



Coex 评估测试 使用手册

版本: 1.0

版权 @ 2020

www.bouffalolab.com

1 准备	4
2 烧录	5
2.1 连接	5
2.2 软件下载	7
2.3 串口工具配置	8
3 测试准备	10
4 测试评估一: wifi ping + ble adv	11
5 测试评估二: wifi ping + ble connect(1s 发送 5 次数据包, 长度为 23byte)	14
6 测试评估三: wifi running iperf + ble connect(1s 发送 5 次数据包, 长度为 23byte)	20

List of Figures

2.1	正面	6
2.2	背面	7
2.3	烧写工具界面	8
2.4	串口工具	9
4.1	模块成功连接 WiFi	11
4.2	模块开启 ping	12
5.1	APP 扫描到 BLE	15
5.2	BLE 连接成功	16
6.1	模块开启 ipc	20
6.2	PC 端 lperf 开启 sever 模式	21
6.3	APP 扫描到 BLE	23
6.4	BLE 连接成功	24

1. 硬件：BL602 模块一个，Windows PC 一台，USB 转串口线一根。
2. 软件：烧写工具，烧录 bl602_demo_event.bin 文件，路径：bouffalolab_release_bl_iot_sdk.zip/App_Demos/bl602_demo_event/build_out/bl602_demo_event.bin，选择任意一款串口工具
3. 手机上下载任意一款蓝牙调试 APP.

2.1 连接

BL602 模块的相关引脚连接如下图所示，其中图 1 是模块的正面图，其标号 1 处用跳线帽短接，标号 2 处将左边两根排针短接，标号 3 处将上面的两根排针短接；图 2 是模块的背面图，烧录时将 IO8 和 HI 两根排针短接，烧录完成后将 IO8 和 LOW 两根排针短接并重新上电。用 USB 线连接 PC 和模块，此时模块上的电源灯常亮，表明模块通电正常。

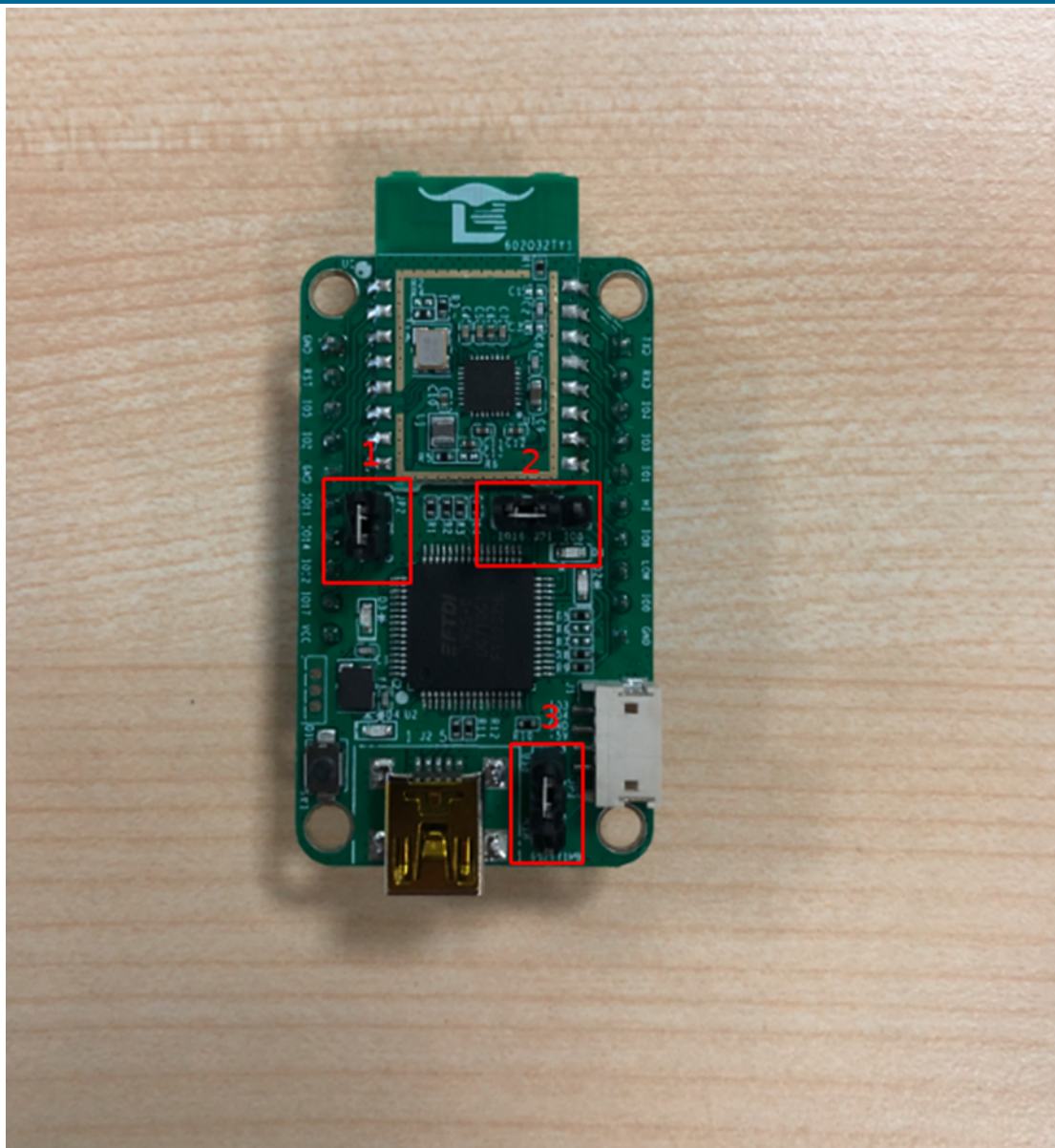


图 2.1: 正面



图 2.2: 背面

2.2 软件下载

打开解压后文件中的烧写工具 `flash_tool` 目录，双击 `BLDevCube.exe`，chip type 选择 BL602/604，打开后界面参数参考下图配置：

