


风驰教你从零开始操作 STM8 寄存器

自从风驰开源 STM8 基于库的操作例程和教程，深受广大网友喜欢，应广大网友的要求，风驰继续编写 STM8 基于寄存器的例程和教程。

如果你是一位新手，照着我的步骤来操作，三天必然入门 STM8，熟悉 STM8 的寄存器编程。风驰所有的例程都是在 IAR 环境下编写的。

当你安装好 IAR，在 IAR 安装的目录下，你可以找到 iostm8s207rb.h。

 C:\Program Files\IAR Systems\Embedded Workbench 6.0 Evaluation\stm8\inc

这是我安装 IAR 的根目录。这里面有所有 STM8 的头文件，根据你所使用的芯片型号，在你建的工程中添加相对应的头文件，在风驰 STM8 开发板上是选用 STM8S207RB 这个型号的芯片，所有我就添加 iostm8s207rb.h 这个头文件。

那如何去用这个头文件呢？又是如何去编程呢？下面我就从工程模板讲起。先看看 iostm8s207rb.h 里面的东西。

```

/*-----
*   Port A register definitions
*-----*/
/* Port A data output latch register */
#ifdef __IAR_SYSTEMS_ICC__
typedef struct
{
    unsigned char ODR0    : 1;
    unsigned char ODR1    : 1;
    unsigned char ODR2    : 1;
    unsigned char ODR3    : 1;
    unsigned char ODR4    : 1;
    unsigned char ODR5    : 1;
    unsigned char ODR6    : 1;
    unsigned char ODR7    : 1;
} __BITS_PA_ODR;
#endif
__IO_REG8_BIT(PA_ODR, 0x5000, __READ_WRITE, __BITS_PA_ODR);

/* Port A input pin value register */
#ifdef __IAR_SYSTEMS_ICC__
typedef struct
{
    unsigned char IDR0    : 1;
    unsigned char IDR1    : 1;
    unsigned char IDR2    : 1;
    unsigned char IDR3    : 1;
    unsigned char IDR4    : 1;
    unsigned char IDR5    : 1;
    unsigned char IDR6    : 1;
    unsigned char IDR7    : 1;
} __BITS_PA_IDR;
#endif
__IO_REG8_BIT(PA_IDR, 0x5001, __READ, __BITS_PA_IDR);

```

这是里面的一小部分内容，在 iostm8s207rb.h 里面有很多结构体或者是宏定义，都是被封装好的。以上面两个结构体为例，说明其作用。

__BITS_PA_ODR 是定义了 PORTA 的数据输出寄存器，其中的

unsigned char ODR0 ~ ODR7 是定义了 8 个位变量，每个位变量占 1 位的空间，整个结构体占一个字节，表示的是一个 8 位的寄存器。

定义了这样一个表示 PORTA 口数据输出寄存器的结构体之后，还得把它跟 PORTA 口数据输出寄存器所占用的内存地址联系起来，这个工作由下面这个宏定义完成：

__IO_REG8_BIT(PA_ODR, 0x5000, __READ_WRITE, __BITS_PA_ODR);

跟踪进去可以看到这个宏定义的原型：

```

/*-----
 * Define NAME as an I/O reg with bit accesss
 * Access of 8 bit reg:  NAME
 * Access of bit(s):    NAME_bit.noXX
 *-----*/
#define __IO_REG8_BIT(NAME, ADDRESS, ATTRIBUTE, BIT_STRUCT) \
    __near __no_init volatile ATTRIBUTE union \
    { \
        unsigned char NAME; \
        BIT_STRUCT NAME ## _bit; \
    } @ ADDRESS;

```

可以看到这是一个联合体,其中通过宏定义传递过来的 **NAME** 被定义为一个 **unsigned char** 类型或是前面所定义的位结构体,其实质是一样的,都是对这一个字节的空间进行操作,只是方式不同而已。

