青风带你玩蓝牙 nRF52832 系列教程

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

3.2 蓝牙主机扫描详解

前面的一篇主机讲义里谈到了主机扫描参数，特意留下来，这一节里单独详细的进行讨论。主机扫描和从机广播是对应的，从机里广播我们相信的讲过，这里来谈谈主机扫描。读者阅读的时候对照上一讲主机串口进行学习。

通过并且通过蓝牙主机串口例子，修改程序，设计主动扫描和被动扫描器。

1：主机扫描概念：

1.1 主机扫描参数配置

扫描是一个在一定范围内用来寻址其他低功耗蓝牙设备广播的过程。扫描者设备在扫描过程中会使用广播信道。与广播过程不同的是，扫描过程没有严格的时间定义和信道规则。扫描过程应该按照由 Host 层所设定扫描定时参数还运行。扫描定时参数在程序代码中有体现：

首先我们看下 Nrf52832 的主机程序如下所示，下面来分析下主机流程：

在主函数 main.c 文中 scan_start();//开始扫描

```
static void scan_start(void)//主机开始扫描
{
    uint32_t err_code;

    ret = sd_ble_gap_scan_start(&m_scan_params, &m_scan_buffer);
    APP_ERROR_CHECK(ret);

    ret = bsp_indication_set(BSP_INDICATE_SCANNING);
```

```
    APP_ERROR_CHECK(ret);
}
```

开始扫描函数里出现的两个函数： 第一个函数 sd_ble_gap_scan_start(&m_scan_params, &m_scan_buffer)是一个协议栈封装函数，也就是使用 &m_scan_params 定义的参数开始进行扫描：

```
static const ble_gap_scan_params_t m_scan_params =
  {
    .active       = SCAN_ACTIVE,//主动扫描
    .interval     = SCAN_INTERVAL,//扫描间隔
    .window       = SCAN_WINDOW,//扫描窗口
    .timeout            = SCAN_DURATION,//扫描超时时间
    .scan_phys          = BLE_GAP_PHY_1MBPS,//扫描的物理层速度
    .filter_policy      = BLE_GAP_SCAN_FP_ACCEPT_ALL,//扫描筛选策略
  };
```

这就是关于扫描参数的详细内容。下面来一一进行分析：

.active 参数表示选择扫描模式，扫描模式分为被动扫描和主动扫描。

.window 表示扫描窗口，每次扫描所持续的时间，在持续时间内，扫描设备一直在广播信道上运行。在 nrf 上设置为 0x0004 and 0x4000 in 0.625ms units (2.5ms to 10.24s).

.interval 表示扫描间隔，控制器间隔多长时间扫描一次。也就是两个连续的扫描窗口的开始时间的时间间隔。在 nrf 上设置为 0x0004 and 0x4000 in 0.625ms units (2.5ms to 10.24s)

.timeout 表示扫描超时，超过指定的时间后，没有扫描到设备将停止扫描。0x0001 and 0xFFFF in seconds, 0x0000 设置为没有 timeout.

扫描窗口和扫描间隔着两个参数非常重要。扫描窗口的设置要小于或者等于扫描间隔，并且都是 0.625ms 的整数倍，这点需要注意。那么这两个参数就决定了控制器主机的扫描占空比。比如，如果设置扫描间隔为 100ms，扫描窗口为 10ms，那么主机控制器的占空比就是 10%。特别注意可以捕获的定向广播数据包的最低占空比为 0.4%，即为没一秒中扫描时间为 3.75ms，这些时间设置在如何的蓝牙 4.x 处理器中都是一致的，并不仅仅限于 nrf 处理器。

如果把时间间隔设置为相同的大小，那么控制器会进行连续扫描，每个间隔会改变扫描频率，也就是切换扫描信道。

.filter_policy 表示扫描策略，也就是说接受任何广播数据或者仅仅接受白名单设备的广播数据包。实际上就是决定是否使用白名单。是否使用白名单过滤广播数据包。

