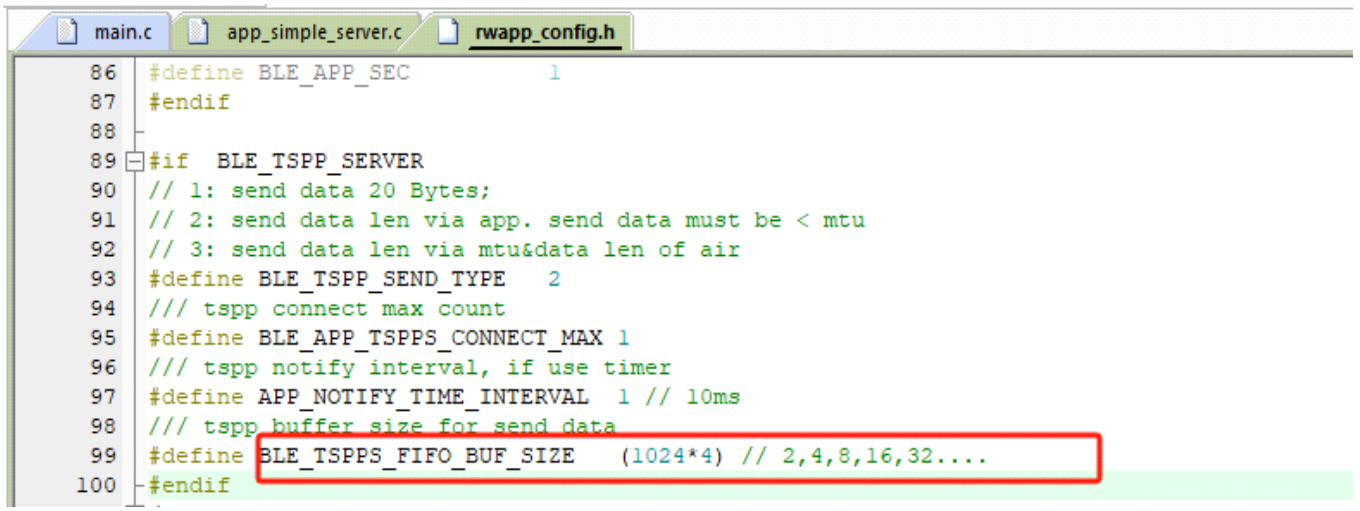
OM6621X 蓝牙私有服务的添加

2024年12月18日 13:15

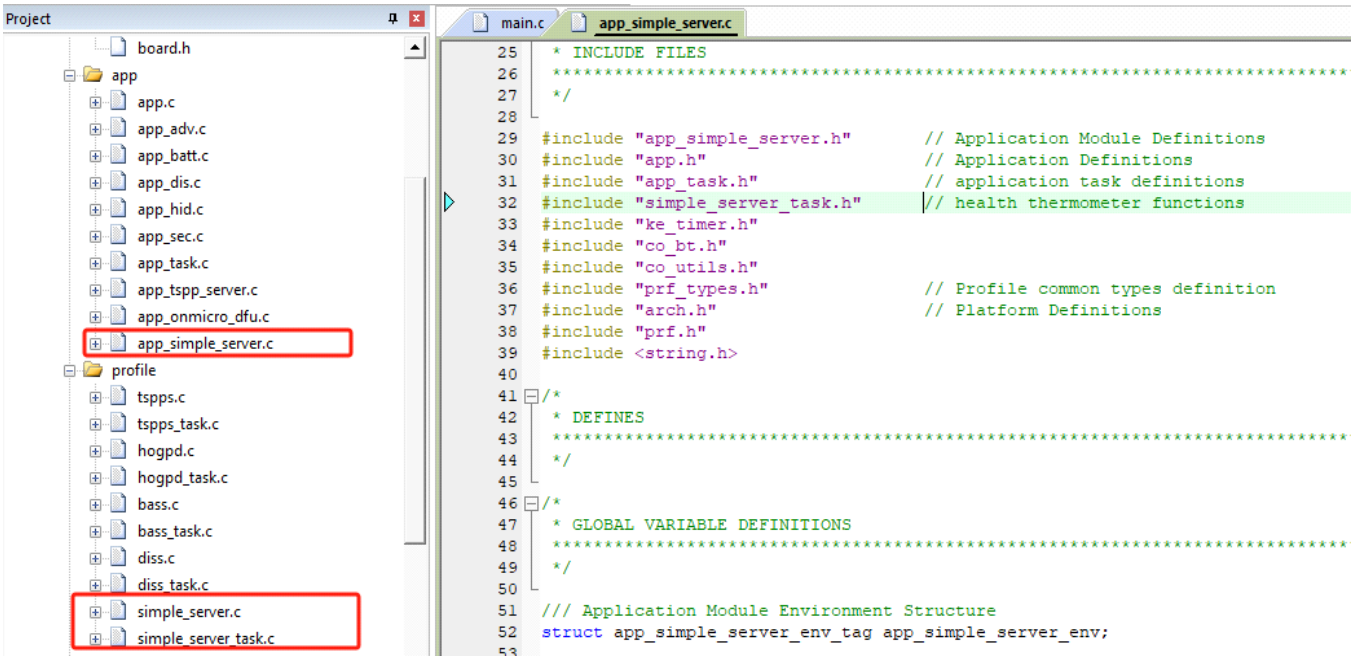
私有服务的添加，一般是参考simple server/tspp server：

区别在于tspp服务默认使能了FIFO，方便透传大数据；

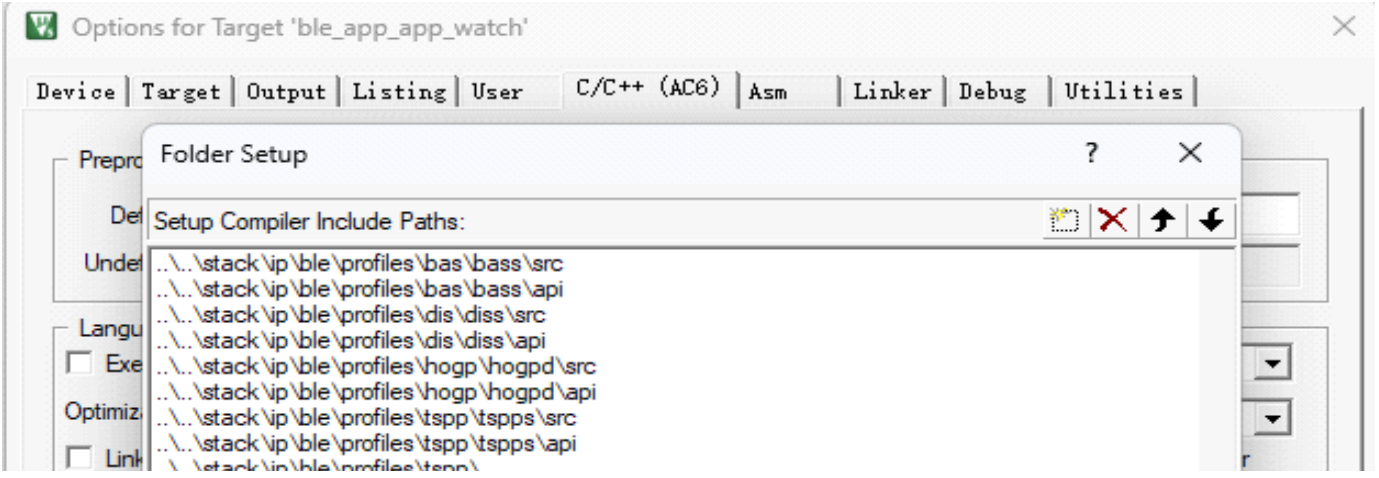


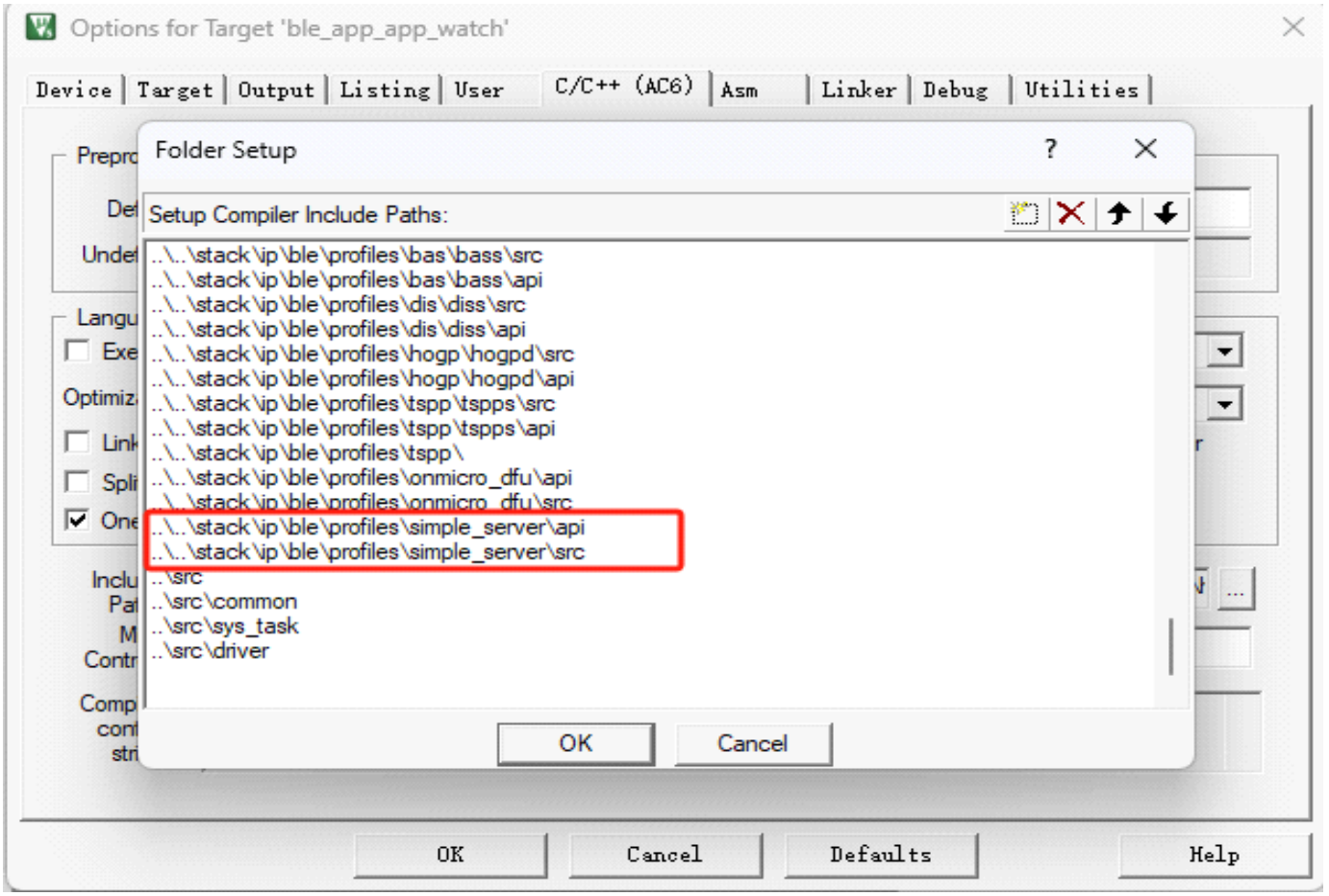
添加simple server的工程流程：

1.添加文件，simple server，文件位置，可以参考工程已有的服务的路径



2.添加文件路径





3.使能服务，添加服务初始化

1）头文件声明

main.c | app_simple_server.c | app_task.c | app_task.h

```
59  #include "app_batt.h"                // Battery Application Definitions
60  #include "bass.h"
61  #endif
62  #if BLE_APP_HID
63  #include "app_hid.h"                 // HID Application Definitions
64  #include "hogpd.h"
65  #endif
66  #if BLE_TSPP_SERVER
67  #include "app_tspp_server.h"
68  #include "tspps.h"
69  #endif
70
71  #if BLE_APP_ONMICRO_DFU
72  #include "onmicro_dfu.h"
73  #include "app_onmicro_dfu.h"
74  #endif //BLE_APP_ONMICRO_DFU
75
76  #if BLE_SIMPLE_SERVER
77  #include "app_simple_server.h"
78  #include "simple_server.h"
79  #endif
```

