

MG32x02z

BLE 库用户手册

Version: 1.12

目录

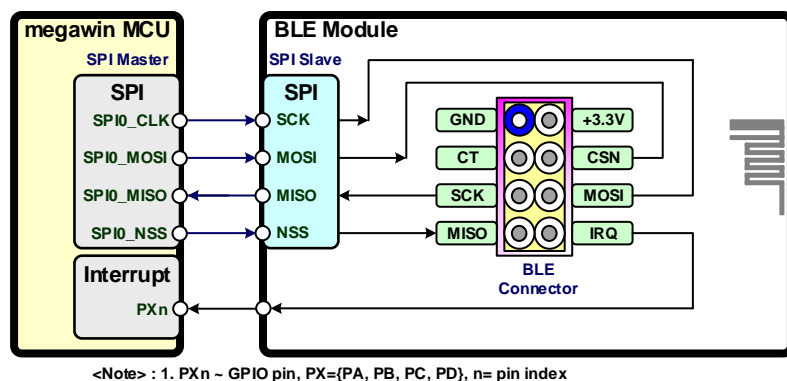
目录.....	2
1. 硬件连接.....	4
1.1. BLE 到 MCU 的 SPI 连接.....	4
1.2. BLE 模块实物图.....	4
1.3. 实际 BLE 连接至 MCU 实物图.....	4
2. BLE_SPI.c.....	6
2.1. 概述.....	6
2.2. 函数列表.....	6
2.3. 函数原型叙述.....	7
2.3.1. SPI0_Init.....	7
2.3.2. SPI_WriteBuf.....	7
2.3.3. SPI_ReadBuf.....	7
3. BSP.c.....	8
3.1. 概述.....	8
3.2. 函数列表.....	8
3.3. 函数原型叙述.....	9
3.3.1. BLE_EXIC_Init.....	9
3.3.2. EXINT1_IRQHandler.....	9
3.3.3. BLE_URT0_Init.....	9
3.3.4. IWDT_Init.....	9
3.3.5. BLE_SPI0_Init.....	10
3.3.6. BLE_LED_Init.....	10
3.3.7. LED_Flash.....	10
3.3.8. BSP_Init.....	10
3.3.9. IsIrqEnabled.....	11
3.3.10. IrqMcuGotoSleepAndWakeup.....	11
3.3.11. GetSysTickCount.....	11
3.3.12. SysTick_Handler.....	11
4. retarget.c.....	12
4.1. 概述.....	12
4.2. 函数列表.....	12
4.3. 函数原型叙述.....	13
4.3.1. URT_IRQHandler.....	13
4.3.2. moduleOutData.....	13
4.3.3. CheckComPortInData.....	13
4.3.4. UsrProcCallback.....	13
5. BLE MG126 Library.....	14
5.1. 概述.....	14
5.2. 接口函数列表.....	14
5.3. 接口函数说明.....	15
5.3.1. radio_initBle.....	15
5.3.2. radio_initBle_TO.....	15
5.3.3. ble_run_interrupt_start.....	15
5.3.4. SetBleIntRunningMode.....	15
5.3.5. SetLePinCode.....	15

5.3.6. ble_set_adv_type	15
5.3.7. ble_set_adv_data	16
5.3.8. ble_set_adv_rsp_data	16
5.3.9. ble_set_name	16
5.3.10. ble_set_interval	16
5.3.11. radio_standby	16
5.3.12. radio_resume	17
5.3.13. ble_set_wakeupdly	17
5.3.14. ble_set_adv_enableFlag	17
5.3.15. ble_set_role	17
5.3.16. ble_disconnect	17
5.3.17. ble_disc	18
5.3.18. ble_set_feature_supported	18
5.3.19. ble_set_md_enable	18
5.3.20. get_ble_version	18
5.3.21. GetFirmwareInfo	18
5.3.22. ser_write_rsp_pkt	18
5.3.23. att_notFd	19
5.3.24. att_ErrorFd_eCode	19
5.3.25. att_server_rdyByGrTypeRspDeviceInfo	19
5.3.26. att_server_rdyByGrTypeRspPrimaryService	20
5.3.27. att_server_rd	20
5.3.28. sconn_notifydata	20
5.3.29. sconn_indicationdata	21
5.3.30. SIG_ConnParaUpdateReq	21
5.3.31. sconn_GetConnInterval	21
5.3.32. GetRssiData	21
5.3.33. radio_setBleAddr	21
5.3.34. SetFixAdvChannel	21
5.3.35. test_carrier	22
5.3.36. test_SRRCCarrier	22
5.3.37. test_PRBS9	22
5.3.38. test_RX	22
5.4. 中断服务程序方式运行模式	22
5.4.1. 主要运行模式	22
5.4.2. SysTick IRQ 处理	23
5.4.3. BLE IRQ 处理	24
6. 修订历史	25

