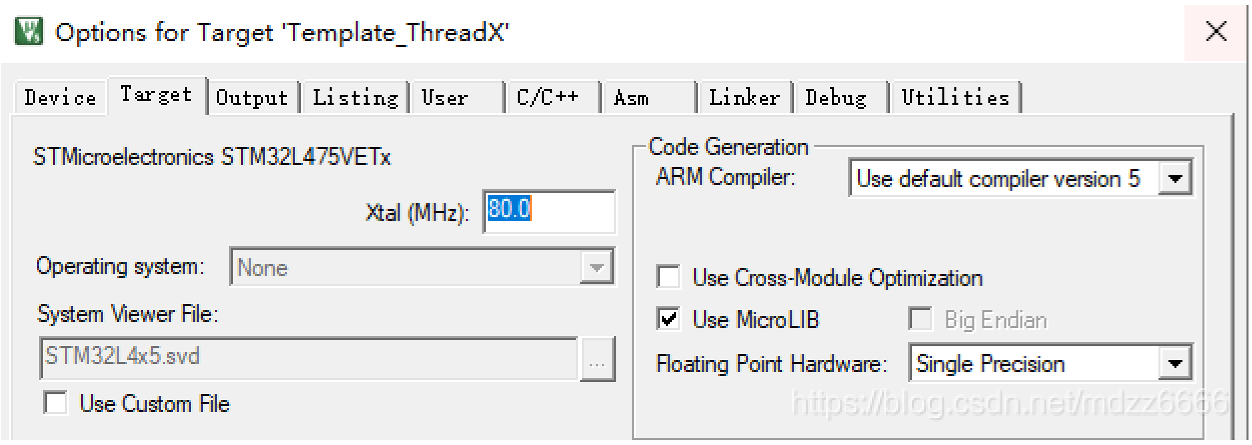
ThreadX最小移植及tx_initialize_low_level.s的简要分析

blog.csdn.net/mdzz6666/article/details/116752547

平台介绍

硬件平台：STM32L475VET6（M4内核的芯片应该都适用）；
ThreadX版本：6.1.3；
IDE：KEIL5 v5.31.0.0版本、STM32CubeMX；
ARM编译器：AC5


Options for Target 'Template_ThreadX'

Device | Target | Output | Listing | User | C/C++ | Asm | Linker | Debug | Utilities

STMicroelectronics STM32L475VETx

Xtal (MHz): 80.0

Operating system: None

System Viewer File: STM32L4x5.svd

☐ Use Custom File

Code Generation

ARM Compiler: Use default compiler version 5

☐ Use Cross-Module Optimization

☑ Use MicroLIB ☐ Big Endian

Floating Point Hardware: Single Precision

https://blog.csdn.net/mdzz6666

移植

准备一个简单的裸机程序

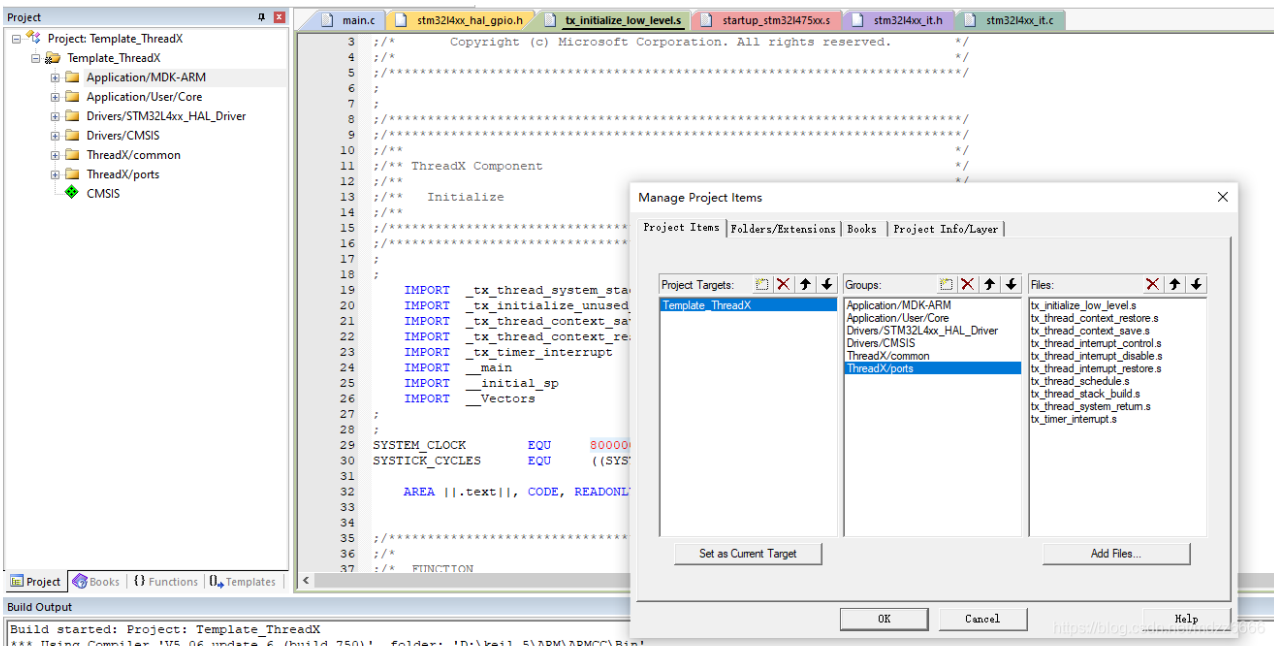
使用STM32CubeMX生成一个MDK-ARM平台的LED工程。

复制ThreadX源码到LED工程文件夹

将ThreadX源码下的common文件夹，ports>cortex_m4>ac5文件夹拷贝到LED工程文件夹下。
注：将ports>cortex_m4>ac5>example_build文件夹下的tx_initialize_low_level.s文件拷贝到ports>cortex_m4>ac5>src文件夹下，方便后期在工程中添加文件。详细步骤可参考安富莱_**STM32-V6**开发板**ThreadX**内核教程（**V0.4**）**.pdf**第4.4.1~4.4.3章节。

在LED工程中添加ThreadX文件

在LED工程中添加common和port文件、设置好头文件路径。



```
;/*       Copyright (c) Microsoft Corporation. All rights reserved.          */
;/*                                                                          */
;/**************************************************************************/
;
;
;/**************************************************************************/
;/**************************************************************************/
;/**                                                                       */
;/** ThreadX Component                                                     */
;/**                                                                       */
;/**   Initialize                                                          */
;/**
;/*********************************
;/*********************************
;
;
    IMPORT  _tx_thread_system_sta
    IMPORT  _tx_initialize_unused
    IMPORT  _tx_thread_context_sa
    IMPORT  _tx_thread_context_re
    IMPORT  _tx_timer_interrupt
    IMPORT  __main
    IMPORT  __initial_sp
    IMPORT  __Vectors
;
;
SYSTEM_CLOCK        EQU     80000
SYSTICK_CYCLES      EQU     ((SYS

    AREA ||.text||, CODE, READONL


;*********************************
;/*
;/*  FUNCTION
```

