**关于RT-Thread BSP制作的知识总结**

说明：1. BSP制作硬件平台为STM32F405RGT6;

2.在git中下载使用RT-Thread的3.15版本，内部已经有1个STM32F405的BSP，可以作为制作参考；

3.制作务必参考官方提供的《STM32系列BSP制作教程》系列文档，路径如下：

rt-thread\bsp\stm32\docs。本文档目录架构按照《STM32系列BSP制作教程》操作步骤进行。

4.文中涉及menuconfig的界面操作时，注意界面左上角处的路径。

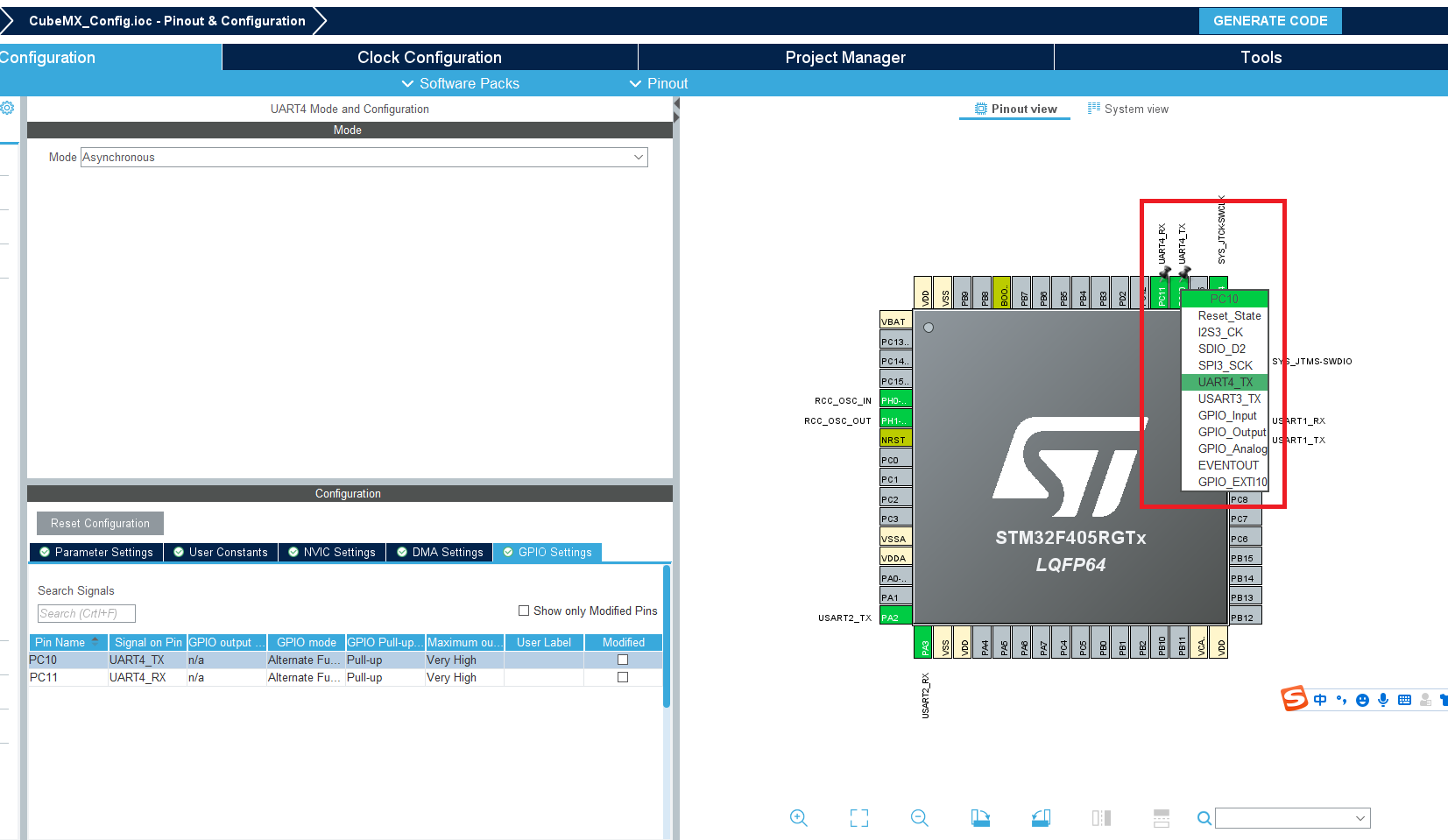
## 一、STM32CubeMx使用

STM32CubeMx主要是用来帮助我们生成一些外设的驱动。stm32 系列的BSP**只利用**了 CubeMX 工具生成的`stm32XXxx\_hal\_conf.h`和`stm32XXxx\_hal\_msp.c` 文件。在HAL库中， `stm32XXxx\_hal\_conf.h` 文件里提供的宏开关会决定 HAL 库将哪些外设驱动添加到工程中。而`stm32XXxx\_hal\_msp.c` 文件中则存放了在 CubeMX 工具中开启的外设驱动的配置代码。

