EC2X&AG35-QuecOpen

NF3303 BLE User Guide

LTE Module Series

Rev. EC2X&AG35-QuecOpen_NF3303_BLE_User_Guide_V1.0

Date: 2018-03-21

Status: Temporary

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://quectel.com/support/sales.htm

For technical support, or to report documentation errors, please visit:
http://quectel.com/support/technical.htm
Or email to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2018-03-21	Quinn ZHAO	Initial

Contents

Table Index

1 Introduction

The document is designed to help customers to quickly use the Bluetooth function of NF3303 WIFI/BT module on Quectel EC2X&AG35-QuecOpen platform to achieve business needs.

2 Hardware Interface

NF3303 Bluetooth module communicates with EC2X or AG35 via hardware-flow-controlled four-wire serial port. In addition EC2X and AG35 need reserve one GPIO as power control pin for NF3303. If sleep and wake up function is required, also need to reserve two pins more, one is input interrupting for EC2X or AG35, another is the general GPIO output pin.

2.1. The Pin Relationship Between EC2X and NF3303

Table 1: The pin relationship between EC2X and NF3303

EC2X		NF3303	
Pin No.	Definition	Pin No.	Definition
62	GPIO_75(Output)	40	BT_DEVWAKE (Input)
1	GPIO_25 (Input)	39	BT_HOST_WAKE_U(Output)
139	PMU_GPIO1019(Output)	24	BT_EN (Input)
65	UART_RTS_BLSP6/GPIO_22(Input)	10	BT_UART_RTS_N (Output)
68	UART _RXD_BLSP6/ GPIO_21(Input)	7	BT_UART_TXD (Output)
67	UART _TXD_BLSP6/ GPIO_20(Output)	6	BT_UART_RXD (Input)
64	UART _CTS_BLSP6/ GPIO_23(Output)	11	BT_UART_CTS_N (Input)

2.2. The Pin Relationship between AG35 and NF3303

Table 2: The pin relationship between EC2X and NF3303

AG35		NF3303	
Pin No.	Definition	Pin No.	Definition

61	GPIO_75(Output)	40	BT_DEVWAKE (Input)
144	GPIO_25 (Input)	39	BT_HOST_WAKE_UP(Output)
3	PMU_GPIO1019(Output)	24	BT_EN (Input)
79	UART_RTS_BLSP6/GPIO_22(Input)	10	BT_UART_RTS_N (Output)
78	UART _RXD_BLSP6/ GPIO_21(Input)	7	BT_UART_TXD (Output)
77	UART _TXD_BLSP6/ GPIO_20(Output)	6	BT_UART_RXD (Input)
80	UART _CTS_BLSP6/ GPIO_23(Output)	11	BT_UART_CTS_N (Input)

3 Driver Adaptation

According to Chapter 2, need adapt serial port driver, BT_EN pin is common GPIO output driver, BT_HOST_WAKE_UP pin interrupt input drive, BT_DEVWAKE ordinary GPIO output driver.

3.1. Serial Port Driver Adaptation

There are 4 serial ports on EC2X-QuecOpen Module: main serial port, debug serial port, serial port 1 and serial port 2. Serial port 1 and serial port 2 with same function, both support RTS/CTS, can be as a peripheral communication serial port. Which, RTS/CTS of serial port 1 is multiplexed with I2C, serial port 2 is multiplexed with SPI. The recommendation is choosing main serial port or serial port 2 as BLE communication serial port.

Please refer to ***Quectel_EC20 R2.1_QuecOpen_GPIO_Assignment_Speadsheet***.

