HJSIP®

HJ-568IMH_Product Specification

Based on OnMicro OM6626
8.5*9.2*2.2mm
onboard/ceramic/external antenna
BLE6.1 wireless transmission module

Version：V1.0
Date：2026/06/26

Version:	**V1.0**
Date:	**2026/06/26**
State:	**Published**

Preface

Thanks for using the Bluetooth module provided by HJSIP.The HJ-568IMH series standard Bluetooth module is a product based on the high-performance and low-power IoT Bluetooth transceiver of OnMicro OM6626. The module is packaged in a stamp-type enclosure and supports onboard/ ceramic/external antennas. It features low power consumption, small size, and strong anti-interference capability, and is suitable for various application scenarios.

This module is mainly used for data communication, and the company does not assume responsibility for property losses or personal injuries caused by improper operations of users. Please develop the product according to the technical specifications and reference design in the manual. At the same time, pay attention to the general safety matters that should be concerned about when using mobile products.

Before the announcement, the company has the right to modify the content of this manual according to the needs of technological development.

Copyright Notice

Contents

Contact Information

Tangshan Hongjia Electronic Technology Co., Ltd.

Address: No. 203, Unit 1, Building 4, B46 Shop, Zhengtaili Chunxiao, Nvzhizhai Town, Lunan District, Tangshan City, Hebei Province (Opposite Building 204, Chunxiao Phase 1, 6-story building)
Technical Support: liyinfei@tshjdz.com or liujiahang@tshjdz.com

Version History

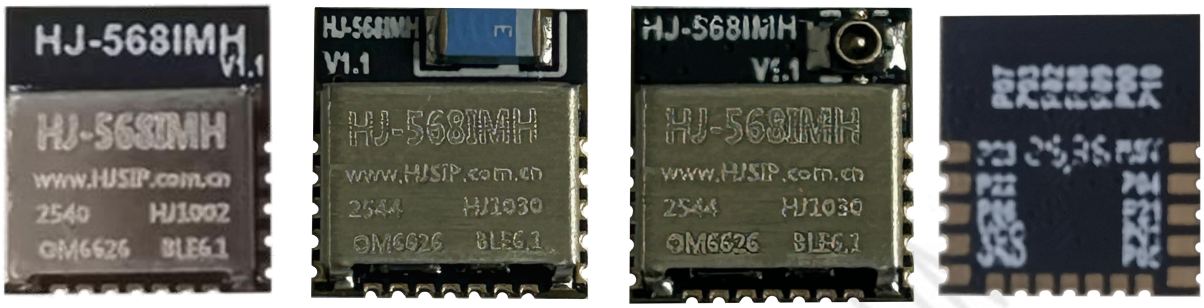
Version	Date	Change Description	Reviser	Reviewer
V1.0	2026/06/26	Initial Version	WYW	LMY
V1.1	2026/06/30	7.2:Add remarks to software instructions.	WYW	LMY

Applicable module selection

No.	Module model	Type	Description
1	HJ-568IMH_SPPv2	Serial port transparent transmission standard version	Built-in serial port transparent firmware, the firmware module is a two-way communication bridge between Bluetooth devices or mobile phones and MCU, users do not need to understand the Bluetooth protocol stack, through the serial port command operation and serial port data can be received, simple operation, shorten the user development cycle, speed up the product to market.
2	HJ-568IMH_CUSv2	Customized version for customers	This version supports customer customized firmware, customers according to product needs to propose functions, we will customize modules with dedicated firmware versions to supply to customers.
3	HJ-568IMH_DEV	Customer development version	Provide standard SDK to facilitate users' secondary development with "OPEN CPU".

1 Introduction

The HJ-568IMH series standard Bluetooth module is a high-performance IoT Bluetooth transceiver. The module is packaged in a postage stamp-sized format and supports onboard/ ceramic/ external antennas. The product also features low power consumption, small size, and strong anti-interference capability, and is suitable for various application scenarios.



Onboard antenna **Ceramic antenna** **External antenna** **Reverse view**

Figure 1.1：Product images

2 Product overview

2.1 Key features

Table 2-1：HJ-568IMH key characteristics

Characteristic	Description
Function	- Supports BLE 6.1, embeds low-power Bluetooth protocol stack and GATT services - Supports BLE master-slave integration (supports up to 20 connections, enables master and slave to work simultaneously without interfering with each other) - Supports data transmission, or provides bare modules for customers to develop their own firmware independently
supply voltage	1.71V ～ 3.6V
RF characteristics	- Working frequency: 2.4GHz, supporting free ISM band - Transmit power：-30dBm～+8dBm - High receiving sensitivity：-98dBm@1Mbps -95dBm@2Mbps - TX peak current＜4mA@0 dBm RX peak current＜3mA@1Mbps - Wireless transmission distance of the onboard/ceramic antenna in open areas:30~100m
Size	8.5*9.2*2.2mm
Packaging	Stamp hole packaging, with pad spacing of 0.8mm and 1mm
weight	0.16g
current	- Sleep current＜0.8μA(No RAM retention, wake on by GPIO) - Sleep current＜1.2μA(16K RAM retention, wake on by RTC) - Broadcasting current at a one-second interval 9μA(0dBm) or 12.5μA(+4dBm) - Broadcasting current at a 2-second interval 4.8μA(0dBm)
Operation temperature	-40 ～ +85℃
storage temperature	-40 ～ +85℃
GPIO Port	15（The RST pin is used as a GPIO.）
Product standards	BQB，FCC，CE，TELEC，ROHS

2.2 Application Scenarios

- 2.4GHz Bluetooth Low Energy system;
- Smart home, wireless remote control;
- Consumer electronics products such as sports and healthcare;
- Industrial monitoring;
- Intelligent transportation, etc.

2.3 Functional block diagram

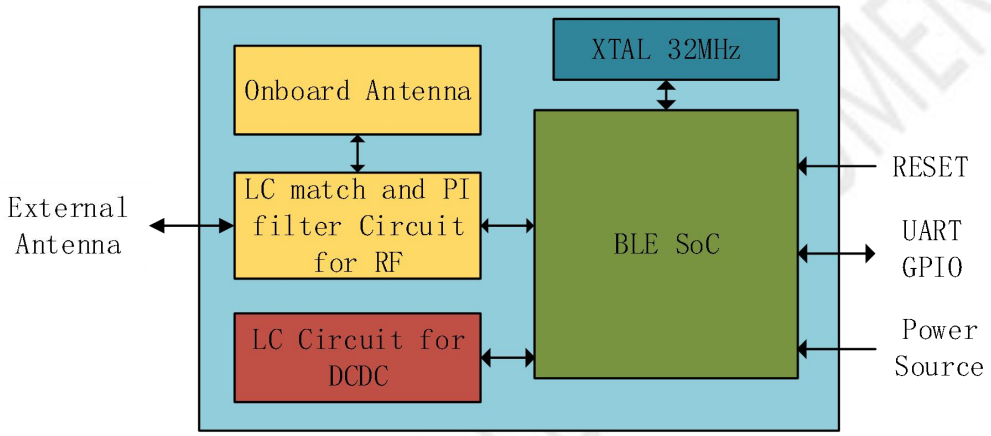


Figure 2.1：HJ-568IMH functional block diagram

2.4 Pins distribution diagram

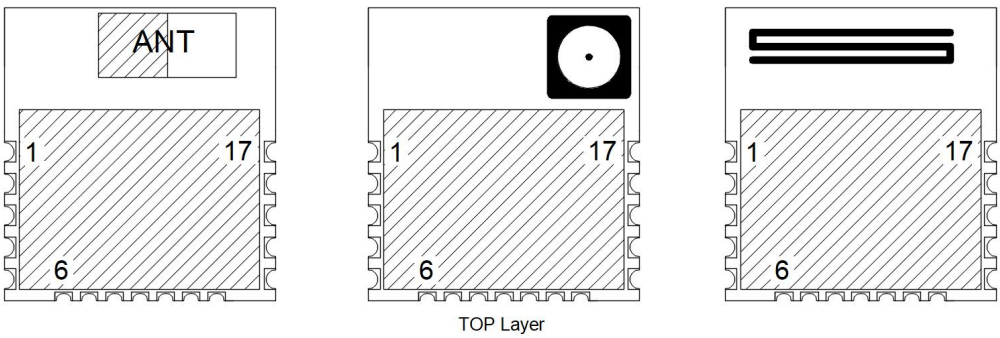


Figure 2.2：HJ-568IMH pins distribution diagram(front view)

2.5 Pins description table

Table 2-2：HJ-568IMH pins definition table

PIN	Name	Type	Description	Remarks
1	RST/P13	nReset Pin/IO	Low effective external reset	The reset function can be disabled and it can be used as an ordinary IO;

			Reset duration > 1s	When used for reset, it should be connected to an external IO. If not connected, a 1k strong pull-up resistor must be added.
2	P04	IO	GPIO	Can be customized with other functionsan
3	P21 (Connection status indication in slave mode)	OUTPUT	Connection status indication in slave mode	**The indicator pin for the connection status of the slave module** When the output is at a high level, the module has successfully been connected as a slave by the host; when the output is at a low level, the module has been disconnected from the host as a slave.
4	P01	IO	GPIO	Can be customized with other functionsan
5	P05	IO	GPIO	Can be customized with other functionsan
6	P10 (BLE-TX)	OUTPUT	UART-TX	**The serial port TX pin of the Bluetooth module** (In transparent transmission mode, this pin serves as the serial port's TX pin and is connected to the RX pin of the MCU)
7	P00	IO	GPIO	Can be customized with other functionsan
8	P09 (BLE-RX)	INPUT	UART-RX	**The serial port RX pin of the Bluetooth module** (When in transparent mode, this pin serves as the serial port's RX pin and is connected to the TX pin of the MCU)
9	P08 (UART RECEIVE ENABLE)	INPUT	Serial port receive enable pin	**Serial port receive enable pin** When the external needs to send serial port data to the BLE module, first change this pin from high level 1 to low level 0 to enable serial port receive. After at least 1ms after changing to low level 0, send data to the serial port. (If power consumption is not considered, this pin can be kept low 0 all the time), after the data transmission is completed, pull this pin high 1 to save power consumption.
10	P02 (UART flow control RTS)	OUTPUT (UART RTS)	Serial port flow control RTS	**BLE serial port flow control RTS output (Enables external serial port data transmission)** After enabling flow control, when the RTS pin outputs a low level of 0, it allows the external device to send data to the Bluetooth; When the RTS pin outputs a high level of 1, it indicates that the Bluetooth is busy and prohibits the external device from sending data to the Bluetooth;
11	P03 (APP_DATA_IND)	OUTPUT	APP data reception indicator	**APP data reception indicator pin** When the module receives data sent by the host or mobile phone, it will first raise this pin to maintain it at a high level for T1 (ms) (T1 can be set, with a range of 1-255ms, and the default is 20ms) before the data is output through the serial port. After the data transmission is completed, it will automatically become a low 0 level. Normally, this pin remaining at a low level represents an idle state. This pin is used as a wake-up signal for long connection low-power devices.
12	P07 (UART flow control CTS)	INPUT(UART CTS)	Serial port flow control CTS	**BLE serial port flow control CTS input (Enable BLE serial port to send data externally)** After enabling serial port flow control, when CTS is externally input with a low level of 0, the Bluetooth serial port allows data to be sent externally; When CTS is externally input with a high level of 1, the Bluetooth serial port stops sending data externally.
13	GND	Power	Power GND	/

		Ground		
14	VCC	Power In	Power supply	DC1.71V ~ 3.6V
15	P06	IO	GPIO	Can be customized with other functionsan
16	P22 (APP enables control)	INPUT Down	Enable APP configuration	**Enable APP configuration** When the pin is changed from a low level 0 to a high level 1, it enables the mobile APP to send instructions to configure all parameters of the module. When the pin is changed from a high level 1 to a low level 0, it prohibits the mobile APP from configuring or reading the parameters of the module. By default, this pin is set to pull-down mode.
17	P23 (Host mode connection status indicator)	OUTPUT	Host connection status indicator	**Host connection status indicator pin** When the output is a high voltage level, the module has successfully connected to the external slave device; when the output is a low voltage level, the module has disconnected from the external slave device.

3 RF Features

The module is equipped with an onboard/ ceramic/external antenna.

3.1 Antenna interface and operating frequency band

Module antenna interface characteristic impedance 50 ohms.

The radio frequency working band is 2.402 ~ 2.480GHz.

3.2 BLE RF features

Table 3-1：RF features

Attribute	Value	Remarks
Wireless modulation mode	GFSK	/
Frequency range	2.402 ~ 2.480GHz	bandwidth：2MHz
Number of channels	40	/
Air speed	1Mbps、2Mbps	/
Rf port impedance	50Ω	/
Transmission power	Max. +8dBm	/
Emission current	4mA (@0dBm,1Mbps)； 15mA (@8dBM,1Mbps)	/
Receive current	TYP. 3mA	1Mbps
Receiving sensitivity	-98dBm (@1Mbps) -95dBm (@2Mbps)	/
Antenna	onboard/ceramic/external antenna	/

4 Electrical Parameters

4.1 Absolute Maximum Ratings

Table 4-1：Absolute Maximum Ratings

Parameter	MIN	MAX	Unit
VCC	1.7	3.7	V

IO Supply Voltage	0	VCC	V
Operating Temperature	-40	+85	℃
Storage Temperature	-40	+85	℃

4.2 DC Characteristics

Table 4-2：Recommended Operating Conditions

Parameter	MIN	TYP	MAX	Unit
VCC	1.71	3.3	3.6	V
IO Supply Voltage	0	3.3	VCC	V
Dormant working current	/	＜2	/	μA
Maximum Operating Current	/	4.5	/	mA
Operating Temperature	-40	+25	+85	℃

Table 4-3：Dc features of I/O ports

IO Pin	Drive capability	Min	Max	Unit
Input low level	/	0	0.4	V
Input high level	/	1.8	VCC	V
Output low level	5mA	0	0.6	V
Output high level	5mA	1.8	VCC	V

Table 4-4：Electricity consumption

Test condition	TYP	Unit
sleep mode	＜2	μA
Broadcast at 20ms interval in slave mode	505	μA
Broadcast in slave mode with 1S gap	9	μA
Maintain the connection in slave mode with a 20ms connection gap	138	μA
Maintain the connection in slave mode with a 7.5ms connection gap	350	μA
Scan in host mode	5.1	mA
In host mode, the connection is maintained at a 20ms interval	150	μA

5 Notices for Hardware Design

- All IO ports can be used for connection.
- The module cannot be placed in a metal casing. If a metal casing is used, the antenna must be led out.
- The circuit board on each layer is fully copper-covered and connected to GND. Ensure that the copper-covered area is large enough (except for the antenna part), and ensure good grounding.
- For products installing this wireless module, metal components such as screws and inductors should be kept as far away as possible from the RF antenna part of the wireless module.
- Do not place other components on the antenna of the wireless module to prevent shielding from affecting the wireless performance.
- The wireless module should be placed on the sides of the motherboard as far as possible, and the antenna part should be close to the sides or corners of the motherboard. The motherboard PCB under the antenna of the module should be hollowed out with keepout layer. If can't

hollow out, copper or cable should not be placed under the antenna, otherwise,the RF performance will be affected.