2）加入服务list

app_simple_server.c | app.c

```
 * ENUMERATIONS
 ****************************************************************************************************
 */

/// List of service to add in the database
enum appm_svc_list
{
    #if (BLE_APP_DIS)
    APPM_SVC_DIS,
    #endif //(BLE_APP_DIS)
```

app_simple_server.c | app.c

```
 * ENUMERATIONS
 ****************************************************************************************************
 */

/// List of service to add in the database
enum appm_svc_list
{
    #if (BLE_APP_DIS)
    APPM_SVC_DIS,
    #endif //(BLE_APP_DIS)
    #if (BLE_APP_BATT)
    APPM_SVC_BATT,
    #endif //(BLE_APP_BATT)
    #if (BLE_SIMPLE_SERVER)
    APPM_SVC_SIMPLE_SERVER,
    #endif //(BLE_SIMPLE_SERVER)
    #if (BLE_TSPP_SERVER)
    APPM_SVC_TSPP_SERVER,
    #endif //(BLE_TSPP_SERVER)
    #if (BLE_APP_HID)
    APPM_SVC_HIDS,
    #endif //(BLE_APP_HID)
    #if (BLE_APP_ONMICRO_DFU)
    APPM_SVC_ONMICRO_DFU,
    #endif //(BLE_APP_ONMICRO_DFU)
    APPM_SVC_LIST_STOP,
};
```

3）添加功能list（app.c）

```
/// Application Task Descriptor
extern const struct ke_task_desc TASK_DESC_APP;

/// List of functions used to create the database
static const appm_add_svc_func_t appm_add_svc_func_list[APPM_SVC_LIST_STOP] =
{
    #if (BLE_APP_DIS)
    (appm_add_svc_func_t)app_dis_add_dis,
    #endif //(BLE_APP_DIS)
    #if (BLE_APP_BATT)
    (appm_add_svc_func_t)app_batt_add_bas,
    #endif //(BLE_APP_BATT)
    #if (BLE_SIMPLE_SERVER)
    (appm_add_svc_func_t)app_simple_server_add_service,
    #endif //(BLE_SIMPLE_SERVER)
    #if (BLE_TSPP_SERVER)
    (appm_add_svc_func_t)app_tspp_server_add_service,
    #endif //(BLE_TSPP_SERVER)
    #if (BLE_APP_HID)
    (appm_add_svc_func_t)app_hid_add_hids,
    #endif //(BLE_APP_HID)
    #if (BLE_APP_ONMICRO_DFU)
    (appm_add_svc_func_t)app_onmicro_dfu_add_service,
    #endif //(BLE_APP_ONMICRO_DFU)
};
```

4)添加服务init（app.c）

```
void appm_init(void)
{
    // Reset the application manager environment
    memset(&app_env, 0, sizeof(app_env));

    // Create APP task
    ke_task_create(TASK_APP, &TASK_DESC_APP);
```

```
void appm_init(void)
{
    // Reset the application manager environment
    memset(&app_env, 0, sizeof(app_env));

    // Create APP task
    ke_task_create(TASK_APP, &TASK_DESC_APP);

    // Initialize Task state
    ke_state_set(TASK_APP, APPM_INIT);

    // Get the Device Name to add in the Advertising Data (Default one or NVDS one)
    #if (NVDS_SUPPORT)
    {
    #if (NVDS_SUPPORT)
    {
    /*---------------------------------------------------
    // load device information:
    #if (BLE_APP_SEC)
    #if (BLE_APP_DIS)
    #if (BLE_APP_BATT)
    #if (BLE_APP_HID)
    #if (BLE_TSPP_SERVER)
    app_tspp_server_init();
    #endif //BLE_TSPP_SERVER
    #if (BLE_SIMPLE_SERVER)
    app_simple_server_init();
    #endif //BLE_SIMPLE_SERVER
```

5)添加服务回调（app.c）

```
void appm_reg_svc_itf(void)
{
    struct prf_itf_pair itf_pair[] = {
        #if (BLE_APP_DIS)
        { TASK_ID_DISS, diss_prf_itf_get() },
        #endif //(BLE_APP_DIS)
        #if (BLE_APP_BATT)
        { TASK_ID_BASS, bass_prf_itf_get() },
        #endif //(BLE_APP_BATT)
        #if (BLE_APP_HID)
        { TASK_ID_HOGPD, hogpd_prf_itf_get() },
        #endif //(BLE_APP_HID)
        #if (BLE_TSPP_SERVER)
        { TASK_ID_TSPPS, tspp_server_prf_itf_get() },
        #endif //(BLE_TSPP_SERVER)
        #if (BLE_SIMPLE_SERVER)
        { TASK_ID_SIMPLE_SERVER, simple_server_prf_itf_get() },
        #endif //(BLE_SIMPLE_SERVER)
    };
    prf_itf_register(itf_pair, sizeof(itf_pair)/sizeof(itf_pair[0]));
}
```

6)添加消息处理链接（app_task.c）

```
799 static int appm_msg_handler(ke_msg_id_t const msgid,
800                              void *param,
801                              ke_task_id_t const dest_id,
802                              ke_task_id_t const src_id)
803 {
804     // Retrieve identifier of the task from received message
805     ke_task_id_t src_task_id = MSG_T(msgid);
806     // Message policy
807     uint8_t msg_pol = KE_MSG_CONSUMED;
808
809     switch (src_task_id)
810     {
```

```
799 static int appm_msg_handler(ke_msg_id_t const msgid,
800                               void *param,
801                               ke_task_id_t const dest_id,
802                               ke_task_id_t const src_id)
803 {
804     // Retrieve identifier of the task from received message
805     ke_task_id_t src_task_id = MSG_T(msgid);
806     // Message policy
807     uint8_t msg_pol = KE_MSG_CONSUMED;
808
809     switch (src_task_id)
810     {
811         case (TASK_ID_GAPC):
812         {
824         case (TASK_ID_GATTC):
825         {
829         #if (BLE_APP_DIS)
837         #if (BLE_SIMPLE_SERVER)
838         case (TASK_ID_SIMPLE_SERVER):
839         {
840             // Call the Audio Mode 0 Module
841             msg_pol = app_get_handler(&app_simple_server_handlers, msgid, param, src_id);
842         } break;
843         #endif //(BLE_SIMPLE_SERVER)
```