3.2. BT_EN Pin Driver Adaptation

QuecOpen SDK already support most pins in EC2X&AG35, to realize output function for general GPIO through application layer configuration. Please according to actual hardware, refer to Quectel_EC20 R2.1_QuecOpen_GPIO_Assignment_Speadsheet to determine GIOI. EC2X/G35 is according to the power-on time sequence requirement of NF3303 module to pull up or pull low BT_EN pin. The specific process is: pull low BT_EN -> keep 200ms -> pull up BT_EN. Please according to actual hardware, modify the functions **ql_bt_en_pin_init** and **ql_bt_module_enable** which in the file **ql_nf3303_ble_common.c** in the path of **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source.** Example is shown as following.

```
/*Operate BT_EN to enable BT Module*/
void ql_bt_module_enable()
{
    /*Pull Up PM_ENABLE to Power the Module*/
    system("echo 1 > /sys/kernel/debug/regulator/rome_vreg/enable");
    /*Reset BT Module*/
    ioctl(bt_power_fd, 0);     //pull Down BT_EN
    usleep(200000);
    ioctl(bt_power_fd, 1);     //pull Up BT_EN
}
```

3.3. BT_HOST_WAKE Pin Driver Adaptation

QuecOpen SDK already support most pins in EC2X&AG35 to realize interruption function through application layer configuration. Please refer to **Quectel_EC20 R2.1_QuecOpen_GPIO_Assignment_Speadsheet** to determine GPIO. The modification path is file **ql_NF3303_ble_sleep.c** in **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source**, macro definition **BT_HostWakePin** is the actual hardware used pin. Example is shown as following.

```
#if defined(__QUECTEL_PROJECT_AG35C__) || defined(__QUECTEL_PROJECT_AG35CE__)
static Enum_PinName BT_HostWakePin = PINNAME_GPIO3;
static Enum_PinName BT_DevWakePin = PINNAME_GPIO6;
#else
static Enum_PinName BT_HostWakePin = PINNAME_GPIO1;
static Enum_PinName BT_DevWakePin = PINNAME_GPIO6;
#endif
```

3.4. BT_DEVWAKE Pin Driver Adaptation

QuecOpen SDK already support most pins in EC2X&AG35, to realize output function for general GPIO through application layer configuration. Please according to actual hardware, refer to **Quectel_EC20 R2.1_QuecOpen_GPIO_Assignment_Speadsheet** to determine GPOI. The modification path is file **ql_NF3303_ble_sleep.c** in **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source**, macro definition **BT_DevWakePin** is the actual hardware used pin. Example is shown as following.

```
#if defined(__QUECTEL_PROJECT_AG35C__) || defined(__QUECTEL_PROJECT_AG35CE__)
static Enum_PinName BT_HostWakePin = PINNAME_GPIO3;
static Enum_PinName BT_DevWakePin = PINNAME_GPIO6;
#else
static Enum_PinName BT_HostWakePin = PINNAME_GPIO1;
static Enum_PinName BT_DevWakePin = PINNAME_GPIO6;
#endif
```

4 Sleep Wake-up Function Introduction

The NF3303 module uses two pins to implement the sleep wake-up function. BT_HOSTWAKE is an interruption input pin for the EC20&AG35 and an output pin for the NF3303 Bluetooth module. BT_DEVWAKE is an ordinary output pin for the EC20&AG35 and an input pin for the NF3303 module.

4.1. BT_HOST_WAKE_UP Pin Function Introduction

When there is no data interaction, BT_HOST_WAKE_UP is low-level. After the mobile phone sends out a read/write data interaction request, NF3303 send data to EC2X\AG35, will pull up BT_HOST_WAKE_UP pin, i.e. rising edge interruption occurs to the EC2X or AG35. After the data is sent, pull low the BT_HOST_WAKE pin to generate a falling edge interruption, that is, a falling edge interruption is sent to EC2X or AG35.

4.2. BT_DEVWAKE Pin Function Introduction

When EC2X\AG35 send data to NF3303, Bluetooth protocol stack will generate callback function, according to this callback function EC2X\AG35 pull up BT_DEV_WAKE pin. After data is sent, Bluetooth protocol stack will generate another callback function, EC2X\AG35 pull low BT_DEV_WAKE according to this callback function.

4.3. Introduction EC2X&AG35 RTS Pin Introduction

The judgement of Bluetooth module whether send data to EC2X/AG35 is according to its CTS, i.e. whether RTS of EC2X and AG35 is low-lever. If it’s low-level, will send data to EC2X /AG35. Therefore, when EC2X/AG35 in sleep mode, please maintain the RTS output is high-level in case the data lose.