- Please pay attention to checking the pin diagrams for all pins, and for the IOs connected to them, please pay attention to the IO mode and status.
- For input power that is not battery-powered and is powered by a mains power supply, it is recommended to use a magnetic bead or inductor for filtering.

6 Appoint

6.1 Protocol stack byte sequence specification

In the instructions for all functions listed in this document, all bytes use Big-end mode. All instruction characters are encoded using ASCII. All instructions begin with '<' and end with '>' without any return line breaks at the end.

All instructions are divided into two parts according to the direction of sending, one part is sent from the MCU to the BLE module, mainly responsible for setting and querying, using uppercase ASCII characters. The other part is sent from the BLE module to the MCU and is mainly responsible for answering and feedback, using lowercase ASCII characters.

For instructions sent from the MCU to the BLE module, the instructions with the setting function start with "<ST", and the instructions with the reading function start with "<RD".

For instructions sent from the BLE module to the MCU, the BLE module replies to the result of the execution of the setup instruction with an instruction beginning with "<st". The BLE module replies to the result of a read command with an instruction beginning with "<rd".

For example, if we set the BLE name to "HongJia", we need to send the command "<ST_NAME=HongJia>" and return "<st_name=ok>" after the setting is successful. For example, if the BLE name is "HongJia" and the command "<RD_NAME>" is sent, the command "<rd_name=HongJia>" is returned after read successed.

[Note: All instruction formats in this manual are sent as characters, including the angle brackets "< >" and all characters within them. Note that there is no carriage return, line feed, or any other character after ">".]

6.2 Noun interpretation

We can use the MCU to send instructions to the BLE module to control the BLE module. At the same time, the BLE module will process and send feedback information to the MCU after receiving instructions from the MCU. The instructions sent from the MCU to the BLE module constitute the instruction stream. Each instruction that reads or writes to the same property constitutes an instruction pack. Feedback information sent from the BLE module to the MCU constitutes a response stream. Each type of feedback to the same instruction packet constitutes a reply packet.

The syntax, functions, and parameters of instruction and reply packets are explained in detail in Chapters 9 and 10 of the HJ-568IMH software documentation.

7 Module working mode description

7.1 BLE working mode

The HJ-568IMH is equipped with a high-performance protocol stack, enabling it to implement any function of the BLE standard.

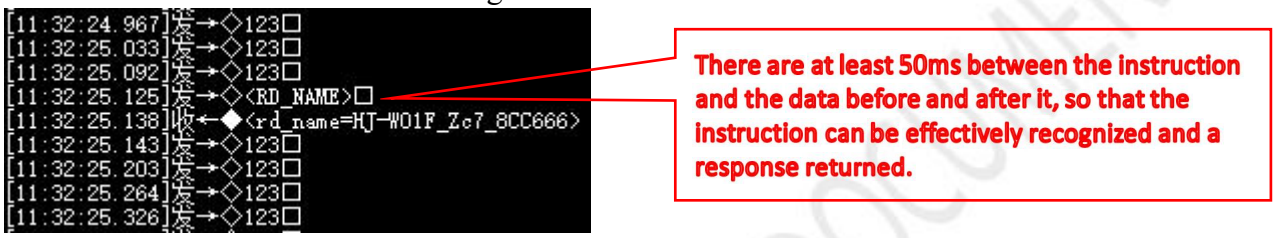
7.2 Data transparent mode and command mode

Adaptive for both instruction mode and data mode, automatically differentiating between data and instructions!

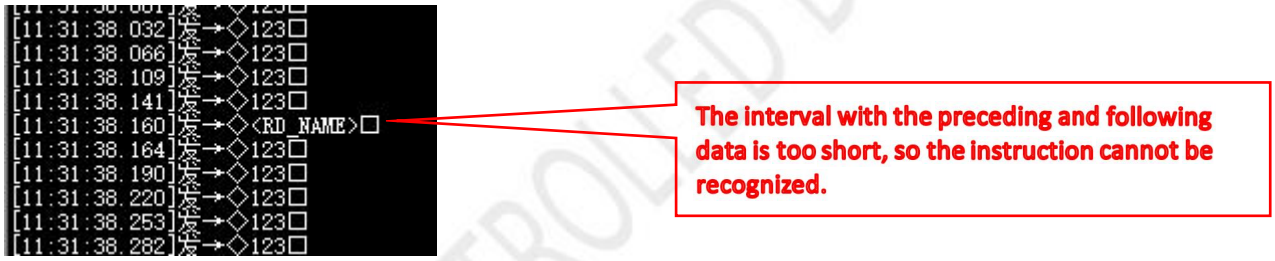
[Note1: Instructions and data are NOT automatically partitioned. There should be an interval of at least 50ms between instructions and data.]

Example：

Interval between instruction and data greater than 50ms：【Correct】

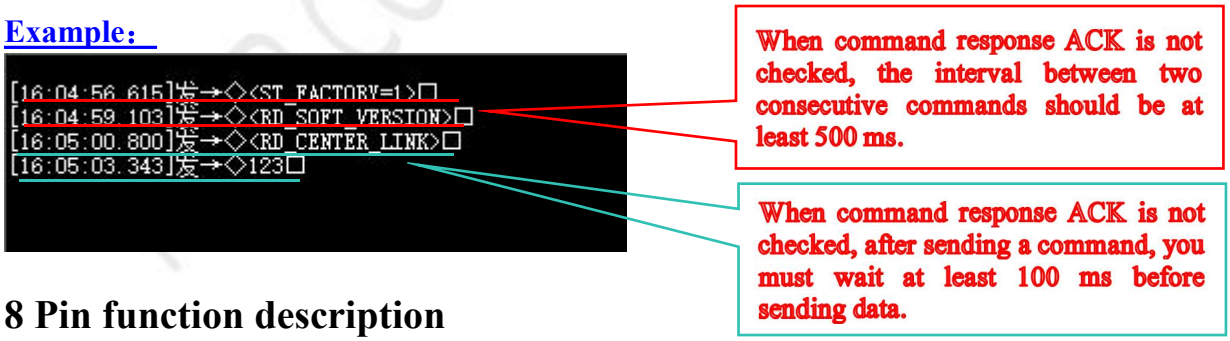


Interval between instruction and data too short：【Incorrect】



[Note2:If Ack responses are not checked for command acknowledgment, the interval between two consecutive commands must be at least 500 ms. If the interval is insufficient, the settings may fail.]

Example：



8 Pin function description

Please refer to chapter 2.5.

9 General and Slave-Related Commands

9.1 Set/Read Module English Name

Table 9.1-1：Set/Read Module English Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_NAME=xx..xx>
Read Command	<RD_NAME>
Description: 1. xx..xx is the module English name to be set, maximum 29 bytes. 2. Parameters are saved after successful setting. Example: To set the module name to BLE123, send <ST_NAME=BLE123>.	

Table 9.1-2：Set/Read Module English Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_name=ok>
Read Successful	<rd_name=xx..xx>
Description: 1. On successful write, <st_name=ok> is returned. 2. On successful read, <rd_name=xx..xx> is returned, where xx..xx is the current module English name.	

9.2 Start/Stop Advertising and Query

Table 9.2-1：Start/Stop Advertising and Query Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_ADV_ONOFF=x>
Read Command	<RD_ADV_ONOFF>
Description: 1. x=1 means enable advertising, x=0 means disable advertising. 2. If there is an existing slave connection when advertising is disabled, the connection will be actively disconnected first. 3. The advertising switch state is saved after successful setting. Example: To enable advertising, send <ST_ADV_ONOFF=1>; to disable advertising, send <ST_ADV_ONOFF=0>.	

Table 9.2-2：Start/Stop Advertising and Query Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_adv_onoff=ok>
Read Successful	<rd_adv_onoff=x>
Description: 1. On successful write, <st_adv_onoff=ok> is returned. 2. On successful read, <rd_adv_onoff=x> is returned, where x=1 means advertising enabled, x=0 means advertising disabled.	

9.3 Set/Read Advertising Data

Table 9.3-1：Set/Read Advertising Data Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_ADV_DATA=xx..xx>
Read Command	<RD_ADV_DATA>
Description: 1. xx..xx is the advertising data to be set. The maximum length is limited to 16 bytes. The advertising data uses characters (ASCII codes) to represent HEX data, so the number of characters in the command must be a multiple of 2.	

2. Chinese characters can be set, following UTF-8 encoding; one Chinese character occupies 3 bytes of HEX data, with a maximum of 5 Chinese characters.
Example: To set the advertising data to 0x12 0x13 0x14 0x15 0x16, send <ST_ADV_DATA=1213141516>.

Table 9.3-2：Set/Read Advertising Data Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_adv_data=ok>
Read Successful	<rd_adv_data=xx..xx>
Description: 1. On successful write, <st_adv_data=ok> is returned. 2. In the read response, xx..xx is the read advertising data, all in character form of HEX data. Note that 2 characters represent 1 byte of HEX data; pay attention to the conversion.	

9.4 Set/Read Advertising Interval

Table 9.4-1：Set/Read Advertising Interval Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_ADV_GAP=xxxx>
Read Command	<RD_ADV_GAP>
Description: 1. xxxx is the advertising interval, in ms. Setting range: 20~10000ms. 2. Parameters are saved after successful setting. If not currently connected, advertising will be restarted after successful setting to apply the new parameters. If currently connected, the existing connection will not be disconnected; the new advertising interval will take effect during subsequent re-advertising. Example: To set the advertising interval to 1000ms, send <ST_ADV_GAP=1000>.	

Table 9.4-2：Set/Read Advertising Interval Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_adv_gap=ok>
Read Successful	<rd_adv_gap=xxxx>
Description: 1. On successful write, <st_adv_gap=ok> is returned. 2. On successful read, <rd_adv_gap=xxxx> is returned, where xxxx is the current advertising interval, in ms.	

9.5 Slave Mode Active Disconnect / Query Connection Status

Table 9.5-1：Slave Mode Active Disconnect / Query Connection Status Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_LINK=0>
Read Command	<RD_CLIENT_LINK>
Description: 1. The write command is used to actively disconnect the current connection in slave mode.	

Table 9.5-2：Slave Mode Active Disconnect / Query Connection Status Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_link=ok> 或 <st_client_link=no_exist>
Read Successful	<rd_client_link=0> 或 <rd_client_link=mac,mtu>
Description:	

1. On successful disconnection, <st_client_link=ok> is returned.
2. If no connection currently exists, <st_client_link=no_exist> is returned.
3. Read response: if not connected, returns <rd_client_link=0>; if connected, returns <rd_client_link=mac,mtu>, where mac is the peer MAC address and mtu is the maximum number of bytes per data packet sent or received by the module connected to the master. Generally, the Bluetooth protocol specifies a packet size of 20 bytes, while our module extends this to a maximum of 160 bytes. The actual value of this parameter depends on the phone model and the performance of the Bluetooth module.

9.6 Set/Read Module Transmit Power

Table 9.6-1：Set/Read Module Transmit Power Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_TX_POWER=x>
Read Command	<RD_TX_POWER>
Description: 1. x is the transmit power value. Supported values: -47, -18, -9, -5, -1, 0, +0.5, +1, +1.5, +2, +2.5, +3, +3.5, +4, +4.5, +5, +5.5, +6, +7, +8, +8.5. 2. After successful setting, the current RF transmit power is updated immediately and the parameters are saved. Example: To set the transmit power to 0dBm, send <ST_TX_POWER=0>; to set it to +8.5dBm, send <ST_TX_POWER=+8.5>.	

Table 9.6-2：Set/Read Module Transmit Power Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_tx_power=ok>
Read Successful	<rd_tx_power=x>
Description: 1. On successful write, <st_tx_power=ok> is returned. 2. On successful read, <rd_tx_power=x> is returned, where x is the current saved transmit power value.	

9.7 Module Reset Command

Table 9.7-1：Module Reset Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_RESET_BLE>
Description: 1. After a successful response is sent, the module performs a BLE reset.	

Table 9.7-2：Module Reset Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_reset_ble=ok>
Description: 1. After <st_reset_ble=ok> is returned, the module enters the BLE reset process and resets after 50ms.	

9.8 Set/Read UART Baud Rat

Table 9.8-1：Set/Read UART Baud Rate Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_BAUD=x>

Read Command	<RD_BAUD>
Description: 1. x is the baud rate. Supported values: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600, 1000000, 1500000, 2000000. 2. After successful setting, parameters are saved and the UART parameter update flag is set. Example: To set the UART baud rate to 115200, send <ST_BAUD=115200>; to set it to 1000000bps, send <ST_BAUD=1000000>.	

Table 9.8-2：Set/Read UART Baud Rate Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_baud=ok>
Read Successful	<rd_baud=x>
Description: 1. On successful write, <st_baud=ok> is returned. 2. On successful read, <rd_baud=x> is returned, where x is the current UART baud rate.	

9.9 Set/Read Slave Minimum Connection Interval

Table 9.9-1：Set/Read Slave Minimum Connection Interval Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CON_MIN_GAP=xxxxx>
Read Command	<RD_CON_MIN_GAP>
Description: 1. xxxxx is the slave minimum connection interval, with the input unit being 0.1ms. 2. Setting range: 75 to 40000, corresponding to 7.5ms to 4000ms. 3. The minimum connection interval must be less than or equal to the current maximum connection interval; otherwise, the command will not take effect. 4. After successful setting, parameters are saved. If the slave is currently connected, the module will initiate a BLE connection parameter update request to the peer, causing the current connection to be renegotiated with the new connection interval and other parameters. 5. The read response returns <rd_con_min_gap=xxxxx>, with the return value also in 0.1ms units. Example: To set the minimum connection interval to 20ms, send <ST_CON_MIN_GAP=200> (in 0.1ms units).	

Table 9.9-2：Set/Read Slave Minimum Connection Interval Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_con_min_gap=ok>
Read Successful	<rd_con_min_gap=xxxxx>
Description: 1. On successful write, <st_con_min_gap=ok> is returned. 2. On successful read, <rd_con_min_gap=xxxxx> is returned, where xxxxx is the current minimum connection interval, in 0.1ms units. 3. If the parameter range or size relationship is not satisfied, no success response will be returned.	

