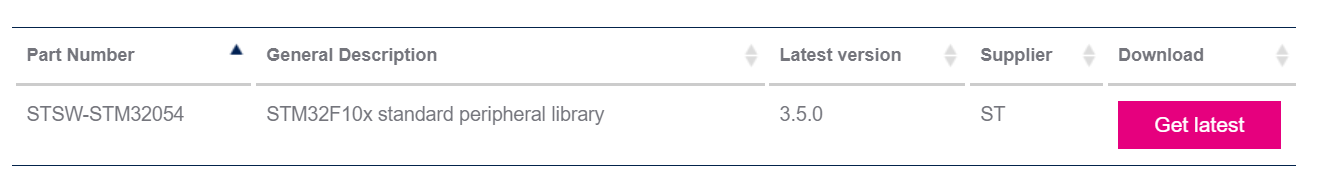
# Platform

Proteus\_8.9\_SP2 (stm32f103c6)

Keil MDK5

# 1. download STM32F103C6 library (version 3.5.0)

https://www.st.com/content/st\_com/en/products/embedded-software/mcu-mpu-embedded-software/stm32-embedded-software/stm32-standard-peripheral-libraries/stsw-stm32054.html



# 2. copy file to project

new BOOT, FreeRTOS, Interrupt, Library, Src directory in project directory. copy file

1. copy to boot directory

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\CoreSupport\core\_cm3.c

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\CoreSupport\core\_cm3.h

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\startup\arm\startup\_stm32f10x\_ld.s

2.copy to library directory

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\STM32F10x\_StdPeriph\_Driver\

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\stm32f10x.h

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\system\_stm32f10x.c

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\system\_stm32f10x.h

3.copy to interrupt diectory

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Project\STM32F10x\_StdPeriph\_Examples\GPIO\IOToggle\stm32f10x\_it.c

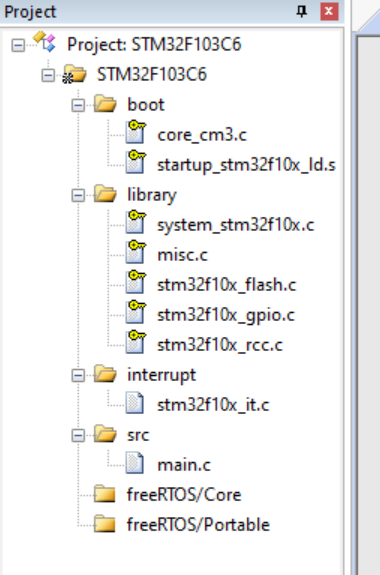
STM32F10x\_StdPeriph\_Lib\_V3.5.0\Project\STM32F10x\_StdPeriph\_Examples\GPIO\IOToggle\stm32f10x\_it.h

4.copy to src directory

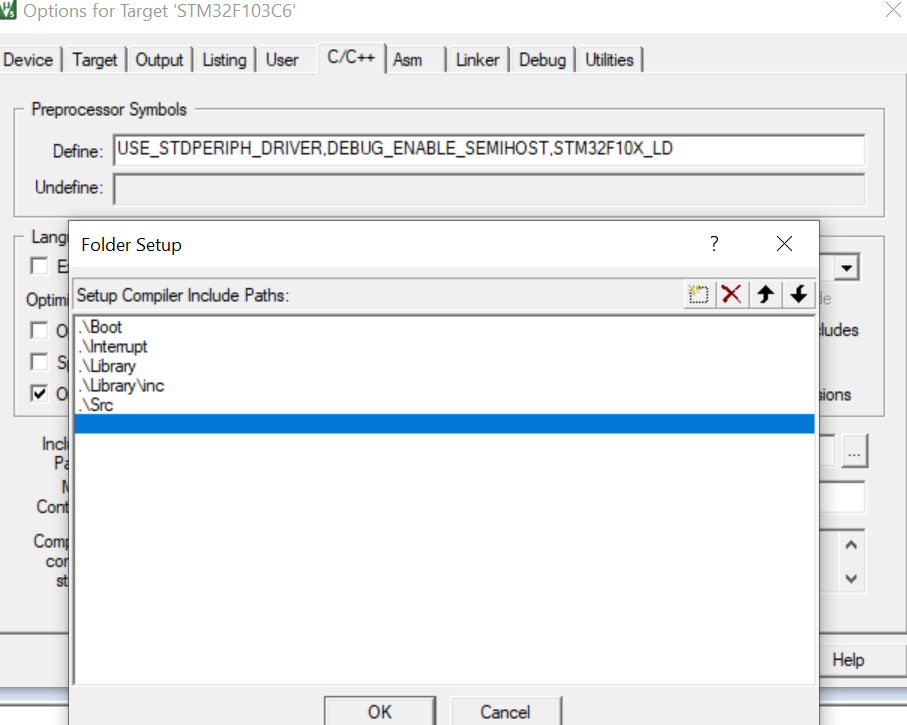
STM32F10x\_StdPeriph\_Lib\_V3.5.0\Project\STM32F10x\_StdPeriph\_Examples\GPIO\IOToggle\main.c

