BT671B User Guide

Release 5.1.7

Table of contents

Chapter 1

Introduction

[中文]

1.1 Description

This design guide is suitable for engineers developing FSC-BT671B series Bluetooth modules

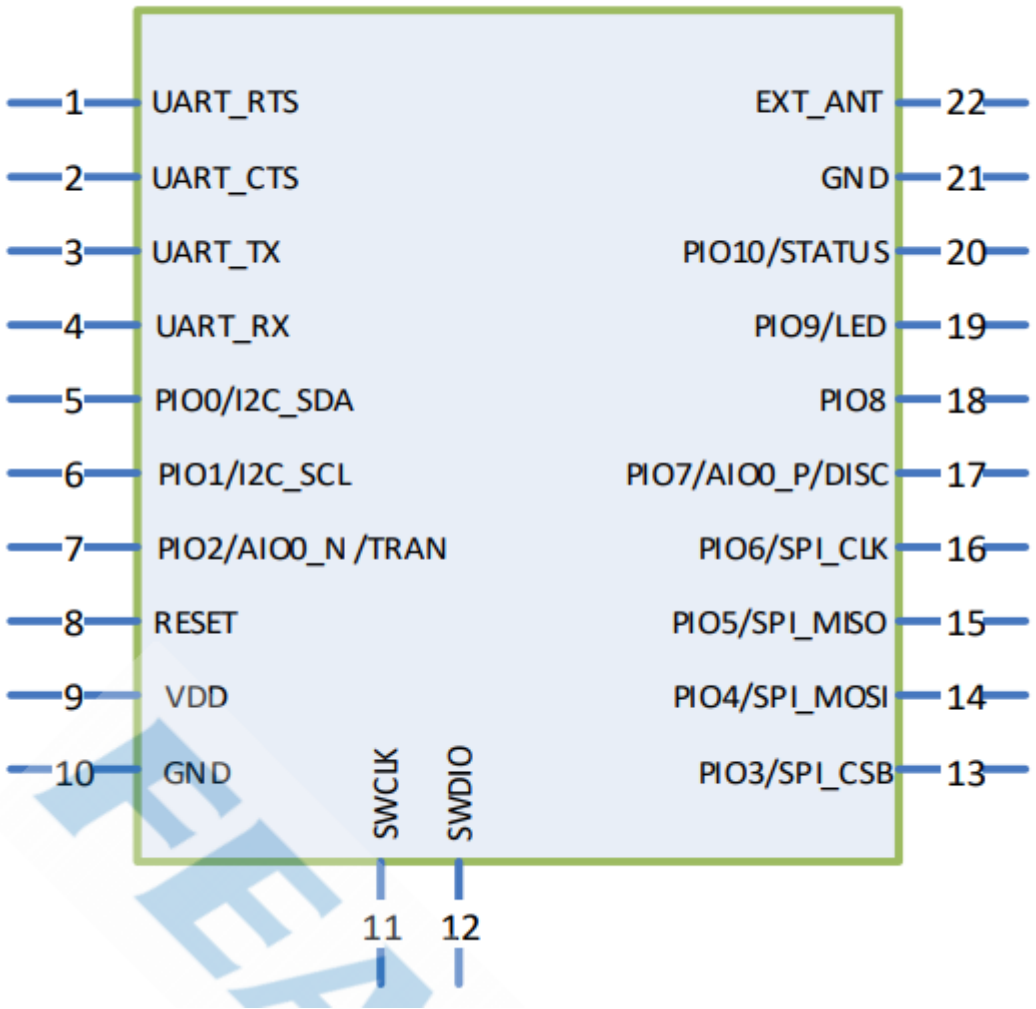
1.2 Module Default Settings

Name	FSC-BT671B
Service UUID	FFF0
Write UUID	FFF2
Notify UUID	FFF1
UART Baudrate	115200/8/N/1

Chapter 2

Hardware Description

2.1 Pin Diagram



2.2 Pin Description

Pin	Pin Name	Type	Pin Descriptions
1	UART_RTS	I/O	UART clear to send active low / NC
2	UART_CTS	I/O	UART request to send active low / NC
3	UART_TX	O	UART data output
4	UART_RX	I	UART data input
8	RESET	I	External reset input: Active LOW, with an inter an internal pull-up.
9	VDD	Power	Power supply voltage 3.3V
10	GND	GND	Power Ground
11	SWCLK	I/O	SWD CLK line(Default)
12	SWDIO	I/O	SWD DATA line(Default)
17	SLP_IND	O	Indicates the sleep state of the module, sleep output high level.
18	WAKE_UP	I	set high level enter sleep, set low level exit sleep.
19	LED	O	Bluetooth not connected output square wave, Bluetooth connection output high level
20	STATUS	O	Bluetooth not connected output low level, Bluetooth connection output high level
22	EXT_ANT	ANT	RF signal output