9.10 Set/Read Slave Maximum Connection Interval

Table 9.10-1：Set/Read Slave Maximum Connection Interval Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CON_MAX_GAP=xxxxx>
Read Command	<RD_CON_MAX_GAP>
Description:	

1. xxxxx is the slave maximum connection interval, with the input unit being 0.1ms.
2. Setting range: 75 to 40000, corresponding to 7.5ms to 4000ms.
3. The maximum connection interval must be greater than or equal to the current minimum connection interval; otherwise, the command will not take effect.
4. After successful setting, parameters are saved. If the slave is currently connected, the module will initiate a BLE connection parameter update request to the peer, causing the current connection to be renegotiated with the new connection interval and other parameters.
5. The read response returns <rd_con_max_gap=xxxxx>, with the return value also in 0.1ms units.
Example: To set the maximum connection interval to 30ms, send <ST_CON_MAX_GAP=300> (in 0.1ms units).

Table 9.10-2：Set/Read Slave Maximum Connection Interval Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_con_max_gap=ok>
Read Successful	<rd_con_max_gap=xxxxx>
Description: 1. On successful write, <st_con_max_gap=ok> is returned. 2. On successful read, <rd_con_max_gap=xxxxx> is returned, where xxxxx is the current maximum connection interval, in 0.1ms units. 3. If the parameter range or size relationship is not satisfied, no success response will be returned.	

9.11 Set/Read Slave Connection Supervision Timeout

Table 9.11-1：Set/Read Slave Connection Supervision Timeout Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CON_TIMEOUT=xxxx>
Read Command	<RD_CON_TIMEOUT>
Description: 1. xxxx is the slave connection supervision timeout, in ms. 2. Setting range: 500 to 8000. 3. The connection supervision timeout must be greater than 2 times the current maximum connection interval; otherwise, the command will not take effect. 4. After successful setting, parameters are saved. If the slave is currently connected, the module will initiate a BLE connection parameter update request to the peer, causing the current connection to be renegotiated with the new connection interval and other parameters. Example: To set the connection supervision timeout to 3000ms, send <ST_CON_TIMEOUT=3000>.	

Table 9.11-2：Set/Read Slave Connection Supervision Timeout Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_con_timeout=ok>
Read Successful	<rd_con_timeout=xxxx>
Description: 1. On successful write, <st_con_timeout=ok> is returned. 2. On successful read, <rd_con_timeout=xxxx> is returned, where xxxx is the current connection supervision timeout, in ms. 3. If the parameter range or the relationship with the maximum connection interval is not satisfied, no success response will be returned.	

9.12 Set/Read Slave Latency

Table 9.12-1：Set/Read Slave Latency Command Packet Format

Command Stream Direction：MCU->BLE Module

Command Type	Command Format
Write Command	<ST_CON_LATENCY=xxx>
Read Command	<RD_CON_LATENCY>
Description: 1. xxx is the slave connection latency parameter. 2. Setting range: 0 to 255. Note: The relationship among connection supervision timeout, maximum connection interval, and latency must satisfy: Connection Supervision Timeout > Maximum Connection Interval × (LATENCY + 1) × 3, in ms. 3. After successful setting, parameters are saved. If the slave is currently connected, the module will initiate a BLE connection parameter update request to the peer, causing the current connection to be renegotiated with the new connection interval and other parameters. Example: To set the connection latency to 1, send <ST_CON_LATENCY=1>.	

Table 9.12-2：Set/Read Slave Latency Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_con_latency=ok>
Read Successful	<rd_con_latency=xxx>
Description: 1. On successful write, <st_con_latency=ok> is returned. 2. On successful read, <rd_con_latency=xxx> is returned, where xxx is the current latency parameter.	

9.13 Read Module BLE MAC Address

Table 9.13-1：Read Module BLE MAC Address Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Read Command	<RD_BLE_MAC>
Description: 1. mac is the module BLE MAC address, consisting of 12 HEX characters.	

Table 9.13-2：Read Module BLE MAC Address Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Read Successful	<rd_ble_mac=mac>
Description: 1. On successful read, <rd_ble_mac=mac> is returned, where mac is the module BLE MAC address, consisting of 12 HEX characters, returned in big-endian mode.	

9.14 Set/Read T1 Value

Table 9.14-1：Set/Read T1 Value Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_T1=xxx>
Read Command	<RD_T1>
Description: 1. xxx is the T1 time value, in ms. Setting range: 1~255ms. Example: To set T1 to 10ms, send <ST_T1=10>.	

Table 9.14-2：Set/Read T1 Value Command Packet Response Format

Response Stream Direction：BLE Modele->MCU

Response Type	Response Format
Write Successful	<st_t1=ok>
Read Successful	<rd_t1=xxx>

Description:
1. On successful write, <st_t1=ok> is returned.
2. On successful read, <rd_t1=xxx> is returned, where xxx is the current T1 parameter value.
3. When the module receives data sent from the mobile APP or an external device connected to this module, the BLE module needs to send the data out through the module's UART TX pin. Regardless of whether the module acts as a master or slave, after this pin is pulled high for T1 (T1 is configurable, range 1-255, corresponding to 10-2550ms), the data will be sent out through the module's UART TX pin. After the data transmission is completed, the pin is automatically pulled low. Normally, this pin remains low to indicate idle state. This pin is used as a wake-up indicator for long-connection low-power devices.

9.15 Set/Cancel User-Defined MAC Address

Table 9.15-1：Set/Cancel User-Defined MAC Address Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Set User-Defined MAC	<ST_OWN_MAC=mac>
Cancel User-Defined MAC	<ST_OWN_MAC=0>

Description:
1. mac must be 12 HEX characters.
2. <ST_OWN_MAC=0> means cancel the user-defined MAC.
3. After successful setting or cancellation, parameters are saved.
4. After the new MAC address is set, the module will automatically restart and then advertise using the new user-defined MAC address. After canceling the user-defined MAC address, the module will revert to the default MAC address.

Table 9.15-2：Set/Cancel User-Defined MAC Address Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_own_mac=ok>

Description:
1. On successful setting or cancellation, <st_own_mac=ok> is returned.

9.16 Set Temporary MAC Address

Table 9.16-1：Set Temporary MAC Address Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_TEMP_MAC=mac>

Description:
1. mac must be 12 HEX characters.
2. This command only sets a temporary MAC address, which will be automatically erased after reset or power-on, restoring the original MAC address.

Table 9.16-2：Set Temporary MAC Address Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_temp_mac=ok>

Description:
1. On successful setting, <st_temp_mac=ok> is returned.

9.17 Set/Read UART Parity Check Mode

Table 9.17-1：Set/Read UART Parity Check Mode Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_UART_PARITY=x>
Read Command	<RD_UART_PARITY>
Description: 1. x supports 0, 1, 2. 0 means no parity; 2 means odd parity; 1 means even parity. 2. After successful setting, parameters are saved and the UART parameter update flag is set. Example: To set no parity, send <ST_UART_PARITY=0>; even parity, send <ST_UART_PARITY=1>; odd parity, send <ST_UART_PARITY=2>.	

Table 9.17-2：Set/Read UART Parity Check Mode Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_uart_parity=ok>
Read Successful	<rd_uart_parity=x>
Description: 1. On successful write, <st_uart_parity=ok> is returned. 2. On successful read, <rd_uart_parity=x> is returned, where x is the current UART parity mode.	

9.18 Set/Read APP_DATA_READY Delay Time

Table 9.18-1：Set/Read APP_DATA_READY Delay Time Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_AD_WAIT=xxx>
Read Command	<RD_AD_WAIT>
Description: 1. xxx is the APP_DATA_READY delay wait time, in ms. Setting range: 1~255ms. Example: To set it to 100ms, send <ST_AD_WAIT=100>. 2. After successful setting, parameters are saved. 3. <ST_T1=xxx>: Sets the wait time before APP data is sent. <ST_AD_WAIT=xxx>: Sets the wait time after APP data transmission is completed, used to confirm that the data has been fully sent before ending the transmission process.	

Table 9.18-2：Set/Read APP_DATA_READY Delay Time Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_ad_wait=ok>
Read Successful	<rd_ad_wait=xxx>
Description: 1. On successful write, <st_ad_wait=ok> is returned. 2. On successful read, <rd_ad_wait=xxx> is returned, where xxx is the current APP_DATA_READY delay time, in ms.	

9.19 Set/Read UART RX Internal Pull-Up Switch

Table 9.19-1：Set/Read UART RX Internal Pull-Up Switch Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_UART_PULL=x>
Read Command	<RD_UART_PULL>
Description: 1. x=1 means enable UART RX internal pull-up (input pull-up resistance is approximately 200-300KΩ), x=0	

means disable.
2. After successful setting, parameters are saved and take effect immediately.
Example: To enable UART RX internal pull-up, send <ST_UART_PULL=1>; to disable, send <ST_UART_PULL=0>.

Table 9.19-2：Set/Read UART RX Internal Pull-Up Switch Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_uart_pull=ok>
Read Successful	<rd_uart_pull=x>

Description:
1. On successful write, <st_uart_pull=ok> is returned.
2. On successful read, <rd_uart_pull=x> is returned, where x=1 means pull-up enabled, x=0 means disabled.

9.20 Factory Reset

Table 9.20-1：Factory Reset Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_FACTORY=1>

Description:
1. After execution, the module restores default parameters.

Table 9.20-2：Factory Reset Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_factory=ok>

Description:
1. On successful factory reset, <st_factory=ok> is returned.

9.21 Set/Read Chinese Bluetooth Name

Table 9.21-1：Set/Read Chinese Bluetooth Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CH_NAME=xx..xx>
Read Command	<RD_CH_NAME>

Description:
1. xx..xx is the ASCII character code of the HEX data of the Chinese name to be set. Since Chinese characters follow UTF-8 encoding, 1 Chinese character occupies 3 bytes of HEX data. Therefore, when setting a Chinese name, the number of bytes must be a multiple of 3. A maximum of 9 Chinese characters can be set, because a total of 29 bytes can be set, and 9 Chinese characters occupy 27 bytes.
2. For example, the HEX data corresponding to the Chinese name "宏佳电子" is "e5ae8fe4bdb3e794b5e5ad90", then directly send "<ST_CH_NAME=e5ae8fe4bdb3e794b5e5ad90>". You may refer to the website "https://tool.lu/hexstr/" for conversion testing.

Table 9.21-2：Set/Read Chinese Bluetooth Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_ch_name=ok>
Read Successful	<rd_ch_name=xx..xx>

Description:
1. On successful write, <st_ch_name=ok> is returned.
2. On successful read, <rd_ch_name=xx..xx> is returned, where xx..xx is the ASCII character code of the HEX

data.

9.22 Set/Read UART Enable Pin Active Level

Table 9.22-1：Set/Read UART Enable Pin Active Level Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_WAKEUP_HIGH=x>
Read Command	<RD_WAKEUP_HIGH>
Description: 1. To set the UART enable pin function to active high, set x to 1. 2. To set the UART enable pin function to active low, set x to 0.	

Table 9.22-2：Set/Read UART Enable Pin Active Level Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_wakeup_high=ok>
Read Successful	<rd_wakeup_high=x>
Description: 1. On successful write, <st_wakeup_high=ok> is returned. 2. On successful read, <rd_wakeup_high=x> is returned, where x is the current UART enable pin active level setting.	

9.23 Set/Read 16-bit Format Slave Primary Service UUID

Table 9.23-1：Set/Read 16-bit Format Slave Primary Service UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_16BIT_MAJOR_UUID=xxxx>
Read Command	<RD_CLIENT_16BIT_MAJOR_UUID>
Description: 1. xxxx is the character form of the 2-byte slave primary service UUID HEX data. 2. After successful setting, parameters are saved and the mode switches to 16-bit UUID mode. Example: To set the slave service UUID to FFF0, send <ST_CLIENT_16BIT_MAJOR_UUID=FFF0>.	

Table 9.23-2：Set/Read 16-bit Format Slave Primary Service UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_16bit_major_uuid=ok>
Read Successful	<rd_client_16bit_major_uuid=xxxx>
Description: 1. On successful write, <st_client_16bit_major_uuid=ok> is returned. 2. On successful read, <rd_client_16bit_major_uuid=xxxx> is returned, where xxxx is the current 16-bit primary service UUID.	

9.24 Set/Read 16-bit Format Slave Notify Channel UUID

Table 9.24-1：Set/Read 16-bit Format Slave Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_16BIT_NOTIFY_UUID=xxxx>
Read Command	<RD_CLIENT_16BIT_NOTIFY_UUID>
Description:	

1. xxxx is the character form of the 2-byte slave notify channel UUID HEX data.
2. After successful setting, parameters are saved and the mode switches to 16-bit UUID mode.
Example: To set the slave notify UUID to FFF1, send <ST_CLIENT_16BIT_NOTIFY_UUID=FFF1>.

Table 9.24-2：Set/Read 16-bit Format Slave Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_16bit_notify_uuid=ok>
Read Successful	<rd_client_16bit_notify_uuid=xxxx>
Description: 1. On successful write, <st_client_16bit_notify_uuid=ok> is returned. 2. On successful read, <rd_client_16bit_notify_uuid=xxxx> is returned, where xxxx is the current 16-bit notify channel UUID.	

9.25 Set/Read 16-bit Format Slave Write Channel UUID

Table 9.25-1：Set/Read 16-bit Format Slave Write Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_16BIT_WRITE_UUID=xxxx>
Read Command	<RD_CLIENT_16BIT_WRITE_UUID>
Description: 1. xxxx is the character form of the 2-byte slave write channel UUID HEX data. 2. After successful setting, parameters are saved and the mode switches to 16-bit UUID mode. Example: To set the slave write UUID to FFF2, send <ST_CLIENT_16BIT_WRITE_UUID=FFF2>.	

Table 9.25-2： Set/Read 16-bit Format Slave Write Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_16bit_write_uuid=ok>
Read Successful	<rd_client_16bit_write_uuid=xxxx>
Description: 1. On successful write, <st_client_16bit_write_uuid=ok> is returned. 2. On successful read, <rd_client_16bit_write_uuid=xxxx> is returned, where xxxx is the current 16-bit write channel UUID.	