1. 硬件连接

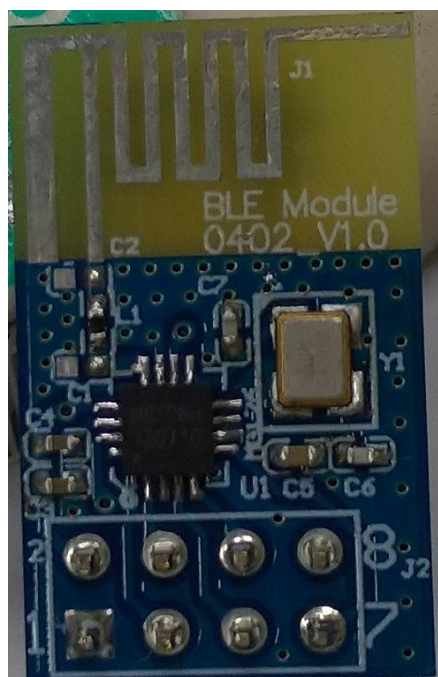
1.1. BLE 到 MCU 的 SPI 连接

下图展示了 BLE 到 MCU 的 SPI 连接。



1.2. BLE 模块实物图

下图展示了 BLE 模块实物图。

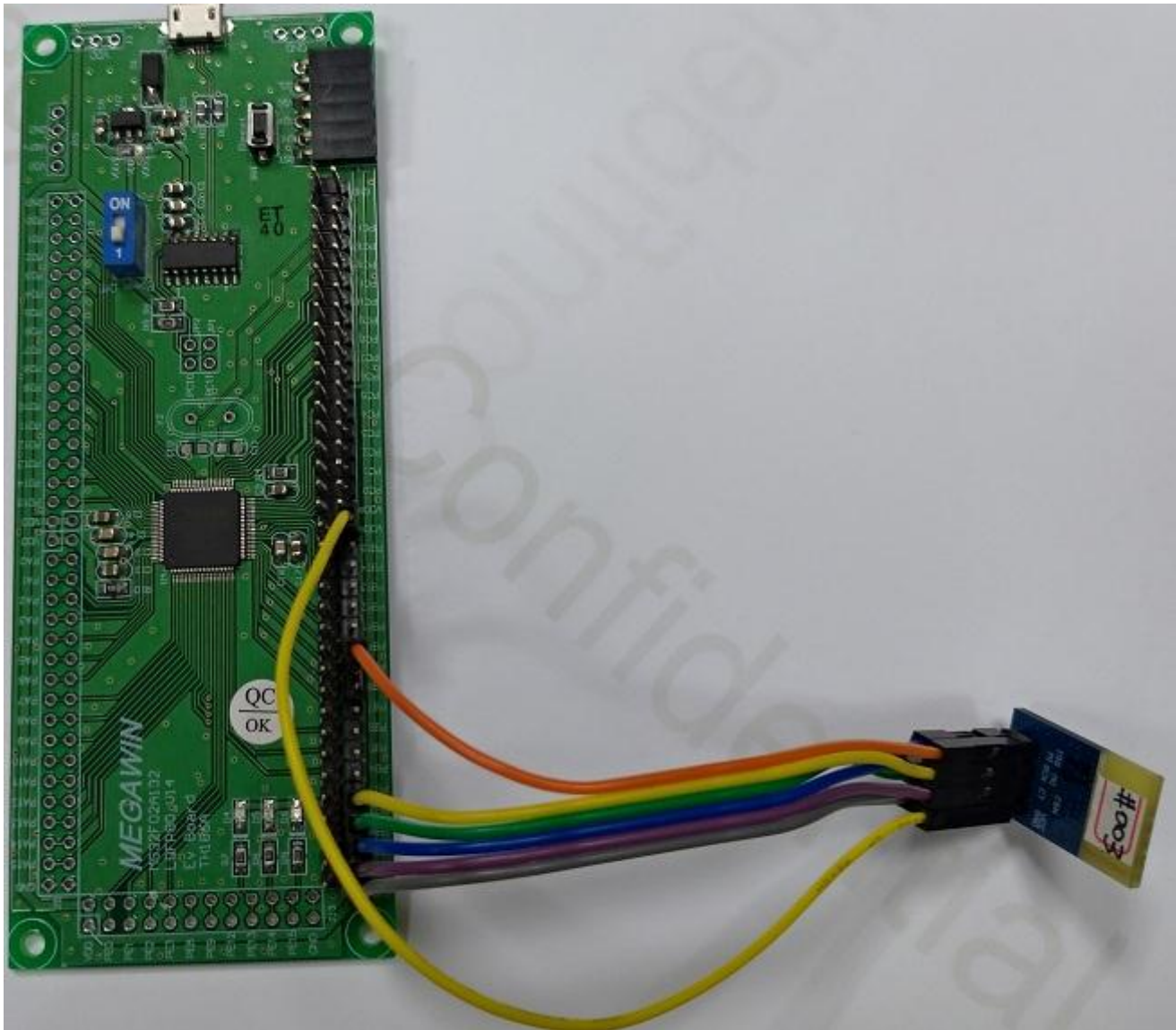


注意：

1. BLE 模块工作电压为 1.9 ~3.6V。
2. CT 引脚为测试引脚（无需连接）。

1.3. 实际 BLE 连接至 MCU 实物图

下图展示了实际 BLE 模块连接到 MCU 的实物图。



2. BLE_SPI.c

2.1. 概述

本文介绍了适用于 BLE 应用程序 BLE_SPI.c 协议的功能 SPI 初始化 SPI 读和写。

2.2. 函数列表

应用程序编程接口函数在文件 BLE_SPI.c 中定义，测试接口函数在 BLE_SPI.c 中定义，其中主要包括以下功能：

```
void SPI0_Init (void)
unsigned char SPI_WriteBuf (unsigned char reg, unsigned char const *pBuf, unsigned char len)
unsigned char SPI_ReadBuf (unsigned char reg, unsigned char *pBuf, unsigned char len)
```

2.3. 函数原型叙述

2.3.1. SPI0_Init

函数原型: void SPI0_Init (void)

	描述
简述	SPI0 / URTx(SPI mode) 模块初始化
参数 1	无
返回信息	无

2.3.2. SPI_WriteBuf

函数原型: unsigned char SPI_WriteBuf (unsigned char reg, unsigned char const *pBuf, unsigned char len)