7)连接后，服务使能（app_task.c）

```
555 static int gapc_connection_req_ind_handler(ke_msg_id_t const msgid,
556                                              struct gapc_connection_req_ind const *param,
557                                              ke_task_id_t const dest_id,
558                                              ke_task_id_t const src_id)
559 {
560     app_env.conidx = KE_IDX_GET(src_id);
561     #if(BLE_APP_SEC)
565   //log_debug("Device type(%d) conidx(%d) connected, ", param->peer_addr_type, app_env.conidx);
566   //log_debug_array_ex("ADDR", &param->peer_addr, 6);
567     // Check if the received Connection Handle was valid
568     if (app_env.conidx != GAP_INVALID_CONIDX)
569     {
570         // Retrieve the connection info from the parameters
571         app_env.conhdl = param->conhdl;
572
573         // Send connection confirmation
574         struct gapc_connection_cfm *cfm = KE_MSG_ALLOC(GAPC_CONNECTION_CFM,
578         #if(BLE_APP_SEC)
583         // Send the message
584         ke_msg_send(cfm);
585
586         /*----------------------------------------------------------------
590         #if (BLE_APP_BATT)
595         #if (BLE_APP_HID)
600         #if (BLE_TSPP_SERVER)
604         #if (BLE_SIMPLE_SERVER)
605         // Enable TSPP_SERVER Service
606         app_simple_server_enable_prf(app_env.conidx);
607         #endif //(BLE_SIMPLE_SERVER)
608         // We are now in connected State
609         ke_state_set(dest_id, APPM_CONNECTED);
610         #if (BLE_APP_SEC)
611         if (is_bond)
612         {
613             // Ask for the peer device to either start encryption
614             //app_sec_send_security_req(app_env.conidx);
615         }
616         #endif // (BLE_APP_SEC)
617         APP_HANDLER(gap_connected, msgid, param, dest_id, src_id); // APP layer Callback
618     }
```

4.按照需求修改对应的服务UUID以及属性特征

比如实现以下私有服务：

The log is empty.

Unknown Service
UUID: a6ed0401-d344-460a-8075-b9e8ec90d71b
PRIMARY SERVICE

Unknown Characteristic
UUID: a6ed0402-d344-460a-8075-b9e8ec90d71b
Properties: NOTIFY
Descriptors:
Client Characteristic Configuration
UUID: 0x2902
Value: Notifications enabled

Unknown Characteristic
UUID: a6ed0403-d344-460a-8075-b9e8ec90d71b
Properties: WRITE NO RESPONSE

Unknown Characteristic
UUID: a6ed0404-d344-460a-8075-b9e8ec90d71b
Properties: INDICATE, WRITE NO RESPONSE
Descriptors:
Client Characteristic Configuration
UUID: 0x2902
Value: Indications enabled

1)添加服务UUID

main.c | app_simple_server.c | simple_server.c | simple_server.h

```
22  * @{
23  ****************************************************************************************
24  */
25
26 /*
27  * INCLUDE FILES
28  ****************************************************************************************
29  */
30 #include "rwip_config.h"
31 #include "simple_server_task.h"
32 #include "prf_types.h"
33 #include "prf.h"
34
35 /*
36  * DEFINES
37  ****************************************************************************************
38  */
39
40 #define ATT_SVC_SIMPLE_SERVER_SERVICE {0x1B, 0xD7, 0x90, 0xec, 0xE8, 0xB9, 0x75, 0x80, 0x0A, 0x46, 0x44, 0xD3, 0x01, 0x04, 0xED, 0xAE}
41
42 #define ATT_SVC_SIMPLE_SERVER_CHAC1   {0x1B, 0xD7, 0x90, 0xec, 0xE8, 0xB9, 0x75, 0x80, 0x0A, 0x46, 0x44, 0xD3, 0x02, 0x04, 0xED, 0xAE}
43
44 #define ATT_SVC_SIMPLE_SERVER_CHAC2   {0x1B, 0xD7, 0x90, 0xec, 0xE8, 0xB9, 0x75, 0x80, 0x0A, 0x46, 0x44, 0xD3, 0x03, 0x04, 0xED, 0xAE}
45
46 #define ATT_SVC_SIMPLE_SERVER_CHAC3   {0x1B, 0xD7, 0x90, 0xec, 0xE8, 0xB9, 0x75, 0x80, 0x0A, 0x46, 0x44, 0xD3, 0x04, 0x04, 0xED, 0xAE}
47
```

