目录

青风带你玩蓝牙 nRF52832 系列教程

----------作者：青风

出品论坛：　www.qfv8.com　青风电子社区

作者： 青风
出品论坛: www.qfv8.com
淘宝店：http://qfv5.taobao.com
QQ 技术群：346518370
硬件平台：青云 QY-nRF52832 开发板

第二章 接收信号强度 rssi 和蓝牙发射功率

对于无线通信设备，对于设备的发射功率以及对面设备发送过来的信号强度 rssi 的获取都是读者设计的时候必须考虑的问题。无线设备信号的好坏，直接影响到了通信的质量和通信距离。因此本章将针对这两个问题进行展开讨论。其中接收信号强度 rssi 是用来判断通信双方互相接收的信号强度，表示通信中实际接收到的信号质量。而发生功率则是由信号发射端配置的信号功率强度，对最后接收的信号强度有直接影响。

2.1：nRF52832 蓝牙 BLE 的 rssi 获取：

RSSI 的英文全称为 Received Signal Strength Indication，中文意思是接收的信号强度。这个参数用于指示无线发送层的接收信号强度，用来判定链接质量，以及是否增大广播发送功率的强度。对应 RSSI 的应用，常见的比如通过接收到的信号强弱可以判断信号点与接收点的距离，进而根据相应数据进行定位计算。

对应信号强度的单位为 dBm，表示为分贝毫瓦(全写为“decibel relative to one milliwatt”)，为一个指代功率的绝对值。

例如：按照公式 Rss=10lgP，只需将接受到的信号功率 P（mw）代入就是接收信号强度。

[例 1] 如果发射功率 P 为 1mw，折算为 dBm 后为 0dBm。

[例 2] 对于 40W 的功率，按 dBm 单位进行折算后的值应为：

10lg（40W/1mw)=10lg（40000）=10lg4+10lg10+10lg1000=46dBm。

首先我们需要知道的是接收的无线信号强度 dbm 都是负数，最大是 0。因此测量出来的 dbm 值肯定都是负数。因为 dbm 值只在一种情况下为 0，那就是在理想状态下传输链路没有信号损失，一般我们认为 dbm 为 0 是其最大值，意味着接收方把发射方发射的所有无线信号都接收到了，即无线路由器发射多少功率，接收的一段就获得多少功率。当然这是在理想状态下测量的，在实际中无线接收设备即使紧挨着无线的发射天线也不会达到 dbm 为 0 的效果。所以说测量出来的 dbm 值都是负数，而不要盲目的认为负数就是信号不好。

本例讲演示如何通过从机设备来接收主机设备的 rssi 信号强度。本节在匹配的 SDK15.0 的蓝牙串口例子基础上就行编写，目标是通过从机获取连接的主机的信号强度。

2.1.1 BLE 定时器配置

●定时器初始化

本例在 SDK15 下的串口蓝牙例子下进行修改，我们采用定时器在协议栈下的来定时读取接收到的信号强度，因此首先设置一个定时器，定时器 ID 声明为 m_rssi_timer_id，注册一个定时器超时回调函数 rssi_timeout_handler。具体代码如下所示：

```
1.  //定时器初始化
2.  static void timers_init(void)
3.  {
4.  uint32_t err_code;
5.  // 初始化定时器，给一个定时器空间
6.  APP_TIMER_INIT(APP_TIMER_PRESCALER,APP_TIMER_OP_QUEUE_SIZE, false);
7.  //创建一个定时，设置定时器模式为重复定时，注册超时回调函数：
8.  err_code=app_timer_create(&m_rssi_timer_id, APP_TIMER_MODE_REPEATED,
9.                              rssi_timeout_handler);
10. APP_ERROR_CHECK(err_code);
11. }}
```

●定时超时回调

在 app_timer_create（）函数中，注册的 rssi_timeout_handler（）作为定时器函数创建的一个定时器超时中断处理，需要处理的是你只需要执行 rssi 更新或者打印：

```
1.  static void rssi_timeout_handler(void)
2.  {
3.  int8_t rssi=0;
4.  uint8_t p_ch_index;
5.  //开始向应用程序报告接收信号的强度
6.  sd_ble_gap_rssi_start(m_conn_handle,BLE_GAP_RSSI_THRESHOLD_INVALID,0);
7.  //获取最后一个连接事件接收到的信号强度报告
8.  sd_ble_gap_rssi_get(m_conn_handle, &rssi,&p_ch_index);
9.  NRF_LOG_INFO("rssi: %d\r\n",rssi);//打印接收信号强度
10. NRF_LOG_INFO("Channel: %d\r\n",p_ch_index);//打印信号的广播频道
11. }
```