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Project\STM32F10x\_StdPeriph\_Examples\GPIO\IOToggle\stm32f10x\_conf.h

add file to project

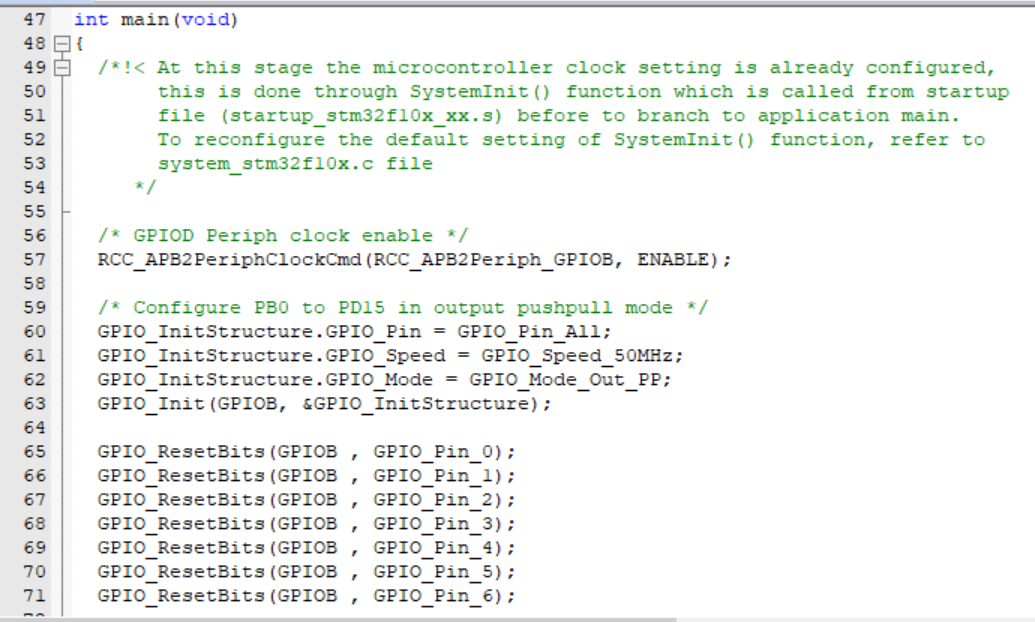


define and config

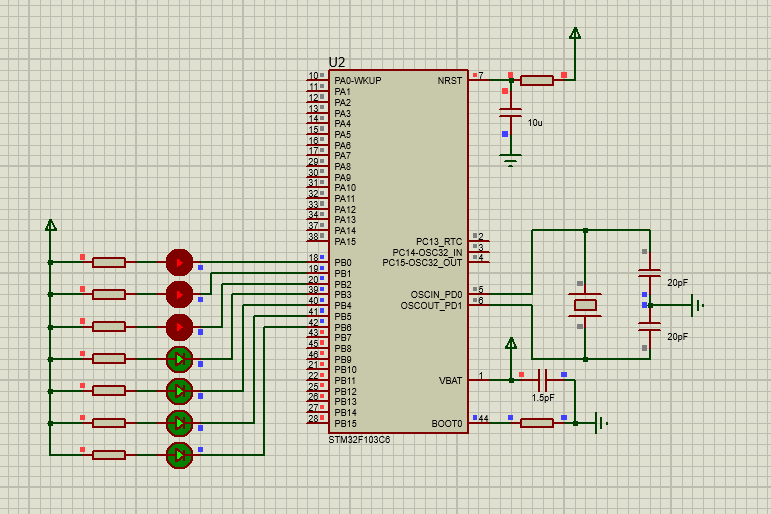


# 3. modify and test

修改Main文件



测试结果



# 4. porting UCOS

1. download UCOS

https://github.com/weston-embedded/uC-OS3

https://github.com/weston-embedded/uC-CPU

https://github.com/weston-embedded/uC-LIB

2. porting uc-lib

uC-LIB-develop\lib\_ascii.c

uC-LIB-develop\lib\_ascii.h

uC-LIB-develop\lib\_def.h

uC-LIB-develop\lib\_math.c

uC-LIB-develop\lib\_math.h

uC-LIB-develop\lib\_mem.c

uC-LIB-develop\lib\_mem.h

uC-LIB-develop\lib\_str.c

uC-LIB-develop\lib\_str.h

uC-LIB-develop\Cfg\Template\lib\_cfg.h

3. porting ucos-cpu

uC-CPU-develop\cpu\_cache.h

uC-CPU-develop\cpu\_core.c

uC-CPU-develop\cpu\_core.h

uC-CPU-develop\cpu\_def.h

//Cortex-M0 ARM-v6的架构

//Cortex-M3、 Cortex-M4和 Cortex-M7等都是采用的ARM-v7的架构

uC-CPU-develop\ARM-Cortex-M\ARMv7-M\ARM\cpu\_a.asm

uC-CPU-develop\ARM-Cortex-M\ARMv7-M\ARM\cpu.h

