Lenze GATT SDK INTRODUCTION

Bluetooth Low-Energy Device is developed according to the user's demand, and this document will be introducing how BLE gadgets are developed .

And the procedure will be : 1.understand and realize the on-demand protocol ;2.Test agreement on this ;3.protocol authoring capabilities ; 4.Test the function of the gadget .

Chapter 1 :understand and realize the on-demand protocol

Step 1 understand the user's demands

1. Some key Attribute of the Ble

Service uuid:0x1802
Characteristic uuid:0x2a06 （write，notify）

Service uuid:0x1803
Characteristic uuid:0x2a06 （write ，notify）

Service uuid:0x1804
Characteristic uuid:0x2a07 （read，notify）

Service uuid:0x180f
Characteristic uuid:0x2a19 （read，notify）

Service uuid:0xffe0

Characteristic uuid:0xffe1　（read，notify）

2.　the protocol when connecting to the smart phone

1．Service uuid:0x1802

Characteristic uuid:0x2a06　（write，notify）

Function : **On/Off the anti-loss function,and call for alert when enable it .**

==

2．Service uuid:0x1803

Characteristic uuid:0x2a06　（write ，notify）

Function:　**enable the various distance alert**

==

3．Service uuid:0x1804

Characteristic uuid:0x2a07　（read，notify）

Function: **indicate the status of anti-loss device**

==

4．Service uuid:0x180f

Characteristic uuid:0x2a19　（read）

Function: **battery percentage**

==

5．Service uuid:0xffe0

Characteristic uuid:0xffe1　（read，notify）

Function: **indicate the button-press-status of anti-loss device**

Step 2 Analyzing the demand

Table of the BLE Attribute ：

(UUID：1802):　immediateAlert

	Characterister UUID	Properties	Introduction
immediateAlertLevel_value	0x2A06	W/N	0xf2 0xa0　enable immediate Alert function
			0xf2 0xa1　disable immediate Alert Function
			0xf2 0xb0　**call device**
			0xf2 0xb1 **stop calling the ,or make device quiet**

(UUID：1803):　Link Loss

	Characterister	Propertie	Introduction

	UUID	s	
linkLoss_value	0x2A06	W/N	0xf3 0xa1 **enable the short distance alert**
			0xf3 0xa2 **enable the middle distance alert**
			0xf3 0xa3 **enable the long distance alert**

(UUID：1804): TX POWER :RF transmitting Power

	Characterister UUID	Properties	Introduction
TxPower_value	0x2A07	R/N	Check the current status of the Anti-loss device in a format like:0xf4 data1 data2 data3 data1 : **distance level for alert** ,0x01->short ; 0x02->middle; 0x03->far data2:**the lost and found function is enabled or not**,0x01->Anti-loss function is enabled ; 0x00-> Anti-loss function is disabled data3:**The buzzer status of the Anti-loss device**, 0x01->buzzer on ; 0x00>buzzer off

(UUID：180F): Battery Level

	Characterister UUID	Properties	Introduction
Battery Level	0x2A19	R/N	0-100 battery percentage

(UUID：FFE0): PrivateServices

	Characterister UUID	Properties	Introduction
KeyPress	0xFFE1	R/N	Listen for button notification button press 0xfe 0x01 Button release 0xfe 0x00

step 3　Add attribute into the app_att.c

Firstly ,define new attribute or add the general attribute into the app_att.c　table list .

////////////////////// Immediate Alert Service UUID 1802 ///////////////////////////////

//define immediate Alert Service ,since this is the general attribute , you can search in the coming files after press the ‘ ctrl+H’　button.

static const u16 immediateAlert_serviceUUID　= SERVICE_UUID_IMMEDIATE_ALERT;

//define the character of Immediate Alert Attribute

static const u8 immediateAlertLevel_prop =CHAR_PROP_WRITE_WITHOUT_RSP | CHAR_PROP_WRITE | CHAR_PROP_NOTIFY;

//define the maximum quantity of　Immediate Alert attribute value: 2 bytes.

u8 immediateAlertLevel_value[2] = {0x01,0x01};

//////////////////////　　linkLoss Service UUID　1803　///////////////////////////////

//define linkLoss Service ,since this is the general attribute , you can search in the coming files //after press the ‘ ctrl+H’　button.

static const u16 linkLoss_serviceUUID　= SERVICE_UUID_LINK_LOSS;

//define the character of linkLoss Service

static const u8 linkLoss_prop =CHAR_PROP_WRITE | CHAR_PROP_NOTIFY;//CHAR_PROP_WRITE | CHAR_PROP_NOTIFY; u8 linkLoss_value[2] = {0x00};

//define the maximum quantity of　configuration　of linkLoss Service : 2 bytes.

u8 linkLoss_valueInCCC[2];

//////////////////////　　TX Power Servic UUID　　1804　　///////////////////////////////

//define TxPower Service , since this is the general attribute , you can search in the coming files //after press the ‘ ctrl+H’　button.

static const u16 TxPower_serviceUUID　=　SERVICE_UUID_TX_POWER;

//define the character of TxPower Service

static const u8 TxPower_prop = CHAR_PROP_READ | CHAR_PROP_NOTIFY;

//define the maximum quantity of　configuration　of TxPower Service value :4 bytes.

u8 Txpower_value[4] = {0xF4,0x03,0x00,0x00};

//////////////////////　　Battery Service UUID　180f　　///////////////////////////////

//define my_bat Service , since this is the general attribute , you can search in the coming files //after press the ‘ ctrl+H’　button.

const u16 my_batServiceUUID　　　　　　= SERVICE_UUID_BATTERY;

//define the character of my_bat Service

const u16 my_batCharUUID= CHARACTERISTIC_UUID_BATTERY_LEVEL;

//define the maximum quantity of my_bat Service attribute : 2 bytes.

u8 my_batVal = 100;

//define the maximum quantity of configuration of my_bat Service value :4 bytes.

u8 generalValInCCC[2];

////////////////////////// Private Service UUID 0xffe0 ////////////////////////////////////

//define FFE0 Service , since this is the private main attribute , you cannot search in the coming //files after press the ‘ ctrl+H’ button.

static const u16 FFE0_UUID = 0xffe0;

////define FFE1 Service , since this is the private son attribute

static const u16 FFE1_charUUID = 0xffe1;

//define the character of FFE1 Service

static const u8 FFE1_prop = CHAR_PROP_NOTIFY | CHAR_PROP_READ;

//define the maximum quantity of ffe1 Service attribute : 2 bytes.

u8 FFE1_value[2] = {0x00};

Secondly ,Add the already defined attributes into the attribute_t my_Attributes[] ={}; list ,then finish the attribute part .

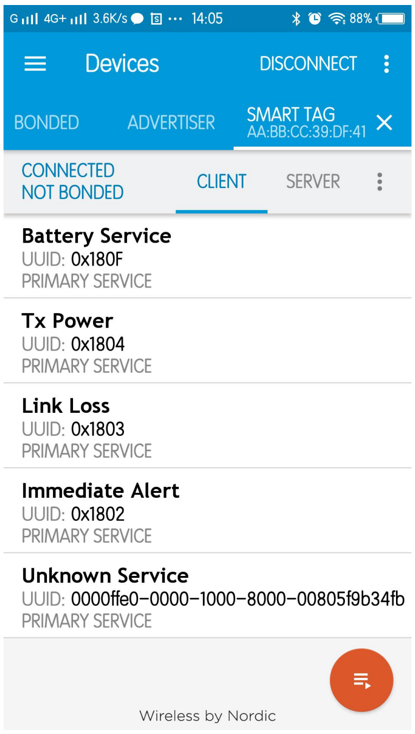
```
const attribute_t my_Attributes[] =
{
    {18,0,0,0,0},
/*************************************************************************
    1                Immediate Alert   (Services)   1802
*************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&immediateAlert_serviceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&immediateAlertLevel_prop)},
    {0,2,sizeof(immediateAlertLevel_value),(u8*)(&my_2a06_UIID),
(u8*)(&immediateAlertLevel_value)},
//  {0,2,sizeof(immediateAlertLevel_value),(u8*)(&clientCharacterCfgUUID),
(u8*)(immediateAlertLevel_value)},   //value
/*************************************************************************
    4                Link Loss   (Services)   1803
*************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&linkLoss_serviceUUID)},
    {0,1,1,(u8*)(&my_characterUUID),        (u8*)(&linkLoss_prop)},
    {0,2,sizeof(linkLoss_value),            (u8*)(&my_2a06_UIID),
(u8*)(&linkLoss_value)},
//  {0,2,sizeof(linkLoss_value),(u8*)(&clientCharacterCfgUUID),     (u8*)(linkLoss_value)},
//value

/*************************************************************************
```

```
            5                    TX_POWER    (Services) 1804
    ***************************************************************************/
        {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&TxPower_serviceUUID)},
        {0,2,1,(u8*)(&my_characterUUID),         (u8*)(&TxPower_prop)},
        {0,2,sizeof(Txpower_value),              (u8*)(&my_2a07_UUID),
(u8*)(&Txpower_value)},
    //  {0,2,sizeof(Txpower_value),(u8*)(&clientCharacterCfgUUID),     (u8*)(Txpower_value)},
//value
    /***************************************************************************
                        Battery (Services)    180F
    ***************************************************************************/
        {4,2,2,    (u8*)(&my_primaryServiceUUID),   (u8*)(&my_batServiceUUID)},
        {0,2,1,    (u8*)(&my_characterUUID),     (u8*)(&PROP_READ_NOTIFY)},     //prop
        {0,2,sizeof(my_batVal), (u8*)(&my_batCharUUID),(u8*)(&my_batVal)}, //value
        {0,2,sizeof(generalValInCCC),     (u8*)(&clientCharacterCfgUUID),
(u8*)(generalValInCCC)},   //value

    /***************************************************************************
        14                         Private    (Services)
    ***************************************************************************/
        {4,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&FFE0_UUID)},
        {0,2,1,(u8*)(&my_characterUUID),         (u8*)(&FFE1_prop)},
        {0,2,sizeof(FFE1_value),(u8*)(&FFE1_charUUID),   (u8*)(&FFE1_value)},
        {0,2,sizeof(FFE1_value),(u8*)(&clientCharacterCfgUUID),            (u8*)(FFE1_value)},
//value
};
```

Chapter 2　test the format of attribute

step 1　generate .bin file and download into the development board

Up to now, we can use the Nordic App in the android phone to search the device we've developed ,and if you can see the below screenshot ,that means you have done with configuration of the attributes.



