密级：公开资料

BDE-BLEM201

透传模块使用指南

Version 1.1

（文档仅适用 **固件版本：v1.1.1**）

广州碧德电子科技有限公司

2014 年 **3** 月 **28** 日

目　录

1. 概述

BDE-BLEM201 模块是基于 TI CC2541 芯片设计的兼容蓝牙 4.0 低功耗（BLE）单模蓝牙模块。它主要应用于智能穿戴式设备、便携式医疗设备、运动健身设备、智慧家庭、消费电子、工业控制等，可满足低功耗、低时延、近距离无线数据通信的要求。BDE-BLEM201 透传模块可以让开发者无须了解低功耗蓝牙协议，直接使用类似串口通信方式、开发支持低功耗蓝牙通信的智能产品。

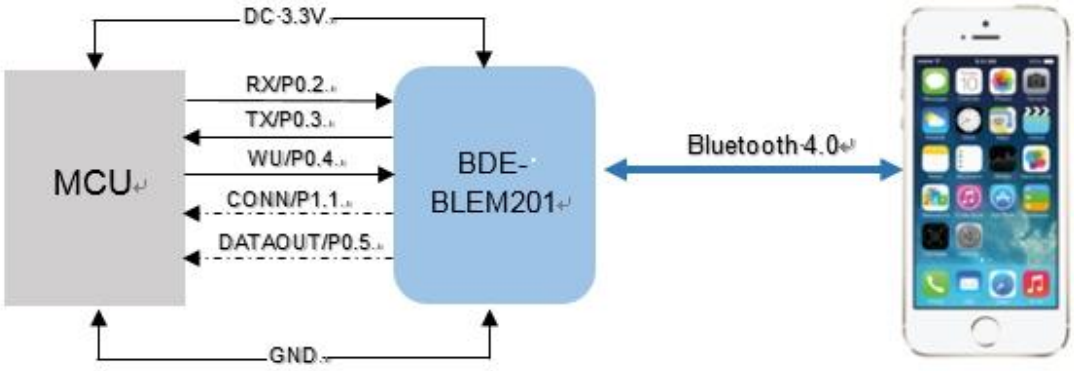
本文档是 BDE-BLEM201 透传模块的使用说明文档，包括模块的主要功能、应用场景、使用方法、逻辑结构、硬件接口及各项指标特性。

2. 应用

BDE-BLEM201 透传模块可把所有来自 MCU 的串口透传数据通过 BLE 无线信道透明传输给另一端设备。另一端设备可以是智能手机（iOS/Android），也可以是另一个 BDE-BLEM201 透传模块。

2.1 模块与手机连接通信

需先将 BDE-BLEM201 透传模块配置为从设备，手机作为主设备（碧德电子可为开发者提供 iOS/Android 透传库方便快速开发手机应用），在模块与手机成功建立连接后即可开始双向的数据透明传输。原理框图如图 2.1。



注：实线为必接线，虚线可根据需求考虑是否使用

图 2.1 模块与手机连接通信

2.2 模块间连接通信

需先将两模块分别配置为主从设备，在主从设备建立连接后即可进行双向的数据透明传输。原理框图如图 2.2。

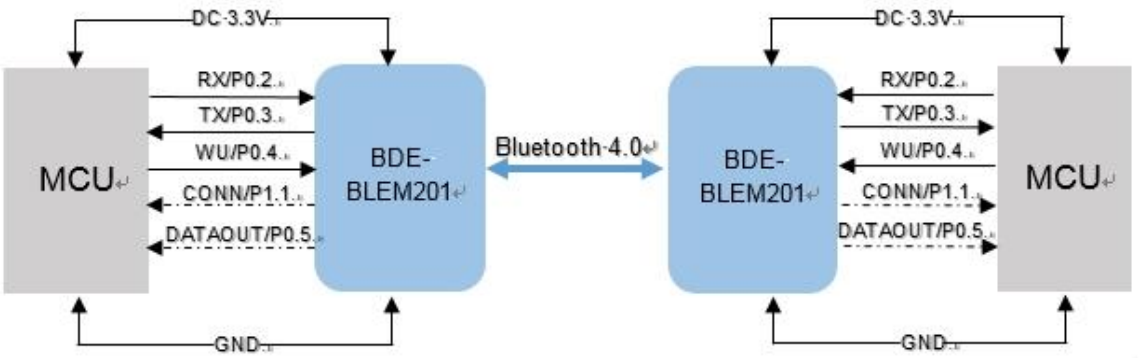


图 2.2 模块间连接通信

2.3 模块间广播通信

除了通过一对一的连接方式(2.1 和 2.2 都是连接方式通信)进行双向的数据传输外，BDE-BLEM201 透传模块还支持通过一对多的广播方式实现单向（从设备到主设备）的数据传输，一个主设备可以同时获得多个从设备的广播数据。模块与 MCU 的接线参考图 2.1 或图 2.2。

为实现模块间广播通信，开发者需要保持主设备扫描和从设备广播一直都为开启状态，通过向从设备发送 setAdvData 指令来更新从设备的广播数据（具体串口指令参见 3.2 节）。需要注意的是，由于主设备扫描和从设备广播一直都是开启的，主设备可能多次扫描到一个从设备的同一广播数据包，因此开发者需要做好重复报文的过滤工作。常用的重复报文的过滤方法有：利用广播数据的第一个字节作为报文序号，每次更新广播数据时将报文序号加 1，主设备根据这个报文序号判断接收到的广播数据是否为新的数据。注：报文序号溢出后将归零。

模块间广播通信不要求主从设备建立连接，常被应用于数据量小且是偶发的多点数据采集系统中，如环境温度、体重等数据采集系统。

3. 使用说明

3.1 模块出厂默认配置

BDE-BLEM201 透传模块的出厂默认配置如下表：

参数	默认值
串口配置	115200bps，no parity, one stop bit
模块名字	“bde spp dev”
模块角色	从模块
模块状态	广播
广播间隔	200ms
发射功率	0dbm
TX 延迟发送	5ms

3.2 数据透明传输

3.2.1 工作准备

开发者在应用开发前，需参考第二章原理框图、第四章模块封装引脚定义和 3.1.2、3.1.3 节描述将 MCU 和透传模块连接好，确保 MCU 和模块的串口能正常通信。