这里注意一点，如果定向广播数据包中的目的地址并非是自己的，那么改数据必须被抛弃，即使广播数据包的发送者在自己的白名单内也不例外。官方对筛选定义如下四种情况：

#define BLE_GAP_SCAN_FP_ACCEPT_ALL 0x00
接收所有的广播包，除去广播地址不是指向该设备的定向广播。
#define BLE_GAP_SCAN_FP_WHITELIST 0x01
接收在白名单里的所有广播，除去广播地址不是指向该设备的定向广播。
#define BLE_GAP_SCAN_FP_ALL_NOT_RESOLVED_DIRECTED 0x02
接收所有的广播包，包含定向广播包。这里如果广播 MAC 地址是私密地址，这里是无法被解析的。
#define BLE_GAP_SCAN_FP_WHITELIST_NOT_RESOLVED_DIRECTED 0x03
接收白名单里的所有的广播包，包含定向广播包。这里如果广播 MAC 地址是私密地址，这里是无法被解析的。

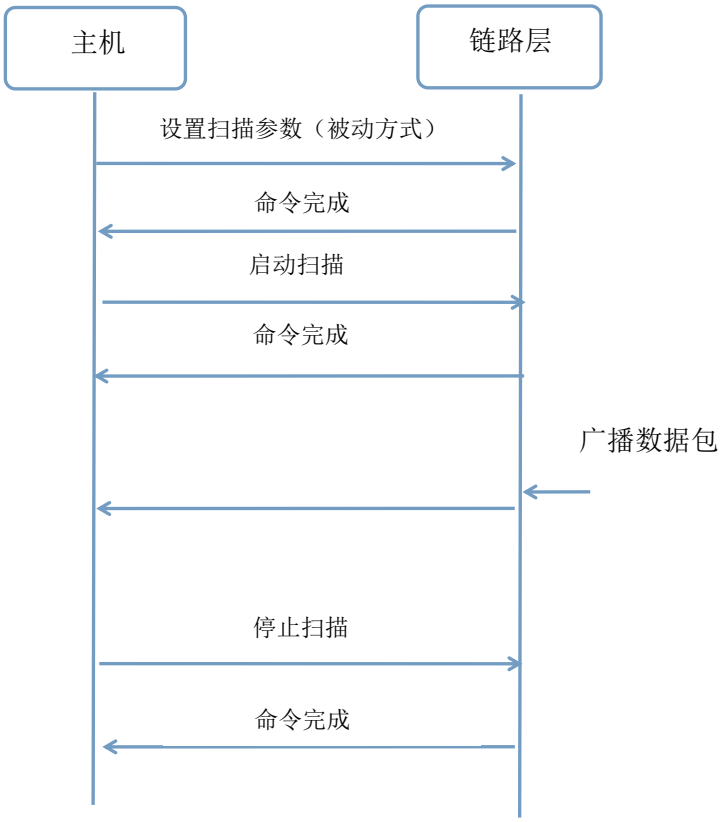
下面详细分析下.active 两种扫描模式：

1.2 被动扫描定义

被动扫描：在被动扫描中，扫描者设备应该仅仅去监听广播包，而不向广播者设备发送任何数据。

一旦上面的扫描参数设置完毕，主机就可以在协议栈中使用 LE Set Scan Enable 命令启动扫描。扫描过程中，如果控制器接收到的符合过滤策略和其他规则的广播数据包，则发送一个 LE Advertising Report 事件给主机。除了广播者的设备地址外，报告事件还包括广播数据包中的数据，以及接收广播数据包时的信号接收强度。我们可以利用该信号强度以及位于广播数据包中的发送功率，共同确定信号的路径损失，从而给出大致的范围，这个应用就是防丢器和蓝牙定位。被动扫描不需要向从机发送任何数据。

