YC31xx Bluetooth 应用说明

V1.2

Yichip Microelectronics

Revision History

Version	Date	Author	Description
V1.0	2020-02-14	Zhiteng.Yi	Initial version
V1.1	2020-03-01	Zhiteng.Yi	Add lib description
V1.2	2020-06-30	Kun.chen	Add BNEP

Confidentiality Level:

confidential

目录

1 文档说明 5
1.1 编写目的 5
1.2 适用范围 5
1.3 蓝牙应用 5
2 结构体说明 5
2.1 HCI_TypeDef 5
2.2 BT_BufTypeDef 6
3 库函数说明 6
3.1 BT_SetBtAddr 6
3.2 BT_SetBleAddr 6
3.3 BT_setBtName 7
3.4 BT_SetBleName 7
3.5 BT_setVisbility 7
3.6 BT_SendSppData 8
3.7 BT_SendBleData 8
3.8 Bt_GetBtStatus 8
3.9 BT_setParingMode 9
3.10 BT_SetPincode 9
3.11 BT_GetVersion 9
3.12 BT_BtDisconnect 9
3.13 BT_setNVRAM 10
3.14 BT_EnterSleepMode 10
3.15 BT_ConfirmGkey 10
3.16 BT_SetSppFlowcontrol 11
3.17 BT_PasskeyEntry 11
3.18 BT_SetLEParing 11
3.19 BT_SetLEAdvData 12
3.20 BT_SetLEScanData 12
3.21 BT_SetLESendConnUpdate 12
3.22 BT_SetLEAdvParm 13
3.23 BT_RejectJustWork 13
3.24 BT_Set_FixedPasskey 13
3.25 BT_BleDisconnect 13
3.26 BT_SetCOD 14
3.27 BT_SetTxPower 14
3.28 BT_DeleteService 14
3.29 BT_AddBleService 15
3.30 BT_AddBleCharacteristic 15
3.31 BT_ReadBTData 16
3.32 BT_GetEventOpcode 16
3.33 BT_Init 16
3.34 BT_DnsReq 16

3.35 BT_ConnectBnep 17
3.36 BT_disconnectBnep 17
3.37 BT_ConnectTcp 17
3.38 BT_BnepSendTcpData 18
3.39 BT_BnepSendTcpBigData 18
3.40 BT_BnepSendUdpData 18
3.41 BT_disconnectTcp 19
4 Demo 示例 19
4.1 BT&BLE 19
4.2 BNEP 22

1 文档说明

1.1 编写目的

为使用 BlueTooth 相关 Demo 及 BlueTooth 库函数提供指南

1.2 适用范围

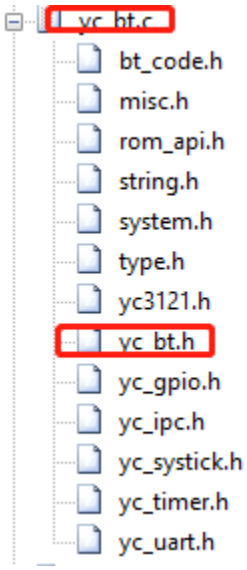
YC31xx 系列芯片

1.3 蓝牙应用

蓝牙可以用中断和非中断两种方式接收数据。本文档对蓝牙 sdk 接口 yc_bt.h 和相关蓝牙 demo 使用提供说明。关于蓝牙详细透传指令请参照易兆 HCI 指令文档。

Blue Tooth Demo 路径为 ModuleDemo\ BlueTooth

蓝牙库文件为如下图 yc_bt.c 与 yc_bt.h,路径为 Librarier\sdk



BlueTooth Demo 中共有两个个示例 Demo，依次做简要说明

BT&BLE demo 示例实现了 SPP 及 BLE 的使用。

BNEP demo 示例 BNEP 的连接,域名的解析,TCP 的连接,TCP 数据的发送, UDP 数据的发送以及 TCP 和 BNEP 断开。

2 结构体说明

2.1 HCI_TypeDef

说明：蓝牙 HCI 指令结构体

元素名称	类型	说明	参数项

type	uint8_t	包类型	01（cmd） 02（event）
opcode	Uint8_t	操作码	详见 bt.h 中 cmd 和 event 中宏定义
Datalen	uint8_t	内容长度	最大 255
P_data	uint8_t *	内容	