#### 1.1 外设设置

在STM32CubeMx中配置的外设基本参数，这些参数有些会体现在stm32f4xx\_hal\_msp.c源文件中，而stm32f4xx\_hal\_msp.c文件是我们工程中使用的，因此我们需要根据自己的实际情况如实配置涉及的参数。我们必须对驱动熟悉才能正确检查并配置外设参数。

实际上，我们在CubeMX中也可以修改引脚。如果默认的的引脚并不是我们实际使用的，那么我们直接在芯片图中找到实际引脚，然后右击点击**复用即可**。

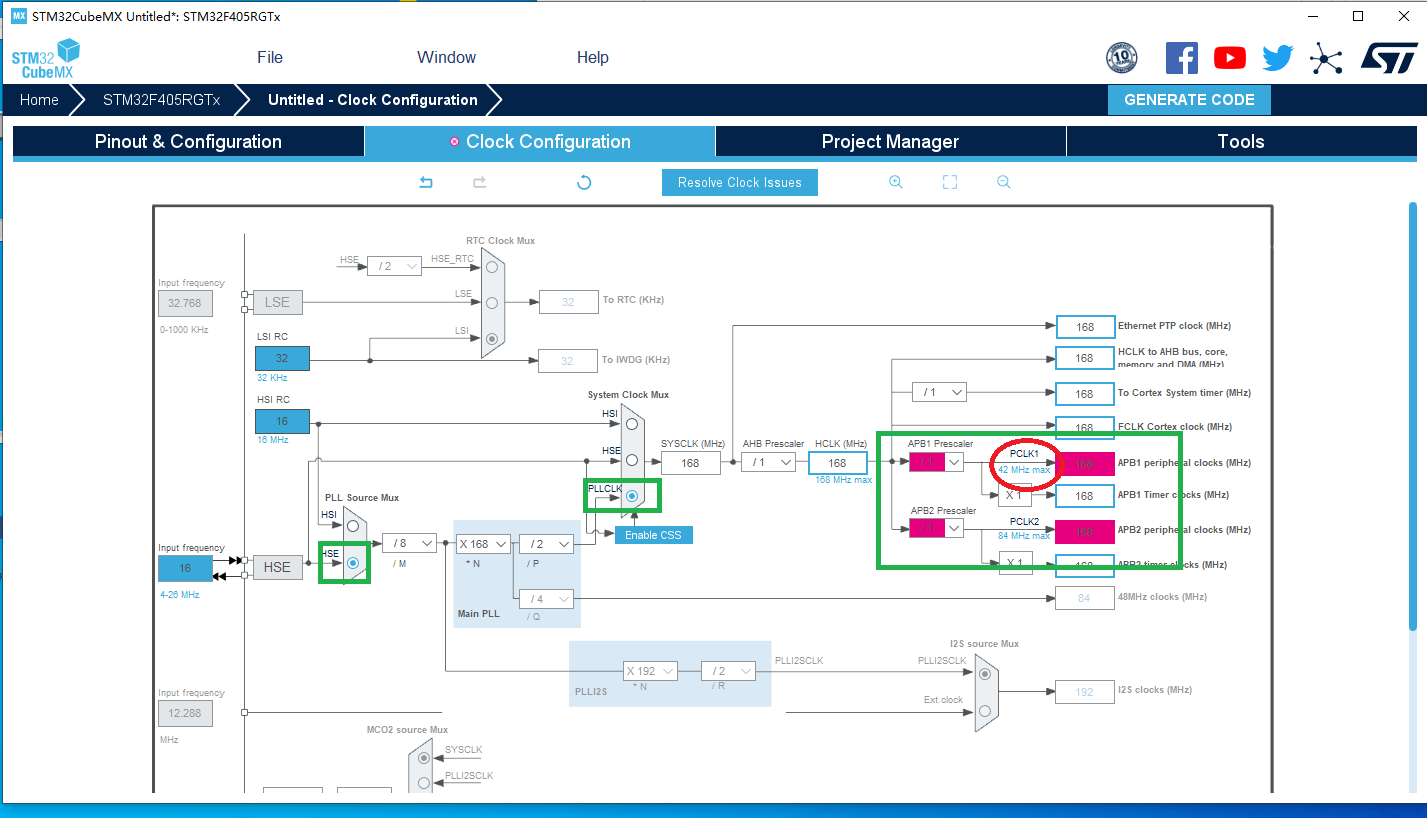


外设一般在源文件Src文件夹下的stm32f4xx\_hal\_msp.c（以f4为例）中；Inc文件夹下的stm32f4xx\_hal\_conf.h（以f4为例）文件包含STM32CubeMx时钟设置时的外部晶振及内部晶振大小，它其实是由STM32CubeMx生成的。

#### 1.2 时钟配置

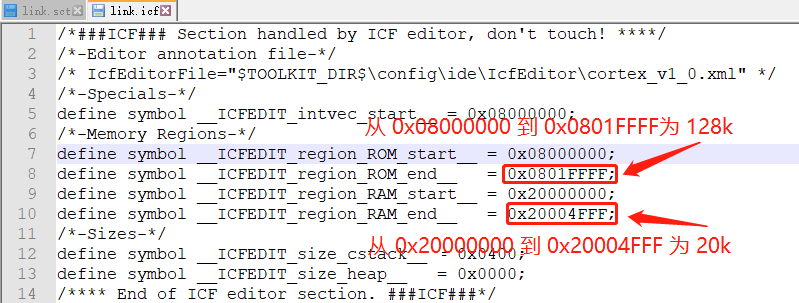
我们一般使用的是外部晶振，并通过PLLCLK来得到系统时钟，因此需要选择HSE和PLLCLK。选择按钮比较小，容易被忽略。

当我们配置的参数不合理时（比如超过了某个外设时钟最大值），CubeMx会以红色醒目的方式提醒我们有错误，并且在编译时候提出警告，因此，我们要即使修改自己的错误。



## 二、配置FLASH和RAM相关参数

board.h用来配置FLASH和RAM的相关参数，该文件配置的参数需要与文件夹linker\_scripts下的链接文件相对应。linker\_scripts文件下总共3个链接文件，实际我们只要修改自己**对应的编译器的链接文件**即可，比如 MDK 使用的链接脚本 为link.sct。注意iar 使用的 link.icf文件里面实质为FLASH和RAM**的地址，不是大小。**



对于一般的SRAM（连续存储），严格按照RTT提供的教程《STM32系列BSP制作教程》修改即可。