当需要在 BDE-BLEM201 透传模块与手机（iOS/Android）之间实现透传通信时，需先将 BDE-BLEM201 透传模块配置为从模块。当需要进行模块间通信时，需要分别配置主模块和从模块。主从模块配置好后，开发者可通过一对一的连接方式或一对多的广播方式来进行数据传输。主从模块配置方法参见 3.3 节。

为方便初期调试/评估，碧德电子为开发者提供了 PC 端的串口调测工具“BDE-SerialPort-BLEM201CMD”（使用方法参见该工具说明书），只需用 USB 转串 Dongle 连接透传模块和 PC，即可操作所有串口配置指令；碧德电子还为开发者提供了 iOS 端配合串口透传模组使用的通讯软件“BDE Utility”，用 SPS 功能可进行 BLE 数据透传。

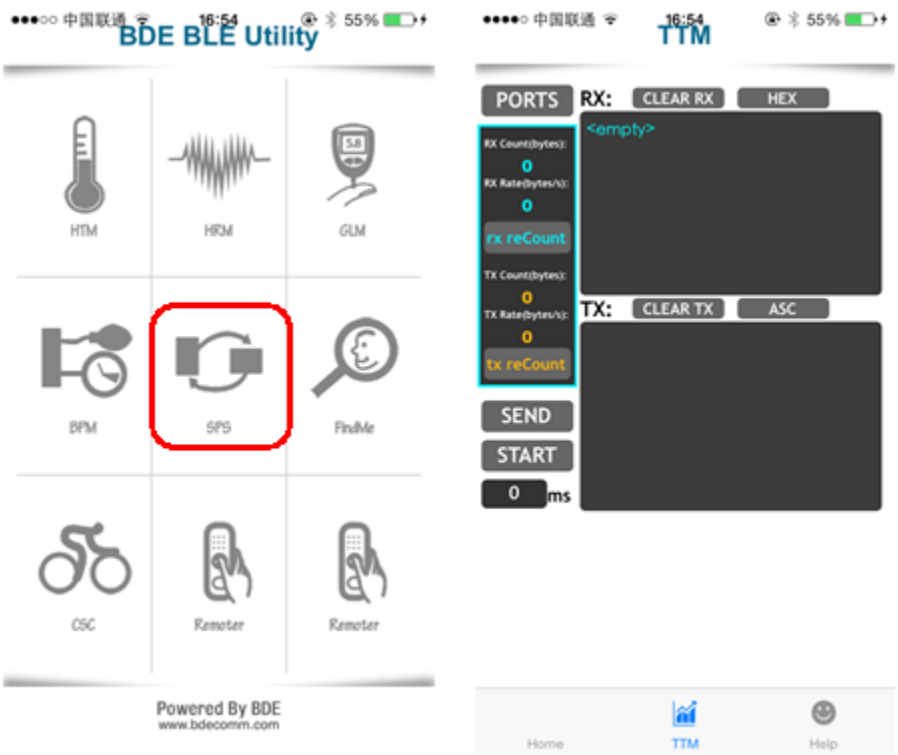


图 3.1 BDE Utility 串口透传测试软件

3.2.2 数据发送

MCU 在向模块发送串口数据前需先拉低 WU/P0.4 引脚来唤醒模块。在数据发送期间，WU/P0.4 必须始终保持为低电平，数据发送完毕后需要再拉高 WU/P0.4 引脚，使其再次进入睡眠，降低功耗。

每次向 BDE-BLEM201 透传模块连续发送的串口数据不能超过 200 个字节。

3.2.3 数据接收

模块一旦接收到另一端设备的 BLE 数据后，在转发给 MCU 之前将自动拉低 P0.5 引脚以通知 MCU 接收数据。数据发送期间 P0.5 引脚保持为低电平。待全部串口数据发送完毕后，P0.5 引脚将被拉高。

3.3 串口指令

BDE-BLEM201 透传模块的串口数据流中包含两种数据类型：串口指令和透传数据。利用 BDE-BLEM201 透传模块丰富的串口指令集，开发者可以设置模块主从角色、波特率、广播和连接间隔，开启或关闭广播，开启或关闭扫描，连接或断开连接等。

BDE-BLEM201 透传模块的指令数据和透传数据在同一数据流中，因此指令数据需要一定的指令封装格式和指令约束条件，以便模块能从串口数据流中分离出指令数据。

3.3.1 指令格式

BDE-BLEM201 透传模块的指令数据是 ASCII 码。它由四部分组成，如下所示：

指令开始标志	指令码	参数列表	指令结束标志

- 指令开始标志：**”SPP:**”

  这四个连续的字符用于指示一个指令包的开始。

- 指令码：

  指令的具体标识，用于区分不同的指令。如:setRole。

- 参数列表：

  不同的指令有不同的参数、不同的参数数目（大于等于 0）。指令中每两个参数间需要用空格分隔。指令码和参数列表之间也需要用空格分隔。

- 指令结束标志：”\r\n\0”

  这 3 个字符(即回车符、换行符和空字符)用于指示一个指令包的结束。

  例：设置模块为从设备: “**SPP: setRole p\r\n\0**”

  其中，”SPP:”为指令开始标志；”setRole”为指令码，表示切换角色操作；”p”为参数，表示将模块设置为从设备；”\r\n\0”为指令结束标志。

3.3.2 指令约束

使用 BDE-BLEM201 透传模块的指令数据有三个约束条件：

1) 一条完整的指令应该要被连续地发送到模块，或一条完整的指令的发送总时间不能超过 100ms。(在 9600bps 波特率下，发送一个字节大约需要 1ms，连续发送 60 个字节大约需要 60ms。而一条正确的指令的长度一般在 30 个字节之内，因此只要字节是连续发送地就不必担心指令超时问题)
2) “SPP:”和”\r\n\0”之间的数据不能超过 50 个字节。
3) 使用正确的、已定义了的指令码。如：“SPP: setrole p\r\n\0”数据包会被理解为透传数据，因为 setrole(r 没有大写)不是正确的指令码。

注意事项：透传模块的指令数据和透传数据在同一个数据流中，指令数据只会作用到模块状态参数切换。当开发者希望将指令数据作为透传数据传输到另一端设备时，可使用 sendData 指令来实现。

