青风带你玩蓝牙 nRF51822 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-nRF51822 开发板

3.1 点亮你的第一个 LED 灯

在讲第一个实例之前，我要先对许多初学蓝牙的朋友说明几个关键的学习问题：

首先是学习资料的准备，在新的处理器出来后，我们要如何入门，如何进行开发，这时相关的技术手册就是必须的了，以后我们的讲解与分享中都会回到技术手册，来分析下如何采用手册做到空手入门，实际上这也是工程师的必经之路。

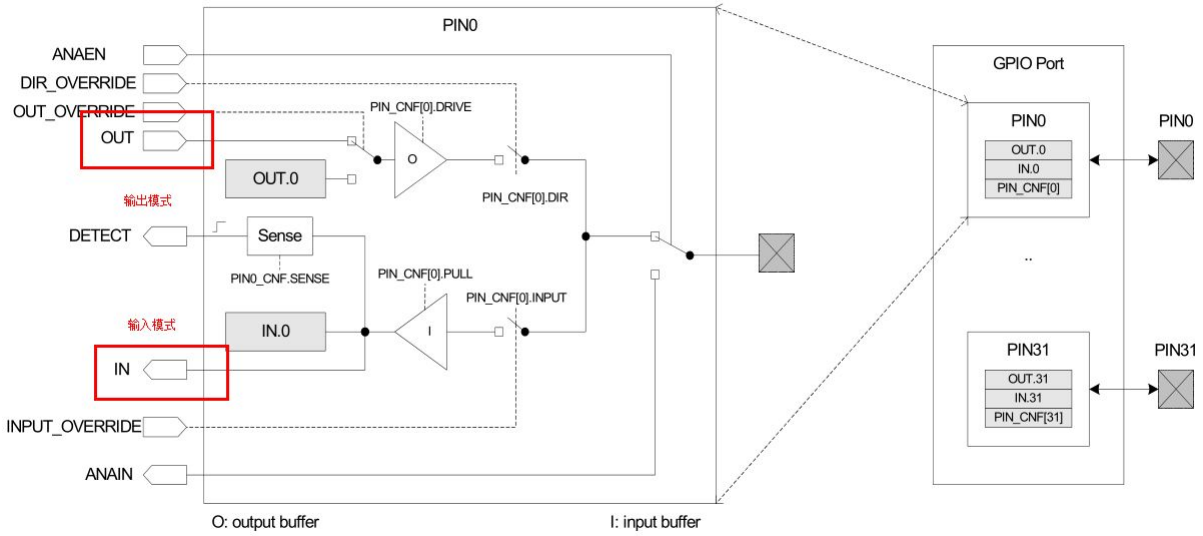
蓝牙技术属于无线射频技术，协议栈的使用十分重要，对协议栈的认识和工作原理的理解是十分重要的，这一点需要大家多花时间多花精力了，我们给的教程只是带领大家步入蓝牙的门，所谓学的更多知道的越少，如果你希望成为一个蓝牙技术高手需要学的东西很多。

3.1.1 原理分析：

大家之前学习过 51 单片机，也使用过 IO 口。如果学过了 ARM CORTEX M0，M0 的 IO 口配置与 51 单片机有点区别，51 不需要设置 IO 口为输入或者输出，而 CORTEX M0 的 IO 口有多种状态需要设置，nRF51822 实际上就是 CORTEX M0 内核。那么下面我们一一介绍：

首先看看 IO 口的模式，查看 nRF51822 参考手册，端口可以配置为 4 种模式：输

入模式，输出模式，复用模式，模拟通道模式。由于 nRF51822 的 IO 管脚复用了其它的外设功能，比如 I2C, SPI, UART 等。而通用 IO 口具有输入和输出模式：



实际 nRF51822 上所包含的寄存器是非常的简单的，相比之前我们学过的同样是 cortex m0 内核的 stm32f051 相比其 IO 寄存器要少很多，下面我把数据手册的手册进行了标注，下面我们就一一介绍：