在上面的代码中，我们要开始收集接收信号强度 rssi 的信号报告，这里面需要调用到两个协议栈函数，众所周知协议栈函数是以 sd 开头命名的，未开源的函数，两个函数如下定义：

◎sd_ble_gap_rssi_start(m_conn_handle, BLE_GAP_RSSI_THRESHOLD_INVALID,0);

◎sd_ble_gap_rssi_get(m_conn_handle, &rssi, &p_ch_index);

使用这两个函数之前，需要具体对这两个函数进行说明，了解其功能和参数，如下面的表 2.1 和表 2.2 所示：

表 2.1 sd_ble_gap_rssi_start（）函数

函数：sd_ble_gap_rssi_start(m_conn_handle, BLE_GAP_RSSI_THRESHOLD_INVALID,0);
功能：开始向应用程序报告接收到的信号强度。当 RSSI 值发生变化时，将报告一个新的事件，直到函数 sd_ble_gap_rssi_stop 被调用。产生事件 BLE_GAP_EVT_RSSI_CHANGED 时，新的 RSSI 数据就可以用了。 事件的生成频率取决于形参 threshold_dbm 和形参 kip_count 输入参数.}

参数：[in] conn_handle 连接句柄。 参数：[in] threshold_dbm 在触发 BLE_GAP_EVT_RSSI_CHANGED 事件之前最小的变化 dBm 值。如果 threshold_dbm 等于参数 BLE_GAP_RSSI_THRESHOLD _INVALID，事件将会被关闭。 参数：[in] skip_count 在发送一个新的参数 BLE_GAP_EVT_RSSI_CHANGED 事件之前，更改了一个或多个 threshold_dbm 的 RSSI 样本的数量值。
返回值:NRF_SUCCESS 成功激活 RSSI 报告。 返回值:NRF_ERROR_INVALID_STATE RSSI 报告已经在进行中。 返回值:BLE_ERROR_INVALID_CONN_HANDLE 提供的连接句柄无效。

表 2.2 sd_ble_gap_rssi_get（）函数

函数：sd_ble_gap_rssi_get(m_conn_handle, &rssi,&p_ch_index);
功能：获取最后一个连接事件的接收信号强度。在使用这个函数之前，必须调用 sd_ble_gap _rssi_start 来开始报告 RSSI。直到在调用函数 sd_ble_gap_rssi_start 之后第一次采样 RSSI，将返回 NRF_ERROR_NOT_FOUND。
参数：conn_handle 连接句柄。 参数：[out] p_rssi 指向应该存储 RSSI 测量值的位置的指针。 参数：[out] p_ch_index 指向 RSSI 测量通道索引的存储位置的指针。
返回值:NRF_SUCCESS 成功读取 RSSI。 返回值:NRF_ERROR_NOT_FOUND 没有例子被发现。 返回值:NRF_ERROR_INVALID_ADDR 提供无效的指针 返回值:BLE_ERROR_INVALID_CONN_HANDLE 提供的连接句柄无效。 返回值:NRF_ERROR_INVALID_STATE RSSI reporting is not ongoing.

调用这两个函数后，当主机设备和我们设备连接后，我们直接通过串口 LOG 打印主机 rssi 信号强度的值：NRF_LOG_INFO("rssi: %d\r\n",rssi);

同时打印 RSSI 测量通道 NRF_LOG_INFO("Channel: %d\r\n",p_ch_index);

或者我们可以反向的告知手机，手机上显示从机接收到的主机信号强度。这里面就可以直接用串口蓝牙的上传函数 ble_nus_string_send，因为上传属性被定义了通知类型，所以数据会被上传到手机通知中，当手机通知使能就可以观察到变化数字。