uC-CPU-develop\ARM-Cortex-M\ARMv7-M\cpu\_c.c

uC-CPU-develop\Cfg\Template\cpu\_cfg.h

uC-CPU-develop\BSP\Template\bsp\_cpu.c

4. porting ucos-source

uC-OS3-develop\Source\os.h

uC-OS3-develop\Source\os\_cfg\_app.c

uC-OS3-develop\Source\os\_core.c

uC-OS3-develop\Source\os\_dbg.c

uC-OS3-develop\Source\os\_flag.c

uC-OS3-develop\Source\os\_mem.c

uC-OS3-develop\Source\os\_msg.c

uC-OS3-develop\Source\os\_mutex.c

uC-OS3-develop\Source\os\_prio.c

uC-OS3-develop\Source\os\_q.c

uC-OS3-develop\Source\os\_sem.c

uC-OS3-develop\Source\os\_stat.c

uC-OS3-develop\Source\os\_task.c

uC-OS3-develop\Source\os\_tick.c

uC-OS3-develop\Source\os\_time.c

uC-OS3-develop\Source\os\_tmr.c

uC-OS3-develop\Source\os\_trace.h

uC-OS3-develop\Source\os\_type.h

uC-OS3-develop\Source\os\_var.c

5. porting ucos-portable

uC-OS3-develop\Cfg\Template\os\_app\_hooks.c

uC-OS3-develop\Cfg\Template\os\_app\_hooks.h

uC-OS3-develop\Cfg\Template\os\_cfg.h

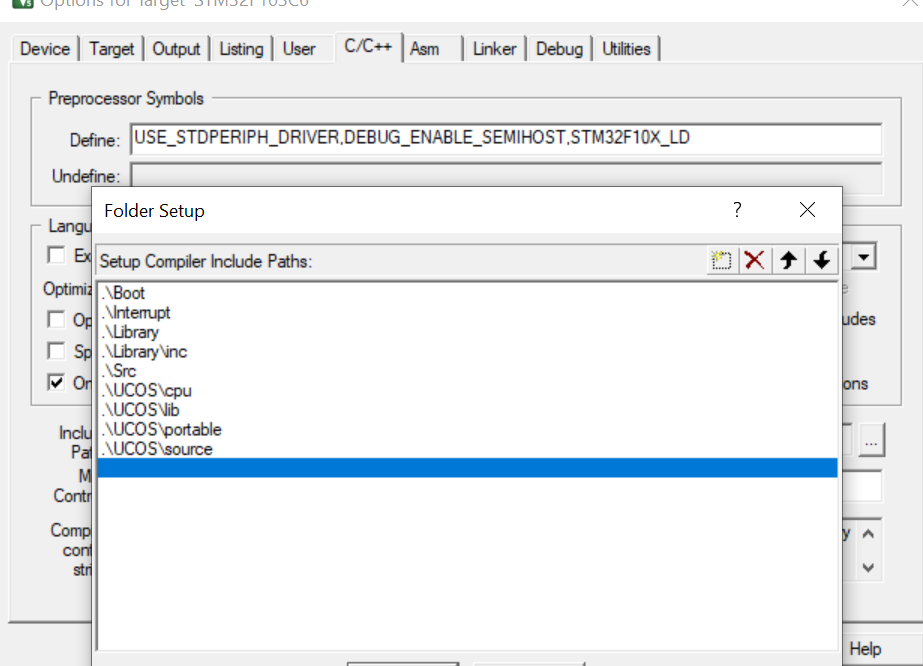
uC-OS3-develop\Cfg\Template\os\_cfg\_app.h