4.4. Code Ideas of Sleep Wake-up Function

4.4.1. Sleep Wake-up Initialization Phase

Register interruption service function of BT_HOST_WAKE_UP. Create the wakelock which sleep and wake-up function require. Pull up BT_DEV_WAKE pin.

4.4.2. Code Logic of Receiving rising Edge Interruption.

First to check whether wakelock is lock, if yes, don’t need to lock, if no, lock it.

4.4.3. Code Logic of Receiving Falling Edge Interruption.

A cycle timer with an interval of 5 s is started at system startup. When this cycle timer timeout, first to check whether wakelock is unlock, if yes, don’t need to unlock, if no, unlock it.

4.4.4. Code Logic of BT_DEV_WAKE Pin

If receiving Bluetooth protocol stack callback function **NFBT_WARNING_BT_ALLOW_SLEEP,** will pull low BT_DEV_WAKE. If receiving Bluetooth protocol stack callback function **NFBT_WARNING_BT_WAKE,** will pull up BT_DEV_WAKE.

5 Introduction of Bluegate Protocol Stack

Bluegate protocol stack provided by NF3303 module manufacturer, to simplify the design of Bluetooth function and improve the development efficiency. Developers can directly operate Bluegate to implement most Bluetooth features. As a middleware, Bluegate implements a variety of commonly used protocol stacks and profiles, providing a set of application programming interfaces (API) for developers to call. Please according actual application requirements and refer to ***nFore Bluegate Programming Guide.pdf.***

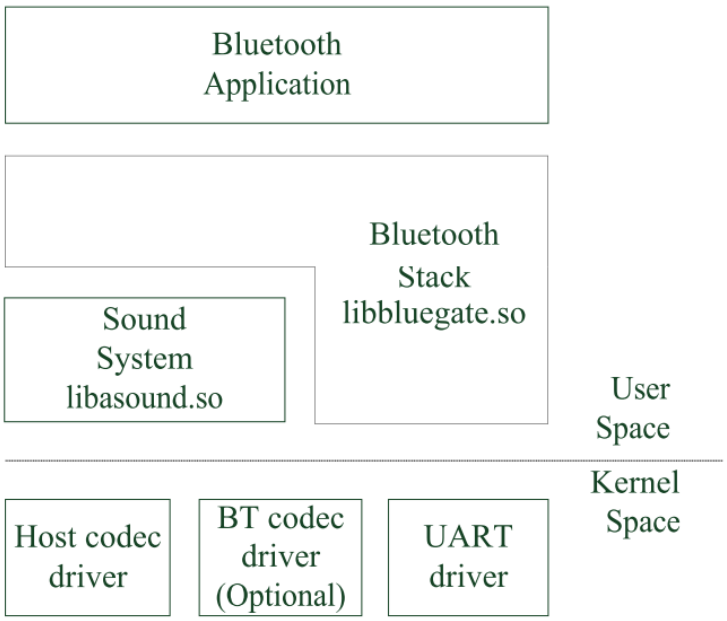


Figure 1: Bluegate Frames Diagram

5.1. Configure Communiation Serial Port

Under the premise that the serial port driver has been configured correctly, import the file **bluegate_hw.conf** which in the path **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/lib/firmware** into /etc/bluetooth/ directory. Bluegate_hw.conf file information is shown as below.

```
# UART device port where Bluetooth controller is attached
UartPort = /dev/ttyHSL2

# Firmware patch file location
FwPatchFilePath = /etc/bluetooth/
FwPatchFileName = BCM4339_003.001.009.0119.0000.hcd
```

The text information of the modified configuration file is as follow.

```
# UART device port where Bluetooth controller is attached
UartPort = /dev/ttyHSL2

# Firmware patch file location
FwPatchFilePath = /etc/bluetooth/
FwPatchFileName = BCM4339_003.001.009.0119.0000.hcd
```