●开始定时

当配置函数定时器后，定时器开始定时，开始定时器以设置时间间隔开始定时，定时器知道为对应的 ID，本次声明的 m_timer_rssi_id。最后定时器会在规定对应时间内执行超时中断操作，具体代码如下：

```
1.  //开始定时开始定时
2.  static void application_timers_start(void)
3.  {
4.   //定时时间间隔
5.   uint32_t err_code;
6.   //开始定时
7.   err_code = app_timer_start(m_timer_rssi_id, TIME_LEVEL_MEAS_INTERVAL, NULL);
8.       APP_ERROR_CHECK(err_code);
9.  }
```

2.1.2 主函数编写

主函数写一个测试函数，主要是加入定时器初始化和定时器开始定时函数，以更新获得的 rssi 信号强度，编写代码如下：

```
1.  int main(void)
2.  {
3.  bool erase_bonds;
4.
5.  // 初始化过程
6.  log_init();
7.  timers_init();//初始化定时器
8.  ..................
9.  ..................
10. application_timers_start();//添加定时器开始定时
11. advertising_start(erase_bonds);
12. // 进入主循环
13. for (;;)
14. {
15. idle_state_handle();
16. }
```

修改后进行编译，编译通过会提示 OK。

2.1.3 应用与调试

首先采用 nrfgo studio 软件下载协议栈，然后使用 keil 软件直接下载应用程序。应用程序下载成功后，程序开始运行，同时开发板上广播 LED 开始广播。本实验需要连接开发板的串口转 USB 接口。同时需要打开串口助手，串口助手设置如下图 2.1 所示，波特率为 115200，点击打开串口。当使用手机 app nrf connect 和开发板相连接时，会观察到输出的接收的 RSSI 信号强度和测试频道如下图 2.1 所示：

图：2.1 从机输出接收信号强度

2.2：蓝牙 BLE 的发送功率设置：

蓝牙接收信号强度 rssi 的直接影响因素就是蓝牙信号的发送功率。发射功率就是你所使用的设备(开发板、手机)所发射出来给主机或从机设备的信号强度。同时在实际应用当中，时常也需要修改蓝牙的发射功率，以达到省电的目的。那么本章内容就主要来探讨一下蓝牙发射功率 power 的设置。

本例在匹配的 SDK15.0 的蓝牙串口例子基础上就行编写，使用的协议栈为：s132。

2.2.1 发射功率控制

在 nRF52832 中可以设置为 9 个发射等级，分别是-40dBm、-20dBm、-16dBm、-12dBm、-8dBm、-4dBm、0dBm、+3dBm 和+4dBm。Nrf52840 可以设置 10 个发射等级，比 nRF52832 多一个+8dBm。如果没有在代码中设置发射功率，默认是为 0dbm，如果设置发生功率越高，其设备功耗越高，信号强度越好。本例在 SDK15.下的串口蓝牙例子下添加发送功率，需要添加下面几个地方：

●首先设置广播的 tx_power_level，这里就是初始化广播的发送功率。代码如下：

```
1.  static void advertising_init(void)
2.  {
3.  uint32_t                  err_code;
4.  ble_advertising_init_t init;
5.  int8_t   tx_power_level = TX_POWER_LEVEL;//设置发射功率
6.  memset(&init, 0, sizeof(init));
7.
8.  init.advdata.name_type               = BLE_ADVDATA_FULL_NAME;
9.  init.advdata.include_appearance = false;
10. init.advdata.flags = BLE_GAP_ADV_FLAGS_LE_ONLY_LIMITED_DISC_MODE;
11. init.advdata.p_tx_power_level          = &tx_power_level;
12.
13. init.srdata.uuids_complete.uuid_cnt = sizeof(m_adv_uuids) / sizeof(m_adv_uuids[0]);
14. init.srdata.uuids_complete.p_uuids   = m_adv_uuids;
15.
16. init.config.ble_adv_fast_enabled   = true;
17. init.config.ble_adv_fast_interval = APP_ADV_INTERVAL;
18. init.config.ble_adv_fast_timeout   = APP_ADV_DURATION;
19. init.evt_handler = on_adv_evt;
20.
21. err_code = ble_advertising_init(&m_advertising, &init);
22. APP_ERROR_CHECK(err_code);
23.
24. ble_advertising_conn_cfg_tag_set(&m_advertising, APP_BLE_CONN_CFG_TAG);
25. }
```

●第二：在 GAP 初始化的时候，这里需要调用协议栈函数 sd_ble_gap_tx_power_set（），通过这个函数来设置功率等级 tx_power_level ，这个函数可以动态的修改发送功率。关于该函数的具体定