uC-OS3-develop\Ports\ARM-Cortex-M\ARMv7-M\os\_cpu\_c.c

uC-OS3-develop\Ports\ARM-Cortex-M\ARMv7-M\ARM\os\_cpu\_a.asm

uC-OS3-develop\Ports\ARM-Cortex-M\ARMv7-M\ARM\os\_cpu.h

uC-OS3-develop\Template\bsp\_os\_dt.c



修改os\_cpu.c

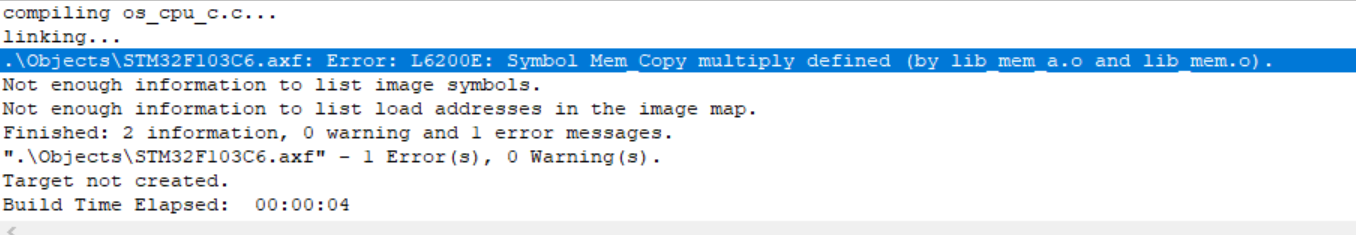
#include "os.h"

修改 cpu\_cfg.h

#if 1

#define CPU\_CFG\_NVIC\_PRIO\_BITS 4u

#endif



删除lib\_mem\_a.asm

修改启动文件

将PendSV\_Handler 和 SysTick\_Handler 替换成 OS\_CPU\_PendSVHandler 和 OS\_CPU\_SysTickHandler

;DCD PendSV\_Handler ; PendSV Handler

;DCD SysTick\_Handler ; SysTick Handler

DCD OS\_CPU\_PendSVHandler

DCD OS\_CPU\_SysTickHandler

;PendSV\_Handler PROC

; EXPORT PendSV\_Handler [WEAK]

OS\_CPU\_PendSVHandler PROC

EXPORT OS\_CPU\_PendSVHandler [WEAK]

B .

ENDP

;SysTick\_Handler PROC

; EXPORT SysTick\_Handler [WEAK]

OS\_CPU\_SysTickHandler PROC

EXPORT OS\_CPU\_SysTickHandler [WEAK]

B .

ENDP

UCOS不能同时执行两个任务，

调用OSTimeDlyHMSM(0,0,0,500,OS\_OPT\_TIME\_HMSM\_STRICT,&err); LED不闪烁

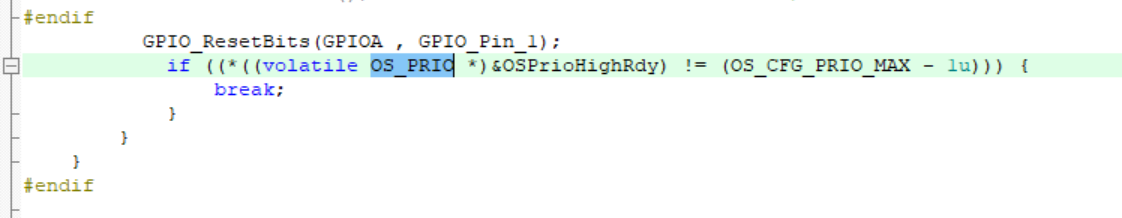
进入OSTimeDlyHMSM， 发现LED在OSSched() 里不闪烁

进入OSSched(), 发现经过 #if (OS\_CFG\_TASK\_IDLE\_EN > 0u) CPU\_INT\_EN()处，LED不亮



失能OS\_CFG\_TASK\_IDLE\_EN后

经过这里



最后发现原因：在OS\_IdleTask 写LED 闪烁代码，发现LED闪烁，说明了任务切换的时候，进入了空闲任务

OSTimeDlyHMSM 调用为什么会进入空闲任务？

经查询资料发现：疑似没有初始化SystemTick，在AppTaskStart函数中加上如下内容，再次测试结果

CPU\_INT32U cpu\_clk\_freq;

CPU\_INT32U cnts;