	描述
简述	SPI0 / URTx 资料发送至 BLE
参数 1	reg, BLE 使用的指令
参数 2	pBuf, 发送数据缓冲区指针
参数 3	len, 数据长度 0 ~ 255 字节
返回信息	无

2.3.3. SPI_ReadBuf

函数原型: unsigned char SPI_ReadBuf (unsigned char reg, unsigned char *pBuf, unsigned char len)

	描述
简述	SPI0 / URTx 数据接收
参数 1	reg, BLE 使用的指令
参数 2	pBuf, 接收数据的指针地址
参数 3	len, 接收数据的长度 0 ~ 255 字节
返回信息	无

3. BSP.c

3.1. 概述

本文介绍了适用于 BLE 应用程序的 BSP.c 技术协议，功能包含初始化 BLE 使用的 EXIC, URT0, SPI/URT_x, IWD_T 模块，并提供了 SysTick 与 EXINT1 中断服务。

3.2. 函数列表

应用程序编程接口函数在文件 BSP 中定义.c 测试接口函数在 BSP 中定义.c，主要包括以下功能：

```
void BLE_EXIC_Init(void)
void EXINT1_IRQHandler(void)
void BLE_URT0_Init(void)
void IWDT_Init(void)
void BLE_SPI0_Init(void)
void BLE_LED_Init(void)
void LED_Flash(void)
void BSP_Init(void)
char IsIrqEnabled(void)
void IrqMcuGotoSleepAndWakeup(void)
unsigned int GetSysTickCount(void)
void SysTick_Handler(void)
```


3.3. 函数原型叙述

3.3.1. BLE_EXIC_Init

函数原型: void BLE_EXIC_Init (void)

	描述
简述	BLE 使用的 EXIC 模块初始化
参数 1	无
返回信息	无

3.3.2. EXINT1_IRQHandler

函数原型: void EXINT1_IRQHandler (void)

	描述
简述	EXINT1 中断服务代码
参数 1	无
返回信息	无

3.3.3. BLE_URT0_Init

函数原型: void BLE_URT0_Init (void)

	描述
简述	BLE 使用的 URT0 模块初始化
参数 1	无
返回信息	无

3.3.4. IWDT_Init

函数原型: void IWDT_Init (void)

	描述
简述	IWDT 模块初始化
参数 1	无
返回信息	无

3.3.5. BLE_SPI0_Init

函数原型: void BLE_SPI0_Init (void)

	描述
简述	BLE 使用的 SPI0 / URTx 模块和引脚初始化
参数 1	无
返回信息	无

注: PB0 AFS GPB0 (软件 NSS)、PB1 AFS SPI0_MISO、PB2 AFS SPI0_CLK 和 PB3 AFS SPI0_MOSI。

3.3.6. BLE_LED_Init

函数原型: void BLE_LED_Init (void)

	描述
简述	BLE 使用的 LED 模块初始化
参数 1	无
返回信息	无

3.3.7. LED_Flash

函数原型: void LED_Flash (void)

	描述
简述	LED 闪烁
参数 1	无
返回信息	无

3.3.8. BSP_Init

函数原型: void BSP_Init(void)

	描述
简述	BSP 初始化
参数 1	无
返回信息	无

3.3.9. IsIrqEnabled

函数原型: char IsIrqEnabled (void)

	描述
简述	确认 EXINT1 是否触发
参数 1	无
返回信息	无

3.3.10. IrqMcuGotoSleepAndWakeup

函数原型: void IrqMcuGotoSleepAndWakeup (void)

	描述
简述	MCU 执行睡眠模式并等待外部触发
参数 1	无
返回信息	无

3.3.11. GetSysTickCount

函数原型: unsigned int GetSysTickCount(void)

	描述
简述	取得 SysTick 计数值
参数 1	无
返回信息	无

3.3.12. SysTick_Handler

函数原型: void SysTick_Handler(void)

	描述
简述	SysTick 中断服务代码
参数 1	无
返回信息	无

4. retarget.c

4.1. 概述

本文介绍了适用于 BLE 应用程序的 `retarget.c` 函数。包含 URT0 中断服务程序、并提供 BLE 数据缓冲与 URT0 数据间相互传输、定时 IWDG 计数刷新、LED 闪烁…。

4.2. 函数列表

应用程序编程接口函数在文件 `retarget.c` 中定义，测试接口函数在 `retarget.c` 中定义，这主要包括以下功能：

```
void URT_IRQHandler (void)
void moduleOutData (u8 *data, u8 len)
void CheckComPortInData (void)
void UsrProcCallback (void)
```

4.3. 函数原型叙述

4.3.1. URT_IRQHandler

函数原型: void URT0_IRQHandler (void)

	描述
简述	URT0 TX/RX 中断服务程序
参数 1	无
返回信息	无

4.3.2. moduleOutData

函数原型: void moduleOutData(u8*data, u8 len)

	描述
简述	将要发送的数据移至发送缓冲区
参数 1	数据起始位置
参数 2	要移动的数据长度
返回信息	无

4.3.3. CheckComPortInData

函数原型: void CheckComPortInData(void)

	描述
简述	确认来自 COM 口的输入数据
参数 1	无
返回信息	无

4.3.4. UsrProcCallback

函数原型: void UsrProcCallback(void)