9.26 Set/Read 128-bit Format Slave Primary Service UUID

Table 9.26-1：Set/Read 128-bit Format Slave Primary Service UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_128BIT_MAJOR_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CLIENT_128BIT_MAJOR_UUID>
Description: 1. UUID is the character form of 16-byte HEX data, consisting of 32 HEX characters. 2. After successful setting, parameters are saved and the mode switches to 128-bit UUID mode. Example: To set the 128-bit slave service UUID, send <ST_CLIENT_128BIT_MAJOR_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 9.26-2：Set/Read 128-bit Format Slave Primary Service UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_128bit_major_uuid=ok>

Read Successful	<rd_client_128bit_major_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. On successful write, <st_client_128bit_major_uuid=ok> is returned. 2. On successful read, <rd_client_128bit_major_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> is returned, where the return value is the current 128-bit primary service UUID.	

9.27 Set/Read 128-bit Format Slave Notify Channel UUID

Table 9.27-1：Set/Read 128-bit Format Slave Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_128BIT_NOTIFY_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CLIENT_128BIT_NOTIFY_UUID>
Description: 1. UUID is the character form of 16-byte HEX data, consisting of 32 HEX characters. 2. After successful setting, parameters are saved and the mode switches to 128-bit UUID mode. Example: To set the 128-bit slave notify UUID, send <ST_CLIENT_128BIT_NOTIFY_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 9.27-2：Set/Read 128-bit Format Slave Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_128bit_notify_uuid=ok>
Read Successful	<rd_client_128bit_notify_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. On successful write, <st_client_128bit_notify_uuid=ok> is returned. 2. On successful read, <rd_client_128bit_notify_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> is returned, where the return value is the current 128-bit notify channel UUID.	

9.28 Set/Read 128-bit Format Slave Write Channel UUID

Table 9.28-1：Set/Read 128-bit Format Slave Write Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLIENT_128BIT_WRITE_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CLIENT_128BIT_WRITE_UUID>
Description: 1. UUID is the character form of 16-byte HEX data, consisting of 32 HEX characters. 2. After successful setting, parameters are saved and the mode switches to 128-bit UUID mode. Example: To set the 128-bit slave write UUID, send <ST_CLIENT_128BIT_WRITE_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 9.28-2：Set/Read 128-bit Format Slave Write Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_client_128bit_write_uuid=ok>
Read Successful	<rd_client_128bit_write_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. On successful write, <st_client_128bit_write_uuid=ok> is returned. 2. On successful read, <rd_client_128bit_write_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> is returned, where the return value is the current 128-bit write channel UUID.	

9.29 Set/Read High-Speed Mode

Table 9.29-1：Set/Read High-Speed Mode Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_HIGH_SPEED=x>
Read Command	<RD_HIGH_SPEED>
Description: 1. x=1 means enable high-speed mode, x=0 means disable high-speed mode. 2. After successful setting, parameters are saved. **[Note: The following data rates can only be achieved when there is a single connection!]** 3. If the mobile phone supports BLE 5.0 or above, the high-speed mode can be enabled via command. After enabling BLE high-speed mode, the communication rate can be increased to Android @ ~70KB/S, Apple @ ~60KB/S (bytes). 4. In low-speed mode, our communication rate can still reach 30KB/S, so if the rate is sufficient, please use low-speed mode, because high-speed mode sacrifices communication distance. Choose according to your needs.	

Table 9.29-2：Set/Read High-Speed Mode Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_high_speed=ok>
Read Successful	<rd_high_speed=x>
Description: 1. On successful write, <st_high_speed=ok> is returned. 2. On successful read, <rd_high_speed=x> is returned, where x is the current high-speed mode switch state.	

9.30 Set/Read Pairing Passkey

Table 9.30-1：Set/Read Pairing Passkey Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_PASSKEY=xxxxxx> 或 <ST_PASSKEY=0>
Read Command	<RD_PASSKEY>
Description: 1. When setting Passkey, it must be a 6-digit number. 2. <ST_PASSKEY=0> means disable pairing Passkey. Example: To set the pairing password to 123456, the command format is: <ST_PASSKEY=123456>.	

Table 9.30-2：Set/Read Pairing Passkey Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_passkey=ok>
Read Successful	<rd_passkey=xxxxxx> 或 <rd_passkey=null>
Description: 1. On successful setting or disabling, <st_passkey=ok> is returned. 2. When reading a set Passkey, <rd_passkey=xxxxxx> is returned. 3. When no Passkey is set, <rd_passkey=null> is returned.	

9.31 Set/Read Connection Secret Key

Table 9.31-1：Set/Read Connection Secret Key Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_SECRET=xx..xx>

Read Command	<RD_SECRET>
Description: 1. xx..xx is the password to be set, in character format, with a maximum length of 8 bytes. For example, to set the password to "123456", the setting command is "<ST_SECRET=123456>".	

Table 9.31-2：Set/Read Connection Secret Key Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_secret=ok>
Read Successful	<rd_secret=xx..xx> 或 <rd_secret=null>
Description: 1. On successful write, <st_secret=ok> is returned. 2. When reading a set connection secret key, <rd_secret=xx..xx> is returned. 3. When no connection secret key is set, <rd_secret=null> is returned. When the password is empty, there is no password authentication process during connection.	

9.32 Clear Connection Secret Key

Table 9.32-1：Clear Connection Secret Key Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_CLEAR_SECRET=1>
Description: 1. Used to clear the saved connection secret key. 2. After successful clearing, the connection secret key length is set to 0 and parameters are saved. When the password is empty, there is no password authentication process during connection.	

Table 9.32-2：Clear Connection Secret Key Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_clear_secret=ok>
Description: 1. On successful clearing, <st_clear_secret=ok> is returned.	

9.33 Set/Read Connection Secret Key Timeout

Table 9.33-1：Set/Read Connection Secret Key Timeout Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	Command Format
Write Command	<ST_SECRET_TIMEOUT=xxx>
Read Command	<RD_SECRET_TIMEOUT>
Description: 1. xxx is the connection secret key confirmation timeout. Its value represents the timeout in seconds. Setting range: 1~255, corresponding to 1~255s. For example, to set the timeout to 50s, the write command is "<ST_SECRET_TIMEOUT=50>".	

Table 9.33-2：Set/Read Connection Secret Key Timeout Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	Response Format
Write Successful	<st_secret_timeout=ok>
Read Successful	<rd_secret_timeout=xxx>
Description: 1. On successful write, <st_secret_timeout=ok> is returned.	

2. On successful read, <rd_secret_timeout=xxx> is returned, where xxx is the current connection secret key confirmation timeout.

9.34 Set/Read Configuration Secret Key

Table 9.34-1：Set/Read Configuration Secret Key Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CFG_SECRET=xx..xx>
Read Command	<RD_CFG_SECRET>
Description: 1. xx..xx is the password to be set, with a maximum of 8 characters. The default configuration password is "123456". For example, to set the password to "562105", send the command <ST_CFG_SECRET=562105>. 2. After the password is set, when configuring the BLE, the command format is "password,configuration command", with a comma separating the password and the configuration command. For example, if the configuration password is set to "562105" and you want to "read the Bluetooth English name", send the command "562105,<RD_NAME>". 3. Hardware pin APP configuration enable, command APP configuration enable, and setting APP configuration command password are three independent APP configuration enable methods, which do not affect each other and can all independently complete APP command configuration enable.	

Table 9.34-2：Set/Read Configuration Secret Key Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_cfg_secret=ok>
Read Successful	<rd_cfg_secret=xx..xx>
Description: 1. On successful write, <st_cfg_secret=ok> is returned. 2. On successful read, <rd_cfg_secret=xx..xx> is returned, where xx..xx is the current configuration secret key content.	

9.35 Set/Read Extended Advertising Data

Table 9.35-1：Set/Read Extended Advertising Data Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_EXT_ADV_DATA=xx..xx>
Read Command	<RD_EXT_ADV_DATA>
Description: 1. xx..xx is the extended advertising data, with a maximum length of 220 bytes. 2. After setting, the extended advertising data flag is enabled and the data is saved. 3. If not currently connected, advertising will be stopped and the advertising data update will be triggered. If currently connected, the existing connection will not be disconnected; the parameters will be saved and take effect during subsequent re-advertising. 4. When extended advertising data is not enabled, the read response returns <rd_ext_adv_data=disable>.	

Table 9.35-2：Set/Read Extended Advertising Data Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_ext_adv_data=ok>
Read Successful	<rd_ext_adv_data=xx..xx> 或 <rd_ext_adv_data=disable>
Description: 1. On successful write, <st_ext_adv_data=ok> is returned.	

2. When extended advertising data is enabled, <rd_ext_adv_data=xx..xx> is returned.
3. When extended advertising data is not enabled, <rd_ext_adv_data=disable> is returned.

9.36 Set/Read UART Hardware Flow Control

Table 9.36-1：Set/Read UART Hardware Flow Control Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_UART_FC=x>
Read Command	<RD_UART_FC>
Description: 1. x=1 means enable RTS/CTS hardware flow control, x=0 means disable. Example: To enable RTS/CTS hardware flow control, send <ST_UART_FC=1>; to disable, send <ST_UART_FC=0>.	

Table 9.36-2：Set/Read UART Hardware Flow Control Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_uart_fc=ok>
Read Successful	<rd_uart_fc=x>
Description: 1. On successful write, <st_uart_fc=ok> is returned. 2. On successful read, <rd_uart_fc=x> is returned, where x is the current RTS/CTS hardware flow control switch state.	

9.37 Set/Read Slave Indicate Switch

Table 9.37-1：Set/Read Slave Indicate Switch Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CLIENT_INDICATE=x>
Read Command	<RD_CLIENT_INDICATE>
Description: 1. Set whether the slave-side data reporting method enables Indicate. x=0 means disable Indicate and use Notify; x=1 means enable Indicate. On successful setting, <st_client_indicate=ok> is returned. The parameters are saved and the module will restart to take effect. Example: To enable indicate, send <ST_CLIENT_INDICATE=1>; to disable, send <ST_CLIENT_INDICATE=0>.	

Table 9.37-2：Set/Read Slave Indicate Switch Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_client_indicate=ok>
Read Successful	<rd_client_indicate=x>
Description: 1. On successful write, <st_client_indicate=ok> is returned. 2. On successful read, <rd_client_indicate=x> is returned, where x is the current indicate switch state.	

9.38 Read Software Version

Table 9.38-1：Read Software Version Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**

Read Command	<RD_SOFT_VERSION>
Description: 1. Returns the current firmware software version number.	

Table 9.38-2：Read Software Version Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Read Successful	<rd_soft_version=xxx>
Description: 1. On successful read, <rd_soft_version=xxx> is returned. The version number is defined by the current firmware code.	

10 Master-Related Commands

10.1 Set/Read 16-bit Format Master Mode Primary Service UUID

Table 10.1-1：Set/Read 16-bit Format Master Mode Primary Service UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_16BIT_MAJOR_UUID=xxxx>
Read Command	<RD_CENTER_16BIT_MAJOR_UUID>
Description: 1. xxxx is the HEX data character ASCII code of the primary service UUID to be set. The 16-bit length is 2 bytes. 2. For example, if the primary service UUID is 0xFFF0, send <ST_CENTER_16BIT_MAJOR_UUID=FFF0>. Example: To set the primary service UUID to FFF0, send <ST_CENTER_16BIT_MAJOR_UUID=FFF0>.	

Table 10.1-2：Set/Read 16-bit Format Master Mode Primary Service UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_16bit_major_uuid=ok>
Read Successful	<rd_center_16bit_major_uuid=xxxx>
Description: 1. xxxx is the 2-byte UUID in HEX data character form.	

10.2 Set/Read 16-bit Format Master Mode Notify Channel UUID

Table 10.2-1：Set/Read 16-bit Format Master Mode Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_16BIT_NOTIFY_UUID=xxxx>
Read Command	<RD_CENTER_16BIT_NOTIFY_UUID>
Description: xxxx is the HEX data character ASCII code of the notify channel UUID to be set. The 16-bit length is 2 bytes. Example: To set the master notify UUID to FFF1, send <ST_CENTER_16BIT_NOTIFY_UUID=FFF1>.	

Table 10.2-2：Set/Read 16-bit Format Master Mode Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU

Response Type	Response Format
Write Successful	<st_center_16bit_notify_uuid=ok>
Read Successful	<rd_center_16bit_notify_uuid=xxxx>
Description: 1. xxxx is the 2-byte UUID in HEX data character form.	

10.3 Set/Read 16-bit Format Master Mode Write Channel UUID

Table 10.3-1：Set/Read 16-bit Format Master Mode Write Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_16BIT_WRITE_UUID=xxxx>
Read Command	<RD_CENTER_16BIT_WRITE_UUID>
Description: 1. xxxx is the HEX data character ASCII code of the write channel UUID to be set. The 16-bit length is 2 bytes. Example: To set the master write UUID to FFF2, send <ST_CENTER_16BIT_WRITE_UUID=FFF2>.	

Table 10.3-2：Set/Read 16-bit Format Master Mode Write Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_16bit_write_uuid=ok>
Read Successful	<rd_center_16bit_write_uuid=xxxx>
Description: 1. xxxx is the 2-byte UUID in HEX data character form.	

10.4 Set/Read 16-bit Format Master Mode Second Notify Channel UUID

Table 10.4-1：Set/Read 16-bit Format Master Mode Second Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_16BIT_2NOTIFY_UUID=xxxx>
Read Command	<RD_CENTER_16BIT_2NOTIFY_UUID>
Description: 1.xxxx is the HEX data character ASCII code of the second notify channel UUID to be set. The 16-bit length is 2 bytes. Example: To set the second notify UUID to FFF3, send <ST_CENTER_16BIT_2NOTIFY_UUID=FFF3>.	

Table 10.4-2：Set/Read 16-bit Format Master Mode Second Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_16bit_2notify_uuid=ok>
Read Successful	<rd_center_16bit_2notify_uuid=xxxx>
Description: 1. xxxx is the 2-byte UUID in HEX data character form.	

10.5 Set/Read 128-bit Format Master Mode Primary Service UUID

Table 10.5-1：Set/Read 128-bit Format Master Mode Primary Service UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_128BIT_MAJOR_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CENTER_128BIT_MAJOR_UUID>
Description: 1. xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx is the HEX data character ASCII code of the primary service UUID to be set. The 128-bit length is 16 bytes. Example: To set the 128-bit primary service UUID, send <ST_CENTER_128BIT_MAJOR_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 10.5-2：Set/Read 128-bit Format Master Mode Primary Service UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_128bit_major_uuid=ok>
Read Successful	<rd_center_128bit_major_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. The return value is the 16-byte UUID in HEX data character form.	