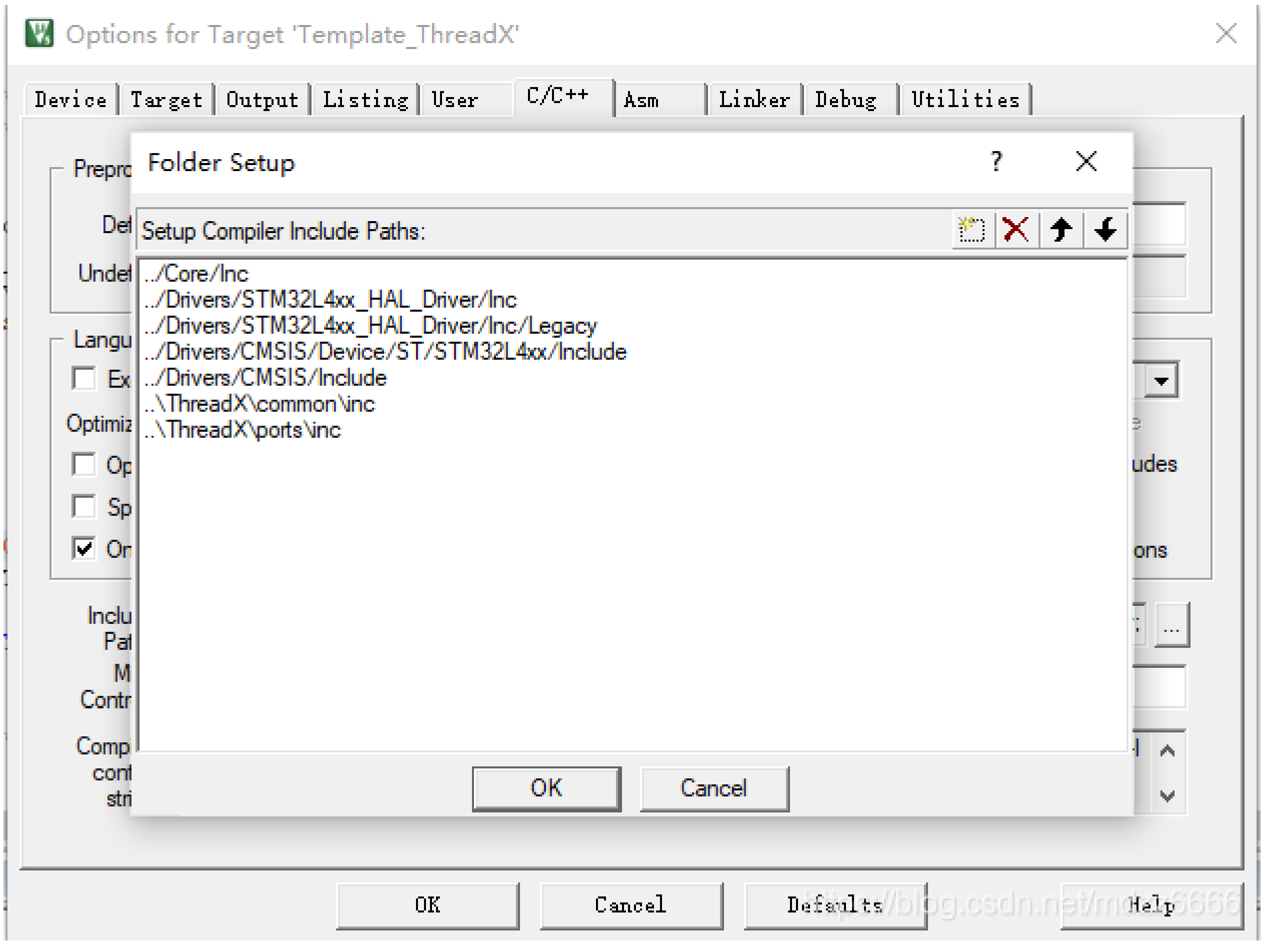
```
Build started: Project: Template_ThreadX
*** Using Compiler 'V5.06 update 6 (build 750)', folder: 'D:\keil_5\ARM\ARMCC\Bin'
```

Manage Project Items

Project Items | Folders/Extensions | Books | Project Info/Layer

Project Targets:	Groups:	Files:
Template_ThreadX	Application/MDK-ARM	tx_initialize_low_level.s
	Application/User/Core	tx_thread_context_restore.s
	Drivers/STM32L4xx_HAL_Driver	tx_thread_context_save.s
	Drivers/CMSIS	tx_thread_interrupt_control.s
	ThreadX/common	tx_thread_interrupt_disable.s
	ThreadX/ports	tx_thread_interrupt_restore.s
		tx_thread_schedule.s
		tx_thread_stack_build.s
		tx_thread_system_return.s
		tx_timer_interrupt.s

https://blog.csdn.net/mdzz6666



Setup Compiler Include Paths:

```
../Core/Inc
../Drivers/STM32L4xx_HAL_Driver/Inc
../Drivers/STM32L4xx_HAL_Driver/Inc/Legacy
../Drivers/CMSIS/Device/ST/STM32L4xx/Include
../Drivers/CMSIS/Include
..\ThreadX\common\inc
..\ThreadX\ports\inc
```

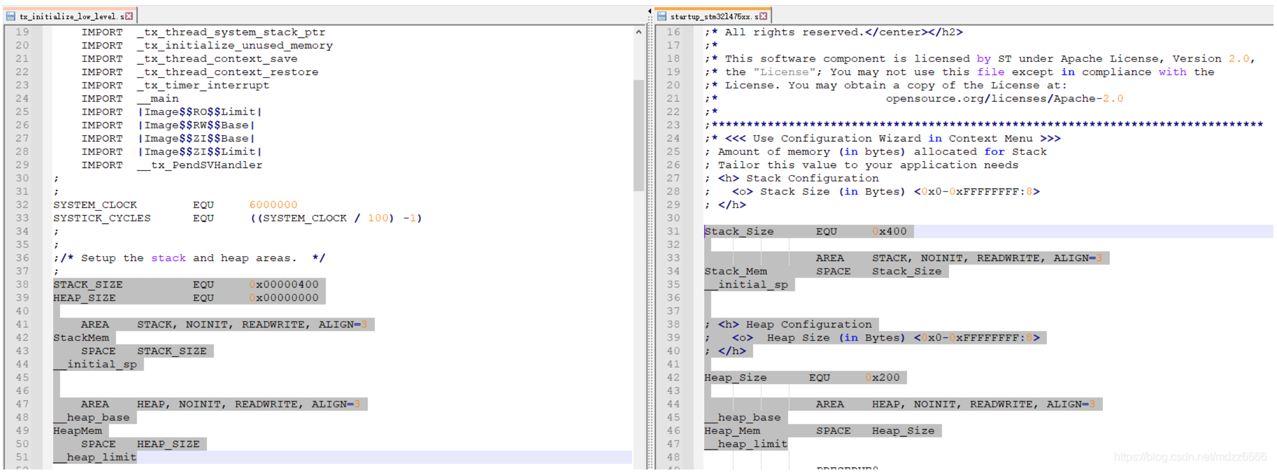
https://blog.csdn.net/mdzz6666

修改tx_initialize_low_level.s文件

修改思路：tx_initialize_low_level.s是ThreadX提供启动文件，其满足ThreadX需求却不满足mcu需求。而LED工程本身自带启动文件startup_stm32l475xx.s又不满足ThreadX的需求。所以我们需要将这两个文件结合起来使用，针对两个文件交集的部分，我们以LED工程自带的启动文件为准。