2）修改属性列表

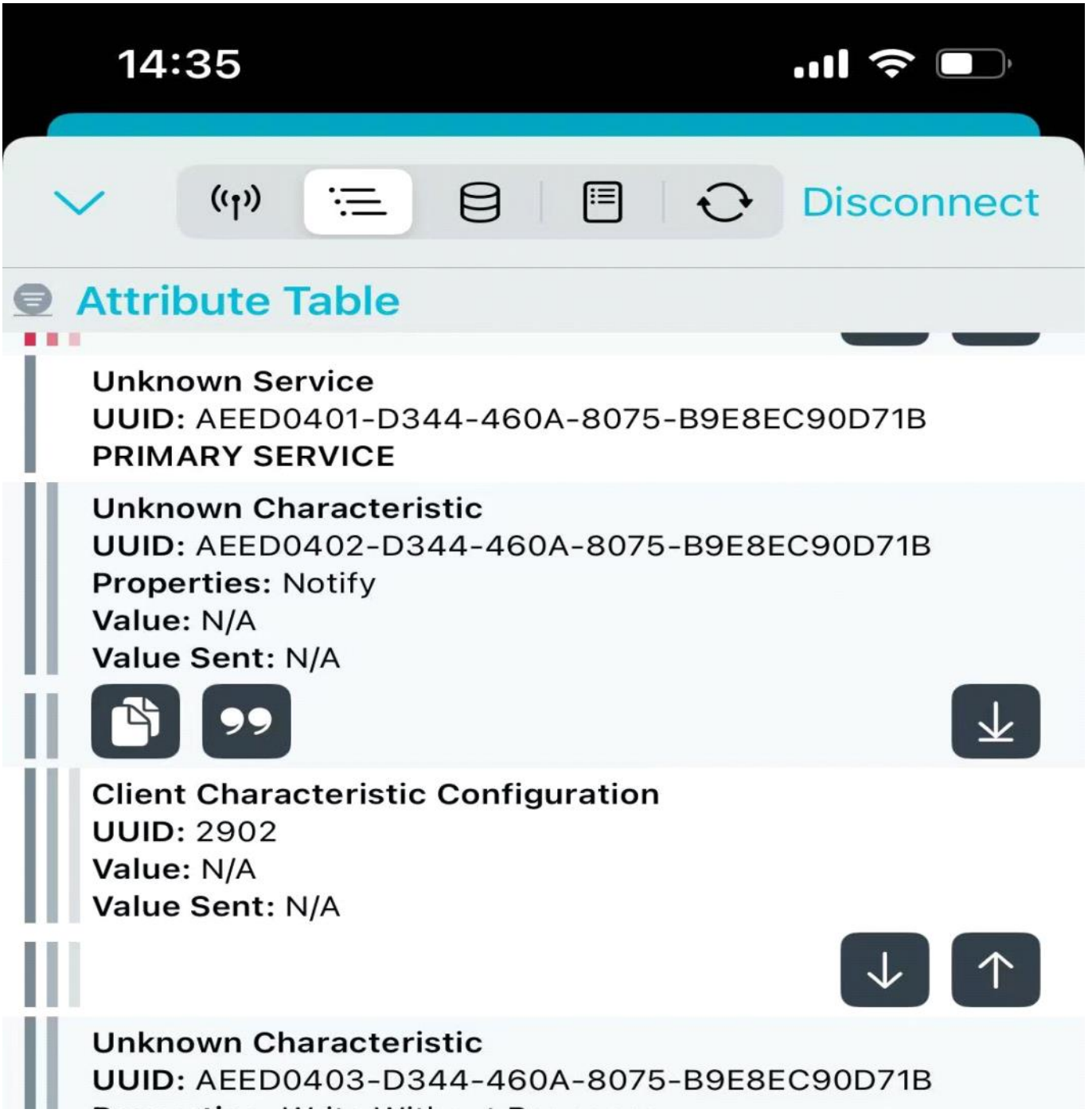
```
/// SIMPLE_SERVER Service Attributes Indexes
enum
{
    SIMPLE_SERVER_IDX_SVC,

    SIMPLE_SERVER_IDX_DEMO_CHAR1,
    SIMPLE_SERVER_IDX_DEMO_VAL1,
    SIMPLE_SERVER_IDX_DEMO_NTF_CFG,

    SIMPLE_SERVER_IDX_DEMO_CHAR2,
    SIMPLE_SERVER_IDX_DEMO_VAL2,

    SIMPLE_SERVER_IDX_DEMO_CHAR3,
    SIMPLE_SERVER_IDX_DEMO_VAL3,
    SIMPLE_SERVER_IDX_DEMO_IND_CFG,

    SIMPLE_SERVER_IDX_NB,
};
```