Register	Offset	Description
REGISTERS		
OUT	0x504	Write GPIO port
OUTSET	0x508	Set individual bits in GPIO port
OUTCLR	0x50C	Clear individual bits in GPIO port
IN	0x510	Read GPIO port
DIR	0x514	Direction of GPIO pins
DIRSET	0x518	Setting DIR register
DIRCLR	0x51C	Clearing DIR register
PIN_CNF[0]	0x700	Configuration of pin 0
..	..	..
PIN_CNF[31]	0x77C	Configuration of pin 31

端口输出（写）
写1或者写0
端口输入（读）
IO口方向设置
方向置位
方向清零
对应0到31端口位设置

Table 14 Register overview

大家发现一个非常有趣的现象没，MO 内核的处理器很多厂家都喜欢用 51 做位代号。

言归正传，如果大家使用 nRF51822 官方提供的库函数编程，可以在"nrf_gpio.h"库文件中找到设置 IO 模式的结构体 nrf_gpio_port_dir_t，这里完全是对照参考手册进行编写的，实际上 nRF51822 的库是比较简单的，大家完全可以直接用寄存器操作 ：

```
01. typedef enum
02. {
03.     NRF_GPIO_PORT_DIR_OUTPUT,         ///<  Output
04.     NRF_GPIO_PORT_DIR_INPUT           ///<  Input
```

```
05. } nrf_gpio_port_dir_t;
```

首先我们来介绍下输入和输出模式也就是 NRF_GPIO_PORT_DIR_INPUT 和 NRF_GPIO_PORT_DIR_OUTPUT。

其中输出模式寄存器为推挽输出。

再来谈谈输入模式，输入的模式可以分为上拉和下拉模式，这就比较简单了，同样大家使用库函数编程的时候，可以在"nrf_gpio.h"文件中找到设置输入模式的结构体 nrf_gpio_pin_pull_t ，如数据手册上描述：

C	RW	PULL		
		DISABLED	0	No pull
ID	**RW**	**Field**	**Value ID**	**Value**	**Description**
			PULLDOWN	1	Pull down on pin
			PULLUP	2	Pull up on pin

```
06. typedef enum
07. {
08.   NRF_GPIO_PIN_NOPULL    = GPIO_PIN_CNF_PULL_Disabled,     //无上拉下拉
      NRF_GPIO_PIN_PULLDOWN = GPIO_PIN_CNF_PULL_Pulldown,       // 下拉
      NRF_GPIO_PIN_PULLUP    = GPIO_PIN_CNF_PULL_Pullup,        //上拉
09. } nrf_gpio_pin_pull_t;
```

E	RW	SENSE		Pin sensing mechanism
		DISABLED	0	Disabled
		HIGH	1	Sense for high level
		LOW	2	Sense for low level

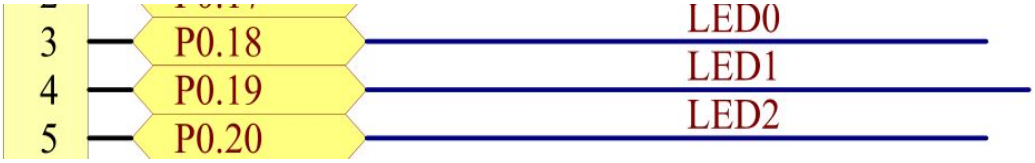
并且对于输入管脚可以设置不同的 sense 级别，如手册上所描述：

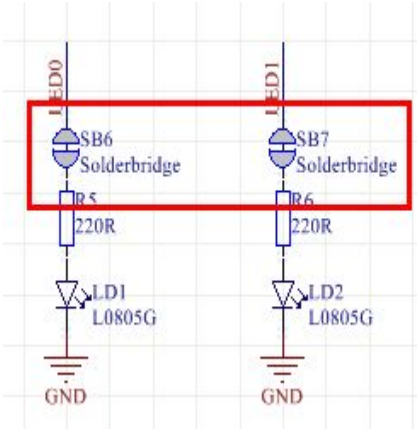
```
10. typedef enum
11. {
12.     NRF_GPIO_PIN_NOSENSE      = GPIO_PIN_CNF_SENSE_Disabled,
    ///<  Pin sense level disabled.
13.     NRF_GPIO_PIN_SENSE_LOW   = GPIO_PIN_CNF_SENSE_Low,
    ///<  Pin sense low level.
14.     NRF_GPIO_PIN_SENSE_HIGH = GPIO_PIN_CNF_SENSE_High,
    ///<  Pin sense high level.
15. } nrf_gpio_pin_sense_t;
```