对于SRAM的内存不连续存储的（比如：srm32f405RGT6，SRAM 为192KB（128+64），128K(0x2000 0000 - 0x2001 FFFF)， 64K (0x1000 0000 - 0x1000 FFFF），还是设置为主内存IRAM1的大小，以如下代码的从高地址向低地址的分配空间的方式为heap开辟空间。当然对应的**链接文件**和该处设置一致。而 IRAM2留给系统内存去使用。注意在**keil软件里面勾选IRAM2**.

/\*

Note：The total size of SRAM is 192, but doesn't continue.

128K(0x2000 0000 - 0x2001 FFFF)

64K (0x1000 0000 - 0x1000 FFFF)

on the one hand Make sure you tick the **IRAM2 in Keil tools**

on the other hand we write IRAM1 size at here but not actual SRAM size.

if you input 192 the program will fly.

\*/

#define STM32\_SRAM1\_SIZE (128)

#define STM32\_SRAM1\_END (0x20000000 + STM32\_SRAM1\_SIZE \* 1024)

#if defined(\_\_CC\_ARM) || defined(\_\_CLANG\_ARM)

extern int Image$$RW\_IRAM1$$ZI$$Limit;

#define HEAP\_BEGIN (&Image$$RW\_IRAM1$$ZI$$Limit)

#elif \_\_ICCARM\_\_

#pragma section="CSTACK"

#define HEAP\_BEGIN (\_\_segment\_end("CSTACK"))

#else

extern int \_\_bss\_end;

#define HEAP\_BEGIN (&\_\_bss\_end)