	描述
简述	刷新 IDWT 计数器、LED 闪烁、确认 COM 口输入数据与 URT0 输出数据
参数 1	无
返回信息	无

5. BLE MG126 Library

5.1. 概述

本文是 MG BLE 协议栈接口函数说明，适用于 MG 芯片的 BLE 应用开发。

5.2. 接口函数列表

应用界面函数定义在文件 mg_api.h 中，测试接口函数定义在 mg_test_api.h 中，主要包括以下函数：

- 1) void radio_initBle(unsigned char txpwr, unsigned char**addr);
- 2) unsigned char radio_initBle_TO(unsigned char txpwr, unsigned char**addr, unsigned short ms_timeout);
- 3) void ble_run_interrupt_start(unsigned short adv_interval);
- 4) void SetBleIntRunningMode(void);
- 5) void SetLePinCode(unsigned char *PinCode/*6 0~9 digitals*/);
- 6) void ble_set_adv_type(unsigned char type);
- 7) void ble_set_adv_data(unsigned char* adv, unsigned char len);
- 8) void ble_set_adv_rsp_data(unsigned char* rsp, unsigned char len);
- 9) void ble_set_name(unsigned char* name,unsigned char len);
- 10) void ble_set_interval(unsigned short interval);
- 11) void radio_standby(void);
- 12) void radio_resume(void);
- 13) unsigned char ble_set_wakeupdly(unsigned short counter);
- 14) void ble_set_adv_enableFlag(char sEnableFlag);
- 15) unsigned char ble_set_role(unsigned char role_new, unsigned short scan_window);
- 16) void ble_disconnect(void);
- 17) unsigned char ble_disc(unsigned char reason);
- 18) void ble_set_feature_supported(unsigned int feature_local);
- 19) void ble_set_md_enable(unsigned char enable_flag);
- 20) unsigned char *get_ble_version(void);
- 21) unsigned char *GetFirmwareInfo(void);
- 22) void ser_write_rsp_pkt(unsigned char pdu_type);
- 23) void att_notFd(unsigned char pdu_type, unsigned char attOpcode, unsigned short attHd);
- 24)void att_ErrorFd_ecode(unsigned char pdu_type, unsigned char attOpcode, unsigned short attHd, unsigned char errCode);
- 25) void att_server_rdByGrTypeRspDeviceInfo(unsigned char pdu_type);
- 26) void att_server_rdByGrTypeRspPrimaryService(unsigned char pdu_type,unsigned short start_hd,unsigned short end_hd,unsigned char*uuid,unsigned char uuidlen);
- 27) void att_server_rd(unsigned char pdu_type,unsigned char attOpcode,unsigned short att_hd,unsigned char* attValue,unsigned char datalen);
- 28) unsigned char sconn_notifydata(unsigned char* data, unsigned char len);
- 29) unsigned char sconn_indicationdata(unsigned char* data, unsigned char len);
- 30) void SIG_ConnParaUpdateReq(unsigned short IntervalMin, unsigned short IntervalMax, unsigned short SlaveLatency,unsigned short TimeoutMultiplier);
- 31) unsigned short sconn_GetConnInterval(void);
- 32) Unsigned char GetRssiData(void);
- 33) void radio_setBleAddr(u8 addr[6]);
- 34) void SetFixAdvChannel(unsigned char isFixCh37Flag);
- 35) void test_carrier(unsigned char freq, unsigned char txpwr);
- 36) void test_SRRCCarrier(unsigned char freq, unsigned char txpwr);
- 37) void test_PRBS9(unsigned char freq, unsigned char txpwr);
- 38) void test_RX(unsigned char freq);

5.3. 接口函数说明

5.3.1. radio_initBle

函数原型: void radio_initBle(unsigned char txpwr, unsigned char**addr/*Output*/);

函数功能: 初始化蓝牙芯片及蓝牙协议栈

输入参数: txpwr 参数用于初始化蓝牙芯片发射功率, 可取的值为 TXPWR_0DBM, TXPWR_3DBM 等

输出参数: addr 参数返回蓝牙 MAC 地址信息, 6 个字节长度。

返回值: 无

注意事项: 在 SetBleIntRunningMode 之后调用。

5.3.2. radio_initBle_TO

函数原型: unsigned char radio_initBle_TO(unsigned char txpwr, unsigned char**addr, unsigned short ms_timeout);

函数功能: 初始化蓝牙芯片及蓝牙协议栈

输入参数: txpwr 参数用于初始化蓝牙芯片发射功率, 可取的值为 TXPWR_0DBM, TXPWR_3DBM 等
ms_timeout 参数用于蓝牙芯片初始化超时时间, 单位为毫秒, 建议值为 10~50

输出参数: addr 该参数返回蓝牙 MAC 地址信息, 6 个字节长度。

返回值: 0 表示初始化失败, 非 0 表示初始化成功。

注意事项: 在 SetBleIntRunningMode 之后调用。

5.3.3. ble_run_interrupt_start

函数原型: void ble_run_interrupt_start(unsigned short adv_interval);

函数功能: 采用中断服务程序方式运行蓝牙协议

输入参数: adv_interval, 参数的单位为 0.625ms, 如 160 表示 100ms 的广播间隔

输出参数: 无

返回值: 无

注意事项: 请参考程序设计指南“中断服务程序方式运行模式”

5.3.4. SetBleIntRunningMode

函数原型: void SetBleIntRunningMode(void);

函数功能: 设置中断服务程序方式

输入参数: 无

输出参数: 无

返回值: 无

注意事项: 请参考程序设计指南“中断服务程序方式运行模式”

5.3.5. SetLePinCode

函数原型: void SetLePinCode(unsigned char *PinCode/*6 0~9 digitals*/);