tx_initialize_low_level.s与startup_stm32l475xx.s的简单分析

①两个启动文件都有对于堆栈指针的描述，我们删除左边，以右边为准


tx_initialize_low_level.s

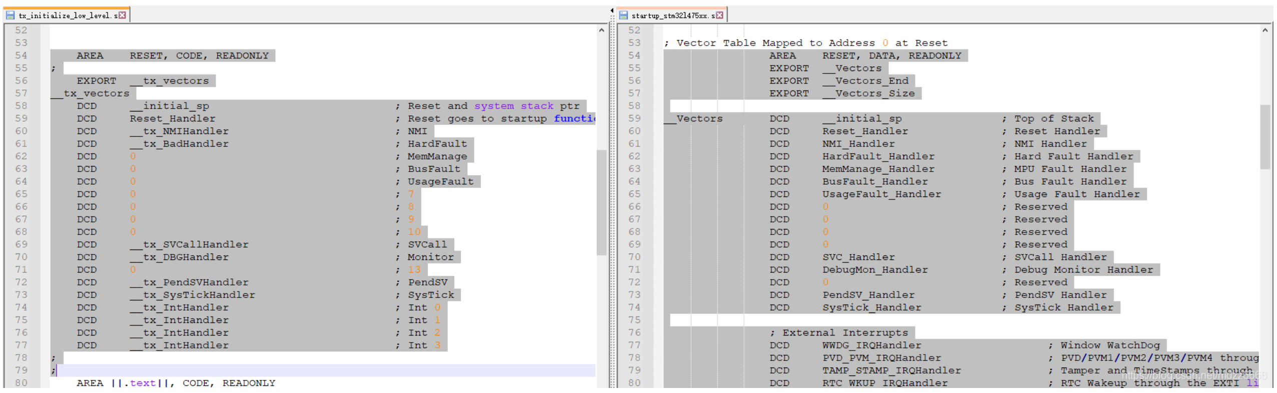
```
        IMPORT   _tx_thread_system_stack_ptr
        IMPORT   _tx_initialize_unused_memory
        IMPORT   _tx_thread_context_save
        IMPORT   _tx_thread_context_restore
        IMPORT   _tx_timer_interrupt
        IMPORT   __main
        IMPORT   |Image$$RO$$Limit|
        IMPORT   |Image$$RW$$Base|
        IMPORT   |Image$$ZI$$Base|
        IMPORT   |Image$$ZI$$Limit|
        IMPORT   __tx_PendSVHandler
;
;
SYSTEM_CLOCK         EQU     6000000
SYSTICK_CYCLES       EQU     ((SYSTEM_CLOCK / 100) -1)
;
;
;/* Setup the stack and heap areas.  */
;
STACK_SIZE           EQU     0x00000400
HEAP_SIZE            EQU     0x00000000

        AREA    STACK, NOINIT, READWRITE, ALIGN=3
StackMem
        SPACE   STACK_SIZE
__initial_sp


        AREA    HEAP, NOINIT, READWRITE, ALIGN=3
__heap_base
HeapMem
        SPACE   HEAP_SIZE
__heap_limit
```

startup_stm32l475xx.s

```
;* All rights reserved.</center></h2>
;*
;* This software component is licensed by ST under Apache License, Version 2.0,
;* the "License"; You may not use this file except in compliance with the
;* License. You may obtain a copy of the License at:
;*                        opensource.org/licenses/Apache-2.0
;*
;*******************************************************************************
;* <<< Use Configuration Wizard in Context Menu >>>
; Amount of memory (in bytes) allocated for Stack
; Tailor this value to your application needs
; <h> Stack Configuration
;   <o> Stack Size (in Bytes) <0x0-0xFFFFFFFF:8>
; </h>

Stack_Size      EQU     0x400

                AREA    STACK, NOINIT, READWRITE, ALIGN=3
Stack_Mem       SPACE   Stack_Size
__initial_sp


; <h> Heap Configuration
;   <o>  Heap Size (in Bytes) <0x0-0xFFFFFFFF:8>
; </h>

Heap_Size      EQU     0x200

                AREA    HEAP, NOINIT, READWRITE, ALIGN=3
__heap_base
Heap_Mem        SPACE   Heap_Size
__heap_limit
```

https://blog.csdn.net/mdzz6666

②两个启动文件都有中断向量表，但tx_initialize_low_level.s只描述了部分中断向量，不够完整，所以我们删除左边，以右边为准

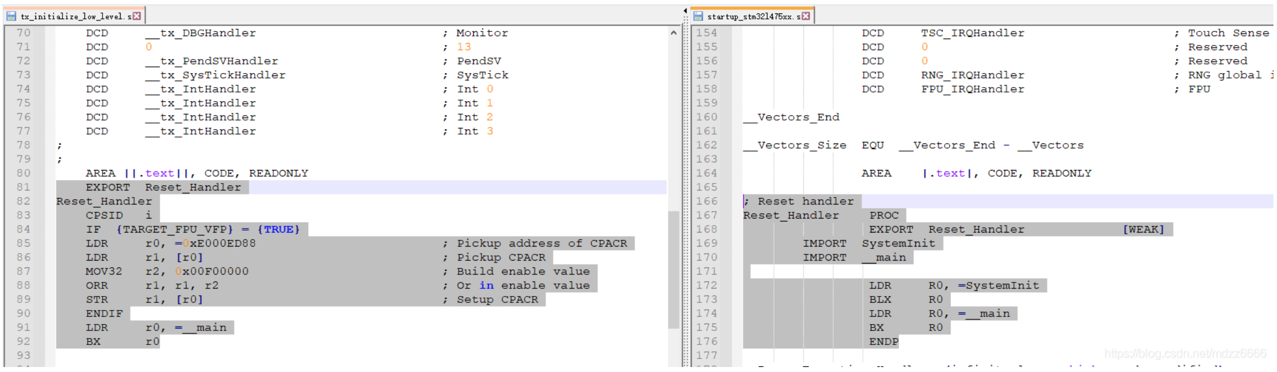

tx_initialize_low_level.s

```

        AREA    RESET, CODE, READONLY
;
        EXPORT  __tx_vectors
__tx_vectors
        DCD     __initial_sp                            ; Reset and system stack ptr
        DCD     Reset_Handler                           ; Reset goes to startup functi
        DCD     __tx_NMIHandler                         ; NMI
        DCD     __tx_BadHandler                         ; HardFault
        DCD     0                                       ; MemManage
        DCD     0                                       ; BusFault
        DCD     0                                       ; UsageFault
        DCD     0                                       ; 7
        DCD     0                                       ; 8
        DCD     0                                       ; 9
        DCD     0                                       ; 10
        DCD     __tx_SVCallHandler                      ; SVCall
        DCD     __tx_DBGHandler                         ; Monitor
        DCD     0                                       ; 13
        DCD     __tx_PendSVHandler                      ; PendSV
        DCD     __tx_SysTickHandler                     ; SysTick
        DCD     __tx_IntHandler                         ; Int 0
        DCD     __tx_IntHandler                         ; Int 1
        DCD     __tx_IntHandler                         ; Int 2
        DCD     __tx_IntHandler                         ; Int 3
;
;
        AREA ||.text||, CODE, READONLY
```