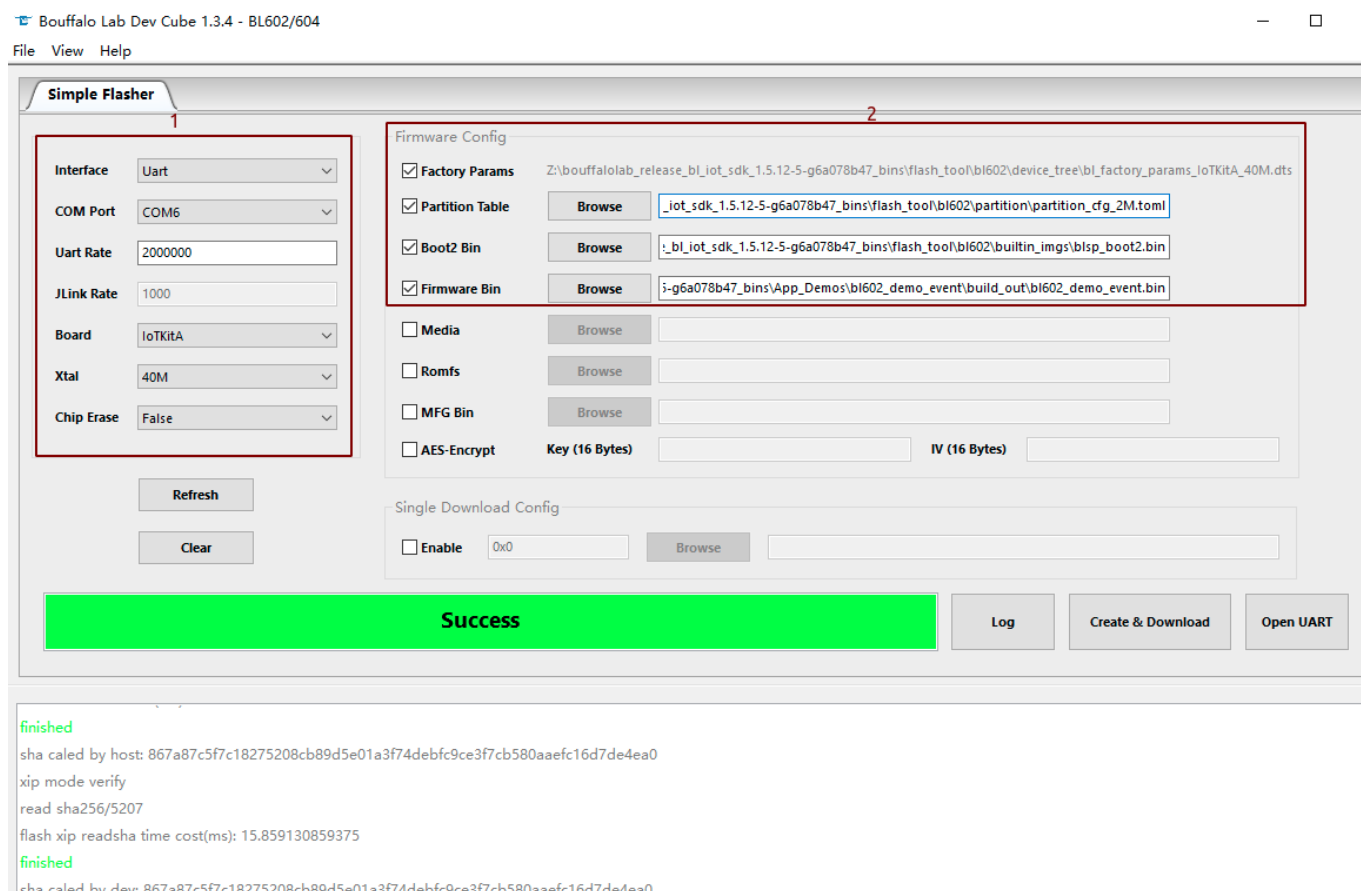


图 2.3: 烧写工具界面

其中图 3 的框 1 中 COM Port 选项根据实际串口情况选择（右击我的电脑-> 管理-> 设备管理器-> 端口，查看端口号，模块是双串口，选择端口号较小的），框 2 中的相关路径依据实际情况选择。配置完成后点击 Download 按钮下载。