G 4G+ 3.6K/s 14:05 88%

Devices　DISCONNECT

BONDED　ADVERTISER　SMART TAG AA:BB:CC:39:DF:41

CONNECTED NOT BONDED　CLIENT　SERVER

Battery Service
UUID: 0x180F
PRIMARY SERVICE

Tx Power
UUID: 0x1804
PRIMARY SERVICE

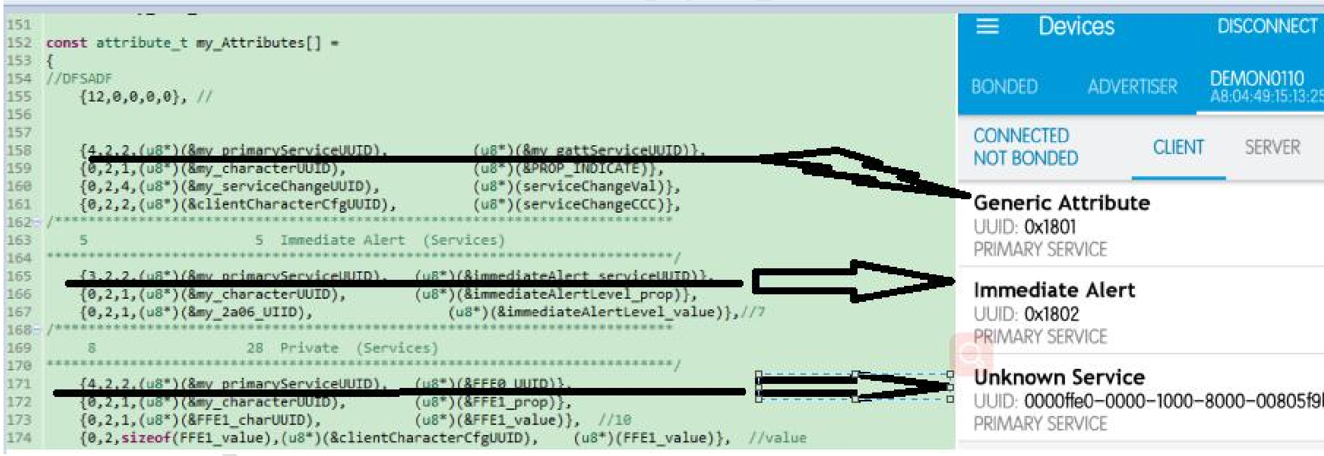
Link Loss
UUID: 0x1803
PRIMARY SERVICE

Immediate Alert
UUID: 0x1802
PRIMARY SERVICE

Unknown Service
UUID: 0000ffe0-0000-1000-8000-00805f9b34fb
PRIMARY SERVICE

Wireless by Nordic

You can understand the relationship between the program and the list in Nordic in a corresponding way .



```
const attribute_t my_Attributes[] =
{
//DFSADF
    {12,0,0,0,0}, //


    {4,2,2,(u8*)(&my_primaryServiceUUID),          (u8*)(&my_gattServiceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),               (u8*)(&PROP_INDICATE)},
    {0,2,4,(u8*)(&my_serviceChangeUUID),           (u8*)(serviceChangeVal)},
    {0,2,2,(u8*)(&clientCharacterCfgUUID),         (u8*)(serviceChangeCCC)},
/*********************************************************************************
    5                    5  Immediate Alert  (Services)
*********************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&immediateAlert_serviceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&immediateAlertLevel_prop)},
    {0,2,1,(u8*)(&my_2a06_UIID),                (u8*)(&immediateAlertLevel_value)},//7
/*********************************************************************************
     8                  28  Private  (Services)
*********************************************************************************/
    {4,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&FFE0_UUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&FFE1_prop)},
    {0,2,1,(u8*)(&FFE1_charUUID),           (u8*)(&FFE1_value)},  //10
    {0,2,sizeof(FFE1_value),(u8*)(&clientCharacterCfgUUID),    (u8*)(FFE1_value)},  //value
```

Devices　DISCONNECT

BONDED　ADVERTISER　DEMON0110 A8:04:49:15:13:25

CONNECTED NOT BONDED　CLIENT　SERVER

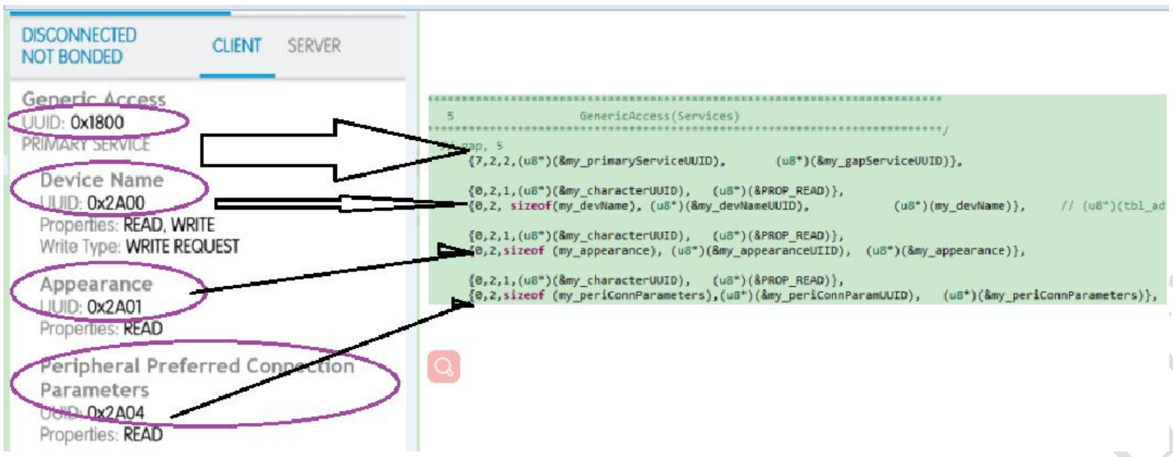
Generic Attribute
UUID: 0x1801
PRIMARY SERVICE

Immediate Alert
UUID: 0x1802
PRIMARY SERVICE

Unknown Service
UUID: 0000ffe0-0000-1000-8000-00805f9
PRIMARY SERVICE

Look at the above graph , each underlined attribute refer to the father-Attribute in black bold,and the first factor in the underlined father-Attribute {n},indicates n handles are contained in this Main -attribute, and among the main-attribute ,each son-Attribute start with 0 as their first factor ,as a result,the first factor can be used to distinguish the father and son attribute.

You can find the extended list with both the father-Attribute and son-attribute.



```
*****************************************************************************
  5                  GenericAccess(Services)
*****************************************************************************/
//gap, 5
    {7,2,2,(u8*)(&my_primaryServiceUUID),        (u8*)(&my_gapServiceUUID)},

    {0,2,1,(u8*)(&my_characterUUID),    (u8*)(&PROP_READ)},
    {0,2, sizeof(my_devName), (u8*)(&my_devNameUUID),           (u8*)(my_devName)},    // (u8*)(tbl_ad

    {0,2,1,(u8*)(&my_characterUUID),    (u8*)(&PROP_READ)},
    {0,2,sizeof (my_appearance), (u8*)(&my_appearanceUIID),  (u8*)(&my_appearance)},

    {0,2,1,(u8*)(&my_characterUUID),    (u8*)(&PROP_READ)},
    {0,2,sizeof (my_periConnParameters),(u8*)(&my_periConnParamUUID),   (u8*)(&my_periConnParameters)},
```

step 2　utilize the attribute and test until it works right

In fact , it's rare for us to success within just one try ,the protocol may has something wrong , we will list out the possible problems and solutions to give you some inspirations.

No.1 Why I changed the definition in the attribute list ,but still can't see any effect after connect to the app?

Analyse : our mobile phone will record the mac with attribute list that we connected recently ,so if you just changed the attribute and forgot changing the mac address ,then you can't see what's new .

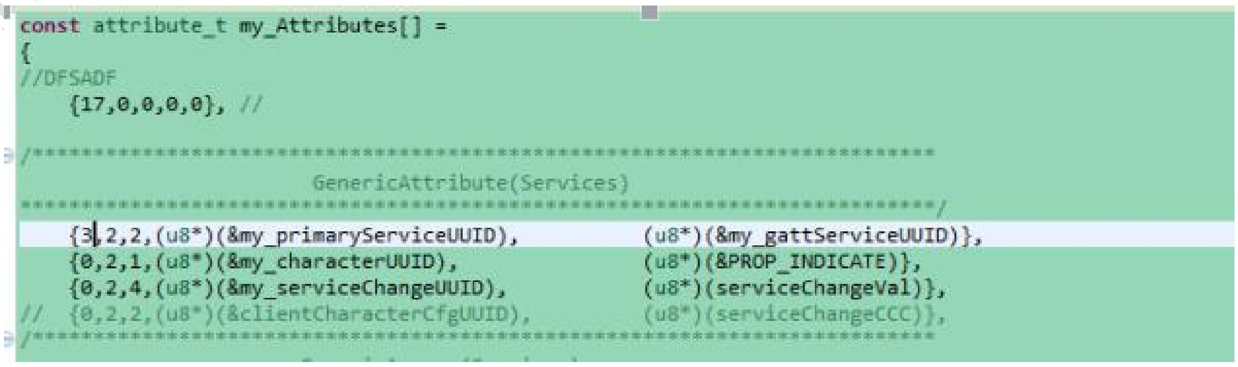
No.2 why app failed to indicate the correct number in the attribute list?

Analyse : If we covered the 4th line in the list ,and modify the `tbl_mac [] into  {0x25, 0x13, 0x15, 0x49, 0x04, 0xa8};`

Then in the following figure ,we can see what's acquired in the app.

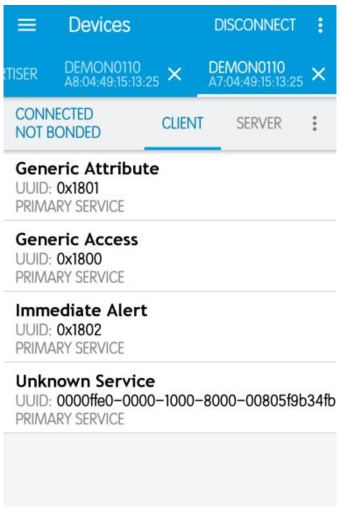
Since in the first attribute-group,the first factor :att_Num, tells there are 4 handles in the main-attribute , but actually the quantity is 3. the APP will ignore the first attribute-group,and skip to the attribute-group with correct factor and att_Num.

Because in the first group of parent services ,the first factor :Att_num shows the father-attribute contains 4 handles, and the actual handle value is 3, after connect to Nordic app ,the first set of services is not assumed completed, and will directly skip to the 2nd corresponding set of father-attribute. We turn the Att_num into 3, and the program will turn into below,



```
const attribute_t my_Attributes[] =
{
//DFSADF
    {17,0,0,0,0}, //

/*****************************************************************************
                        GenericAttribute(Services)
*****************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),          (u8*)(&my_gattServiceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),               (u8*)(&PROP_INDICATE)},
    {0,2,4,(u8*)(&my_serviceChangeUUID),           (u8*)(serviceChangeVal)},
//  {0,2,2,(u8*)(&clientCharacterCfgUUID),         (u8*)(serviceChangeCCC)},
/*****************************************************************************
```

Correspondingly,we can get the attribute list as below on our mobile phone.



Generic Attribute
UUID: 0x1801
PRIMARY SERVICE

Generic Access
UUID: 0x1800
PRIMARY SERVICE

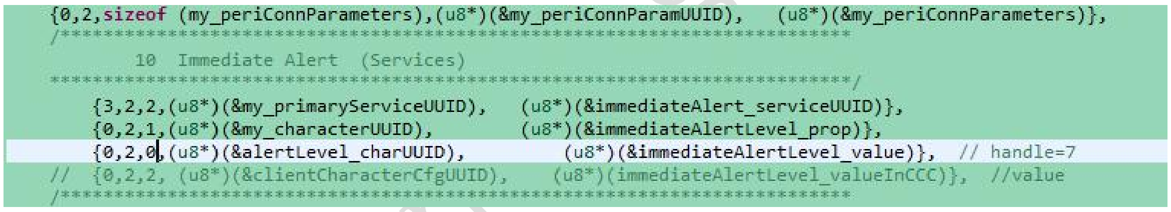
Immediate Alert
UUID: 0x1802
PRIMARY SERVICE

Unknown Service
UUID: 0000ffe0-0000-1000-8000-00805f9b34fb
PRIMARY SERVICE

No.3 while succeed in building ,but app fail to acquire any attributes

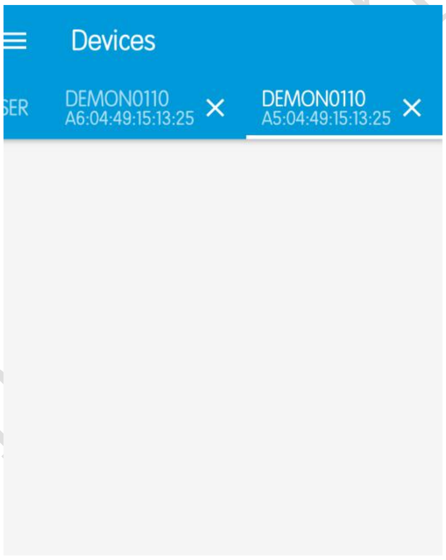
Analyse : something wrong with factors within each attribute.