关于其数字的复用功能会在相应外设里进行介绍，我们这里只是谈通用 IO 口，也就是 GPIO，现在就来点亮一个 LED 灯，学过 stm32f051 的同学可以回忆下，stm32f051 是通过通过设置 IO 端口为输出来点亮 LED 等。同样内核的 nRF51822 设置上基本是一致的，这对于学过 051 的同学来说上手是十分迅速的。

3.1.2 硬件准备：

如下图所示： 青云 QY-nRF51822 开发板上，通过管脚 P0.18 和管脚 P0.19 连接 2 个 LED 灯，我们下面的任务首先来点亮它。IO 管脚接分别接一个发光二级管，因此当把 IO 管脚定义为输出高电平的时候，就可以点亮发光二极管了。图中 SB6 和 SB7 分别为两个外设隔离点，当不需要使用 LED 灯的时候可以把这两个点割开。





PCB 板上如图所示：

2.1.3 软件准备以及编写：

按照前哨篇里的介绍，首先建立一个工程项目，采用库函数来在驱动 IO 口首先要添加几个驱动库，如下图所示：

上图 lib 组下红色框框中的几个文件都是我们需要编写库函数，可以在后面的工程

中使用。那边用户在使用中，只需要编写 main.c 主函数就 OK，整个工程项目大家如果加入分层的思想那么就对之后的移植非常有利。写发和我之前写的 stm32f051 一致，移植性较好。打个比方:底层和应用程隔离。底层驱动和应用层无关，main.c 使用的函数在 led.c 驱动中已经些好，这些才和硬件有关，这是需要移植到不同硬件时，main 主函数是可以不做任何修改的，只需要修改和底层相关的 led.c 驱动。

下面来分析下 led .c 的驱动编写：

```
16. void LED_Init(void)
17. {
18.    nrf_gpio_cfg_output(LED_0);
19.    nrf_gpio_cfg_output(LED_1);
20. }
21.
22. void LED1_Open(void)
23. {
24.    nrf_gpio_pin_set(LED_0);
25.                  }
26.
27. void LED1_Close(void)
28. {
29.  nrf_gpio_pin_clear(LED_0);
30. }
31. void LED1_Toggle(void)
32. {
33.    nrf_gpio_pin_toggle(LED_0);
34. }
35.
```

上面的函数中 nrf_gpio_cfg_output 函数和函数在 stm32f0xx_gpio.c 驱动文件中所定义了。分别表示复位和置位相关 IO 管脚。

那么主函数的编写就比较简单了，我们需要调用下面 2 个头文件，才能够直接使用我们定义的子函数。如下使用 LED_Open()函数就能够点亮一个 LED 灯了，是不是很简单。

```
36. #include <stdbool.h>
37. #include <stdint.h>
38. #include "nrf_delay.h"
39. #include "nrf_gpio.h"
40. #include "led.h"
41.
42. int main(void)
43. {
44.    //初始化 led 灯
45.    LED_Init();
46.    while(true)
47.    {
```

```
48.         LED1_Open();
49.         LED2_Close();
50.     }
51. }
```

那么加入一个小的延迟 delay 函数和打开与关闭 LED 子函数相结合，就可以实现 LED 闪烁的功能了，写一个软件延迟，函数如下所示：

```
52. #include <stdbool.h>
53. #include <stdint.h>
54. #include "nrf_delay.h"
55. #include "nrf_gpio.h"
56. #include "led.h"
57.
58. int main(void)
59. {
60.     //初始化 led 灯
61.     LED_Init();
62.     while(true)
63.     {
64.         LED1_Open();
65.         LED2_Close();
66.         nrf_delay_ms(500); //延迟 500ms
67.         LED2_Open();
68.         LED1_Close();
69.         nrf_delay_ms(500);
70.     }
71. }
72.
```

下载到 青云 QY-nRF51822 蓝牙开发板上运行后的效果如下图所示，按下复位键，LED 开始闪烁：

如图所示，按下复位键后，上方的用户 led 灯不停闪烁