义如下表 2.3 所示：

表 2.3 sd_ble_gap_tx_power_se（）函数

函数： sd_ble_gap_tx_power_set(uint8_t role, uint16_t handle, int8_t tx_power)
功能：设置设备的发射功率。
参数：role 要设置信号传输的角色，请参见 BLE_GAP_TX_POWER_ROLES 设置可能的角色。 参数：handle 句柄参数根据角色进行解释: 如果角色是@ref BLE_GAP_TX_POWER_ROLE_CONN，此值是特定的连接句柄。 如果角色是@ref BLE_GAP_TX_POWER_ROLE_ADV，则使用广播句柄标识的广播集将使用指定的传输功率，如果参数 ble_gap_adv_properties_t::include_tx_power 设置了，则将其包含在广播包头中。 对于所有其他角色，句柄将被忽略。 参数：tx_power 无线电传输功率(有关可接受的值，请参阅说明)。支持的 tx_power 值:-40dBm、-20dBm、-16dBm、-12dBm、-8dBm、-4dBm、0dBm、+3dBm 和+4dBm。 注意：当一个连接被创建时，它将从连接的发起者或广播者那里继承传输能力。发起者将具有与扫描者相同的传输功率。
返回值:NRF_SUCCESS 成功改变了发射功率。 返回值:NRF_ERROR_INVALID_PARAM 提供的参数无效 返回值:BLE_ERROR_INVALID_ADV_HANDLE 找不到广播句柄 返回值:BLE_ERROR_INVALID_CONN_HANDLE 提供的连接句柄无效。

在函数中，对应设备的角色定义，函数库中提供了结构体参数 BLE_GAP_TX_POWER_ROLES 进行了定义，关于这个结构体如下代码所示：

```
1.  enum BLE_GAP_TX_POWER_ROLES
2.  {
3.    BLE_GAP_TX_POWER_ROLE_ADV       = 1,    /*广播角色 */
4.    BLE_GAP_TX_POWER_ROLE_SCAN_INIT = 2,    /*扫描和发起角色 */
5.    BLE_GAP_TX_POWER_ROLE_CONN      = 3,    /*连接角色*/
6.  };
```

对于从机设备来说，我们开始广播的时候属于广播角色，因此选择 BLE_GAP_TX_POWER_ROLE_ADV 作为角色定义。形式参数 tx_power 定义为 TX_POWER_LEVEL，这个参数作为一个可变值，读者可以采用串口 AT 指令方式或者软件定时的方法动态控制发射功率。

```
1.  static void tx_power_set(void)
2.  {
3.  ret_code_t err_code = sd_ble_gap_tx_power_set(BLE_GAP_TX_POWER_ROLE_ADV,
4.                                                m_advertising.adv_handle,
5.                                                TX_POWER_LEVEL);
6.  APP_ERROR_CHECK(err_code);
7.  }
```

比如在主函数开头的宏定义中，定义发射功率的大小为：

```
#define  TX_POWER_LEVEL                (0)
```

如果读者不需要动态的修改发送功率，那么可以在主函数中直接调用 tx_power_set（）函数。

调用该函数后，工程编译通过提示 OK。

2.2.2 应用与调试

首先采用 nrfgo studio 软件下载协议栈，然后使用 keil 软件直接下载应用程序。应用程序下载成功后，程序开始运行，同时开发板上广播 LED 开始广播。本实验采用手机写 nrf connect app 软件，返现服务应用名为 QFDZ_power 如下图 2.2 所示，打开广播参数内容，查看设置的发射功率值：

图 2.2：发送功率设置

2.3：本章小结

本章结合了 nRF52xx 系列蓝牙 BLE 设备的应用，通过获取接收信号的强度 rssi，可以获知发射端发送到接收端的信号，接收端最终的接收强度。而发射端的信号强度则由发射功率设置函数进行直接控制。发送信号最终在传输途中会有损耗，，也就是说发射的信号功率会在发送途中衰减。这种损耗会受到很多因素的影响，其中距离是一个主要因素。因此对于接收信号强度 rssi 的大小可以用于简单的判断接收和发射双方之间的距离。本章详细的讨论了接收信号的强度 rssi 的获取和发射功率的配置。