2.3 Hardware Design Notes

- The simple test of the module only needs to connect VDD/GND/UART_RX/UART_TX 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, so as to prevent the Bluetooth distance from not reaching the best effect
- The module does not have an antenna by default, you need to use an external antenna
- If need low power consumption, please connect PIN17 and PIN18

Chapter 3

Function Description

3.1 GPIO state

LED PIN is PIN 19

State	Description
1Hz square wave	disconnect state
high level	connected state

Connection status PIN is PIN 20

State	Description
low level	disconnect state
high level	connected state

3.2 Mode

Throughput mode	when disconnected, the data received by the serial port is analyzed according to the AT command; when connected, all the data received by the serial port is sent to the remote Bluetooth as it is.
Command mode	when disconnected, the data received by the serial port is analyzed according to the AT command; when connected, The data received by the serial port is still parsed according to the AT command. When it is necessary to send data to the remote end, send AT+LESEND command.

3.3 GATT Throughput Service

type	UUID	Permissions	Description
Service	0xFFF0		GATT Through-put Service
Write	0xFFF2	Write, Write Without Response	APP send data to module
Notify	0xFFF1	Notify	Module send data to remote

3.4 Performance parameters

type	Min.	Type.	Max.	Description
power on		230ms		loading finished
wake up		200ms		wake up from sleep

3.5 Data transfer rate

Baudrate	packet size	interval	conn inter-val	BLE method	rate
921600	244	16ms	15ms	Notify	50000 Byte/s

3.6 Low Power description

mode	enter sleep	exit sleep
AT+LPM=1	The serial port is idle for more than 5s The Bluetooth connected does not enter sleep	The first serial data frame wakes up enter connected exit sleep
AT+LPM=2	PIN18 input high level to enter sleep	PIN18 input low level to exit sleep

Chapter 4

AT command description

4.1 Command description

- Throughout this specification:
- Content between { } is optional
- Content behind << represents a COMMAND from Host
- Content behind >> represents a RESPONSE/EVENT to Host

4.2 Command

The command is the control command sent by the host to the module actively. After receiving the command, the module needs to reply **<CR><LF>OK<CR><LF>** as a response.

4.3 Command Format

AT+Command{=Param1{,Param2{,Param3…}}}<CR><LF>

- All commands start with “AT”, end with <CR><LF>
- <CR> means “carriage return”, corresponds to hex value 0x0D
- <LF> means “line feed”, corresponds to hex value 0x0A

- If Command has Parameter, Parameter follows behind ‘=’
- If Command has multiple Parameters, Parameter must be separated by ‘,’
- If Command has Response, Response starts with <CR><LF>, ends with <CR><LF>
- Module will always report command’s execution result by using “OK” for success or “ERR<code>” for failure

Example:

Read module’s name

<< AT+VER

>> +VER=1.0.0,FSC-BT671B

>> OK

Write invalid baudrate

<< AT+BAUD=0

>> ERROR

4.4 Event

Event is the data that the module actively sends to the host. Generally used to indicate a change of state or data received.

4.5 Event Format

<CR><LF>+Indication{=Param1{,Param2{,Param3…}}}<CR><LF>

- All Events start with <CR><LF>, end with <CR><LF>
- If Event has Parameter, Parameter follow behind ‘=’
- If Event has multiple Parameters, Parameter must be separated by ‘,’
- Use command AT+SEP to replace default separator for conflict prevention

Example:

Receive data from APP in command mode

>> +DATA=0,10,1234567890

Chapter 5

Commands Table

5.1 AT - Serial communication test

Command	**AT**
Response	**OK**
Description	When powering on or changing the baud rate, test the UART communication between the host and the module

Example:

<< AT

>> OK

5.2 AT+VER - Get Firmware Version

Command	**AT+VER**
Response	**+VER=Param**
`Param`	Firmware version
Description	Get Firmware Version

Example:

<< AT+VER

>> +VER=1.0.0,FSC-BT671B

>> OK

5.3 AT+ADDR - Get MAC Address

Command	**AT+ADDR**
Response	**+ADDR=Param**
Param	Module’s MAC address (12 Bytes ASCII)
Description	Get MAC Address

Example:

<< AT+ADDR

>> +ADDR=DC0D30010203

>> OK

5.4 AT+NAME - Get/Set Local Name

Command	**AT+NAME{=Param1{,Param2}}**
Param1	local name(1~29 Bytes ASCII)
Param2	MAC address suffix(0/1,default:0) 0: Disable suffix 1: Enable suffix “XXXX”(lower 4 bytes of MAC address) after local name
Response	**+NAME=Param**
Param	local name
Description	Write local name if parameter exist, otherwise read current local name

Example:

Read current local name

<< AT+NAME

>> +NAME=FSC-BT671B

>> OK

Change module’s local name to “ABC”

<< AT+NAME=ABC

>> OK

Change module’s local name to “ABC” and enable suffix

<< AT+NAME=ABC,1

>> OK

5.5 AT+BAUD - Get/Set Uart Baudrate

Command	**AT+BAUD{=Param}**
`Param`	baudrate (1200/2400/4800/9600/19200/38400/57600/115200/ 230400/460800/921600, default:115200)
Response	**+BAUD=Param**
`Param`	baudrate
Description	Module will change baudrate to target value immediately

Example:

Query baudrate

<< AT+BAUD

>> +BAUD=115200

>> OK

Change baudrate

<< AT+BAUD=9600

>> OK

5.6 AT+TXPOWER - Get/Set TX Power

Command	**AT+TXPOWER{=Param}**
`Param`	tx power
Response	**+TXPOWER=Param**
`Param`	tx power
Description	Get/Set TX Power

Example:

Get tx power

<< AT+TXPOWER

>> +TXPOWER=5

>> OK

Set tx power to 0dbm

<< AT+TXPOWER=0

>> OK

5.7 AT+LPM - Get/Set low power config

Command	**AT+LPM{=Param}**
`Param`	low power mode (0/1/2, default: 0) 0: disable 1: enable low power and wake up by UART 2: enable low power and wake up by IO
Response	**+LPM=Param**
`Param`	low power mode
Description	Get/Set low power config

Example:

Get low power config

<< AT+LPM

>> +LPM=0

>> OK

Set low power config

<< AT+LPM=1

>> OK

5.8 AT+ADVIN - Get/Set advertising interval

Command	**AT+ADVIN{=Param}**
`Param`	advertising interval (20~10000 ms, default: 152ms)
Response	**+ADVIN=Param**
`Param`	advertising interval
Description	Get/Set advertising interval

Example:

Get advertising interval

<< AT+ADVIN

>> +ADVIN=152

>> OK

Set advertising interval to 100ms

<< AT+ADVIN=100

>> OK

5.9 AT+LECHCNT - Get/Set the maximum number of connections

Command	**AT+LECHCNT{=Param}**
`Param`	Maximum number of connections allowed (1~8, default:1)
Response	**+LECHCNT=Param**
`Param`	Maximum number of connections allowed
Description	A restart is required after setting the maximum number of connections

Example:

Read the maximum number of connections

<< AT+LECHCNT

>> +LECHCNT=1

>> OK

Set a maximum of 8 connections

<< AT+LECHCNT=8

>> OK

5.10 AT+IDCFG - Get/Set 8 bytes ID

Command	**AT+IDCFG{=Param}**
`Param`	8 bytes ID
Response	**+IDCFG=Param**
`Param`	8 bytes ID
Description	Get/Set 8 bytes ID. Help users save 8 bytes IDs.

Example:

Get ID

<< AT+IDCFG

>> +IDCFG=12345678

>> OK

Set ID

<< AT+IDCFG=12345678

>> OK

5.11 AT+LEDISC - Disconnect specified connection

Command	**AT+LEDISC{=Param}**
`Param`	index
Response	**OK**
Description	Disconnect the bluetooth connection of the specified index

Example:

Disconnect device with index 1

<< AT+LEDISC=1

>> OK

Disconnect all connections

<< AT+LEDISC

>> OK

5.12 AT+LECCONN - Initiate a connection to the specified address

Command	**AT+LECCONN{=Param1{,Param2{,Param3{,Param4}}}}**
`Param1`	12-byte device address + 1-byte address type
`Param2`	Communication Service UUID
`Param3`	Communication UUID with write/write without rsp permission
`Param4`	Communication UUID with notification permission
Response	**OK**
Description	Initiate a connection to the specified device, the parameter consists of 12 bytes (device address) and 1 byte (address type), generally the address type is “0” or “1”. address How to get the address type: Send AT+SCAN=1 command, get parameter2. example: +SCAN=0,0,DC0D30001ED4,-65,10,FSC-BT946 Connection command: AT+LECCONN=DC0D30001ED40