下面图表示了被动扫描广播的过程：



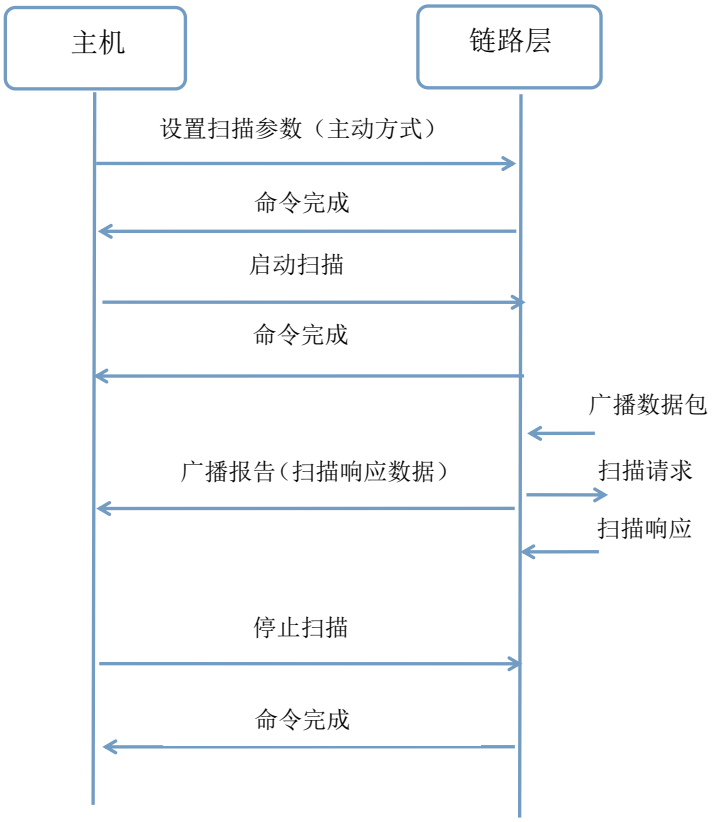
主机如果想停止广播，还是使用 LE Set Scan Enable 命令，只是参数设置为“停止扫描”

1.3 主动扫描定义

主动扫描更加复杂，主动扫描不单单可以捕获到对端设备的广播数据包，还可以捕获可能的扫描响应包。

在参数配置和启动扫描方面，主动扫描和被动扫描一致。不同点就是可以捕获扫描响应包。并且区分广播数据包和扫描响应数据包

下面图表示了主动扫描广播的过程：



控制器收到扫描数据包后将向主机发送一个广播报告事件（adv_report），该事件同样包括了链路层数据包的广播类型。因此，主机能够判断对端设备是否可以连接或者扫描，并且区分出广播数据包和扫描响应数据包。

2：主机扫描器设计：

主机扫描可以扫描广播的关键数据，在很多场合都可以加以应用，通过上节介绍的基础知识，下面来设计我们的扫描器：

2.1 被动扫描设计：

1：我们以蓝牙主机串口的实例为基础，首先需设置的是扫描参数，也就是在 scan_start 中修改为被动扫描。

```
/*
 * 扫描参数设置
 */
static ble_gap_scan_params_t const m_scan_params =
{
    .active      = 0,
```

```
    .interval        = SCAN_INTERVAL,
    .window          = SCAN_WINDOW,
    .timeout         = SCAN_DURATION,
    .scan_phys       = BLE_GAP_PHY_1MBPS,
    .filter_policy   = BLE_GAP_SCAN_FP_ACCEPT_ALL,
};
```

也就需要修改设置 .active =0

2：控制器收到扫描数据包后将向主机发送一个广播报告事件，在 nrf 官方库给了一个扫描报告的结构体,内容如下:

扫描报告结构体：

```
1.  typedef struct
2.  {
3.    /*广播报告类型 */
4.     ble_gap_adv_report_type_t   type;
5.
6.     /* 蓝牙配对设备地址，如果 peer_addr 被解析:那么参数 ble_gap_addr_t::addr_id_peer 被设
       置为 1，同时该地址是对等方的身份地址。*/
7.     ble_gap_addr_t                  peer_addr;
8.
9.     /*当参数 ble_gap_adv_report_type_t::directed 被设置为 1 ，包含目的地址的广播事件。如果
       协议栈能够解析地址， 参数 ble_gap_addr_t::addr_id_peer 被设置为 1，direct_addr 包含本
       地的的认证地址。如果广播事件的目标地址是 BLE_GAP_ADDR_TYPE_RANDOM_
10. PRIVATE_RESOLVABLE 类型，协议栈是服务解析的，应用程序可以尝试解析这个地址，以查
       明广播活动是否指向我们。 */
11.    ble_gap_addr_t                  direct_addr;
12.
13.     /* 指向被接收的主广播包的物理层 PHY*/
14.    uint8_t                         primary_phy;
15.
16.  /*指向被接收的第二个广播包的物理层 PHY。看参数 BLE_GAP_PHYS.如果在二级广播频道
       上没有收到信息包，则此字段为 0。 */
17.    uint8_t                         secondary_phy;
18.
19.  /*TX 功率报告由广播客户端在最后收到的包头。.如果最后收到的数据包不包含 Tx 字段，则这
       个字段被设置为 BLE_GAP_POWER_LEVEL_INVALID */
20.    int8_t                          tx_power;
21.
22.     /*接收的信号强度*/
23.    int8_t                          rssi;
24.
25.     /**接收到最后一个广告包的频道索引(0-39)。 */
26.    uint8_t                         ch_index;
27.
```

```
28.     /*设置接收广播数据的 ID。*/
29.    uint8_t                          set_id;
30.
31.     /*接收的广播数据的广播数据 ID 号*/
32.    uint16_t                         data_id:12;
33.
34. /*接收的广播数据或者扫描回包数据。Received advertising or scan response data. 如果参数
      ble_gap_adv_report_type_t::status 没有被设置为参数 BLE_GAP_ADV_DATA_STATUS_
35. INCOMPLETE_MORE_DATA, 参数 sd_ble_gap_scan_start 中提供的数据缓存立即释放*/
36.    ble_data_t                       data;
37.
38.  /*在此扩展广播事件中，下一个广播包的偏移量。注意：只有当@ref
      ble_gap_adv_report_type_t::status 被设置为@ref
      BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_MORE_DATA 时，该字段才会被设置。*/
39.  ble_gap_aux_pointer_t        aux_pointer;
40.  }
41.    ble_gap_evt_adv_report_t;
42.
```

●首先来谈下第一个参数 ble_gap_adv_report_type_t type,这个广播事件类型的结构体定义如下所示：

```
typedef struct
{
   uint16_t connectable      : 1; /*可连接广播事件类型。 */
   uint16_t scannable        : 1; /*可扫描广播事件类型。 */
   uint16_t directed         : 1; /*定向广播事件类型。 */
   uint16_t scan_response    : 1; /*收到扫描响应。*/
   uint16_t extended_pdu     : 1; /*收到一套加长广播。 */
   uint16_t status           : 2; /*数据状态，参考参数 BLE_GAP_ADV_DATA_STATUS. */
   uint16_t reserved         : 9; /*保留部分 */
} ble_gap_adv_report_type_t;
```

你可以通过报告输出对应参数，判断你这个扫描的广播类型，广播数据状态等参数，其中数据状态可以具有下面几种情况：

```
 /*广播活动的所有数据已经收到。*/
#define BLE_GAP_ADV_DATA_STATUS_COMPLETE                  0x00
/*需要接收的更多数据。*/
#define BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_MORE_DATA 0x01
 /*不完整的数据。缓冲区大小不足以接收更多。*/
#define BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_TRUNCATED 0x02
 /*未能接收到剩余数据。*/
#define BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_MISSED      0x03
```