3.3.3 指令集

BDE-BLEM201 透传模块拥有丰富的串口指令集。

- 主模块支持的指令：

指令码	功能
setRole	设置模块角色
getRole	获取模块角色
setName	设置模块名字
getName	获取模块名字
setBR	设置波特率
getBR	获取波特率
setTxDly	设置串口输出延时
getTxDly	获取串口输出延时
setDBM	设置发射功率
getDBM	获取发射功率
setConnInt	设置连接参数
getConnInt	获取连接参数
setScan	开启/关闭扫描
getAddr	获取模块的地址
connect	连接指定的从设备
disconnect	断开连接
getStatus	查询模块当前状态
saveConfigure	保存当前配置
clearConfigure	清除保存的配置
sendData	发送指定长度的透传数据
getVersion	获取模块固件版本号
setDirectConnAddr	设置该主设备上电直接连接的从设备地址
getDirectConnAddr	获取该主设备上电直接连接的从设备地址

- 从模块支持的指令：

指令码	功能
setRole	设置模块角色
getRole	获取模块角色
setName	设置模块名字
getName	获取模块名字
setBR	设置波特率
getBR	获取波特率
setTxDly	设置串口输出延时
getTxDly	获取串口输出延时
setDBM	设置发射功率
getDBM	获取发射功率
setAdvInt	设置广播间隔
getAdvInt	获取广播间隔
setAdvData	设置广播数据
getAdvData	获取广播数据
setAdv	开启/关闭广播
setConnInt	设置连接参数
getConnInt	获取连接参数
getAddr	获取模块的地址
disconnect	断开连接
getStatus	查询模块当前状态
saveConfigure	保存当前配置
clearConfigure	清除保存的配置
sendData	发送指定长度的透传数据
getVersion	获取模块固件版本号
setConnectableAddr	设置允许连接该从设备的主设备地址
getConnectableAddr	获取允许连接该从设备的主设备地址

- 模块发送给 MCU 的响应指令

指令码	功能
ok	操作成功
err	操作失败
dev	扫描响应

BDE-BLEM201 透传模块的串口指令使用说明参见附录 A。

3.4 配置主从模块

3.4.1 配置主模块

将 BDE-BLEM201 透传模块配置为主设备的步骤如下：

1) 向模块发送 setRole 指令：“SPP: setRole c\r\n\0”；
2) MCU 接收到 ok 响应后，开发者可根据实际需要设置模块的名字(setName)、波特率(setBR)、发射功率(setDBM)等；
3) 向模块发送开启扫描指令：“SPP: setScan on\r\n\0”；
4) 成功开启扫描后，模块会将扫描到的从设备以 dev 指令响应给 MCU：“SPP: dev addr FF:22:11:22:33:FF name =bde spp dev\r\n\0”；
5) 向模块发送 connect 指令使主设备连接指定的从设备，如：“SPP: connect FF:22:11:22:33:FF\r\n\0”（开发者可通过向从设备发送 getAddr 指令获取从设备地址）。连接成功后，模块的 CONN/P1.1 引脚会被置为 0。

注意：设置完所有参数后，开发者需要发送 saveConfigure 指令将设置的参数（包括模块的状态）保存到 flash 中，以便模块以后每次重新上电都以本次设置的参数初始化模块。如不发送 saveConfigure 指令，模块重新上电后会恢复设置前的参数，而不是开发者本次设置的参数。

3.4.2 配置从模块

将 BDE-BLEM201 透传模块配置为从设备的步骤如下：

1) 向模块发送 setRole 指令：“SPP: setRole p\r\n\0”；
2) MCU 接收到 ok 响应后，开发者可根据实际需要设置模块的广播间隔(setAdvInt)、名字(setName)、波特率(setBR)、发射功率(setDBM)等；
3) 发送开启广播指令：“SPP: setAdv on\r\n\0”；
4) 成功开启广播后，便可以等待主设备或手机(iOS/Android)上支持 SPP Service 的 APP 进行扫描连接。从设备被连接后就可以进行数据透明传输了。

注意：设置完所有参数后，开发者需要发送 saveConfigure 指令将设置的参数（包括模块的状态）保存到 flash 中，以便模块以后每次重新上电都以本次设

置的参数初始化模块。如不发送 saveConfigure 指令，模块重新上电后会恢复设置前的参数，而不是开发者本次设置的参数。

3.4.3 配置配对的主从模块

一对配对的主从模块在上电时能自动建立连接，其配置步骤如下：

1) 向两模块分别发送 setRole 指令："SPP: setRole c\r\n\0"和"SPP: setRole p\r\n\0"；
2) 向两模块分别发送 getAddr 指令来获取两模块的地址："SPP: getAddr\r\n\0"；
3) 获得地址后，再分别使两模块记录对方的地址。向从设备发送 setConnectableAddr 指令，向主设备发送 setDirectConnAddr 指令。如：从设备的地址为 FF:11:11:11:11:FF，主设备的地址为 FF:22:22:22:22:FF，向从设备发送"SPP: setConnectableAddr FF:22:22:22:22:FF\r\n\0" 指令，向主设备发送"SPP: setDirectConnAddr FF:11:11:11:11:FF\r\n\0"指令。
4) 分别向两模块发送 saveConfigure 指令。
5) 重新上电后两模块便会自动建立连接。

如果想取消自动连接使模块能连接上其他设备，需将 setConnectableAddr 和 setDirectConnAddr 指令的参数全设为 0，即向主从设备分别发送"SPP: setDirectConnAddr 00:00:00:00:00:00\r\n\0" 和 "SPP: setConnectableAddr 00:00:00:00:00:00\r\n\0"指令，然后再保存（saveConfigure）。或者直接向两模块发送 clearConfigure 指令，下次上电时两模块都会恢复出厂默认配置。

3.5 SPP Service

BDE-BLEM201 透传模块的数据透明传输功能是由 SPP Service/Profile 来实现的。SPP Service 相关的 UUID 如下表：

类型	UUID	属性
SPP Service	0xFFB0	NC
SPP Data Characteristic	0xFFB2	Write without Response、Notify
SPP Command Characteristic	0xFFB1	Write、Notify