2.2 BT_BufTypeDef

说明：蓝牙数据缓冲 buf 结构体，上层应用单包数据大小大于默认 BufSize 时需要修改 BufSize，以免 buf 溢出。

元素名称	类型	说明	参数项
pBuf	uint8_t*	接收蓝牙返回数据的 buff	
BufSize	int	Buff 的大小	
count	int	当前指针数值	
ReadIndex	int	读指针	
WriteIndex	int	写指针	

3 库函数说明

3.1 BT_SetBtAddr

函数原型：Boolean BT_SetBtAddr(uint8_t * bt_addr);
说明： 设置 BT3.0(SPP) MAC 地址

参数	方向	说明
uint8_t * bt_addr	IN	BT3.0(SPP) MAC 地址

返回值	说明
TRUE	成功
FALSE	失败

3.2 BT_SetBleAddr

函数原型：Boolean BT_SetBleAddr(uint8_t * bt_addr);
说明： 设置 BLE MAC 地址

参数	方向	说明
uint8_t * bt_addr	IN	BLE MAC 地址

返回值	说明
TRUE	成功
FALSE	失败

3.3 BT_setBtName

函数原型：Boolean BT_SetBtName(uint8_t * bt_name, uint16_t name_len);

说明： 设置 BT3.0(SPP)名称

参数	方向	说明
uint8_t * bt_name	IN	BT3.0(SPP)名称，bt_name 字符串最后一个字节不能为'\0',否则某些手机可能会显示有问题
uint16_t name_len	IN	数据长度

返回值	说明
TRUE	成功
FALSE	失败

3.4 BT_SetBleName

函数原型：Boolean BT_SetBleName(uint8_t* ble_name, uint16_t name_len);

说明： 设置 BLE 名称

参数	方向	说明
uint8_t* ble_name	IN	BLE 名称，ble_name 字符串最后一个字节不能为'\0',否则某些手机可能会显示有问题
uint16_t name_len	IN	数据长度

返回值	说明
TRUE	成功
FALSE	失败

3.5 BT_setVisbility

函数原型 Boolean BT_SetVisibility(Boolean bt_discoverable, Boolean bt_connectability, Boolean ble_discoverable);

说明： 设置可 SPP/BLE 发现.

参数	方向	说明
Boolean bt_discoverable	IN	TRUE OR FALSE
Boolean bt_connectability	IN	TRUE OR FALSE
Boolean ble_discoverable	IN	TRUE OR FALSE

返回值	说明
TRUE	设置成功
FALSE	设置失败

3.6 BT_SendSppData

函数原型：Boolean BT_SendSppData(uint8_t * spp_data, uint16_t DataLen);
说明：　　发送 Spp 数据.

参数	方向	说明
uint8_t * spp_data	IN	待发送 Spp_data
uint16_t　DataLen	IN	待发送数据长度，长度不能大于 255

返回值	说明
TRUE	发送 spp 数据成功
FALSE	发送 spp 数据失败

3.7 BT_SendBleData

函数原型：BT_SendBleData(uint8_t * ble_data, uint16_t DataLen);
说明：　　发送 Ble 数据

参数	方向	说明
uint8_t * ble_data	IN	待发送 Ble_data（包含一字节 handle）
uint16_t DataLen	IN	待发送数据长度，长度不能大于 255

返回值	说明
TRUE	发送 ble 数据成功
FALSE	发送 ble 数据失败

3.8 Bt_GetBtStatus

函数原型：uint8_t Bt_GetBtStatus(void);
说明：　　获取蓝牙状态

参数	方向	说明
none		

返回值	说明
uint8_t	bit0:BT 3.0 Can be discover bit1:BT 3.0 Can be connect bit2:BT 4.0 Can be discover and connect bit4:BT 3.0 connected bit5:BT 4.0 connected bit7:get status timer out

3.9 BT_setParingMode

函数原型：Boolean BT_SetParingMode(uint8_t mode);
说明：　设置 SPP 配对模式

参数	方向	说明
uint8_t　mode	IN	0x00:pincode 0x01:just work 0x02:passkey 0x03:confirm

返回值	说明
TRUE	成功
FALSE	失败

3.10 BT_SetPincode

函数原型：Boolean BT_SetPincode(uint8_t* Pincode,uint8_t len);
说明：　设置 pincode 模式下的配对码

参数	方向	说明
uint8_t* Pincode	IN	配对码
uint8_t len	IN	长度取值范围 0x01~0x10

返回值	说明
TRUE	成功
FALSE	失败

3.11 BT_GetVersion

函数原型：uint32_t BT_GetVersion(void);
说明：　获取蓝牙固件版本号

参数	方向	说明
None		

返回值	说明
非零	版本号
0	失败

3.12 BT_BtDisconnect

函数原型：Boolean BT_BtDisconnect(void);
说明：　断开 BT3.0（SPP）连接

参数	方向	说明
None		

返回值	说明
TRUE	成功
FALSE	失败

3.13 BT_setNVRAM

函数原型：Boolean BT_SetNVRAM(uint8_t * NvData,int len);
说明：　　设置配对信息 NVRRAM，每次初始化蓝牙后下发

