

实验 1 - GPIO 输出控制 LED

1. 实验目的

掌握 NRF52832 的 GPIO 的配置方式和输出控制。

2. 实验内容

使用 NRF52832 的 GPIO P0.17 输出控制 LED 指示灯 D1 的亮灭。

程序运行后，开发板上的 D1 指示灯以 200ms 的间隔闪烁。

3. 实验设备

硬件	
1.	IK-52832DK 开发板
2.	USB MINI 数据线
3.	JLINK 仿真器
4.	JTAG-SWD 转接板、排线
软件	
1.	win7/win8.1 系统
2.	MDK5.18A 集成开发环境

4. 实验原理

4.1. 电路原理

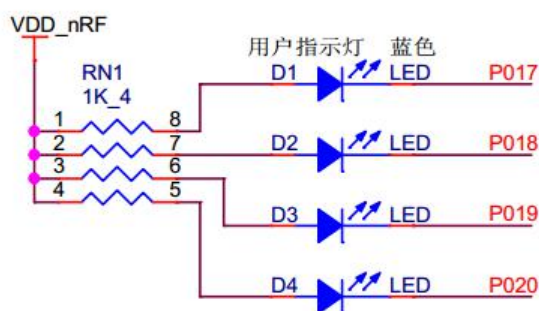


图 1：指示灯驱动电路

开发板上配置的四个用户指示灯 D1、D2、D3、D4，分别有 GPIO P0.17、P0.18、P0.19 和 P0.20 控制，当 GPIO 输出高电平时，LED 两端电压相等，LED 上没有电流流过，LED 处于熄灭状态，当 GPIO 输出低电平时，LED 两端存在正向压差，电流流过 LED，LED 被

点亮。

4.2. GPIO 配置

本实验中，GPIO P0.17 用于驱动指示灯 D1 的亮灭，所以需要将 P0.17 配置为输出。之后通过操作寄存器 OUTSET 和 OUTCLR 控制指示灯的亮灭。

- 通过 PIN_CNF[n] 寄存器配置 P0.17 为输出。
- 通过向 OUTSET 寄存器相应的位写 1 让 P0.17 输出高电平，熄灭指示灯。
- 通过向 OUTCLR 寄存器相应的位写 1 让 P0.17 输出低电平，点亮指示灯。

各个寄存器的功能如下：

- PIN_CNF[n]：GPIO 配置寄存器，其中的 n 代表硬件的引脚 0~31。每一个 PIN 单独对应一个该寄存器，其主要用来配置引脚的方向、上拉下拉、驱动配置、以及检测配置等。
- OUTSET：0~31 位对应于管脚 P0.00~P0.31，如要某个管脚输出高电平，向对应的位写“1”即可，写“0”无效。
- OUTCLR：0~31 位对应于管脚 P0.00~P0.31，如要某个管脚输出低电平，向对应的位写“1”即可，写“0”无效。

按照上述的描述，配置 P0.17 的程序如下：

```
nrf_gpio_cfg_output(LED_1); //配置 P0.17 为输出
```

其中 LED_1 是宏定义：#define LED_1 17

进入 nrf_gpio_cfg_output 函数，其代码如下：

```
__STATIC_INLINE void nrf_gpio_cfg_output(uint32_t pin_number)
{
    nrf_gpio_cfg(
        pin_number,
        NRF_GPIO_PIN_DIR_OUTPUT,
        NRF_GPIO_PIN_INPUT_DISCONNECT,
        NRF_GPIO_PIN_NOPULL,
        NRF_GPIO_PIN_S0S1,
        NRF_GPIO_PIN_NOSENSE);
}
```

上述代码执行后，GPIO P0.17 被配置为：

- 电平检测禁止。
- 标准驱动。
- 上拉禁止。
- 输入缓冲断开。
- 方向配置为输出。

4.3. 驱动 LED

GPIO P0.17 配置好后，即可通过 P0.17 输出高低电平控制指示灯 D1 的亮灭，调用下列函数即可实现 P0.17 输出高低电平。

1. `nrf_gpio_pin_set(LED_1);`

进入该函数，其代码如下：

```
_STATIC_INLINE void nrf_gpio_pin_set(uint32_t pin_number)
{
    NRF_GPIO->OUTSET = (1UL << pin_number);
}
```