函数功能: 设置蓝牙配对 PIN

输入参数: PinCode, 6 个 0~9 的数字

输出参数: 无

返回值: 无

注意事项: 在 ble_run_interrupt_start 前调用。缺省 PIN 为 000000, 如果需要改变 PIN 可以调用此函数。

5.3.6. ble_set_adv_type

函数原型: void ble_set_adv_type(unsigned char type);

函数功能: 设置 BLE 广播 Adv PDU Header

输入参数: type, 0-adv_ind, 2-adv_nonconn_ind, 缺省为 0x80。

输出参数: 无

返回值: 无

注意事项: 本函数可以在任意时候调用, 广播数据内容立即起效。

5.3.7. ble_set_adv_data

函数原型: void ble_set_adv_data(unsigned char* adv, unsigned char len);

函数功能: 设置 BLE 广播资料

输入参数: adv, 广播数据指针

len, 广播数据长度

输出参数: 无

返回值: 无

注意事项: 本函数可以在任意时候调用, 广播数据内容立即起效。

5.3.8. ble_set_adv_rsp_data

函数原型: void ble_set_adv_rsp_data(unsigned char* rsp, unsigned char len);

函数功能: 设置 BLE 广播扫描应答数据

输入参数: rsp, 广播扫描应答数据指针

len, 广播扫描应答数据长度

输出参数: 无

返回值: 无

注意事项: 1) 本函数可以在任意时候调用, 广播数据内容立即起效。

2) 调用本函数会导致函数 ble_set_name()失效, 但 app.c 中函数 getDeviceInfoData()依然有效。

5.3.9. ble_set_name

函数原型: void ble_set_name(unsigned char* name, unsigned char len);

函数功能: 设置 BLE 广播应答包内名字内容

输入参数: name, 名字数据指针

len, 名字数据长度

输出参数: 无

返回值: 无

注意事项: 1) 本函数可以在任意时候调用, 数据内容立即起效。

2) 调用本函数仅会改变广播扫描应答的数据内容, 为了与 GATT 对应内容一致, 需要同步修改 app.c 中 DeviceInfo 的内容, 具体可参考 app.c 中函数 updateDeviceInfoData()的实现。

5.3.10. ble_set_interval

函数原型: void ble_set_interval(unsigned short interval);

函数功能: 设置 BLE 广播间隔时间

输入参数: interval, 广播间隔, 范围: 0x0020 ~ 0x4000, 单位 0.625ms

输出参数: 无

返回值: 无

注意事项: 本函数可以在任意时候调用, 数据内容立即起效。

5.3.11. radio_standby

函数原型: void radio_standby(void);

函数功能: 蓝牙芯片进入 standby, 处于最省电状态

输入参数: 无

输出参数: 无

返回值: 无

注意事项：此函数一般在 MCU 进入 standby 前调用，也可以在 UsrProcCallback 或者 ble 进入 sleep 后调用。

系统进入 standby 后功耗约为 2uA。

5.3.12. radio_resume

函数原型：void radio_resume(void);

函数功能：蓝牙芯片从 standby 恢复

输入参数：无

输出参数：无

返回值：无

注意事项：本函数可以在广播进入 standby 以后需要恢复广播的时候调用。

5.3.13. ble_set_wakeupdly

函数原型：unsigned char ble_set_wakeupdly(unsigned short counter);

函数功能：设置 mcu 唤醒时间。mcu 使用低功耗，irq 中断唤醒会有延迟，需要调用此接口设置唤醒延迟时间。Mcu 不使用低功耗模式，不需要设置唤醒时间。

输入参数：counter：mcu 唤醒延迟时间，单位 1/64 ms。范围：0x0000~0x0040

输出参数：无

返回值：0 – 失败； 1 – 成功

注意事项：本函数在 radio_initBle 前调用。

5.3.14. ble_set_adv_enableFlag

函数原型：void ble_set_adv_enableFlag(char sEnableFlag);

函数功能：设置 BLE 是否广播

输入参数：sEnableFlag，1--运行广播；0--停止广播

输出参数：无

返回值：无

注意事项：本函数可以在任意时候调用，立即起效。

5.3.15. ble_set_role

函数原型：unsigned char ble_set_role(unsigned char role_new, unsigned short scan_window);

函数功能：设置 ble 角色为外设（0）或者中心（1）。如果是中心角色，设置扫描窗口。

输入参数：role_new，0 – 外设；1—中心。____缺省角色为外设

scan_window，单广播信道的扫描窗口，单位 0.625ms。范围：0x0004~0x4000

输出参数：无

返回值：0 – 失败； 1 – 成功

注意事项：本函数仅在非连接状态调用，不调用的话角色缺省为外设。scan_window 参数仅在中心角色有效，是单个广播信道的扫描窗口长度。扫描从 37 广播信道开始。

5.3.16. ble_disconnect

函数原型：void ble_disconnect(void);

函数功能：断开已有的连接

输入参数：无

输出参数：无

返回值：无

注意事项：本函数可以在任意时候调用，立即起效。

5.3.17. ble_disc

函数原型: unsigned char ble_disc(unsigned char reason);

函数功能: 断开已有的连接

输入参数: reason - 连接断开原因 (请按照 BLE 协议填写)

输出参数: 无

返回值: 0-失败; 1-成功

注意事项: 本函数可以在任意时候调用, 立即起效。

5.3.18. ble_set_feature_supported

函数原型: void ble_set_feature_supported(unsigned int feature_local);

函数功能: 设置 BLE 的 feature (32bit)