1)att_length is Less than the actual length



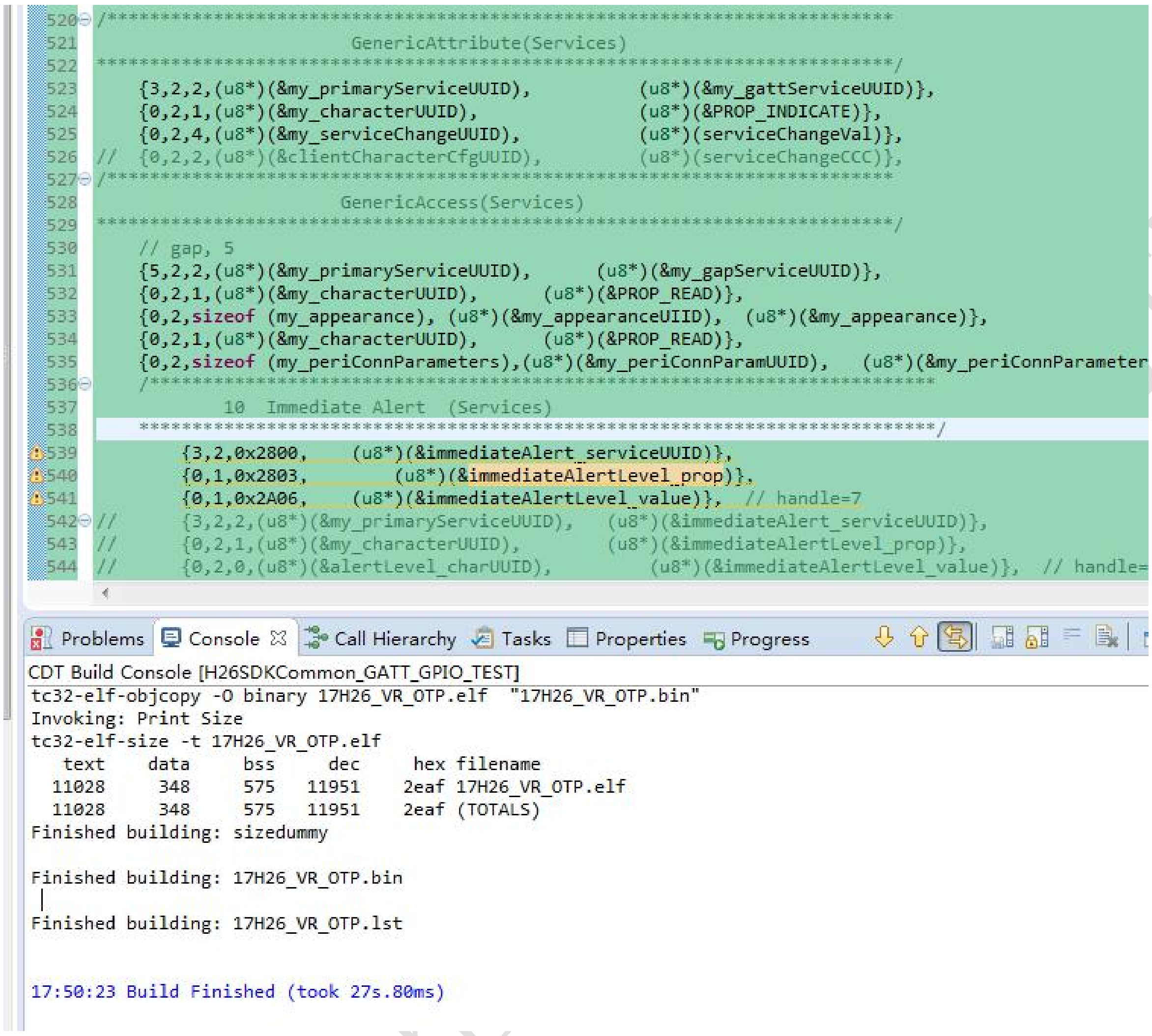
```
    {0,2,sizeof (my_periConnParameters),(u8*)(&my_periConnParamUUID),   (u8*)(&my_periConnParameters)},
    /************************************************************************

         10  Immediate Alert  (Services)
    ************************************************************************/
        {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&immediateAlert_serviceUUID)},
        {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&immediateAlertLevel_prop)},
        {0,2,0,(u8*)(&alertLevel_charUUID),         (u8*)(&immediateAlertLevel_value)},  // handle=7
    //  {0,2,2, (u8*)(&clientCharacterCfgUUID),    (u8*)(immediateAlertLevel_valueInCCC)},  //value
    /************************************************************************
```

Which will cause the disorder in app end after connection:



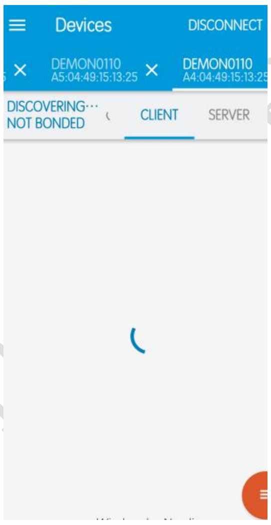
2)The structure in the list does not match the system-defined attribute_t structure

In this case, although the Xx.bin file can also be compiled, equipment broadcast can also be searched , but the phone will   be stuck in the process while accessing to the service phase, as shown below, the underlined yellow warning part of the attribute_t structure is inconsistent, although through the compiler's detection, but the smart phone cannot get any services.



```
/*************************************************************************
                        GenericAttribute(Services)
*************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),          (u8*)(&my_gattServiceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),               (u8*)(&PROP_INDICATE)},
    {0,2,4,(u8*)(&my_serviceChangeUUID),           (u8*)(serviceChangeVal)},
//  {0,2,2,(u8*)(&clientCharacterCfgUUID),         (u8*)(serviceChangeCCC)},
/*************************************************************************
                       GenericAccess(Services)
*************************************************************************/
    // gap, 5
    {5,2,2,(u8*)(&my_primaryServiceUUID),      (u8*)(&my_gapServiceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),       (u8*)(&PROP_READ)},
    {0,2,sizeof (my_appearance), (u8*)(&my_appearanceUIID),  (u8*)(&my_appearance)},
    {0,2,1,(u8*)(&my_characterUUID),       (u8*)(&PROP_READ)},
    {0,2,sizeof (my_periConnParameters),(u8*)(&my_periConnParamUUID),   (u8*)(&my_periConnParameter
    /*************************************************************************
              10  Immediate Alert  (Services)
    *************************************************************************/
        {3,2,0x2800,    (u8*)(&immediateAlert_serviceUUID)},
        {0,1,0x2803,         (u8*)(&immediateAlertLevel_prop)},
        {0,1,0x2A06,    (u8*)(&immediateAlertLevel_value)},  // handle=7
//      {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&immediateAlert_serviceUUID)},
//      {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&immediateAlertLevel_prop)},
//      {0,2,0,(u8*)(&alertLevel_charUUID),         (u8*)(&immediateAlertLevel_value)},  // handle=
```

```
CDT Build Console [H26SDKCommon_GATT_GPIO_TEST]
tc32-elf-objcopy -O binary 17H26_VR_OTP.elf  "17H26_VR_OTP.bin"
Invoking: Print Size
tc32-elf-size -t 17H26_VR_OTP.elf
   text    data     bss     dec     hex filename
  11028     348     575   11951    2eaf 17H26_VR_OTP.elf
  11028     348     575   11951    2eaf (TOTALS)
Finished building: sizedummy

Finished building: 17H26_VR_OTP.bin

Finished building: 17H26_VR_OTP.lst


17:50:23 Build Finished (took 27s.80ms)
```



Typical problems are summed up as the above ,and in the actual operation ,there might be many unexpected problems and phenomena, but basically could be caused by one or more of these problems .Hope that in doing this part of the exception., users could pay more attention to the details within the attribute_t.

Chapter 3 Protocol Authoring Capabilities

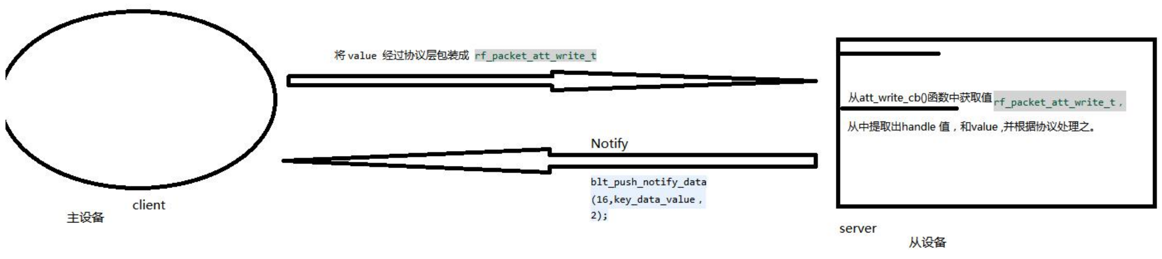
So far as we introduced about the BLE, on the surface , as long as there are some modification in the attribute ,we were able to write down the corresponding control program ,and complete the authoring capabilities.

These are only from the perspective of application layer however.In the blue tooth system ,the link layer ,communication layer, each layer has a corresponding identification code that needs to be included in the total packet . what's more ,there will be some encryption each time when we send the data to the other end of corresponding object.

Step 1 Understand the DATA transmission process

Just now we talked about the Bluetooth protocol and the structure of the data. So how do our application data modify different services and complete the processing of the settings in the service table? This involves our data receive callback function ATT_WRITE_CB (). In a word, understanding the following diagram may be helpful when understand the whole process of data transmission.

The leftmost ellipse in the figure below represents the client device, typically a mobile phone or a Bluetooth central device. While the right side represents the server-side of the device typically a peripheral device, in this article we are going to introduce the program of the 17h26 from the perspective of peripheral device.


将 value 经过协议层包装成 rf_packet_att_write_t
从att_write_cb()函数中获取值 rf_packet_att_write_t，
从中提取出handle 值，和value ,并根据协议处理之。
Notify
blt_push_notify_data
(16,key_data_value，
2);
client
主设备
server
从设备

Further description about this part ,is contained in the reference<Lenze 17H26 BLE SDK User Guide new>.