startup_stm32l475xx.s

```

; Vector Table Mapped to Address 0 at Reset
                AREA    RESET, DATA, READONLY
                EXPORT  __Vectors
                EXPORT  __Vectors_End
                EXPORT  __Vectors_Size

__Vectors       DCD     __initial_sp               ; Top of Stack
                DCD     Reset_Handler              ; Reset Handler
                DCD     NMI_Handler                ; NMI Handler
                DCD     HardFault_Handler          ; Hard Fault Handler
                DCD     MemManage_Handler          ; MPU Fault Handler
                DCD     BusFault_Handler           ; Bus Fault Handler
                DCD     UsageFault_Handler         ; Usage Fault Handler
                DCD     0                          ; Reserved
                DCD     0                          ; Reserved
                DCD     0                          ; Reserved
                DCD     0                          ; Reserved
                DCD     SVC_Handler                ; SVCall Handler
                DCD     DebugMon_Handler           ; Debug Monitor Handler
                DCD     0                          ; Reserved
                DCD     PendSV_Handler             ; PendSV Handler
                DCD     SysTick_Handler            ; SysTick Handler

                ; External Interrupts
                DCD     WWDG_IRQHandler                   ; Window WatchDog
                DCD     PVD_PVM_IRQHandler                ; PVD/PVM1/PVM2/PVM3/PVM4 throug
                DCD     TAMP_STAMP_IRQHandler             ; Tamper and TimeStamps through
                DCD     RTC_WKUP_IRQHandler               ; RTC Wakeup through the EXTI li
```

③两个文件都有中断处理函数，我们删除左边，以右边为准


tx_initialize_low_level.s

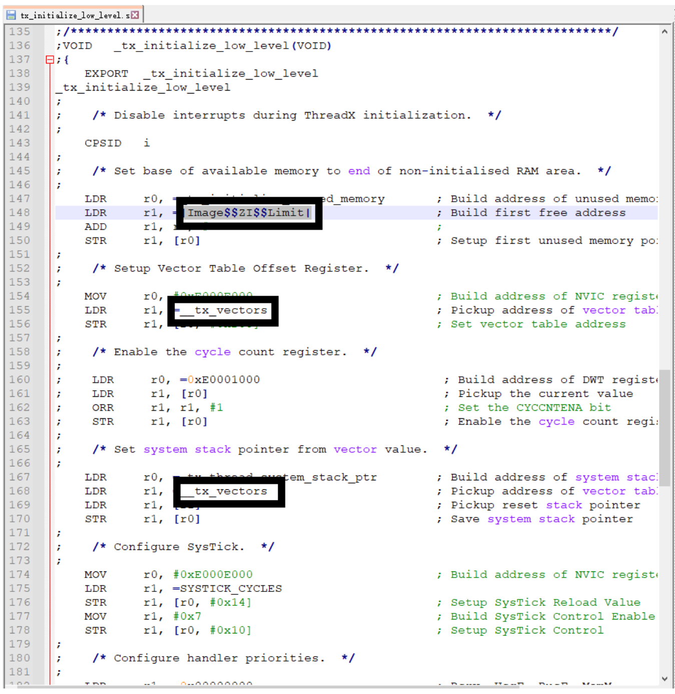
```
        DCD     __tx_DBGHandler                         ; Monitor
        DCD     0                                       ; 13
        DCD     __tx_PendSVHandler                      ; PendSV
        DCD     __tx_SysTickHandler                     ; SysTick
        DCD     __tx_IntHandler                         ; Int 0
        DCD     __tx_IntHandler                         ; Int 1
        DCD     __tx_IntHandler                         ; Int 2
        DCD     __tx_IntHandler                         ; Int 3
;
;
        AREA ||.text||, CODE, READONLY
        EXPORT  Reset_Handler
Reset_Handler
        CPSID   i
        IF  {TARGET_FPU_VFP} = {TRUE}
        LDR     r0, =0xE000ED88                         ; Pickup address of CPACR
        LDR     r1, [r0]                                ; Pickup CPACR
        MOV32   r2, 0x00F00000                          ; Build enable value
        ORR     r1, r1, r2                              ; Or in enable value
        STR     r1, [r0]                                ; Setup CPACR
        ENDIF
        LDR     r0, =__main
        BX      r0

```

startup_stm32l475xx.s

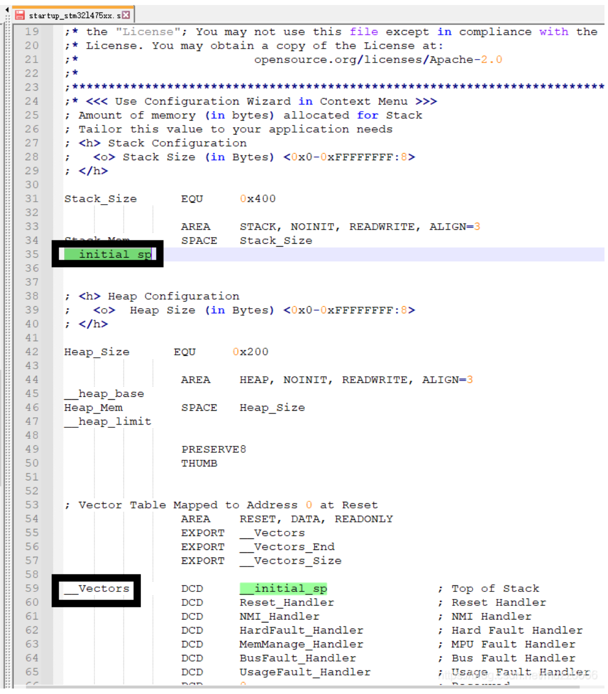
```
                DCD     TSC_IRQHandler                    ; Touch Sense
                DCD     0                                 ; Reserved
                DCD     0                                 ; Reserved
                DCD     RNG_IRQHandler                    ; RNG global i
                DCD     FPU_IRQHandler                    ; FPU

__Vectors_End

__Vectors_Size  EQU  __Vectors_End - __Vectors

                AREA    |.text|, CODE, READONLY

; Reset handler
Reset_Handler    PROC
                 EXPORT  Reset_Handler             [WEAK]
        IMPORT  SystemInit
        IMPORT  __main

                 LDR     R0, =SystemInit
                 BLX     R0
                 LDR     R0, =__main
                 BX      R0
                 ENDP

```

https://blog.csdn.net/mdzz6666

④修改tx_initialize_low_level.s文件中的VOID _tx_initialize_low_level(VOID)代码段。将 `|Image$$ZI$$Limit|` 和 `__tx_vectors` 替换为startup_stm32l475xx.s文件中的 `__initial_sp` 和 `__Vectors` 。