SPP Data Characteristic 用于实现数据的透明传输。该 characteristic 的属性为 Write without Response 和 Notify。在进行数据传输前，应先使能 SPP Data

Characteristic 的 Notify 属性，即将 SPP Data Characteristic 的 Client Characteristic Configuration 的值更改为 0x0001。

SPP Command Characteristic 用于实现对模块进行空中配置或获取配置参数。该 characteristic 的属性为 Write 和 Notify。在对该 characteristic 操作前，应先使能 SPP Command Characteristic 的 Notify 属性，即将 SPP Command Characteristic 的 Client Characteristic Configuration 的值更改为 0x0001。(当前版本还没有实现该功能)

4. 模块封装与引脚定义

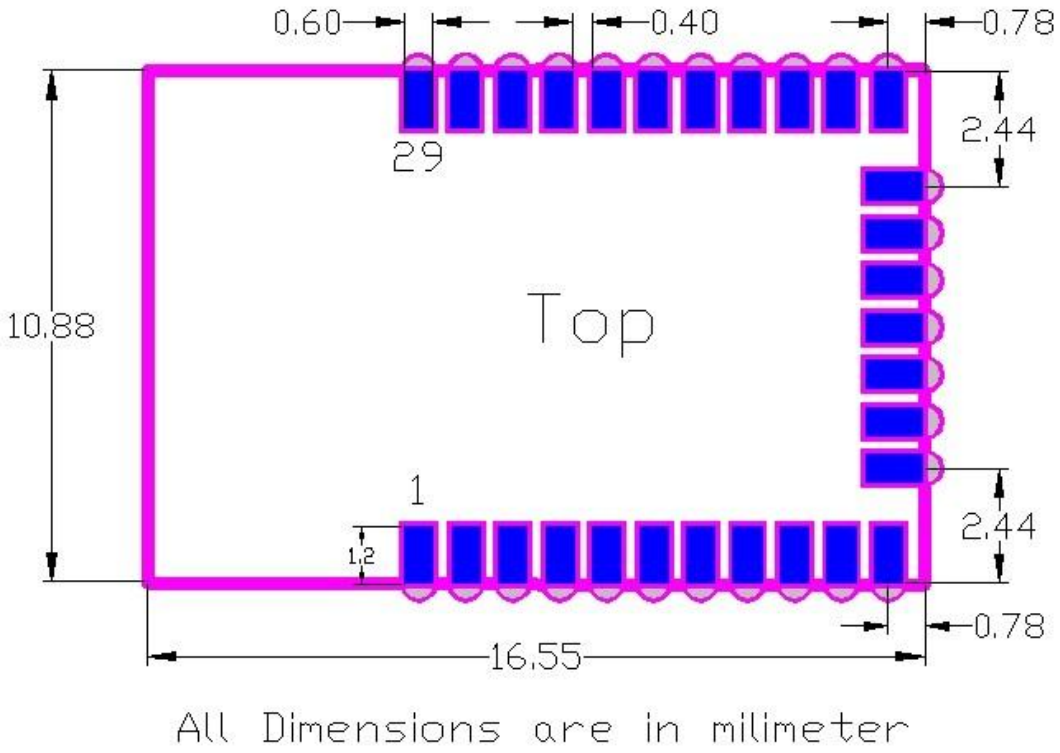


图 4.1 模块尺寸

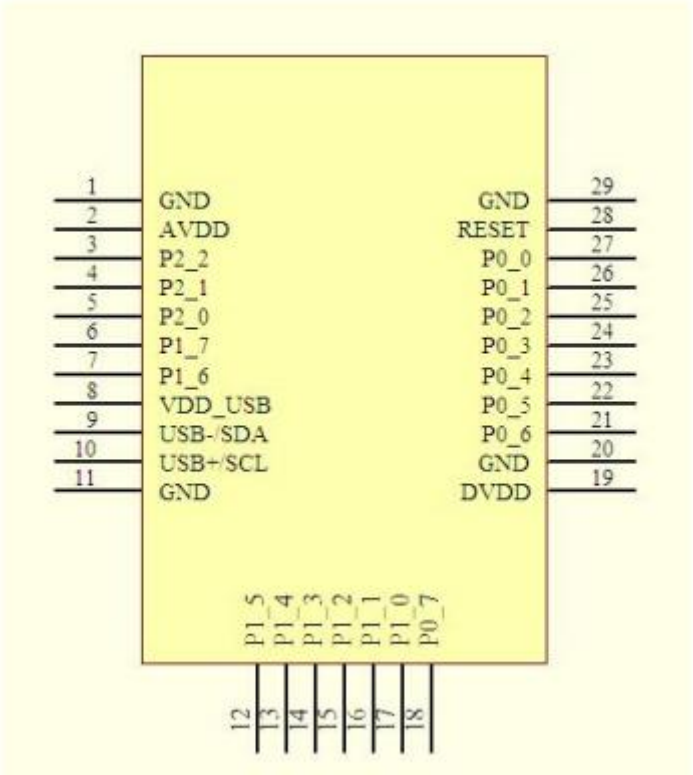


图 4.2 模块引脚分布

表 4.1 模块引脚定义

引脚名称	引脚序号	I/O	说明
UART_RX/P0.2	25	I	模块串口接收端
UART_TX/P0.3	24	O	模块串口发送端
WU/P0.4	23	I	模块唤醒引脚，往模块发送 UART 数据前需下拉该引脚，发送期间需保持低电平，发送完毕后拉高该引脚以便模块能重新进入睡眠
DATAOUT/P0.5	22	O	模块串口数据输出指示，可用于唤醒 MCU： 0：模块有数据需要发送 1：模块无数据需要发送
CONN/P1.1	16	O	模块连接状态指示： 0：模块处于连接状态 1：模块处于未连接状态
VDD	2, 19	-	电源输入正极
GND	1, 11, 20, 29	-	电源地