Step 2 get a basic knowledge of BLE 4.0 protocol

Here is a general reference about the protocol of BLE 4.0 , the specific communication theory can be found in <>

下图中的低功耗蓝牙 4.0 协议只是一个参考，具体的通信原理需参见：《Principle and implementation method of Bluetooth 4.0 protocol with low energy consumption》。

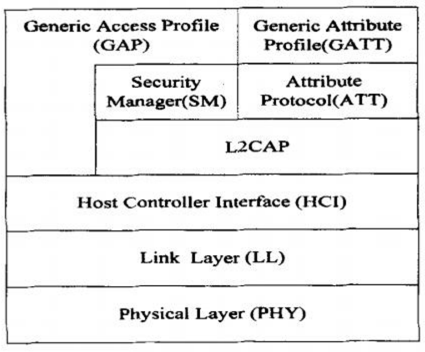
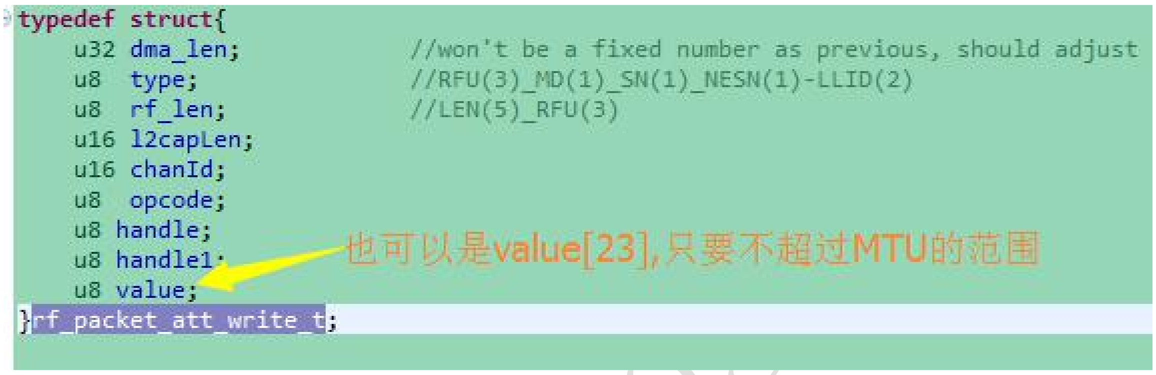

Generic Access Profile (GAP)
Generic Attribute Profile(GATT)
Security Manager(SM)
Attribute Protocol(ATT)
L2CAP
Host Controller Interface (HCI)
Link Layer (LL)
Physical Layer (PHY)

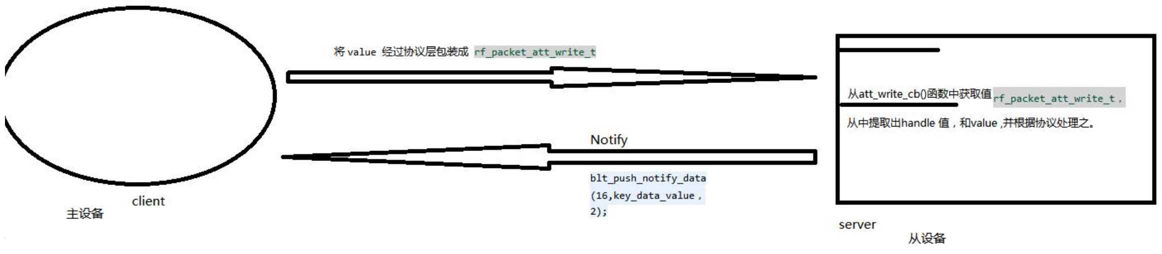
图 1 协议分层结构图

And the communication data is organized based on the format of the above protocol , like below :

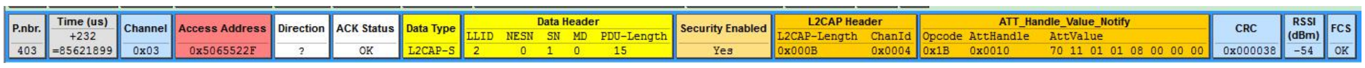


```
typedef struct{
    u32 dma_len;            //won't be a fixed number as previous, should adjust
    u8  type;               //RFU(3)_MD(1)_SN(1)_NESN(1)-LLID(2)
    u8  rf_len;             //LEN(5)_RFU(3)
    u16 l2capLen;
    u16 chanId;
    u8  opcode;
    u8 handle;
    u8 handle1;
    u8 value;
}rf_packet_att_write_t;
```

the application layer is closely related to the data is the yellow arrow in the above refers to (Handle1<<8|handle), where the handle1 is corresponding to the handle of the high-byte data is generally 0, when the total number of handle (we will call it handle) is more than 255 After that, handle1 not equal to 0. When the total number of handle is less than 255,low-byte handle represents the handle data itself.And then ,we see value,the length of data could be modified but should be limited within MTU (maximum transmit unit) (Other data are the same as the Bluetooth protocol structure one by one above,More specification is referred in <CSS V6> .(which related to everything about Bluetooth 4.0 protocol .)


将 value 经过协议层包装成 rf_packet_att_write_t
Notify
blt_push_notify_data
(16,key_data_value ,
2);
从att_write_cb()函数中获取值 rf_packet_att_write_t ,
从中提取出handle 值，和value ,并根据协议处理之。
client
主设备
server
从设备

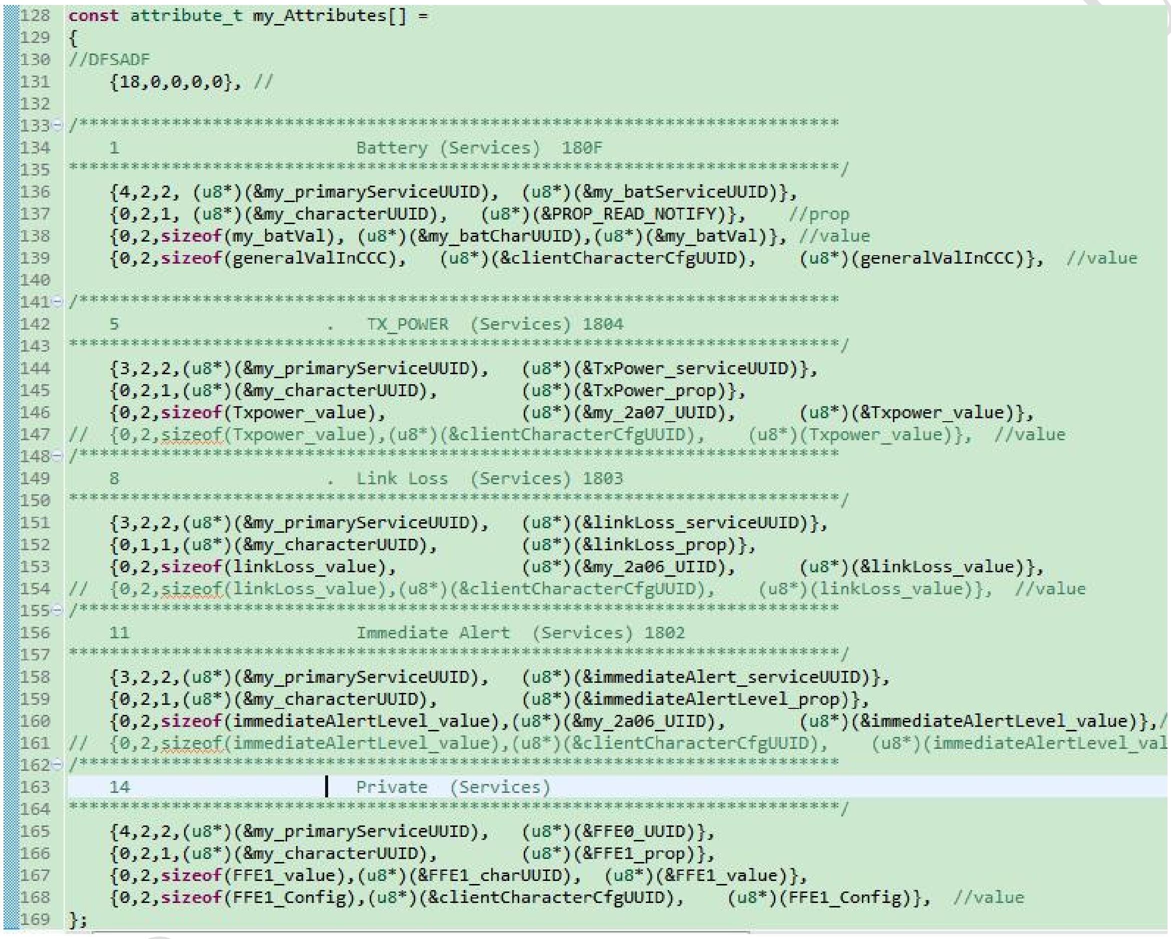
As to the above graph, here is a decrypted data that send over the air .each format of data explanation can be found in <Lenze 17H26 BLE SDK User Guide new>.



P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header					Security Enabled	L2CAP Header		ATT_Handle_Value_Notify			CRC	RSSI (dBm)	FCS
	+232						LLID	NESN	SN	MD	PDU-Length		L2CAP-Length	ChanId	Opcode	AttHandle	AttValue			
403	=85621899	0x03	0x5065522F	?	OK	L2CAP-S	2	0	1	0	15	Yes	0x000B	0x0004	0x1B	0x0010	70 11 01 01 08 00 00 00	0x000038	-54	OK