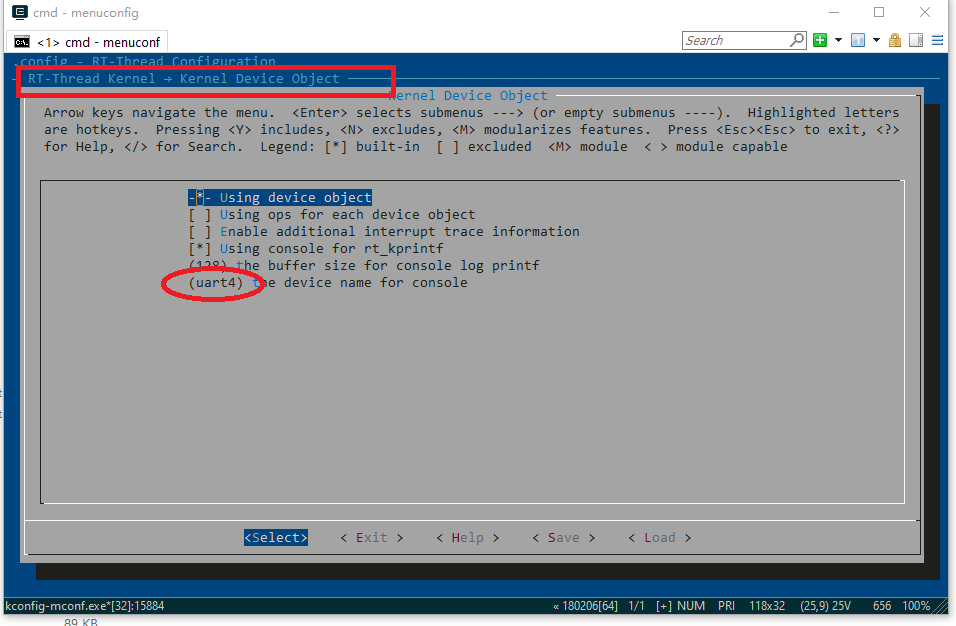
#endif

#define HEAP\_END STM32\_SRAM\_END

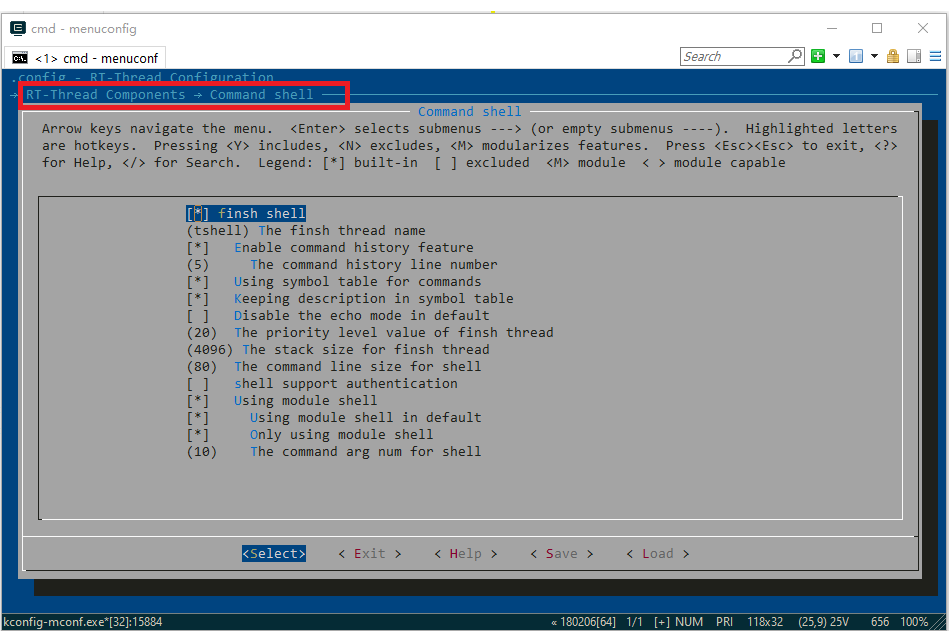
## 三、menuconfig设置

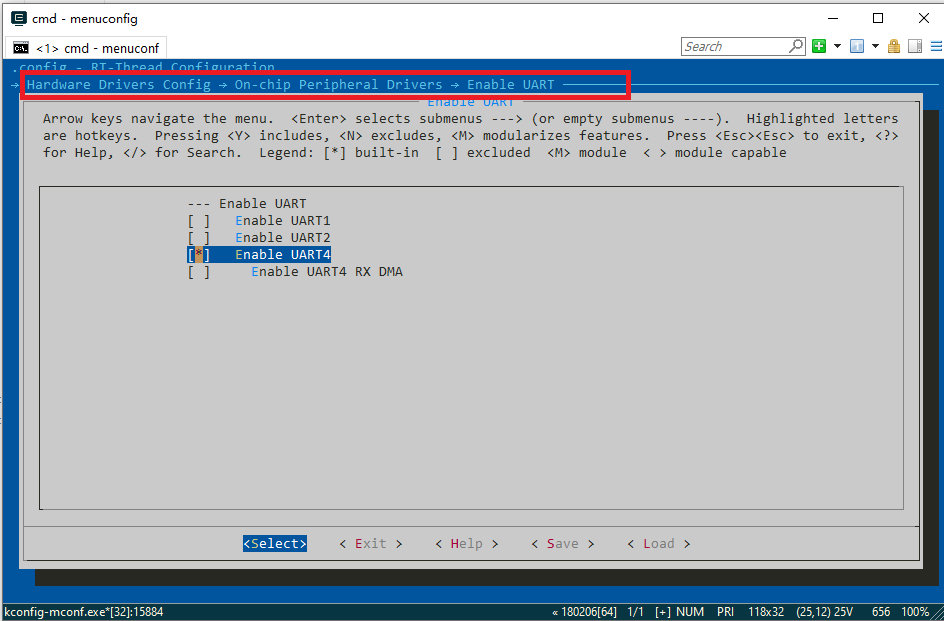
#### 3.1 开启Finsh组件

Finsh组件的开启需要**内核设备对象**、**组件shell**、**匹配串口**同时开启才能正常运行，注意内核设备对象的串口号要与开启的**串口一致**。



使用串口4