5. 性能指标

5.1 数据传输速率

下表为模块间在连接间隔为 7.5ms 时的数据传输速率（数据传输方向：主模块到从模块）。

表 5.1 模块间数据传输速率

连接间隔 (*1.25ms)	发送间隔 (ms)	每次发送字节数	发送速率 (kB/s)	总字节数		丢包率 (百分比)
				发送	接收	
6	10	40	4	667720	667720	0
6	10	40	4	400000	400000	0
6	10	40	4	400000	399959	0.01
6	10	41	4.1	410000	409639	0.09
6	10	42	4.2	420000	420000	0
6	10	42	4.2	420000	419860	0.03
6	10	43	4.3	430000	429678	0.07
6	20	80	4	1200000	1199180	0.07
6	20	80	4	80000	80000	0
6	20	90	4.5	450000	449260	0.16
6	30	120	4	600000	600000	0
6	30	120	4	1200000	1200000	0
6	30	123	4.1	492000	492000	0
6	30	123	4.1	1230000	1230000	0
6	30	126	4.2	504000	503960	0.01
6	30	130	4.33	650000	649939	0.01

由上表可知，当发送间隔为 30ms、每次发送字节数为 123 时，模块间的传输速率可以比较稳定地达到 4.1KB/s。

下表为模块与 iOS 在连接间隔为 18.75ms 时实测的数据传输速率（数据传输方向：iOS 到模块）。

表 5.2 模块与 iOS 的数据传输速率

连接间隔 (*1.25ms)	发送间隔 (ms)	每次发送字节数	发送速率 (kB/s)	总字节数		丢包率 (百分比)
				发送	接收	
15	30	100	3.33	154500	154500	0
15	30	105	3.5	302820	302820	0

15	30	110	3.67	239470	239360	0.05
15	30	108	3.6	425952	425952	0
15	40	144	3.6	361152	361152	0
15	50	180	3.6	423900	423900	0
15	50	180	3.6	1301760	1301760	0

由上表可知，当发送间隔为 50ms、每次发送字节数为 180 时，模块与 iOS 间的数据传输速率可以达到 3.6KB/s。

5.2 功耗

下表为用 Agilent 66319B 电源实测的模块在各种状态下的功耗数据(供电电压为 3.3V)：

表 5.3 模块功耗

状态	广播/连接间隔(ms)	平均电流(uA)
空闲	-	0.11
广播	20	1200
	100	282
	300	100
	1000	32
	2000	18
	3000	13
连接	18.75	1585
	30	990
	60	490
	97.5	310
	150	201
	236.25	135
	300	108
	461.25	70
	997.5	35
	1623.75	25

6. 电气特性

表 6.1 绝对最大额定值

参数	最小值	最大值	单位
储存温度	-40	125	℃
VDD	-0.3	3.9	V
其他管脚	-0.2	VDD+0.3≤3.9	V

表 6.2 推荐的运行条件

参数	最小值	推荐值	最大值	单位
温度	-40	-	85	℃
VDD	2	3.3	3.6	V

附录 A 串口指令说明

- **setRole**

作用：设置模块为主从模块。
支持的角色：主、从
参数个数：1 个
参数取值：

参数值(模块角色)	含义
p	设置模块为从设备
c	设置模块为主设备

说明：

使用该指令时，如果目标角色与当前模块角色不一致，模块会断开连接、停止广播或扫描，然后进入空闲状态。如果一致，模块当前状态将不会被改变。无论属于哪种情况，下面的响应指令都会被产生。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0”(操作失败，reason 的具体值参见附录 B)

- **getRole**

作用：获取模块当前角色

支持的角色：主、从

参数个数：无

响应：

“SPP: ok p\r\n\0”(模块为从设备)

“SPP: ok c\r\n\0”(模块为主设备)

“SPP: err reason\r\n\0”(操作失败，reason 的具体值参见附录 B)

- **setName**

作用：设置模块名字

支持的角色：主、从

参数个数：1 个

参数取值：

参数值(名字)	含义
=string	string 为具体的名字字符串（不能出现”\r\n\0”）

说明：

该指令的参数为’=’与”\r\n\0”之间的内容，允许出现空格，但不允许出现”\r\n\0”。名字的最大长度不能超过 20 个字节。

例：

设置模块的名字为“ bde dev”：“SPP: setName = bde dev\r\n\0”

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0”(操作失败，reason 的具体值参见附录 B)

- **getName**

作用：获取模块名字

支持的角色：主、从

参数个数：无

响应：

“SPP: ok = bde dev\r\n\0”(操作成功，设备名字为” bde dev”)

“SPP: err reason\r\n\0”(操作失败，reason 的具体值参见附录 B)

- **setBR**

作用：设置模块的波特率(bps)

支持的角色：主、从

参数个数：1 个

参数取值：

参数值(波特率)	含义
9600	设置模块波特率为 9600bps
19200	设置模块波特率为 19200bps
38400	设置模块波特率为 38400bps
57600	设置模块波特率为 57600bps
115200	设置模块波特率为 115200bps

说明：

模块只支持上表中的 5 个波特率，且其默认的波特率为 115200bps。当 MCU 向模块发送该指令后，模块会先返回 ok 指令，然后再改变自身的波特率。MCU 在接收到 ok 指令后，也应该改变自己的波特率，以保持和模块一致，避免产生误码。

注意：若忘记了已设置的波特率，开发者可以尝试在不同的波特率下发送 getRole(或其他 get 指令)，直到得到正确的回复，以此来确定模块当前的波特率。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getBR**

作用：获取模块波特率(bps)

支持的角色：主、从

参数个数：无

响应：

“SPP: ok 115200\r\n\0”(模块波特率为 115200bps)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setTxDly**

作用：设置模块串口输出延迟时间（ms）

支持的角色：主、从

参数个数：1 个

参数取值：

参数值(延迟时间)	含义
n	设置串口输出延迟 n ms

说明：

串口输出延迟是配合 DATAOUT/P0.5 引脚使用的，目的是让 MCU 有足够的时间从睡眠状态中唤醒，从而正确的接收模块发送给 MCU 的串口数据。当模块有串口数据要发送给 MCU 时，会先将 P0.5 引脚拉低，延迟指定的时间后，再发送串口数据。在全部的串口数据发送完成后，模块又会将 P0.5 引脚置为高电平。串口输出延迟时间默认为 5ms。该值不应该设置得过大，避免模块因串口数据没有及时发送出去而导致串口缓冲区溢出，造成数据丢失。DATAOUT/P0.5 引脚和串口数据输出的关系图如下：