2.3 串口工具配置

将 IO8 和 LOW 两根排针短接并重新上电，打开串口工具，设置对应的端口号，波特率设定为 2000000 bps。

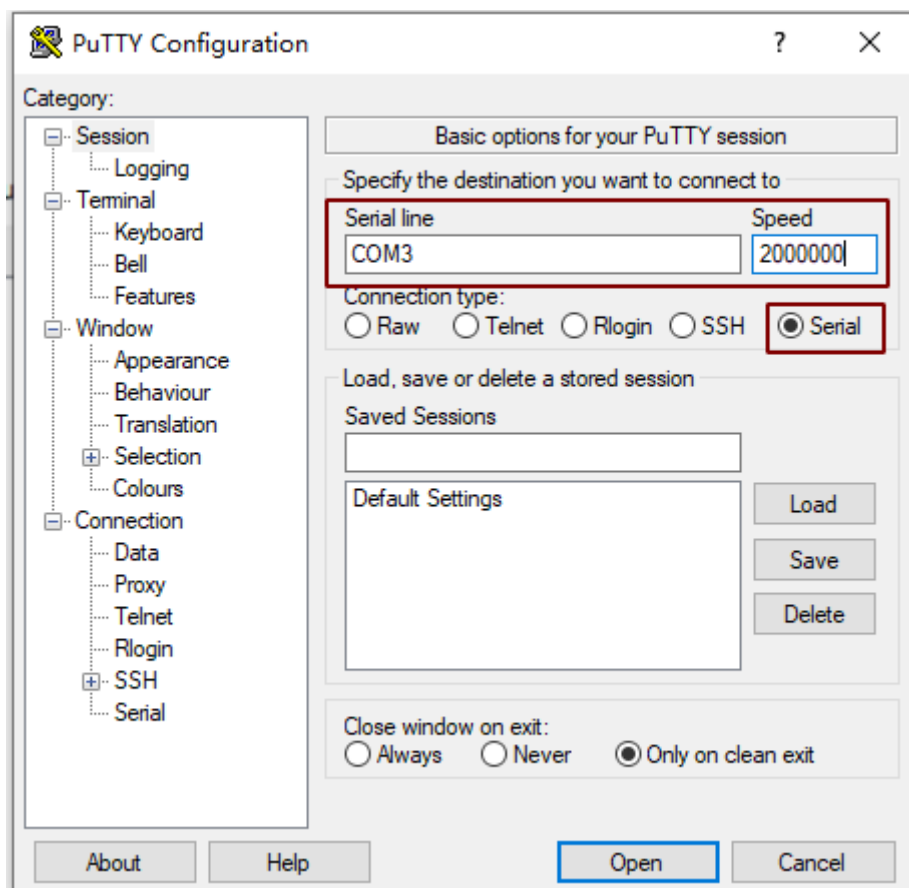


图 2.4: 串口工具

1. PC 与路由器通过有线连接。

测试评估一：wifi ping + ble adv

重启板子，bl602 作为 client，PC 作为 server，APP 以 nRF Master Control Panel / nRF Connect 为例

1. router ssid: bl_test_081, passwd: 12345678
2. 在串口中运行 wifi 相关命令：

```
#stack_wifi
```

```
#wifi_sta_connect bl_test_081 12345678 (连接成功后会获取 IP 地址)
```

```
[lwip] netif status callback
IP: 192.168.8.193
MK: 255.255.255.0
GW: 192.168.8.1
[WF][SM] Exiting wifiConnected_ipObtaining state
[WF][SM] IP GOT IP:192.168.8.193, MASK: 255.255.255.0, Gateway: 192.168.8.1, dns1: 192.16
8.8.1, dns2: 0.0.0.0
[WF][SM] State Action ###wifiConnected_ipObtaining### --->>> ###wifiConnected_IPOK###
[WF][SM] Entering wifiConnected_IPOK state
[APP] [EVT] GOT IP 24583
[SYS] Memory left is 132664 Bytes
```

图 4.1: 模块成功连接 WiFi

3. 在 PC 的 cmd 界面运行命令：\$ping 192.168.81.103 -t (默认 1s ping 一次,192.168.81.103 是设备端获取的 IP 地址)

```
C:\Users\paul>
C:\Users\paul>ping 192.168.81.103 -t

正在 Ping 192.168.81.103 具有 32 字节的数据:
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=4ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=2ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=5ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=2ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=2ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=6ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=5ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=3ms TTL=255
来自 192.168.81.103 的回复: 字节=32 时间=26ms TTL=255
```

图 4.2: 模块开启 ping

4. 在串口中运行 ble 相关命令:

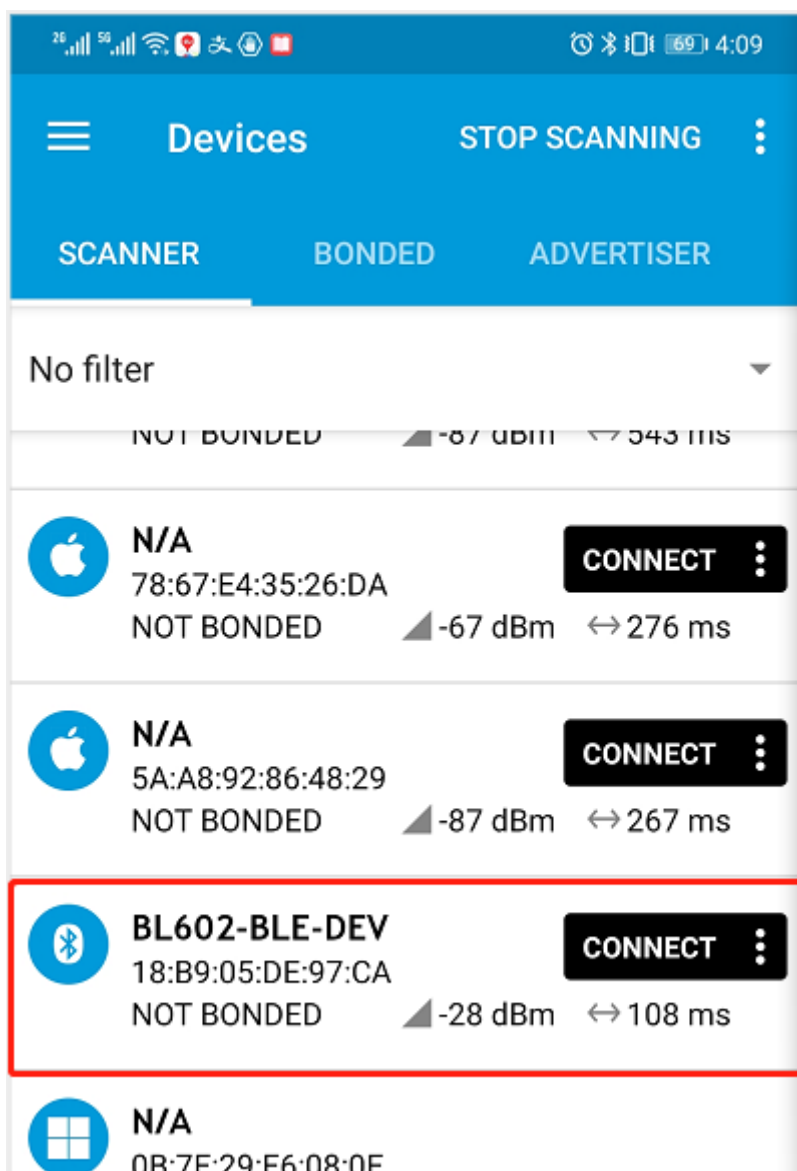
```
#stack_ble
```

```
#ble_init
```

```
#ble_start_adv 0 0 0xa0 0xa0 (发起 adv,interval 为 100ms)
```

```
Advertising started
# [ 4064829][0][32mINFO 0m: b1_sec.c: 126] random number is 7ed9292f
[ 4065655][0][32mINFO 0m: b1_sec.c: 126] random number is 8730dad4
[ 4066496][0][32mINFO 0m: b1_sec.c: 126] random number is 1d81e61b
[ 4067362][0][32mINFO 0m: b1_sec.c: 126] random number is b00a72db
[ 4068204][0][32mINFO 0m: b1_sec.c: 126] random number is 9b23dbc6
[ 4069035][0][32mINFO 0m: b1_sec.c: 126] random number is 017665d8
[ 4069887][0][32mINFO 0m: b1_sec.c: 126] random number is 0e7aabc0
proc_hellow_entry: RISC-V rv32imafc
[ 4070710][0][32mINFO 0m: b1_sec.c: 126] random number is a53d5ec9
```