输入参数: feature_local--本机支持的 LE feature

输出参数: 无

返回值: 无

注意事项: 本函数可以在任意时候调用, 在连接建立过程中起效。

5.3.19. ble_set_md_enable

函数原型: void ble_set_md_enable(char sEnableFlag);

函数功能: 设置 BLE 接收是否支持 more data

输入参数: sEnableFlag, 1--支持 more data; 0--不支持 more data

输出参数: 无

返回值: 无

注意事项: 本函数可以在任意时候调用, 连接后起效。此函数一般用于 OTA 以加快 OTA 速度。不调用此函数缺省不支持 more data。

5.3.20. get_ble_version

函数原型: unsigned char *get_ble_version(void);

函数功能: 获取蓝牙协议栈版本信息字符串

输入参数: 无

输出参数: 无

返回值: 蓝牙协议栈版本信息字符串

5.3.21. GetFirmwareInfo

函数原型: unsigned char *GetFirmwareInfo(void);

函数功能: 获取蓝牙基带版本信息字符串

输入参数: 无

输出参数: 无

返回值: 蓝牙基带版本信息字符串

5.3.22. ser_write_rsp_pkt

函数原型: void ser_write_rsp_pkt(unsigned char pdu_type);

函数功能: 对具有 Write With Response 属性特征值写操作后的应答函数

输入参数: pdu 类型参数, 直接引用回调函数 ser_write_rsp 中对应参数

输出参数: 无

返回值: 无

注意事项: 对需要写应答的特征值, 如果不应答会导致连接的断开。

5.3.23. att_notFd

函数原型: void att_notFd(unsigned char pdu_type, unsigned char attOpcode, unsigned short attHd);

函数功能: 对无效特征值（或没有定义的特征值）进行操作的应答函数

输入参数: pdu_type PDU 类型参数

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

attOpcode 操作类型参数

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

attHd 对应特征值句柄值

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

输出参数: 无

返回值: 无

注意事项: 凡是无效特征值的操作需要应答本函数, 可将本函数作为缺省调用。

5.3.24. att_ErrorFd_eCode

函数原型: void att_ErrorFd_eCode(unsigned char pdu_type, unsigned char attOpcode, unsigned short attHd,

unsigned char errorCode);

函数功能: 对无效操作的应答函数

输入参数: pdu_type PDU 类型参数

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

attOpcode 操作类型参数

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

attHd 对应特征值句柄值

直接引用回调函数 ser_write_rsp、att_server_rdyByGrType 或 server_rd_rsp 中对应参数

errorCode 错误代码, 请参考 ATT Error Code

输出参数: 无

返回值: 无

ATT Error Code define:

```
#define ATT_ERR_INVALID_HANDLE 0x01
#define ATT_ERR_READ_NOT_PERMITTED 0x02
#define ATT_ERR_WRITE_NOT_PERMITTED 0x03
#define ATT_ERR_INVALID_PDU 0x04
#define ATT_ERR_INSUFFICIENT_AUTHEN 0x05
#define ATT_ERR_UNSUPPORTED_REQ 0x06
#define ATT_ERR_INVALID_OFFSET 0x07
#define ATT_ERR_INSUFFICIENT_AUTHOR 0x08
#define ATT_ERR_PREPARE_QUEUE_FULL 0x09
#define ATT_ERR_ATTR_NOT_FOUND 0x0a
#define ATT_ERR_ATTR_NOT_LONG 0x0b
#define ATT_ERR_INSUFFICIENT_KEY_SIZE 0x0c
#define ATT_ERR_INVALID_VALUE_SIZE 0x0d
#define ATT_ERR_UNLIKELY 0x0e
#define ATT_ERR_INSUFFICIENT_ENCRYPT 0x0f
#define ATT_ERR_UNSUPPORTED_GRP_TYPE 0x10
#define ATT_ERR_INSUFFICIENT_RESOURCES 0x11
```

5.3.25. att_server_rdyByGrTypeRspDeviceInfo

函数原型: void att_server_rdyByGrTypeRspDeviceInfo(unsigned char pdu_type);

函数功能：对缺省 Device Info 内容的应答可调用本接口函数

输入参数：pdu 类型参数，直接引用回调函数 att_server_rdyGrType 中对应参数

输出参数：无

返回值：无

注意事项：如果用户直接使用发布包代码，可直接调用本接口函数。

5.3.26. att_server_rdyGrTypeRspPrimaryService

函数原型：void att_server_rdyGrTypeRspPrimaryService(unsigned char pdu_type,
unsigned short start_hd,
unsigned short end_hd,
unsigned char* uuid,
unsigned char uuidlen);

函数功能：应答 Primary Service 的查询，用户需按特征值实际定义的句柄及 UUID 填充对应数据

输入参数：pdu_type PDU 类型参数，直接引用回调函数 att_server_rdyGrType 中对应参数

start_hd, 某个 Service 对应的起始句柄值

end_hd, 某个 Service 对应的结束句柄值

uuid, 某个 Service 对应的 UUID 字符串（Hex 值），如 0x180A 表示为 0x0a, 0x18。

uuidlen, 某个 Service 对应 UUID 字符串的长度

输出参数：无

返回值：无

注意事项：需要严格按照特征值定义规划填充对应参数。

5.3.27. att_server_rd

函数原型：void att_server_rd(unsigned char pdu_type,
unsigned char attOpcode,
unsigned short att_hd,
unsigned char* attValue,
unsigned char datalen);