10.6 Set/Read 128-bit Format Master Mode Notify Channel UUID

Table 10.6-1：Set/Read 128-bit Format Master Mode Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_128BIT_NOTIFY_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CENTER_128BIT_NOTIFY_UUID>
Description: 1. xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx is the HEX data character ASCII code of the notify channel UUID to be set. The 128-bit length is 16 bytes. Example: To set the 128-bit master notify UUID, send <ST_CENTER_128BIT_NOTIFY_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 10.6-2：Set/Read 128-bit Format Master Mode Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_128bit_notify_uuid=ok>
Read Successful	<rd_center_128bit_notify_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. The return value is the 16-byte UUID in HEX data character form.	

10.7 Set/Read 128-bit Format Master Mode Write Channel UUID

Table 10.7-1：Set/Read 128-bit Format Master Mode Write Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module

Command Type	Command Format
Write Command	<ST_CENTER_128BIT_WRITE_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CENTER_128BIT_WRITE_UUID>
Description: 1. xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx is the HEX data character ASCII code of the write channel UUID to be set. The 128-bit length is 16 bytes. Example: To set the 128-bit master write UUID, send <ST_CENTER_128BIT_WRITE_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 10.7-2：Set/Read 128-bit Format Master Mode Write Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_128bit_write_uuid=ok>
Read Successful	<rd_center_128bit_write_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. The return value is the 16-byte UUID in HEX data character form.	

10.8 Set/Read 128-bit Format Master Mode Second Notify Channel UUID

Table 10.8-1：Set/Read 128-bit Format Master Mode Second Notify Channel UUID Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_128BIT_2NOTIFY_UUID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Read Command	<RD_CENTER_128BIT_2NOTIFY_UUID>
Description: 1. xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx is the HEX data character ASCII code of the second notify channel UUID to be set. The 128-bit length is 16 bytes. Example: To set the 128-bit second notify UUID, send <ST_CENTER_128BIT_2NOTIFY_UUID=00112233445566778899AABBCCDDEEFF>.	

Table 10.8-2：Set/Read 128-bit Format Master Mode Second Notify Channel UUID Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_128bit_2notify_uuid=ok>
Read Successful	<rd_center_128bit_2notify_uuid=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
Description: 1. The return value is the 16-byte UUID in HEX data character form.	

10.9 Set/Read Master Mode Write with Response

Table 10.9-1：Set/Read Master Mode Write with Response Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_WR_RSP=x>
Read Command	<RD_WR_RSP>

Description:
1. x=0 means write without response.
2. x=1 means write with response.
Example: To use write without response, send <ST_WR_RSP=0>; to use write with response, send <ST_WR_RSP=1>.

Table 10.9-2：Set/Read Master Mode Write with Response Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_wr_rsp=ok>
Read Successful	<rd_wr_rsp=x>

Description:
1. x is the current write response setting state, with a value of 0 or 1.

10.10 Set/Read Master Mode Accept Slave Connection Parameter Update

Table 10.10-1：Set/Read Master Mode Accept Slave Connection Parameter Update Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_EN_CLIENT_UPDATE=x>
Read Command	<RD_EN_CLIENT_UPDATE>

Description:
1. x=0 means do not accept slave connection parameter update; the module updates the connection according to the master-side set parameters.
2. x=1 means accept slave connection parameter update.

Table 10.10-2：Set/Read Master Mode Accept Slave Connection Parameter Update Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_en_client_update=ok>
Read Successful	<rd_en_client_update=x>

Description:
1. x is the current setting state, with a value of 0 or 1.
2. After the master accepts the parameter request from an external slave, the master-side connection interval and timeout settings in master mode will become invalid, and the master will connect and communicate using the connection parameters reported by the slave. If the slave's connection parameters are very small, the module's power consumption will increase.
3. After the master rejects the parameter request from an external slave, the master will use its own set parameters. In this way, regardless of the slave's parameters, everything is determined by the master. This allows precise control of system power consumption without being affected by the slave's parameters.

10.11 Set/Read Master Mode Connection Supervision Timeout

Table 10.11-1：Set/Read Master Mode Connection Supervision Timeout Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_CON_TIMEOUT=xxxx>

Read Command	<RD_CENTER_CON_TIMEOUT>
Description: 1. xxxx is the connection supervision timeout, in ms. 2. Setting range: 500 to 8000. 3. The set value must be greater than 2 times the current maximum connection interval. Example: To set the master connection supervision timeout to 3000ms, send <ST_CENTER_CON_TIMEOUT=3000>.	

Table 10.11-2：Set/Read Master Mode Connection Supervision Timeout Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_con_timeout=ok>
Read Successful	<rd_center_con_timeout=xxxx>
Description: 1. xxxx is the current connection supervision timeout, in ms.	

10.12 Set/Read Master Mode Minimum Connection Interval

Table 10.12-1：Set/Read Master Mode Minimum Connection Interval Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_CON_MIN_GAP=xxxxx>
Read Command	<RD_CENTER_CON_MIN_GAP>
Description: 1. xxxxx is the minimum connection interval, with the input unit being 0.1ms. 2. Setting range: 75 to 40000, i.e., 7.5ms to 4000ms. 3. The set value must be less than or equal to the current maximum connection interval. Example: To set the master minimum connection interval to 20ms, send <ST_CENTER_CON_MIN_GAP=200> (in 0.1ms units).	

Table 10.12-2：Set/Read Master Mode Minimum Connection Interval Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_con_min_gap=ok>
Read Successful	<rd_center_con_min_gap=xxxx>
Description: 1. xxxx is the current minimum connection interval, in ms.	

10.13 Set/Read Master Mode Maximum Connection Interval

Table 10.13-1：Set/Read Master Mode Maximum Connection Interval Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_CON_MAX_GAP=xxxxx>
Read Command	<RD_CENTER_CON_MAX_GAP>
Description: 1. xxxxx is the maximum connection interval, with the input unit being 0.1ms.	

2. Setting range: 75 to 40000, i.e., 7.5ms to 4000ms.
3. The set value must be greater than or equal to the current minimum connection interval.
Example: To set the master maximum connection interval to 30ms, send <ST_CENTER_CON_MAX_GAP=300> (in 0.1ms units).

Table 10.13-2：Set/Read Master Mode Maximum Connection Interval Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_con_max_gap=ok>
Read Successful	<rd_center_con_max_gap=xxxx>
Description: 1. xxxx is the current maximum connection interval, in ms.	

10.14 Set/Read Proximity Bonding RSSI Threshold

Table 10.14-1：Set/Read Proximity Bonding RSSI Threshold Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_NEAR_RSSI=xxx>
Read Command	<RD_NEAR_RSSI>
Description: 1. x is the proximity bonding RSSI threshold to be set, in character format, range 1~255 (in -dB). When the actual RSSI value of the detected slave is greater than the set value, the slave will be automatically connected. For example, to set the RSSI value to 35, the command is "<ST_NEAR_RSSI=35>". Example: To set the proximity bonding RSSI threshold to 70, send <ST_NEAR_RSSI=70>.	

Table 10.14-2：Set/Read Proximity Bonding RSSI Threshold Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_near_rssi=ok>
Read Successful	<rd_near_rssi=xxx>
Description: 1. xxx is the current proximity bonding RSSI threshold.	

10.15 Set/Read Proximity Bonding Switch

Table 10.15-1：Set/Read Proximity Bonding Switch Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_NEAR_BOND=x>
Read Command	<RD_NEAR_BOND>
Description: 1. x indicates whether to enable proximity bonding mode. 1 means enable; 0 means disable. 2. After enabling proximity bonding mode, the module will automatically turn on the scanning function (automatically set <ST_SCAN_ONOFF=1>), and then start searching for slaves that meet the RSSI threshold and automatically connect and bond. 3. To exit the automatic bonding mode, send the command <ST_BOND_CLIENT_MAC=0> to unbind.	

Table 10.15-2：Set/Read Proximity Bonding Switch Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_near_bond=ok>
Cancel Successful	<st_near_bond=cancel>
Read Successful	<rd_near_bond=x>
Description: 1. x is the current proximity bonding state, with a value of 0 or 1.	

10.16 Set/Read Master Mode Scanning Switch

Table 10.16-1：Set/Read Master Mode Scanning Switch Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_SCAN_ONOFF=x>
Read Command	<RD_SCAN_ONOFF>
Description: 1. x=1 means enable scanning. 2. x=0 means disable scanning. 2. x=0 means disable scanning. Example: To enable scanning, send <ST_SCAN_ONOFF=1>; to disable scanning, send <ST_SCAN_ONOFF=0>.	

Table 10.16-2：Set/Read Master Mode Scanning Switch Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_scan_onoff=ok>
Read Successful	<rd_scan_onoff=x>
Description: 1. x is the current scanning state. 1 means enabled, 0 means disabled.	

10.17 Set/Read Scan Duplicate Filter Switch

Table 10.17-1：Set/Read Scan Duplicate Filter Switch Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_SCAN_FILTER=x>
Read Command	<RD_SCAN_FILTER>
Description: 1. x=1 means enable scan duplicate filtering. 2. x=0 means disable scan duplicate filtering. Example: To enable scan duplicate filtering, send <ST_SCAN_FILTER=1>; to disable, send <ST_SCAN_FILTER=0>.	

Table 10.17-2：Set/Read Scan Duplicate Filter Switch Command Packet Response Format

Response Stream Direction：BLE Modele->MCU

Response Type	Response Format
Write Successful	<st_scan_filter=ok>
Read Successful	<rd_scan_filter=x>

Description:
1. x is the current scan duplicate filtering state.

10.18 Set/Read Master Mode Auto-Connect to Surrounding Slaves

Table 10.18-1：Set/Read Master Mode Auto-Connect to Surrounding Slaves Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_AUTO_CON=x>
Read Command	<RD_CENTER_AUTO_CON>

Description:
1. This command is enabled by default with x=1. When x=1, the module automatically connects to surrounding slaves after power-on; when x=0, it does not automatically connect after power-on.
2. After enabling the central auto-connect function, the module will continuously scan and attempt to connect to eligible slaves (specifically Hongjia Bluetooth slaves; other slaves are not supported). If connection fails or times out, the auto-connect function remains enabled until <ST_CENTER_AUTO_CON=0> is sent to disable it.
3. This function is only valid when the master scanning function is enabled (command <ST_SCAN_ONOFF=1>)!
4. When the master has manually bonded with a slave or bonded via the simple bonding function, the auto-connect function will be temporarily disabled, and the BLE module will connect according to the bonding information.
Example: To enable master auto-connect to slaves, send <ST_CENTER_AUTO_CON=1>; to disable, send <ST_CENTER_AUTO_CON=0>.

Table 10.18-2：Set/Read Master Mode Auto-Connect to Surrounding Slaves Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Write Successful	<st_center_auto_con=ok>
Read Successful	<rd_center_auto_con=x>
Connecting	<st_auto_con_client_mac=connecting>
Connection Timeout	<st_auto_con_client_mac=timeout>
Connection Successful	<st_auto_con_client_mac=xxxxxxxxxxxx>
Connection Disconnected	<st_auto_con_client_mac=disconnect>
Match Failed	<st_auto_con_client_mac=notmatch>

Description:
1. x is the current auto-connect state.
2. During the central auto-connect process, if the discovered or attempted slave does not match the auto-connect conditions, <st_auto_con_client_mac=notmatch> is returned, e.g., when the target device MAC/connection information verification is inconsistent. After this return, the current connection attempt is judged as a mismatch and will not be treated as a successful connection.

10.19 Disconnect / Read Master Mode Connection Status

Table 10.19-1：Disconnect / Read Master Mode Connection Status Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_CENTER_LINK=0>
Read Command	<RD_CENTER_LINK>
Description: 1. The write command is used to disconnect the current master mode connection or cancel the ongoing connection/discovery process.	

Table 10.19-2：Disconnect / Read Master Mode Connection Status Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Disconnect Successful	<st_center_link=ok>
Connection Does Not Exist	<st_center_link=no_exist>
Not Connected	<rd_center_link=0>
Connected	<rd_center_link=mac,mtu>
Description: 1. mac is the connected slave MAC address, consisting of 12 HEX characters, returned in big-endian mode. 2. mac is the connected slave MAC address, consisting of 12 HEX characters, returned in big-endian mode. 3. If the central auto-connect / connect by specified device name / connect by MAC address bonding function is still enabled, the module will continue scanning and may automatically reconnect after disconnection. To prevent reconnection after disconnection, please send the corresponding scan enable switch or unbind command first.	

10.20 Set/Read Scan Surrounding Bluetooth Device Advertising Data Information

Table 10.20-1：Set/Read Scan Surrounding Bluetooth Device Advertising Data Information Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_SCAN_DEVICE_ON=x>
Read Command	<RD_SCAN_DEVICE_ON>
Description: 1. x=0 means disable the module's function of scanning surrounding devices. 2. x=1 means enable the module's function of scanning surrounding devices, and the command executes continuous scanning for 10 seconds and then stops. 3. x=2 means enable scanning device information output and continue scanning until <ST_SCAN_DEVICE_ON=0> is sent to stop. Example: To enable scanning device information reporting, send <ST_SCAN_DEVICE_ON=1>; to cancel, send <ST_SCAN_DEVICE_ON=0>.	

Table 10.20-2：Set/Read Scan Surrounding Bluetooth Device Advertising Data Information Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_scan_device_on=ok>
Cancel Successful	<st_scan_device_on=cancel>
Read Successful	<rd_scan_device_on=x>

Description:
1. x is the current scanning device information output state, with a value of 0, 1, or 2.
2. When the module enables the function of scanning surrounding devices, the Bluetooth module will continuously scan surrounding devices for 10 seconds. The scanned data format is: header + data identifier ADV_DATA= + scanned MAC address + scanned advertising data + RSSI value + tail.
3. Example. Scanned data is as follows:
Send→◇<ST_SCAN_DEVICE_ON=1>□
Receive←◆<st_scan_device_on=ok>
Receive←◆<ADV_DATA=784128c58150,1eff060001092002be6a5637d9a17d346b8227a05fdc6c0d6364e08f019a2e,-47>
Receive←◆<ADV_DATA=5ba86175d80c,02011a0aff4c001005031ca38ffc,-60>
Receive←◆<ADV_DATA=5ba86175d80c,02011a0aff4c001005031ca38ffc,-55>
Receive←◆<ADV_DATA=a4da32d1f375,0201061aff4c00021574278bdab64445208f0c720eaf0599350120d81ac5,-81>
Receive←◆<ADV_DATA=64c62c5bc7d6,02011a0aff4c001005031c2adca5,-51>

10.21 Set/Read Scan Surrounding Bluetooth Device Name

Table 10.21-1：Set/Read Scan Surrounding Bluetooth Device Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_SCAN_NAME_ON=x>
Read Command	<RD_SCAN_NAME_ON>

Description:
1. x=0 means cancel scanning surrounding Bluetooth device names.
2. x=1 means enable scanning surrounding Bluetooth device names, and the command executes continuous scanning for 10 seconds and then stops.
3. x=2 means enable scanning surrounding Bluetooth device names and continue scanning until <ST_SCAN_NAME_ON=0> is sent to stop.
Example: To enable scanning name reporting, send <ST_SCAN_NAME_ON=1>; to cancel, send <ST_SCAN_NAME_ON=0>.