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getTxDly**

作用：获取串口输出延迟时间（ms）

支持的角色：主、从

参数个数：无

响应：

“SPP: ok n\r\n\0”(n 为延迟的时间值，如 5、8 等)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setDBM**

作用：设置模块的发射功率（dbm）

支持的角色：主、从

参数个数：1 个

参数取值：

参数值(发射功率)	含义
0	设置模块的发射功率为 0dbm
4	设置模块的发射功率为 4dbm
-6	设置模块的发射功率为-6dbm
-23	设置模块的发射功率为-23dbm

说明：

模块发射功率的默认值为 0dbm。模块只支持上表中四个功率值。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getDBM**

作用：获取模块的发射功率（dbm）

支持的角色：主、从

参数个数：无

响应：

“SPP: ok n\r\n\0”(n 表示具体的功率值)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setAdvInt**

作用：设置模块广播间隔（625us）

支持的角色：从

参数个数：2 个

参数取值：

参数	取值	含义
参数 1(最小广播间隔)	min	设置模块最小广播间隔为 min*625us
参数 2(最大广播间隔)	max	设置模块最大广播间隔为 max*625us

说明：

模块的最大、最小广播间隔默认为 320（单位：625us）。广播间隔的有效范围为 20ms~10.24s。最大广播间隔不能小于最小广播间隔。广播间隔越大，广播时模块功耗就越低。设置广播间隔成功后，需要重新开启广播。

例：设置模块的最小广播间隔为 80*625us、最大广播间隔为 100*625us 的指令为：“SPP: setAdvInt 80 100\r\n\0”。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getAdvInt**

作用：获取广播间隔（625us）

支持的角色：从

参数个数：无

响应：

“SPP: ok min max\r\n\0”(min 为最小广播间隔，max 为最大广播间隔，单位：625us)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setAdvData**

作用：设置广播数据

支持的角色：从

参数个数：1 个

参数取值：

参数取值(自定义广播数据)	含义
=data	设置模块的广播数据为 data

说明：

由于该指令的参数中允许出现空格，因此需要在设定的数据前加上一个字符’=’，模块会将’=’到“\r\n\0”之间的数据作为有效的数据，因此数据中不能出现”\r\n\0”，广播用户自定义数据最大长度为 22 个字节。

例：

“SPP: setAdvData =0123456789\r\n\0”

当想去除广播数据时，发送空参数即可：“SPP: setAdvData =\r\n\0”。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getAdvData**

作用：获取当前广播数据

支持的角色：从

参数个数：无

响应：

“SPP: ok =data\r\n\0”(data 为当前广播数据)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setAdv**

作用：开启或关闭广播

支持的角色：从

参数个数：1 个

参数取值：

参数取值(动作)	含义
on	开启广播
off	关闭广播

说明：

该指令只有在从设备处于非连接状态时才会被执行。

例：

“SPP: setAdv on\r\n\0”(开启广播)

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setConnInt**

作用：设置连接参数

支持的角色：主、从

参数个数：4 个

参数取值：

参数	取值	含义
参数 1(最小连接间隔)	min	设置最小连接间隔为 min*1.25ms
参数 2(最大连接间隔)	max	设置最大连接间隔为 max*1.25ms
参数 3(latency)	lat	设置 latency 为 lat
参数 4(连接超时)	timeout	设置连接超时为 timeout*10ms

说明：

- 最小连接间隔和最大连接间隔：取值范围均为：0x0006~0x0C80，最大连接间隔不能小于最小连接间隔。连接间隔越大，功耗就越低，传输速率也越低。
- Latency：取值范围为：0x0000~0x01F3。
- 连接超时：取值范围为：0x000A~0x0C80。连接间隔必须小于连接超时时间，即：[max*1.25 + (1 + latency)] < 10 * timeout。
- 模块更新连接参数时需要等待另一端设备参与响应，响应时间跟连接间隔有关，连接间隔越短，响应时间越短，反之亦然,。因此，有时会存在延迟几秒才响应的现象。成功更新连接参数后，模块会响应 ok 指令；更新失败或更新超时时，模块会响应 err 指令，超时时间为 10s。由于模块指令执行时具有单步性，因此在得到响应之前再往模块发送其他指令时都会返回 err，故开发者发送了更新连接参数指令后应该要等待接收到相应的响应后才执行其他操作。
- 如果模块当前处于非连接状态，使用 setConnInt 指令将会返回错误：SPP:err notConn\r\n\0。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getConnInt**

作用：获取连接参数

支持的角色：主、从

参数个数：无

响应：

“SPP: ok interval latency timeout\r\n\0”(interval 为实际的连接间隔，timeout 为连接超时)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setScan**

作用：开启或关闭扫描

支持的角色：主

参数个数：1 个

参数取值：

参数取值(动作)	含义
on	开启扫描
off	关闭扫描

说明：

该指令只能在主设备处于非连接状态下使用。成功开启扫描后，模块会先响应 ok 指令。当模块扫描到从设备时会以 dev 指令回传给 MCU。直到模块连接上一个从设备或关闭扫描，模块才会停止回传 dev 指令。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: dev addr 87:BC:D6:13:33:88 name =bde spp dev\r\n\0”

“SPP: dev addr 87:BC:D6:13:33:88 dat =0123456789\r\n\0”(如果从设备设置了广播数据)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getAddr**

作用：获取模块的地址

支持的角色：主、从

参数个数：无

响应：

“SPP: ok FF:11:22:33:55:FF\r\n\0”

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **connect**

作用：连接指定的从设备

支持的角色：主

参数个数：1 个

参数取值：

参数取值(从设备地址)	含义
XX:XX:XX:XX:XX:XX	连接地址为 XX:XX:XX:XX:XX:XX 的从设备

说明：

地址用 ASCII 码表示，每两个地址字节间用':'分隔，如下：
"SPP: connect 87:BC:D6:13:33:88\r\n\0"

响应：

"SPP: ok\r\n\0"(操作成功)

"SPP: err reason\r\n\0" (操作失败，reason 的具体值参见附录 B)

- **disconnect**

作用：模块主动断开连接

支持的角色：主、从

参数个数：无

说明：

成功断开连接后，模块会回到空闲状态。

响应：

"SPP: ok\r\n\0"(操作成功)

