FSC-BT836x User Guide

Release 2.2.1

Table of contents

[中文版]

This guide applies to the **FSC-BT836x** Series Bluetooth dual-mode data transmission application modules. Specific module models include:

- FSC-BT836
- FSC-BT836B
- FSC-BT836N
- FSC-BT836GN
- FSC-BT836BG

This guide details the hardware specifications, functional features, data communication principles, rapid development kits, quick start,development examples, firmware upgrade methods, and FAQs for the **FSC-BT836x** Series modules. It consists of the following chapters:

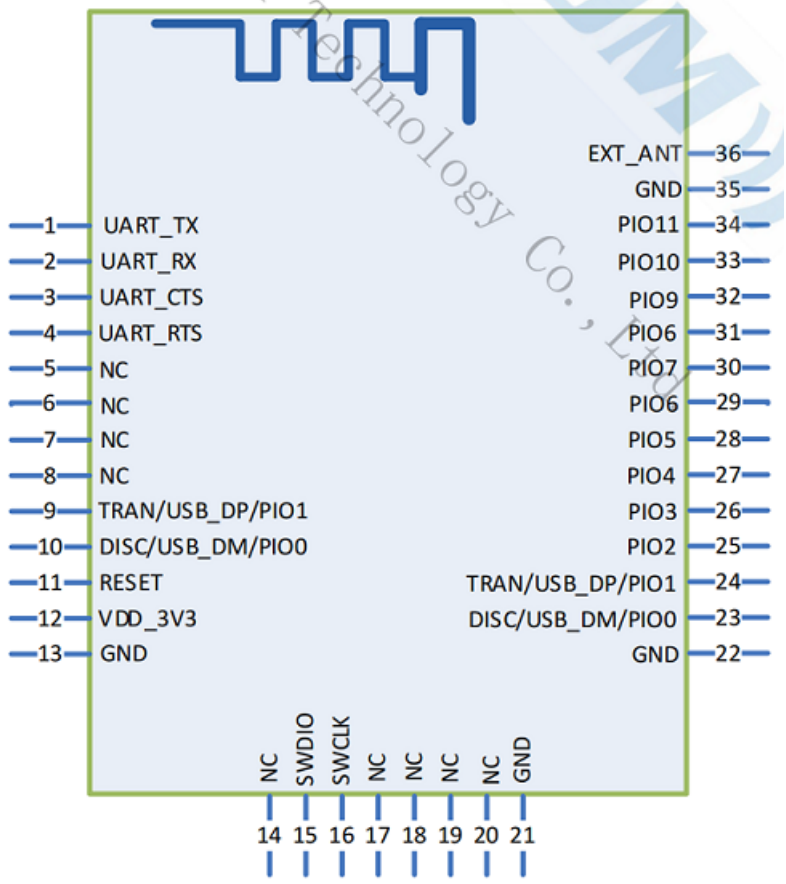
Chapter 1

Hardware Design

[中文版]

1.1 1. PIN Definition Descriptions

Pin Diagram:



Pin Description:

Pin	Pin Name	Type	Description
1	UART_TX	O	Serial Transmitter Data
2	UART_RX	I	Serial Receiver Data
3	UART_CTS	I/O	UART Clear to Send(no connection request)
4	UART_RTS	I/O	UART Request to Send(no connection request)
24	Tran/USB_DP	I/O	UART mode control pin, H=command mode ,L=throughput mode
23	Disc/USB_DM	I/O	Disconnect the connecting pin
11	RESET	I	Active-low reset input
12	VDD	Powe	Power supply voltage 3.3V, LDO power supply preferred
13	GND	GND	GND
15	ICE	I/O	Writing firmware pin
32	LED	O	Bluetooth not connected to output square wave, Bluetooth connected to output high level
33	STATUS	O	Connection state, output. H=Connected , L=No connection
36	EXT_ANT	ANT	Changing the 0 ohm resistance near the antenna, it is possible to connect a Bluetooth antenna externally

Note:

To use the 23 and 24 pin control modules AT+PIOCFG command is required to enable the function

1.2 2 Hardware Design Note

- The module only needs to be connected to VDD/GND/UART_RX/UART_TX ready to use.
- If the MCU needs to obtain the connection status of the Bluetooth module, it needs to be connected to the STATUS pin.
- After drawing the schematic diagram, please send it to Feasycom for review to avoid Bluetooth distance not achieving the best effect.

Chapter 2

Function Description

[中文版]

2.1 1. Default Configuration

Generic Dual-Mode Data Transmission Application Firmware default configuration, using **FSC-BT836B** as an example：

Name	FSC-BT836B
LE-Name	FSC-BT836B-LE
Pin Code	0000
Secure Simple Pairing Mode	On
UART Baudrate	115200/8/N/1

2.2 2. GPIO Indicators

2.2.1 Module Operation Status

Pin	Status	Description
Pin 32	1Hz square wave	Bluetooth Disconnected
Pin 32	High level	Bluetooth Connected

2.2.2 BT Connection Status

Pin	Status	Description
Pin 33	Low Level	Bluetooth Disconnected
Pin 33	High Level	Bluetooth Connected

2.2.3 Data Transmission Mode

Pin	Status	Description
Pin 24	Low Level	Throughput Mode
Pin 24	High Level	Command Mode

2.3 3. Operation Modes

2.3.1 Throughput Mode

- **Bluetooth Not Connected**: Data received via UART is parsed as AT commands.
- **Bluetooth Connected**: All data received via UART is sent as-is to the remote Bluetooth device.

2.3.2 Command Mode

- **Bluetooth Not Connected**: Data received via UART is parsed as AT commands.
- **Bluetooth Connected**: Data received via UART is still parsed as AT commands. Data must be sent to the remote device using AT commands, e.g., `AT+SPPSEND`.

2.4 4. GATT Service

Type	UUID	**Operation**	Description
Service	0xFFF0		Throughput transmission service
Write	0xFFF2	Write,Write Without Response	APP to Module
Notify	0xFFF1	Notify	Module to APP

Chapter 3

Data Communication Principles

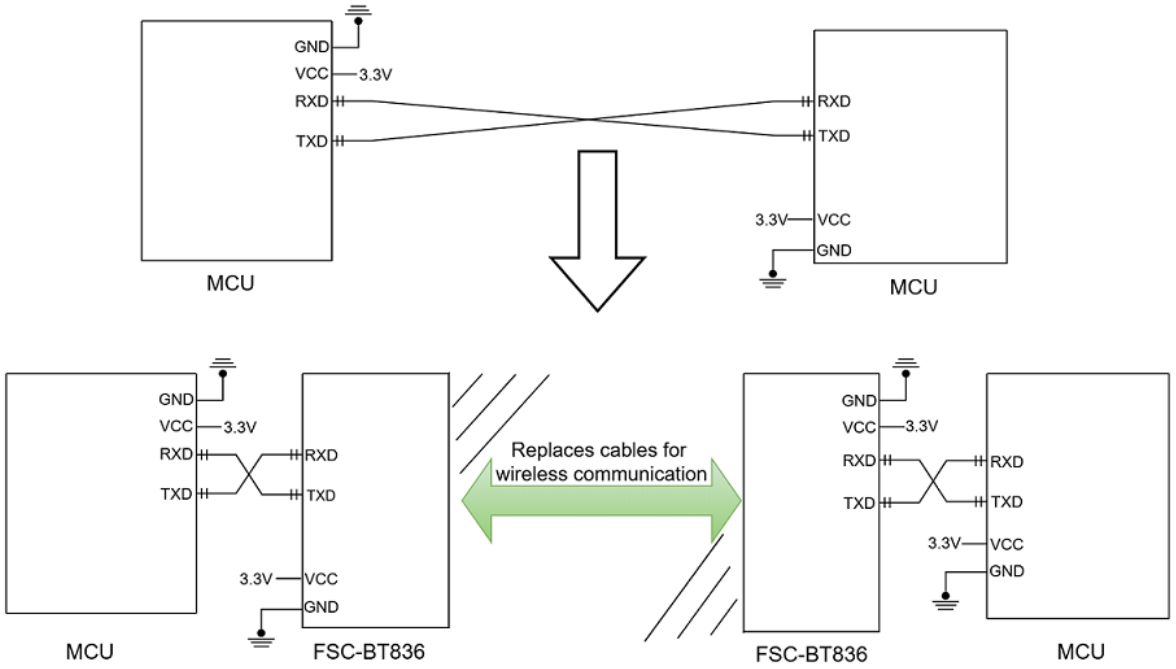
[中文版]

