4G module.

microcom /dev/ttyUSB2

AT+CPIN? #SIM card detection

AT+CIMI #Query SIM card number CIMI

AT+CGSN #Query module IMEI

AT+CSQ #Query signal strength

```
# microcom /dev/ttyUSB2

+CPIN: READY

OK

460065021200496

OK

864394040047898

OK

+CSQ: 23,99

OK
```

3.12. Usage of LoRa Module

Disconnect the power of the IoT Gateway, connect LoRa module (take HLM5934-H01 as an example), connect the antenna adapter cable and the matching LoRa external antenna.

The LoRa module is identified as /dev/spidev1.0 under the system.

```
# ls /dev/spidev* -l
crw-------    1 root     root      153,   0 Jan  1  1970 /dev/spidev0.0
crw-------    1 root     root      153,   1 Jan  1  1970 /dev/spidev1.0
#
```

- Execute the compiled script (Polyhex can provide) to start the LoRa module.

```
# cd lora/
# ls
global_conf.json  lora_pkt_fwd      reset_lgw.sh
```

- Check whether LoRa module works.

./lora_pakt_fwd

```
# ./lora_pkt_fwd &
# *** Packet Forwarder ***
Version: 2.1.0
*** SX1302 HAL library version info ***
Version: 2.1.0;
***
INFO: Little endian host
INFO: found configuration file global_conf.json, parsing it
INFO: global_conf.json does contain a JSON object named SX130x_conf, parsing SX1302 parameters
INFO: com_type SPI, com_path /dev/spidev1.0, lorawan_public 1, clksrc 0, full_duplex 0
INFO: antenna_gain 0 dBi
INFO: Configuring legacy timestamp
INFO: no configuration for SX1261
INFO: Configuring Tx Gain LUT for rf_chain 0 with 16 indexes for sx1250
INFO: radio 0 enabled (type SX1250), center frequency 470600000, RSSI offset -207.000000, tx enabled 1, single input mode 1
INFO: radio 1 enabled (type SX1250), center frequency 471400000, RSSI offset -207.000000, tx enabled 0, single input mode 1
INFO: Lora multi-SF channel 0>  radio 0, IF -300000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 1>  radio 0, IF -100000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 2>  radio 0, IF 100000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 3>  radio 0, IF 300000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 4>  radio 1, IF -300000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 5>  radio 1, IF -100000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 6>  radio 1, IF 100000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora multi-SF channel 7>  radio 1, IF 300000 Hz, 125 kHz bw, SF 5 to 12
INFO: Lora std channel> radio 1, IF -200000 Hz, 250000 Hz bw, SF 7, Explicit header
INFO: FSK channel> radio 1, IF 300000 Hz, 125000 Hz bw, 50000 bps datarate
INFO: global_conf.json does contain a JSON object named gateway_conf, parsing gateway parameters
INFO: gateway MAC address is configured to FFFEDCA6320E9516
INFO: server hostname or IP address is configured to "localhost"
INFO: upstream port is configured to "1700"
INFO: downstream port is configured to "1700"
INFO: downstream keep-alive interval is configured to 10 seconds
INFO: statistics display interval is configured to 30 seconds
INFO: upstream PUSH_DATA time-out is configured to 100 ms
INFO: packets received with a valid CRC will be forwarded
INFO: packets received with a CRC error will NOT be forwarded
INFO: packets received with no CRC will NOT be forwarded
INFO: Beaconing period is configured to 0 seconds
INFO: Beaconing signal will be emitted at 869525000 Hz
INFO: Beaconing datarate is set to SF9
INFO: Beaconing modulation bandwidth is set to 125000Hz
INFO: Beaconing TX power is set to 14dBm
INFO: Beaconing information descriptor is set to 0
INFO: global_conf.json does contain a JSON object named debug_conf, parsing debug parameters
INFO: got 2 debug reference payload
INFO: reference payload ID 0 is 0xCAFE1234
INFO: reference payload ID 1 is 0xCAFE2345
INFO: setting debug log file name to loragw_hal.log
write command lora_en_off size=11
write command lora_pwd_off size=12
write command lora_pwd_on size=11
write command lora_en_on size=10
write command lora_rst_on size=11
CoreCell reset through /dev/lora_reset...
CoreCell power enable through /dev/lora_en...
write command lora_en_on size=10
write command lora_rst_off size=12
write command lora_rst_on size=11
Opening SPI communication interface
Note: chip version is 0x10 (v1.0)
```

```
# INFO: Configuring SX1250_0 in single input mode
INFO: Configuring SX1250_1 in single input mode
INFO: using legacy timestamp
INFO: LoRa Service modem: configuring preamble size to 8 symbols
ARB: dual demodulation disabled for all SF
INFO: [main] concentrator started, packet can now be received
INFO: concentrator EUI: 0x0016c001f10a62ef
```

3.13. Verification of RTC

- Confirm that the HYM8653S driver module is loaded successfully.

dmesg | grep rtc-hym8653

- Read the RTC time.

hwclock --systohc

hwclock --show

```
# hwclock --systohc
# hwclock --show
Wed Nov  9 12:12:16 2022  0.000000 seconds
#
```