Example:

Connect to the specified device, 0 is the address type

<< AT+LECCONN=DC0D3000039E0

>> OK

Initiate a connection to the specified address, using FFF0, FFF2, FFF1 for communication

<< AT+LECCONN=DC0D3000039E0,FFF0,FFF2,FFF1

>> OK

5.13 AT+FILTER - Get/Set scan filter

Command	**AT+FILTER{=Param1{,Param2}}**
`Param1`	scan filter (0~3) 0: delete all conditions 1: address filtering 2: advertising data filtering 3: RSSI filtering
`Param2`	The filter type is 1, and the parameter2 is the address The filter type is 2, and the parameter2 is advertising data The filter type is 3, and the parameter2 is RSSI
Response	**+FILTER=1,Param1** **+FILTER=2,Param2** **+FILTER=3,Param3**
Description	Filter commands will only affect AT+SCAN=2 When filtering advertising data, the parameter2 fields are: tag, offset, len, data After the configuration is complete, start AT+SCAN=2 to filter the scanned devices according to the set parameters AT+FILTER=0, delete all filter conditions. Filter parameters will not be saved to flash. Do not save when power off.

Example:

address filtering

<< AT+FILTER=1,DC0D30010203

>> OK

Filter advertising data \x is hexadecimal

<< AT+FILTER=2,\xFF,\x04,\x02,\xDC\x0D

>> OK

delete all filter conditions

<< AT+FILTER=0

>> OK

5.14 AT+SCAN - SCAN for nearby devices

Command	**AT+SCAN{=Param}**
Param	scanning method (0~2) 0: stop scanning 1: Scan nearby devices, filter duplicate addresses, up to 8 devices can be found 2: Scan BLE advertising packets and output original advertising packets, which can be filtered by AT+FILTER command
Response	**+SCAN=Param1,Param2,Param3,Param4,Param5,Param6** (AT+SCAN=1 event format) **+SCAN=Param21,Param22,Param23,Param24,Param25,Param26** (AT+SCAN=2 event format)
Param1	list number
Param2	MAC Address type (1 byte)
Param3	MAC address (12 bytes)
Param4	RSSI
Param5	Device name length
Param6	Device name
Param21	MAC Address type (1 byte)
Param22	MAC address (12 bytes)
Param23	RSSI
Param24	advertising packet type
Param25	advertising packet length
Param26	advertising packet
Description	AT+SCAN=1 Commonly used for searching before connecting AT+SCAN=2 Often used as a Bluetooth gateway to SCAN for nearby Beacon devices

Example:

SCAN device

<< AT+SCAN=1

>> +SCAN={

>> +SCAN=0,1,70CFC9A98840,-43,24,LE-Bose QuietControl 30

>> +SCAN=1,1,DC0D30001ED4,-65,10,FSC-BT946

>> +SCAN=}

Sniff the BLE advertising and return the raw data (the garbled part is the raw data, using hex format)

<< AT+SCAN=2

>> +SCAN{

>> +SCAN=0,DC0D24CFB6D0,-70,1,31,?. 曹 $ 隙衍觜 40#-BP104D

>> +SCAN}

5.15 AT+TPMODE - Get/Set throughput mode

Command	**AT+TPMODE{=Param}**
`Param`	Throughput mode (0~1, default 1) 0: Command Mode 1: Throughput Mode
Response	**+TPMODE=Param**
`Param`	Throughput mode
Description	When bluetooth connected and in throughput mode, the AT command will be de-active, every byte received via physical UART will be sent to remote.

Example:

Throughput Mode

<< AT+TPMODE=1

>> OK

Command Mode

<< AT+TPMODE=0

>> OK

5.16 AT+ADVEN - Get/Set advertising enable config

Command	**AT+ADVEN{=Param}**
`Param`	enable (0~1, default 1) 0: disable advertising 1: enable advertising
Response	**+ADVEN=Param**
`Param`	enable
Description	Trun on/off advertising

Example:

Turn on advertising

<< AT+ADVEN=1

>> OK

Turn off advertising

<< AT+ADVEN=0

>> OK

5.17 AT+ADVDATA - Get/Set advertising manufacturer specific data

Command	**AT+ADVDATA{=Param}**
`Param`	manufacturer specific data, tag 0xff
Response	**+ADVDATA=Param**
`Param`	manufacturer specific data, tag 0xff
Description	Get/Set advertising manufacturer specific data