3.1 1. Communication Principle

The FSC-BT836x series Bluetooth dual-mode data modules enable wireless communication between devices based on the SPP (Serial Port Profile) and BLE (Bluetooth Low Energy) dual-mode protocols.

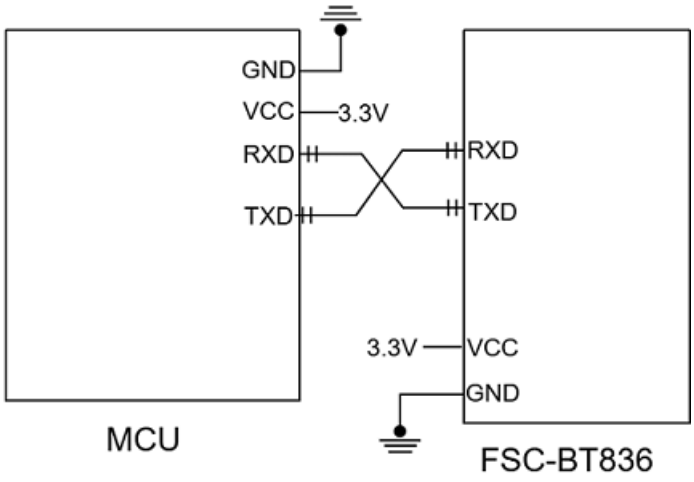
- **SPP Mode**: Emulates traditional serial port communication, establishing a virtual serial link through the RF layer. It supports continuous large data transfers (e.g., file transfer) and is suitable for scenarios like printers.
- **BLE Mode**: Utilizes an event-driven, low-power architecture, defining a “Service-Characteristic” model via the GATT protocol for intermittent small data interactions (e.g., sensor data), ideal for IoT devices.

Both modes share the underlying RF hardware and switch automatically via the protocol stack. The module communicates with the host device (phone/MCU) through the UART interface using AT commands or transparent data transmission to establish connections, exchange data, and manage status.



As shown in the figure, the Bluetooth module replaces the physical cable in full-duplex communication. A device like an MCU (left) sends data via its TXD pin to the Bluetooth module (left). The module's RXD port receives the UART data and automatically transmits it via radio waves over the air to the remote Bluetooth module. The remote Bluetooth module (right) receives the airborne data and delivers it via its TXD pin to the local device, like an MCU (right).

3.2 2. MCU-to-Module Communication

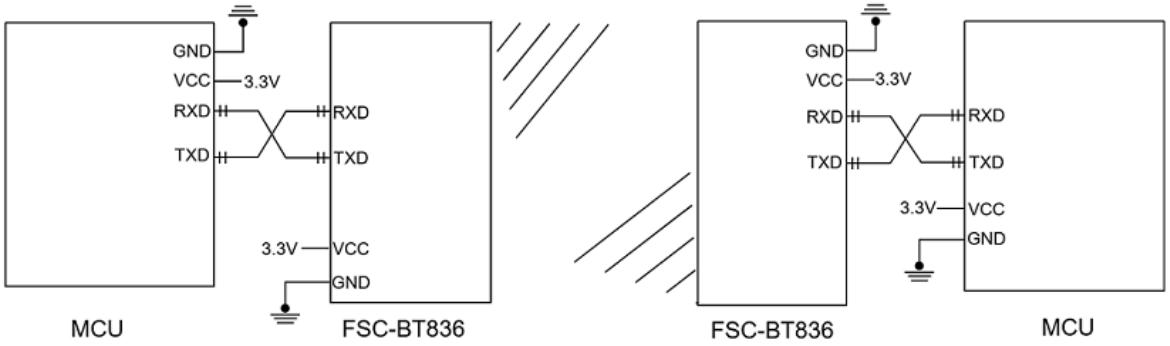


This diagram illustrates the connection between a master MCU (Microcontroller Unit) and an FSC-BT836 Bluetooth module via cross-connected serial ports, enabling command interaction between the master and the Bluetooth module to support wireless communication functions. It is applicable to IoT devices, remote control, and other scenarios.

1. **Serial Communication Interface**: The master MCU’s transmit pin (MCU_TX) is cross-connected to the Bluetooth module’s receive pin (UART_RX), and the master MCU’s receive pin (MCU_RX) is similarly connected to the Bluetooth module’s transmit pin (UART_TX), forming a bidirectional data transmission channel.
2. **Power and Grounding**: The Bluetooth module is powered by 3.3V through the VDD_3V3 pin and shares a common ground (GND) with the master MCU, ensuring level compatibility and signal stability.

3.3 3. Module-to-Module Communication

Two FSC-BT836 Bluetooth modules can establish a Bluetooth connection automatically upon power-up.



A module can act as a master device to connect to a slave device. The host can send commands to control the module for Bluetooth scanning, connection establishment, data transmission, and connection termination.

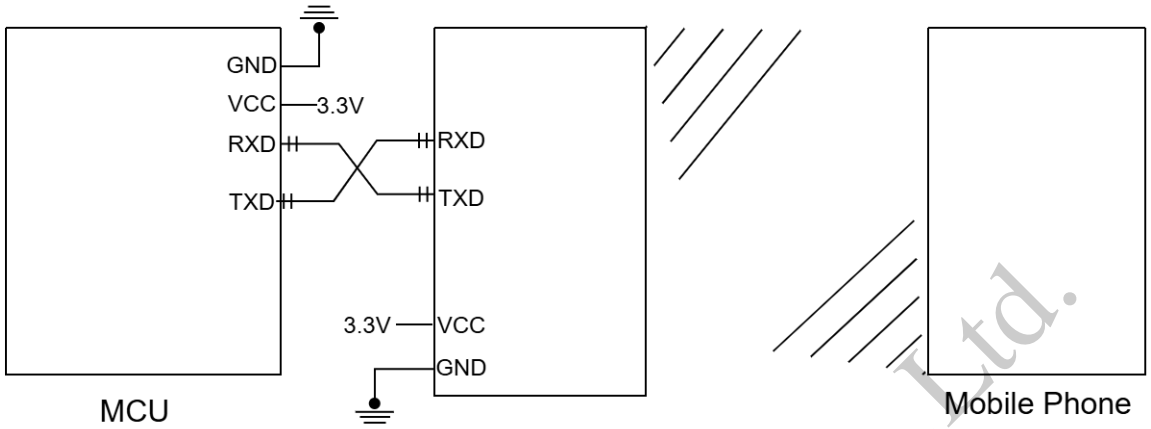
3.4 4. Module-to-Phone Communication

3.4.1 4.1 Why do we need to use an app on a mobile phone for Bluetooth connection and communication?