参数	方向	说明
uint8_t * NvData	IN	发送 NVRAM 数据
int len	IN	170（包含 5 组配对信息）

返回值	说明
TRUE	发送 NVRAM 数据成功
FALSE	发送 NVRAM 数据失败

3.14 BT_EnterSleepMode

函数原型：Boolean BT_EnterSleepMode(void);
说明：　　蓝牙进入休眠模式

参数	方向	说明
None		

返回值	说明
TRUE	成功
FALSE	失败

3.15 BT_ConfirmGkey

函数原型：Boolean BT_ConfirmGkey(uint8_t isMatching);
说明：　　收到 ConfirmGkey 事件后应答 ConfirmGkey 是否匹配

参数	方向	说明
uint8_t isMatching	IN	0x00: key 匹配 0x01: key 不匹配

返回值	说明
TRUE	成功

FALSE	失败

3.16 BT_SetSppFlowcontrol

函数原型：Boolean BT_SetSppFlowcontrol(uint8_t packetNum);
说明：　　下发 SPP 流控授权数量，每接收一包 SPP 数据后减一，为 0 后暂停 SPP 收包

参数	方向	说明
uint8_t packetNum	IN	授权包数

返回值	说明
TRUE	成功
FALSE	失败

3.17 BT_PasskeyEntry

函数原型：Boolean BT_PasskeyEntry(uint8_t *key_data);
说明：　　下发 passkey 密钥

参数	方向	说明
uint8_t *key_data	IN	Passkey，必须为 4 字节

返回值	说明
TRUE	成功
FALSE	失败

3.18 BT_SetLEParing

函数原型：Boolean BT_SetLEParing(uint8_t mode);
说明：　　设置 BLE 配对模式

参数	方向	说明
uint8_t mode	IN	0x00:none 0x01:just work 0x02:pass key 0x81:secure connect just work 0x82:secure connect numeric 0x83:secure connect pass key

返回值	说明
TRUE	成功
FALSE	失败

3.19 BT_SetLEAdvData

函数原型：Boolean BT_SetLEAdvData(uint8_t*　adv_data, int DataLen);
说明：　　设置 BLE adv 数据

参数	方向	说明
uint8_t*　adv_data	IN	adv 数据
int DataLen	IN	长度必须为 0x1f

返回值	说明
TRUE	成功
FALSE	失败

3.20 BT_SetLEScanData

函数原型：Boolean BT_SetLEScanData(uint8_t* scan_data, int DataLen);
说明：　　设置 BLE scan 数据

参数	方向	说明
uint8_t*　scan_data	IN	scan 数据
int DataLen	IN	长度小于 20

返回值	说明
TRUE	成功
FALSE	失败

3.21 BT_SetLESendConnUpdate

函数原型：Boolean BT_SetLESendConnUpdate(uint8_t *data,int len);
说明：　　更新 BLE 连接参数

参数	方向	说明
uint8_t *data	IN	byte0-byte1:min connect interval byte2-byte3:max connect interval byte4-byte5:Slave　latency byte6-byte7:Connection Supervision Timeout
int len	IN	长度必须为 8

返回值	说明
TRUE	成功
FALSE	失败

3.22 BT_SetLEAdvParm

函数原型：Boolean BT_SetLEAdvParm(uint8_t*data,int DataLen);
说明：　　设置 BLE adv 参数

参数	方向	说明
uint8_t *data	IN	Adv 参数
int DataLen	IN	长度必须为 2

返回值	说明
TRUE	成功
FALSE	失败

3.23 BT_RejectJustWork

函数原型：Boolean BT_RejectJustWork(uint8_t justwork);
说明：　　禁用 just work 模式

参数	方向	说明
uint8_t justwork	IN	0:允许 just work 配对 1:禁止 just work 配对

返回值	说明
TRUE	成功
FALSE	失败

3.24 BT_Set_FixedPasskey

函数原型：Boolean BT_Set_FixedPasskey(uint8_t* key);
说明：　　设置 passkey

参数	方向	说明
uint8_t* key	IN	Byte0：0x00：随机 Passkey;0x01：自定义 Passkey Byte4~Byte7：passkey