Example:

Get advertising manufacturer specific data, \x is hexadecimal data

<< AT+ADVDATA

>> +ADVDATA=\x03\x01\x02

>> OK

Set advertising manufacturer specific data, \x is hexadecimal data

<< AT+ADVDATA=\x03\x01\x02

>> OK

5.18 AT+LESEND - send data to remote device

Command	**AT+LESEND{=Param1{,Param2{,Param3}}}**
`Param1`	Connection index
`Param2`	data len
`Param3`	data
Response	**OK**
Description	send data to remote device

Example:

Send 4-byte data to the remote device with connection index 1

<< AT+LESEND=1,4,2022

>> OK

5.19 AT+CHINFO - Get connection list infomation

Command	**AT+CHINFO**
Response	**+CHINFO=Param1,Param2,Param3,Param4**
`Param1`	Connection index
`Param2`	Connection state (1:not connected, 2: connecting, 3: connected)
`Param3`	Connection role (0: peripheral, 1: central)
`Param4`	peer mac address
Description	Get connection list infomation

Example:

Get connection list infomation

<< AT+CHINFO

>> +CHINFO=0,3,0,4F85B3319170

>> +CHINFO=1,3,0,6A8E79C29A99

>> +CHINFO=2,1,0,000000000000

>> +CHINFO=3,1,0,000000000000

>> +CHINFO=4,1,0,000000000000

>> OK

5.20 AT+REBOOT - Soft Reboot

Command	**AT+REBOOT**
Response	**OK**
Description	Module release all Bluetooth connections then reboot

Example:

<< AT+REBOOT

>> OK

5.21 AT+RESTORE - Restore Factory Settings

Command	**AT+RESTORE**
Response	**OK**
Description	Module restore all factory settings then reboot

Example:

<< AT+RESTORE

>> OK

5.22 AT+UARTCFG - Get/Set Uart config

Command	**AT+UARTCFG{=Param}**
`Param`	uart config (0~5) bit0: 0 - 1 stop bit 1 - 2 stop bits bit2~bit3: 00 - None 01 - Odd 10 - Even
Response	**+UARTCFG=Param**
`Param`	uart config (0~5)
Description	Get/Set Uart config

Example:

8 None 1

<< AT+UARTCFG=0

>> OK

8 Odd 1

<< AT+UARTCFG=2

>> OK

8 Even 1

<< AT+UARTCFG=4

>> OK

8 None 2

<< AT+UARTCFG=1

>> OK

8 Odd 2

<< AT+UARTCFG=3

>> OK

8 Even 2

<< AT+UARTCFG=5

>> OK

5.23 AT+PIN - Get/Set Pin Code

Command	**AT+PIN{=Param}**
`Param`	Pin Code (000000 ~ 999999, default 000000)
Response	**+PIN=Param**
`Param`	Pin Code
Description	Get/Set Pin Code

Example:

Get Pin Code

<< AT+PIN

>> +PIN=000000

>> OK

Set Pin Code 123456

<< AT+PIN=123456

>> OK

Chapter 6

Event Table

6.1 +SCAN - AT+SCAN=1 Scan Result

Indication	**+SCAN=Param1,Param2,Param3,Param4,Param5,Param6**
Param1	Index
Param2	MAC address type (1 byte)
Param3	MAC address (12 Bytes ASCII)
Param4	RSSI
Param5	remote device name len
Param6	remote device name
Description	After the module receives AT+SCAN=1, it will continue to scan, and report through +SCAN after finding the device

Example:

SCAN device

<< AT+SCAN=1

>> +SCAN={

>> +SCAN=0,1,70CFC9A98840,-43,24,LE-Bose QuietControl 30

>> +SCAN=1,1,DC0D30001ED4,-65,10,FSC-BT946

>> +SCAN=}

6.2 +SCAN - AT+SCAN=2 Scan Result

Indication	**+SCAN=Param1,Param2,Param3,Param4,Param5,Param6**
Param1	MAC address type (1 byte)
Param2	MAC address (12 Bytes ASCII)
Param3	RSSI
Param4	advertising packet type
Param5	advertising packet size
Param6	advertising packet
Description	After the module receives AT+SCAN=2, it will continue to scan and report through +SCAN after finding the device

Example:

Sniff the BLE advertising and return the original data (the garbled part is the original data, using hex format)