Native phone Bluetooth functionality primarily supports common use cases like audio transfer and file sharing. Some Bluetooth peripheral devices can be connected via the phone's built-in settings (e.g., Bluetooth speakers, headphones, keyboards, mice). However, when a peripheral device, like a module only supporting SPP/GATT protocols, cannot be connected via native phone settings, a specific mobile application, such as the FeasyBlue app, is generally required for con-

nection.

3.4.2 4.2 Communication Application Diagram



Bluetooth Module Side (FSC-BT836): Continuously transmits broadcast data after power-up.

Mobile Terminal : Can discover the broadcast packets via scanning and initiate a connection request to the module (FSC-BT836). Upon successful connection, the Bluetooth module (FSC-BT836) will pull the connection status pin HIGH and respond to status indication commands (valid in Command Mode) to notify the host of the successful Bluetooth connection.

Host: Can send data to the remote (Mobile Terminal) Bluetooth via the UART through the Bluetooth module. Conversely, the remote (Mobile Terminal) Bluetooth can also send data back to the host.

Chapter 4

Quick Development Kit

[中文版]

4.1 1. Datasheet

- FSC-BT836B Datasheet

4.2 2. Evaluation Board

- FSC-DB005：Feasycom’s USB to Serial Bluetooth Data Transmission Application Development Board.
- (Optional) FSC-DB004：Feasycom’s Pin-header Bluetooth Serial Data Transmission Application Development Board.

4.3 3. AT Command Set

- FSC-BT836x General Dual-Mode Data AT Command Set

4.4 4. Serial Port Tool

- Feasycom Serial Port Assistant：A serial port assistant based on Windows PC.

4.5 5. App & SDK

- FeasyBlue： Feasycom App & SDK resource supporting Android and iOS platforms, which enables functions such as Bluetooth BLE & SPP data communication debugging, Feasycom module firmware version reading, and parameter configuration.

4.6 6. Firmware Upgrade

- OTA Upgrade
  - Tool : FeasyBlue APP (Based on Mobile Terminal Android & iOS Application)
  - User Guide: Please refer to FSC-BT836x - OTA Upgrade .
- UART Upgrade
  - Tool : Feasycom-UART-Upgrde-tool-4.6.zip (Based on Windows PC)
  - User Guide: Please refer to FSC-BT836x - UART Upgrade .

Chapter 5

Quick Start

[中文版]

5.1 1. Hardware Preparation

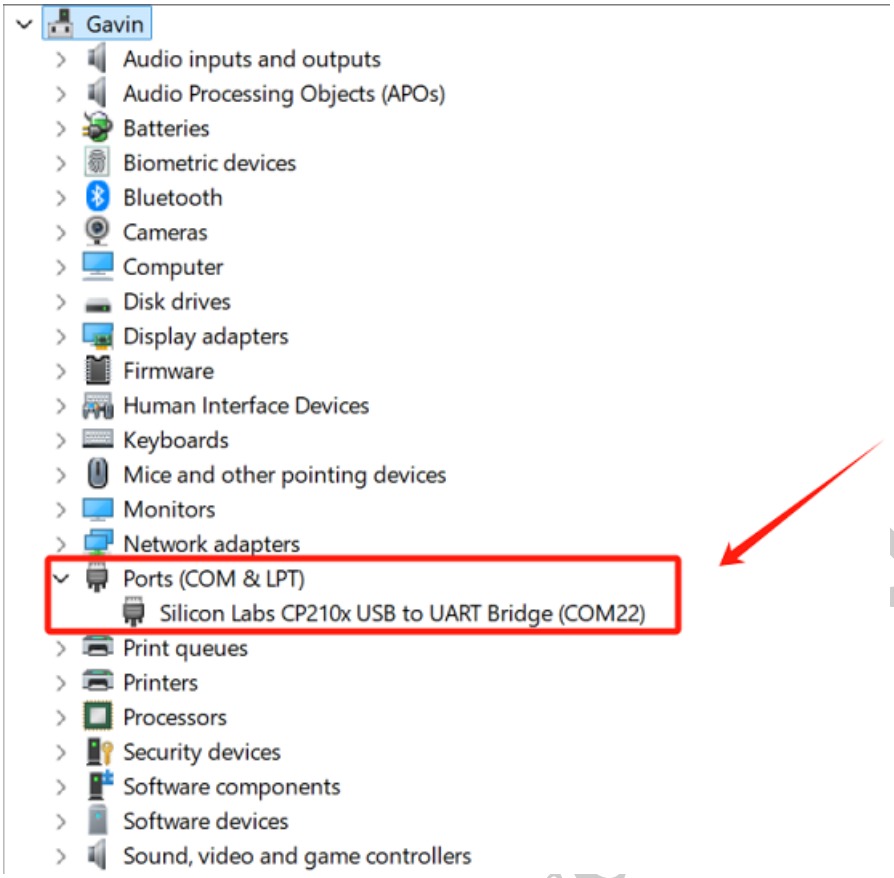
- 1 x FSC-DB005-BT836B Rapid Development Kit : FSC-DB005 USB-to-Serial Rapid Evaluation Board pre-integrated with FSC-BT836B module (Take FSC-BT836B an example)
- 1 x Computer : Windows / Mac
- 1 x Mobile Phone : Android / iOS

