RT-Thread 学习笔记--线程创建与官方 Pin 设备驱动调用

严文年-2018-01-08 记于苏州

一、背景：

通过简单实例，初步了解 RT-Thread 以下功能：

- ENV 工具简单使用方法。
- 实际工程中线程的创建方法。
- RT-Thread BSP 中官方提供的设备驱动调用。

二、配置环境：

- 系统：Windows 7 旗舰版 （64 位）
- ENV：env_released_0.6.4
- RT-Thread：rt-thread-3.0.2
- IDE：Keil 5.24
- 下载器：J-Link V9.3 | ST-Link V2
- 目标芯片：STM32F103RCT6

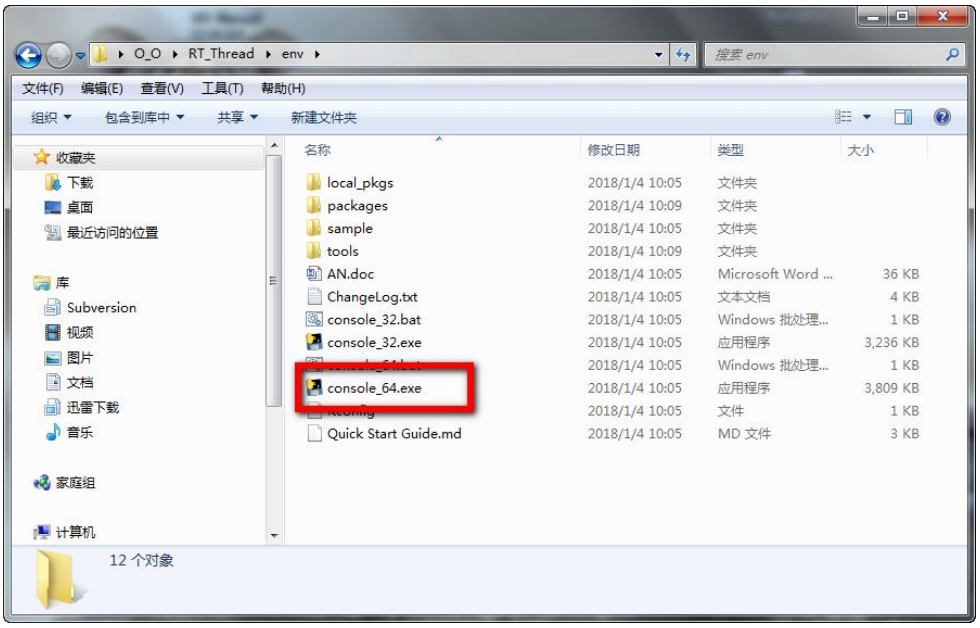
三、参考文档：（编程指南第 2 章，第六章）

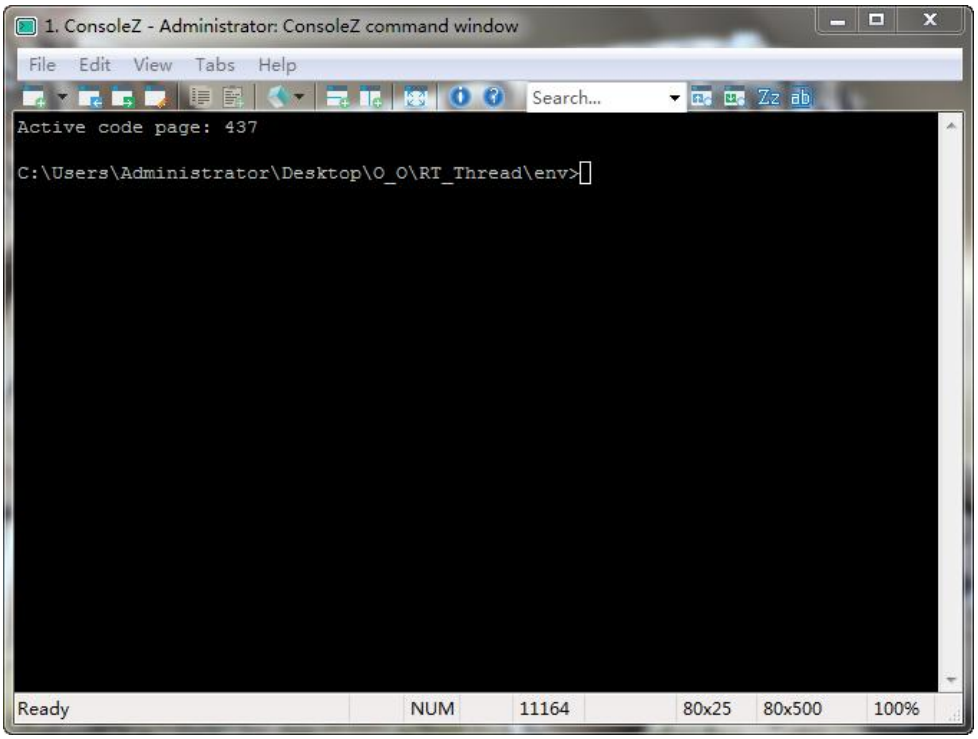
- 2 线程调度与管理
  - 2.1 实时系统的需求
  - 2.2 线程调度器
  - 2.3 线程控制块
  - 2.4 线程状态
  - 2.5 空闲线程
  - 2.6 调度器相关接口
  - 2.7 线程相关接口
    - 2.7.1 线程创建
    - 2.7.2 线程删除
    - 2.7.3 线程初始化
    - 2.7.4 线程脱离
    - 2.7.5 线程启动
    - 2.7.6 当前线程
    - 2.7.7 线程让出处理器
    - 2.7.8 线程睡眠
    - 2.7.9 线程挂起
    - 2.7.10 线程恢复
    - 2.7.11 线程控制
    - 2.7.12 初始化空闲线程
    - 2.7.13 设置空闲线程钩子
  - 2.8 线程设计

- 6 I/O设备管理
  - 6.1 块设备
  - 6.2 I/O设备控制块
  - 6.3 I/O设备管理接口
    - 6.3.1 注册设备
    - 6.3.2 移除设备
    - 6.3.3 初始化所有设备
    - 6.3.4 查找设备
    - 6.3.5 初始化设备
    - 6.3.6 打开设备
    - 6.3.7 关闭设备
    - 6.3.8 读设备
    - 6.3.9 写设备
    - 6.3.10 控制设备
    - 6.3.11 设置数据接收指示
    - 6.3.12 设置发送完成指示
  - 6.4 设备驱动