UartPort must be consistent with the device information of the actual serial port. FwPatchFilePath should be same with the path of actual file **BCM4339_003.001.009.0119.0000.hcd.**

5.2. Importing Chip Firmware and Bluetooth Protocol Stack.

5.2.1. Importing Chip Firmware

After EC2X\AG35 establishing communication with Bluetooth module, will transmit data to Bluetooth module based on HCI command. Thus, the chip firmware needs to be pre-placed in the file system. Please use ADB or other tools push file **BCM4339_003.001.009.0119.0000.hcd** in **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/lib/firmware** to /etc/bluetooth/ directory of EC20/AG35. The reference command is shown as follows.

```
$ adb push BCM4339_003.001.009.0119.0000.hcd /etc/bluetooth/
```

5.2.2. Importing Bluetooth Protocol Stack

The Bluetooth protocol stack (libbluegate.so) is provided as a dynamic library, and running client applications program depends on this library file. Therefore, the Bluetooth protocol stack needs to be pre-placed in the file system. Please use ADB or other tools push files in **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/lib/** to libbluegate.so directory of EC20/AG35. The reference command is shown as follows.

```
$ adb push libbluegate.so /lib/
```

5.3. Building and Compliing

Unzip the SDK archive and enter the ql-ol-sdk folder to compile the routine. The reference command is as follows:

```
$ source ql-ol-crosstool/ql-ol-crosstool-env-init
$ cd ql-ol-extsdk/example/bt/nf3303/
$ make clean
$ make
```

In the SDK, an executable program named "example_nf3303_ble_server" is created under the path "ql-ol-

sdk/ql-ol-extsdk/example/bt/nf3303". This executable program can be pushed to EC2X&AG35 file system. The reference command is shown as follows.

```
$ adb push example_nf3303_ble_server /data/
$ adb shell chmod a+x /data/example_nf3303_ble_server
```

Enter Linux Shell terminal, enter below command, start BLE Server:

```
$ cd /data/
$ ./example_nf3303_ble_server
```

6 Code Use Guidance

6.1. Modify Broadcast Message

The broadcast information is modified by the function "NFBT_GATT_SetAdvertisingData" in the file "ql_nf3303_ble_server.c" whose path is "ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source". Modifiable parameters are the broadcast device name, transmit power, service UUID, and broadcast interval. The following describes the modification method of each modification item.

6.1.1. Modify the Broadcast Device Name.

The broadcast device name is modified by the macro definition "LOCAL_NAME" in the file “ql_nf3303_ble_server.c" whose path is "ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source".

6.1.2. Modify Broadcast Service UUID

The broadcast Service UUID is modified by modifying the local variables " gatt_base_uuid" of main function in the file “ql_nf3303_ble_server.c " whose path is “ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source”.

6.1.3. Modify Broadcast Interval

The modifiable time interval value is as follows, and the modification parameter is the corresponding macro definition. Parameters are shown as below.

```
typedef enum
{
    GATT_ADV_SPEED_SLOW = 0,
    GATT_ADV_SPEED_NORMAL,
    GATT_ADV_SPEED_FAST,
}GATT_ADV_SPEED;
```

6.2. Modify Configuration Parameters of Service, Characteristic and Descriptor

6.2.1. Configure Maximum Service Number Parameters

The maximum service number parameter is defined by the macro definition **BLE_TOTAL_SERVICE_NUM** in the header file **ql_nf3303_bt_callback.h** of the modification path **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/include**. This macro definition can be increased or decreased according to needs, and the default value is 8.

6.2.2. Configure Maximun Service allowed Characteristic and Descriptor Number Parameters

The maximum total number parameter of characteristic and Descriptor is defined by the macro definition **BT_MAX_ATTR_NUM** in the header file **ql_nf3303_bt_callback.h** of the path **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/include**. Assuming that there are multiple Service, the sum of the characteristic and Descriptor of one service is 8, and the sum of other Service characters and Descriptors is less than 8, this value should be set to 8.