5.2 2. Software Preparation

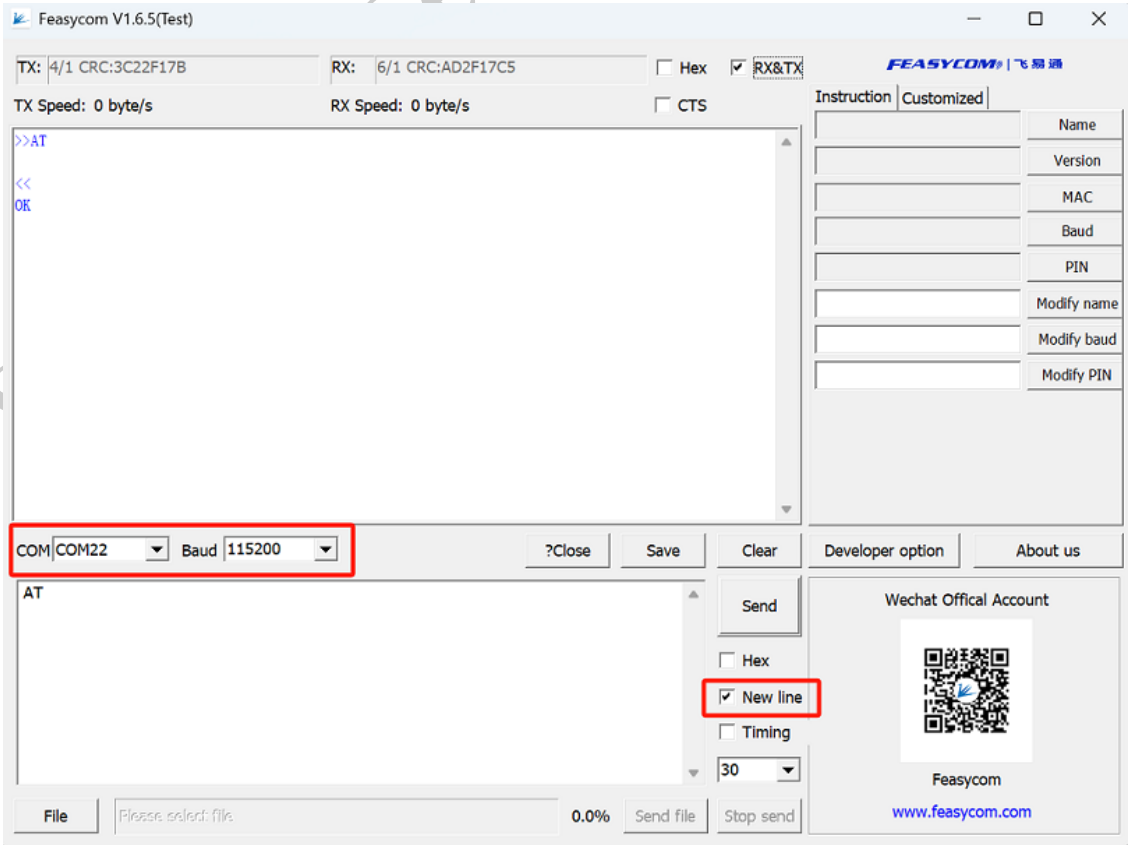
- Feasycom Serial Port Assistant
- FeasyBlue APP
- **Communication Interface**: UART
- **UART Configuration**: 115200/8/N/1

5.3 3. Hardware Connection Method

1. Connect the FSC-DB005-BT836B to the PC via USB. The PC will automatically recognize the serial port and create a virtual COM port.



2. Run the Feasycom Serial Port Assistant on the PC. Set the correct **COM Port** and **Baud Rate**, and check the **Send New Line** option.



5.4 4 Communication Test

The following lists a few basic universal AT command test examples. For more commands, please refer to the accompanying FSC-BT836x General Dual-Mode Data AT Command Set .

5.4.1 AT - UART Communication Test

Command	AT\r\n
Response	**\r\nOK\r\n**
Description	Test the UART communication between HOST and Module after power on, baudrate changed, etc.

Example:

```
send:     >>AT\r\n
response: <<\r\nOK\r\n   //Successfully connected.
```

5.4.2 AT+NAME - Read/Write BR/EDR Name

Example: Read BR/EDR Bluetooth Name

```
send:     <<AT+NAME\r\n
response: >>\r\n+NAME=FSC-BT836B-XXXX\r\n   //Example, please refer␣
↪to the actual reading result
response: >>\r\nOK\r\n
```

5.4.3 AT+VER - Read Current Firmware Version

Example: Read Current Firmware Version

```
send:     <<AT+VER\r\n
response: >>\r\n+VER=1.0.0,FSC-BT836B\r\n     //Example, please␣
↪refer to the actual reading result
response: >>\r\nOK\r\n
```

Chapter 6

Development Examples

[中文版]

6.1 Data Throughput Mode Application

6.1.1 What is Throughput Mode?

The FSC-BT836B Bluetooth dual-mode data transmission module operates with two data transmission modes: **Throughput Mode** and **Command Mode**.

The generic data throughput firmware for the FSC-BT836x series modules defaults to throughput mode. To switch modes, refer to the FSC-BT836B AT Command Set and use the **AT+TPMODE** command. The operation and differences between the two modes are as follows:

- **Throughput Mode**:

  Bluetooth Not Connected: Data received via UART is parsed as AT commands.

  Bluetooth Connected: All data received via UART is sent as-is to the remote Bluetooth device. It does not contain any data headers or framing and does not require AT commands to send data.

- **Command Mode**:

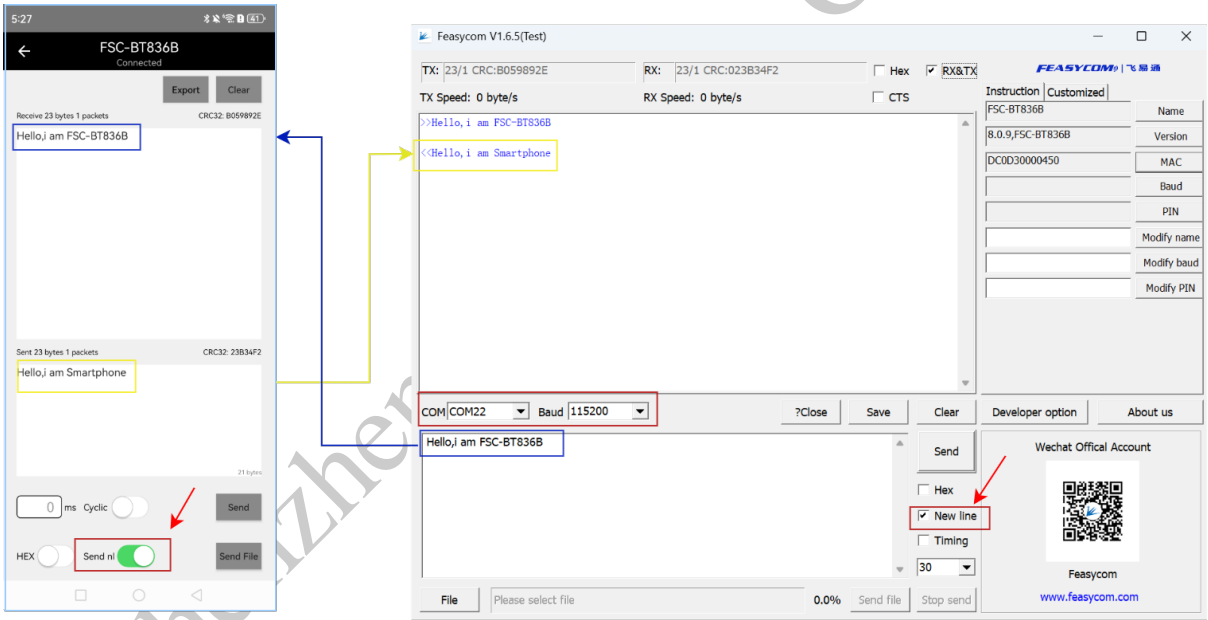
  Bluetooth Not Connected: Data received via UART is parsed as AT commands.

  Bluetooth Connected: Data received via UART is still parsed as AT com-

mands. It will contain specific response indication headers and framing. Data must be sent to the remote device using AT commands, such as *AT+LESEND*.

6.1.2 Between Module and Phone Application

1. **Module**: Upon power-up, the module continuously transmits advertisement packets.
2. **Phone**: Open the FeasyBlue APP, scan for nearby Bluetooth device advertisements, find the target Bluetooth module, and establish a connection.
3. **Connection Success**: The status pin on the module side will be pulled HIGH, indicating a successful connection.
4. **Data Transmission**: After a successful connection in Transparent Transmission Mode, data received by the module via UART is automatically transmitted over the air to the remote end (phone side).