Table 10.21-2：Set/Read Scan Surrounding Bluetooth Device Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_scan_name_on=ok>
Cancel Successful	<st_scan_name_on=cancel>
Read Successful	<rd_scan_name_on=x>

Description:
1. x is the current scanning name output state, with a value of 0, 1, or 2.
2. When the module enables the function of scanning surrounding devices, the Bluetooth module will continuously scan surrounding devices for 10 seconds. The scanned data format is: header + data identifier NAME_DATA= + scanned MAC address + scanned BLE device name + RSSI value + tail.
3. Example. Scanned data is as follows:
[13:53:17.678]Send→◇<ST_SCAN_NAME_ON=1>□
[13:53:17.695]Receive←◆<st_scan_name_on=ok>
[13:53:17.755]Receive←◆<NAME_DATA=016d6b00670b,Mgc_Ibc3,-74>
[13:53:18.546]Receive←◆<NAME_DATA=016d6b006b05,Mgc_Ibc3,-62>
[13:53:19.046]Receive←◆<NAME_DATA=016d6b006b05,Mgc_Ibc3,-66>
[13:53:19.418]Receive←◆<NAME_DATA=806fb01fefaa,Sce859f634579a935C,-74>
[13:53:19.556]Receive←◆<NAME_DATA=016d6b006b05,Mgc_Ibc3,-63>

[13:53:22.686]Receive←◆<NAME_DATA=806fb01fefaa,Sce859f634579a935C,-92>

10.22 Set/Cancel Temporary MAC Connection

Table 10.22-1：Set/Cancel Temporary MAC Connection Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_TMP_CON_CLIENT_MAC=mac>
Cancel Command	<ST_TMP_CON_CLIENT_MAC=0>
Description: 1. mac is the target slave MAC address, consisting of 12 HEX characters. 2. Temporary MAC connection will not be saved as a bonded MAC. 3. After successful setting, the master module will actively connect to the slave with the specified MAC address. Regardless of whether the connection succeeds or fails, the connected slave will not be saved. After power-on reset or disconnection, it will not automatically reconnect. If reconnection is needed, this command must be sent again.	

Table 10.22-2：Set/Cancel Temporary MAC Connection Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_tmp_con_client_mac=ok>
Connecting	<st_tmp_con_client_mac=connecting>
Connection Timeout	<st_tmp_con_client_mac=timeout>
Connection Successful	<st_tmp_con_client_mac=mac>
Connection Disconnected	<st_tmp_con_client_mac=disconnect>
Match Failed	<st_tmp_con_client_mac=notmatch>
Description: 1. mac is the MAC address of the successfully connected slave, consisting of 12 HEX characters.	

10.23 Set/Cancel Connect by Printer Name

Table 10.23-1：Set/Cancel Connect by Printer Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_PRINT_NAME_CON=name>
Cancel Command	<ST_PRINT_NAME_CON=0>
Description: 1. name is the target printer name filter string, with a maximum length of 149 bytes. It is case-sensitive and in character format. For example, if the printer name to connect is "HP1008", the command format is: <ST_PRINT_NAME_CON=HP1008>.	

Table 10.23-2：Set/Cancel Connect by Printer Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_print_name_con=ok>
Connecting	<st_print_name_con=connecting>

Connection Timeout	<st_print_name_con=timeout>
Connection Successful	<st_print_name_con=name>
Connection Disconnected	<st_print_name_con=disconnect>
Description: 1. name is the name of the successfully connected device.	

10.24 Set/Read/Cancel Connect by Bluetooth Name

Table 10.24-1：Set/Read/Cancel Connect by Bluetooth Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_NAME_CON=name>
Read Command	<RD_NAME_CON>
Cancel Command	<ST_NAME_CON=0>
Description: 1. name is the target Bluetooth name, in character format, with a maximum length of 29 bytes. For example, if the device name to connect is "BLE123", the command format is: <ST_NAME_CON=BLE123>. Example: To connect by name BLE123, send <ST_NAME_CON=BLE123>; to cancel name connection, send <ST_NAME_CON=0>.	

Table 10.24-2：Set/Read/Cancel Connect by Bluetooth Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_name_con=ok>
Read Successful	<rd_name_con=name> 或 <rd_name_con=0>
Connecting	<st_name_con=connecting>
Connection Timeout	<st_name_con=timeout>
Connection Successful	<st_name_con=name>
Connection Disconnected	<st_name_con=disconnect>
Match Failed	<st_name_con=notmatch>
Description: 1. name is the currently set or successfully connected Bluetooth name. 2. <rd_name_con=0> indicates that connect by name is not set.	

10.25 Initiate Service Discovery Connection by MAC

Table 10.25-1：Initiate Service Discovery Connection by MAC Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_DISCOVERY_DEVICE_MAC=mac>
Description: 1. mac is the target slave MAC address, consisting of 12 HEX characters. 2. After connecting to the target device, this command outputs the discovered GATT service information through the UART. 3. The command output format is detailed in Appendix 1.	

4. Example:
First scan surrounding devices, then pick a device to connect and discover services:
[00:23:44.660]Send→◇<ST_SCAN_NAME_ON=1>□
[00:23:44.678]Receive←◆<st_scan_name_on=ok><NAME_DATA=f1220129d2c4,AIMA-29D2C4,-81><NAME_DATA=d134741418c5,Q_BASH_D134741418C5,-90><NAME_DATA=f12201422452,AIMA-422452,-86>
[00:23:45.569]Receive←◆<NAME_DATA=ee29d350a31b,PD EM,-47>
[00:23:47.434]Receive←◆<NAME_DATA=ac822646a107,U-ACA106,-96>
[00:23:54.679]Receive←◆<st_scan_name_on=timeout>

[00:24:06.665]Send→◇<ST_DISCOVERY_DEVICE_MAC=ee29d350a31b>□
[00:24:06.684]Receive←◆<st_discovery_device_mac=ok>

[00:24:07.624]Receive←◆<disc_service:uuid16=0x1800,name=generic_access>

[00:24:07.805]Receive←◆<disc_char:handle=0x0003,uuid16=0x2a00,name=device_name,props=write_req|read,desc=null>

[00:24:07.890]Receive←◆<disc_char:handle=0x0005,uuid16=0x2a01,name=appearance,props=read,desc=null>

[00:24:07.980]Receive←◆<disc_char:handle=0x0007,uuid16=0x2a04,name=peripheral_preferred_connection_parameters,props=read,desc=null>
<disc_char:handle=0x0009,uuid16=0x2aa6,name=central_address_resolution,props=read,desc=null>

[00:24:08.070]Receive←◆<disc_service:uuid16=0x1801,name=generic_attribute>

[00:24:08.345]Receive←◆<disc_char:handle=0x000c,uuid16=0x2a05,name=service_changed,props=ind,desc=0x000d.0x2902.client_characteristic_configuration>

[00:24:08.430]Receive←◆<disc_service:uuid16=0x180a,name=device_information>

[00:24:08.614]Receive←◆<disc_char:handle=0x0010,uuid16=0x2a29,name=manufacturer_name_string,props=read,desc=null>

[00:24:08.699]Receive←◆<disc_char:handle=0x0012,uuid16=0x2a24,name=model_number_string,props=read,desc=null>

[00:24:08.791]Receive←◆<disc_char:handle=0x0014,uuid16=0x2a25,name=serial_number_string,props=read,desc=null>

[00:24:08.880]Receive←◆<disc_char:handle=0x0016,uuid16=0x2a27,name=hardware_revision_string,props=read,desc=null>

[00:24:08.970]Receive←◆<disc_char:handle=0x0018,uuid16=0x2a26,name=firmware_revision_string,props=read,desc=null>

[00:24:09.059]Receive←◆<disc_char:handle=0x001a,uuid16=0x2a28,name=software_revision_string,props=read,desc=null>

[00:24:09.150]Receive←◆<disc_char:handle=0x001c,uuid16=0x2a23,name=system_id,props=read,desc=null>
<disc_char:handle=0x001e,uuid16=0x2a50,name=pnp_id,props=read,desc=null>

[00:24:09.239]Receive←◆<disc_service:uuid16=0xfff0,name=null>

[00:24:09.516]Receive←◆<disc_char:handle=0x0021,uuid16=0xfff1,name=null,props=ntf,desc=0x0022.0x290

2.client_characteristic_configuration|0x0023.0x2901.characteristic_user_description>

[00:24:09.695]Receive←◆<disc_char:handle=0x0025,uuid16=0xfff2,name=null,props=write_req|write_cmd|read,desc=0x0026.0x2901.characteristic_user_description>

[00:24:09.966]Receive←◆<disc_char:handle=0x0028,uuid16=0xfff3,name=null,props=ntf|write_req|write_cmd|read,desc=0x0029.0x2902.client_characteristic_configuration|0x002a.0x2901.characteristic_|truncated>
<disc_done>

Table 10.25-2：Initiate Service Discovery Connection by MAC Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_discovery_device_mac=ok>
Busy	<st_discovery_device=busy>
Invalid Parameter	<st_discovery_device=invalid>

Description:
1. busy is returned when discovery is not allowed to start.
2. invalid is returned when MAC format is incorrect or discovery start fails.

10.26 Initiate Service Discovery Connection by Name

Table 10.26-1：Initiate Service Discovery Connection by Name Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_DISCOVERY_DEVICE_NAME=name>

Description:
1. name is the target Bluetooth name, maximum 31 bytes.
2. After connecting to the target device, this command outputs the discovered GATT service information through the UART.
3. The command output format is detailed in Appendix 1.
4. Example:
First scan surrounding devices, then pick a device to connect and discover services:
[00:19:45.904]Send→◇<ST_SCAN_NAME_ON=1>□
[00:19:45.923]Receive←◆<st_scan_name_on=ok><NAME_DATA=d134741418c5,Q_BASH_D134741418C5,-93>
[00:19:46.022]Receive←◆<NAME_DATA=f12201422452,AIMA-422452,-85>
[00:19:46.749]Receive←◆<NAME_DATA=f1220129d2c4,AIMA-29D2C4,-86>
[00:19:47.672]Receive←◆<NAME_DATA=bc89f87206f7,midea,-95>
[00:19:55.925]Receive←◆<st_scan_name_on=timeout>
[00:20:23.975]Send→◇<ST_DISCOVERY_DEVICE_NAME=AIMA-422452>□
[00:20:24.000]Receive←◆<st_discovery_device_name=ok>

[00:20:24.131]Receive←◆<disc_service:uuid16=0x1800,name=generic_access>

[00:20:24.298]Receive←◆<disc_char:handle=0x0003,uuid16=0x2a00,name=device_name,props=read,desc=null>
<disc_char:handle=0x0005,uuid16=0x2a01,name=appearance,props=read,desc=null>

[00:20:24.382]Receive←◆<disc_service:uuid16=0x1801,name=generic_attribute>

[00:20:24.582]Receive←◆<disc_char:handle=0x0008,uuid16=0x2a05,name=service_changed,props=ind,desc=0x0009.0x2902.client_characteristic_configuration>

[00:20:24.653]Receive←◆<disc_char:handle=0x000b,uuid16=0x2b29,name=null,props=write_req|read,desc=null>
<disc_char:handle=0x000d,uuid16=0x2b2a,name=null,props=read,desc=null>
<disc_service:uuid16=0x180a,name=device_information>

[00:20:24.851]Receive←◆<disc_char:handle=0x0010,uuid16=0x2a50,name=pnp_id,props=read,desc=null>

[00:20:24.937]Receive←◆<disc_service:uuid16=0xff78,name=null>

[00:20:25.107]Receive←◆<disc_char:handle=0x0013,uuid16=0x7800,name=null,props=ntf|write_req,desc=0x0014.0x2902.client_characteristic_configuration>

[00:20:25.302]Receive←◆<disc_char:handle=0x0016,uuid16=0x7801,name=null,props=ntf|write_req,desc=0x0017.0x2902.client_characteristic_configuration>

[00:20:25.387]Receive←◆<disc_char:handle=0x0019,uuid16=0x7802,name=null,props=ntf|write_req,desc=0x001a.0x2902.client_characteristic_configuration>

[00:20:25.511]Receive←◆<disc_char:handle=0x001c,uuid16=0x7803,name=null,props=ntf,desc=0x001d.0x2902.client_characteristic_configuration>

[00:20:25.721]Receive←◆<disc_char:handle=0x001f,uuid16=0x7804,name=null,props=ntf|write_req,desc=0x0020.0x2902.client_characteristic_configuration>
<disc_service:uuid16=0x2600,name=null>

[00:20:25.931]Receive←◆<disc_char:handle=0x0023,uuid16=0x7001,name=null,props=write_req|write_cmd,desc=null>

[00:20:26.002]Receive←◆<disc_char:handle=0x0025,uuid16=0x7002,name=null,props=ntf,desc=0x0026.0x2902.client_characteristic_configuration>
<disc_service:uuid16=0xff79,name=null>

[00:20:26.232]Receive←◆<disc_char:handle=0x0029,uuid16=0x7901,name=null,props=ntf|write_req,desc=0x002a.0x2902.client_characteristic_configuration>

[00:20:26.411]Receive←◆<disc_service:uuid16=0x7810,name=null>

[00:20:26.682]Receive←◆<disc_char:handle=0x002d,uuid16=0x0002,name=null,props=ntf|write_req,desc=0x002e.0x2902.client_characteristic_configuration>

[00:20:26.847]Receive←◆<disc_char:handle=0x0030,uuid16=0x0003,name=null,props=ntf|write_req,desc=0x0031.0x2902.client_characteristic_configuration>
<disc_service:uuid16=0x1812,name=human_interface_device>

[00:20:27.027]Receive←◆<disc_char:handle=0x0034,uuid16=0x2a4a,name=hid_information,props=read,desc=null>
<disc_char:handle=0x0036,uuid16=0x2a4c,name=hid_control_point,props=write_cmd,desc=null>

[00:20:27.162]Receive←◆<disc_char:handle=0x0038,uuid16=0x2a4b,name=report_map,props=read,desc=null>

[00:20:27.327]Receive←◆<disc_char:handle=0x003a,uuid16=0x2a33,name=boot_mouse_input_report,props=ntf|read,desc=0x003b.0x2902.client_characteristic_configuration>

[00:20:27.448]Receive←◆<disc_char:handle=0x003d,uuid16=0x2a4d,name=report,props=ntf|read,desc=0x003e.0x2908.report_reference|0x003f.0x2902.client_characteristic_configuration>

[00:20:27.658]Receive←◆<disc_done>

Table 10.26-2：Initiate Service Discovery Connection by Name Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful	<st_discovery_device_name=ok>
Busy	<st_discovery_device=busy>
Invalid Parameter	<st_discovery_device=invalid>
Description: 1. busy is returned when discovery is not allowed to start. 2. invalid is returned when name length is incorrect or discovery start fails.	