5. 手机端打开 APP, 查看是否扫描到设备 BL602-BLE-DEV, 如果扫描到说明测试成功:



测试评估二：wifi ping + ble connect(1s 发送 5 次数据包，长度为 23byte)

重启板子，bl602 作为 client，PC 作为 server，APP 以 BLE 调试助手为例。

1-3. 步骤如同测试评估一中所示：

4. 在串口中运行 ble 相关命令：

```
#stack_ble
```

```
#ble_init
```

```
#ble_start_adv 0 0 0xa0 0xa0 (发起 adv,interval 为 100ms)
```

5. 打开 APP, 扫描到 BL602-BLE-DEV 并连接：

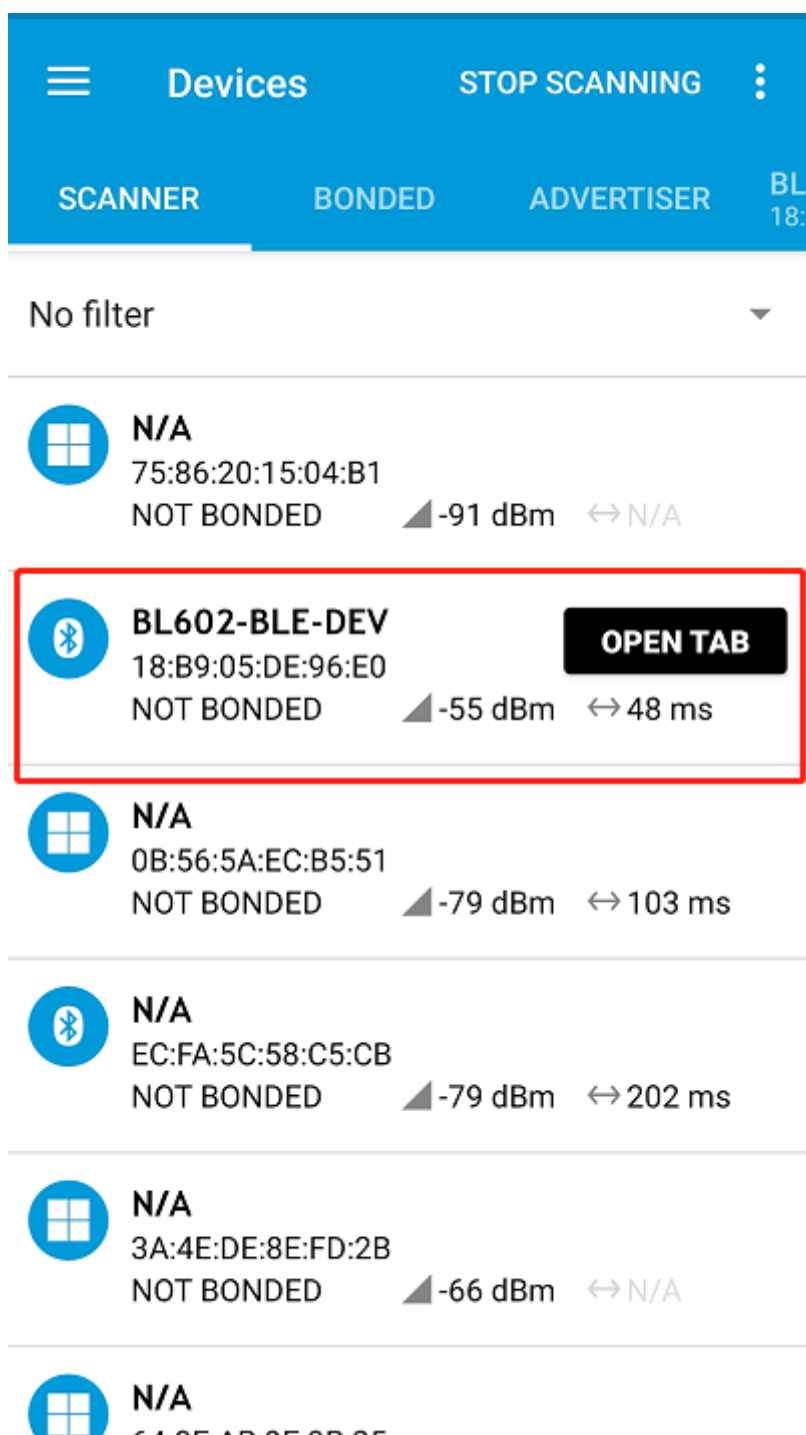


图 5.1: APP 扫描到 BLE

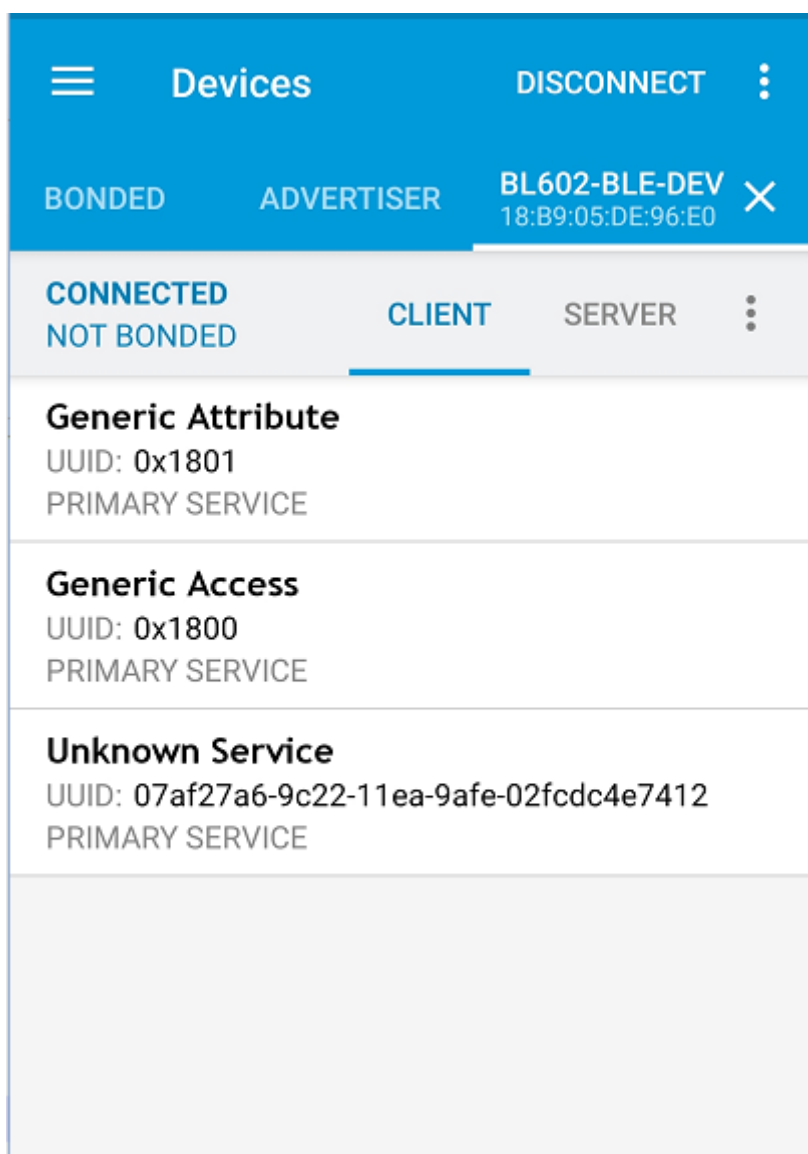


图 5.2: BLE 连接成功

6. 在串口中运行 ble 相关命令:



```
#ble_conn_update 0x6 0x6 0x0 0x1f4 (连接 interval 为 7.5ms)
```

7. 在串口中查看连接参数已更新:


```
proc_hellow_entry: RISC-V rv32imafc
pa 37447893d, ce trk 7.09, action: capcode 46 -> 45
ble_conn_update 0x6 0x6 0x0 0x1f4
[btsnoop]:opcode =[0x2013],len =[0xe],data=[0000060006000000f40100000000]
[btsnoop]:Stop
[btsnoop]:pkt_type =[0x3],len =[0x4],data=[00011320]
[btsnoop]:Stop
conn update initiated

# [btsnoop]:pkt_type =[0x4],len =[0xa],data=[03000000006000000f401]
[btsnoop]:Stop
LE conn param updated: int 0x0006 lat 0 to 500
pa 42670755d, ce trk 6.74, action: capcode 45 -> 44
```

8. 在 APP 中找到服务特性，并且写入相应的数据：

 **BL602-BLE-DEV** 断开连接 

Services

Log

Generic Attribute

> UUID: 00001801-0000-1000-8000-00805f9b34fb

PRIMARY SERVICE

Generic Access

> UUID: 00001800-0000-1000-8000-00805f9b34fb

PRIMARY SERVICE

Unknown Service

✓ UUID: 07af27a6-9c22-11ea-9afe-02fcdc4e7412

PRIMARY SERVICE

Unknown Characteristic


UUID: 07af27a7-9c22-11ea-9afe-02fcdc4e7412

Properties:NOTIFY

Descriptors:

Client Characteristic Configuration


UUID: 00002902-0000-1000-8000-00805f9b34fb

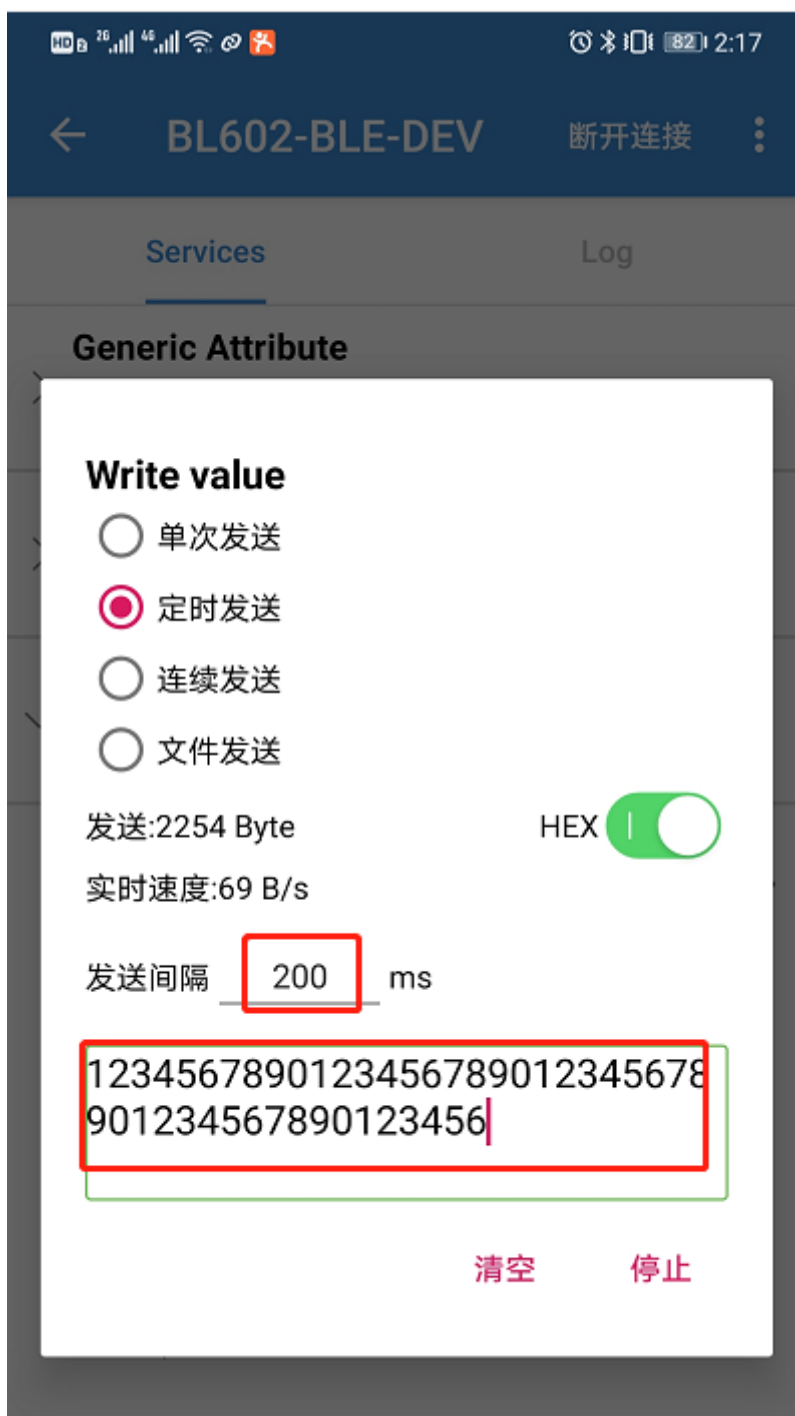


Unknown Characteristic

UUID: 07af27a8-9c22-11ea-9afe-02fcdc4e7412

Properties:





9. 查看 wifi 与 ble 是否稳定连接

测试评估三: wifi running iperf + ble connect(1s 发送 5 次数据包, 长度为 23byte)

重启板子, bl602 作为 client, PC 作为 server, APP 以 BLE 调试助手为例。

1-2. 步骤如同测试评估一中所示:

3. 在串口中运行命令: \$ipc 192.168.81.101 (192.168.81.101 是 PC 的 IP 地址)

```

MK: 255.255.255.0
GW: 192.168.81.1
[WF][SM] Exiting wifiConnected_ipObtaining state
[WF][SM] IP GOT IP:192.168.81.105, MASK: 255.255.255.0, Gatewa
[WF][SM] State Action ###wifiConnected_ipObtaining### --->>> #
[WF][SM] Entering wifiConnected_IPOK state
[APP] [EVT] GOT IP 23503
[SYS] Memory left is 85632 Bytes
proc_hellow_entry: RISC-V rv32ima1c
ipc 192.168.81.101

# Connect to iperf server successful!
5.0898(5.0898 5.0898 5.0898) Mbps!
push back

```

图 6.1: 模块开启 ipc

4. 在 PC 的 cmd 界面运行命令: \$iperf.exe -s -u -i 1

```
D:\Paul\tools\iperf>iperf -s -i1
```

```
Server listening on TCP port 5001  
TCP window size: 64.0 KByte (default)
```

```
-----  
[380] local 192.168.81.101 port 5001 connected with 192.168.81.103 port 51649
```

[ID]	Interval	Transfer	Bandwidth
[380]	0.0- 1.0 sec	442 KBytes	3.62 Mbits/sec
[380]	1.0- 2.0 sec	223 KBytes	1.83 Mbits/sec
[380]	2.0- 3.0 sec	523 KBytes	4.28 Mbits/sec
[380]	3.0- 4.0 sec	538 KBytes	4.41 Mbits/sec
[380]	4.0- 5.0 sec	548 KBytes	4.49 Mbits/sec
[380]	5.0- 6.0 sec	574 KBytes	4.70 Mbits/sec
[380]	6.0- 7.0 sec	559 KBytes	4.58 Mbits/sec
[380]	7.0- 8.0 sec	559 KBytes	4.58 Mbits/sec
[380]	8.0- 9.0 sec	524 KBytes	4.29 Mbits/sec
[380]	9.0-10.0 sec	553 KBytes	4.53 Mbits/sec
[380]	10.0-11.0 sec	533 KBytes	4.37 Mbits/sec
[380]	11.0-12.0 sec	5.08 KBytes	41.6 Kbits/sec
[380]	12.0-13.0 sec	452 KBytes	3.70 Mbits/sec
[380]	13.0-14.0 sec	498 KBytes	4.08 Mbits/sec
[380]	14.0-15.0 sec	533 KBytes	4.37 Mbits/sec
[380]	15.0-16.0 sec	574 KBytes	4.71 Mbits/sec
[380]	16.0-17.0 sec	147 KBytes	1.20 Mbits/sec
[380]	17.0-18.0 sec	564 KBytes	4.69 Mbits/sec

图 6.2: PC 端 Iperf 开启 sever 模式

5. 在串口中运行 ble 相关命令:

```
#stack_ble
```

```
#ble_init
```

```
#ble_start_adv 0 0 0xa0 0xa0 (发起 adv,interval 为 100ms)
```

```
# pa 300603924d, ce trk 6.04, action: capcode 50 -> 49
2.5021(2.2100 26.2022 4.8002) Mbps!
ble_start_adv 0 0 0xa0 0xa0
adv_type 0x0
tmp 0x0
interval min 0xa0
interval max 0xa0
Advertising started

# [ 305646][ ][32mINFO ][0m: b1_sec.c: 126] random number is 983611b2
[ 306485][ ][32mINFO ][0m: b1_sec.c: 126] random number is 01be97be
pa 304289910d, ce trk 5.78, action: capcode 49 -> 48
[ 307336][ ][32mINFO ][0m: b1_sec.c: 126] random number is 7061f56c
[ 308171][ ][32mINFO ][0m: b1_sec.c: 126] random number is 58d8b249
[ 309010][ ][32mINFO ][0m: b1_sec.c: 126] random number is c7c4d058
4.5207(2.2109 22.0388 4.8902) Mbps!
[ 309852][ ][32mINFO ][0m: b1_sec.c: 126] random number is 67a500b0
proc_hellow_entry: RISC-V rv32imafc
[ 310689][ ][32mINFO ][0m: b1_sec.c: 126] random number is 507c4047
pa 307669003d, ce trk 5.43, action: capcode 48 -> 47
[ 311526][ ][32mINFO ][0m: b1_sec.c: 126] random number is 984f5937
[ 312368][ ][32mINFO ][0m: b1_sec.c: 126] random number is 312a29e0
[ 313189][ ][32mINFO ][0m: b1_sec.c: 126] random number is 3f238cb7
[ 314033][ ][32mINFO ][0m: b1_sec.c: 126] random number is 9be0b04b
push back
```

align center

Ble 开启 ADV

6. 手机打开 APP，扫描到设备 BL602-BLE-DEV 并连接:

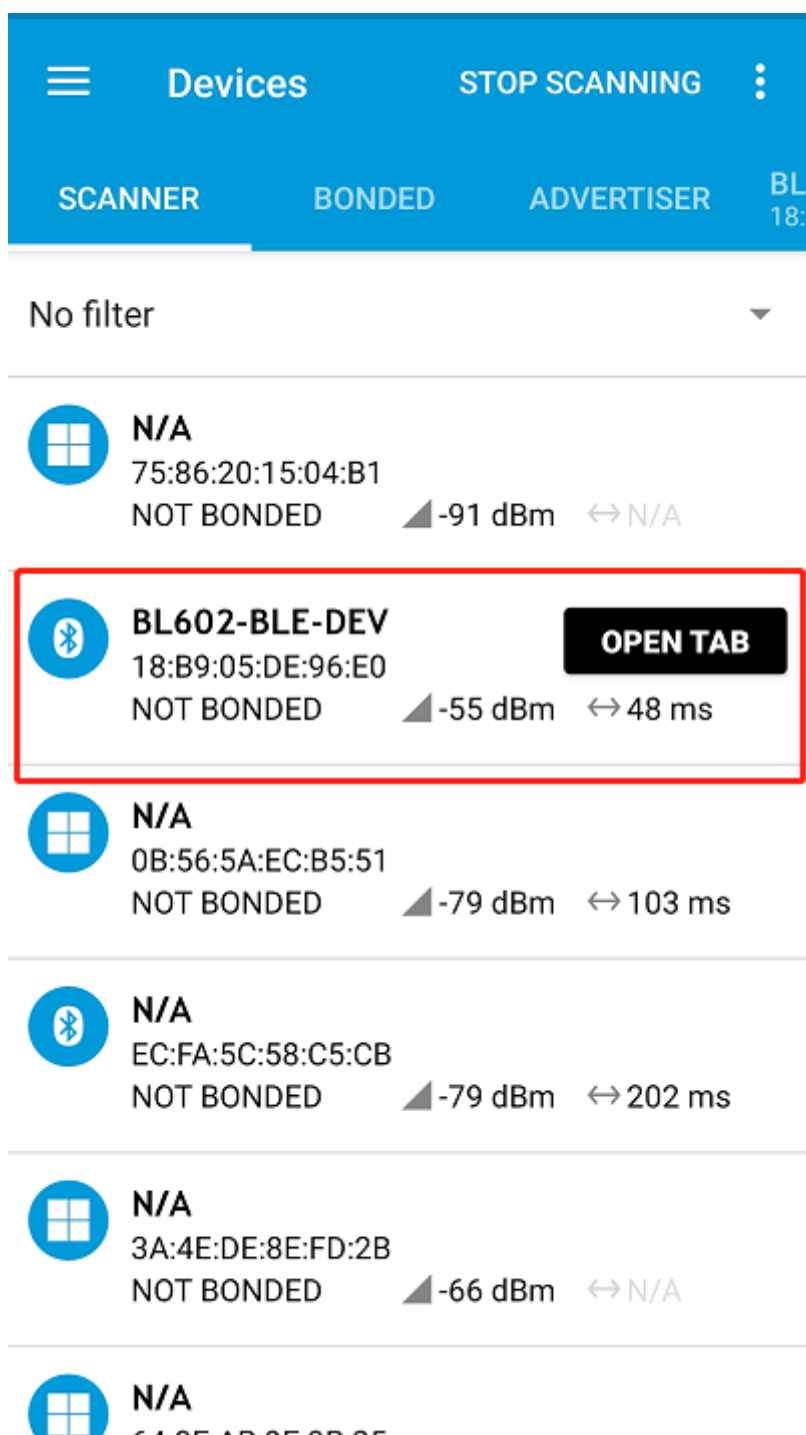


图 6.3: APP 扫描到 BLE

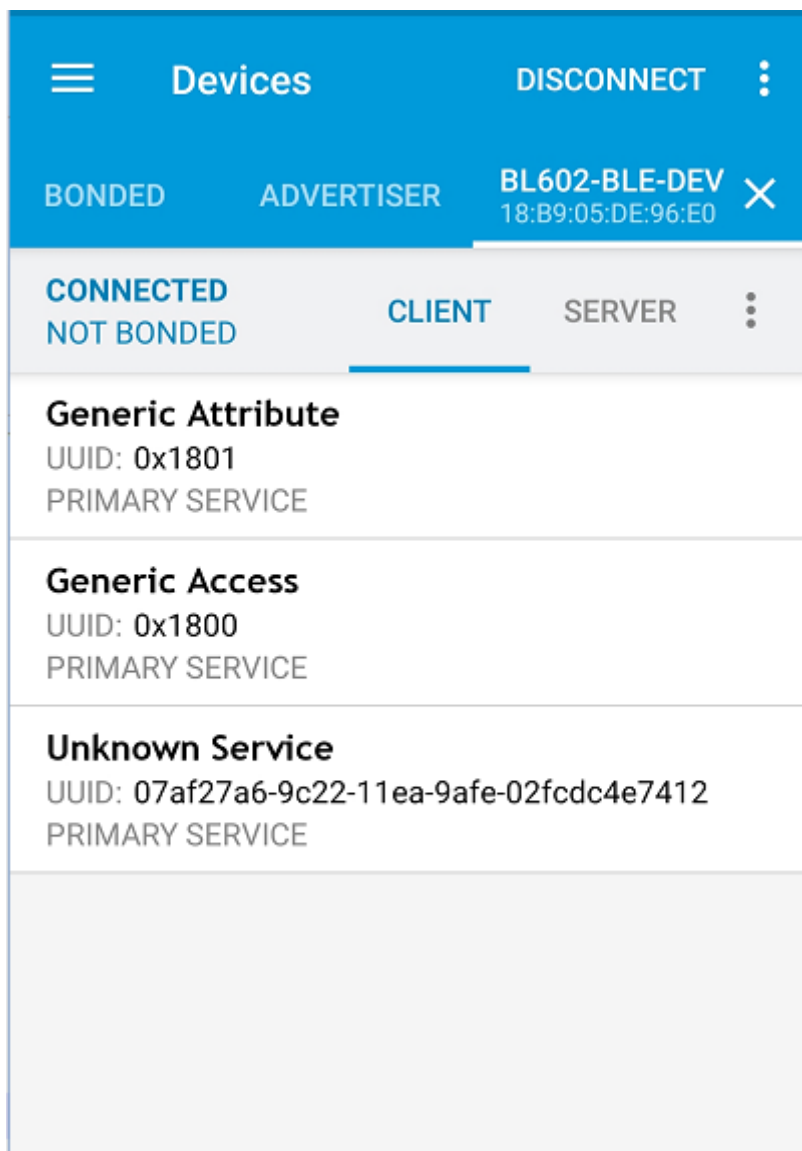


图 6.4: BLE 连接成功



7. 连接成功后，在串口中运行 ble 连接参数更新命令：

```
#ble_conn_update 0x28 0x28 0x0 0x1f4 (连接 interval 为 50ms)
```



```
2.6620(2.2109 5.2423 4.8902) Mbps!  
ble_conn_update 0x28 0x28 0x0 0x1f4  
conn update initiated  
# LE conn param updated: int 0x0028 lat 0 to 500  
3.9912(2.2109 5.2252 4.8902) Mbps!  
proc_hellow_entry: RISC-V rv32imafc  
push back  
push back  
push back  
push back  
push back
```

8. 在 APP 中找到服务特性，并且写入相应的数据：


 **BL602-BLE-DEV** 断开连接 


Services Log