6.1.3 Between Modules Application

A demonstration of SPP communication and data transparent transmission between an FSC-BT836B and an FSC-BT986 Bluetooth module is as follows:

1. Scan for nearby SPP devices

   The FSC-BT836B scans for nearby Bluetooth SPP devices. Operation is as follows:

```
Send: <<AT+SCAN=1                    // Scan for nearby␣
↪Bluetooth SPP devices
Response: >>OK
          >>+SCAN={                     // Scan started
          >>+SCAN=0,0,001D439A2001,-45,10,FSC-BT986
          >>+SCAN=1,1,DC0D3023CA1C,-71,14,0000-1007Z ggg
          >>+SCAN=2,0,DC0D30000015,-88,23,FSC-BT1038C-LE-
↪AKM-0015
          >>+SCAN=3,0,DC0D300017A8,-85,11,FSC-BT3721V
          >>+SCAN=5,0,A4405B28A838,-89,10,AirGo AS01
          >>+SCAN=}                    // Scan ended
```

2. Establish SPP connection request

   The FSC-BT836B establishes an SPP connection with the FSC-BT986 using the AT+SPPCONN command. Operation is as follows:

```
Send: <<AT+SPPCONN=001D439A2001       // Initiate SPP␣
↪connection to remote FSC-BT986
Response: >>OK
```

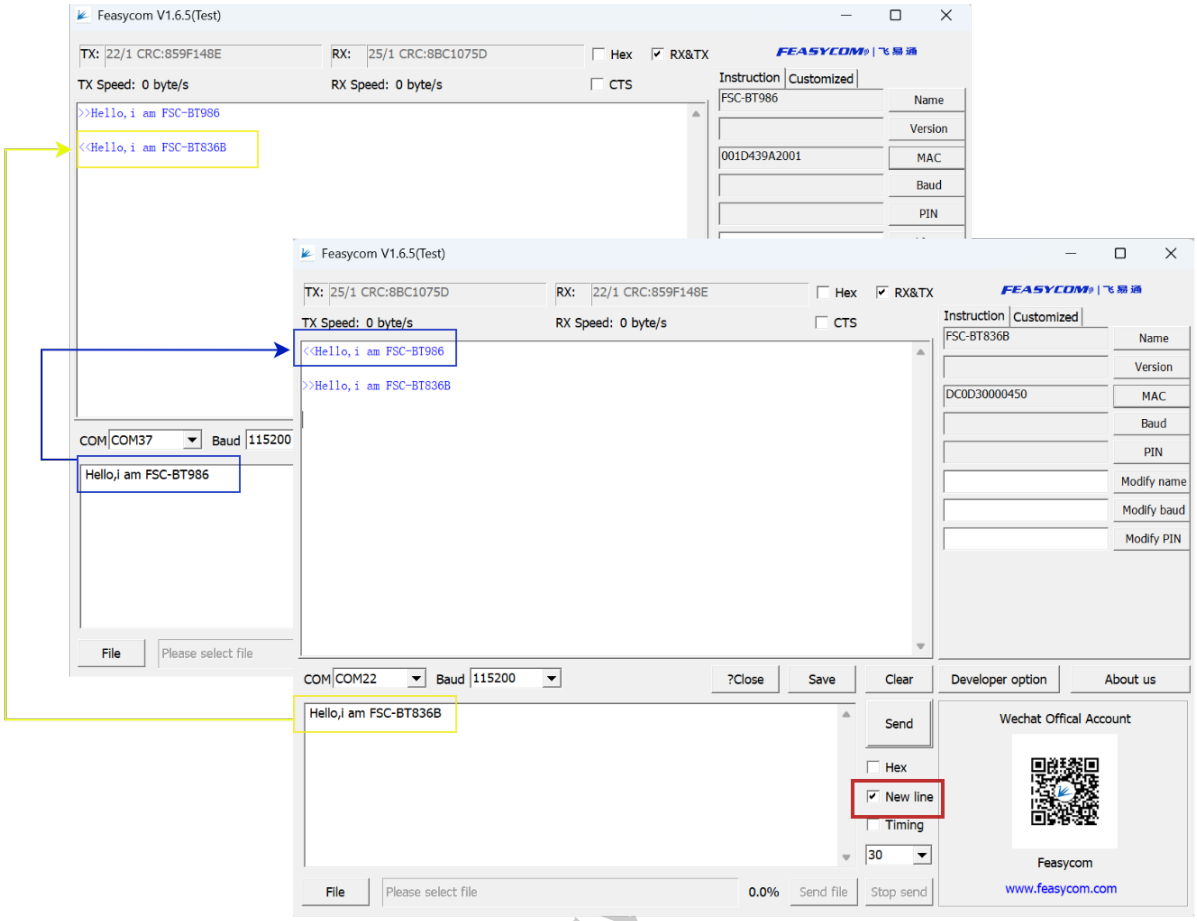
3. SPP connection established successfully

   In Transparent Transmission Mode, after a successful Bluetooth connection, the UART will not receive event response indicators. The connection status can be determined by the level state of the status pin (e.g., Pin 10) on the FSC-BT836B, as follows:

   - **High Level (H):** Indicates Bluetooth is successfully connected.
   - **Low Level (L):** Indicates Bluetooth is not connected or the connection has been disconnected.

4. Send data

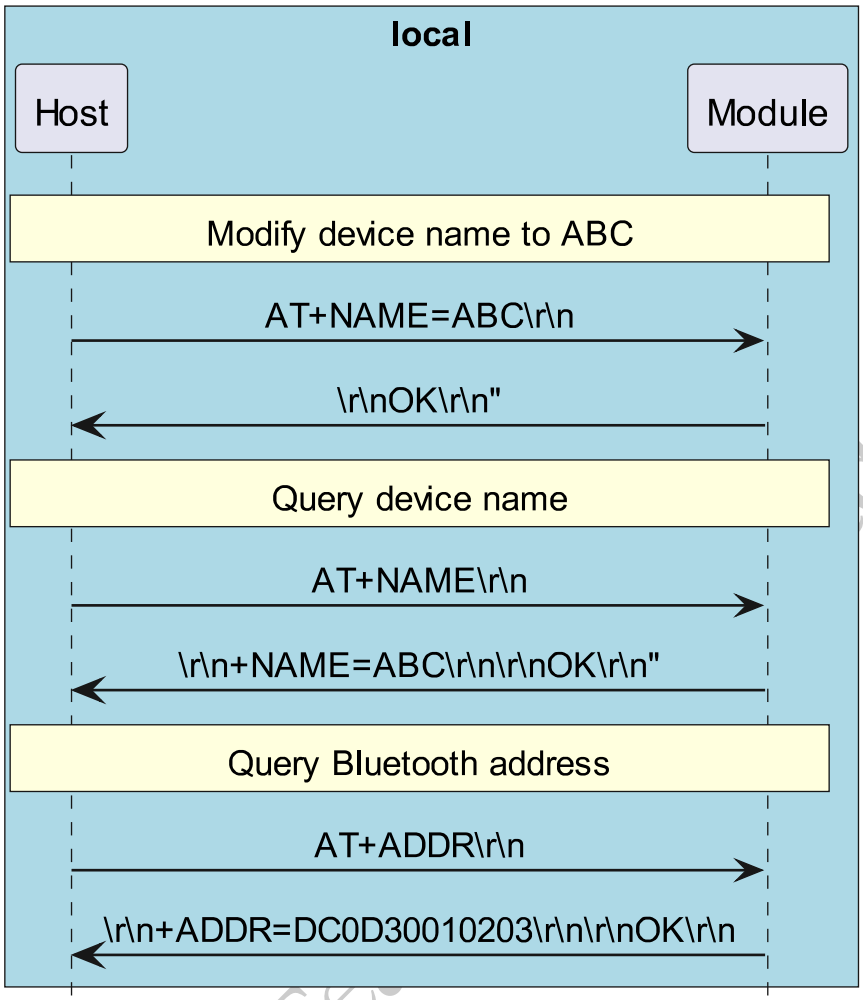
   Transparent transmission mode is enabled by default in the generic data transmission firmware. After a successful SPP connection, data can be sent directly without using AT commands, as shown in the figure below:



6.2 Read/Write Module Default Parameters

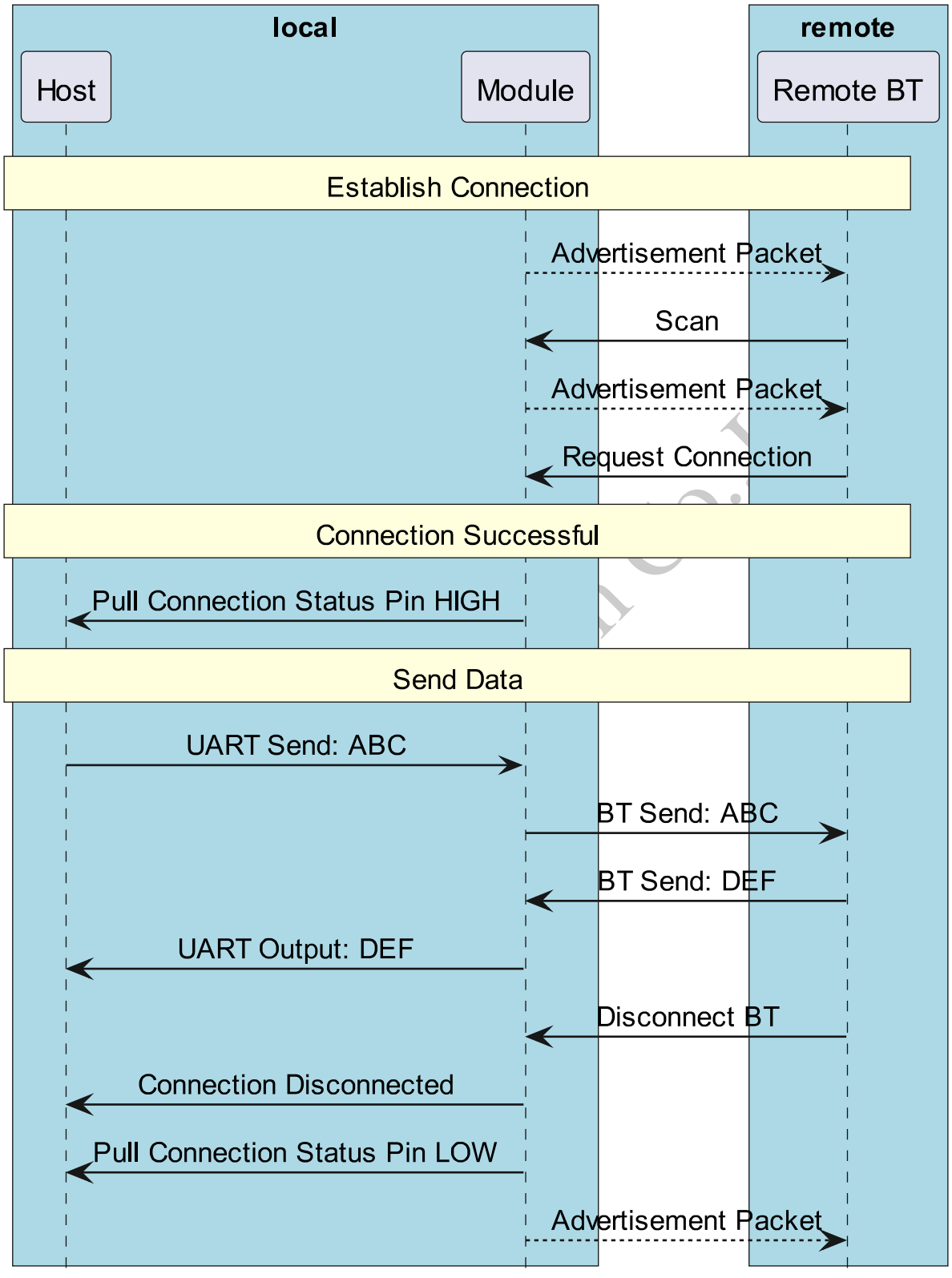
When Bluetooth is not connected, the module parses UART data as AT commands. The host can query and modify the module’s default parameters. The following example demonstrates:

1. Write the device name to ABC
2. Read the device name
3. Read the Bluetooth address



6.3 Data Transmission Flow

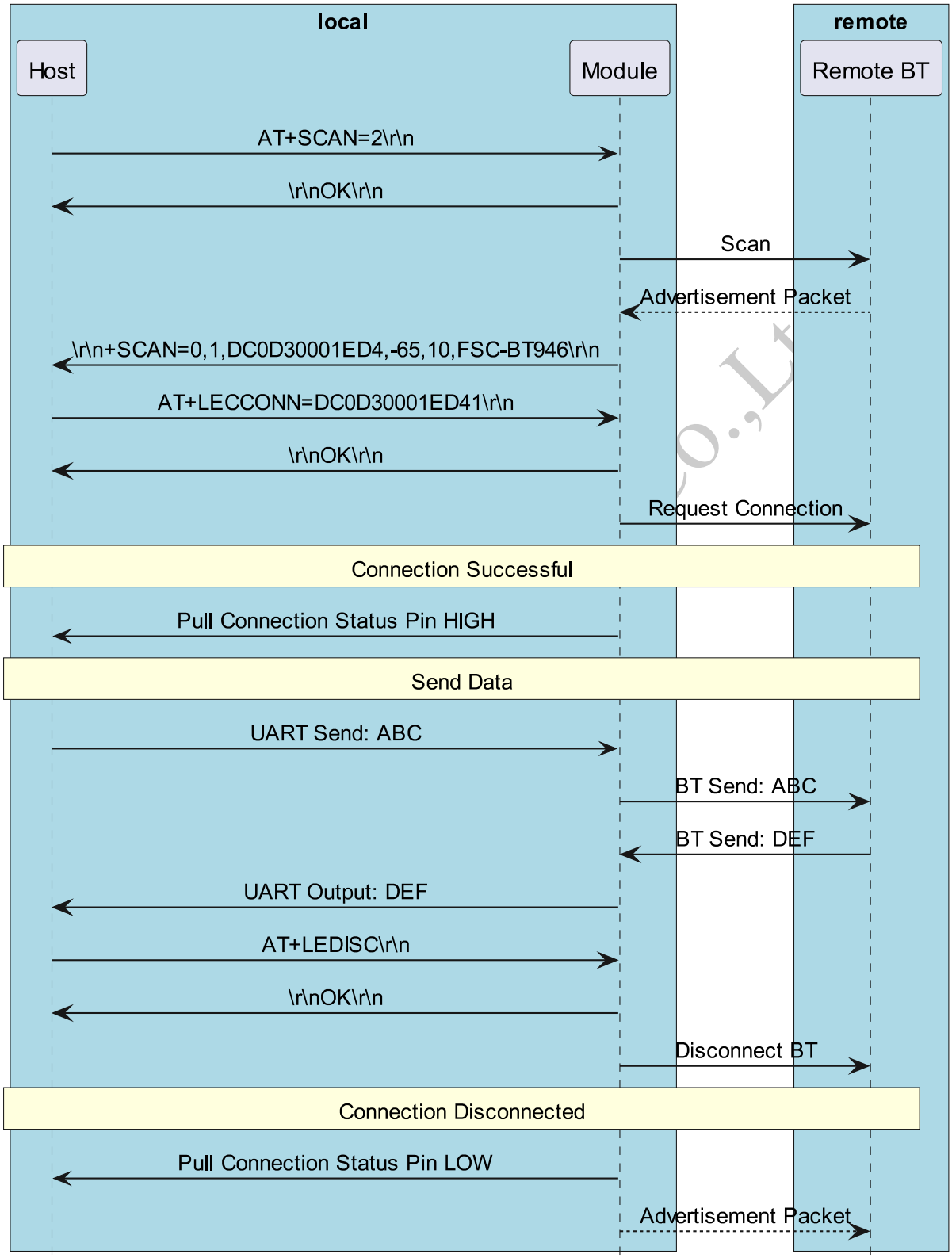
Upon power-up, the module continuously transmits advertisement data. A remote Bluetooth device (e.g., phone) can discover these advertisement packets via scanning and initiate a connection request to the module. Upon successful connection, the module pulls its connection status pin HIGH to notify the host of the successful Bluetooth connection. The host can send data to the remote Bluetooth device via the Bluetooth module, and the remote Bluetooth device can also send data to the host.