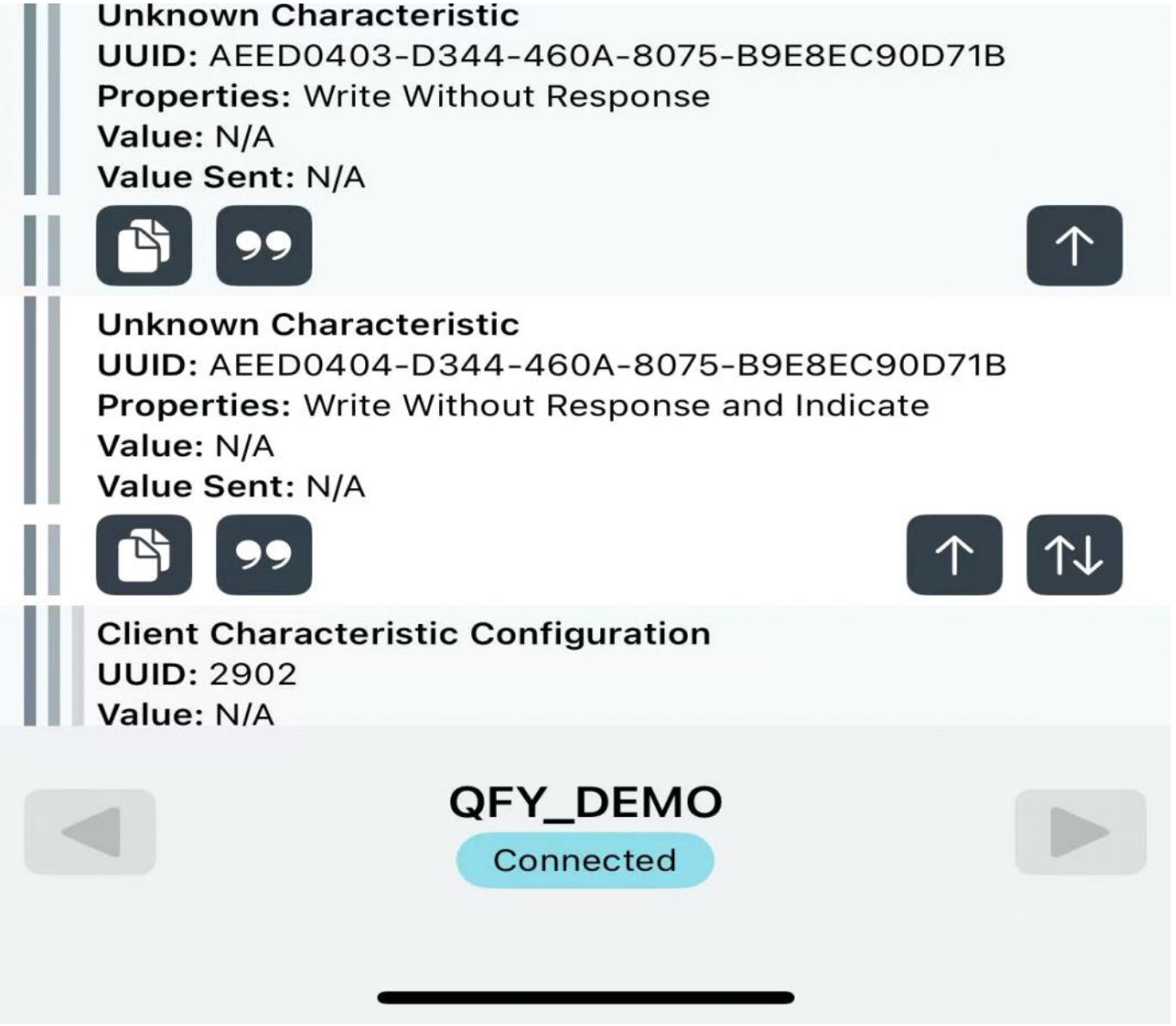
main.c | app_simple_server.c | simple_server.c

```
37  */
38 /// Full SIMPLE_SERVER Database Description - Used to add attributes into the database
39 const struct attm_desc_128 simple_server_att_db[SIMPLE_SERVER_IDX_NB] =
40 {
41     // Service Declaration
42     [SIMPLE_SERVER_IDX_SVC]              =   {ATT_16_TO_128_ARRAY(ATT_DECL_PRIMARY_SERVICE),  PERM(RD, ENABLE), 0, 0},
```

```
main.c | app_simple_server.c | simple_server.c
 */
/// Full SIMPLE_SERVER Database Description - Used to add attributes into the database
const struct attm_desc_128 simple_server_att_db[SIMPLE_SERVER_IDX_NB] =
{
    // Service Declaration
    [SIMPLE_SERVER_IDX_SVC]              =   {ATT_16_TO_128_ARRAY(ATT_DECL_PRIMARY_SERVICE),  PERM(RD, ENABLE), 0, 0},

    // Characteristic Declaration
    [SIMPLE_SERVER_IDX_DEMO_CHAR1]       =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC),   PERM(RD, ENABLE), 0, 0},
    // Characteristic Value
    [SIMPLE_SERVER_IDX_DEMO_VAL1]        =   {ATT_SVC_SIMPLE_SERVER_CHAC1, PERM(NTF, ENABLE), PERM(RI, ENABLE) | PERM(UUID_LEN, UUID_128),
                                             SIMPLE_SERVER_MAX_CHAC_LEN},
    // Characteristic - Client Characteristic Configuration Descriptor
    [SIMPLE_SERVER_IDX_DEMO_NTF_CFG]     =   {ATT_16_TO_128_ARRAY(ATT_DESC_CLIENT_CHAR_CFG),  PERM(RD, ENABLE)|PERM(WRITE_REQ, ENABLE), 0, 0},

    // Characteristic Declaration
    [SIMPLE_SERVER_IDX_DEMO_CHAR2]       =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC),   PERM(RD, ENABLE), 0, 0},
    // Characteristic Value
    [SIMPLE_SERVER_IDX_DEMO_VAL2]        =   {ATT_SVC_SIMPLE_SERVER_CHAC2, PERM(WRITE_COMMAND, ENABLE), PERM(RI, ENABLE) | PERM(UUID_LEN, UUID_128),
                                             SIMPLE_SERVER_MAX_CHAC_LEN},

     // Characteristic Declaration
    [SIMPLE_SERVER_IDX_DEMO_CHAR3]       =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC),   PERM(RD, ENABLE), 0, 0},
    // Characteristic Value
    [SIMPLE_SERVER_IDX_DEMO_VAL3]        =   {ATT_SVC_SIMPLE_SERVER_CHAC3, PERM(WRITE_COMMAND, ENABLE)|PERM(IND, ENABLE), PERM(RI, ENABLE) | PERM(UUID_LEN, UUID_128),
                                             SIMPLE_SERVER_MAX_CHAC_LEN},
    // Characteristic - Client Characteristic Configuration Descriptor
    [SIMPLE_SERVER_IDX_DEMO_IND_CFG]     =   {ATT_16_TO_128_ARRAY(ATT_DESC_CLIENT_CHAR_CFG),  PERM(RD, ENABLE)|PERM(WRITE_REQ, ENABLE), 0, 0},
};
```

3)扫描服务验证





最后一步就是数据接口TX/ RX实现：

RX：

这个write的接口，需要根据属性列表做响应的修改，可以参考tspp 服务

```
152 __STATIC int gattc_write_req_ind_handler(ke_msg_id_t const msgid, struct gattc_write_req_ind const *param,
153                                          ke_task_id_t const dest_id, ke_task_id_t const src_id)
154 {
155     struct gattc_write_cfm * cfm;
156     uint8_t att_idx = 0;
157     uint8_t conidx = KE_IDX_GET(src_id);
158     // retrieve handle information
159     uint8_t status = simple_server_get_att_idx(param->handle, &att_idx);
160     log_debug("%s handle:%d(idx:%d).\n", __func__, param->handle, att_idx);
161 
162     // If the attribute has been found, status is GAP_ERR_NO_ERROR
163     if (status == GAP_ERR_NO_ERROR)
164     {
165         struct simple_server_env_tag* simple_server_env = PRF_ENV_GET(SIMPLE_SERVER, simple_server);
166 
167         // Only update configuration if value for stop or notification enable
168         if (att_idx == SIMPLE_SERVER_IDX_DEMO_NTF_CFG)
169         {
186         // Only update configuration if value for stop or notification enable
187         if (att_idx == SIMPLE_SERVER_IDX_DEMO_IND_CFG)
188         {
189             // Extract value before check
190             uint16_t ntf_cfg = co_read16p(&param->value[0]);
191             if((ntf_cfg == PRF_CLI_STOP_NTFIND) || (ntf_cfg == PRF_CLI_START_NTF) || (ntf_cfg == PRF_CLI_START_IND))
192             {
193                 // Conserve information in environment
```