6.2.3. Modify Service, Characteristic and Descriptor Configured Parameters

Below codes is in file **ql_nf3303_ble_server.c** of path **ql-ol-sdk/ql-ol-extsdk/example/bt/nf3303/source.** Take characteristic 0 of Service 0 as an example to explain how to modify. Please modify according to the annotation below and parameters without annotation need not be modified. The places that need to be modified is marked in red.

```
//Service 0 Definition
//Please fill Service UUID in the line below
ql_gatt_demo.ql_gatt_service_s[0].service_uuid= QUECTEL_SERVICE0_UUID
//Please fill characteristic and Descriptor total number of Service in the line below
ql_gatt_demo.ql_gatt_service_s[0].total_attr_num= BLE_SERVICE0_ATTR_NUM;
ql_gatt_demo.ql_gatt_service_s[0].is_srv_valid = 0x1;
ql_gatt_demo.ql_gatt_service_s[0].status = SRV_REG;

// Service 0 characteristic 0 Definition
//Please fill ATTRTYPE_CHARACTERISTIC or ATTRTYPE_DESCRIPTOR in the line below
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].type = ATTRTYPE_CHARACTERISTIC
//Please fill UUID of characteristic or Descriptor in the line below
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].uuid = DEVINFO_MANUFACTURER_NAME_UUID
//Please fill Permission of characteristic or Descriptor in the line below
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].permission = GATT_PERMIT_READ
```

```
//Please fill Property of characteristic or Descriptor in the line below
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].property=
GATT_PROP_READ|GATT_PROP_NOTIFY;
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].is_attr_valid = 0x1;
//Please fill data bytes number of characteristic or Descriptor in the line below
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].data_bytes = 100;
tmp_len = ql_gatt_demo.ql_gatt_service_s[0].attributes[0].data_bytes;
ql_gatt_demo.ql_gatt_service_s[0].attributes[0].data = (char *)calloc(tmp_len, sizeof(char));
if(NULL == ql_gatt_demo.ql_gatt_service_s[0].attributes[0].data)
ERR("Calloc Failed:\n");
//Please fill initial data for characteristic or Descriptor in the line below
tmp_len = sprintf(ql_gatt_demo.ql_gatt_service_s[0].attributes[0].data, "%s", "Quectel NF3303 BT
Module");
if (tmp_len < 0) {
ERR("Sprintf Failed:\n");
}
```

7 Testing and Verifying

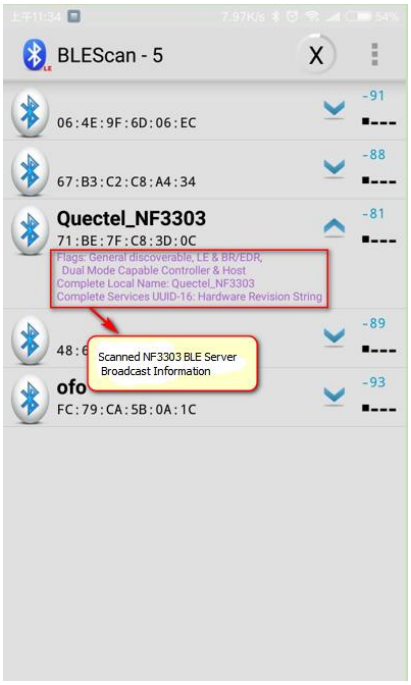
BLE is turned off on mobile phone by default, therefore mobile phone terminals need to download APP for function verification. If it's Android platform, please download BLE Deng APP. If it's iphone platform, please download LightBlue APP. This document is based on Android BLE Deng APP.

7.1. Basic Function Testing

This chapter mainly introduces the scan, connection and data interaction testing. Please turn on Bluetooth function and BLE Deng APP.

