

## 4 进入无线世界

Nordic 的 nRF24 系列中短距离通讯芯片，广泛运用在无线鼠标、无线键盘等设备。

从这一刻起，你将进入充满魔力的无线世界！

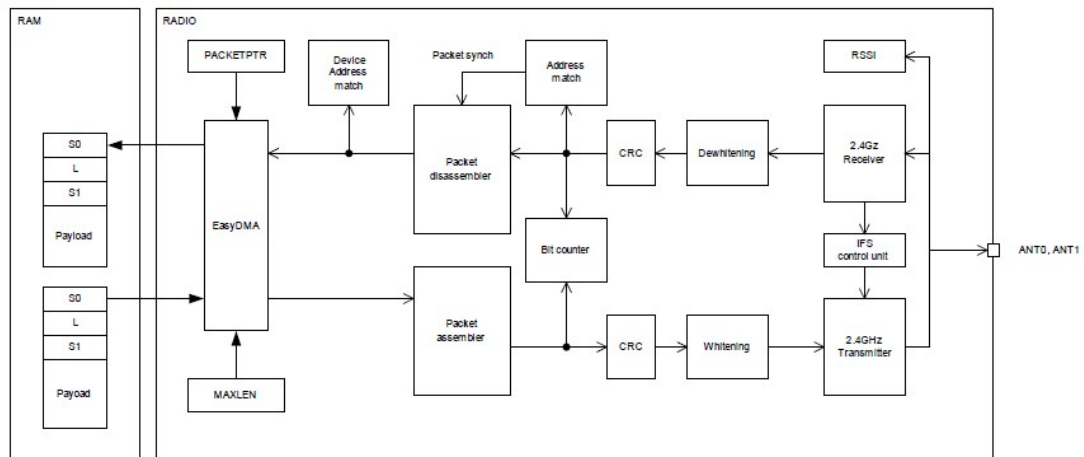
### 4.1 nRF51822 的 Radio

nRF51 系列的 2.4GHz 无线收发器专门为 2.400 到 2.4835GHz 的 ISM 频段优化过。无线调制方式和 packet 结构使这款无线收发器可以用于 Bluetooth® low energy (BLE), ANT™, Enhanced ShockBurst™, 和其他 2.4GHz 协议。无线收发器的接收和发射数据直接在系统内存中存取，极大的增加了灵活性和包数据管理的效率。

- 基本的调制方式
- GFSK modulation
- 数据白化（加噪）
- 7 bit linear feedback shift register (programmable IV)
- On-air data rates
- 250 kbps
- 1 Mbps
- 2 Mbps
- Transmitter with programmable output power of +4 dBm to -20 dBm, in 4 dB steps
- Transmitter whisper mode: -30 dBm
- RSSI function (1 dB resolution,  $\pm 6$  dB accuracy)
- Receiver with integrated channel filters achieving maximum sensitivity
- -96 dBm @ 250 kbps
- -93 dBm @ 1 Mbps BLE
- -90 dBm @ 1 Mbps
- -85 dBm @ 2 Mbps
- RF Synthesizer
- 1 MHz frequency programming resolution
- 1 MHz non-overlapping channel spacing at 1 Mbps and 250 kbps
- 2 MHz non-overlapping channel spacing at 2 Mbps
- Works with low-cost  $\pm 60$  ppm 16 MHz crystal oscillators
- Baseband controller
- EasyDMA3 RX and TX packet transfer directly to and from RAM
- Dynamic payload length
- On-the-fly packet assembly/disassembly and AES CCM payload encryption
- 8 bit, 16 bit, and 24 bit CRC check (programmable polynomial and initial value)

*EasyDMA - is an integrated DMA implementation requiring no configuration to take advantage of flexible data management and avoid copy operations to and from RAM.*

### 4.1.1 RADIO 的框图



### 4.1.2 关于 EasyDMA

## 4.2 RADIO 的寄存器

## 4.3 官方 led\_radio 代码样例

官方的 led\_radio 代码样例分为 **rx** 端和 **tx** 端。其中 nRF51822 AK 对应 **rx** 端，使用 Nordic\nrf51822\Board\pca10001\led\_radio\_example 内的代码；nRF51822 USB Dongle 对应 **tx** 端，使用 Nordic\nrf51822\Board\pca10000\led\_radio\_example 内的代码。

本例中程中需要使用 nRF51822 AK 和 nRF51822 USB Dongle 两个设备，其中 USB Dongle 需要与 PC 端串口调试软件相连（详见第三章“Hello, World”）。

*注意：根据硬件情况选择是否使能硬件 flow control。*

### 4.3.1 tx 端（USB dongle）main 函数请看：

```
int main(void)
{
    init();

    simple_uart_putstring((const uint8_t *)"\n\rPress '0' or '1': ");
    while(true)
    {
        uint8_t c = simple_uart_get();
        if (c != '0' && c != '1')
            continue;
        simple_uart_put(c);
        // Place the read character in the payload, enable the radio and
        // send the packet:
    }
}
```

```

packet[0] = c;
NRF_RADIO->EVENTS_READY = 0U;
NRF_RADIO->TASKS_TXEN = 1;
while (NRF_RADIO->EVENTS_READY == 0U)
{
}
NRF_RADIO->TASKS_START = 1U;
NRF_RADIO->EVENTS_END = 0U;
while(NRF_RADIO->EVENTS_END == 0U)
{
}
NRF_RADIO->EVENTS_DISABLED = 0U;
// Disable radio
NRF_RADIO->TASKS_DISABLE = 1U;
while(NRF_RADIO->EVENTS_DISABLED == 0U)
{
}
}
}