10.27 Set/Read/Cancel Bonded MAC Connection

Table 10.27-1：Set/Read/Cancel Bonded MAC Connection Command Packet Format

Command Stream Direction：MCU->BLE Module	
Command Type	**Command Format**
Write Command	<ST_BOND_CLIENT_MAC=mac>
Read Command	<RD_BOND_CLIENT_MAC>
Cancel Command	<ST_BOND_CLIENT_MAC=0>
Description: 1. mac is the target slave MAC address, consisting of 12 HEX characters. 2. After setting, the bonded MAC is saved and scanning connection is enabled.	

Table 10.27-2：Set/Read/Cancel Bonded MAC Connection Command Packet Response Format

Response Stream Direction：BLE Modele->MCU	
Response Type	**Response Format**
Set Successful / Cancel Successful	<st_bond_client_mac=ok>
Read Successful	<rd_bond_client_mac=mac>
Connecting	<st_bond_client_mac=connecting>
Connection Timeout	<st_bond_client_mac=timeout>
Connection Successful	<st_bond_client_mac=mac>
Connection Disconnected	<st_bond_client_mac=disconnect>
Match Failed	<st_bond_client_mac=notmatch>
Description: 1. mac is the MAC address of the bonded or successfully connected slave, consisting of 12 HEX characters. 2. After setting, the module will automatically turn on the scanning function (automatically set <ST_SCAN_ONOFF=1>), and then start searching for the MAC address to be bonded until connection is successful. 3. After this function is enabled, when the module is not connected to the slave, the module power consumption will reach about 13mA. After successful connection, the power consumption will drop significantly. If the connection interval is large enough, the power consumption can be reduced. 4. On successful write, <st_bond_client_mac=ok> is returned. If only this command is returned without a connection response, it means "the device with this MAC address was not found." If the device with this MAC address is found, <st_bond_client_mac=connecting> is returned. If connection is not established within the set time, <st_bond_client_mac=timeout> is returned.	

11 BLE Module Default Parameters

- UART RX pin pull-up enabled

- Default UART baud rate：115200bps，N 8 1，no parity

- Default transmit power：+0dBm

- Module normal advertising enabled

- Slave default advertising interval：1000ms

- Slave default minimum connection interval：20ms

- Slave default maximum connection interval：20ms

- Slave default connection supervision timeout：3s

- Number of masters that can connect to the module as a slave：1 个
- T1 default value: 5ms
- Default wait time after data transmission is completed: 100ms
- Default slave service UUID: 0xFFF0; slave WRITE write channel UUID: 0xFFF2; slave NOTIFY notification channel UUID: 0xFFF1
- Default master service UUID: 0xFFF0; master WRITE write channel UUID: 0xFFF2; master NOTIFY notification channel UUID: 0xFFF1
- The Bluetooth module defaults to low-speed mode. To enable high-speed mode, please set accordingly.

12 Antenna

Below chart shows a few options of external antenna which has been tested and approved to use with HJ-568IMH.

#	Photo	Model No.	Type	Gain	Spec	Certification
1		HJ2400-XLJ261001WQ	External	3.04dBi	Download from“https://www.HJSIP.com.cn”	FCC,CE,TELEC
2		HJ2400-BA261001TQ	External	2dBi	Download from“https://www.HJSIP.com.cn”	FCC,CE,TELEC
3		HJ2400-FPC2075FSK	PCB	≤2.91dBi	Download from“https://www.HJSIP.com.cn”	FCC,CE,TELEC

13 Machine Dimension

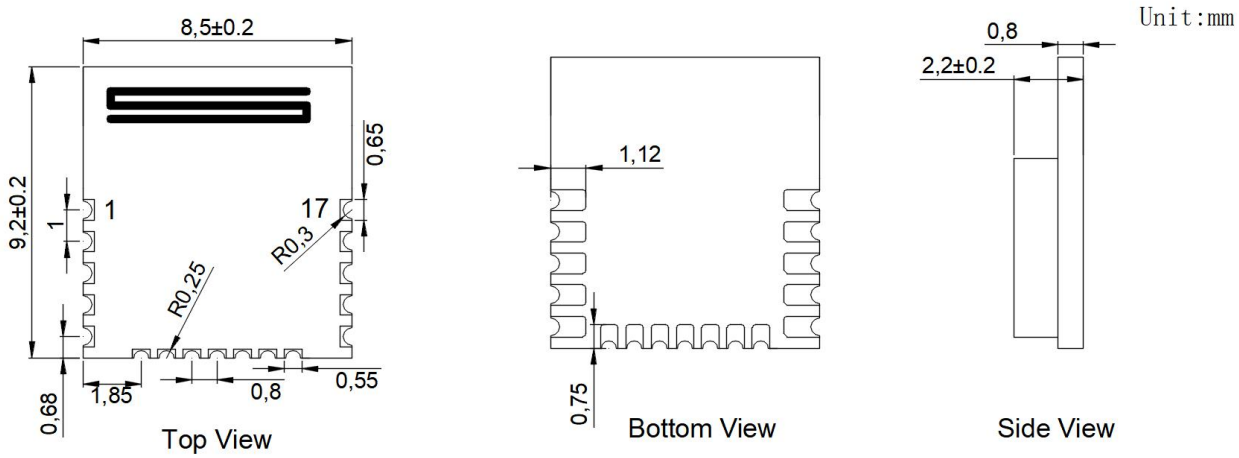


Figure 13.1:HJ-568IMH dimension

Note:

The mechanical dimension of different types of antenna modules are the same.

14 SMT production

14.1 Precautions for ultrasonic welding

Please carefully consider using ultrasonic welding technology. If it is necessary to use ultrasonic welding technology, please use 40KHz high frequency ultrasound welding technology. Keep the module away from the ultrasonic soldering line and the fixing column during the design method to prevent damage to the module!

For specific ultrasonic welding matters, please contact our company for technical consultation.

14.2 Soldering Recommendations

HJ-580XP module use high temperature resistant materials, manufacturing by Lead-free Process. The maximum temperature resistance is 265℃. Ten continuous reflow soldering has no effect on properties and strength. Specific parameters as shown in Table 14-1.

Table 14-1：Reflow soldering parameters

Parameter	Value
Features	Lead-free process
Average ramp up rate(T_{SMAX} to Tp)	max3℃/sec. max
Temperature Min($T_S$min)	150℃
Temperature Max(TSmax)	200℃
Preheat time (Min to Max) (tS)	80~100sec
Peak Temperature (T_P)	250±5℃
Ramp-down Rate	6℃/sec. max
Time 25℃ to Peak Temp (T_P)	8 min. max

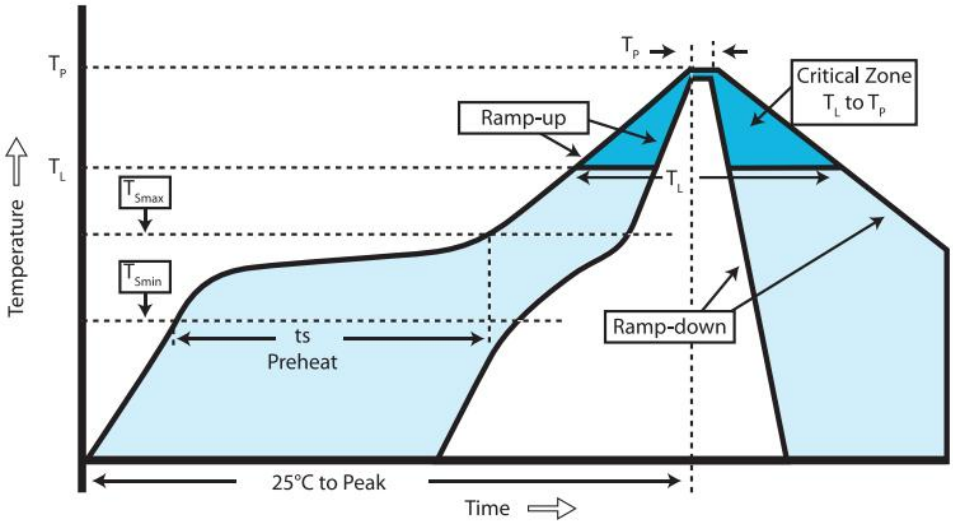


Figure 14.1：Temperature Curve of Reflow Welding

14.3 Humidity sensing properties

The HJ-568IMH module has a humidity sensitivity of level 3.

If any of the following two conditions are met, the HJ-568IMH module should be fully baked prior before reflow soldering, otherwise the module may cause permanent damage during reflow soldering.

- After unpacking or vacuum packaging is damaged and air leaks, the module needs to be SMT within 168 hours when the temperature is less than 30 degrees and the relative humidity is less than 60%. If the above conditions are not met, bake.
- Vacuum packaging is not opened, but beyond the shelf life, also need to be baked.

15 Packaging

15.1 Packaging method

Table 15-1：Packaging method

Type	Packaging method	Minimum packaging quantity（PCS）
HJ-568IMH_XX	Roll tape	1500

Use chip-grade anti-static aluminum foil bags to seal and pack with braid. Each bag is put in desiccant. Industrial grade vacuum pump ensures no air leakage, moisture, water and dust (IP65). The actual packaging effect is shown in Figure 15.1.

Figure 9.1:package figure

15.2 Label information

All packages are labeled with cargo information, ROHS label, anti-static label, etc.

【A】Tangshan Hongjia electronic Technology Co., LTD
【B】HJ-XXX-XXX
【C】Pb Free Reflow(260℃)
【D】Date Code:2508　HJ0218
【E】Note: Must be stored in a vacuum Seal
【F】Warning: Humidity sensitivity level MSL:XX
【G】QTY:1500PCS SEAL DATE:20250218

Figure 15.2:Product label drawing

Table 15-2：Module information description

No.	Description
A	company name
B	product model
C	Lead-free reflow mark and reflow temperature setting value
D	Production date Example: 2508 HJ0218 represents the product produced in the 8th week of 2025, on February 18
E	Storage precautions
F	Humidity sensitivity level
G	Quantity of product + date of sale

16 Certification

16.1 FCC Certificate（USA）

TCB

GRANT OF EQUIPMENT AUTHORIZATION

Certification
Issued Under the Authority of the
Federal Communications Commission
By:

TCB

DERYCOM CERTIFICATION SERVICES, INC.
1100 Falcon Avenue
Glencoe, MN 55336

Date of Grant: 05/08/2026

Application Dated: 05/08/2026

Tangshan HongJia electronic technology co., LTD.
352 No. 2 # building power springs in Qianxi
County, Tangshan City, Hebei Provin
Tangshan,
China

Attention: Liu Ying ,

NOT TRANSFERABLE

EQUIPMENT AUTHORIZATION is hereby issued to the named GRANTEE, and is VALID ONLY for the equipment identified hereon for use under the Commission's Rules and Regulations listed below.

FCC IDENTIFIER:	2AGPM-HJ568IMH
Name of Grantee:	Tangshan HongJia electronic technology co., LTD.
Equipment Class:	**Digital Transmission System**
Notes:	**Bluetooth(BLE 6.1) ultra-low power module with integrated antenna**
Modular Type:	**Single Modular**

Grant Notes	FCC Rule Parts	Frequency Range (MHZ)	Output Watts	Frequency Tolerance	Emission Designator
	15C	**2402.0 - 2480.0**	**0.0014**		

Single Modular Approval. Output power listed is conducted power. This device is equipped with five antennas: Option 1- RFANT3216120A5T Series; Option 2-Model: HJ-568IMH; Option 3-Model: HJ2400-FPC2075FSK; Option 4-Model: HJ2400-BA261001TQ; Option 5-Model: HJ2400-XLJ261001WQ. OEM integrators and end-users must be provided with transmitter operation conditions for satisfying RF exposure compliance. Only the antenna tested with the device or similar antennas with equal or lesser gain may be used with this transmitter.

16.2 TELEC Certificate（JAPAN）

DERYCOM CERTIFICATION

Certificate Number: 222-260234
Rev. No. 01
Date of Issue: 08-05-2026

Japan Certificate of Construction Type

Certificate Holder

Tangshan HongJia electronic technology co., LTD.
352 No. 2 # building power springs in Qianxi County, Tangshan City, Hebei Provin,
Tangshan, China

Product Description

Product Name: Bluetooth(BLE 6.1) ultra-low power module with integrated antenna
Model Name: HJ-568IMH, HJ-565IMH, HJ-566IMH, HJ-567IMH, HJ-569IMH
Brand Name: HJSIP®
Hardware Version: V1.1
Software Version: V2.1.1

This Certificate remains valid as long as the stated product stays in compliance with essential requirements of the Japan Radio Act.

COMMON SEAL

Joe Chew

Joe Chew, Product Certifier
DERYCOM CERTIFICATION SERVICES, INC.
1100 Falcon Avenue
Glencoe, MN 55336
USA
Certification Body ID number: 222

DERYCOM CERTIFICATION SERVICES, INC.
1100 Falcon Avenue, Glencoe, MN 55336, USA
Tel: +1 (320) 288-7687, Web: www.derycom-us.com

F/CSD/19 Rev. No. 01 Page 1 / 3

Japan Certificate of Construction Type

16.3 CE Certificate(Europe)

Tel: 86-755-3290 2199
E-mail: sztz@sztz-testing.com
Web: www.sztz-testing.com

Shenzhen Tongzhou Testing Co., Ltd.
Address: 1st Floor, Building 1, Haomai High-tech Park,
Huating Road 387, Dalang Street, Longhua, Shenzhen, China

Verification of Conformity

Registration No. : TZ0032260207E0

Applicant : Tangshan HongJia electronic technology co., LTD.
352 No. 2 # building power springs in Qianxi, County, Tangshan City, Hebei Provin, Tangshan, China

Product : Bluetooth(BLE 6.1) ultra-low power module with integrated antenna

Trademark : HJSIP™

Model(s) : HJ-568IMH, HJ-565IMH, HJ-566IMH, HJ-567IMH, HJ-569IMH

Manufacturer : Tangshan HongJia electronic technology co., LTD.
352 No. 2 # building power springs in Qianxi, County, Tangshan City, Hebei Provin, Tangshan, China

Essential Requirement	Relevant Standards	Test Report No.	Result
Art.3.1(a) Health	EN 62479:2010, EN 50663:2017	TZ0032260207EHE01	Conform
Art.3.1(a) Safety	EN IEC 62368-1:2020 + A11: 2020	TZ0032260207ESA01	Conform
Art.3.1(b) EMC	ETSI EN 301 489-1 V2.2.3 (2019-11), ETSI EN 301 489-17 V3.3.1 (2024-09),	TZ0032260207ERF01	Conform
Art.3.2 Radio	ETSI EN 300 328 V2.2.2 (2019-07)	TZ0032260207ERF03	Conform

This Verification is based on an evaluation of a sample of the above mentioned product. This Verification is issued on a voluntary basis according with the **RED Directive 2014/53/EU**. It confirms that the listed product complies with the principal protection requirements of the directive and is based on the technical specifications applicable at the time of issuance.
This Verification is part of the full test report(s) and should be used in conjunction with it <them>.