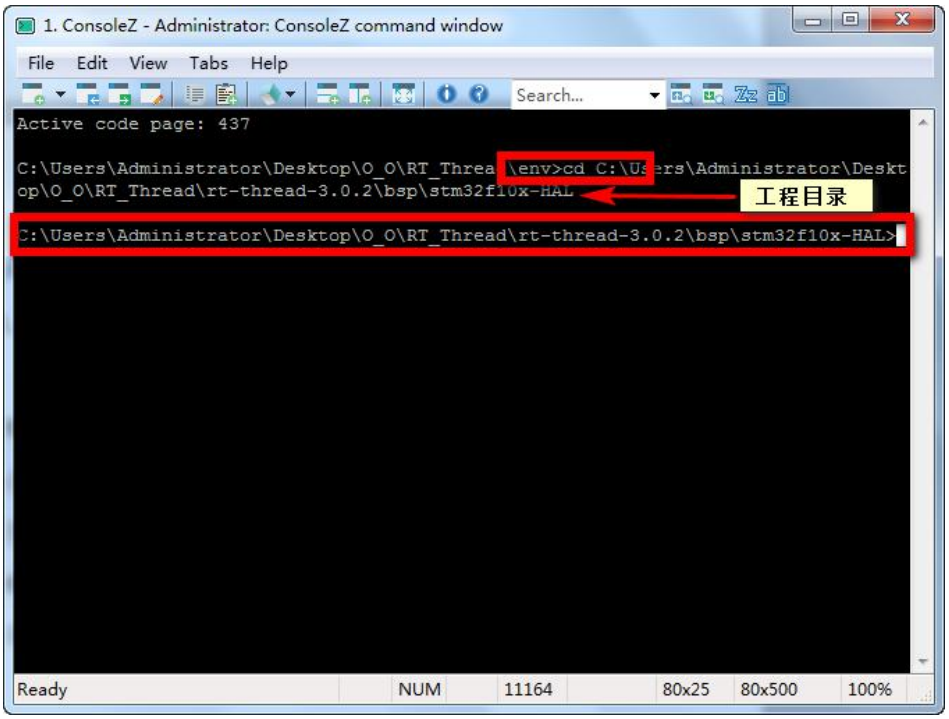
四、配置步骤：

1．打开 ENV 工具：（根据自己电脑系统选择要打开的可执行文件：console_xx.exe）

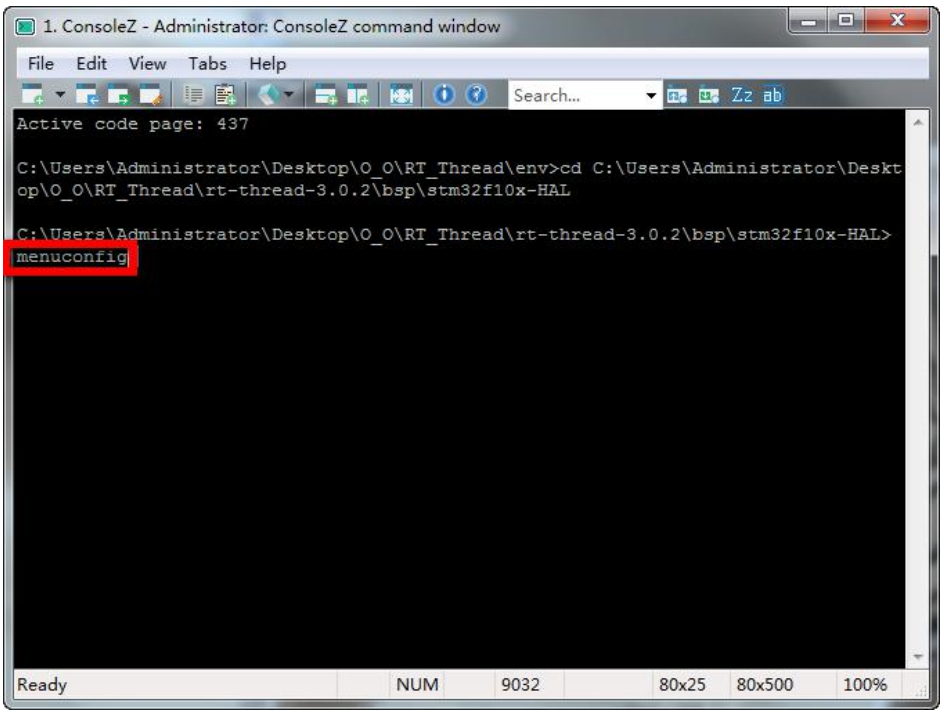


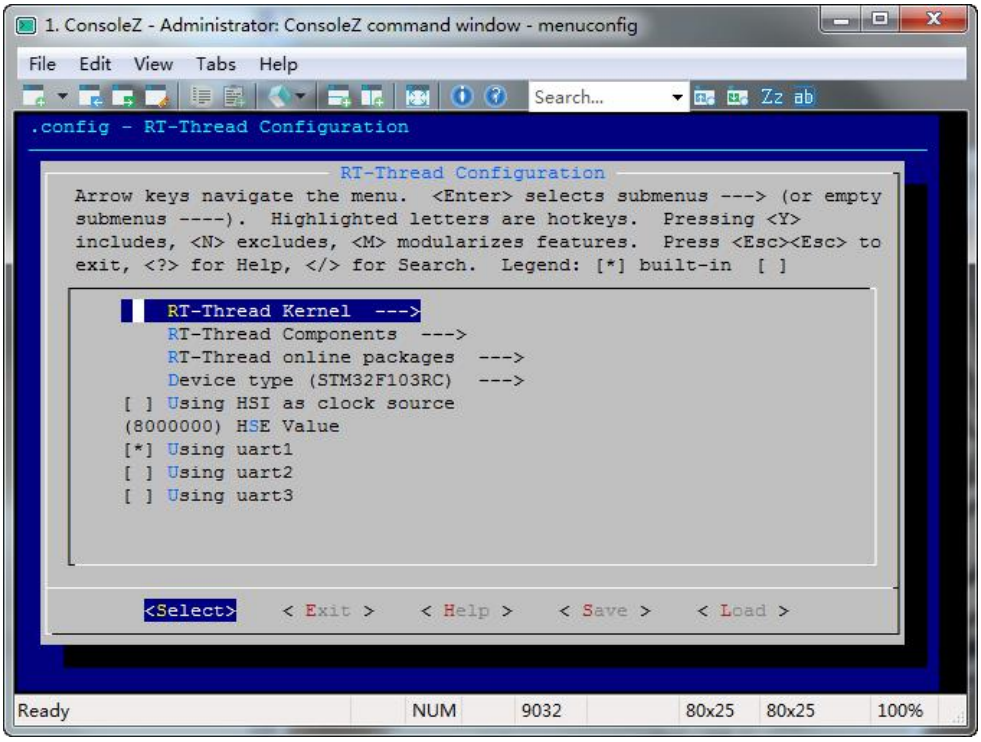


2．切换到 BSP 待配置工程目录：（命令：cd [待配置工程目录]）为方便目录切换，可将 ENV 与 RT-Thread 放置到相同根目录。

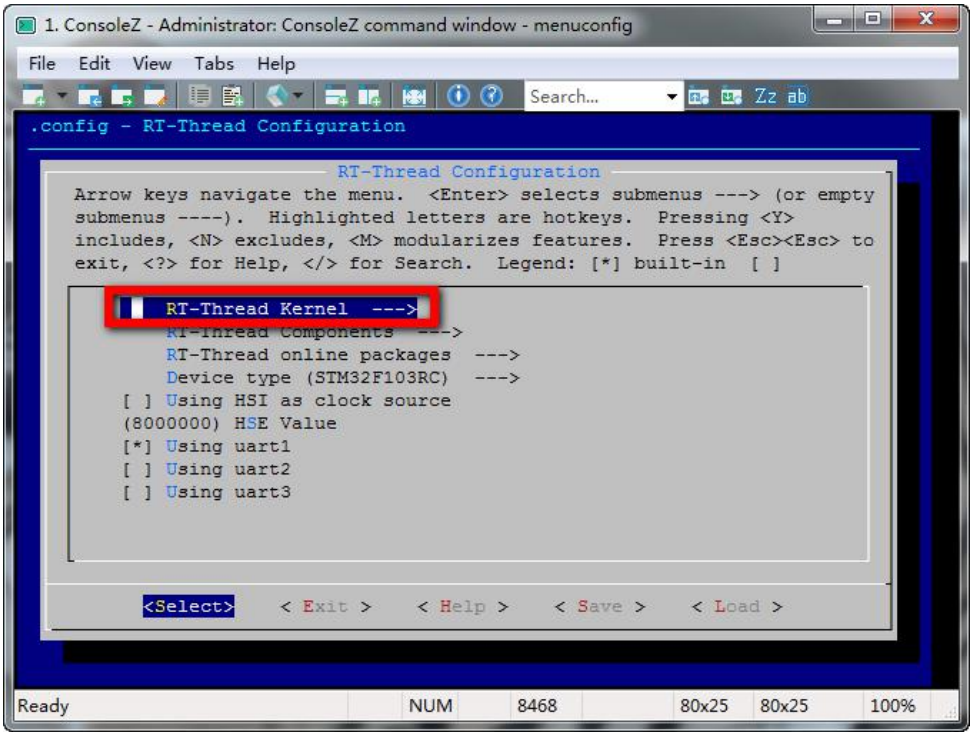


3．打开配置界面：（命令：menuconfig）

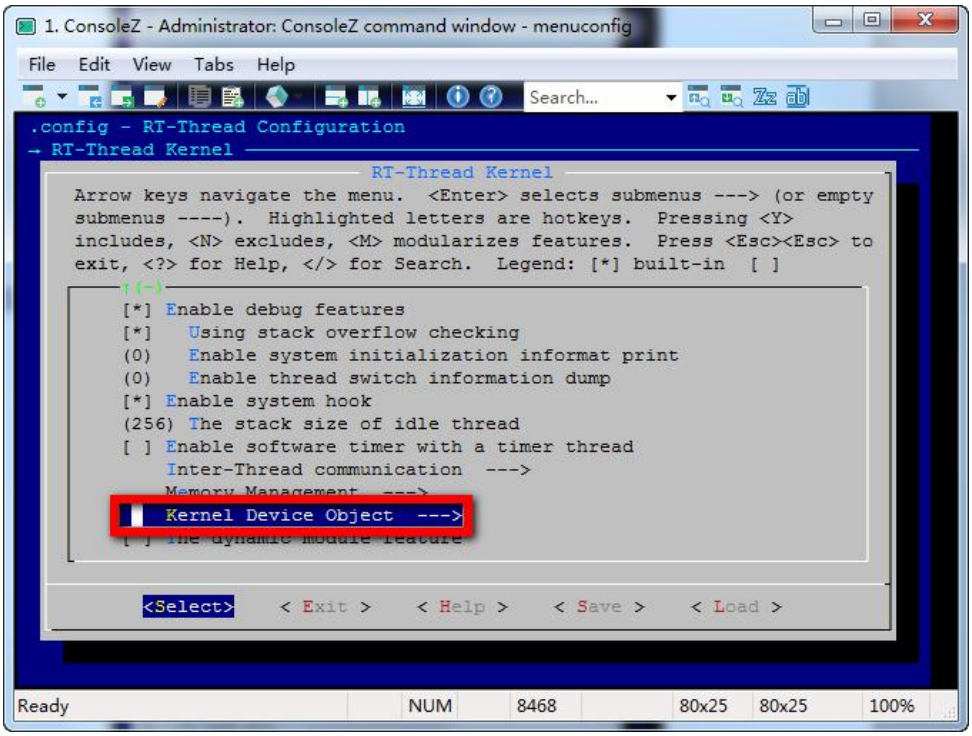




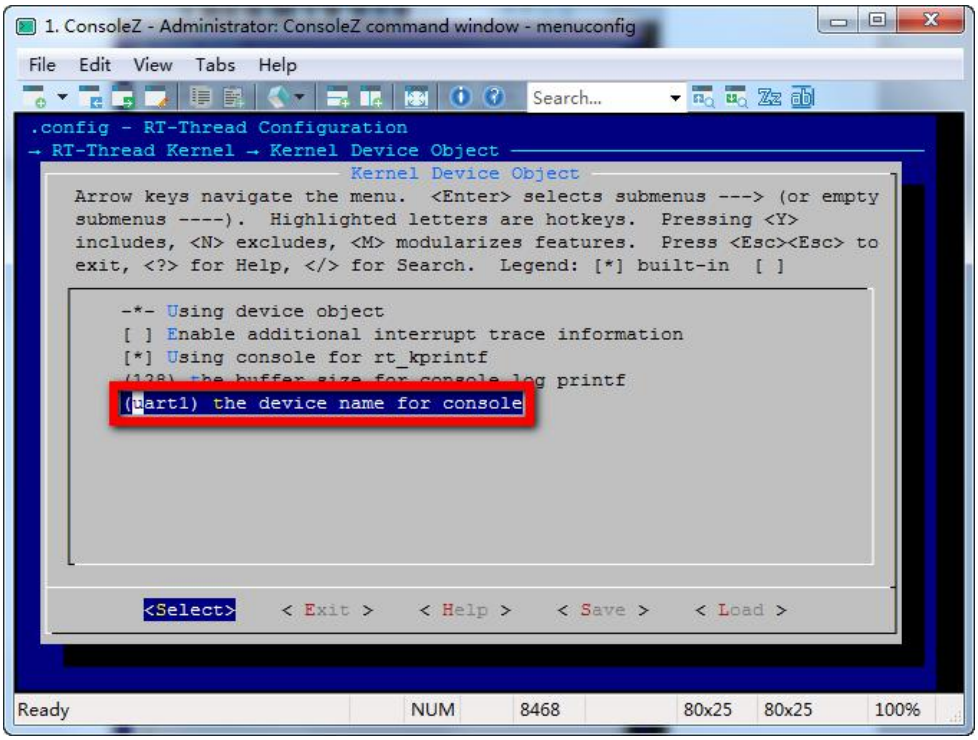
4. 配置 Finsh shell 默认串口为 uart1，进入 RT-Thread Kernel ---> 目录:



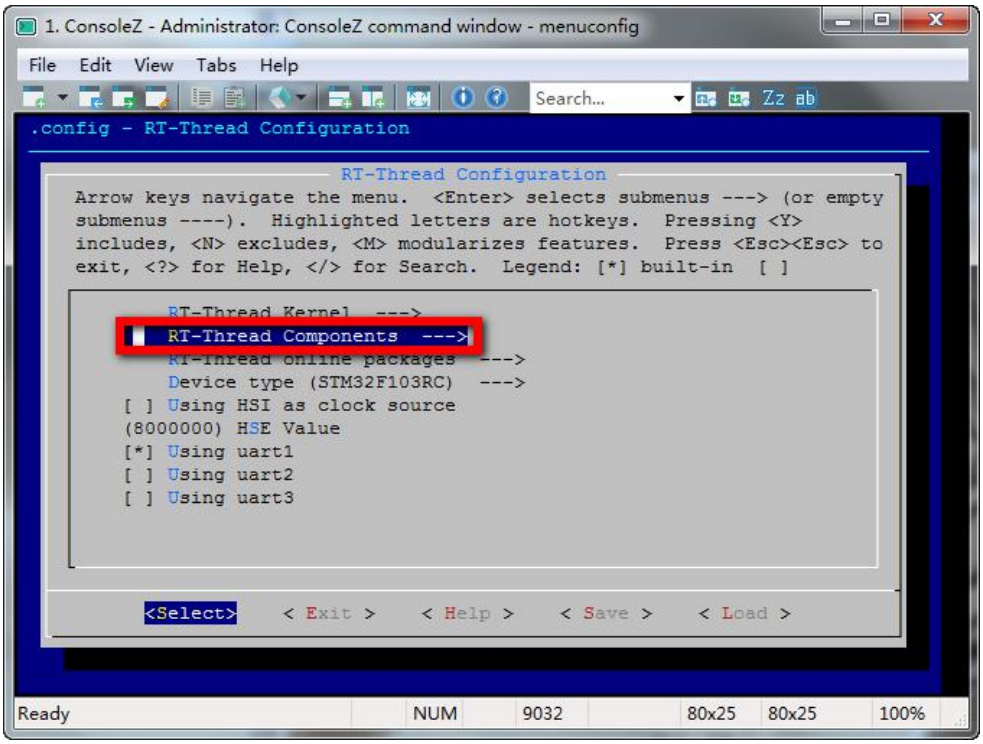
5．进入 Kernel Device Object ---->子目录：



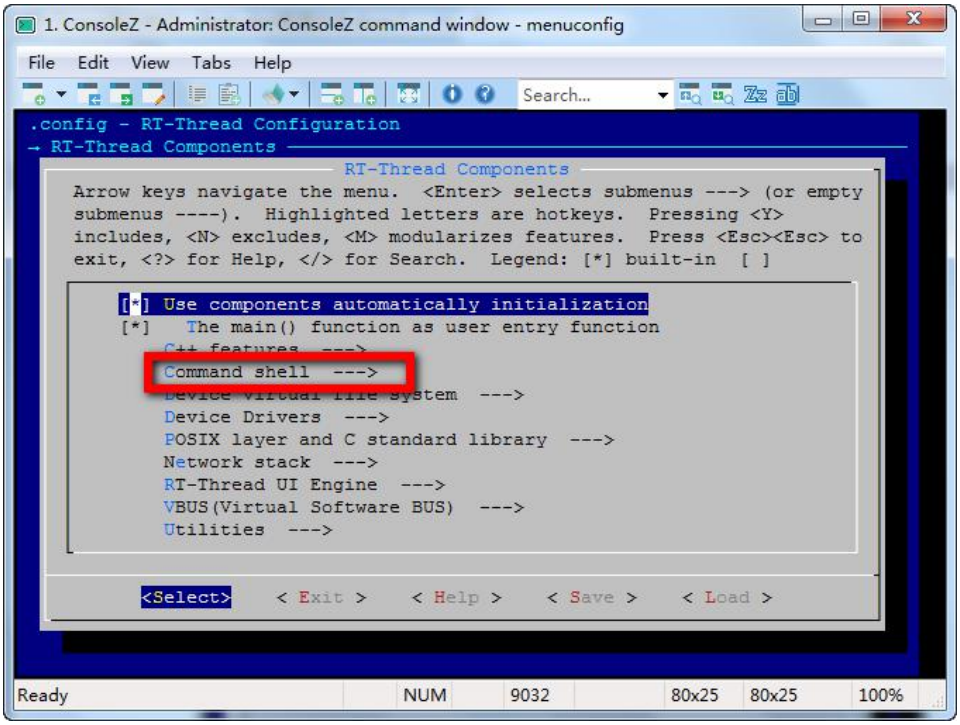
6．设置 finsh shell 默认串口为：【uart1】



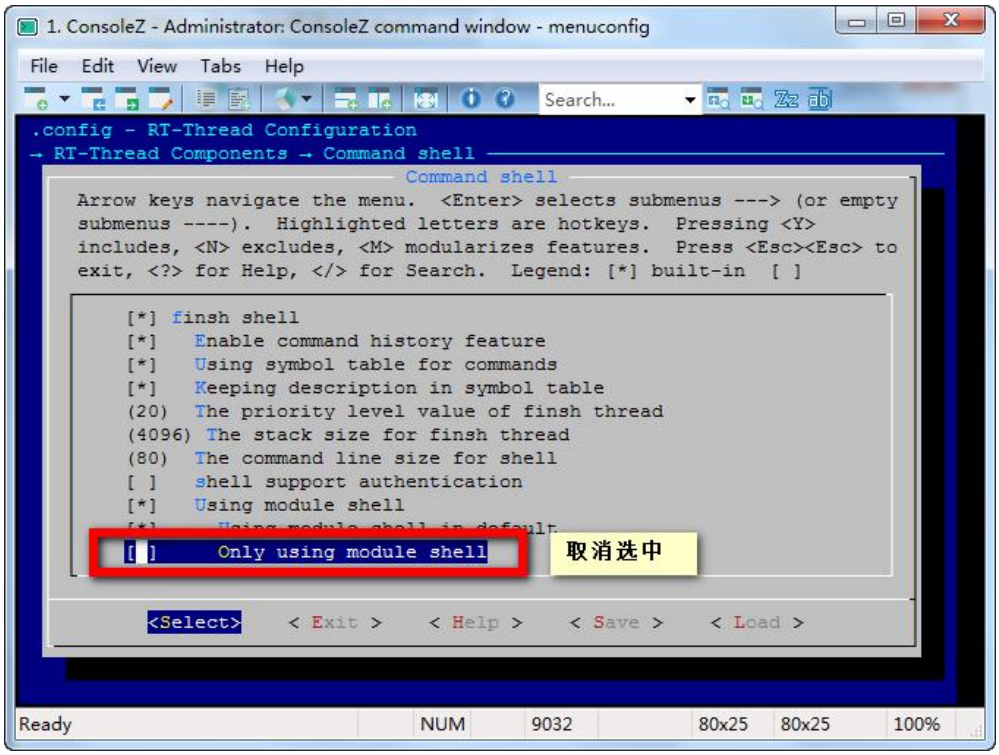
7. 设置 finsh 为全功能模式，进入 finsh 配置目录：



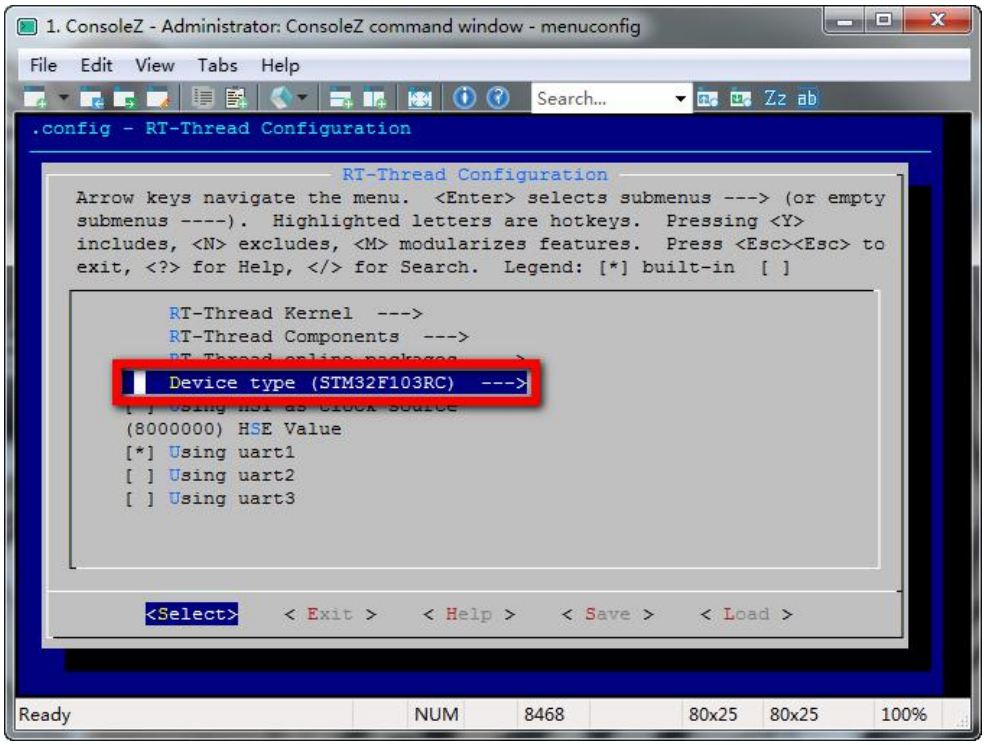
8. 进入 Command shell ---> 子目录



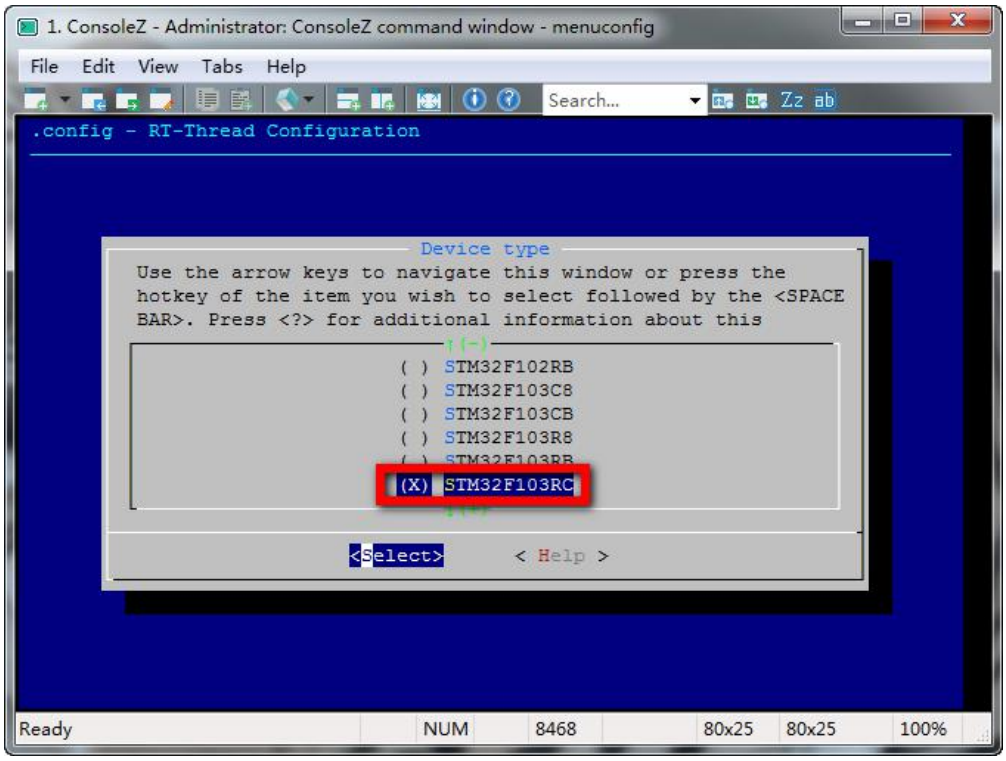
9．配置 Finsh：（取消仅为 msh 模式）



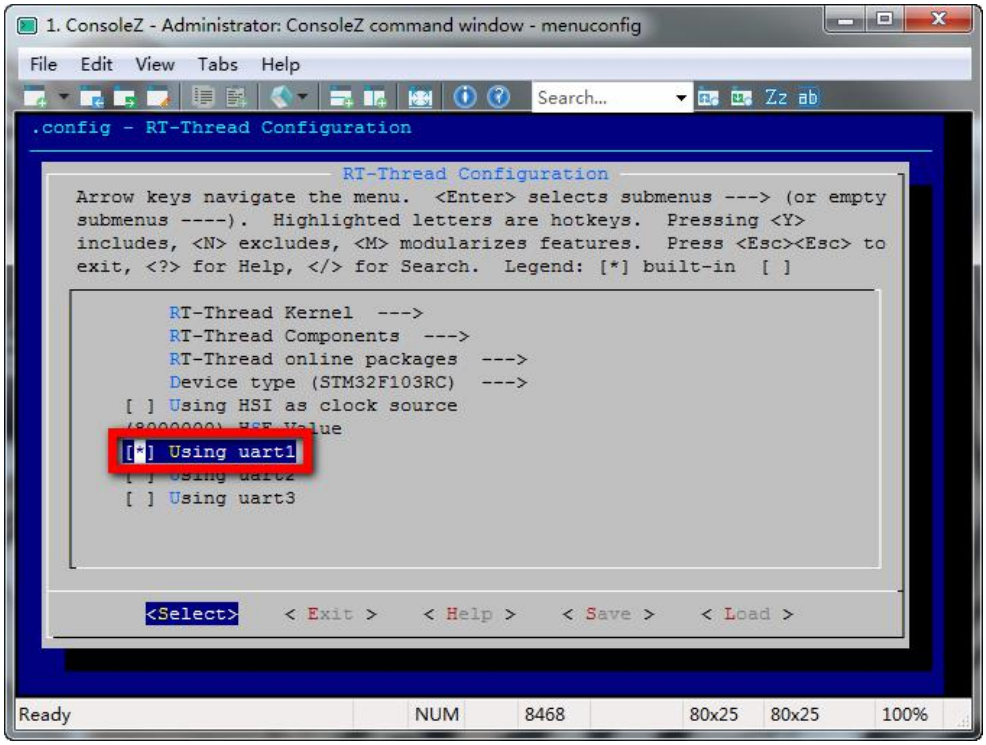
10．配置目标 IC，进入 Device type (STM32F103RC) --->目录



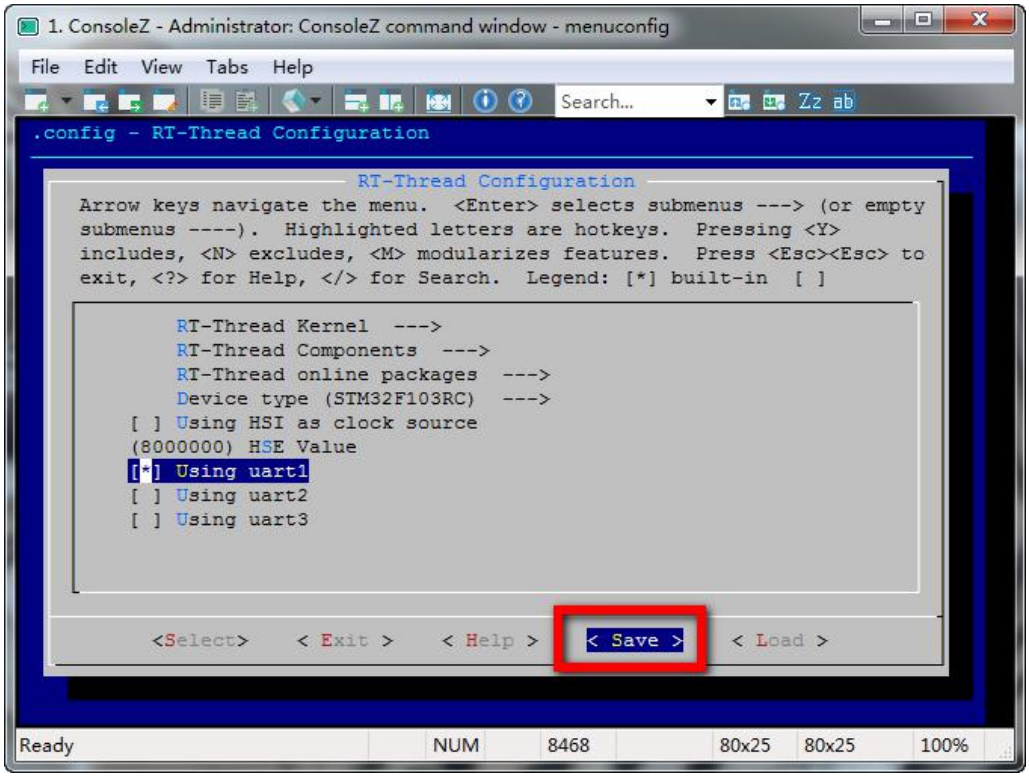
11. 选择目标 IC:

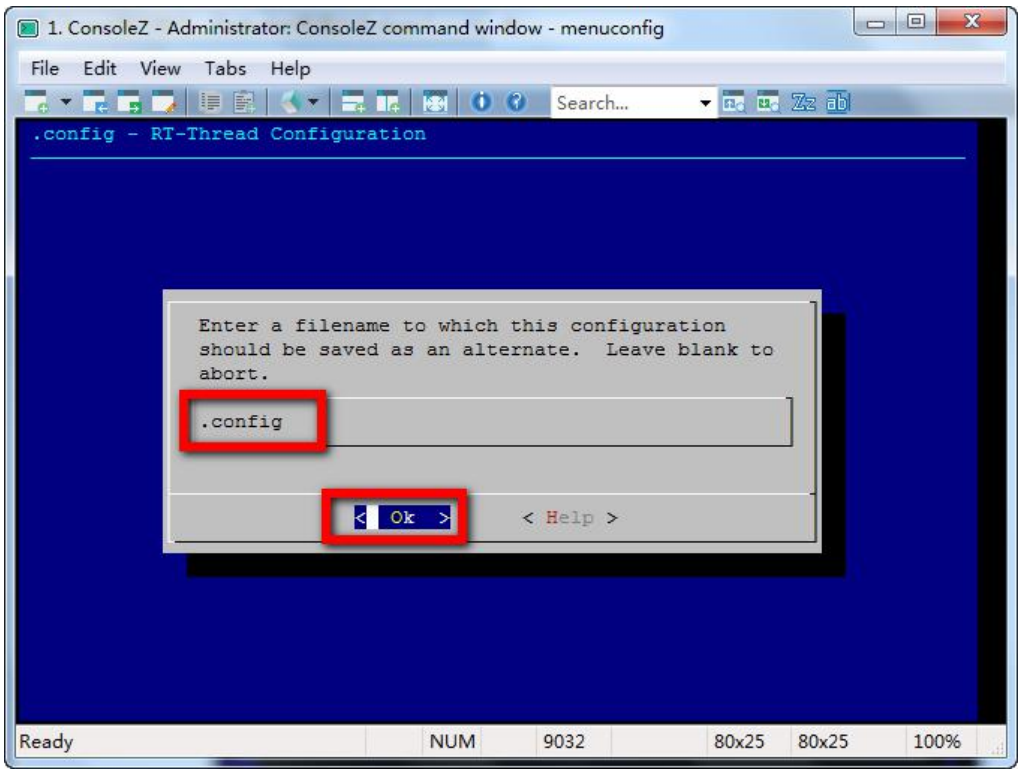


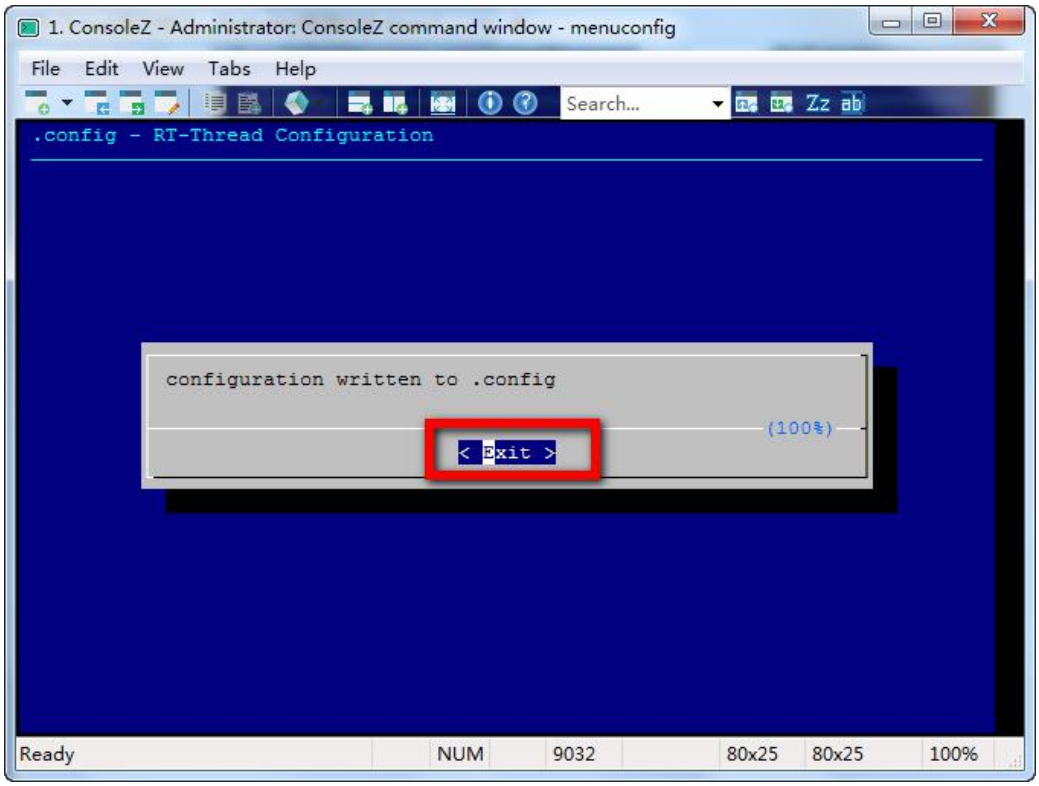
12. 选择串口 1 设备:



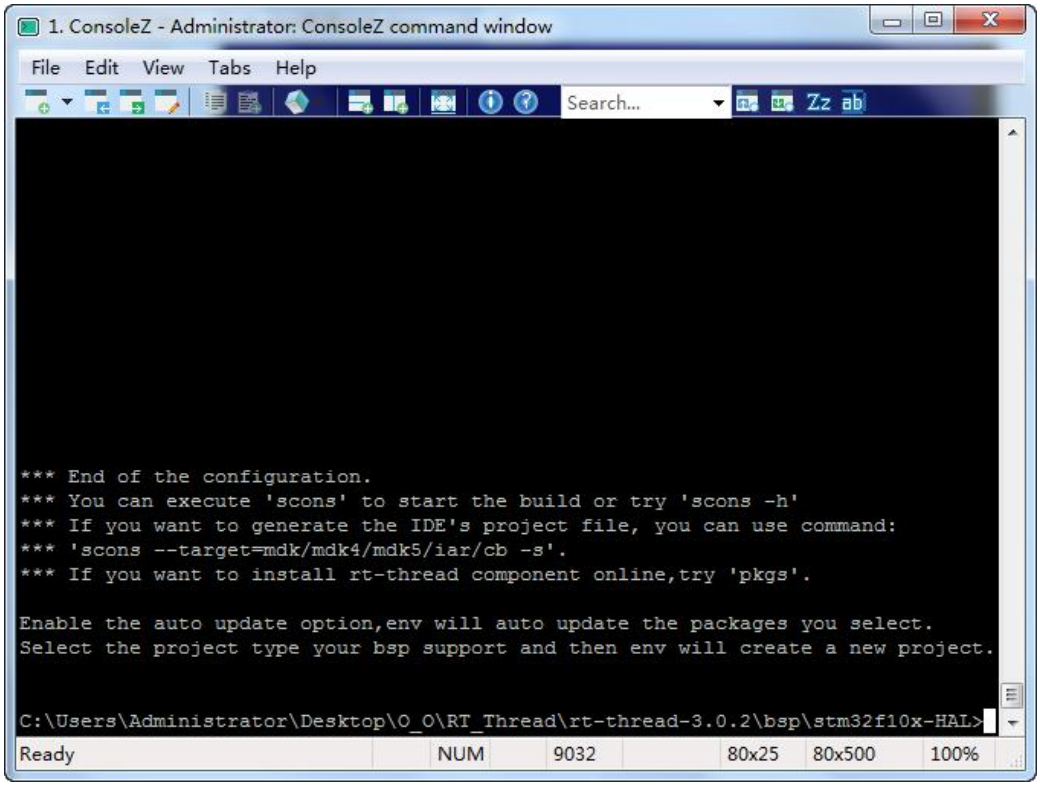
13. 保存配置：


1. ConsoleZ - Administrator: ConsoleZ command window - menuconfig
File Edit View Tabs Help
Search...
.config - RT-Thread Configuration
RT-Thread Configuration
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]
RT-Thread Kernel --->
RT-Thread Components --->
RT-Thread online packages --->
Device type (STM32F103RC) --->
[ ] Using HSI as clock source
(8000000) HSE Value
[*] Using uart1
[ ] Using uart2
[ ] Using uart3
<Select> < Exit > < Help > < Save > < Load >
Ready NUM 8468 80x25 80x25 100%


1. ConsoleZ - Administrator: ConsoleZ command window - menuconfig
File Edit View Tabs Help
Search...
.config - RT-Thread Configuration
Enter a filename to which this configuration should be saved as an alternate. Leave blank to abort.
.config
< Ok > < Help >
Ready NUM 9032 80x25 80x25 100%

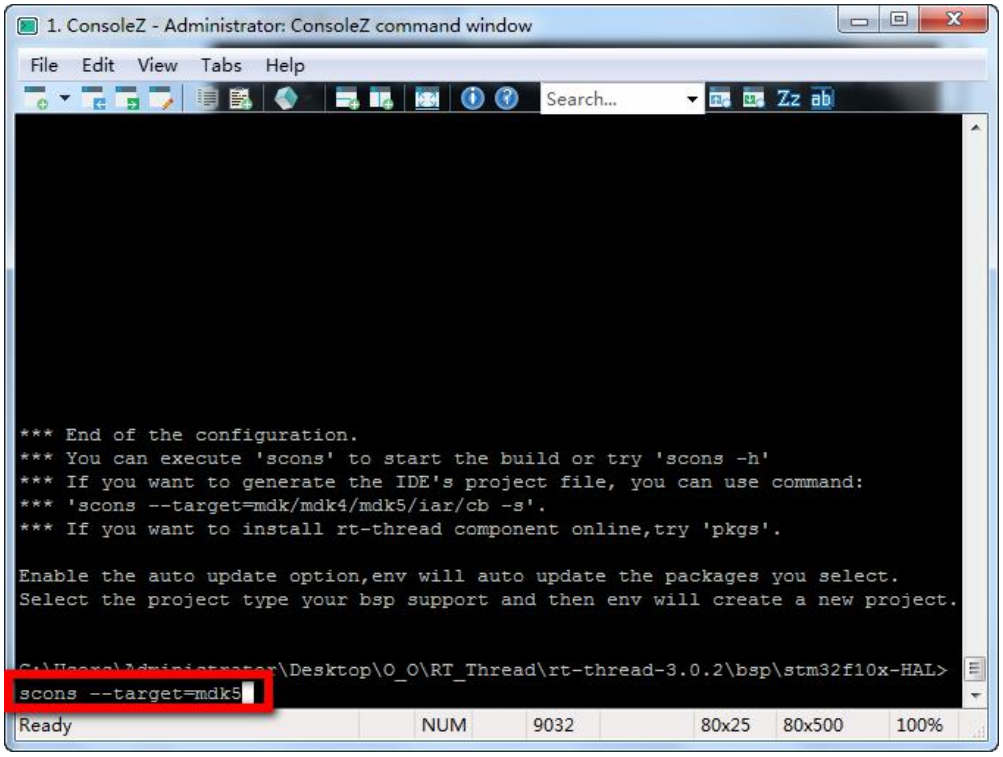


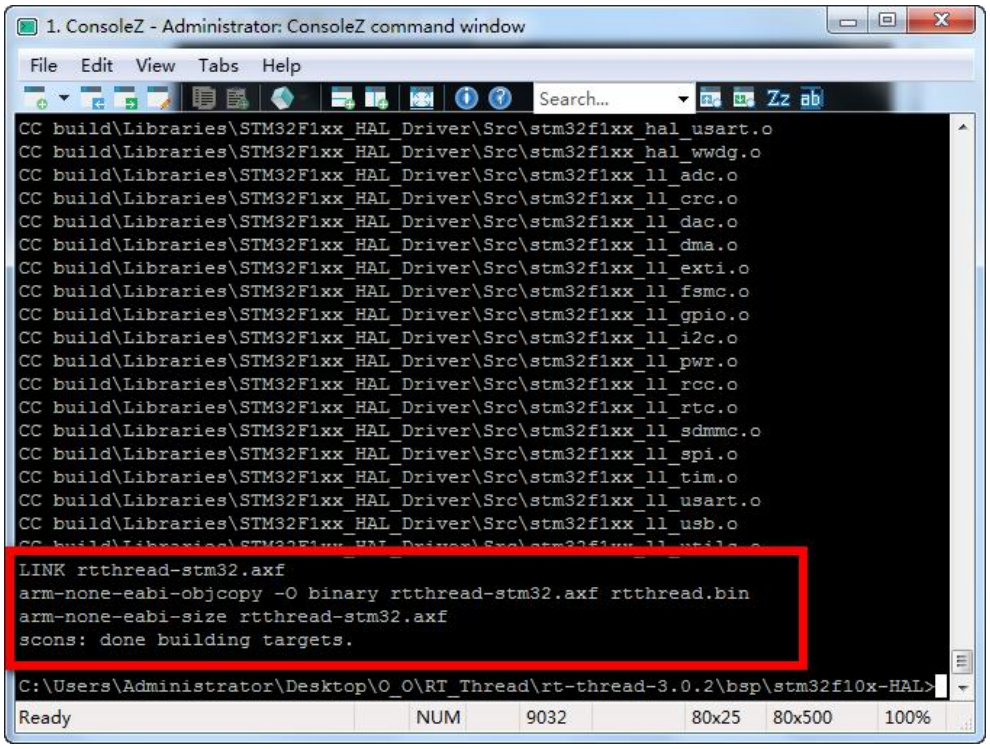
14. 退出 ENV 配置：（连续按键盘 ESC 按键）



五、更新工程：

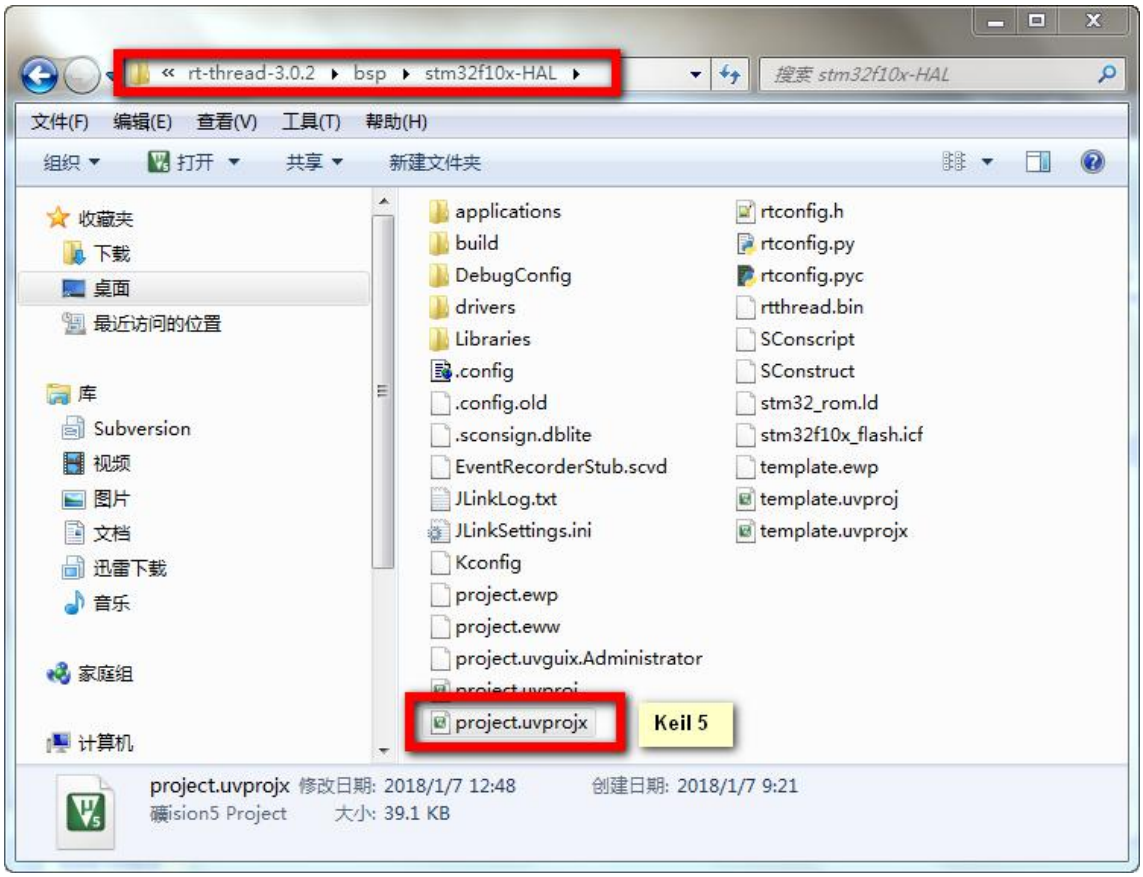
1. 将重新配置的参数更新到工程文档：（命令：scons --target=mdk5）