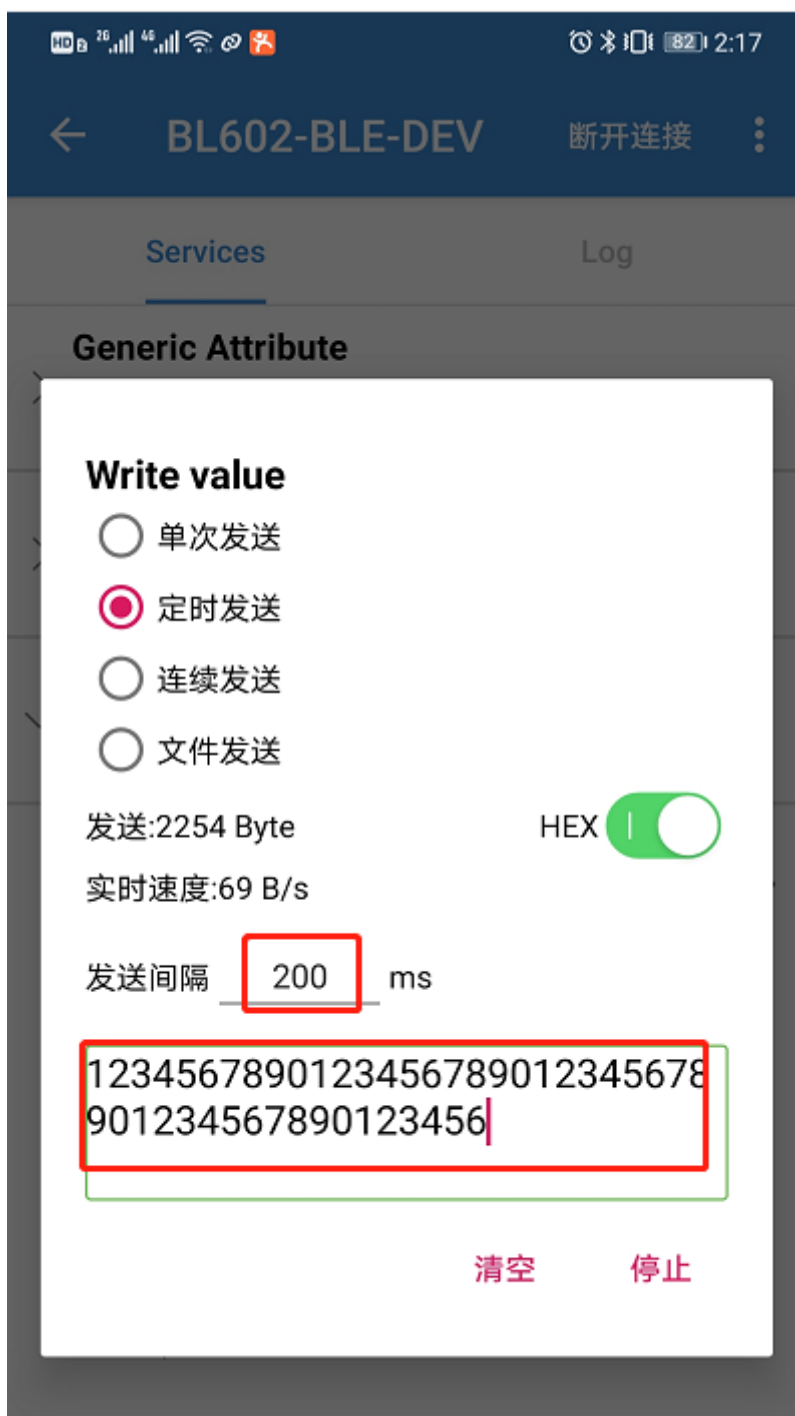
Generic Attribute
> UUID: 00001801-0000-1000-8000-00805f9b34fb
PRIMARY SERVICE

Generic Access
> UUID: 00001800-0000-1000-8000-00805f9b34fb
PRIMARY SERVICE

Unknown Service
✓ UUID: 07af27a6-9c22-11ea-9afe-02fcdc4e7412
PRIMARY SERVICE

Unknown Characteristic 
UUID: 07af27a7-9c22-11ea-9afe-02fcdc4e7412
Properties:NOTIFY
Descriptors:
Client Characteristic Configuration
UUID: 00002902-0000-1000-8000-00805f9b34fb

Unknown Characteristic 
UUID: 07af27a8-9c22-11ea-9afe-02fcdc4e7412
Properties:



9. 查看 ble 是否稳定连接，iperf 的速率是否正常