__near 属性限制了指针可以指向的地址范围;

__no_init 属性用于禁止系统启动时变量的初始化;

@符号用于指定地址

所以, **__IO_REG8_BIT(PA_ODR, 0x5000, __READ_WRITE, __BITS_PA_ODR);**

这个宏定义的作用就是在联合体中将 **NAME** 定义成 **unsigned char** 类型或是 **BIT_STRUCT** 结构体,并把这个联合体的地址指定为 **PORTA** 数据输出寄存器的地址 **0x5000**。

通过查寄存器手册可以看到 **PORTA** 的数据输出寄存器的地址就是 **0x5000**。所以,这个宏定义将两者联系起来。

这样处理之后,我们可以对 **PA_ODR** 进行操作。例如 **PA_ODR=0xff**; 这种操作方式就是对一个 8 位的寄存器进行读写。

同理, **__IO_REG8_BIT(PA_IDR, 0x5001, __READ, __BITS_PA_IDR);** 这个函数说明 **PA** 的输入结果寄存器的地址为 **0x5001**, 只可读取。例如 **temp= PA_IDR**。这种操作方式就是对一个 8 位的寄存器进行读。

再往下看:

```

#define PA_ODR_ODR0 PA_ODR_bit.ODR0
#define PA_ODR_ODR1 PA_ODR_bit.ODR1
#define PA_ODR_ODR2 PA_ODR_bit.ODR2
#define PA_ODR_ODR3 PA_ODR_bit.ODR3
#define PA_ODR_ODR4 PA_ODR_bit.ODR4
#define PA_ODR_ODR5 PA_ODR_bit.ODR5
#define PA_ODR_ODR6 PA_ODR_bit.ODR6
#define PA_ODR_ODR7 PA_ODR_bit.ODR7

#define PA_IDR_IDR0 PA_IDR_bit.IDR0
#define PA_IDR_IDR1 PA_IDR_bit.IDR1
#define PA_IDR_IDR2 PA_IDR_bit.IDR2
#define PA_IDR_IDR3 PA_IDR_bit.IDR3
#define PA_IDR_IDR4 PA_IDR_bit.IDR4
#define PA_IDR_IDR5 PA_IDR_bit.IDR5
#define PA_IDR_IDR6 PA_IDR_bit.IDR6
#define PA_IDR_IDR7 PA_IDR_bit.IDR7

```

这个就是对一个 8 位的寄存器进行位的宏定义,有了这个宏定义,我们就可以进行位操作了,这样就可以封装成 51 的编程风格。 **PD0_out=~PD0_out;**

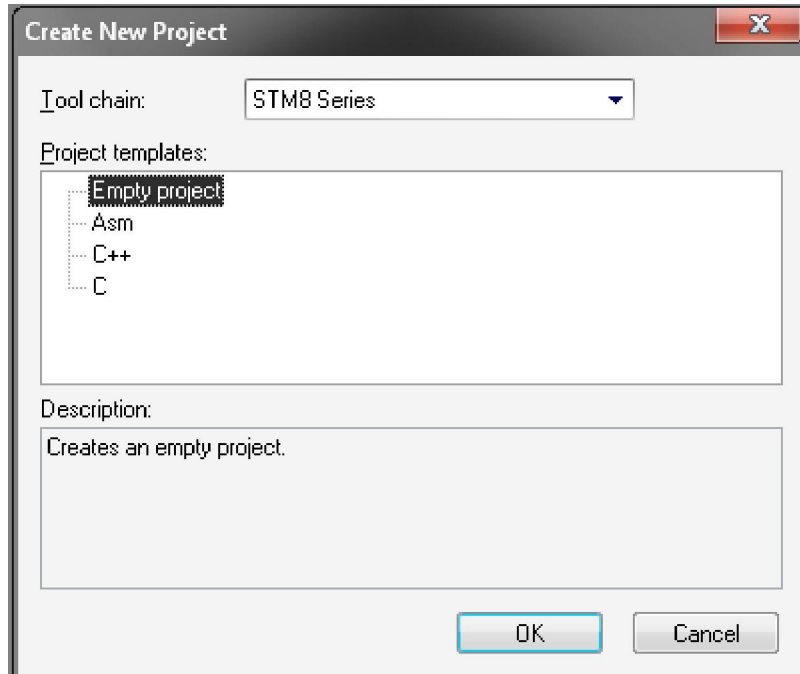
在 **iostm8s207rb.h** 里面全部是这种手法对寄存器进行封装,大家应该明白了吧。当然,大家在编程时最好还是要有 **STM8S 微控制器参考手册.pdf**, 在编程之