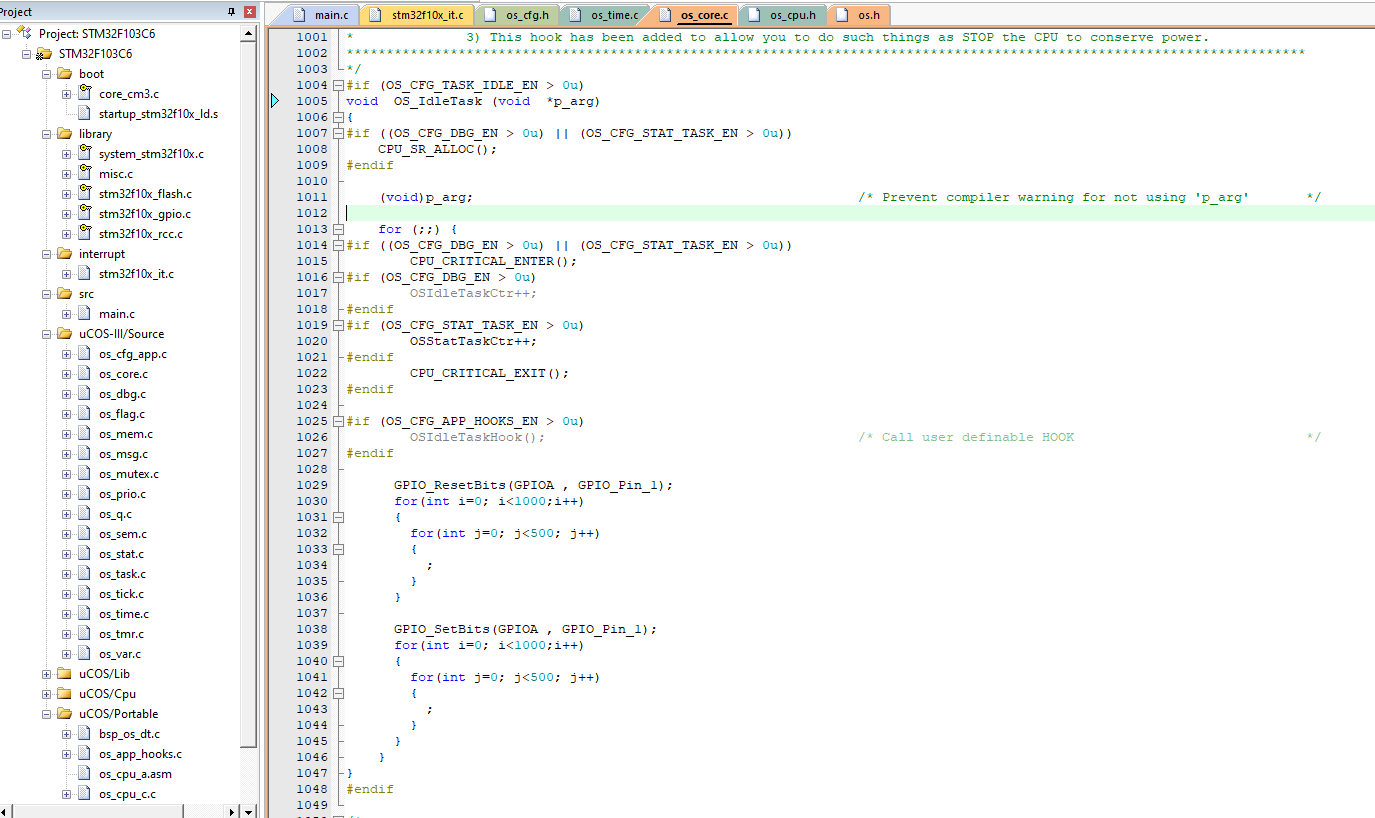
cpu\_clk\_freq = BSP\_CPU\_ClkFreq(); //获取 CPU 内核时钟频率（SysTick 工作时钟）

cnts = cpu\_clk\_freq / (CPU\_INT32U)OSCfg\_TickRate\_Hz; //根据用户设定的时钟节拍频率计算 SysTick 定时器的计数值

OS\_CPU\_SysTickInit(cnts);

原文地址

[(14条消息) uC/OS-III移植后发现程序停在空闲任务出不来\_Dlu.的博客-CSDN博客](https://blog.csdn.net/m0_46563076/article/details/104867322)



OSSched()任务调度

开始是为task1->切换到task2->切换到IdleTask->systick中断执行中进行任务切换回task1、2