<< AT+SCAN=2

>> +SCAN{

>> +SCAN=0,DC0D24CFB6D0,-70,1,31,?. 曹 $ 隙衔觜 40#-BP104D

>> +SCAN}

6.3 +GATTSTAT - GATT State

Indication	**+GATTSTAT=Param1,Param2**
`Param1`	Connection Index
`Param2`	GATT State (1~3) 1: Standby 2: Connecting 3: Connected
Description	In the command mode, if the connection status of the module changes, it will be actively reported through +GATTSTAT

Example:

Connected

>> +GATTSTAT=0,3

6.4 +DATA - GATT Received Incoming Data

Indication	**+DATA=Param1,Param2,Param3**
`Param1`	Index
`Param2`	Payload length
`Param2`	Payload
Description	In the throughput mode: report data without +DATA header In the command mode: report data with +DATA header +DATA=1,5,12345

Example:

received data 1234567890

>> +DATA=1,10,1234567890

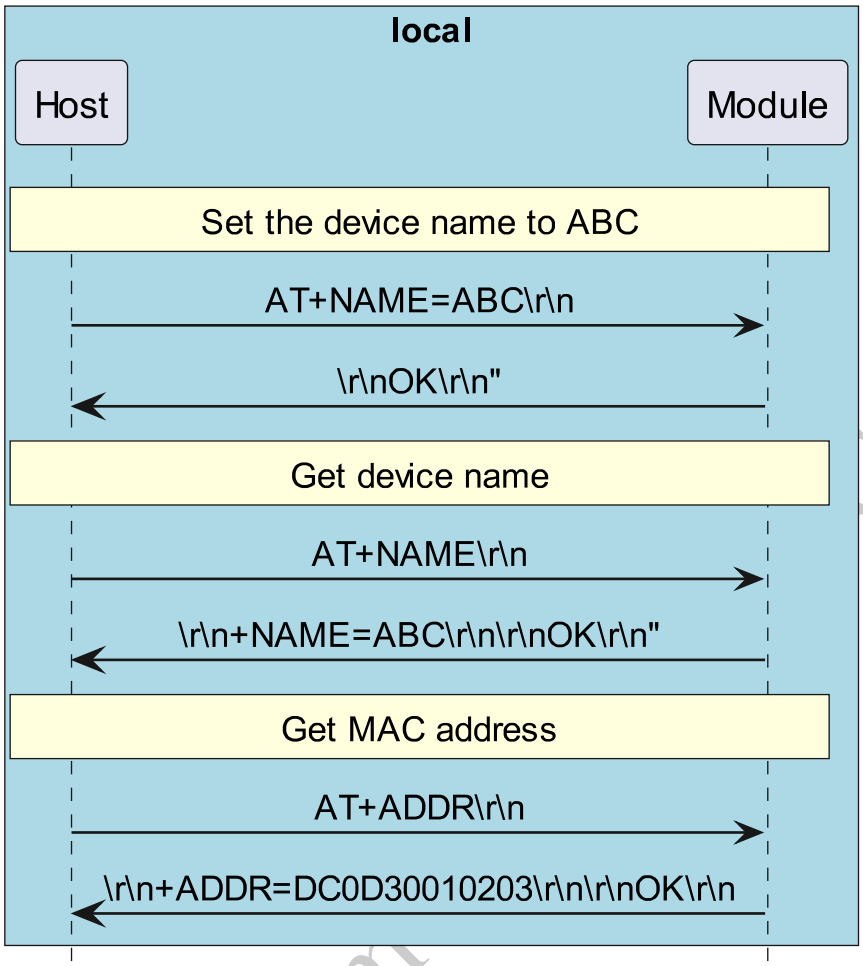
Chapter 7

Application scenarios

7.1 Get/Set the default parameters of the module

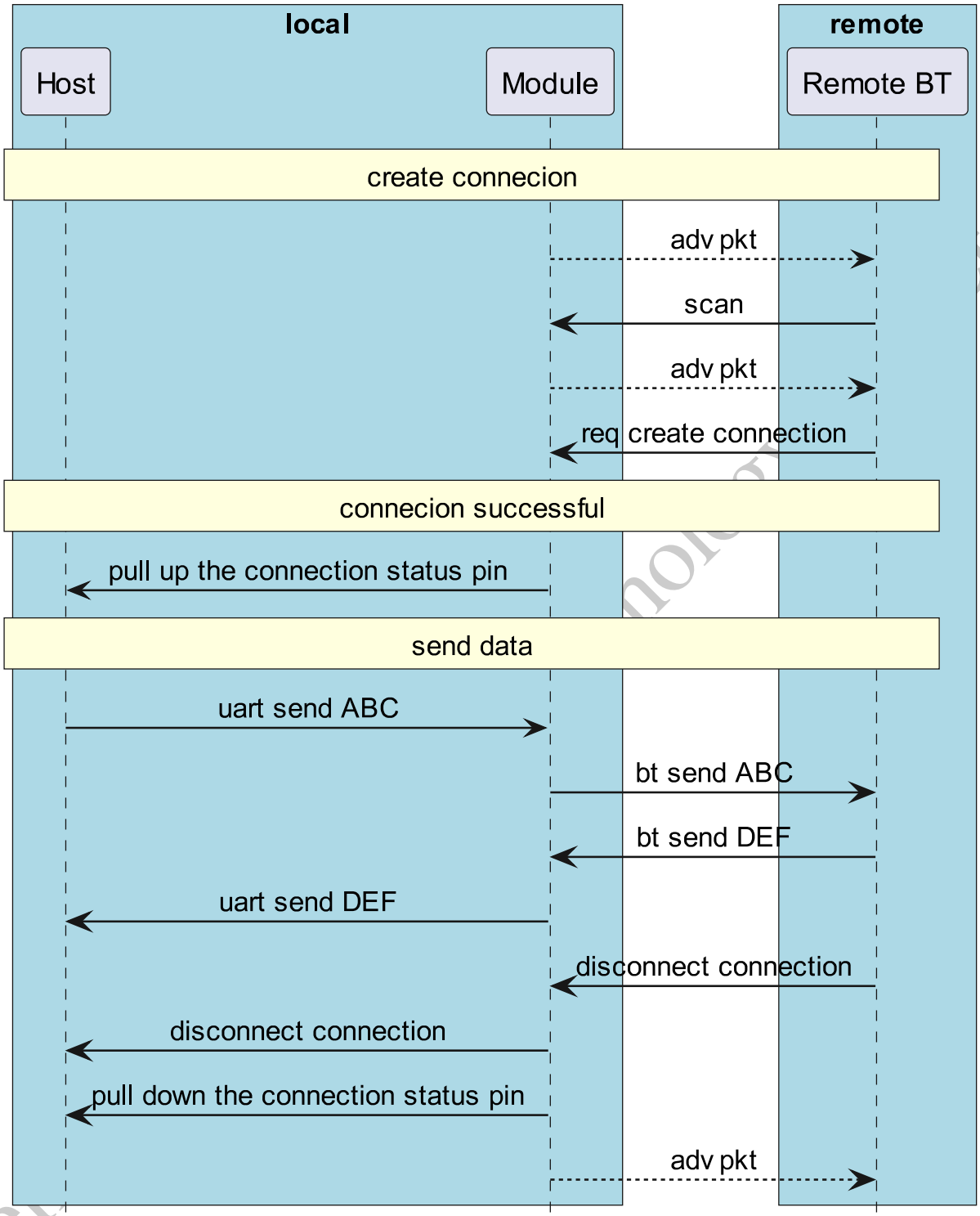
When the module is disconnect state, it will parse the uart data according to the AT command. The host can get and set the default parameters of the module, as shown in the figure below:

1. Set the device name to ABC
2. Get device name
3. Get MAC address



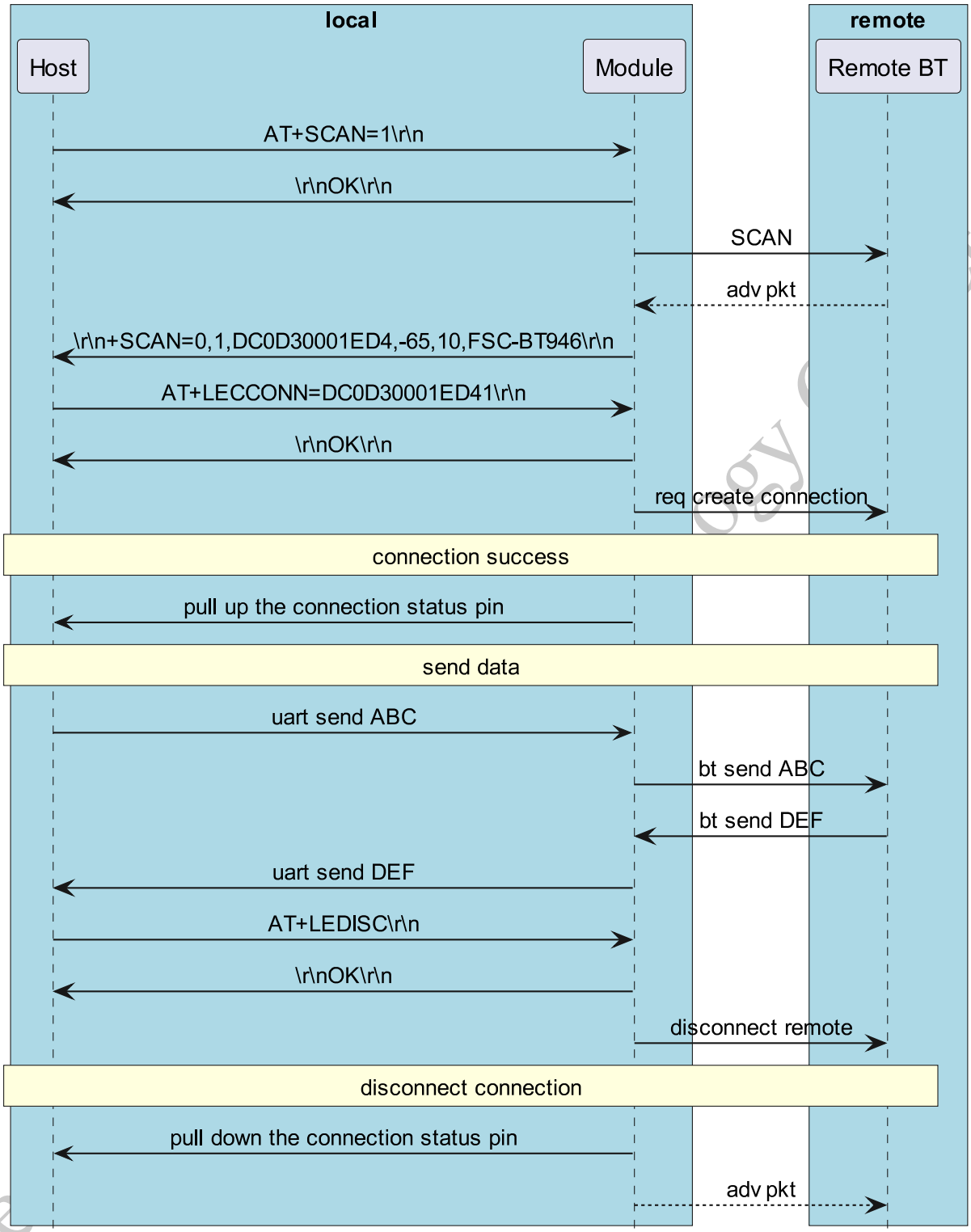
7.2 Process of sending data

When the module is powered on, it will continue to send advertising data, and the remote device (mobile phone) can scan the advertising packet. And initiate a connection request to the module. After the connection is successful, the module will pull up the connection status pin to notify the host that the Bluetooth connection is successful. The host can send data to the remote device through the Bluetooth module, and the remote device can also send data to the host.



7.3 The module acts as the central to connect to the remote device

The module can be used as a master device to connect to the slave device, and the host can send commands to control the module to scan and connect and disconnect. The figure below shows the process of connecting other devices:


local
remote
Host
Module
Remote BT
AT+SCAN=1\r\n
\r\nOK\r\n
SCAN
adv pkt
\r\n+SCAN=0,1,DC0D30001ED4,-65,10,FSC-BT946\r\n
AT+LECCONN=DC0D30001ED41\r\n
\r\nOK\r\n
req create connection
connection success
pull up the connection status pin
send data
uart send ABC
bt send ABC
bt send DEF
uart send DEF
AT+LEDISC\r\n
\r\nOK\r\n
disconnect remote
disconnect connection
pull down the connection status pin
adv pkt

Chapter 8

FAQ

8.1 How does an IOS mobile phone scan the Bluetooth MAC address?

For security reasons, the IOS system converts the Bluetooth MAC address into a UUID at the bottom layer and sends it to the upper layer application. Therefore, the APP cannot get the MAC address of the device.

FSC-BT671B will put the MAC address in the advertising packet by default, and the APP can get the MAC address from the advertising packet through the following methods.

```
- (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
{
```

(continues on next page)

(continued from previous page)

```
    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)
            {
                // The parameter after tempByte is the␣
↪Bluetooth address
                return true
            }
        }else if([key isEqualToString:@"kCBAdvDataLocalName
↪"])
        {
            //there is name
            //NSString *szName = [dict objectForKey: key];
        }
    }
    return false;
}
```

Chapter 9

Appendix

9.1 Download PDF Document

Download PDF Document