函数功能：读取某特征值的值。

输入参数：pdu_type PDU 类型参数，直接引用回调函数 server_rd_rsp 中对应参数

attOpcode 操作对应的值，直接引用回调函数 server_rd_rsp 中对应参数

att_hd 特征值对应的句柄值，直接引用回调函数 server_rd_rsp 中对应参数

attValue 特征值对应的值字符串指针

datalen 特征值字符串长度

输出参数：无

返回值：无

注意事项：需要按需将对应特征值内容作为应答内容，如果对应特征值内容无效可应答 att_notFd（）。

5.3.28. sconn_notifydata

函数原型：unsigned char sconn_notifydata(unsigned char* data, unsigned char len);

函数功能：通过蓝牙发送数据

输入参数：data 需要发送的数据指针

len 数据长度

输出参数：无

返回值：成功发送的数据字节数

注意事项：本接口函数会根据系统缓存情况自动拆包发送数据，但不得阻塞循环调用。调用前需要确认 cur_notifyhandle 正确。

5.3.29. sconn_indicationdata

函数原型: unsigned char sconn_indicationdata(unsigned char* data, unsigned char len);

函数功能: 通过蓝牙发送数据

输入参数: data 需要发送的数据指针

len 数据长度

输出参数: 无

返回值: 成功发送的数据字节数

注意事项: 本接口函数会根据系统缓存情况自动拆包发送数据, 但不得阻塞循环调用。调用前需要确认 cur_notifyhandle 正确。

5.3.30. SIG_ConnParaUpdateReq

函数原型: void SIG_ConnParaUpdateReq(unsigned short IntervalMin, unsigned short IntervalMax, unsigned short

SlaveLatency, unsigned short TimeoutMultiplier);

函数功能: 在连接状态下, 尝试使用使用者期望范围内的发射间隔。最终的发射间隔由 central 设备决定。

输入参数: IntervalMin - 最小发射间隔, 单位 1.25ms。范围 6 到 3200。

IntervalMax - 最大发射间隔, 单位 1.25ms。范围 6 到 3200。

SlaveLatency - 响应延迟, 范围 0~500, 还要小于(SupervisionTimeout / (IntervalMax*2)) -1。

TimeoutMultiplier - 连接断开时间。单位 10ms。范围 10 到 3200。

输出参数: 无

返回值: 无

注: SlaveLatency 建议范围 0-5。

5.3.31. sconn_GetConnInterval

函数原型: unsigned short sconn_GetConnInterval(void);

函数功能: 获得目前蓝牙连接当前使用的发射间隔数值, 单位为 1.25ms。

输入参数: 无

输出参数: 无

返回值: 当前蓝牙连接的间隔。

5.3.32. GetRssiData

函数原型: Unsigned char GetRssiData(void);

函数功能: 接收广播包或者数据报时调用此函数, 可以获取接收信号强度原始数据。可以根据参考的 RSSI 数值, 用此数据来比较接收信号强度。

输入参数: 无

输出参数: 无

返回值: 接收信号强度相对值, 比如 191 比 190 大 1dB。

5.3.33. radio_setBleAddr

函数原型: void radio_setBleAddr(unsigned char addr[6]);

函数功能: 设置客户自己的蓝牙设备地址。

输入参数: addr - 蓝牙设备地址, 6 字节

输出参数: 无

返回值: 无

注意事项: 在协定栈初始化后调用。

5.3.34. SetFixAdvChannel

函数原型: void SetFixAdvChannel(unsigned char isFixCh37Flag);

函数功能：在调试阶段，为了便于空中抓包，协议栈会固定在 37 信道广播，协议栈缺省在所有通道广播。

输入参数：isFixCh37Flag 设置是否仅在 37 通道广播。

输出参数：无

返回值：无

注意事项：仅用于调试，需要调试时在协议栈初始化后调用 SetFixAdvChannel(1)。

5.3.35. test_carrier

函数原型：void test_carrier(unsigned char freq, unsigned char txpwr);

函数功能：测试实际载波发射功率，协议栈会固定在 2400+freq 频点发送载波，理想发射功率参考 txpwr。

输入参数：freq - 载波中心频率为(2400+freq)MHz

txpwr - 0x43 对应理想射频发射功率 0dBm

输出参数：无

返回值：无

注意事项：用于载波和晶体频偏测试。在协议栈初始化后调用 test_carrier(80, 0x43)，然后 while(1)循环等待。

5.3.36. test_SRRCCarrier

函数原型：void test_SRRCCarrier(unsigned char freq, unsigned char txpwr);

函数功能：用于 SRRRC 测试-载波。

输入参数：freq - 载波中心频率为(2400+freq)MHz

txpwr - 0x43 对应理想射频发射功率 0dBm

输出参数：无

返回值：无

注意事项：用于 SRRRC 定频载波发射测试。在协议栈初始化后调用，然后 while(1)循环等待。

5.3.37. test_PRBS9

函数原型：void test_PRBS9(unsigned char freq, unsigned char txpwr);

函数功能：用于 SRRRC 测试-PRBS 调制信号发射

输入参数：freq - 载波中心频率为(2400+freq)MHz

txpwr - 0x43 对应理想射频发射功率 0dBm

输出参数：无

返回值：无

注意事项：用于 SRRRC 调制信号发射测试。在协议栈初始化后调用，然后 while(1)循环等待。

5.3.38. test_RX

函数原型：void test_RX(unsigned char freq);