从上面的代码可以看出，该函数正是通过向 **OUTSET** 寄存器相应的位写“1”来实现 GPIO 输出高电平的，和我们前面描述的一致。

2. `nrf_gpio_pin_clear(LED_1);`

功能：根据输入的管脚号设置该管脚输出低电平。

3. `nrf_gpio_pin_toggle(LED_1);`

功能：根据输入的管脚号翻转该管脚的输出状态。在对管脚状态取反时调用该函数更方便。

5. 开发板电路连接

本实验需要用跳线帽短接 P17 管脚，如下图红框所示：

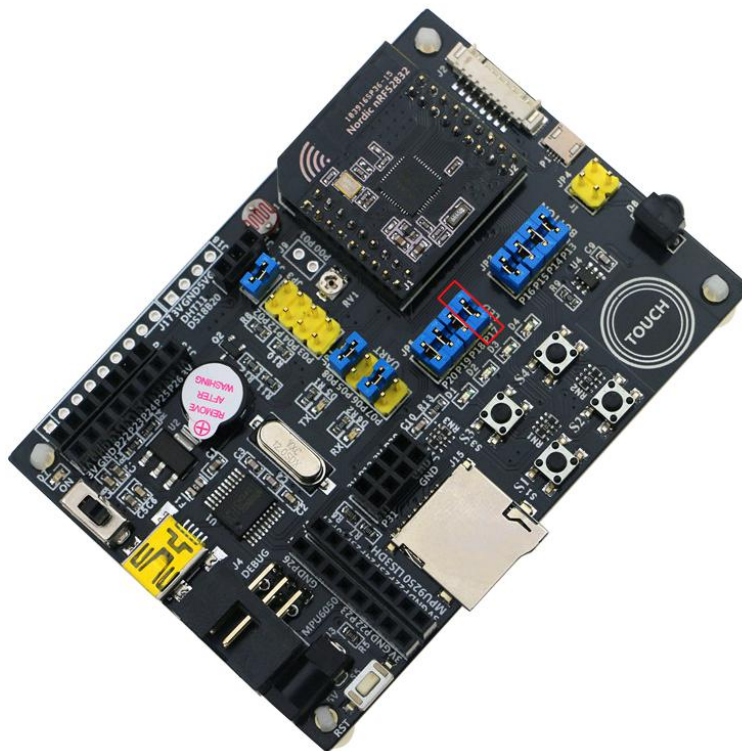





图 2：开发板跳线连接

6. 实验步骤

- 拷贝出“...\6 - 开发板应用\3 - 基础实验\实验1 - GPIO 输出控制 LED”目录下的 LED 文件夹，存放到合适的目录，如“D:\NRF52832”。**强烈建议不要在资料包中直接打开工程，因为包含了中文路径且工程路径较深，可能会出现问題。**
- 启动 MDK5.18A。
- 在 MDK5 中执行“Project→Open Project”打开“...\LED\project\”目录下的工程“LED.uvproj”。

- 点击编译按钮编译工程 。注意查看编译输出栏，观察编译的结果，如果有错误，修改程序，直到编译成功为止。编译后生成的 HEX 文件“LED.hex”位于工程目录下的“Objects”文件夹中。

```
linking...
Program Size: Code=408 RO-data=224 RW-data=4 ZI-data=2052
FromELF: creating hex file...
".\_build\led.axf" - 0 Error(s), 0 Warning(s). 错误: 0, 警告: 0表示编译通过
Build Time Elapsed: 00:00:04
```

- 点击下载按钮下载程序 。如果需要对程序进行仿真，点击 Debug 按钮  即可将程序下载到 NRF52832 进行仿真。
- 程序运行后，开发板上的 D1 指示灯以 200ms 的间隔闪烁。

7. 实验程序

```
/*
*****
* 描 述 : main 函数
* 入 参 : 无
* 返回值 : 无
*****
*/

int main(void)
{
    //配置用于控制 LED 指示灯的管脚
    nrf_gpio_cfg_output(LED_1); //配置 P0.17 为输出
    nrf_gpio_pin_set(LED_1);    //指示灯 D1 初始状态为熄灭

    while (true)
    {
        //指示灯 D1 以 200ms 的间隔闪烁
        nrf_gpio_pin_toggle(LED_1);
        nrf_delay_ms(200);
    }
}
```