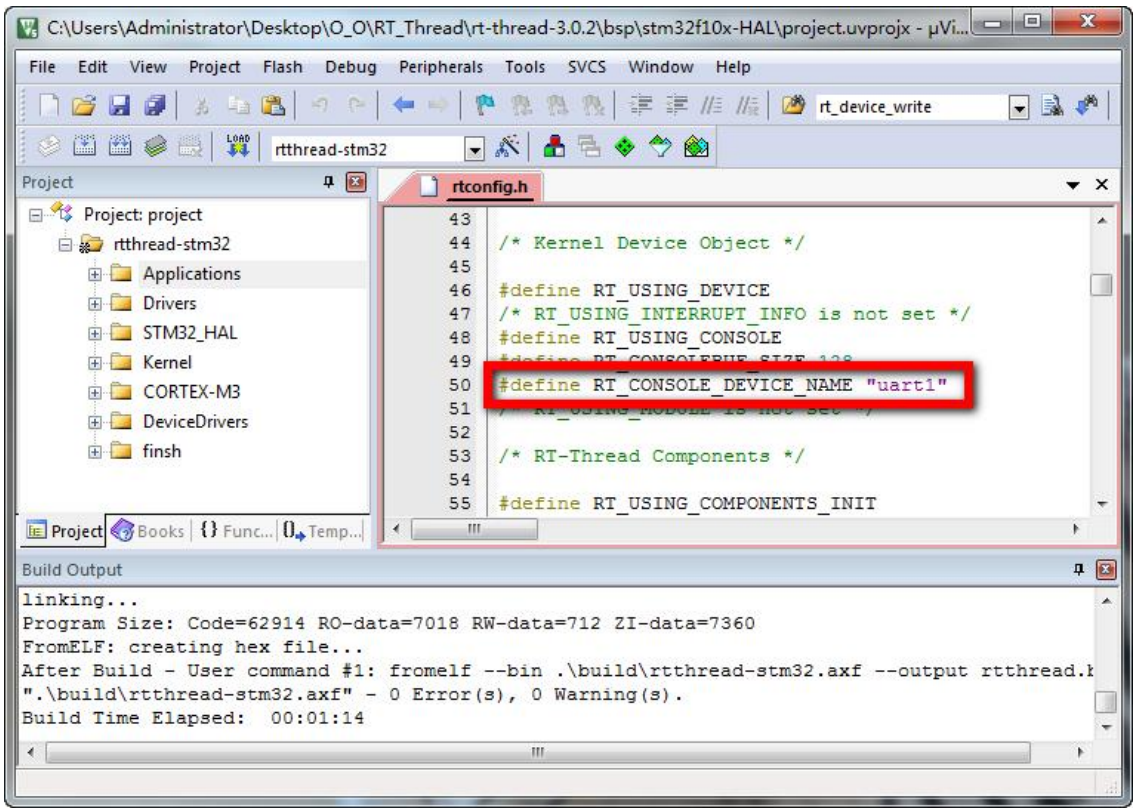


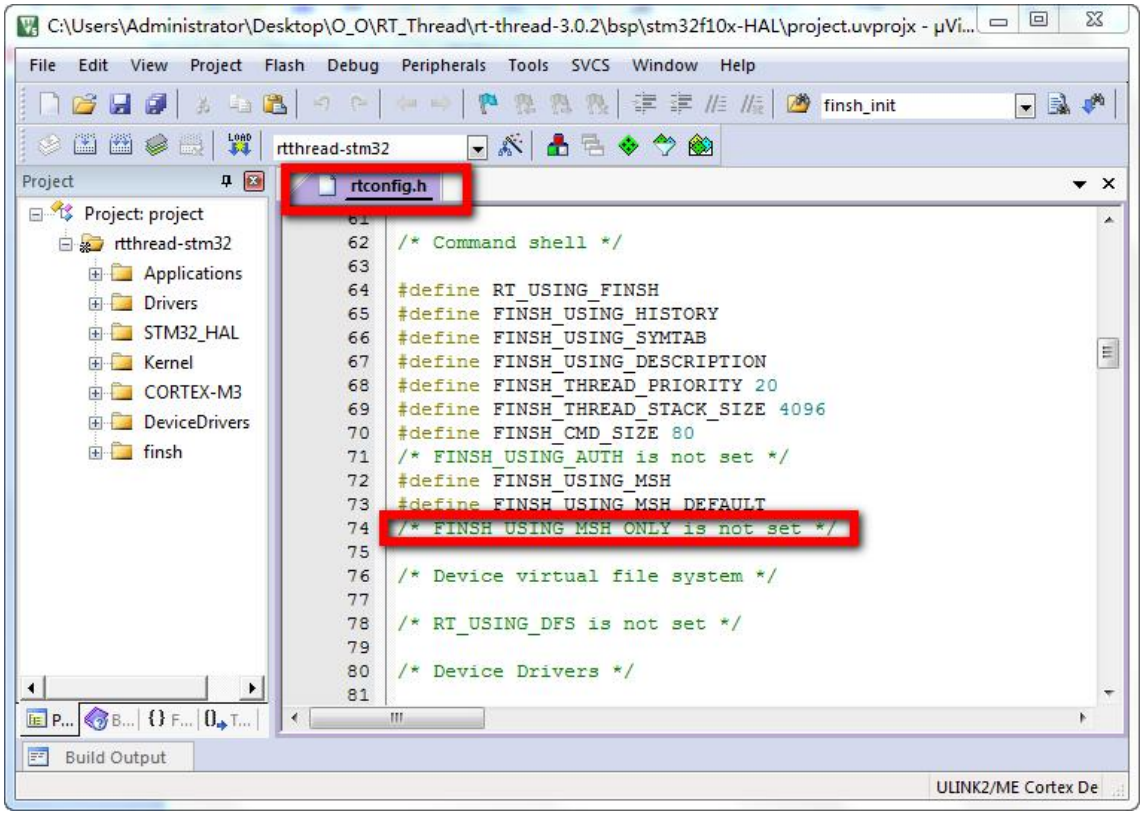
六、配置参数确认：

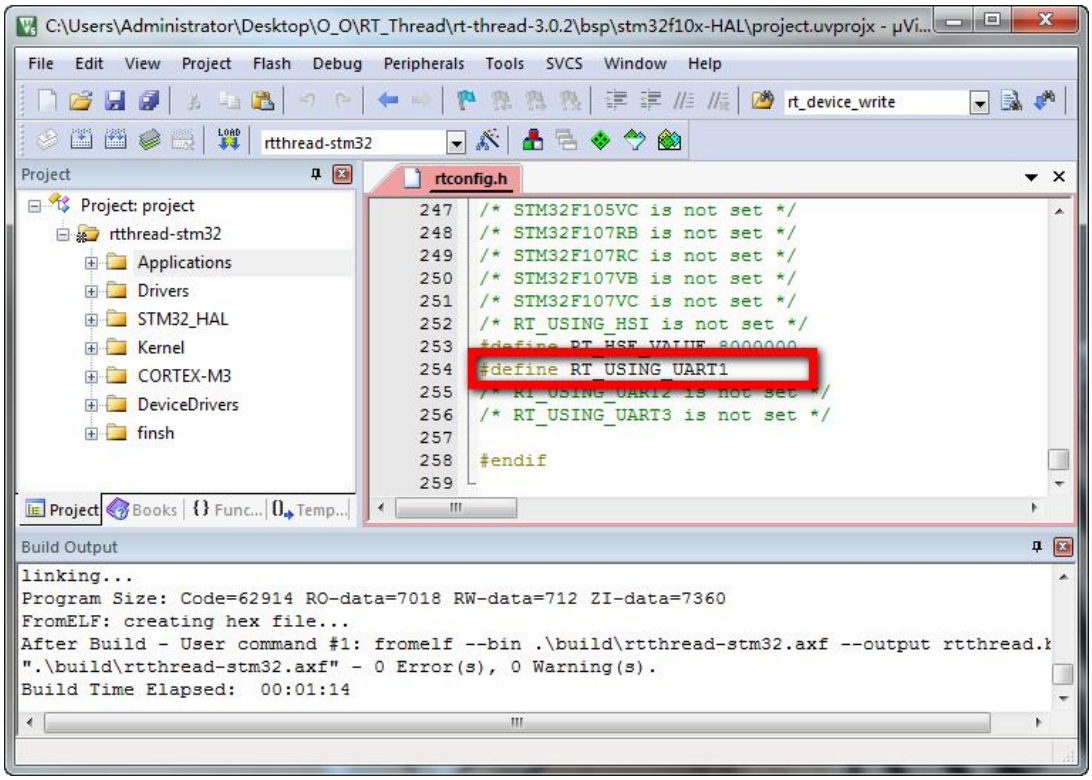
1. 打开更新后的项目工程文档



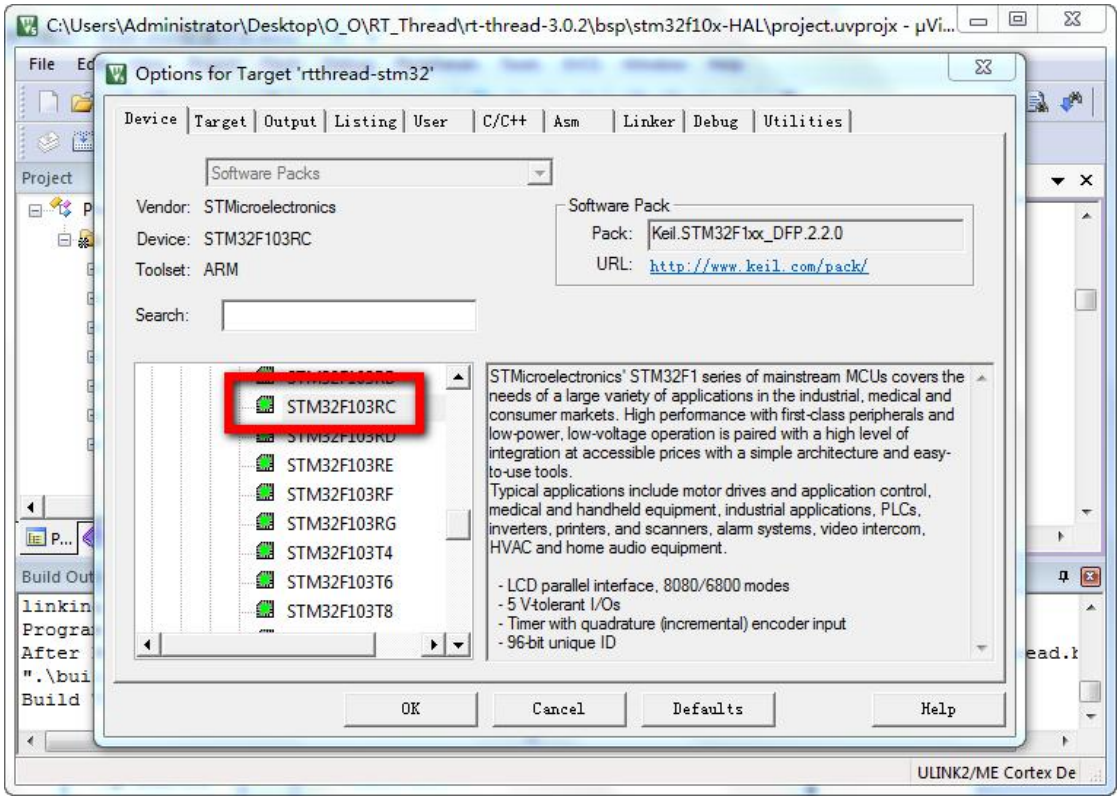
2. 确认配置信息是否更新

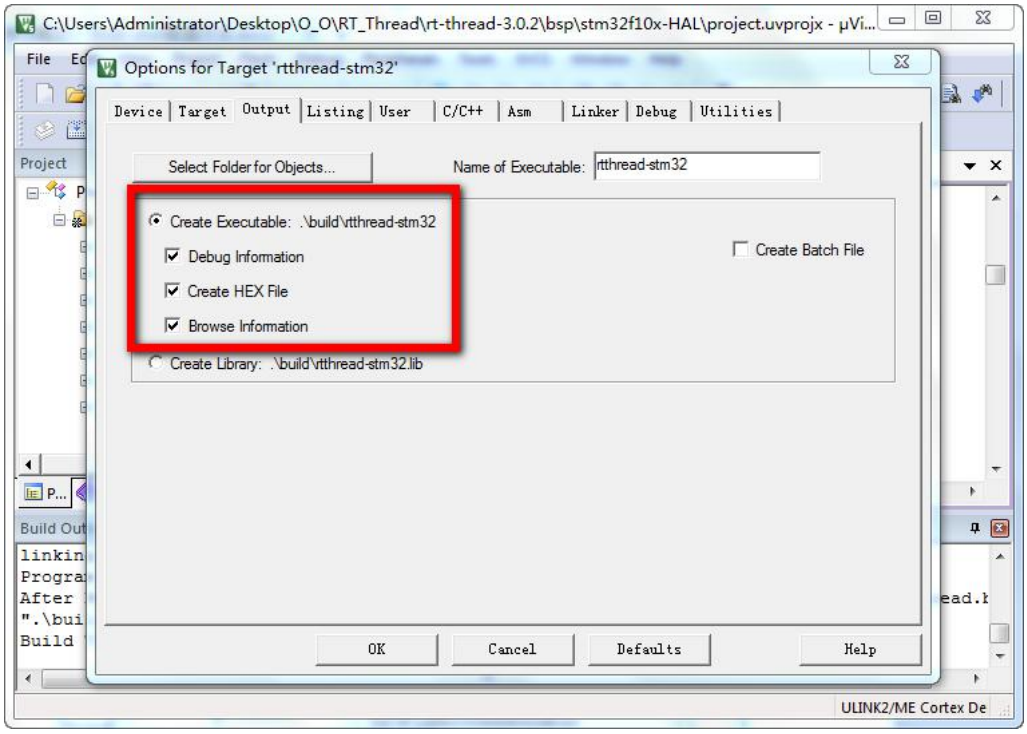

C:\Users\Administrator\Desktop\O_O\RT_Thread\rt-thread-3.0.2\bsp\stm32f10x-HAL\project.uvprojx - µVi...
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
rt_device_write
rtthread-stm32
Project
Project: project
rtthread-stm32
Applications
Drivers
STM32_HAL
Kernel
CORTEX-M3
DeviceDrivers
finsh
rtconfig.h
/* Kernel Device Object */
#define RT_USING_DEVICE
/* RT_USING_INTERRUPT_INFO is not set */
#define RT_USING_CONSOLE
#define RT_CONSOLE_DEVICE_NAME "uart1"
/* RT-Thread Components */
#define RT_USING_COMPONENTS_INIT
Build Output
linking...
Program Size: Code=62914 RO-data=7018 RW-data=712 ZI-data=7360
FromELF: creating hex file...
After Build - User command #1: fromelf --bin .\build\rtthread-stm32.axf --output rtthread.b
".\build\rtthread-stm32.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:01:14


C:\Users\Administrator\Desktop\O_O\RT_Thread\rt-thread-3.0.2\bsp\stm32f10x-HAL\project.uvprojx - µVi...
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
finsh_init
rtthread-stm32
Project
Project: project
rtthread-stm32
Applications
Drivers
STM32_HAL
Kernel
CORTEX-M3
DeviceDrivers
finsh
rtconfig.h
/* Command shell */
#define RT_USING_FINSH
#define FINSH_USING_HISTORY
#define FINSH_USING_SYMTAB
#define FINSH_USING_DESCRIPTION
#define FINSH_THREAD_PRIORITY 20
#define FINSH_THREAD_STACK_SIZE 4096
#define FINSH_CMD_SIZE 80
/* FINSH_USING_AUTH is not set */
#define FINSH_USING_MSH
#define FINSH_USING_MSH_DEFAULT
/* FINSH_USING_MSH_ONLY is not set */
/* Device virtual file system */
/* RT_USING_DFS is not set */
/* Device Drivers */
Build Output
ULINK2/ME Cortex De