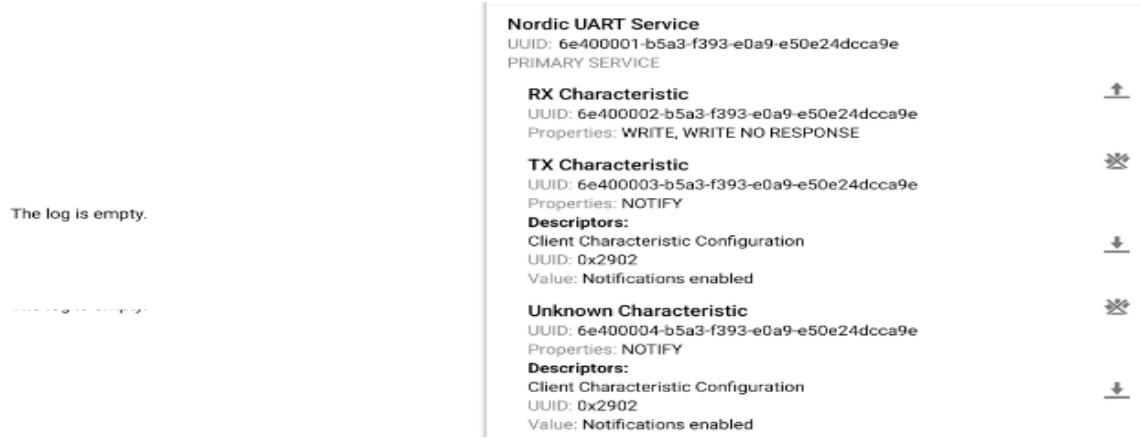
```
190             uint16_t ntf_cfg = co_read16p(&param->value[0]);
191             if((ntf_cfg == PRF_CLI_STOP_NTFIND) || (ntf_cfg == PRF_CLI_START_NTF) || (ntf_cfg == PRF_CLI_START_IND))
192             {
193                 // Conserve information in environment
194                 simple_server_env->ind_cfg[conidx] = ntf_cfg;
195             }
196             // Inform APP of configuration change
197             struct simple_server_demo_ntf_cfg_ind * ind = KE_MSG_ALLOC(SIMPLE_SERVER_IND_CFG_IND,
200             ind->conidx = conidx;
201             ind->ntf_cfg = simple_server_env->ind_cfg[conidx];
202
203             ke_msg_send(ind);
204         }
205         else if (att_idx == SIMPLE_SERVER_IDX_DEMO_VAL2 || att_idx == SIMPLE_SERVER_IDX_DEMO_VAL3)
206         {
```

TX：

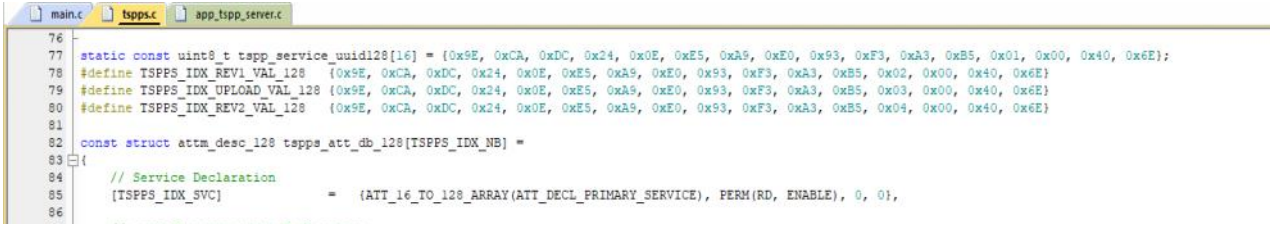
```
void app_simple_server_send_data(uint8_t conidx, uint8_t* pdata, uint16_t len)
{
    // Allocate the message
    uint16_t att_handle = simple_server_get_att_handle(SIMPLE_SERVER_IDX_DEMO_VAL1);
    struct simple_server_env_tag* simple_server_env = PRF_ENV_GET(SIMPLE_SERVER, simple_server);
    struct gattc_send_evt_cmd * req = KE_MSG_ALLOC_DYN(GATTC_SEND_EVT_CMD,
            KE_BUILD_ID(prf_get_task_from_id(TASK_ID_SIMPLE_SERVER), conidx),
            KE_BUILD_ID(TASK_APP, conidx),
            gattc_send_evt_cmd,
            len);
    // Fill in the parameter structure
    if(simple_server_env->ntf_cfg[conidx] == PRF_CLI_START_NTF){
        req->operation = GATTC_NOTIFY;
    }else{
        req->operation = GATTC_INDICATE;
    }
    req->handle    = att_handle;
    req->length = len;
    memcpy(req->value, pdata, len);
    // Send the message
    ke_msg_send(req);
}
```