6.4 Module Acting as Master to Connect to Remote Device

The module can act as a master device to connect to a slave device. The host can send commands to control the module to perform scanning, connection, and disconnection. The follow-

ing diagram illustrates the process of connecting to another device:



Chapter 7

Firmware Upgrade

[[中 文 版]]([固 件 升 级—FSC-BTDMD 2.3 documentatihttps://document.feasycom.com/docs/fsc-btdmd/CN/latest/bt836x/8-firmware-upgrade-cn.html)

7.1 1. OTA Upgrade

7.1.1 1.1 OTA Upgrade Tool

- FeasyBlue APP (Based on Mobile Terminal Android & iOS Application)

7.1.2 1.2 OTA Upgrade guide

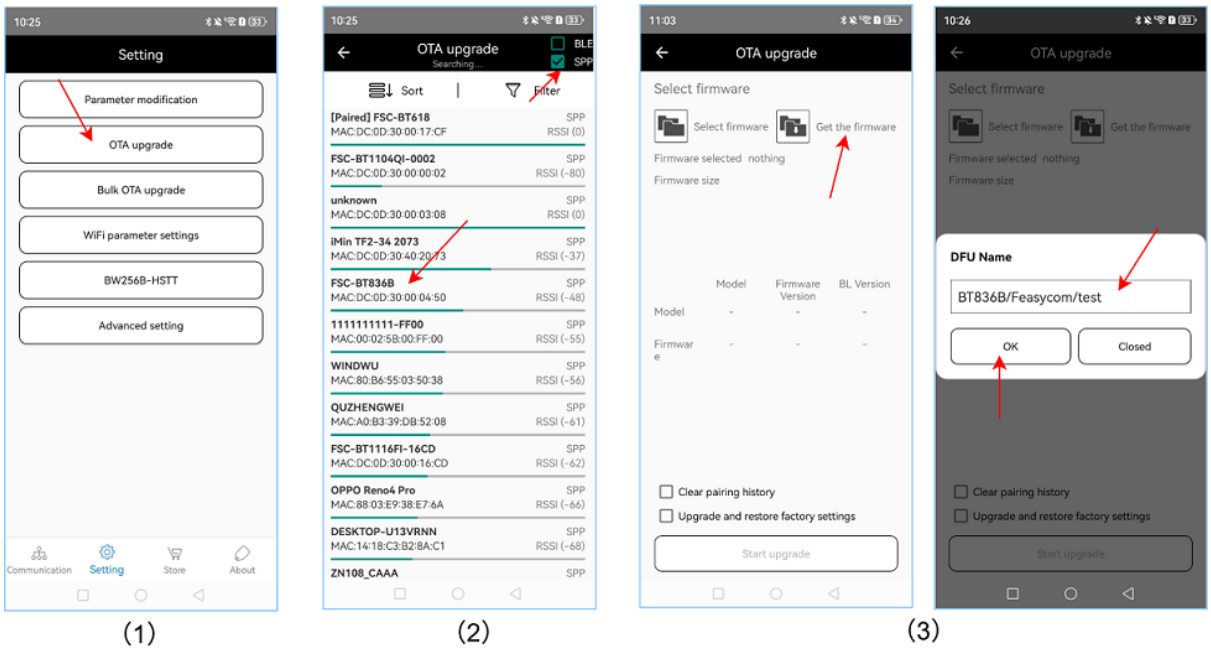
1. Run the FeasyBlue APP, select **Settings - OTA Upgrade** to enter the OTA Upgrade function section. After entering, you will navigate to the device search interface to select the Bluetooth device to be upgraded.
2. Search for and select the Bluetooth device that needs upgrading. After selecting the device, you will enter the **Load Firmware** function interface to load the firmware upgrade file.
3. There are two ways to load the firmware upgrade file:
    - **Select Firmware**: Choose to load the firmware upgrade file stored in the mobile phone’s local storage (the firmware upgrade file is provided by Feasycom).
    - **Get Firmware**: Enter the DFU name to download and import the corresponding firmware upgrade file from the cloud server via the network (the DFU name is provided by Feasycom).