●peer_addr：扫描设备的 MAC 地址

direct_addr：扫描定向广播的 MAC 地址

这两个都是扫描的设备从机地址，但是 direct_addr 是定向广播的设备 MAC 地址。

●primary_phy：主广播的物理层速度

secondary_phy：第二扩展广播的物理层速度

蓝牙 4.x 协议规定蓝牙广播数据包每包数据最大只支持 31 字节数据传输，广播信道限制在 37,38,39 三个信道。在原有的用于传输广播数据的 PDU（ADV_IND、ADV_DIRECT_IND、ADV_NONCONN_IND 以及 ADV_SCAN_IND，称作 legacy PDUs）的基础上，蓝牙 5 增加了扩展的 PDU（ADV_EXT_IND、AUX_ADV_IND、AUX_SYNC_IND 以及 AUX_CHAIN_IND，称作 extended advertising PDUs)，同时也允许蓝牙在除开 37,38,39 三个通道之外的其他 37 个信道上发送长度介于 0-255 字节的数据。

蓝牙 5.0 把广播信道抽象为两类，一种叫主广播信道(primary advertisement channels)，另一种叫次广播信道，或者第二广播信道（Secondary Advertising Packets）。

主广播信道只工作在 37，38，39 三个信道，最大广播字节为 31 字节，广播的数据类型增加了一个 ADV_EXT_IND 指令，ADV_EXT_IND 指令即为告知监听设备，我要广播大数据包广播了。ADV_EXT_IND 指令包含要在第二类次广播信道上发送的内容，第二广播信道发送广播数据的信道，物理 PHY 层，1M PHY，Coded PHY，2M PHY 等。

目前 PHY 物理层上速度设置可以如下：

```
#define BLE_GAP_PHY_AUTO                    0x00        /**< Automatic PHY selection. Refer @ref sd_ble_gap_phy_update for more information.*/
#define BLE_GAP_PHY_1MBPS                   0x01        /**< 1 Mbps PHY. */
#define BLE_GAP_PHY_2MBPS                   0x02        /**< 2 Mbps PHY. */
#define BLE_GAP_PHY_CODED                   0x04        /**< Coded PHY. */
#define BLE_GAP_PHY_NOT_SET                 0xFF        /**< PHY is not configured. */
```

●tx_power：广播发射功率

Rssi： 接收信号的强度以 DB 为单位

●ble_data_t data

```
typedef struct
{
uint8_t       *p_data;   /*广播或者扫描回应的数据*/
uint16_t       len;      /*广播或者扫描回应的数据长度*/
} ble_data_t;
```

扫描广播后，广播参数就需要通过 adv_report 报告给我们，我们可以通过串口进行输出，在蓝牙派发函数 on_ble_evt 中进行报告输出，当发生广播报告时打印输出报告内容:

```
413  static void ble_evt_handler(ble_evt_t const * p_ble_evt, void * p_context)
414  {
415      ret_code_t            err_code;
416      ble_gap_evt_t const * p_gap_evt = &p_ble_evt->evt.gap_evt;
417
418      switch (p_ble_evt->header.evt_id)
419      {
420          case BLE_GAP_EVT_ADV_REPORT:
421              on_adv_report(&p_gap_evt->params.adv_report);
422              break; // BLE_GAP_EVT_ADV_REPORT
423
```

我们可以再广播报告中添加自己的内容，如下所示：

```
ret_code_t err_code;
  //MAC 输出
NRF_LOG_INFO("Connecting to target %02x%02x%02x%02x%02x%02x",
                        p_adv_report->peer_addr.addr[0],
                        p_adv_report->peer_addr.addr[1],
                        p_adv_report->peer_addr.addr[2],
                        p_adv_report->peer_addr.addr[3],
                        p_adv_report->peer_addr.addr[4],
                        p_adv_report->peer_addr.addr[5]
                        );

     //TX 发送功耗
      NRF_LOG_INFO(" tx_power %d", p_adv_report->tx_power);
      //接收信号强度
      NRF_LOG_INFO(" rssi %d", p_adv_report->rssi);
      //广播数据或者扫描回包的长度
      NRF_LOG_INFO(" data len: %d", p_adv_report->data.len);
     //广播名称
   NRF_LOG_RAW_HEXDUMP_INFO(p_adv_report->data.p_data, p_adv_report->data.len);
```