7.1.1. Scan Testing

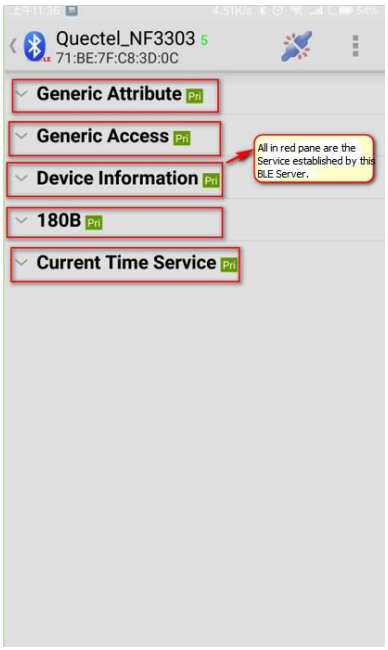
Turn on BLE Deng App and it will automatically scans surrounding BLE device.



As shown above, mobile phone scanned cm256sm Bluetooth module successfully.

7.1.2. Connection Testing

Please click 'Quectel NF3303', connect mobile phone with NF3303. After the connection is established, BLE Deng App will does 'Service Discovery', then scans all Service on cm256sm, shown as following.



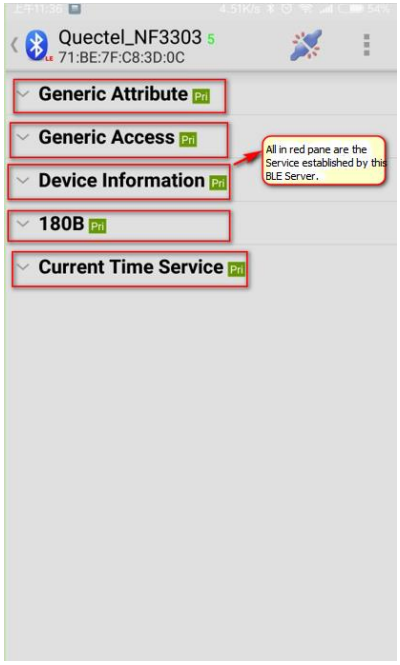
7.1.3. Data Interaction Testing

7.1.3.1. Check Characteristic Contained by Service

Select a Service and click to view the characteristic contained by Service. Select "Device Information" Service here. As shown below.

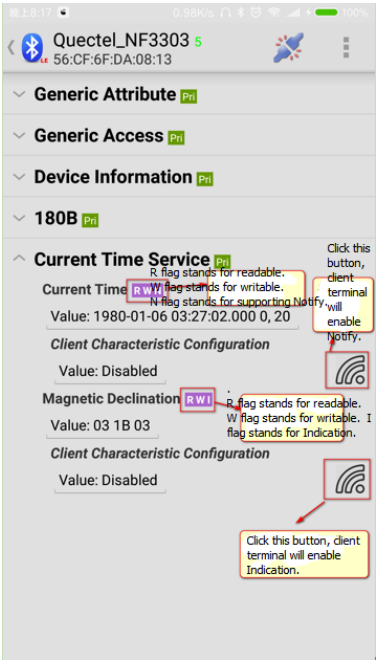
7.1.3.2. Read and Write Testing

As shown below, Characteristic FirmwareRevisionString can be read and wrote. Please modify value to BlueGate v8.8.8.8. Results as shown below.



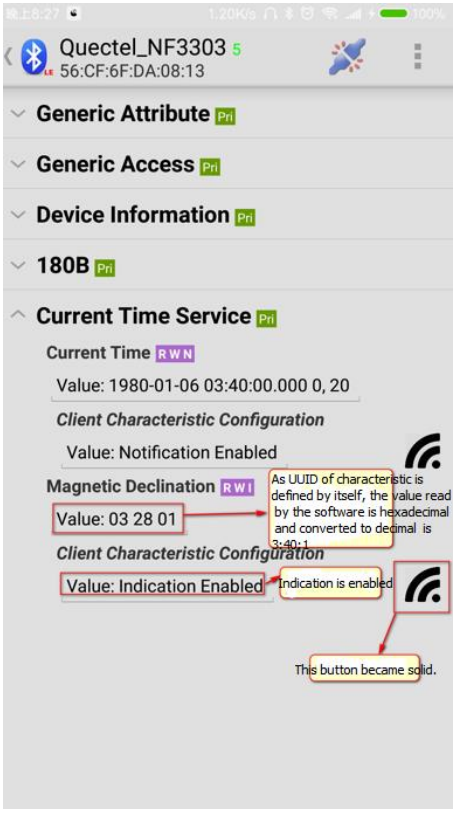
7.1.3.3. Notification and Indication Function Testing