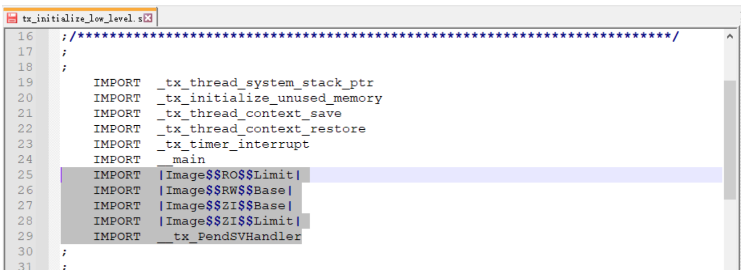

tx_initialize_low_level.s

```
;/**************************************************************************/
;VOID   _tx_initialize_low_level(VOID)
;{
    EXPORT  _tx_initialize_low_level
_tx_initialize_low_level
;
;    /* Disable interrupts during ThreadX initialization.  */
;
    CPSID   i
;
;    /* Set base of available memory to end of non-initialised RAM area.  */
;
    LDR     r0, =_tx_initialize_unused_memory       ; Build address of unused memo
    LDR     r1, =|Image$$ZI$$Limit|                 ; Build first free address
    ADD     r1, r1, #4                              ;
    STR     r1, [r0]                                ; Setup first unused memory po
;
;    /* Setup Vector Table Offset Register.  */
;
    MOV     r0, #0xE000E000                         ; Build address of NVIC regist
    LDR     r1, =__tx_vectors                       ; Pickup address of vector tab
    STR     r1, [r0, #0xD08]                        ; Set vector table address
;
;    /* Enable the cycle count register.  */
;
;    LDR     r0, =0xE0001000                        ; Build address of DWT regist
;    LDR     r1, [r0]                               ; Pickup the current value
;    ORR     r1, r1, #1                             ; Set the CYCCNTENA bit
;    STR     r1, [r0]                               ; Enable the cycle count regi
;
;    /* Set system stack pointer from vector value.  */
;
    LDR     r0, =_tx_thread_system_stack_ptr        ; Build address of system stac
    LDR     r1, =__tx_vectors                       ; Pickup address of vector tab
    LDR     r1, [r1]                                ; Pickup reset stack pointer
    STR     r1, [r0]                                ; Save system stack pointer
;
;    /* Configure SysTick.  */
;
    MOV     r0, #0xE000E000                         ; Build address of NVIC regist
    LDR     r1, =SYSTICK_CYCLES
    STR     r1, [r0, #0x14]                         ; Setup SysTick Reload Value
    MOV     r1, #0x7                                ; Build SysTick Control Enable
    STR     r1, [r0, #0x10]                         ; Setup SysTick Control
;
;    /* Configure handler priorities.  */
;
```

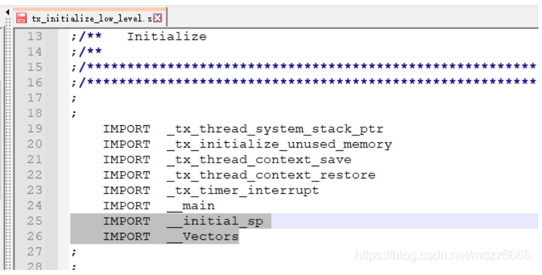

startup_stm32l475xx.s

```
;* the "License"; You may not use this file except in compliance with the
;* License. You may obtain a copy of the License at:
;*                        opensource.org/licenses/Apache-2.0
;*
;*******************************************************************************
;* <<< Use Configuration Wizard in Context Menu >>>
; Amount of memory (in bytes) allocated for Stack
; Tailor this value to your application needs
; <h> Stack Configuration
;   <o> Stack Size (in Bytes) <0x0-0xFFFFFFFF:8>
; </h>

Stack_Size      EQU     0x400

                AREA    STACK, NOINIT, READWRITE, ALIGN=3
Stack_Mem       SPACE   Stack_Size
__initial_sp


; <h> Heap Configuration
;   <o>  Heap Size (in Bytes) <0x0-0xFFFFFFFF:8>
; </h>

Heap_Size      EQU     0x200

                AREA    HEAP, NOINIT, READWRITE, ALIGN=3
__heap_base
Heap_Mem        SPACE   Heap_Size
__heap_limit

                PRESERVE8
                THUMB


; Vector Table Mapped to Address 0 at Reset
                AREA    RESET, DATA, READONLY
                EXPORT  __Vectors
                EXPORT  __Vectors_End
                EXPORT  __Vectors_Size

__Vectors       DCD     __initial_sp               ; Top of Stack
                DCD     Reset_Handler              ; Reset Handler
                DCD     NMI_Handler                ; NMI Handler
                DCD     HardFault_Handler          ; Hard Fault Handler
                DCD     MemManage_Handler          ; MPU Fault Handler
                DCD     BusFault_Handler           ; Bus Fault Handler
                DCD     UsageFault_Handler         ; Usage Fault Handler
```

因为 `__initial_sp` 和 `__Vectors` 声明在startup_stm32l475xx.s文件，所以应该在tx_initialize_low_level.s文件中引入上述两个变量。 `IMPORT |Image$$RO$$Limit| IMPORT |Image$$RW$$Base| IMPORT |Image$$ZI$$Base| IMPORT |Image$$ZI$$Limit| IMPORT __tx_PendSVHandler` 未使用，可以删除，修改好的代码如右图所示

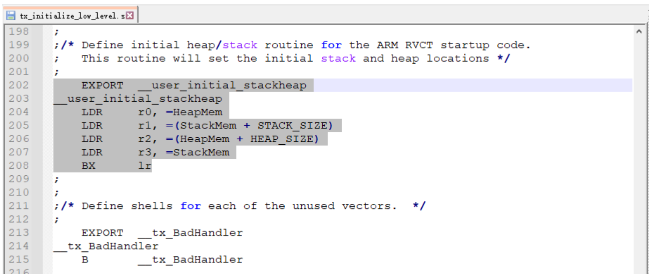

tx_initialize_low_level.s

```
;/**************************************************************************/
;
;
    IMPORT  _tx_thread_system_stack_ptr
    IMPORT  _tx_initialize_unused_memory
    IMPORT  _tx_thread_context_save
    IMPORT  _tx_thread_context_restore
    IMPORT  _tx_timer_interrupt
    IMPORT  __main
    IMPORT  |Image$$RO$$Limit|
    IMPORT  |Image$$RW$$Base|
    IMPORT  |Image$$ZI$$Base|
    IMPORT  |Image$$ZI$$Limit|
    IMPORT  __tx_PendSVHandler
;
;
```

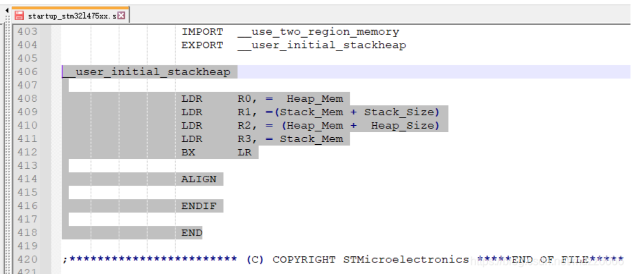

tx_initialize_low_level.s

```
;/**   Initialize
;/**
;/*****************************************************************
;/*****************************************************************
;
;
    IMPORT  _tx_thread_system_stack_ptr
    IMPORT  _tx_initialize_unused_memory
    IMPORT  _tx_thread_context_save
    IMPORT  _tx_thread_context_restore
    IMPORT  _tx_timer_interrupt
    IMPORT  __main
    IMPORT  __initial_sp
    IMPORT  __Vectors
;
;
```