具体的数据流向，还需要添加log，看看是不是符合预期。

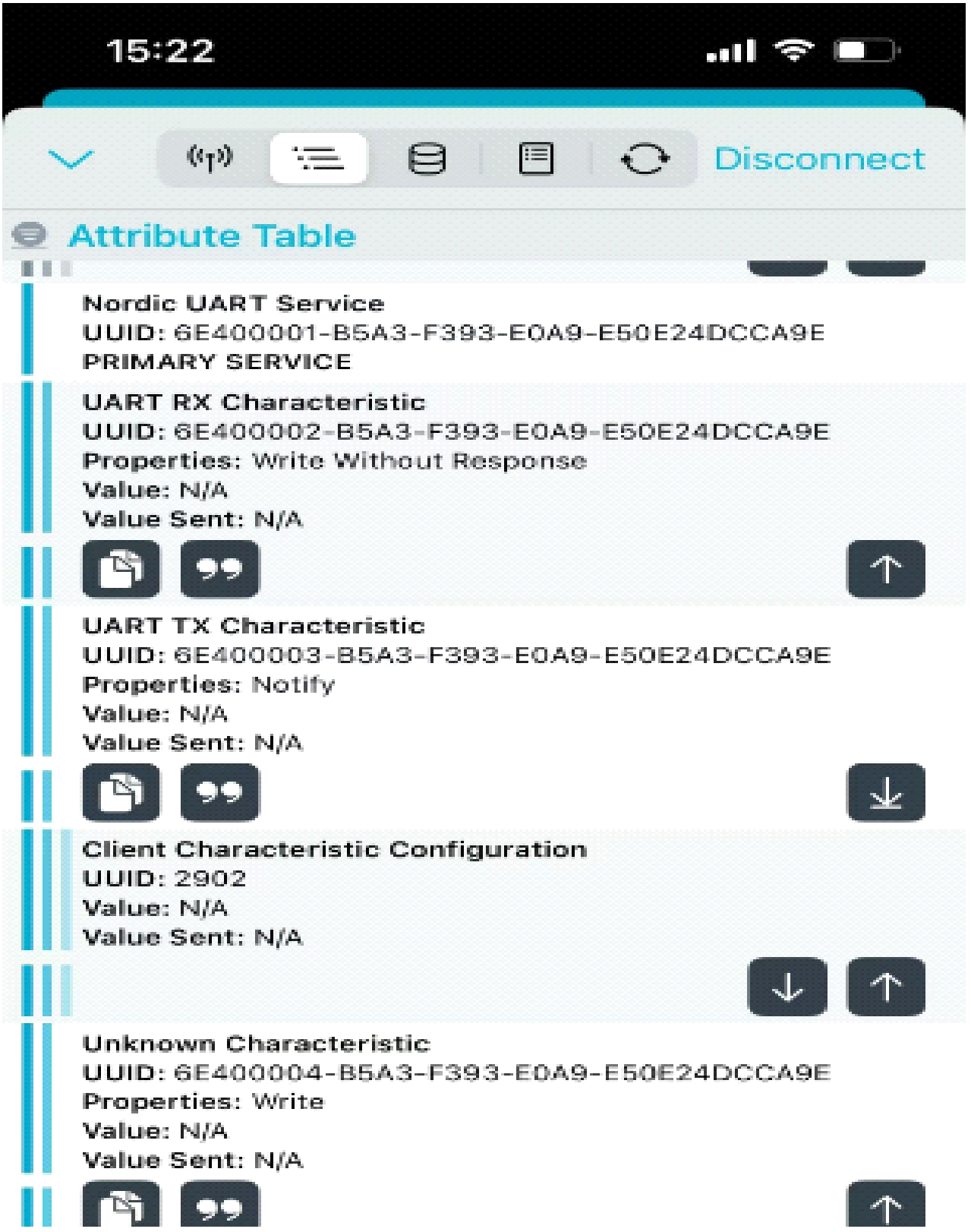
案例2：添加nordic UART 服务

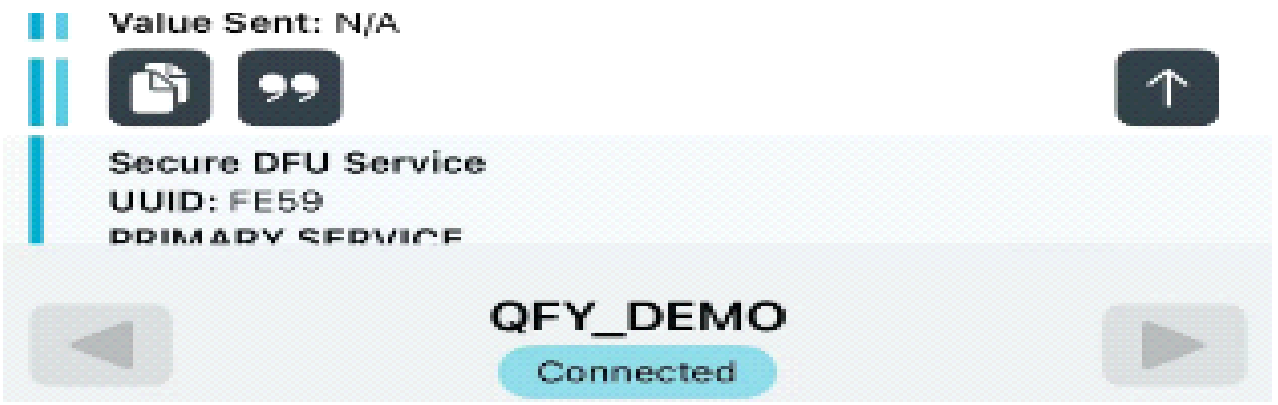


前面的步骤都省略，直接在把tspp改成nordic uart server



```
const struct attm_desc_128 tspps_att_db_128[TSPPS_IDX_NB] =
{
    // Service Declaration
    [TSPPS_IDX_SVC]             =   {ATT_16_TO_128_ARRAY(ATT_DECL_PRIMARY_SERVICE), PERM(RD, ENABLE), 0, 0},

    // rev1 Characteristic Declaration
    [TSPPS_IDX_REV1_CHAR]       =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC), PERM(RD, ENABLE), 0, 0},
    // rev1 Characteristic Value
    [TSPPS_IDX_REV1_VAL]        =   {TSPPS_IDX_REV1_VAL_128, PERM(WRITE_COMMAND, ENABLE), PERM_VAL(UUID_LEN, PERM_UUID_128), TSPPS_UPLOAD_MAX_LEN},

    // upload Characteristic Declaration
    [TSPPS_IDX_UPLOAD_CHAR]     =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC), PERM(RD, ENABLE), 0, 0},
    // upload Characteristic Value
    [TSPPS_IDX_UPLOAD_VAL]      =   {TSPPS_IDX_UPLOAD_VAL_128, PERM(NTF, ENABLE), PERM(RI, ENABLE)|PERM_VAL(UUID_LEN, PERM_UUID_128), TSPPS_UPLOAD_MAX_LEN},
    // upload Characteristic - Client Characteristic Configuration Descriptor
    [TSPPS_IDX_UPLOAD_NTF_CFG]  =   {ATT_16_TO_128_ARRAY(ATT_DESC_CLIENT_CHAR_CFG), PERM(RD, ENABLE) | PERM(WRITE_REQ, ENABLE), 0, 0},

    // rev2 Characteristic Declaration
    [TSPPS_IDX_REV2_CHAR]       =   {ATT_16_TO_128_ARRAY(ATT_DECL_CHARACTERISTIC), PERM(RD, ENABLE), 0, 0},
    // rev2 Characteristic Value
    [TSPPS_IDX_REV2_VAL]        =   {TSPPS_IDX_REV2_VAL_128, PERM(WRITE_REQ, ENABLE), PERM_VAL(UUID_LEN, PERM_UUID_128), TSPPS_UPLOAD_MAX_LEN},
};
```




Value Sent: N/A
Secure DFU Service
UUID: FE59
QFY_DEMO
Connected