Step 3　Understand the sending mechanism among 17H26

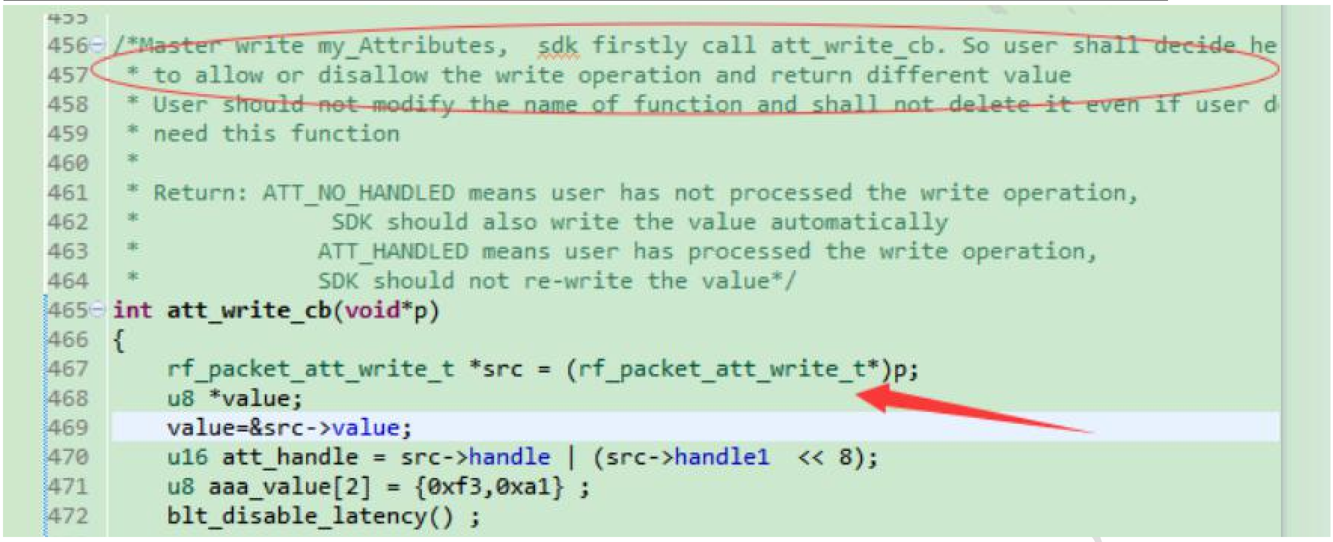
Handle in the BLE service list, the attribute Handle value starts from the 0x0001, with a step 1 increment, and the array's subscript starts from 0, adds this virtual attribute in the attribute Table, just makes each Attribute's subscript in the data equals the value of its Attribute Handle. When the attribute table is defined, the number of attributes in the attribute table below is settled. As a result, the mobile phone will be able to know the current attribute Handle of the attribute service,based on the previous list of the services. For example,if the user needs to notify FFE1 related data, the handle used is 16.



```
const attribute_t my_Attributes[] =
{
//DFSADF
    {18,0,0,0,0}, //

/*****************************************************************************
    1                       Battery (Services)  180F
*****************************************************************************/
    {4,2,2, (u8*)(&my_primaryServiceUUID),  (u8*)(&my_batServiceUUID)},
    {0,2,1, (u8*)(&my_characterUUID),   (u8*)(&PROP_READ_NOTIFY)},    //prop
    {0,2,sizeof(my_batVal), (u8*)(&my_batCharUUID),(u8*)(&my_batVal)}, //value
    {0,2,sizeof(generalValInCCC),   (u8*)(&clientCharacterCfgUUID),    (u8*)(generalValInCCC)},  //value

/*****************************************************************************
    5                     .   TX_POWER  (Services) 1804
*****************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&TxPower_serviceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&TxPower_prop)},
    {0,2,sizeof(Txpower_value),             (u8*)(&my_2a07_UUID),      (u8*)(&Txpower_value)},
//  {0,2,sizeof(Txpower_value),(u8*)(&clientCharacterCfgUUID),    (u8*)(Txpower_value)},  //value
/*****************************************************************************
    8                     .  Link Loss  (Services) 1803
*****************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&linkLoss_serviceUUID)},
    {0,1,1,(u8*)(&my_characterUUID),        (u8*)(&linkLoss_prop)},
    {0,2,sizeof(linkLoss_value),            (u8*)(&my_2a06_UIID),      (u8*)(&linkLoss_value)},
//  {0,2,sizeof(linkLoss_value),(u8*)(&clientCharacterCfgUUID),    (u8*)(linkLoss_value)},  //value
/*****************************************************************************
    11                      Immediate Alert  (Services) 1802
*****************************************************************************/
    {3,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&immediateAlert_serviceUUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&immediateAlertLevel_prop)},
    {0,2,sizeof(immediateAlertLevel_value),(u8*)(&my_2a06_UIID),      (u8*)(&immediateAlertLevel_value)},/
//  {0,2,sizeof(immediateAlertLevel_value),(u8*)(&clientCharacterCfgUUID),    (u8*)(immediateAlertLevel_val
/*****************************************************************************
    14                      Private  (Services)
*****************************************************************************/
    {4,2,2,(u8*)(&my_primaryServiceUUID),   (u8*)(&FFE0_UUID)},
    {0,2,1,(u8*)(&my_characterUUID),        (u8*)(&FFE1_prop)},
    {0,2,sizeof(FFE1_value),(u8*)(&FFE1_charUUID),  (u8*)(&FFE1_value)},
    {0,2,sizeof(FFE1_Config),(u8*)(&clientCharacterCfgUUID),    (u8*)(FFE1_Config)},  //value
};
```

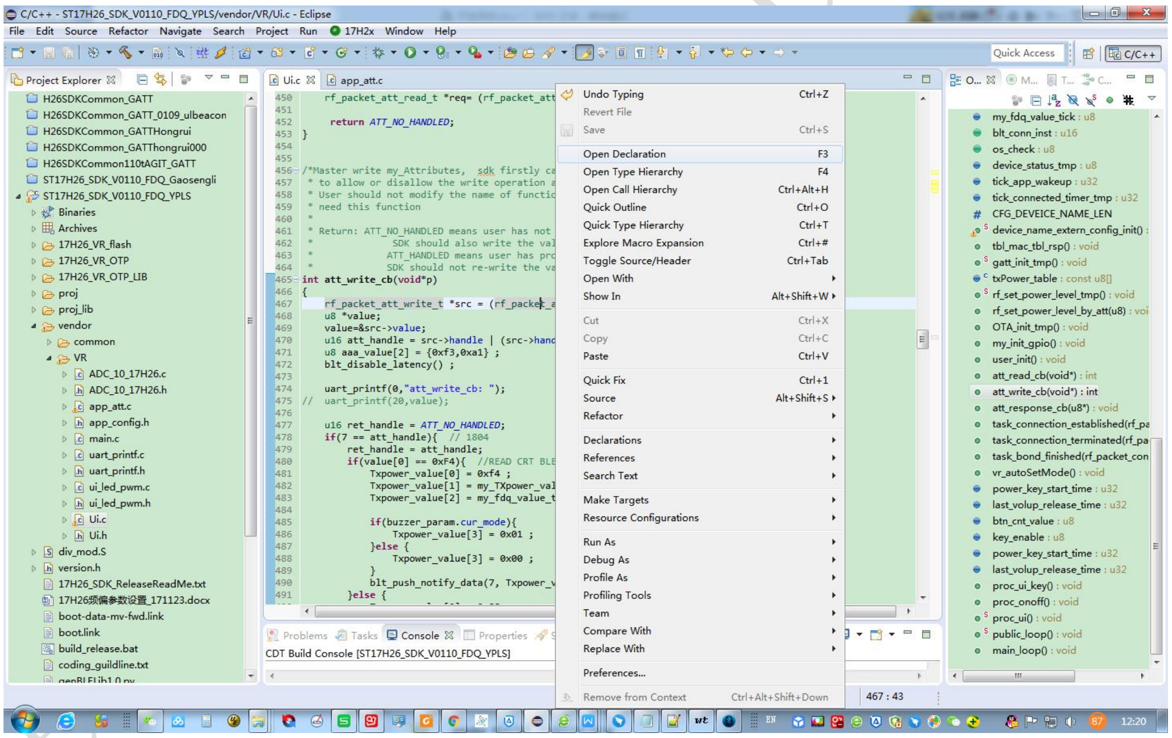
Step 4 understand the receiving mechanism among 17H26

The pointer type indicated by the red arrow shown in the following illustration, is the Bluetooth data packet that we described above. Our data exists in this packet, which is not greater than the MTU, the maximum amount of data in a protocol, usually 23 bytes.



```
/*Master write my_Attributes,  sdk firstly call att_write_cb. So user shall decide he
 * to allow or disallow the write operation and return different value
 * User should not modify the name of function and shall not delete it even if user d
 * need this function
 *
 * Return: ATT_NO_HANDLED means user has not processed the write operation,
 *              SDK should also write the value automatically
 *             ATT_HANDLED means user has processed the write operation,
 *             SDK should not re-write the value*/
int att_write_cb(void*p)
{
    rf_packet_att_write_t *src = (rf_packet_att_write_t*)p;
    u8 *value;
    value=&src->value;
    u16 att_handle = src->handle | (src->handle1  << 8);
    u8 aaa_value[2] = {0xf3,0xa1} ;
    blt_disable_latency() ;
```

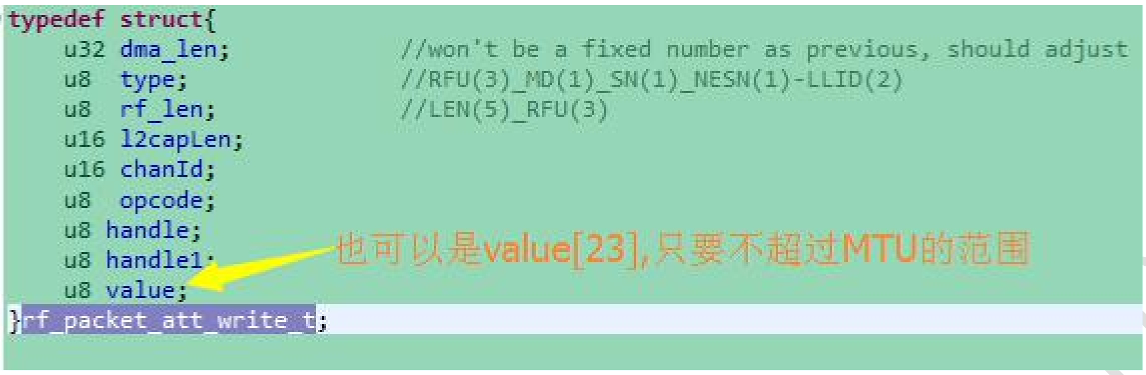
The red arrow in the picture indicates the structure which contains the command data in it .We can extend it to the MTU .And the method is down below:



```
    rf_packet_att_read_t *req= (rf_packet_att

    return ATT_NO_HANDLED;
}


/*Master write my_Attributes,  sdk firstly ca
 * to allow or disallow the write operation a
 * User should not modify the name of functic
 * need this function
 *
 * Return: ATT_NO_HANDLED means user has not
 *             SDK should also write the val
 *            ATT_HANDLED means user has prc
 *            SDK should not re-write the va
int att_write_cb(void*p)
{
    rf_packet_att_write_t *src = (rf_packet_a
    u8 *value;
    value=&src->value;
    u16 att_handle = src->handle | (src->hanc
    u8 aaa_value[2] = {0xf3,0xa1} ;
    blt_disable_latency() ;

    uart_printf(0,"att_write_cb: ");
//  uart_printf(20,value);

    u16 ret_handle = ATT_NO_HANDLED;
    if(7 == att_handle){  // 1804
        ret_handle = att_handle;
        if(value[0] == 0xF4){  //READ CRT BLE
            Txpower_value[0] = 0xf4 ;
            Txpower_value[1] = my_TXpower_val
            Txpower_value[2] = my_fdq_value_t

            if(buzzer_param.cur_mode){
                Txpower_value[3] = 0x01 ;
            }else {
                Txpower_value[3] = 0x00 ;
            }
            blt_push_notify_data(7, Txpower_v
        }else {
```