返回值	说明
TRUE	成功
FALSE	失败

3.25 BT_BleDisconnect

函数原型：Boolean BT_BleDisconnect(void);
说明：　　断开 BLE 连接

参数	方向	说明
None		

返回值	说明
TRUE	成功
FALSE	失败

3.26 BT_SetCOD

函数原型：Boolean BT_SetCOD(uint8_t* bt_cod);
说明：　　设置 SPP 设备类型

参数	方向	说明
uint8_t* bt_cod	IN	SPP 设备类型代码(3byte)

返回值	说明
TRUE	成功
FALSE	失败

3.27 BT_SetTxPower

函数原型：Boolean BT_SetTxPower(uint8_t power);
说明：　　设置 TX 发射功率

参数	方向	说明
uint8_t power	IN	0:0db 1:3db 2:5db 3:-3db 4:-5db

返回值	说明
TRUE	成功
FALSE	失败

3.28 BT_DeleteService

函数原型：Boolean BT_DeleteService(void);
说明：　　删除 BLE 服务

参数	方向	说明
None		

返回值	说明

TRUE	成功
FALSE	失败

3.29 BT_AddBleService

函数原型：uint16_t BT_AddBleService(uint8_t* ble_service_uuid, uint16_t service_uuid_len);
说明：　添加 BLE 服务

参数	方向	说明
uint8_t* ble_service_uuid	IN	Byte0：uuid length(2 or 16) byte1-2(16): uuid
uint16_t service_uuid_len	IN	

返回值	说明
非零值	Service handle
0	失败

3.30 BT_AddBleCharacteristic

函数原型：uint16_t BT_AddBleCharacteristic(uint8_t* ble_Characteristic_uuid, uint16_t service_Characteristic_payload_len);
说明：　添加 BLE 特征

参数	方向	说明
uint8_t* ble_Characteristic_uuid	IN	byte0: characterisitic attribute bit0　Broadcast bit1　Read bit2　Write without Response bit3　Write bit4　Notify bit5　Indicate bit6　Authentication Signed Write bit7　Extended Properties byte1: characterisitic uuid length(2 or 16) byte2-3(17):characterisitic uuid byte4(18): write/read payload length(1--20) byte5(19)-x:write/read payload(default:00)
uint16_t service_Characteristic_payload_len	IN	

返回值	说明
非零值	Characteristic handle
0	失败

3.31 BT_ReadBTData

函数原型：int BT_ReadBTData(uint8_t* pbuf);
说明：　从缓冲 buf 中读取蓝牙数据

参数	方向	说明
uint8_t* pbuf	OUT	buf 指针（为防止越界，应分配 255byte）

返回值	说明
数据长度	

3.32 BT_GetEventOpcode

函数原型：int BT_GetEventOpcode(void);
说明：　获取缓冲 buf 中未取出数据的事件代码

参数	方向	说明
None		

返回值	说明
-1	无待处理数据
其他值	事件代码（event opcode）

3.33 BT_Init

函数原型：Boolean BT_Init(void);
说明：　初始化蓝牙模块

参数	方向	说明
None		

返回值	说明
TRUE	成功
FALSE	失败

3.34 BT_DnsReq

函数原型：Boolean BT_DnsReq(uint8_t *dns_data,int len);
说明：查询域名对应的 IP 地址,BNEP 连接状态下才能操作

参数	方向	说明
uint8_t *dns_data	IN	待查询的域名（www.baidu.com）
,int len	IN	数据长度（最大 62BYTE）

返回值	说明
TRUE	发送域名成功
FALSE	发送域名失败

3.35 BT_ConnectBnep

函数原型：Boolean BT_ConnectBnep(uint8_t *phone_mac_addr,int len);
说明：　　Bnep 连接，需手机上先进行配对才能连接

参数	方向	说明
uint8_t*phone_mac_addr	IN	Byte0~Byte5　手机的 mac 地址 Byte6~Byte21　Link key (MAC 地址和 link key 来源是 HCI_EVNET_NVRAM_RSP 内容为全 0 时，默认连接最后配对的手机，信息 存储在 NVRAM 中)
int len	IN	22BYTE