⑤两个文件都有初始化堆，我们删除左边，以右边为准

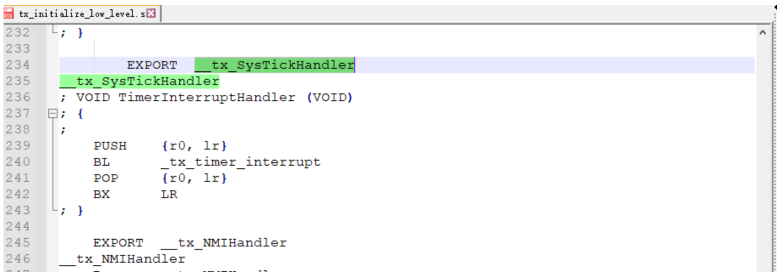

tx_initialize_low_level.s

```
;
;/* Define initial heap/stack routine for the ARM RVCT startup code.
;   This routine will set the initial stack and heap locations */
;
    EXPORT  __user_initial_stackheap
__user_initial_stackheap
    LDR     r0, =HeapMem
    LDR     r1, =(StackMem + STACK_SIZE)
    LDR     r2, =(HeapMem + HEAP_SIZE)
    LDR     r3, =StackMem
    BX      lr
;
;
;/* Define shells for each of the unused vectors.  */
;
    EXPORT  __tx_BadHandler
__tx_BadHandler
    B       __tx_BadHandler
```

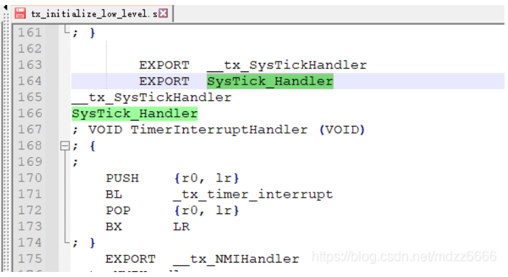

startup_stm32l475xx.s

```
                IMPORT  __use_two_region_memory
                EXPORT  __user_initial_stackheap

__user_initial_stackheap

                LDR     R0, =  Heap_Mem
                LDR     R1, =(Stack_Mem + Stack_Size)
                LDR     R2, = (Heap_Mem +  Heap_Size)
                LDR     R3, = Stack_Mem
                BX      LR

                ALIGN

                ENDIF

                END

;************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE*****
```

⑥在tx_initialize_low_level.s文件定义SysTick_Handler代码段，只需添加两行代码，修改完成后如右图所示

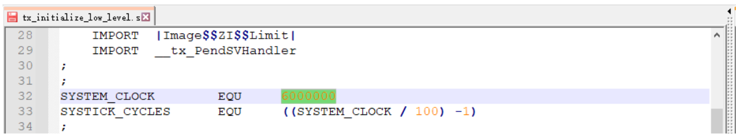

tx_initialize_low_level.s

```
; }

        EXPORT  __tx_SysTickHandler
__tx_SysTickHandler
; VOID TimerInterruptHandler (VOID)
; {
;
    PUSH    {r0, lr}
    BL      _tx_timer_interrupt
    POP     {r0, lr}
    BX      LR
; }

    EXPORT  __tx_NMIHandler
__tx_NMIHandler
```


tx_initialize_low_level.s

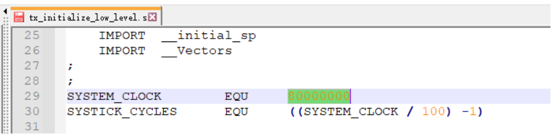
```
; }

        EXPORT  __tx_SysTickHandler
        EXPORT  SysTick_Handler
__tx_SysTickHandler
SysTick_Handler
; VOID TimerInterruptHandler (VOID)
; {
;
    PUSH    {r0, lr}
    BL      _tx_timer_interrupt
    POP     {r0, lr}
    BX      LR
; }
    EXPORT  __tx_NMIHandler
```

⑦修改tx_initialize_low_level.s文件中系统时钟，其与STM32CubeMX中设置的时钟一致，否则会出现奇怪的运行结果。这里我设置的是80M，如右图所示



tx_initialize_low_level.s

```
        IMPORT   |Image$$ZI$$Limit|
        IMPORT   __tx_PendSVHandler
;
;
SYSTEM_CLOCK         EQU      6000000
SYSTICK_CYCLES       EQU      ((SYSTEM_CLOCK / 100) -1)
;
```



tx_initialize_low_level.s

```
        IMPORT   __initial_sp
        IMPORT   __Vectors
;
;
SYSTEM_CLOCK         EQU      80000000
SYSTICK_CYCLES       EQU      ((SYSTEM_CLOCK / 100) -1)
```