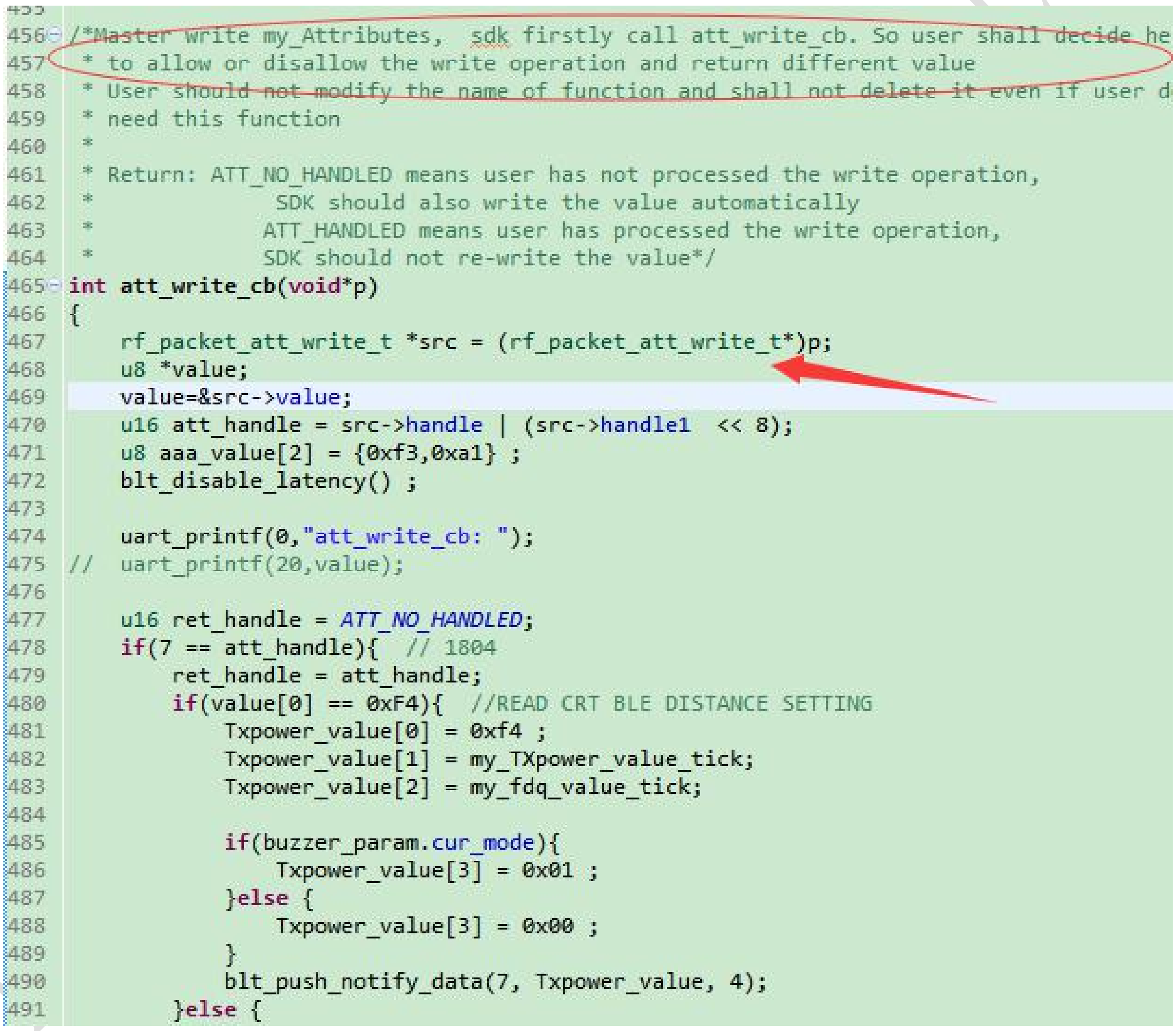
CDT Build Console [ST17H26_SDK_V0110_FDQ_YPLS]

For example ,we select the structure we are going to use ,for example rf_packet_att_write_t ,then click the right side of the mouse , which will open the new interface like below, change value into value[n],n<23,so there will be n bytes data translated at once.



```c
typedef struct{
    u32 dma_len;            //won't be a fixed number as previous, should adjust
    u8  type;               //RFU(3)_MD(1)_SN(1)_NESN(1)-LLID(2)
    u8  rf_len;             //LEN(5)_RFU(3)
    u16 l2capLen;
    u16 chanId;
    u8  opcode;
    u8 handle;
    u8 handle1;
    u8 value;
}rf_packet_att_write_t;
```

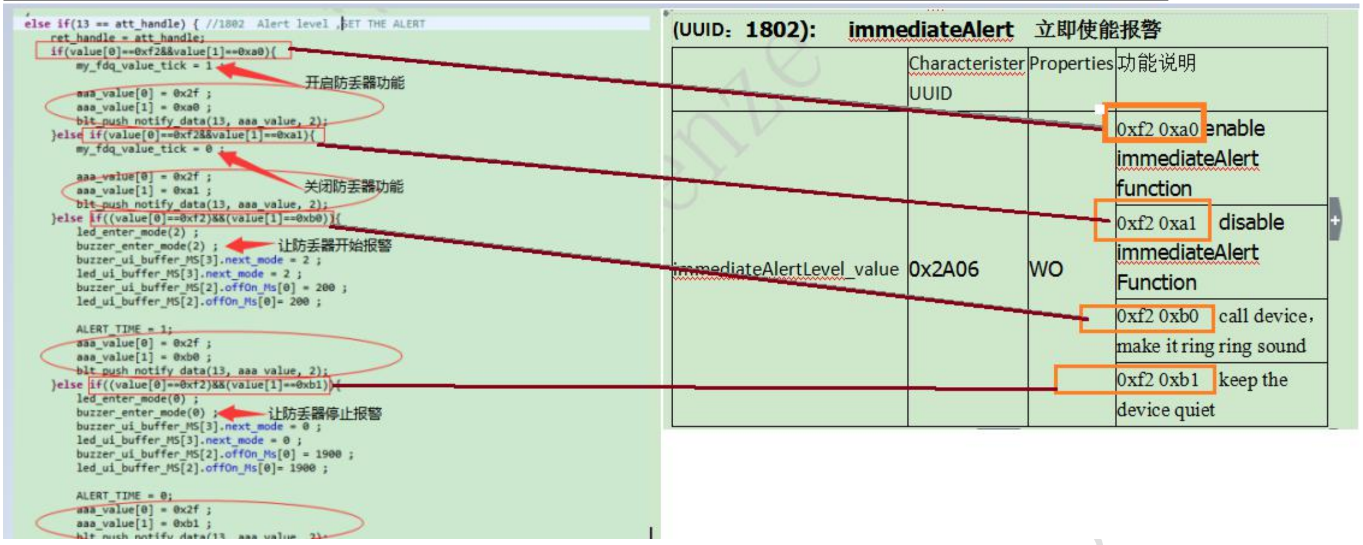
也可以是value[23],只要不超过MTU的范围



```c
/*Master write my_Attributes,  sdk firstly call att_write_cb. So user shall decide he
 * to allow or disallow the write operation and return different value
 * User should not modify the name of function and shall not delete it even if user d
 * need this function
 *
 * Return: ATT_NO_HANDLED means user has not processed the write operation,
 *                SDK should also write the value automatically
 *               ATT_HANDLED means user has processed the write operation,
 *               SDK should not re-write the value*/
int att_write_cb(void*p)
{
    rf_packet_att_write_t *src = (rf_packet_att_write_t*)p;
    u8 *value;
    value=&src->value;
    u16 att_handle = src->handle | (src->handle1  << 8);
    u8 aaa_value[2] = {0xf3,0xa1} ;
    blt_disable_latency() ;

    uart_printf(0,"att_write_cb: ");
//  uart_printf(20,value);

    u16 ret_handle = ATT_NO_HANDLED;
    if(7 == att_handle){  // 1804
        ret_handle = att_handle;
        if(value[0] == 0xF4){  //READ CRT BLE DISTANCE SETTING
            Txpower_value[0] = 0xf4 ;
            Txpower_value[1] = my_TXpower_value_tick;
            Txpower_value[2] = my_fdq_value_tick;

            if(buzzer_param.cur_mode){
                Txpower_value[3] = 0x01 ;
            }else {
                Txpower_value[3] = 0x00 ;
            }
            blt_push_notify_data(7, Txpower_value, 4);
        }else {
```

The reader must have a lot of doubts in mind, such as how to send data from the device to the main device side? What does the handle of the main device mean?

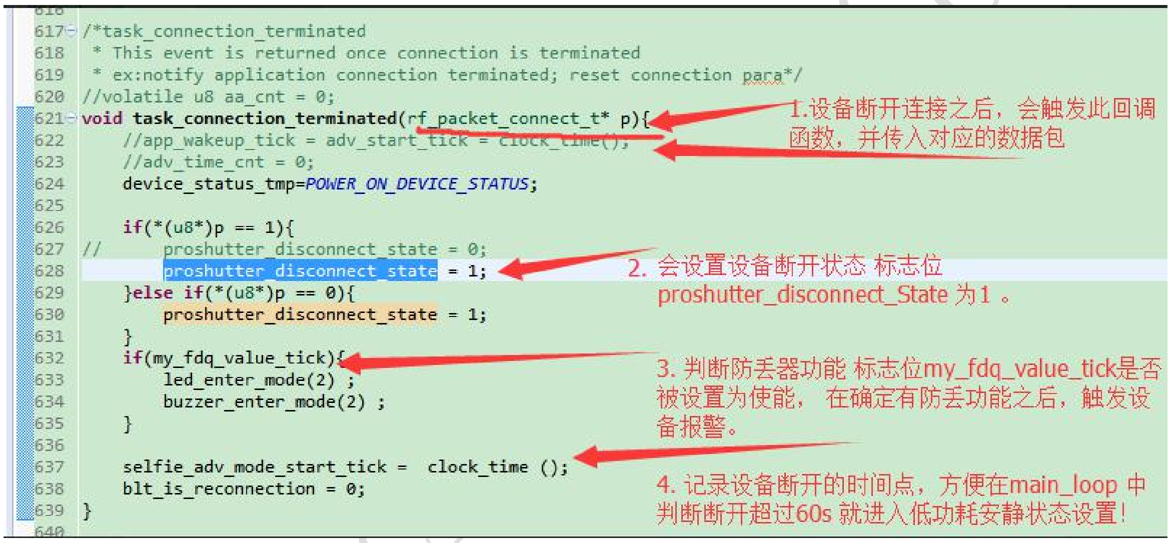
As a matter of fact ,the sending and receiving are the same ,for example ,the device wants to get the immediateAlertLevel_value.We should find the corresponding handle (in our case this handle = 13)which contains this immediateAlertLevel_value ,and try to deal with it.



```
else if(13 == att_handle) { //1802  Alert level ,SET THE ALERT
    ret_handle = att_handle;
    if(value[0]==0xf2&&value[1]==0xa0){
        my_fdq_value_tick = 1 ;

        aaa_value[0] = 0x2f ;
        aaa_value[1] = 0xa0 ;
        blt_push_notify_data(13, aaa_value, 2);
    }else if(value[0]==0xf2&&value[1]==0xa1){
        my_fdq_value_tick = 0 ;

        aaa_value[0] = 0x2f ;
        aaa_value[1] = 0xa1 ;
        blt_push_notify_data(13, aaa_value, 2);
    }else if((value[0]==0xf2)&&(value[1]==0xb0)){
        led_enter_mode(2) ;
        buzzer_enter_mode(2) ;
        buzzer_ui_buffer_MS[3].next_mode = 2 ;
        led_ui_buffer_MS[3].next_mode = 2 ;
        buzzer_ui_buffer_MS[2].offOn_Ms[0] = 200 ;
        led_ui_buffer_MS[2].offOn_Ms[0]= 200 ;

        ALERT_TIME = 1;
        aaa_value[0] = 0x2f ;
        aaa_value[1] = 0xb0 ;
        blt_push_notify_data(13, aaa_value, 2);
    }else if((value[0]==0xf2)&&(value[1]==0xb1)){
        led_enter_mode(0) ;
        buzzer_enter_mode(0) ;
        buzzer_ui_buffer_MS[3].next_mode = 0 ;
        led_ui_buffer_MS[3].next_mode = 0 ;
        buzzer_ui_buffer_MS[2].offOn_Ms[0] = 1900 ;
        led_ui_buffer_MS[2].offOn_Ms[0]= 1900 ;

        ALERT_TIME = 0;
        aaa_value[0] = 0x2f ;
        aaa_value[1] = 0xb1 ;
        blt_push_notify_data(13, aaa_value, 2);
```