返回值	说明
TRUE	Bnep 连接成功
FALSE	Bnep 连接失败

3.36 BT_disconnectBnep

函数原型：Boolean BT_disconnectBnep();
说明：　　断开 bnep 的连接

参数	方向	说明
None		None

返回值	说明
TRUE	断开 bnep 连接成功
FALSE	断开 bnep 连接失败

3.37 BT_ConnectTcp

函数原型：Boolean BT_ConnectTcp(uint8_t *tcp_ip_addr,int len);
说明：　　tcp 的连接，BNEP 连接状态下才能操作

参数	方向	说明
uint8_t *tcp_ip_addr	IN	Byte0　TCP connect handle(0x00 or 0x01) Byte1~Byte4　IP 地址(192.168.1.1 is c0 a8 01 01) Byte5~Byte6　端口号(8888 is 22 b8)
int len	IN	数据长度

返回值	说明
TRUE	Tcp 连接成功

FALSE	Tcp 连接失败

3.38 BT_BnepSendTcpData

函数原型：Boolean BT_BnepSendTcpData(uint8_t *tcpdata,int len);
说明：　　发送 tcp 数据

参数	方向	说明
uint8_t *tcpdata	IN	Byte0　TCP connect handle Byte1~ByteN　想要发送的 tcp 数据
int len	IN	最大不超过 MAX_TCP_DATA_LEN+1

返回值	说明
TRUE	发送 tcp 数据成功
FALSE	发送 tcp 数据失败

3.39 BT_BnepSendTcpBigData

函数原型：Boolean BT_BnepSendTcpBigData(uint8_t *tcpdata,int len);
说明：　　发送 tcp 大包数据

参数	方向	说明
uint8_t *tcpdata	IN	Byte0　TCP connect handle Byte1~ByteN　想要发送的 tcp 数据
int len	IN	最大不超过 MAX_BIG_DATA_LEN+1

返回值	说明
TRUE	成功
FALSE	失败

3.40 BT_BnepSendUdpData

函数原型：Boolean BT_BnepSendTcpBigData(uint8_t *tcpdata,int len);
说明：　　发送 UDP 数据

参数	方向	说明
uint8_t * udpdata	IN	Byte0~Byte3:UDP remote IP(192.168.1.1 is c0 a8 01 01) Byte4~Byte5:UDP local port(8888 is 22 b8) Byte6~Byte7:UDP remote port(12345 is 30 39) Byte8~ByteN:The UDP data need to send(max 247 bytes)
int len	IN	最大不超过 255

返回值	说明
TRUE	成功

FALSE	失败

3.41 BT_disconnectTcp

函数原型：Boolean BT_disconnectTcp(uint8_t tcp_handle);
说明：　　断开 tcp 的连接

参数	方向	说明
uint8_t　tcp_handle	IN	TCP handle(0x00 or 0x01)