```

### 4.3.2 rx 端（AK board）main 函数请看：

```

int main(void)
{
    init();

    while(true)
    {
        // Set payload pointer
        NRF_RADIO->PACKETPTR = (uint32_t)packet;
        NRF_RADIO->EVENTS_READY = 0U;
        // Enable radio and wait for ready
        NRF_RADIO->TASKS_RXEN = 1U;
        while(NRF_RADIO->EVENTS_READY == 0U)
        {
        }
        NRF_RADIO->EVENTS_END = 0U;
        // Start listening and wait for address received event
        NRF_RADIO->TASKS_START = 1U;
        // Wait for end of packet
        while(NRF_RADIO->EVENTS_END == 0U)
        {
        }
        // Write received data to LED0 and LED1 on CRC match
        if (NRF_RADIO->CRCSTATUS == 1U)
        {
            switch(packet[0])
            {
                case '0':
                    nrf_gpio_pin_set(LED0);
                    nrf_gpio_pin_clear(LED1);
                    break;

                case '1':
                    nrf_gpio_pin_set(LED1);
                    nrf_gpio_pin_clear(LED0);
                    break;
            }
        }
    }
}

```

```

    }
    NRF_RADIO->EVENTS_DISABLED = 0U;
    // Disable radio
    NRF_RADIO->TASKS_DISABLE = 1U;
    while(NRF_RADIO->EVENTS_DISABLED == 0U)
    {
    }
}
}

```

### 4.3.3 radio\_configure 函数请看:

```

void radio_configure()
{
    // Radio config
    NRF_RADIO->TXPOWER = (RADIO_TXPOWER_TXPOWER_0dBm <<
RADIO_TXPOWER_TXPOWER_Pos);
    NRF_RADIO->FREQUENCY = 7UL;           // Frequency bin 7, 2407MHz
    NRF_RADIO->MODE = (RADIO_MODE_MODE_Nrf_2Mbit << RADIO_MODE_MODE_Pos);

    // Radio address config
    NRF_RADIO->PREFIX0 = 0xC4C3C2E7UL; // Prefix byte of addresses 3 to 0
    NRF_RADIO->PREFIX1 = 0xC5C6C7C8UL; // Prefix byte of addresses 7 to 4
    NRF_RADIO->BASE0 = 0xE7E7E7E7UL; // Base address for prefix 0
    NRF_RADIO->BASE1 = 0x00C2C2C2UL; // Base address for prefix 1-7
    NRF_RADIO->TXADDRESS = 0x00UL; // Set device address 0 to use when
transmitting
    NRF_RADIO->RXADDRESSES = 0x01UL; // Enable device address 0 to use
which receiving

    // Packet configuration
    NRF_RADIO->PCNF0 = (PACKET0_S1_SIZE << RADIO_PCNF0_S1LEN_Pos) |
(PACKET0_S0_SIZE << RADIO_PCNF0_S0LEN_Pos) |
(PACKET0_PAYLOAD_SIZE << RADIO_PCNF0_LFLEN_Pos);
//lint !e845 "The right argument to operator '|' is certain to be 0"

    // Packet configuration
    NRF_RADIO->PCNF1 = (RADIO_PCNF1_WHITEEN_Disabled <<
RADIO_PCNF1_WHITEEN_Pos) |
(RADIO_PCNF1_ENDIAN_Big << RADIO_PCNF1_ENDIAN_Pos)
|
(PACKET1_BASE_ADDRESS_LENGTH <<
RADIO_PCNF1_BALEN_Pos) |
(PACKET1_STATIC_LENGTH << RADIO_PCNF1_STATLEN_Pos)
|
(PACKET1_PAYLOAD_SIZE << RADIO_PCNF1_MAXLEN_Pos);
//lint !e845 "The right argument to operator '|' is certain to be 0"

    // CRC Config
    NRF_RADIO->CRCCNF = (RADIO_CRCCNF_LEN_Two << RADIO_CRCCNF_LEN_Pos); //
Number of checksum bits
    if ((NRF_RADIO->CRCCNF & RADIO_CRCCNF_LEN_Msk) == (RADIO_CRCCNF_LEN_Two
<< RADIO_CRCCNF_LEN_Pos))
    {
        NRF_RADIO->CRCINIT = 0xFFFFUL; // Initial value
        NRF_RADIO->CRCPOLY = 0x11021UL; // CRC poly: x^16+x^12+x^5+1
    }
}

```

```
else if ((NRF_RADIO->CRCCNF & RADIO_CRCCNF_LEN_Msk) ==  
(RADIO_CRCCNF_LEN_One << RADIO_CRCCNF_LEN_Pos))  
{  
    NRF_RADIO->CRCINIT = 0xFFUL;           // Initial value  
    NRF_RADIO->CRCPOLY = 0x107UL;          // CRC poly:  $x^8+x^2+x^1+1$   
}  
  
nrf_delay_ms(3);  
}
```

## 4.4 演示效果

敲击键盘上的“0”或“1”后，串口数据会发送到 nRF51822 USB Dongle。nRF51822 接收后会回传到串口终端，并且发送无线信号到 nRF51822 AK board，板上对应的 LED“0”或“1”将会亮起。