(UUID：1802): immediateAlert 立即使能报警

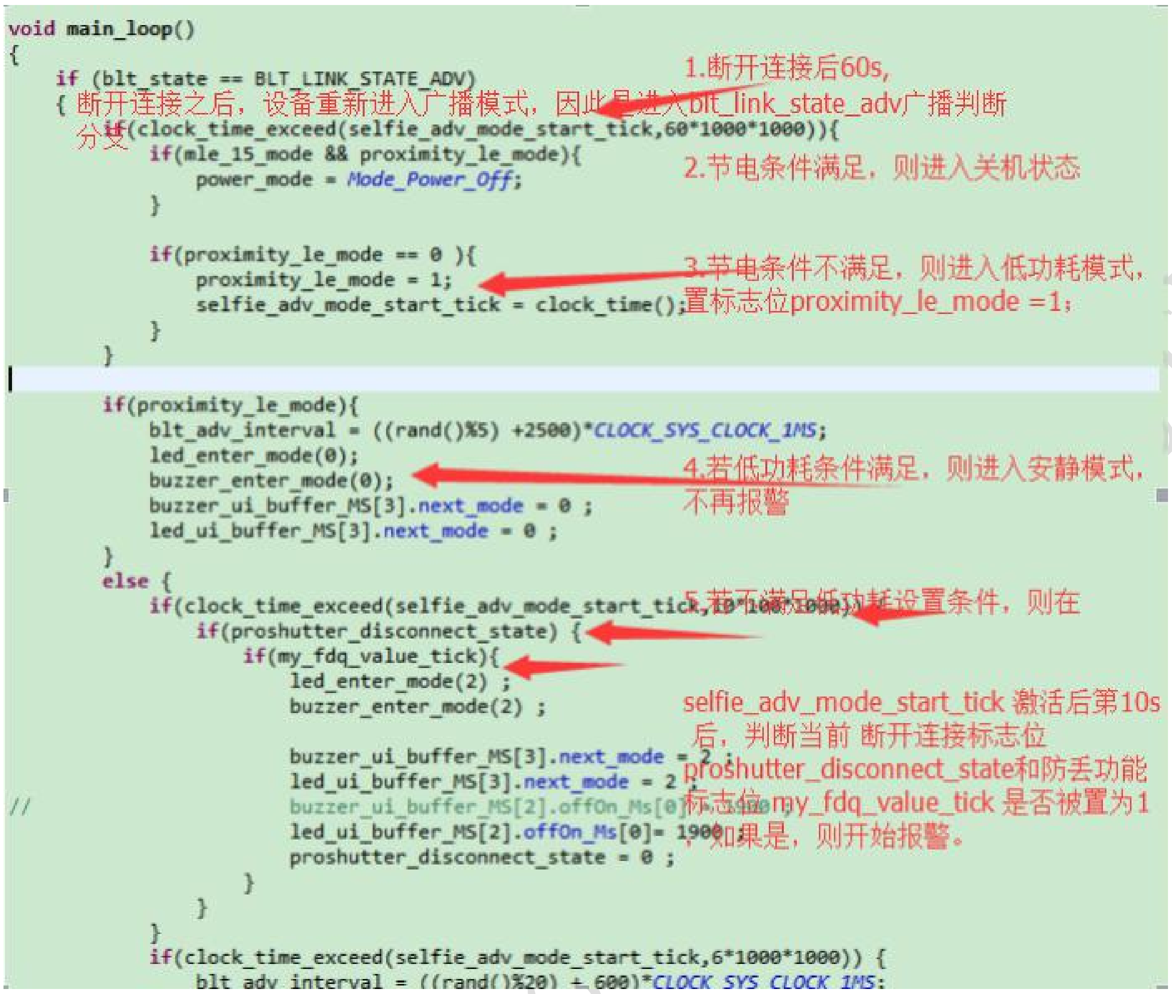
	Characterister UUID	Properties	功能说明
immediateAlertLevel_value	0x2A06	WO	0xf2 0xa0 enable immediateAlert function
			0xf2 0xa1 disable immediateAlert Function
			0xf2 0xb0 call device, make it ring ring sound
			0xf2 0xb1 keep the device quiet



```
/*task_connection_terminated
 * This event is returned once connection is terminated
 * ex:notify application connection terminated; reset connection para*/
//volatile u8 aa_cnt = 0;
void task_connection_terminated(rf_packet_connect_t* p){
    //app_wakeup_tick = adv_start_tick = clock_time();
    //adv_time_cnt = 0;
    device_status_tmp=POWER_ON_DEVICE_STATUS;

    if(*(u8*)p == 1){
//      proshutter_disconnect_state = 0;
        proshutter_disconnect_state = 1;
    }else if(*(u8*)p == 0){
        proshutter_disconnect_state = 1;
    }
    if(my_fdq_value_tick){
        led_enter_mode(2) ;
        buzzer_enter_mode(2) ;
    }

    selfie_adv_mode_start_tick =  clock_time ();
    blt_is_reconnection = 0;
}
```

1.设备断开连接之后，会触发此回调函数，并传入对应的数据包
2. 会设置设备断开状态 标志位 proshutter_disconnect_State 为1 。
3. 判断防丢器功能 标志位my_fdq_value_tick是否被设置为使能， 在确定有防丢功能之后，触发设备报警。
4. 记录设备断开的时间点，方便在main_loop 中判断断开超过60s 就进入低功耗安静状态设置！

In contrast to the control program in the Main_loop, we will understand how to implement the device alarm function in the program, the other parts of the protocol can be understand in the same way .



```
void main_loop()
{
    if (blt_state == BLT_LINK_STATE_ADV)
    {
        if(clock_time_exceed(selfie_adv_mode_start_tick,60*1000*1000)){
            if(mle_15_mode && proximity_le_mode){
                power_mode = Mode_Power_Off;
            }

            if(proximity_le_mode == 0 ){
                proximity_le_mode = 1;
                selfie_adv_mode_start_tick = clock_time();
            }
        }

        if(proximity_le_mode){
            blt_adv_interval = ((rand()%5) +2500)*CLOCK_SYS_CLOCK_1MS;
            led_enter_mode(0);
            buzzer_enter_mode(0);
            buzzer_ui_buffer_MS[3].next_mode = 0 ;
            led_ui_buffer_MS[3].next_mode = 0 ;
        }
        else {
            if(clock_time_exceed(selfie_adv_mode_start_tick,10*1000*1000)){
                if(proshutter_disconnect_state) {
                    if(my_fdq_value_tick){
                        led_enter_mode(2) ;
                        buzzer_enter_mode(2) ;

                        buzzer_ui_buffer_MS[3].next_mode = 2 ;
                        led_ui_buffer_MS[3].next_mode = 2 ;
//                      buzzer_ui_buffer_MS[2].offOn_Ms[0]= 1900 ;
                        led_ui_buffer_MS[2].offOn_Ms[0]= 1900 ;
                        proshutter_disconnect_state = 0 ;
                    }
                }
            }
            if(clock_time_exceed(selfie_adv_mode_start_tick,6*1000*1000)) {
                blt_adv_interval = ((rand()%20) + 600)*CLOCK_SYS_CLOCK_1MS;
```

1.断开连接后60s,

断开连接之后，设备重新进入广播模式，因此是进入blt_link_state_adv广播判断分支

2.节电条件满足，则进入关机状态

3.节电条件不满足，则进入低功耗模式，置标志位proximity_le_mode =1;

4.若低功耗条件满足，则进入安静模式，不再报警

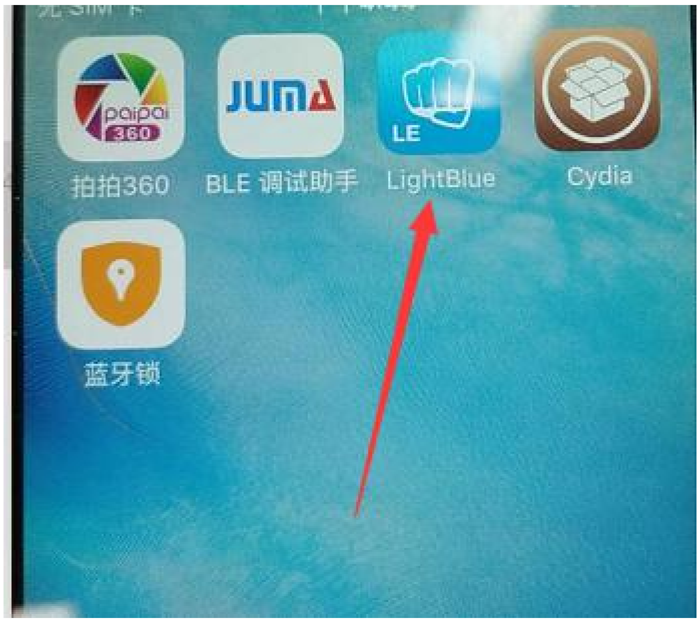
5.若不满足低功耗设置条件，则在selfie_adv_mode_start_tick 激活后第10s后，判断当前 断开连接标志位proshutter_disconnect_state和防丢功能标志位 my_fdq_value_tick 是否被置为1，如果是，则开始报警。

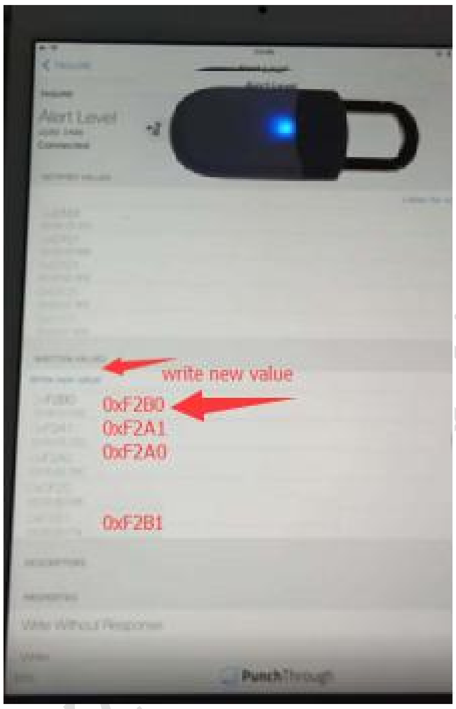
At this point, we have completed the protocol and program one by one . Next, we are going to enter the test module to verify that our program has really achieved the desired goal.