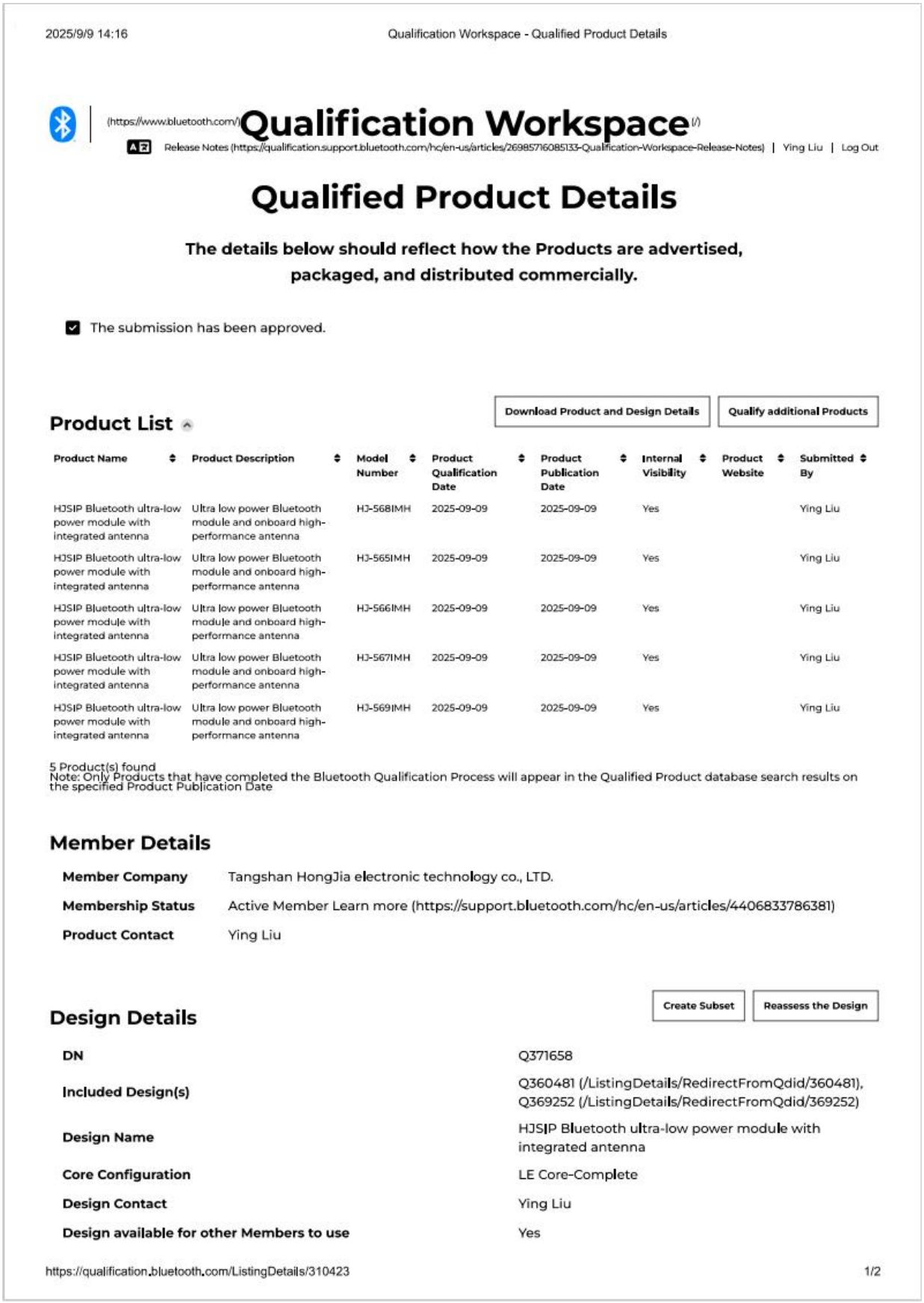
CE

Signed by General Manager (Matte He)
Issue Date: March 30, 2026

This Verification does not replace the regulatory EU Declaration of Conformity (DoC). After preparation of the necessary documentation and establishing compliance to requirements of all applicable directives, the manufacturer may sign a DoC and apply the CE marking.

Page 1 of 1 Version: 1.0

16.4 BQB Certificate (China)



2025/9/9 14:16 Qualification Workspace - Qualified Product Details

(https://www.bluetooth.com/) Qualification Workspace (/)

Release Notes (https://qualification.support.bluetooth.com/hc/en-us/articles/26985716085133-Qualification-Workspace-Release-Notes) | Ying Liu | Log Out

Qualified Product Details

The details below should reflect how the Products are advertised, packaged, and distributed commercially.

☑ The submission has been approved.

Product List

Download Product and Design Details | Qualify additional Products

Product Name	Product Description	Model Number	Product Qualification Date	Product Publication Date	Internal Visibility	Product Website	Submitted By
HJSIP Bluetooth ultra-low power module with integrated antenna	Ultra low power Bluetooth module and onboard high-performance antenna	HJ-568IMH	2025-09-09	2025-09-09	Yes		Ying Liu
HJSIP Bluetooth ultra-low power module with integrated antenna	Ultra low power Bluetooth module and onboard high-performance antenna	HJ-565IMH	2025-09-09	2025-09-09	Yes		Ying Liu
HJSIP Bluetooth ultra-low power module with integrated antenna	Ultra low power Bluetooth module and onboard high-performance antenna	HJ-566IMH	2025-09-09	2025-09-09	Yes		Ying Liu
HJSIP Bluetooth ultra-low power module with integrated antenna	Ultra low power Bluetooth module and onboard high-performance antenna	HJ-567IMH	2025-09-09	2025-09-09	Yes		Ying Liu
HJSIP Bluetooth ultra-low power module with integrated antenna	Ultra low power Bluetooth module and onboard high-performance antenna	HJ-569IMH	2025-09-09	2025-09-09	Yes		Ying Liu

5 Product(s) found
Note: Only Products that have completed the Bluetooth Qualification Process will appear in the Qualified Product database search results on the specified Product Publication Date

Member Details

Member Company	Tangshan HongJia electronic technology co., LTD.
Membership Status	Active Member Learn more (https://support.bluetooth.com/hc/en-us/articles/4406833786381)
Product Contact	Ying Liu

Design Details

Create Subset | Reassess the Design

DN	Q371658
Included Design(s)	Q360481 (/ListingDetails/RedirectFromQdid/360481), Q369252 (/ListingDetails/RedirectFromQdid/369252)
Design Name	HJSIP Bluetooth ultra-low power module with integrated antenna
Core Configuration	LE Core-Complete
Design Contact	Ying Liu
Design available for other Members to use	Yes

https://qualification.bluetooth.com/ListingDetails/310423 1/2

Design available for other users within the same Member company to use	Yes

Supported Layers

View ICS Details | Export ICS

Included Design: Onmicro BLE Host - no CS

View ICS Details

QDID/DN	Q360481 (/ListingDetails/RedirectFromQdid/360481)
Company	Hongkong OnMicro Electronics Limited
Design Name	Onmicro BLE Host - no CS

Included Design: Onmicro BLE Controller – no CS

View ICS Details

QDID/DN	Q369252 (/ListingDetails/RedirectFromQdid/369252)
Company	Hongkong OnMicro Electronics Limited
Design Name	Onmicro BLE Controller – no CS

Appendix 1：

Output Format Description for ST_DISCOVERY_DEVICE_MAC / ST_DISCOVERY_DEVICE_NAME :

Applicable to: Customers, FAE, Test Engineers. This document describes the output format that the customer-side UART will receive when the module acts as a BLE master to perform device discovery on a target device.

1. Functional Overview

ST_DISCOVERY_DEVICE_MAC and ST_DISCOVERY_DEVICE_NAME are used to find a target BLE device by specified MAC or specified advertising name. After the module scans and finds the target device, it will temporarily connect to that device, discover the target device's GATT Services, Characteristics, and Descriptors, and output the discovery results through UART. After discovery is completed, the module will automatically disconnect this temporary central connection.

Note: These two commands are NOT ordinary "scan list" commands. They do not output a list of all surrounding devices. Instead, they target a specific device by MAC or name and perform connection and GATT discovery only on that device.

2. Command Format

2.1 Find Target Device by MAC

```
<ST_DISCOVERY_DEVICE_MAC=AABBCCDDEEFF>
```

AABBCCDDEEFF is a 12-character hexadecimal MAC string without colons, supporting 0-9, A-F, a-f.
For example, if the target MAC is A1:B2:C3:D4:E5:F6, send:

```
<ST_DISCOVERY_DEVICE_MAC=A1B2C3D4E5F6>
```

2.2 Find Target Device by Name

```
<ST_DISCOVERY_DEVICE_NAME=DeviceName>
```

DeviceName is the advertising name of the target device. The current logic performs exact matching of name length and content, with a maximum of 31 bytes.
For example, if the target advertising name is HJ-TEST, send:

```
<ST_DISCOVERY_DEVICE_NAME=HJ-TEST>
```

3. Execution Flow

```
Send discovery command
  -> Module returns ok / busy / invalid
  -> Module scans for target device
  -> Automatically connects after finding the target
  -> After successful connection,discovers target GATT services,characteristics,and descriptors
  -> Outputs discovery results line by line
  -> Outputs <disc_done>
  -> Module automatically disconnects the temporary connection
```

Each UART output record is enclosed in angle brackets, e.g., <disc_service:...>. During actual UART transmission, each record is followed by CRLF, i.e., \r\n.

4. Immediate Command Response

Command format is correct and discovery has started：

```
<st_discovery_device_mac=ok>
<st_discovery_device_name=ok>
```

There is already an existing central connection, currently connecting, or discovery is already in progress, so a new discovery cannot be started:

<st_discovery_device=busy>

Command format is incorrect, parameters are invalid, or underlying scanning failed to start:

<st_discovery_device=invalid>

5. Timeout and Failure Output

When searching by MAC, if the target device is not found before scan timeout:

<st_discovery_device_mac=timeout>

When searching by name, if the target device is not found before scan timeout:

<st_discovery_device_name=timeout>

Target found but connection establishment failed or connection timed out:

<disc_error:connect_fail>

An error occurred during GATT discovery:

<disc_error:reason=error_reason,err=0xcode>

Examples：

<disc_error:reason=gatt_timeout,err=0x00000002>
<disc_error:reason=watchdog_timeout,err=0x00000003>
<disc_error:reason=find_char_rsp,err=0x0000000a>

If the target device actively disconnects during the discovery process:

<disc_error:disconnected>

6. Service Output Format

When a 16-bit UUID service is discovered:

<disc_service:uuid16=0x180f,name=battery>

Field	**Description:**
uuid16=0x180f	Service UUID
name=battery	Standard UUID name.Unknown service names output null

When a 128-bit UUID service is discovered:

<disc_service:uuid128=f0debc9a78563412f0debc9a78563412,name=null>

Note: Current service output does not include service start handle / end handle. Customers should not look for write handles from service lines. The handles actually used for read/write or notification determination are in the disc_char lines.

7. Characteristic Output Format

16-bit UUID characteristic output format:

<disc_char:handle=0x0012,uuid16=0x2a19,name=battery_level,props=ntf|read,desc=0x0013.0x2902.client_characteristic_configuration>

128-bit UUID characteristic output format:

<disc_char:handle=0x0025,uuid128=f1debc9a78563412f0debc9a78563412,name=null,props=write_req|write_cmd,desc=null>

Field	**Description:**
handle	Characteristic value handle. Typically used for GATT read/write.
uuid16 / uuid128	Characteristic UUID.
name	Standard UUID name. Unknown names output null.
props	Characteristic properties, e.g., read, write_req,

	write_cmd, ntf, ind.
desc	The list of descriptors following this characteristic. Outputs null if no descriptors exist.

8. props Field Description:

Field	Meaning
read	This characteristic supports reading.
write_req	This characteristic supports Write Request, with response after writing.
write_cmd	This characteristic supports Write Command, without response after writing.
ntf	This characteristic supports Notification.
ind	This characteristic supports Indication.
null	No available attribute detected or attribute is empty.

If a characteristic supports multiple attributes simultaneously, attributes are separated by "|", for example:

```
props=ntf|read
props=write_req|write_cmd
```

9. desc Field Description:

The desc field indicates the list of descriptors associated with the current characteristic. The format for a single descriptor is:

```
0x descriptor_handle.0x descriptor_uuid.descriptor_name
```

For example, the CCCD descriptor:

```
0x0013.0x2902.client_characteristic_configuration
```

Multiple descriptors are separated by "|":

```
desc=0x0013.0x2902.client_characteristic_configuration|0x0014.0x2901.characteristic_user_description
```

If there are no descriptors:

```
desc=null
```

If there are too many descriptors or a single line is too long, it will be appended with:

```
|truncated
```

10. Completion Output

After GATT discovery is completed normally, the following is output:

```
<disc_done>
```

Receiving <disc_done> indicates that the GATT service, characteristic, and descriptor output for the target device has ended. The module will then automatically disconnect this temporary central connection, and usually no additional ordinary disconnection status will be output.

11. Complete Example：discovery by MAC

Input：

```
<ST_DISCOVERY_DEVICE_MAC=A1B2C3D4E5F6>
```

Possible output：

```
<st_discovery_device_mac=ok>
<disc_service:uuid16=0x1800,name=generic_access>
<disc_char:handle=0x0003,uuid16=0x2a00,name=device_name,props=read,desc=null>
<disc_char:handle=0x0005,uuid16=0x2a01,name=appearance,props=read,desc=null>
```