返回值	说明
TRUE	断开 tcp 连接成功
FALSE	断开 tcp 连接失败

4 Demo 示例

4.1 BT&BLE

```
SysTick_Config(CPU_MHZ/1000); //初始化系统滴答定时器（SDK 接口函数判断指令超时会用到）
BT_Init();                    //蓝牙初始化操作
enable_intr(INTR_BT);   //使能蓝牙中断


if(BT_SetBleName(bt_name,sizeof(bt_name)-1)==TRUE)     //设置 BLE 名称
    MyPrintf("SetBleName_suc ble name:%s\n",bt_name);
else
    MyPrintf("SetBleName_failed\n");

if(BT_SetBtName(bt_name,sizeof(bt_name)-1) == TRUE)//bt 与 ble 名字地址可以设置成一样
    MyPrintf("SetbtName_suc\n");
else
    MyPrintf("SetbtName_fail\n");

    if(BT_SetBleAddr(bt_addr) == TRUE)    //设置 BLE 地址
        MyPrintf("SetBleAddr_suc\n");
    else
        MyPrintf("SetBleAddr_fail\n");

    if(BT_SetBtAddr(bt_addr) == TRUE)    //设置 BT 地址
        MyPrintf("SetBtAddr_suc\n");
    else
```

```
        MyPrintf("SetBtAddr_fail\n");

    if(BT_SetParingMode(0x03) == TRUE)//设置配对模式为 confirmkey
        MyPrintf("set confirmkey mode success\n");
    else
        MyPrintf("set confirmkey mode failed\n");

    if(BT_SetCOD(bt_cod) == TRUE) //设置 COD
        MyPrintf("set COD sucess\n");
    else
        MyPrintf("set COD failed\n");


    if(BT_DeleteService() == TRUE) //   删除用户自定义服务
        MyPrintf("delete service sucess\n");
    else
        MyPrintf("delete service failed\n");

    temp_handle=BT_AddBleService(ble_service_uuid_lsps,sizeof(ble_service_uuid_lsps));
    if( temp_handle!= 0)   //增加服务 返回 handle 无需保存
        MyPrintf("add service sucess,handle=%04x\n",temp_handle);
    else
        MyPrintf("add service failed,return=%04x\n",temp_handle);

    ble_send_handle=BT_AddBleCharacteristic(ble_Characteristic_uuid_lsps_tx,sizeof(ble_Characteristic_uuid
_lsps_tx));
    if( ble_send_handle!= 0)   //增加服务特征   write 返回的 handle 需要保存，发数据使用
        MyPrintf("add Characteristic tx sucess,handle=%04x\n",ble_send_handle);
    else
        MyPrintf("add Characteristic tx failed,return=%04x\n",ble_send_handle);

    temp_handle=BT_AddBleCharacteristic(ble_Characteristic_uuid_lsps_rx,sizeof(ble_Characteristic_uuid_lsp
s_rx));
    if( temp_handle!= 0)
        MyPrintf("add Characteristic rx sucess;handle=%04x\n",temp_handle);
    else
        MyPrintf("add Characteristic rx failed,return=%04x\n",temp_handle);

    temp_handle=BT_AddBleCharacteristic(ble_Characteristic_uuid_flow_ctrl,sizeof(ble_Characteristic_uuid_fl
ow_ctrl));
    if( temp_handle!= 0)
        MyPrintf("add Characteristic flow_ctrl sucess;handle=%04x\n",temp_handle);
    else
        MyPrintf("add Characteristic flow_ctrl failed,return=%04x\n",temp_handle);
```

```
if(BT_SetVisibility(0x01,0x01,0x01) == TRUE)   //设置可发现
    MyPrintf("SetVisibility sucess\n");
else
    MyPrintf("SetVisibility failed\n");

MyPrintf("bt version=%x\n",BT_GetVersion());


memset(NvramData,0xff,170);
qspi_flash_read(IFLASH_NVRAM_ADDR,NvramData,NVRAM_LEN);
//nvram  包含 5 个设备信息

i=0;
if(BT_SetNVRAM(NvramData) == TRUE)
{
    MyPrintf("set nvram success:\n");
    while(i<170)
        MyPrintf("0x%02X ",NvramData[i++]);
    MyPrintf("\n");
}
else
    MyPrintf("set nvram failed\n");
```

4.2 BNEP

BNEP 操作流程及注意事项详见《BNEP 客户示意流程图.pdf》
BNEP demo 采用串口指令操作模式，实现的指令如下：

```
==================================
case 0: show menu
case 1: connect bnep(last pair phone)
case 2: connect tcp(bnep test server:139.224.56.87[12345])
case 3: dns connect tcp(www.baidu.com)
case 4: send tcp data
case 5: send tcp big packet data
case 6: send udp data
case 7: disconnect tcp(139.224.56.87)
case 8: disconnect tcp(www.baidu.com)
case 9: disconnect bnep
==================================
```

程序启动后会将蓝牙初始化为 BNEP 模式，并打印蓝牙名称，需根据名称在手机上完成配对，且手机开启蓝牙网络共享方可通过串口发送上述指令进行操作。

main.c 中的 void BT_Progress()函数为蓝牙事件及数据响应函数，在 main 函数中循环调用。

main.c 中的 void BT_IRQHandler()函数为蓝牙事件中断服务函数，代码如下，此函数中将蓝牙数据从 IPC 接口中取出，并使用 BT_ParseBTData()函数按照 HCI 指令格式进行解析。

```
void BT_IRQHandler()
{
    while(IPC_have_data())
    {
        #ifdef UART_TO_IPC
        if(TRUE==IPC_ReadBtData(&HCI_Rx))
        {
            UART_SendBuf(IPC_UART,(uint8_t*)&HCI_Rx,3);
            UART_SendBuf(IPC_UART,HCI_Rx.p_data,HCI_Rx.DataLen);
        }
        #else
        BT_ParseBTData();
        #endif
    }

    BT_CONFIG &= (~(1<<BT_INIT_FLAG));
}
```