修改完后，编译下载，下载完后，打开串口调试助手，如果外部有任何从机设备，那么主机扫描后会在 RTT Viewer 助手中报告如下图所示：

1：从机设备地址，MAC 地址

2：广播类型为 0：可连接的非定向广播

3. Sacn_rsp=0 :扫描反馈为 0,被动扫描无反馈

4：rssi：扫描的从机信号强度

5：p_data：扫描到的数据

```
0> <info> app: Connecting to target 247F1CE7DADE
0> <info> app: connectable 1
0> <info> app: scannable 1
0> <info> app: adv_report_type_t_response 0
0> <info> app:  primary_phy 1
0> <info> app:  secondary_phy 255
0> <info> app:  tx_power 127
0> <info> app:  rssi -33
0> <info> app:  data len: 28
0>  03 19 C3 03 02 01 06 03|..?.....
0>  03 0A 18 10 09 4E 6F 72|.....Nor
0>  64 69 63 5F 54 65 6D 70|dic_Temp
0>  6C 61 74 65            |late
```

2.2 主动扫描设计：

主动扫描和被动扫描的区别就是把.active 设置为：1

```
 * @brief Parameters used when scanning.扫描参数
 */
static const ble_gap_scan_params_t m_scan_params =
  {
    .active        =1,//主动扫描
    .selective     = SCAN_SELECTIVE,//忽略 NUKOWN 设备
    .p_whitelist = NULL,//白名单
    .interval      = SCAN_INTERVAL,//扫描间隔
    .window          = SCAN_WINDOW,//扫描窗口
    .timeout       = SCAN_TIMEOUT//扫描超时
  };
```

广播报告中代码修改如下所示：

```
    ret_code_t err_code;
      //MAC 输出
    NRF_LOG_INFO("Connecting to target %02x%02x%02x%02x%02x%02x",
                          p_adv_report->peer_addr.addr[0],
                          p_adv_report->peer_addr.addr[1],
                          p_adv_report->peer_addr.addr[2],
                          p_adv_report->peer_addr.addr[3],
                          p_adv_report->peer_addr.addr[4],
                          p_adv_report->peer_addr.addr[5]
                          );

  //TX 发送功耗
        NRF_LOG_INFO(" tx_power %d", p_adv_report->tx_power);
        //接收信号强度
```

```
            NRF_LOG_INFO(" rssi %d", p_adv_report->rssi);
            //广播数据或者扫描回包的长度
            NRF_LOG_INFO(" data len: %d", p_adv_report->data.len);
    //      广播名称
                if (p_adv_report->type.scan_response)
                 {
                      if (p_adv_report->data.len > 0)
                      {
                           NRF_LOG_INFO("Scan response received:");
                           NRF_LOG_RAW_HEXDUMP_INFO(p_adv_report->data.p_data,
p_adv_report->data.len);
                      }
                      else
                      {
                           NRF_LOG_INFO("Empty scan response received.");
                      }
                 }
                 else
                 {
                      NRF_LOG_INFO("Advertising packet received:");
                      NRF_LOG_RAW_HEXDUMP_INFO(p_adv_report->data.p_data,
p_adv_report->data.len);
                 }
```

修改完后，编译下载，下载方法见后面说明，下载完后，打开 RTT Viewer 调试助手，如果外部有任何从机设备，那么主机扫描后会在串口助手中报告如下图所示：

区别于被动扫描，出现了扫描回包 Sacn_rsp=1