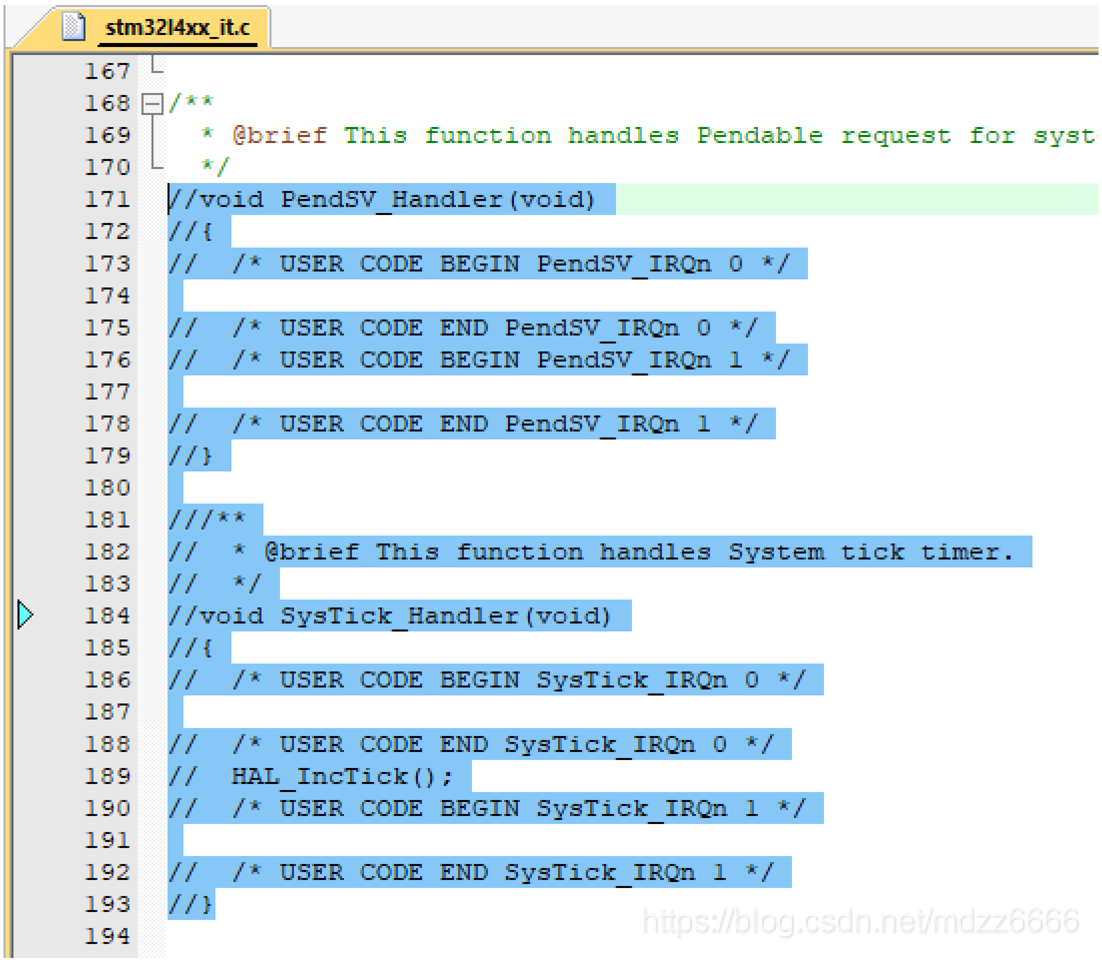
修改完成后的tx_initialize_low_level.s文件如下所示：

```
;/**************************************************************************/
;/*                                                                        */
;/*       Copyright (c) Microsoft Corporation. All rights reserved.        */
;/*                                                                        */
;/**************************************************************************/
;
;
;/**************************************************************************/
;/**************************************************************************/
;/**                                                                       */
;/** ThreadX Component                                                     */
;/**                                                                       */
;/**   Initialize                                                          */
;/**                                                                       */
;/**************************************************************************/
;/**************************************************************************/
;
;
    IMPORT  _tx_thread_system_stack_ptr
    IMPORT  _tx_initialize_unused_memory
    IMPORT  _tx_thread_context_save
    IMPORT  _tx_thread_context_restore
    IMPORT  _tx_timer_interrupt
    IMPORT  __main
    IMPORT  __initial_sp
    IMPORT  __Vectors
;
;
SYSTEM_CLOCK        EQU     80000000
SYSTICK_CYCLES      EQU     ((SYSTEM_CLOCK / 100) -1)

    AREA ||.text||, CODE, READONLY


;/**************************************************************************/
;/*                                                                        */
;/*  FUNCTION                                               RELEASE        */
;/*                                                                        */
;/*    _tx_initialize_low_level                          Cortex-M4/AC5     */
;/*                                                           6.1          */
;/*  AUTHOR                                                                */
;/*                                                                        */
;/*    William E. Lamie, Microsoft Corporation.                            */
;/*                                                                        */
;/*  DESCRIPTION                                                           */
;/*                                                                        */
;/*    This function is responsible for any low-level processor            */
;/*    initialization, including setting up interrupt vectors, setting     */
;/*    up a periodic timer interrupt source, saving the system stack       */
;/*    pointer for use in ISR processing later, and finding the first      */
;/*    available RAM memory address for tx_application_define.             */
;/*                                                                        */
;/*  INPUT                                                                 */
;/*                                                                        */
;/*    None                                                                */
;/*                                                                        */
;/*  OUTPUT                                                                */
;/*                                                                        */
;/*    None                                                                */
;/*                                                                        */
;/*  CALLS                                                                */
;/*                                                                       */
;/*    None                                                               */
;/*                                                                       */
;/*  CALLED BY                                                            */
;/*                                                                       */
;/*    _tx_initialize_kernel_enter           ThreadX entry function       */
;/*                                                                       */
;/*  RELEASE HISTORY                                                      */
;/*                                                                       */
;/*    DATE              NAME                      DESCRIPTION            */
;/*                                                                       */
;/*  09-30-2020     William E. Lamie         Initial Version 6.1          */
;/*                                                                       */
;/**************************************************************************/
;VOID   _tx_initialize_low_level(VOID)
;{
    EXPORT  _tx_initialize_low_level
_tx_initialize_low_level
;
;    /* Disable interrupts during ThreadX initialization.  */
;
    CPSID   i
;
;    /* Set base of available memory to end of non-initialised RAM area.  */
;
    LDR     r0, =_tx_initialize_unused_memory       ; Build address of unused memory
pointer
    LDR     r1, =__initial_sp                           ; Build first free address
    ADD     r1, r1, #4                              ;
    STR     r1, [r0]                                ; Setup first unused memory pointer
;
;    /* Setup Vector Table Offset Register.  */
;
    MOV     r0, #0xE000E000                         ; Build address of NVIC registers
    LDR     r1, =__Vectors                       ; Pickup address of vector table
    STR     r1, [r0, #0xD08]                        ; Set vector table address
;
;    /* Enable the cycle count register.  */
;
;    LDR     r0, =0xE0001000                        ; Build address of DWT register
;    LDR     r1, [r0]                               ; Pickup the current value
;    ORR     r1, r1, #1                             ; Set the CYCCNTENA bit
;    STR     r1, [r0]                               ; Enable the cycle count register
;
;    /* Set system stack pointer from vector value.  */
;
    LDR     r0, =_tx_thread_system_stack_ptr        ; Build address of system stack
pointer
    LDR     r1, =__Vectors                       ; Pickup address of vector table
    LDR     r1, [r1]                                ; Pickup reset stack pointer
    STR     r1, [r0]                                ; Save system stack pointer
;
;    /* Configure SysTick.  */
;
    MOV     r0, #0xE000E000                         ; Build address of NVIC registers
    LDR     r1, =SYSTICK_CYCLES
    STR     r1, [r0, #0x14]                         ; Setup SysTick Reload Value
    MOV     r1, #0x7                                ; Build SysTick Control Enable Value
    STR     r1, [r0, #0x10]                         ; Setup SysTick Control
```

```
;
;    /* Configure handler priorities.  */
;
    LDR     r1, =0x00000000                         ; Rsrv, UsgF, BusF, MemM
    STR     r1, [r0, #0xD18]                        ; Setup System Handlers 4-7 Priority
Registers

    LDR     r1, =0xFF000000                         ; SVCl, Rsrv, Rsrv, Rsrv
    STR     r1, [r0, #0xD1C]                        ; Setup System Handlers 8-11 Priority
Registers
                                                    ; Note: SVC must be lowest priority,
which is 0xFF

    LDR     r1, =0x40FF0000                         ; SysT, PnSV, Rsrv, DbgM
    STR     r1, [r0, #0xD20]                        ; Setup System Handlers 12-15 Priority
Registers
                                                    ; Note: PnSV must be lowest priority,
which is 0xFF
;
;    /* Return to caller.  */
;
    BX      lr
;}
;
;

;/* Define shells for each of the unused vectors.  */
;
    EXPORT  __tx_BadHandler
__tx_BadHandler
    B       __tx_BadHandler

    EXPORT  __tx_SVCallHandler
__tx_SVCallHandler
    B       __tx_SVCallHandler

    EXPORT  __tx_IntHandler
__tx_IntHandler
; VOID InterruptHandler (VOID)
; {
    PUSH    {r0, lr}

;    /* Do interrupt handler work here */
;    /* .... */

    POP     {r0, lr}
    BX      LR
; }

        EXPORT  __tx_SysTickHandler
                EXPORT  SysTick_Handler
__tx_SysTickHandler
SysTick_Handler
; VOID TimerInterruptHandler (VOID)
; {
;
    PUSH    {r0, lr}
    BL      _tx_timer_interrupt
    POP     {r0, lr}
    BX      LR
```

```
; }

    EXPORT   __tx_NMIHandler
__tx_NMIHandler
    B        __tx_NMIHandler

    EXPORT   __tx_DBGHandler
__tx_DBGHandler
    B        __tx_DBGHandler

    ALIGN
    LTORG
    END
```