3. 配置 Keil 相关设置



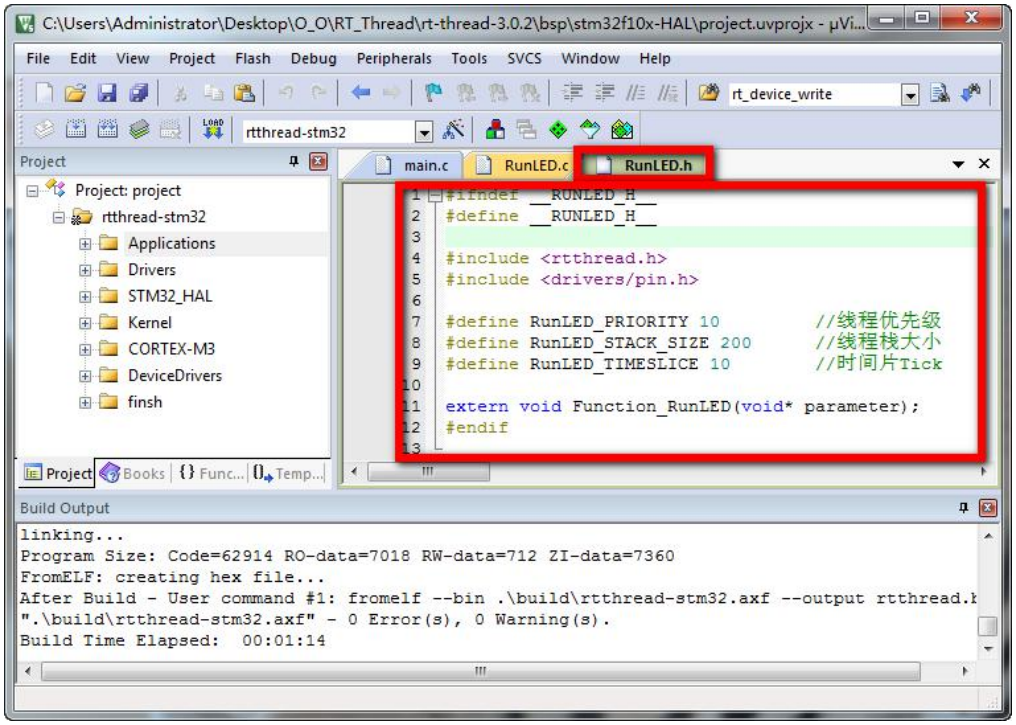


七、添加线程创建代码 及 pin 设备驱动调用代码：

1. 新建：RunLED.h 文件，并添加如下代码：

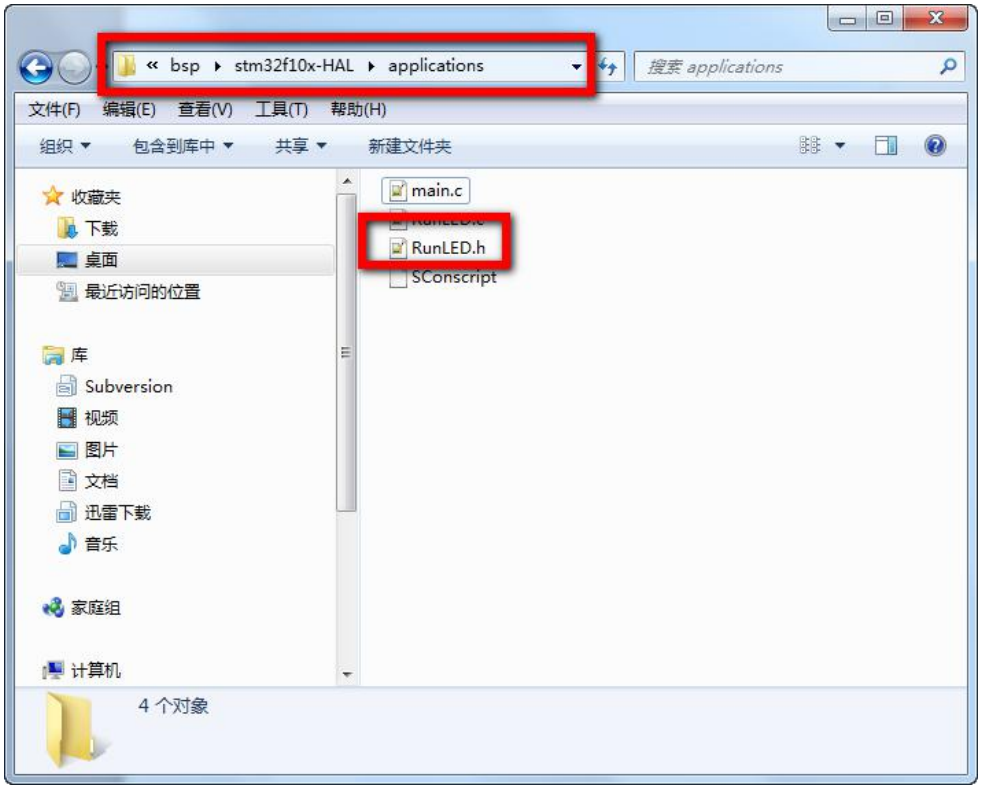
代码：

```
#ifndef __RUNLED_H__
#define __RUNLED_H__

#include <rtthread.h>
#include <drivers/pin.h>

#define RunLED_PRIORITY 10
#define RunLED_STACK_SIZE 200
#define RunLED_TIMESLICE 10

extern void Function_RunLED(void* parameter);
#endif
```



2. 将文件保存到如下目录:

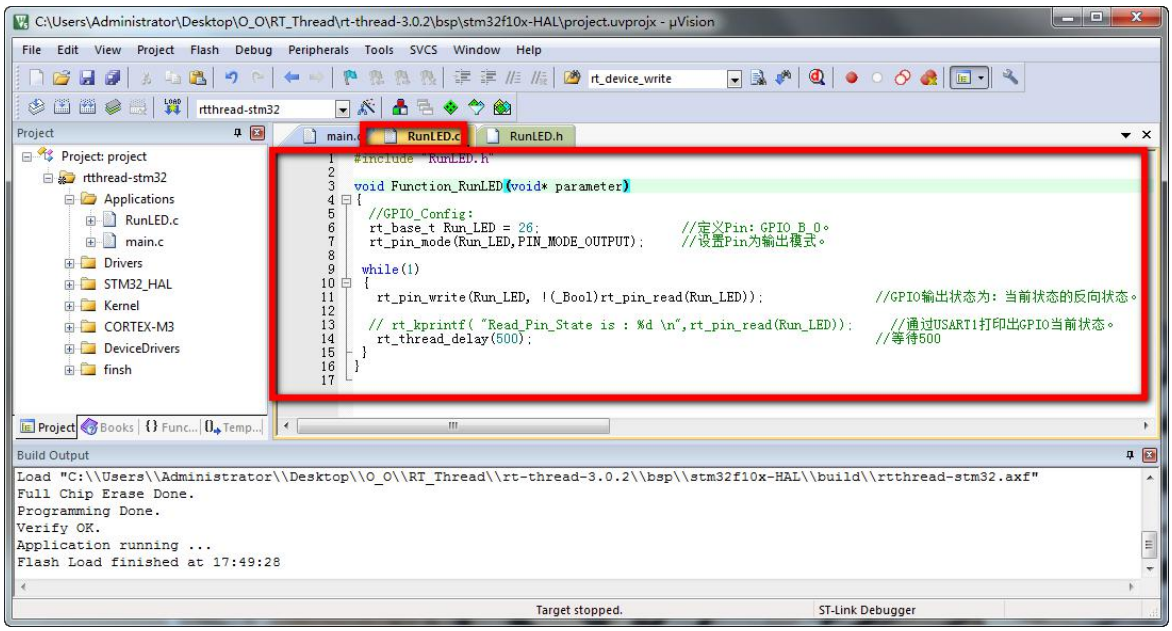
【rt-thread-3.0.2\bsp\stm32f10x-HAL\applications】



3. 新建：RunLED.c 文件，并添加如下代码：

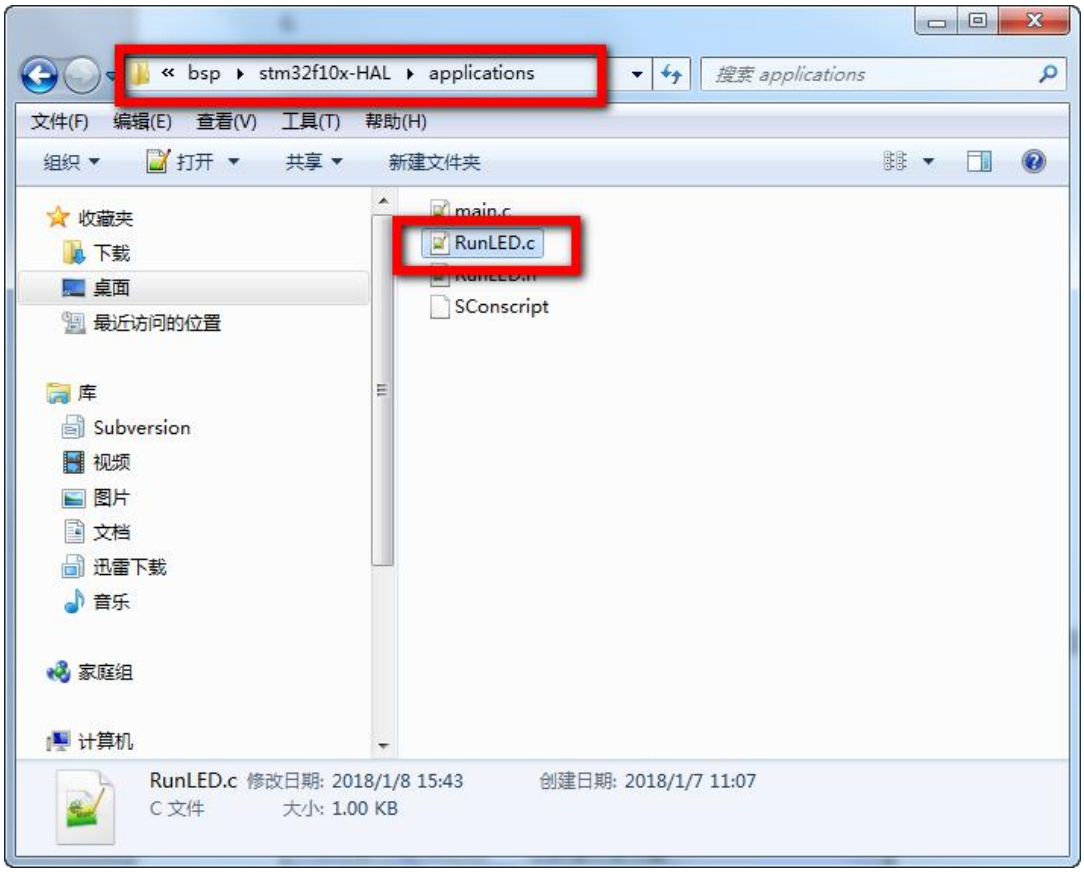
代码：

```
#include "RunLED.h"


void Function_RunLED(void* parameter)
{
    rt_base_t   Run_LED = 26;
    rt_pin_mode(Run_LED,PIN_MODE_OUTPUT);


   while(1)
  {
    rt_pin_write(Run_LED,!(_Bool)rt_pin_read(Run_LED));
    // rt_kprintf( "Read_Pin_State is : %d \n",rt_pin_read(Run_LED));
     rt_thread_delay(500);
   }
}
```