"SPP: err reason\r\n\0" (操作失败，reason 的具体值参见附录 B)

- **getStatus**

作用：获取模块运行状态

支持的角色：主、从

参数个数：无

说明：

从设备的运行状态有三种：空闲、广播和连接（"idle"、"adv"和"connected"）。主设备的运行状态有五种：空闲、有响应的扫描、无响应的扫描、正在连接和已经连接（"idle"、"scan rsp"、"scan norsp"、"connecting"、"connected"）。"scan norsp"和"scan rsp"的区别是：前者是当模块作为主设备时被动地断开连接后(即由从设备断开连接或由于其他因素断开连接)，模块会尝试重新连接上从设备，即开启扫描后搜索上次连接的从设备，但此时的扫描内容是不会上传给 MCU 的。后者是当模块作为主设备时， MCU 向模块发送了 setScan 指令主动地开启了扫描，此时模块的扫描内容会上传给 MCU。

响应：

“SPP: ok idle\r\n\0”(模块处于空闲状态)

“SPP: ok adv\r\n\0”(模块处于广播状态)

“SPP: ok connected\r\n\0”(模块处于连接状态)

“SPP: ok scan rsp\r\n\0”(带响应扫描)

“SPP: ok scan norsp\r\n\0”(不带响应扫描)

“SPP: ok connecting\r\n\0”(正在连接)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **saveConfigure**

作用：保存当前的配置

支持的角色：主、从

参数个数：无

说明：

该指令用于将当前模块的参数和状态保存到 flash 中，以便模块在下次上电时会根据保存好的参数和状态进行初始化设置。若在从设备广播或连接的状态下发送该指令，从设备以后每次重新上电都会自动进行广播；在主设备扫描或连接的状态下发送该指令，主设备以后每次重新上电时会自动进行扫描；若在空闲的状态下发送该指令，模块以后每次重新上电时也会处于空闲状态(有一种情况例外：在空闲状态下配置一对自动连接的主从设备后，以后每次重新上电两模块都会自动进入 scan without respond 和广播状态)。这时需要发送 setAdv、setScan 指令才能开启广播、扫描，而且如果想模块在下次上电时自动广播、扫描，需要发送 saveConfigure 指令。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **clearConfigure**

作用：清除保存的配置

支持的角色：主、从

参数个数：无

说明：

向模块成功发送该指令后，模块下次上电时会恢复出厂时的配置。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **sendData**

作用：发送指定长度的透传数据

支持的角色：主、从

参数个数：1 个

例：

“SPP: sendData 10\r\n\00123456789”

“sendData”与”\r\n\0”之间的”10”代表需要发送的透传数据为 10 个字节，即”\r\n\0”后的 10 个字节。模块接收到该指令时不会对”\r\n\0”后指定长度的数据进行指令分析，而是将这些数据全部认为是透传数据。因此可以利用这条指令来发送指令数据包。如下：

“SPP: sendData 18\r\n\0SPP: setBR 9600\r\n\0”

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getVersion**

作用：获取模块固件版本

支持的角色：主、从

参数个数：无

响应：

“SPP: ok v1.0.0\r\n\0”(当前固件版本为 v1.0.0)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setConnectableAddr**

作用：设置允许连接该从设备的主设备地址

支持的角色：从

参数个数：1 个

参数取值：

参数取值(主设备地址)	含义
XX:XX:XX:XX:XX:XX	设置允许连接模块的主设备地址为 XX:XX:XX:XX:XX:XX

说明：

setConnectableAddr 指令用于防止不相关的主设备连接模块。如果想取消这一功能，需要向模块发送“SPP: setConnectableAddr 00:00:00:00:00:00\r\n\0”，后发送 saveConfigure 指令。

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getConnectableAddr**

作用：获取允许连接该从设备的主设备地址

支持的角色：从

参数个数：无

响应：

“SPP: ok FF:11:22:33:66:FF\r\n\0”

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **setDirectConnAddr**

作用：设置主模块上电后直接连接的从设备地址

支持的角色：主

参数个数：1 个

参数取值：

参数取值(从设备地址)	含义
XX:XX:XX:XX:XX:XX	设置模块上电自动连接的从设备地址为 XX:XX:XX:XX:XX:XX

说明：

setDirectConnAddr 指令用于实现模块上电后自动建立连接的功能。发送完该指令后，模块下次上电会自动扫描搜索后连接地址为 XX:XX:XX:XX:XX:XX 的从设备。将从设备地址参数全设为 0 可以取消该功能(别忘了 saveConfigure 指令)：“SPP: setDirectConnAddr 00:00:00:00:00:00\r\n\0”

响应：

“SPP: ok\r\n\0”(操作成功)

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **getDirectConnAddr**

作用：获取主模块直接连接的从设备地址

支持的角色：主

参数个数：无

响应：

“SPP: ok 87:BC:D6:13:33:88\r\n\0”

“SPP: err reason\r\n\0” (操作失败，reason 的具体值参见附录 B)

- **ok**

作用：操作成功响应

说明：ok 指令由模块发送给 MCU，其参数根据操作的不同而不同。

- **err**

作用：操作失败响应

说明：err 指令由模块发送给 MCU，其参数见附录 B。

- **dev**

作用：扫描响应

说明：dev 指令由模块发送给 MCU。它包含了从设备的相关信息，可分为两种信息类型：名字和数据。

例：

“SPP: dev addr FF:11:22:33:66:FF name =bde spp dev\r\n\0”(名字信息)

“SPP: dev addr FF:11:22:33:66:FF dat =0123456789\r\n\0”(数据信息)

附录 B 操作错误码

err 为操作错误码。当开发者发送的指令不被模块接受时，模块会返回 err 响应：“SPP: err reason\r\n\0”。

err 的参数值和含义如下表：

Reason	含义
“procNotDone”	上一个操作未结束
“invalidParam”	指令的参数不合法
“notConn”	模块处于未连接状态
“alreadyConn”	模块处于连接状态
“advertising”	模块正在广播
“procFailure”	操作失败