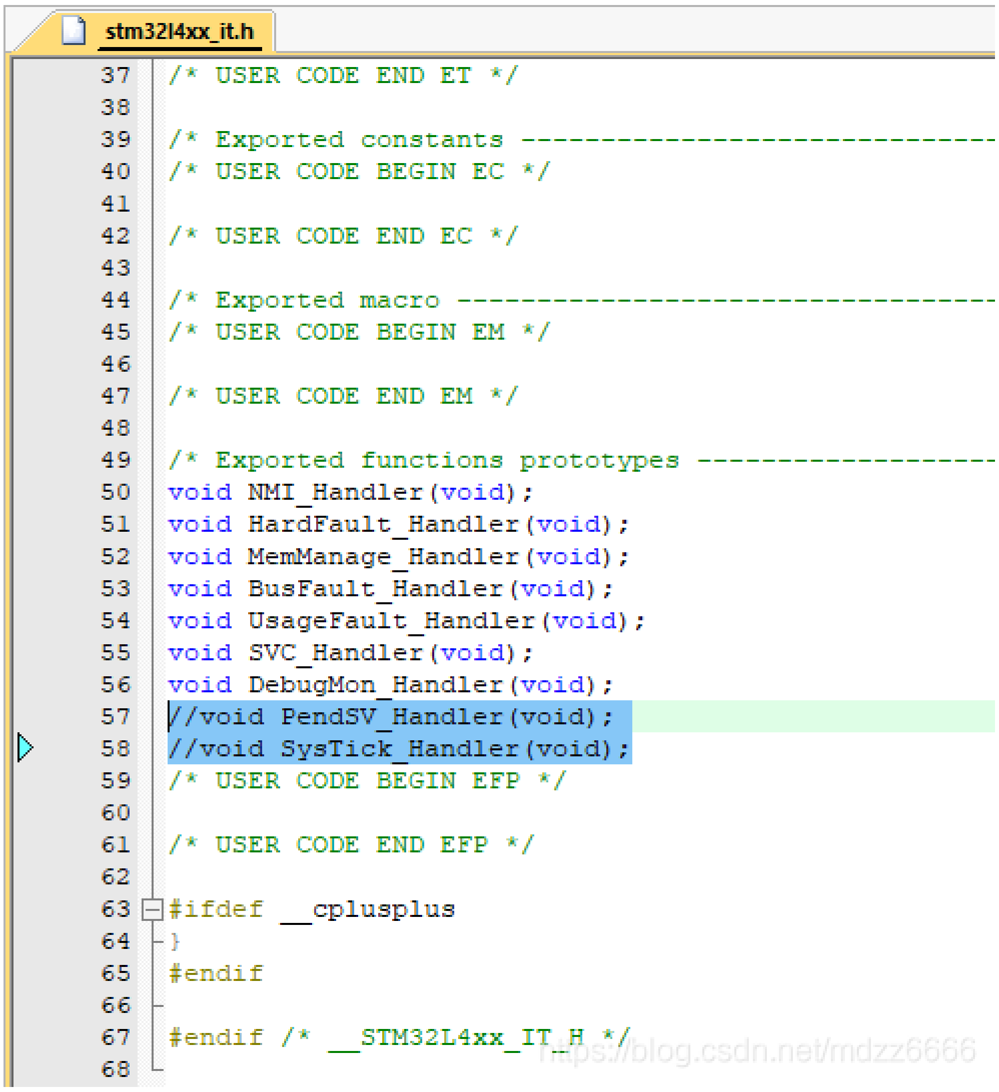
注释LED工程原本的 `void PendSV_Handler(void);` 和 `void SysTick_Handler(void);`



stm32l4xx_it.c

```
/**
  * @brief This function handles Pendable request for syst
  */
//void PendSV_Handler(void)
//{
//  /* USER CODE BEGIN PendSV_IRQn 0 */

//  /* USER CODE END PendSV_IRQn 0 */
//  /* USER CODE BEGIN PendSV_IRQn 1 */

//  /* USER CODE END PendSV_IRQn 1 */
//}

///**
//  * @brief This function handles System tick timer.
//  */
//void SysTick_Handler(void)
//{
//  /* USER CODE BEGIN SysTick_IRQn 0 */

//  /* USER CODE END SysTick_IRQn 0 */
//  HAL_IncTick();
//  /* USER CODE BEGIN SysTick_IRQn 1 */

//  /* USER CODE END SysTick_IRQn 1 */
//}
```

https://blog.csdn.net/mdzz6666



stm32l4xx_it.h

```
/* USER CODE END ET */

/* Exported constants --------------------------------
/* USER CODE BEGIN EC */

/* USER CODE END EC */

/* Exported macro ------------------------------------
/* USER CODE BEGIN EM */

/* USER CODE END EM */

/* Exported functions prototypes --------------------
void NMI_Handler(void);
void HardFault_Handler(void);
void MemManage_Handler(void);
void BusFault_Handler(void);
void UsageFault_Handler(void);
void SVC_Handler(void);
void DebugMon_Handler(void);
//void PendSV_Handler(void);
//void SysTick_Handler(void);
/* USER CODE BEGIN EFP */

/* USER CODE END EFP */

#ifdef __cplusplus
}
#endif

#endif /* __STM32L4xx_IT_H */
```

https://blog.csdn.net/mdzz6666

提供一个简单的测试任务

```
uint8_t my_buff[1024];
uint8_t you_buff[1024];

TX_THREAD my_thread,you_thread;
void my_thread_entry(ULONG thread_input)
{
        /* Enter into a forever loop. */
        while(1)
        {
                tx_thread_sleep(3);
        }
}

void you_thread_entry(ULONG thread_input)
{
        while(1)
        {
                tx_thread_sleep(3);
        }
}

void tx_application_define(void *first_unused_memory)
{
        /* Create my_thread! */
        tx_thread_create(&my_thread, "My Thread",
        my_thread_entry, 0x1234, my_buff, 1024,
        3, 3, TX_NO_TIME_SLICE, TX_AUTO_START);

        tx_thread_create(&you_thread, "You Thread",
        you_thread_entry, 0x1234, you_buff, 1024,
        3, 3, TX_NO_TIME_SLICE, TX_AUTO_START);
}
```

在main函数中添加头文件`#include "tx_api.h"`后，调用`tx_kernel_enter();`即可运行任务。

记录一下踩过的坑

①STM32CubeMX生成的工程默认不勾选Reset and Run，程序下载后不运行，建议勾选。
②tx_initialize_low_level.s文件中的系统时钟最好与STM32CubeMX中保持移植，否则可能出现未知的运行结果。
③开发时建议将KEIL的代码优化设置为0