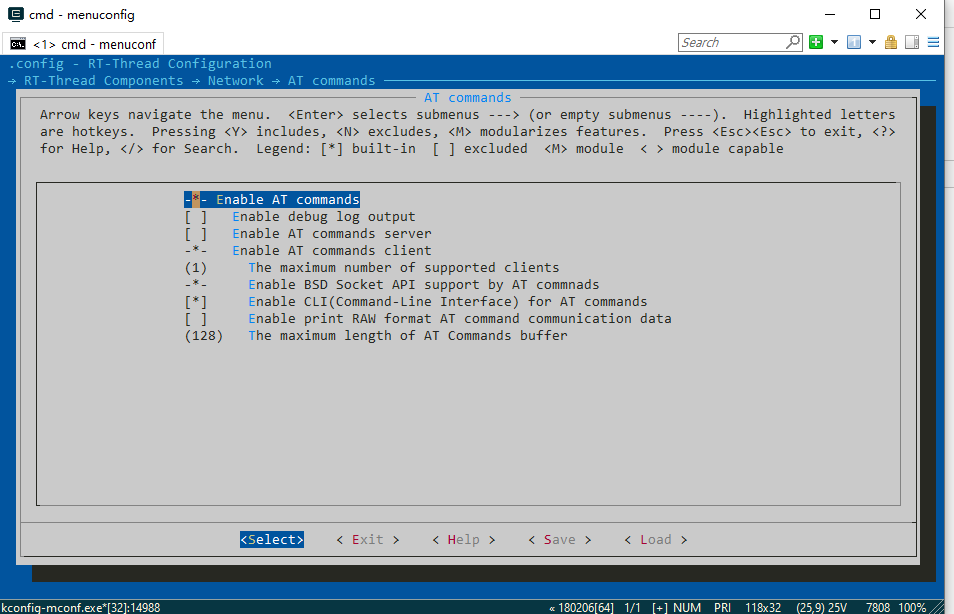


开启串口4

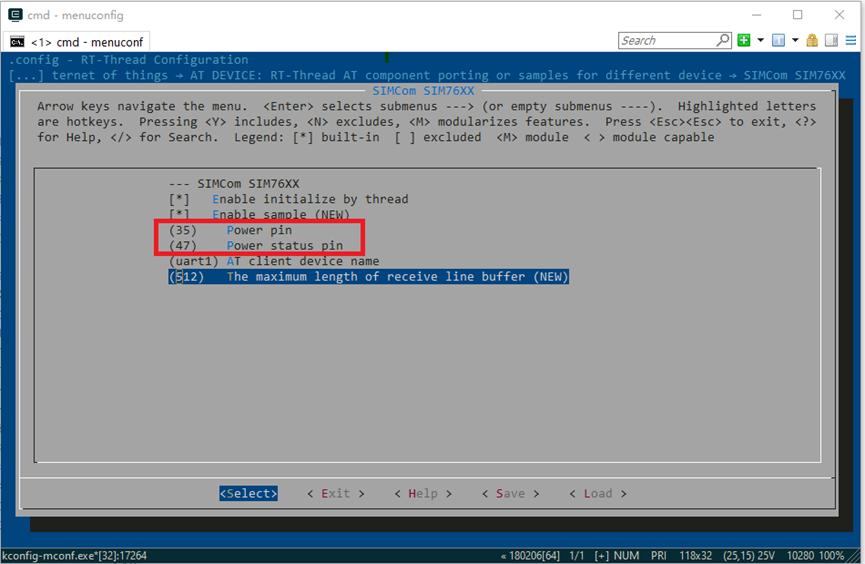
#### 3.2 SIM7600驱动包

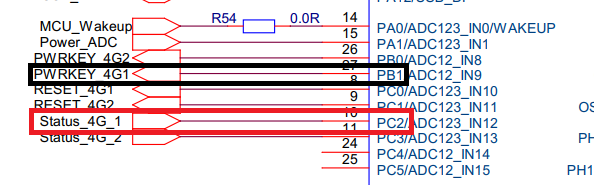
SIM7600是4G模块，它通过串口和MCU进行连接，通信形式是AT指令。

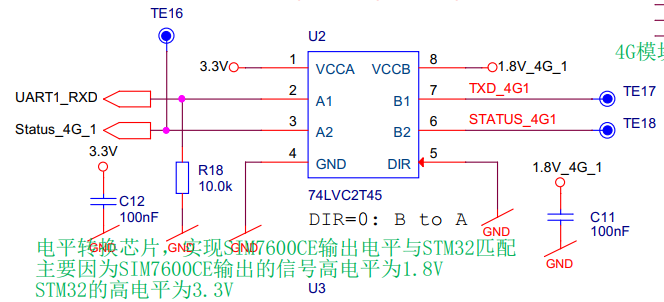
因此，在下载SIM7600软件包之前，还需要在组件中对AT指令进行配置，参考如下：