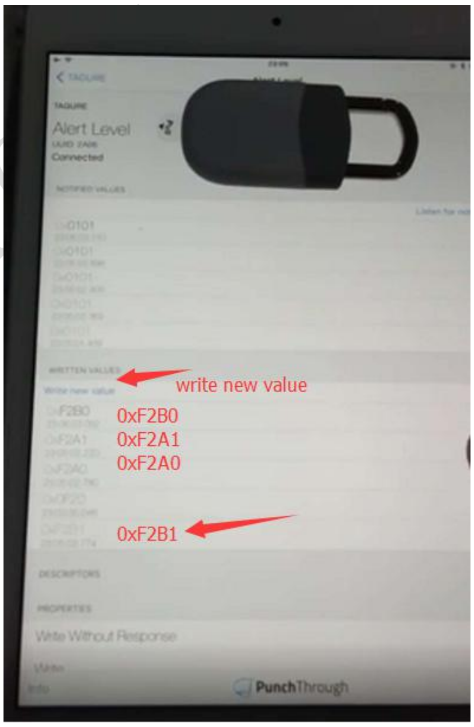
Chapter 4　Test module

We've just implemented the corresponding capability in theory, but we need to test it with other tools to verify that the program is actually complete.For example, in this case, we used the LightBlue APP on apple to complete the relevant tests, and of course, we can use the Nordic APP in Android to complete the test.The effect is basically the same, and the Nordic APP can also display the MAC address of the device, so we recommended reader to use sniffer later.

step 1 test the function when receiving the command

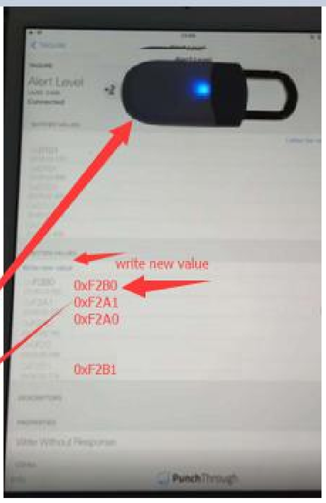

拍拍360
BLE 调试助手
LightBlue
Cydia
蓝牙锁

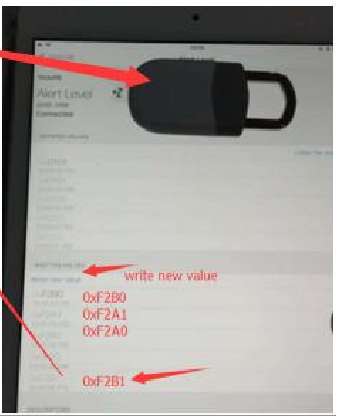

Alert Level
write new value
0xF2B0
0xF2A1
0xF2A0
0xF2B1
Write Without Response
PunchThrough


Alert Level
write new value
0xF2B0
0xF2A1
0xF2A0
0xF2B1
Write Without Response
PunchThrough

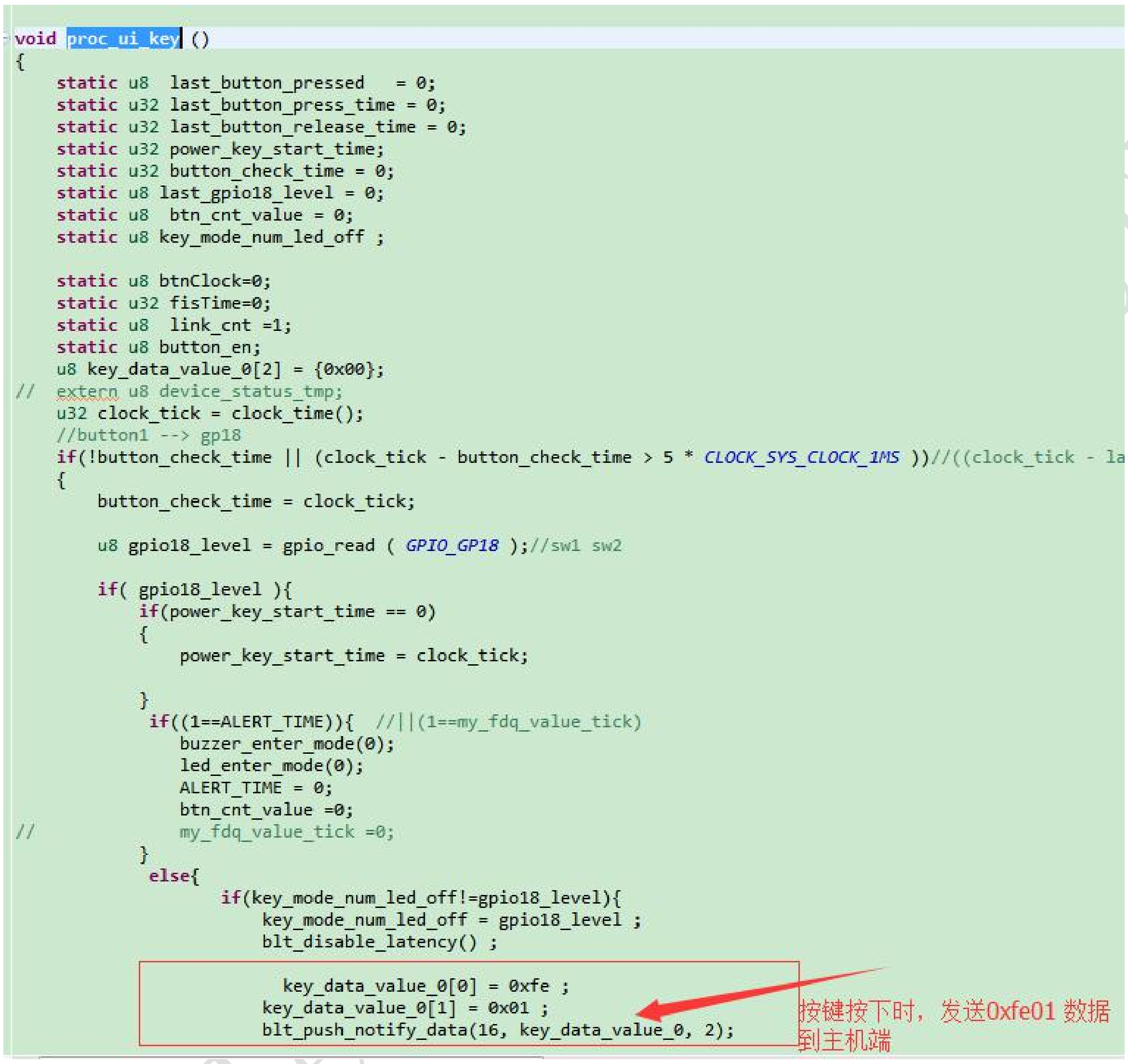
(UUID：1802): immediateAlert 立即使能报警

	Characterister UUID	Properties	功能说明
immediateAlertLevel_value	0x2A06	W/N	0xf2 0xa0 打开防丢功能 enable immediateAlert function
			0xf2 0xa1 关闭防丢功能 disable immediateAlert Function
			0xf2 0xb0 呼叫防丢器 call device
			0xf2 0xb1 停止呼叫防丢器 keep the device quiet



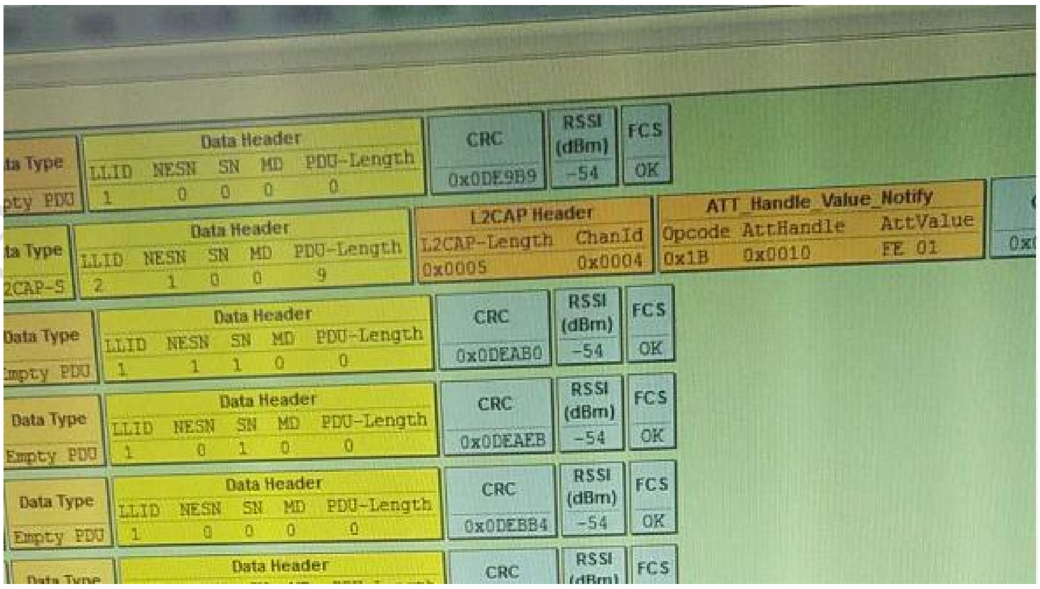


Step 2 test the function when sending commands



```
void proc_ui_key ()
{
    static u8  last_button_pressed   = 0;
    static u32 last_button_press_time = 0;
    static u32 last_button_release_time = 0;
    static u32 power_key_start_time;
    static u32 button_check_time = 0;
    static u8 last_gpio18_level = 0;
    static u8  btn_cnt_value = 0;
    static u8 key_mode_num_led_off ;

    static u8 btnClock=0;
    static u32 fisTime=0;
    static u8  link_cnt =1;
    static u8 button_en;
    u8 key_data_value_0[2] = {0x00};
//  extern u8 device_status_tmp;
    u32 clock_tick = clock_time();
    //button1 --> gp18
    if(!button_check_time || (clock_tick - button_check_time > 5 * CLOCK_SYS_CLOCK_1MS ))//((clock_tick - la
    {
        button_check_time = clock_tick;

        u8 gpio18_level = gpio_read ( GPIO_GP18 );//sw1 sw2

        if( gpio18_level ){
            if(power_key_start_time == 0)
            {
                power_key_start_time = clock_tick;

            }
             if((1==ALERT_TIME)){  //||(1==my_fdq_value_tick)
                buzzer_enter_mode(0);
                led_enter_mode(0);
                ALERT_TIME = 0;
                btn_cnt_value =0;
//              my_fdq_value_tick =0;
            }
             else{
                    if(key_mode_num_led_off!=gpio18_level){
                        key_mode_num_led_off = gpio18_level ;
                        blt_disable_latency() ;

                          key_data_value_0[0] = 0xfe ;
                        key_data_value_0[1] = 0x01 ;
                        blt_push_notify_data(16, key_data_value_0, 2);
```

In the receiving test, we can also use the mobile phone app to send command, next we will introduce the packet Sniffer software and hardware Sniffer sniffer with the grab device sent key values, as following:



```
ta Type   | Data Header                       | CRC        | RSSI  | FCS
          | LLID  NESN  SN  MD  PDU-Length    |            | (dBm) |
pty PDU   | 1     0     0   0   0             | 0x0DE9B9   | -54   | OK

ta Type   | Data Header                       | L2CAP Header          | ATT_Handle_Value_Notify          |
          | LLID  NESN  SN  MD  PDU-Length    | L2CAP-Length  ChanId  | Opcode  AttHandle  AttValue      |
2CAP-S    | 2     1     0   0   9             | 0x0005        0x0004  | 0x1B    0x0010     FE 01         | 0x

Data Type | Data Header                       | CRC        | RSSI  | FCS
          | LLID  NESN  SN  MD  PDU-Length    |            | (dBm) |
mpty PDU  | 1     1     1   0   0             | 0x0DEAB0   | -54   | OK

Data Type | Data Header                       | CRC        | RSSI  | FCS
          | LLID  NESN  SN  MD  PDU-Length    |            | (dBm) |
Empty PDU | 1     0     1   0   0             | 0x0DEAEB   | -54   | OK

Data Type | Data Header                       | CRC        | RSSI  | FCS
          | LLID  NESN  SN  MD  PDU-Length    |            | (dBm) |
Empty PDU | 1     0     0   0   0             | 0x0DEBB4   | -54   | OK

Data Type | Data Header                       | CRC        | RSSI  | FCS
```

Step 3　iterations from test to programming

If there are other problems found in the test, then some parts of the program are not working properly, you need to find this part of the program, and then optimize it.
If the test is fully passed, then the user needs have been realized. You can go to the following steps.