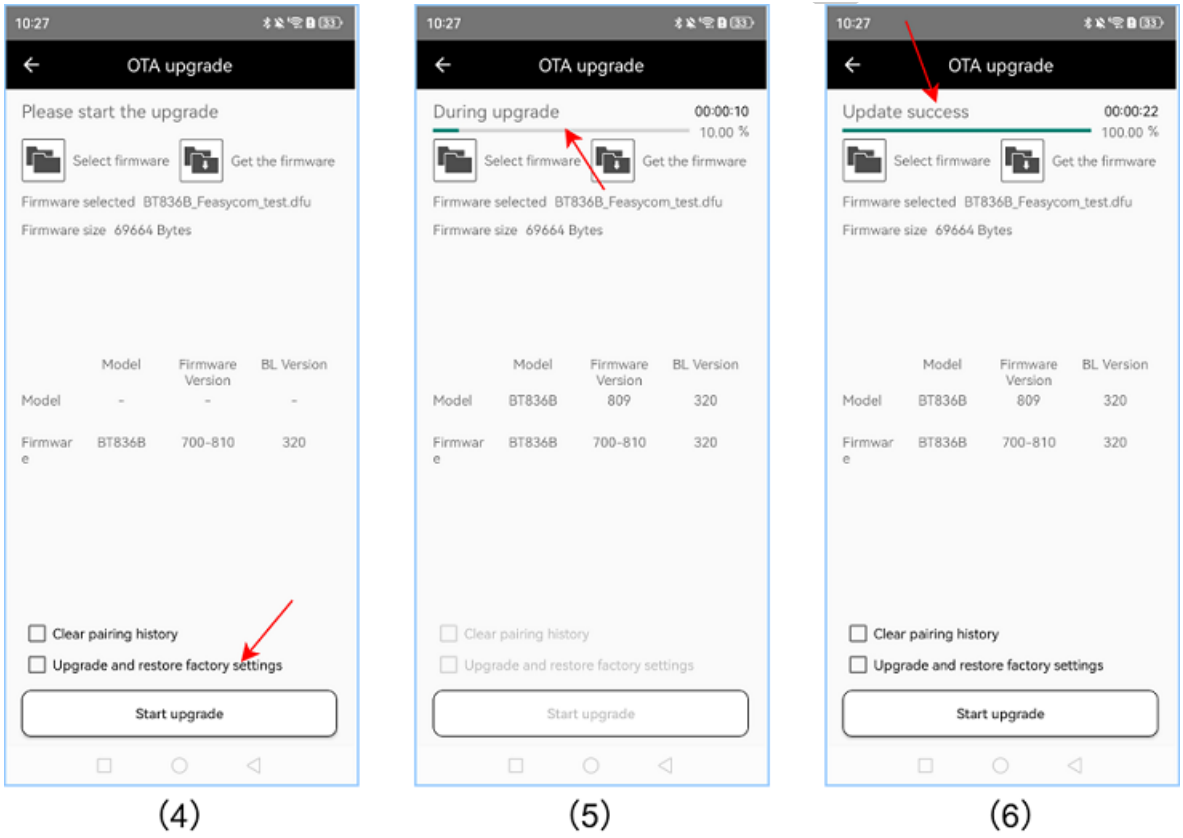
4. After the firmware file is loaded successfully, click the **Start Upgrade** button. The interface will display “Upgrading” and the upgrade progress, indicating that the upgrade mode is activated and the upgrade is in progress.
5. Wait for the upgrade progress bar to complete and the interface to display “Upgrade Completed”—the upgrade is then finished.

**Warning : **

1. When installing and running the FeasyBlue App, please allow the App to access the permissions for **nearby devices**, **location information**, and **media and file access**. Otherwise, you may fail to search for nearby Bluetooth devices and load the firmware upgrade file.
2. If you use the “Obtain Firmware” method (importing the firmware upgrade file by entering the DFU name), note that the mobile phone must be connected to the Internet, and ensure the DFU name is entered correctly (case-sensitive). Otherwise, an error of “network or file error” may occur.
3. Do not disconnect the power during the upgrade process.

7.1.3 1.3 OTA Upgrade Show





7.2 2. UART Upgrade

7.2.1 2.1 UART Upgrade Tool

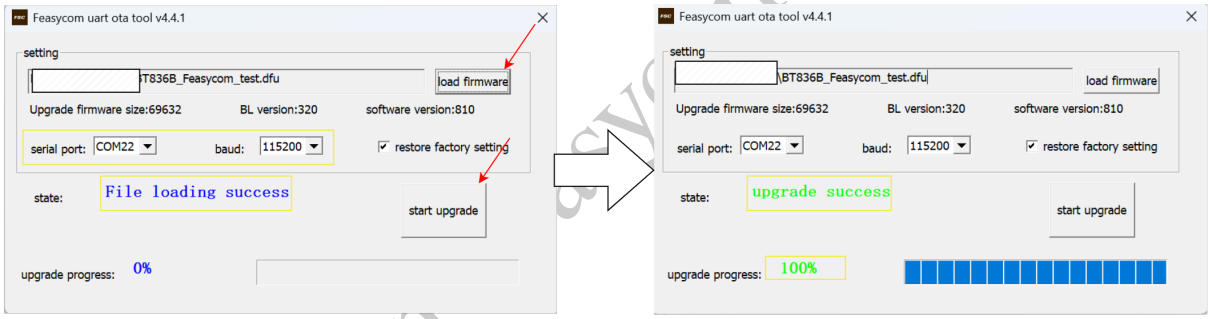
- Feasycom-UART-Upgrde-tool-4.6.zip (Based on Windows PC)

7.2.2 2.2 UART Upgrade Guide

- **Upgrade Firmware:** The **.dfu firmware upgrade file** provided by Feasycom.

1. Run the Feasycom UART Upgrde Tool ;
2. Enter the correct parameter configuration: **serial port**, **baud** ;
3. Click **load firmware** and select the **.dfu firmware upgrade file** stored locally on the PC (The firmware file and path shown in the image are for demonstration only; please use the actual file) ;
4. Click **start upgrade**; the tool interface will display **connection Successful** and simultaneously show the **upgrade progress**, indicating entry into upgrade mode ;
5. When the progress bar displays **100%** and the status shows **upgrade success**, the serial port upgrade is complete.

7.2.3 2.3 UART Upgrade Show



Chapter 8

FAQs

[[中文版]]常见问题汇总—FSC-BTDMD 2.3 documentation

8.1 1. Why do we need to use an app on a mobile phone for Bluetooth connection and communication?

Native phone Bluetooth functionality primarily supports common use cases like audio transfer and file sharing. Some Bluetooth peripheral devices can be connected via the phone’s built-in settings (e.g., Bluetooth speakers, headphones, keyboards, mice). However, when a peripheral device, like a module only supporting SPP/GATT protocols, cannot be connected via native phone settings, a specific mobile application, such as the FeasyBlue app, is generally required for connection.

8.2 2. How to get the Bluetooth MAC address on an iOS phone?

Due to security considerations, the iOS system converts the Bluetooth MAC address into a UUID at the underlying level before presenting it to upper-layer applications. Therefore, apps cannot directly obtain the device’s actual MAC address.

The FSC-BT836x series Bluetooth modules include the MAC address in their broadcasts by default. Apps can retrieve the MAC address from the advertisement packet using the following method.

```
- (void)centralManager:(CBCentralManager *)central
didDiscoverPeripheral:(CBPeripheral *)peripheral
advertisementData:(NSDictionary *)advertisementData RSSI:(NSNumber
*)RSSI
{
    if(![self describeDictonary:advertisementData])
    {
        NSLog(@"is not fsc module");
        return;
    }
}

- (Boolean)describeDictonary: (NSDictionary *) dict
{
    NSArray *keys;
    id key;
    keys = [dict allKeys];
    for(int i = 0; i < [keys count]; i++)
    {
        key = [keys objectAtIndex:i];
        if([key isEqualToString:@"kCBAdvDataManufacturerData"])
        {
            NSData *tempValue = [dict objectForKey:key];
            const Byte *tempByte = [tempValue bytes];
            if([tempValue length] == 6)
            {
                // tempByte Subsequent parameters are the Bluetooth
address
                return true
            }
        }else if([key isEqualToString:@"kCBAdvDataLocalName"])
        {
            //there is name
            //NSString *szName = [dict objectForKey: key];
        }
    }
```

(continues on next page)

(continued from previous page)

```
    return false;
}
```

Chapter 9

Appendix

[Download PDF Version]