4. 将文件保存到如下目录：

【rt-thread-3.0.2\bsp\stm32f10x-HAL\applications】

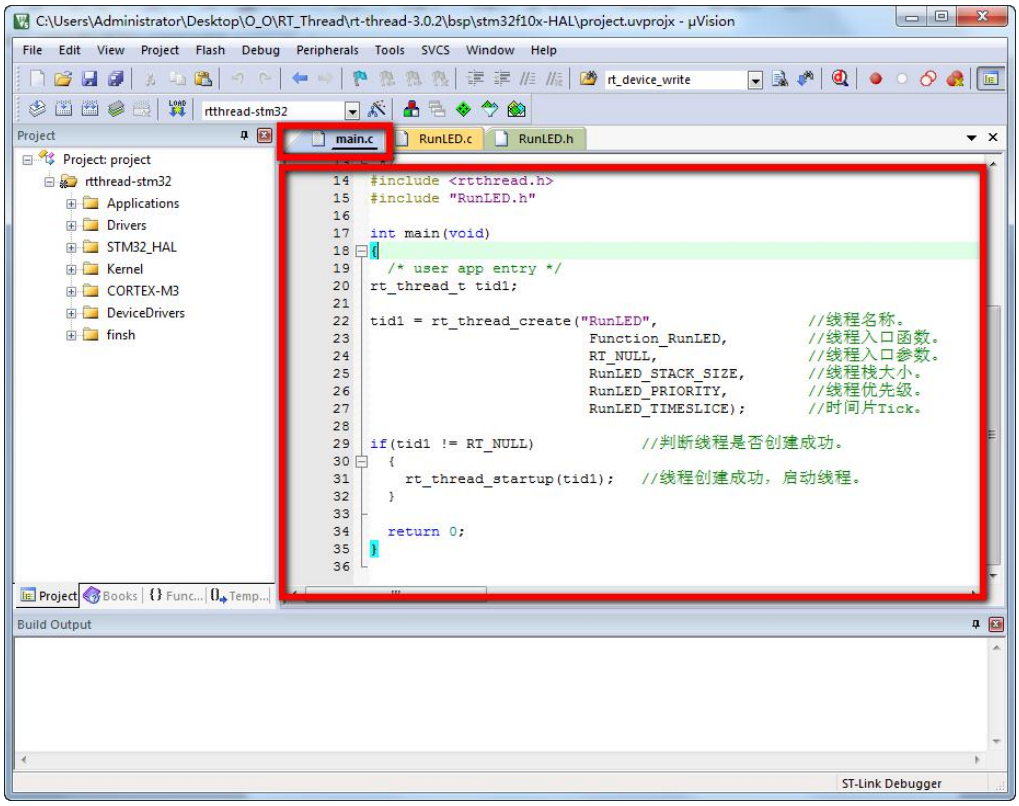


5. 打开：main.c 文件，并添加如下代码：

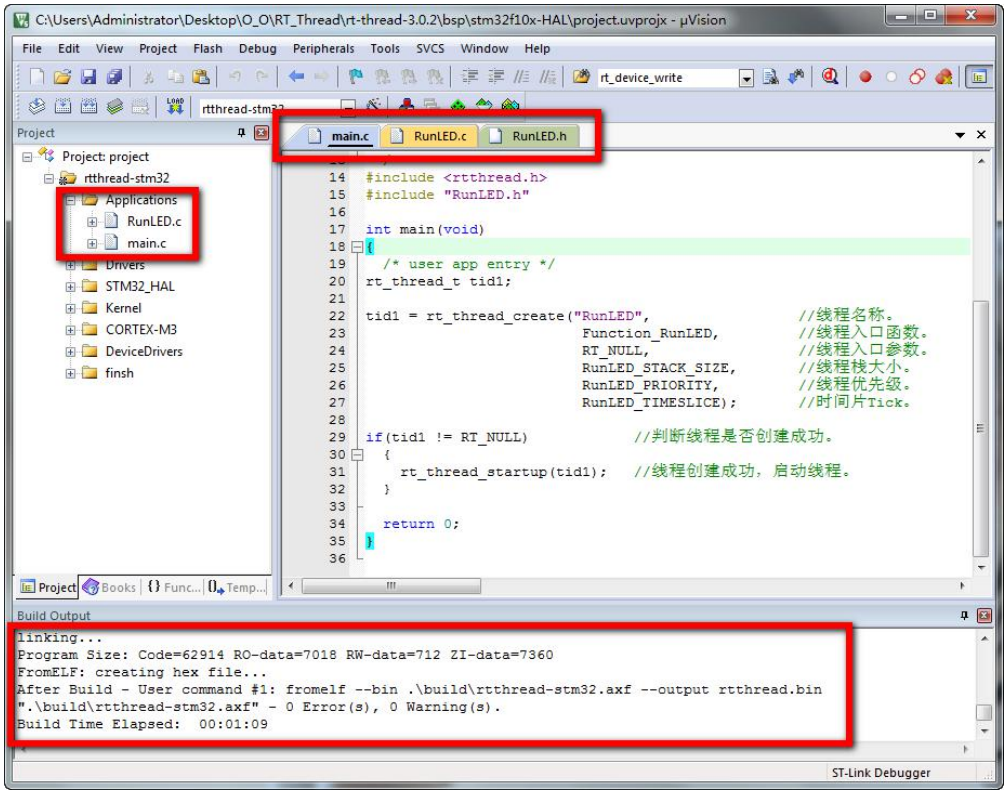
代码：

```
#include <rtthread.h>
#include "RunLED.h"

int main(void)
{
  /* user app entry */
rt_thread_t tid1;

tid1 = rt_thread_create("RunLED",                    //线程名称。
                        Function_RunLED,             //线程入口函数。
                        RT_NULL,                     //线程入口参数。
                        RunLED_STACK_SIZE,           //线程栈大小。
                        RunLED_PRIORITY,             //线程优先级。
                        RunLED_TIMESLICE);           //时间片 Tick。

if(tid1 != RT_NULL)                                  //判断线程是否创建成功。
    {
        rt_thread_startup(tid1);                     //线程创建成功，启动线程。
    }

  return 0;
}
```

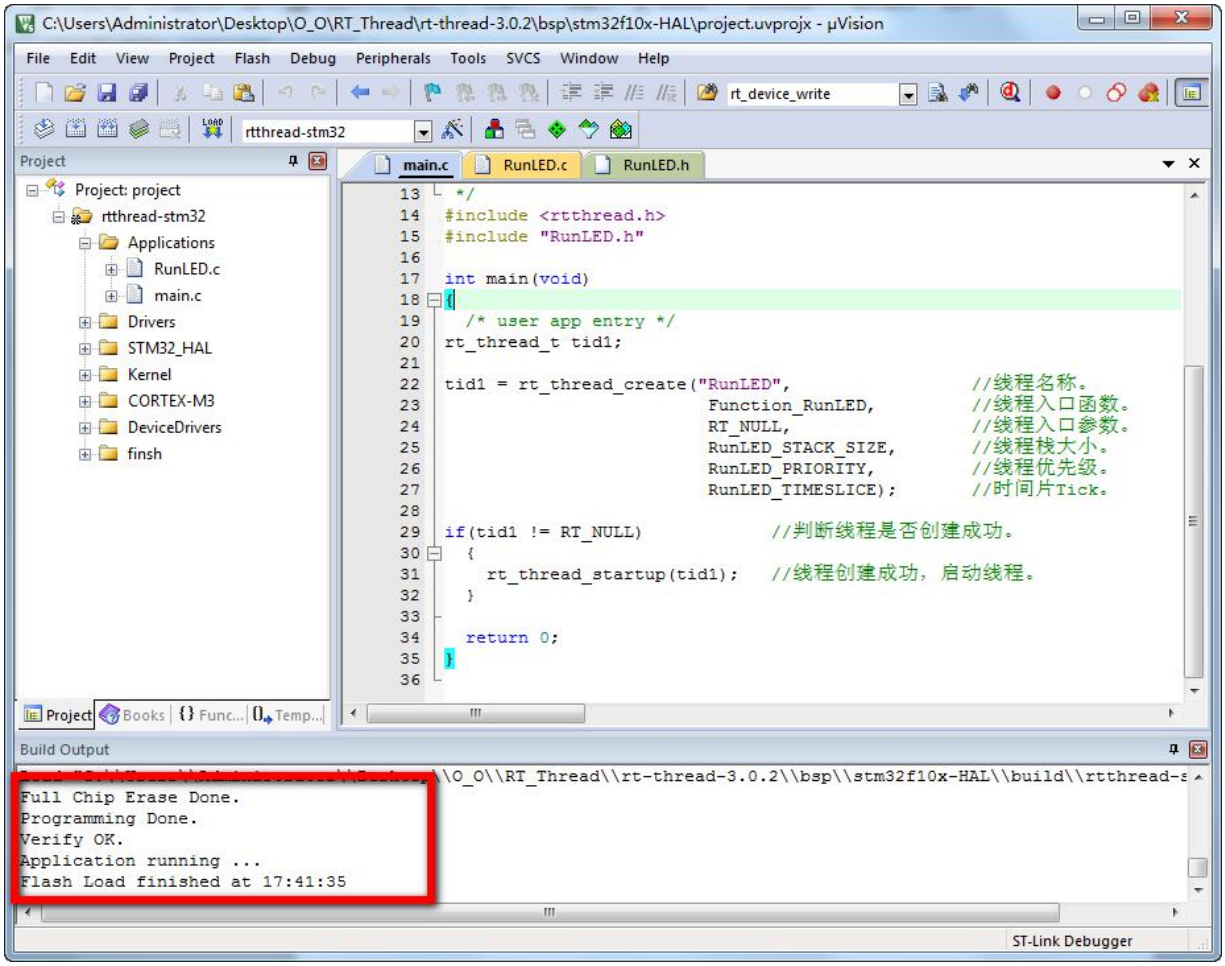
6. 保存 main.c 文件：


C:\Users\Administrator\Desktop\O_O\RT_Thread\rt-thread-3.0.2\bsp\stm32f10x-HAL\project.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
rt_device_write
rtthread-stm32
Project
main.c
RunLED.c
RunLED.h
Project: project
rtthread-stm32
Applications
Drivers
STM32_HAL
Kernel
CORTEX-M3
DeviceDrivers
finsh
#include <rtthread.h>
#include "RunLED.h"
int main(void)
{
/* user app entry */
rt_thread_t tid1;
tid1 = rt_thread_create("RunLED", //线程名称。
Function_RunLED, //线程入口函数。
RT_NULL, //线程入口参数。
RunLED_STACK_SIZE, //线程栈大小。
RunLED_PRIORITY, //线程优先级。
RunLED_TIMESLICE); //时间片Tick。
if(tid1 != RT_NULL) //判断线程是否创建成功。
{
rt_thread_startup(tid1); //线程创建成功，启动线程。
}
return 0;
}
Project Books {} Func... Temp...
Build Output
ST-Link Debugger

7. 编译工程文档：


C:\Users\Administrator\Desktop\O_O\RT_Thread\rt-thread-3.0.2\bsp\stm32f10x-HAL\project.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
rt_device_write
Project
main.c
RunLED.c
RunLED.h
Project: project
rtthread-stm32
Applications
RunLED.c
main.c
Drivers
STM32_HAL
Kernel
CORTEX-M3
DeviceDrivers
finsh
#include <rtthread.h>
#include "RunLED.h"
int main(void)
{
/* user app entry */
rt_thread_t tid1;
tid1 = rt_thread_create("RunLED", //线程名称。
Function_RunLED, //线程入口函数。
RT_NULL, //线程入口参数。
RunLED_STACK_SIZE, //线程栈大小。
RunLED_PRIORITY, //线程优先级。
RunLED_TIMESLICE); //时间片Tick。
if(tid1 != RT_NULL) //判断线程是否创建成功。
{
rt_thread_startup(tid1); //线程创建成功，启动线程。
}
return 0;
}
Project Books {} Func... Temp...
Build Output
linking...
Program Size: Code=62914 RO-data=7018 RW-data=712 ZI-data=7360
FromELF: creating hex file...
After Build - User command #1: fromelf --bin .\build\rtthread-stm32.axf --output rtthread.bin
".\build\rtthread-stm32.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:01:09
ST-Link Debugger

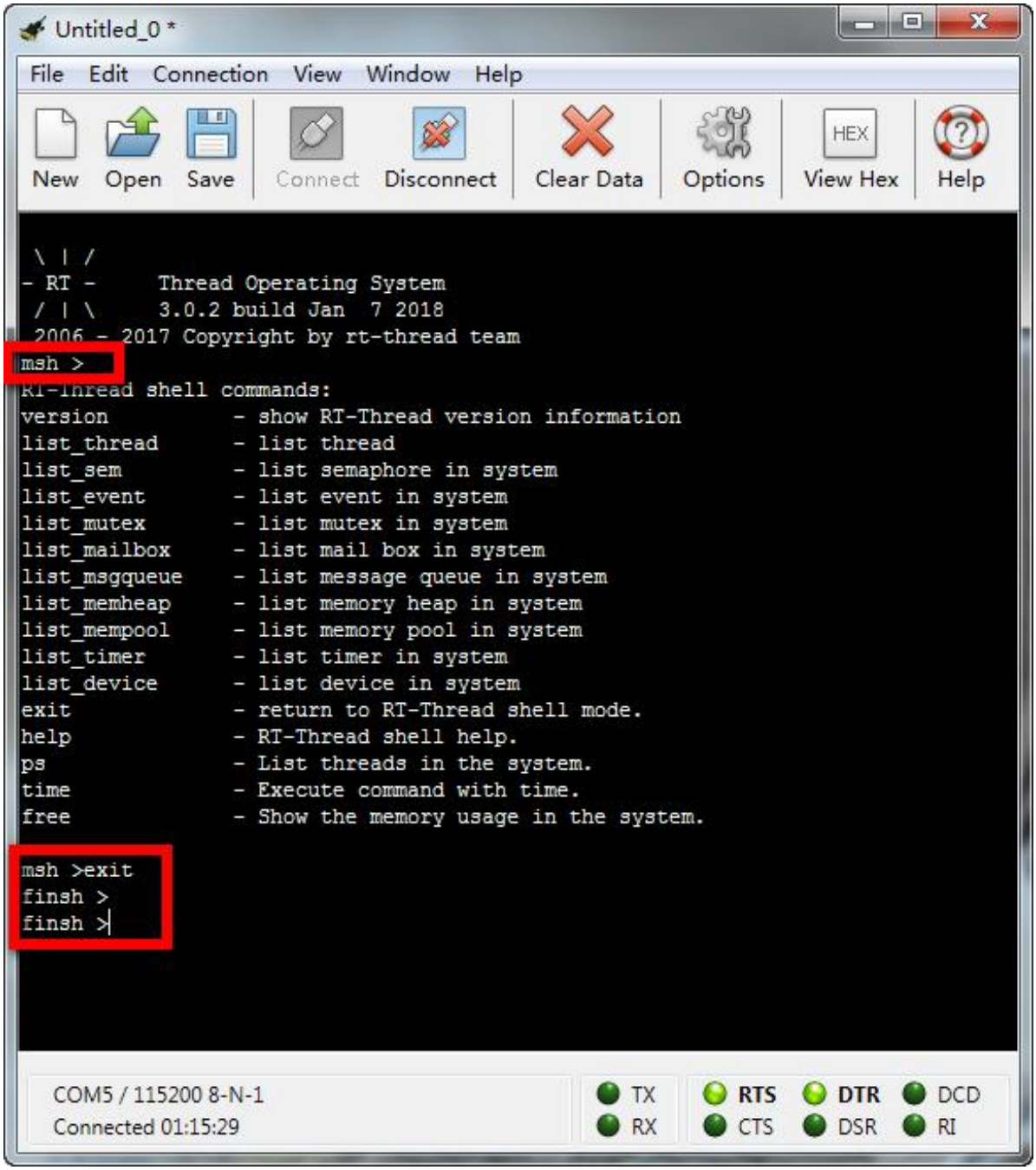
8. 使用 J-Link 或其他 Tools Download 编译成功的 Hex 文件到目标板卡。