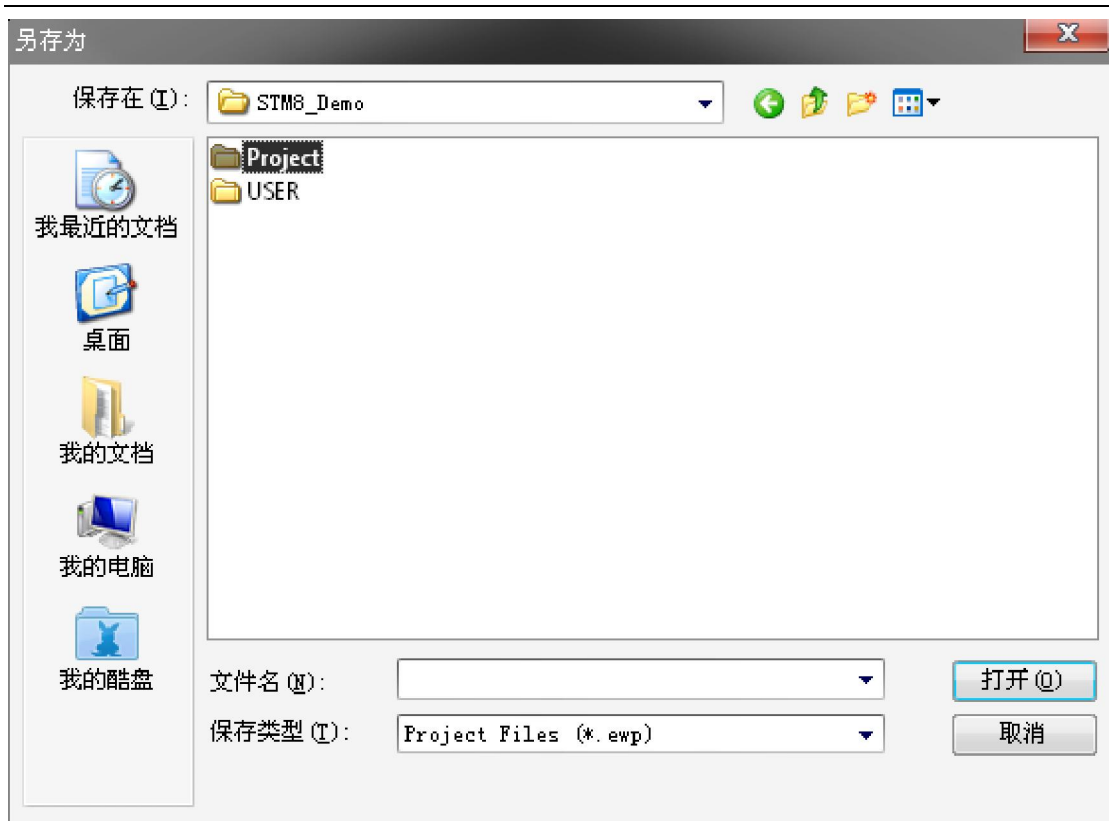
前要看一个每个寄存器表示什么意思。是设置什么用的。

明白了 IAR 中对寄存器的定义之后，我们可以创建自己的工程模板并使用这些定义进行编程。

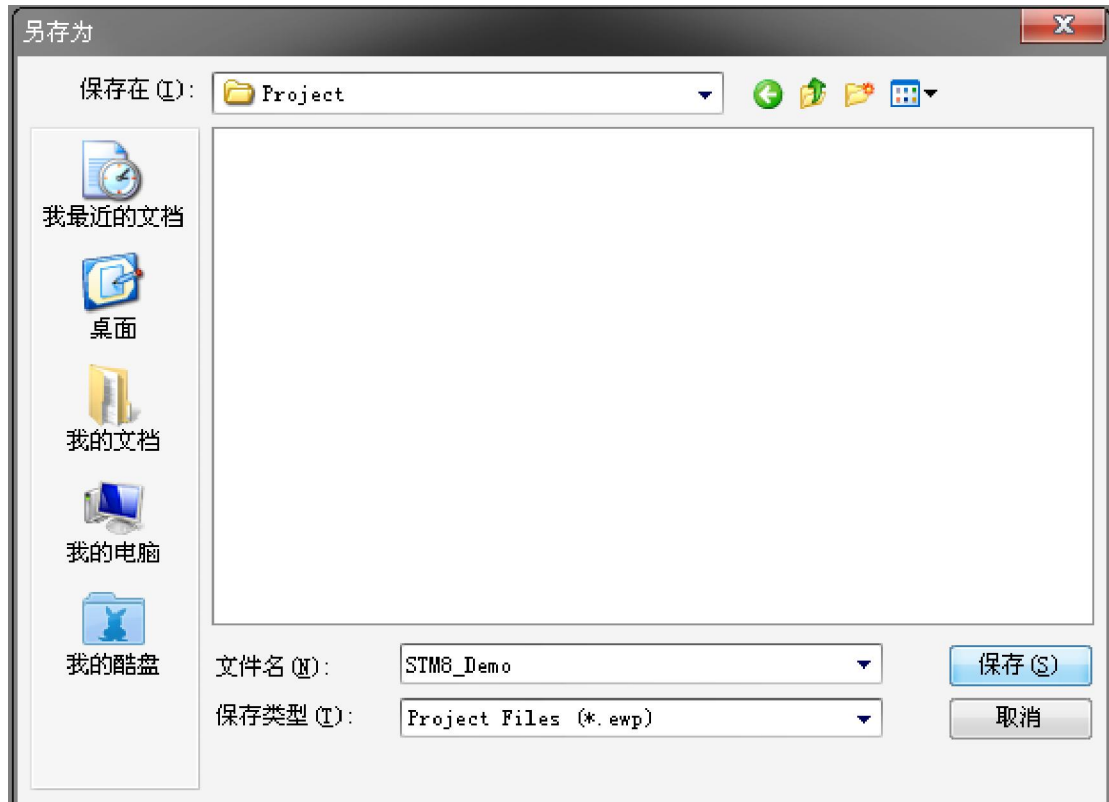
- 1、在任意一个地方建一个文件夹，命名为 STM8_Demo
- 2、在 STM8_Demo 的文件夹里面创建 2 个文件夹，并命名为 USER 和 Project。
- 3、打开 IAR 创建一个空工程保存到我们上面创建的 STM8_Demo 的 Project