Select “Current Time Service”, and click it to check characteristic contained by Service



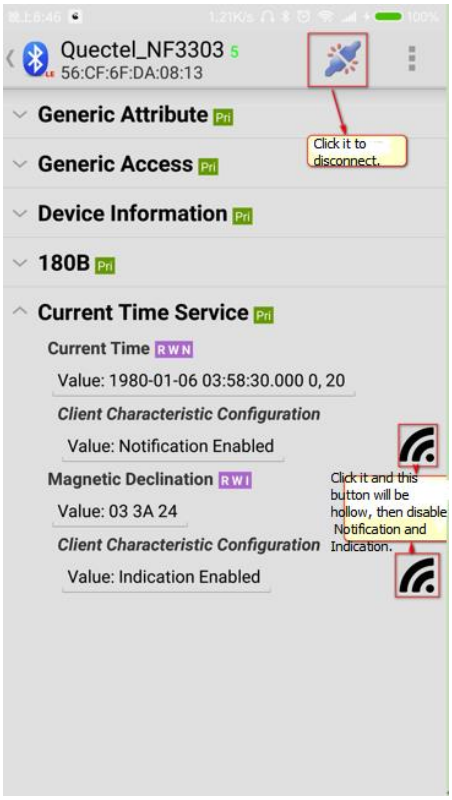
Click Enable Notification button besides “Current Time”, then this button will become solid. The Value represents time will be updated once per 10S. As shown below.

Click Enable Notification button besides “Magnetic Declination”, then this button will become solid. The Value represents time will be updated once per 5S. As shown below.



7.1.4. Disconnect Testing

Close Notification and Indication function, then click disconnect button, as shown as bellow.



7.2. Sleep Wake-up function Testing

Before testing please plug Bluetooth antennas to ensure good signal quality.

7.2.1. NF3303 Waking-up EC2X/G35 Testing

Pull out USB cable, execute demo program, after initialization is done, EX2X & AG35 will go sleep mode as demo program enables autosleep function, data interaction can't be done under serial port terminal. At this time, mobile phone still can scan cm256sm Bluetooth module. After mobile phone connecting with NF3303, Bluetooth module will pull up BT_HOST_WAKE pin and generate riding edge interruption to wake up EC2X/AG35, client application held wakelock. After one time data completed, Bluetooth module will pull low BT_HOST_WAKE pin and generate falling edge interruption, client application release wakelock. If there's no any task holding wakelock, EC2X/G35 will go sleep mode, data interaction can't be done under serial port terminal.

7.2.2. EC2X/AG35 Waking-up NF3303 Testing

When there is no data interaction, Bluetooth protocol stack will generate callback function notification, after client application receive this notification will pull low BT_DEVWAKE pin and allow Bluetooth module go sleep mode. When EC2X/AG35 need do data interaction with Bluetooth, Bluetooth protocol stack will generate callback function notification, after client receive it will pull up BT_DEVWAKE pin to wake up Bluetooth. The oscilloscope can monitor the level change of BT_DEVWAKE pin and test the consumption of Bluetooth module in real time, and verify whether the Bluetooth module is sleeping or waking up.

8 Appendix A References

Table 3: Related Documents

SN	Document Name	Remark
[1]	Quectel_AG35-QuecOpen_Hardware_Design	AG35 Hardware Design Guide
[2]	Quectel_EC20-QuecOpen_Hardware_Design	EC20 Hardware Design Guide
[3]	SA-HRD-211 NF3303 Module Hardware Specification V1.1	NF3303 WIFI/BT Module Spec
[4]	nFore Bluegate Programming Guide 0.10.4	Bluegate Stack API Programming Guide

Table 4: Terms and Abbreviations

Abbreviation	Description
BLE	Bluetooth Low Energy