函数功能：用于测试-使芯片处于接收状态

输入参数：freq - 载波中心频率为(2400+freq)MHz

输出参数：无

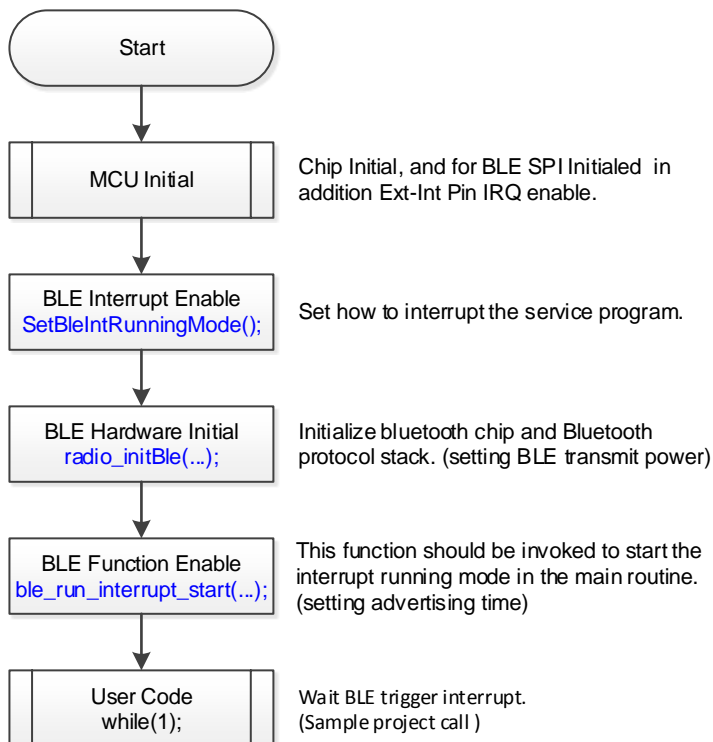
返回值：无

注意事项：用于接收测试，不关心数据。在协议栈初始化后调用，然后 while(1)循环等待。

5.4. 中断服务程序方式运行模式

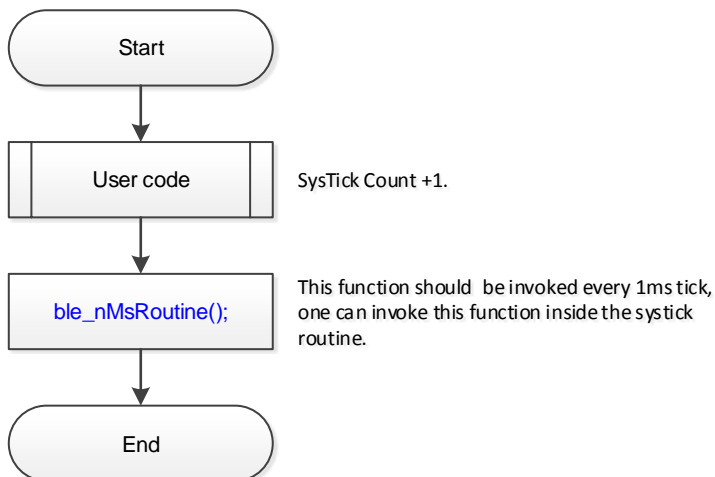
5.4.1. 主要运行模式

下图展示了主要代码的软件流程。



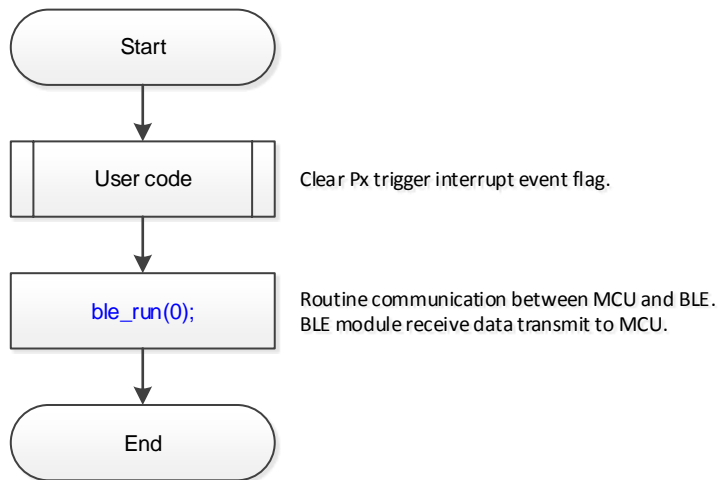
5.4.2. SysTick IRQ 处理

下图展示了 SysTick_Handler 代码的软件流程。



5.4.3. BLE IRQ 处理

下图展示了 EXINTx_IRQHandler 代码的软件流程。



6. 修订历史

Revision V1.10 (2020_1211)		Chapter
1	简体中文版	
Revision V1.10 (2020_0731)		Chapter
1	Initial version	
Revision V1.11 (2020_0731)		Chapter
1	更动第一章说明文件名称	1
2	移除 2.3.1 BSP.c 中 BLE_CSC_Init, 此为多余代码。	2.3.1
3	更新 BLE_SPI.c, BSP.c and retarget.c 概述内容。	1.1、2.1、3.1
Revision V1.12 (2020_0731)		Chapter
1	更新 2.1.概述.	2.1
2	更新 2.2.函数列表	2.2
3	更新 2.3.1. SPI0_Init 支持 SPI/URTx	2.3.1
4	更新 2.3.2. SPI_WriteBuf 支持 SPI/URTx	2.3.2
5	更新 2.3.3. SPI_ReadBuf 支持 SPI/URTx	2.3.3
6	更新 3.3.5.BLE_SPI0_Init 支持 SPI0 / URTx 模块和引脚初始化	3.3.5