中

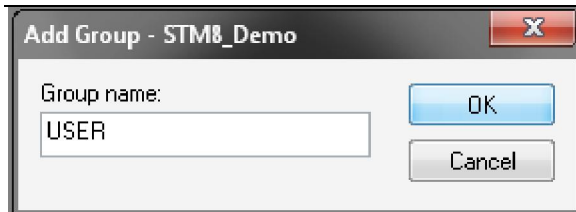




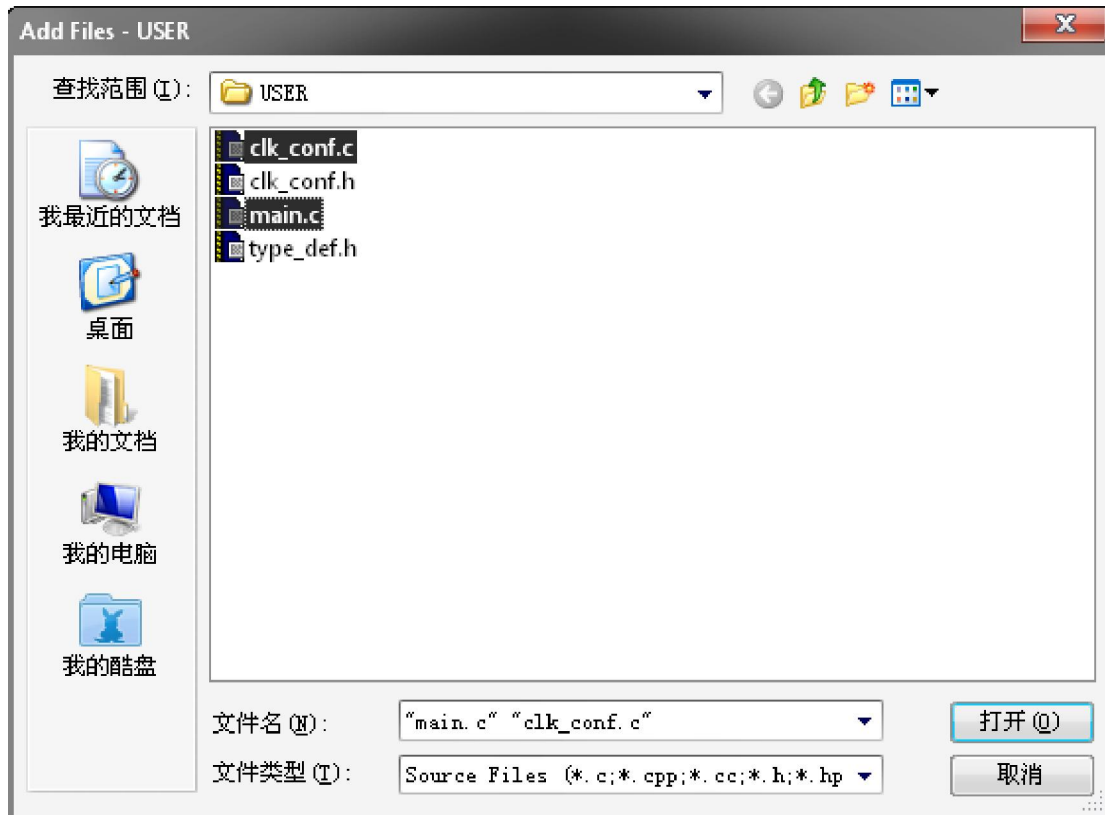
4.对工程文件进行命名，这里命名为 STM8_Demo



5.在工程文件中创建一个 USER 组:



6. 添加别的工程中已有的时钟初始化文件和 main.c 文件(不添加也行,可以自己创建再编写)



至此, 一个“比较干净的”工程就创建完成了。

其中的 Clk_conf 为风驰实现的时钟初始化函数, 当然也可以自己实现。用户代码在 while 循环中实现即可。

同时所有的数据类型和需要添加的头文件 `#include "clk_conf.h"`, 这个头文件是风驰封装的

```

/***** (C) COPYRIGHT 风驰iCreate嵌入式开发工作室 *****/
* 文件名   : type_def.h
* 描述     : 风驰iCreate STM8开发板专用头文件
* 实验平台 : iCreate STM8开发板
* 寄存器版本 : V1.0.0
* 作者     : ling_guansheng QQ: 779814207
* 博客     :
* 修改时间 : 2011-12-20
*****/
#ifndef __type_def_h
#define __type_def_h
/*主控芯片的头文件*/
#include "iostm8s207rb.h"
/*bool 类型头文件*/
#include "stdbool.h"
/*总中断头文件*/
#include "intrinsics.h"
/*常用数据类型定义*/
typedef unsigned char    uint8_t;
typedef unsigned short   uint16_t;
typedef unsigned long     uint32_t;
#define FlagStatus      bool
#define u8               uint8_t
#define u16               uint16_t
#define u32               uint32_t
#define EnableInterrupt  __enable_interrupt()
#endif

int main(void)
{
    /* Infinite loop */

    /*设置内部高速时钟16M为主时钟*/
    Clk_conf();
    LED_conf();
    Set_Led_Off();
    Buttom_conf();
    while(1)
    {
        /* 添加你的代码 */
        /******标准的51编程风格******/
        if(!Key_Scan()) PD0_out=~PD0_out;
        /******标准的51编程风格******/
    }
}

```

规范的编程风格，一目了然

标准的 51 编程风格