附录 C 手机 APP 编程建议

1. 使用碧德电子提供的 iOS、Android 透传库开发

碧德电子将基于蓝牙 4.0 通信的串口透传功能封装成库。基于此库，第三方开发人员不需要对蓝牙 4.0 的术语及概念进行全面的了解，而只需要具备串口透传开发经验甚至于只需要熟悉文件系统操作即可方便快捷地开发基于蓝牙串口透传的应用。库提供的接口将蓝牙设备虚拟成一个端口(Port)，所有操作都是基于端口的操作，包括扫描端口、打开端口、接收数据、读取数据、发送数据等。库包括 iOS 和 Android 两个版本，分别在下面进行描述。

● iOS 平台透传库

首先利用

+(BLESerialComManager *)sharedInstance;

得到单例实例，然后进行以下操作可以进行相关参数的一些设置(暂时不需要)

-(resultCodeType)configure:(paramsPackage4Configure)params

在对端口进行操作前，需要对端口进行扫描发现：

-(resultCodeType)startEnumeratePorts:(float)timeout;

然后在以下代理中返回搜索到的端口

-(void)bleSerilaComManager:(BLESerialComManager*)bleSerialComManager didFoundPort:(BLEPort*)port;

当端口扫描超时的接口

-(void)bleSerilaComManagerDidEnumComplete:(BLESerialComManager *)bleSerialComManager;

停止端口扫描

-(resultCodeType)stopEnumeratePorts;

打开端口

-(resultCodeType)startOpen:(BLEPort*)port withParams:(paramsPackage4Open)params;

端口打开成功与否的结果返回

-(void)bleSerilaComManager:(BLESerialComManager*)bleSerialComManager didOpenPort:(BLEPort*)port withResult:(resultCodeType)result;

接收数据的过程，首先在以下接口里面收到数据的提示

-(void)bleSerialComManager:(BLESerialComManager*)bleSerialComManager didDataReceivedOnPort:(BLEPort*)port withLength:(unsigned int)length;

这时只是收到接收数据的长度，需要再调用读数据接口，将数据读取出来，完成数据的接收过程

-(NSData *)readDataFromPort:(BLEPort*)port withLength:(int)length;

其中可以利用接口-(resultCodeType)clearReadBufferInPort:(BLEPort*)port 清空接收缓冲区。

在数据的发送过程，直接将数据作为参数，调用数据写的接口，完成数据的写过程

-(resultCodeType)writeData:(NSData*)data toPort:(BLEPort*)port;

使用完毕后，关闭串口

-(resultCodeType)closePort:(BLEPort*)port;

-(void)bleSerialComManager:(BLESerialComManager*)bleSerialComManager didClosedPort:(BLEPort*)port;

- **Android 平台透传库**

首先使用 enumAllPorts(float time)枚举所有的端口，在

void didFoundPort(blePort newPort)回调中返回每一个搜到的端口

利用接口

void stopEnum()

停止枚举端口

当完成枚举端口后回调

public void didFinishedEnumPorts();

当打开端口成功后回调

public void didOpenPort(blePort port,Boolean bSuccess);

当关闭端口成功后回调

public void didClosePort(blePort port);

当收到数据时，会回调以下函数

public void didPackageReceived(blePort port,byte[] packageToSend);

调用以下函数发送数据

public void writePort(byte[] value)

当 ACSUtility 不用时，要调用 closeACSUtility 关闭 ACSUtility，以便系统释放资源。

2. 使用 iOS、Android SDK 开发

基于 iOS/Android 原生态的 SDK 进行开发，除了正确使用接口以外，还要理解蓝牙 4.0 规范中的相关概念，熟悉蓝牙 4.0 的相关规范，才能更好地使用接口。

● iOS SDK

以下是 iOS 端基于低功耗蓝牙原生态 SDK 开发接口 Corebluetooth.framework 与串口模块实现串口透传的操作流程。

1) 搜索设备

在串口模块的广播包中包含有 FFB0 的服务 ID，可以指定搜索此类设备，过滤其他设备。

2) 发现服务

在连接完成后，请求发现服务的过程中，应当发现包含有 FFB0 的服务，并且服务应当包含有命令属性 0xFFB1 和数据属性 0xFFB2。

3) 打开以下两个 characteristic 的 notification 属性

完成服务发现后，使能 notification 的属性，串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)和数据属性(UUID:0xFFB2)。

4) 手机模式转换

向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送手机模式命令字(0x06,0x01)。

5) 进入透传模式

向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送进入透传模式命令字(0x01)。

6) 向模块发送周期握手包(8051 平台才使用此机制，其他平台没有)

a) 模块会通过向手机串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送周期性握手包(0x05)。

b) b 收到 a 步骤中的握手包后向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送握手响应包(0x07)。

c) 再次收到周期性握手包，重复 a 过程。

7) 正常收发数据

a) 发送数据(write property):向模块中串口服务(UUID:0xFFB0)当中的数据属性(UUID:0xFFB2)写数据。

b) 接收数据(notification property):模块向串口服务(UUID:0xFFB0)当中的数据属性(UUID:0xFFB2)发送数据。

● **Android SDK**

1) 搜索设备

mBluetoothAdapter.stopLeScan(mLeScanCallback);

2) 连接设备

mBluetoothGatt = device.connectGatt(this, false, mGattCallback);

3) 发现服务

在连接完成后，请求发现服务

mBluetoothGatt.discoverServices()；

4) 打开以下两个 characteristic 的 notification 属性

完成服务发现后，使能 notification 的属性，串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)和数据属性(UUID:0xFFB2)。

5) 手机模式转换

向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送手机模式命令字(0x06,0x01)。

6) 进入透传模式

向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送进入透传模式命令字(0x01)。

7) 向模块发送周期握手包(8051 平台才使用此机制，其他平台没有)

a) 模块会通过向手机串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送周期性握手包(0x05)。

b) 收到 a 步骤中的握手包后向模块中串口服务(UUID:0xFFB0)当中的命令属性(UUID:0xFFB1)发送握手响应包(0x07)。

c) 再次收到周期性握手包，重复 a 过程。

8) 正常收发数据

a) 发送数据(write property):向模块中串口服务(UUID:0xFFB0)当中的数据属性(UUID:0xFFB2)写数据。

b) 接收数据(notification property):模块向串口服务(UUID:0xFFB0)当中的数据属性(UUID:0xFFB2)发送数据。