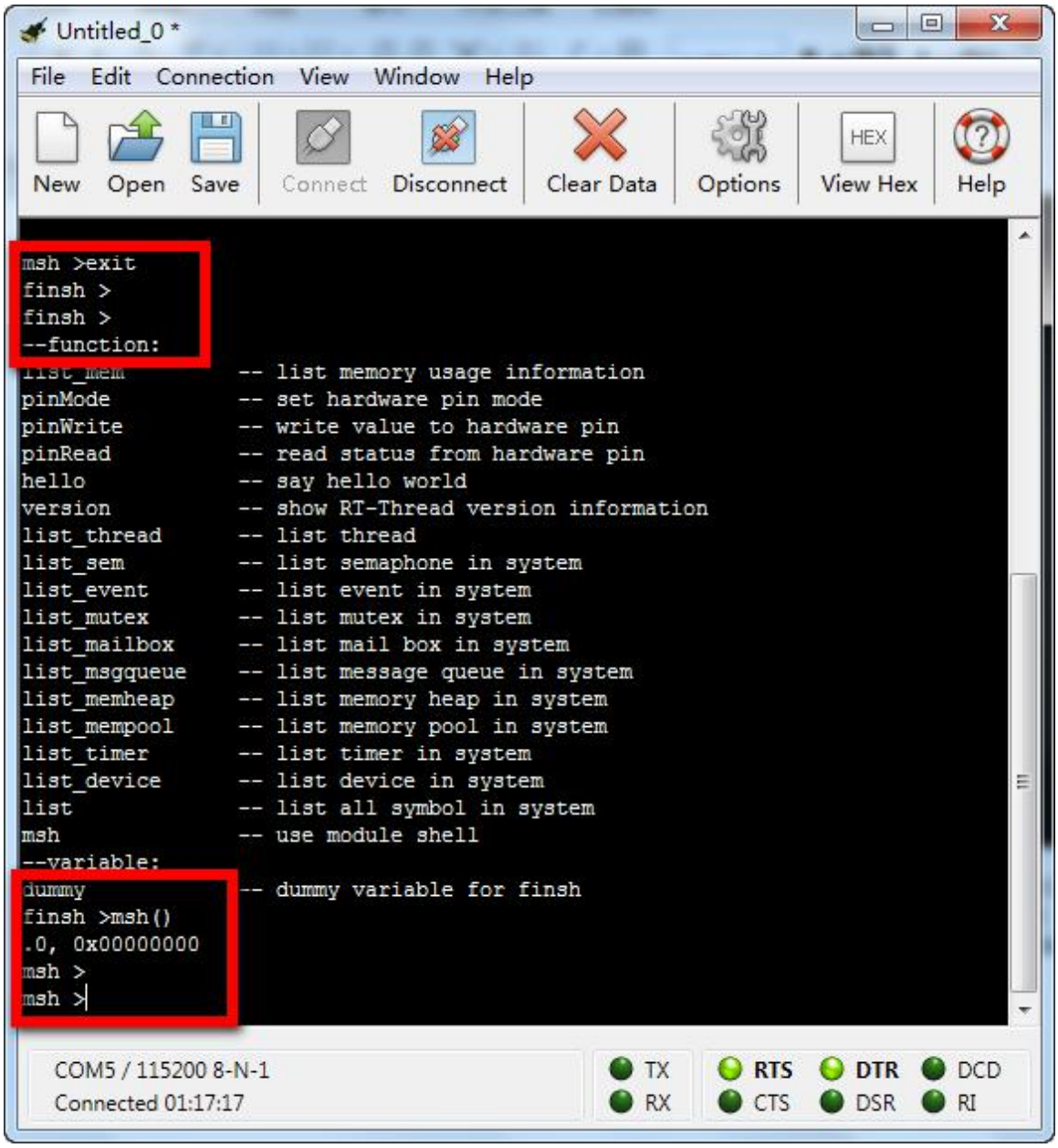
八、结果验证：

4. 打开串口调试工具，验证 finsh shell 功能是否配置成功。

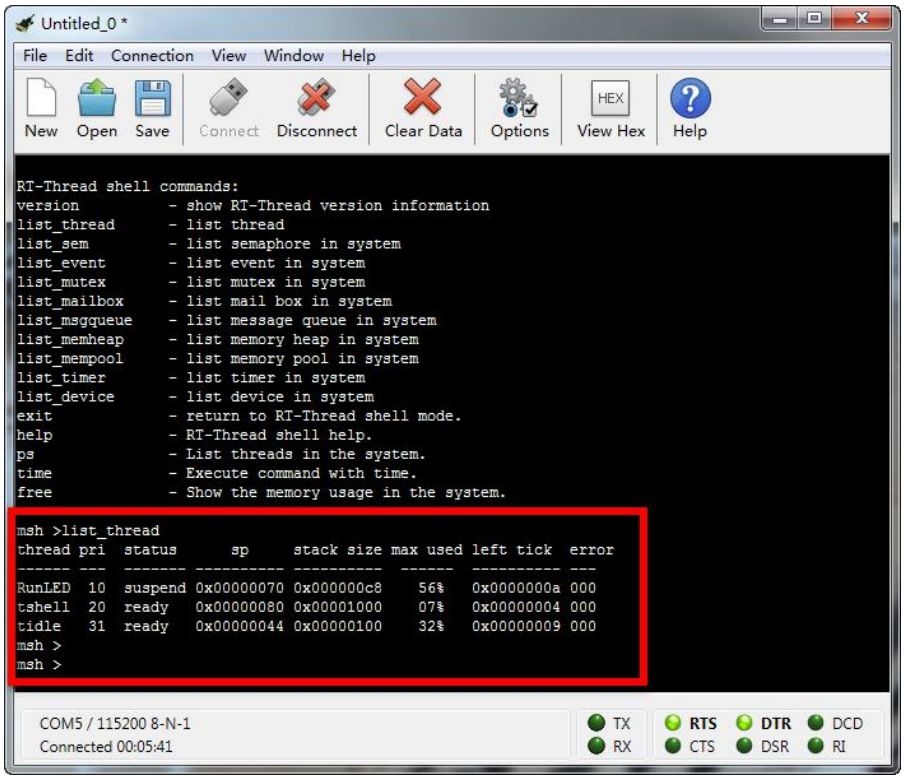
A. 通过键盘【TAB】输出 msh Commands list，通过指令【exit】退出 msh 模式，进入 c-style 模式。



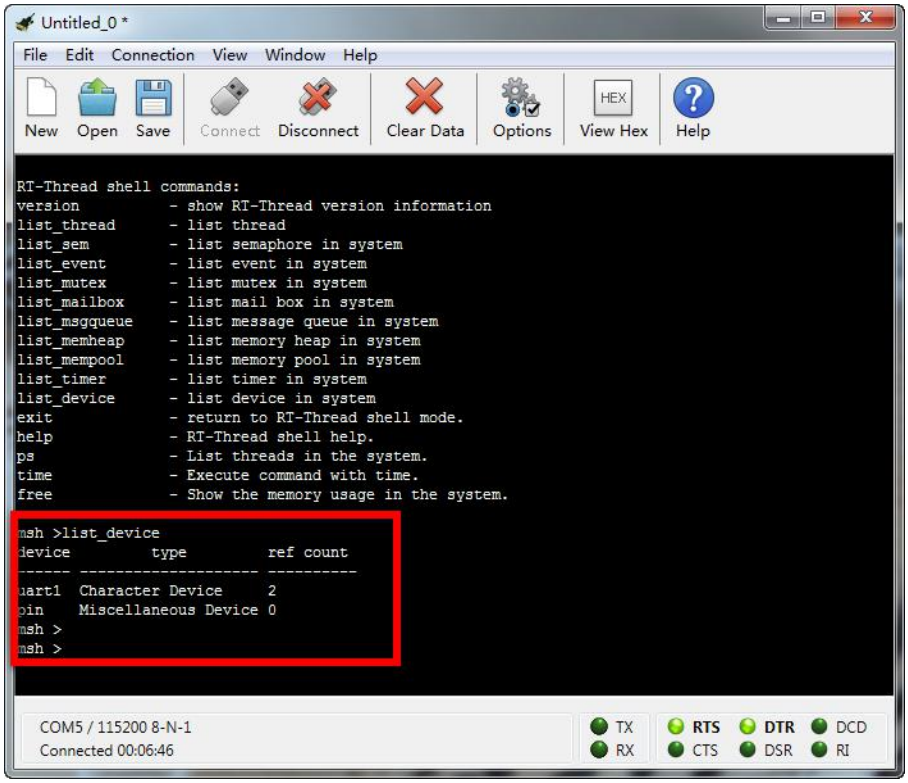
B. 通过键盘【TAB】输出 c-style Commands list，通过指令【msh()】退出 c-style 模式，进入 msh 模式。


Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
msh >exit
finsh >
finsh >
--function:
list_mem -- list memory usage information
pinMode -- set hardware pin mode
pinWrite -- write value to hardware pin
pinRead -- read status from hardware pin
hello -- say hello world
version -- show RT-Thread version information
list_thread -- list thread
list_sem -- list semaphone in system
list_event -- list event in system
list_mutex -- list mutex in system
list_mailbox -- list mail box in system
list_msgqueue -- list message queue in system
list_memheap -- list memory heap in system
list_mempool -- list memory pool in system
list_timer -- list timer in system
list_device -- list device in system
list -- list all symbol in system
msh -- use module shell
--variable:
dummy -- dummy variable for finsh
finsh >msh()
.0, 0x00000000
msh >
msh >
COM5 / 115200 8-N-1
Connected 01:17:17
TX RX RTS CTS DTR DSR DCD RI

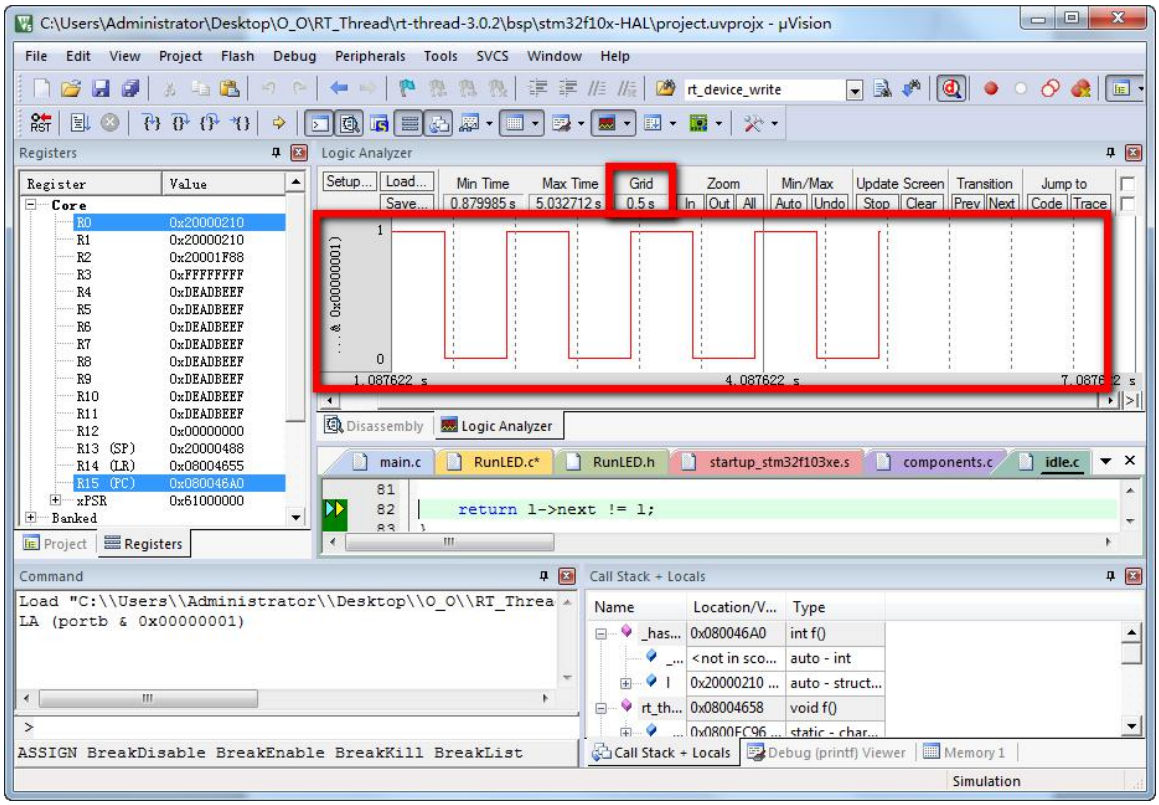
C. 查看新建线程信息：


Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
RT-Thread shell commands:
version - show RT-Thread version information
list_thread - list thread
list_sem - list semaphore in system
list_event - list event in system
list_mutex - list mutex in system
list_mailbox - list mail box in system
list_msgqueue - list message queue in system
list_memheap - list memory heap in system
list_mempool - list memory pool in system
list_timer - list timer in system
list_device - list device in system
exit - return to RT-Thread shell mode.
help - RT-Thread shell help.
ps - List threads in the system.
time - Execute command with time.
free - Show the memory usage in the system.
msh >list_thread
thread pri status sp stack size max used left tick error
RunLED 10 suspend 0x00000070 0x000000c8 56% 0x0000000a 000
tshell 20 ready 0x00000080 0x00001000 07% 0x00000004 000
tidle 31 ready 0x00000044 0x00000100 32% 0x00000009 000
msh >
msh >
COM5 / 115200 8-N-1
Connected 00:05:41
TX RX RTS CTS DTR DSR DCD RI

D. 查看设备信息：


Untitled_0 *
File Edit Connection View Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
RT-Thread shell commands:
version - show RT-Thread version information
list_thread - list thread
list_sem - list semaphore in system
list_event - list event in system
list_mutex - list mutex in system
list_mailbox - list mail box in system
list_msgqueue - list message queue in system
list_memheap - list memory heap in system
list_mempool - list memory pool in system
list_timer - list timer in system
list_device - list device in system
exit - return to RT-Thread shell mode.
help - RT-Thread shell help.
ps - List threads in the system.
time - Execute command with time.
free - Show the memory usage in the system.
msh >list_device
device type ref count
uart1 Character Device 2
pin Miscellaneous Device 0
msh >
msh >
COM5 / 115200 8-N-1
Connected 00:06:46
TX RX RTS CTS DTR DSR DCD RI

E. 查看目标板卡 LED 等闪烁状态：（由于 LED 动态闪烁用图片无法证明，此处仅通过软件仿真演示结果。）



后记：

关于 RT-Thread 线程及设备管理的具体使用方法及技巧，可查阅 RT-Thread 官方提供的编程手册 或 搜索 RT-Thread 学习笔记的相关章节。