SIM7600驱动下载之前，需要根据**板子原理图**配置Power引脚、status引脚和通讯串口编号。







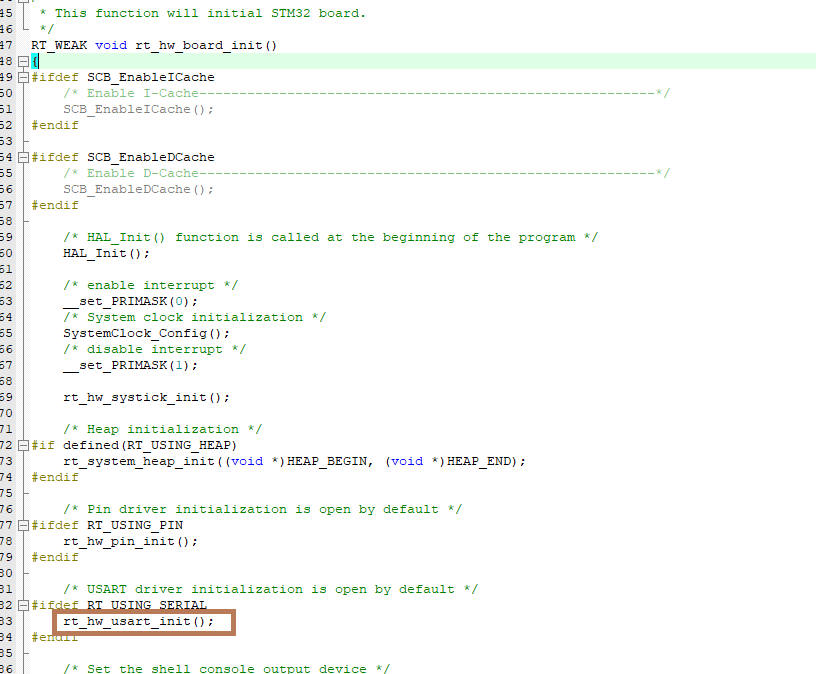
此处的power引脚和status引脚是指MCU端的引脚配置，但是并不是直接引脚编号，而是由RTT操作系**统映射之后的pin编号**，这种映射有点类似Linux的，可以参看文献RTT关于PIN设备描述。映射关系如下：

PA0 -> 0 PA15 -> 15

PB0 ->16 PB15 -> 31 PB1 -> 17(Power引脚)

PC0 -> 32 PC15 -> 47 PC2 -> 34(status引脚)

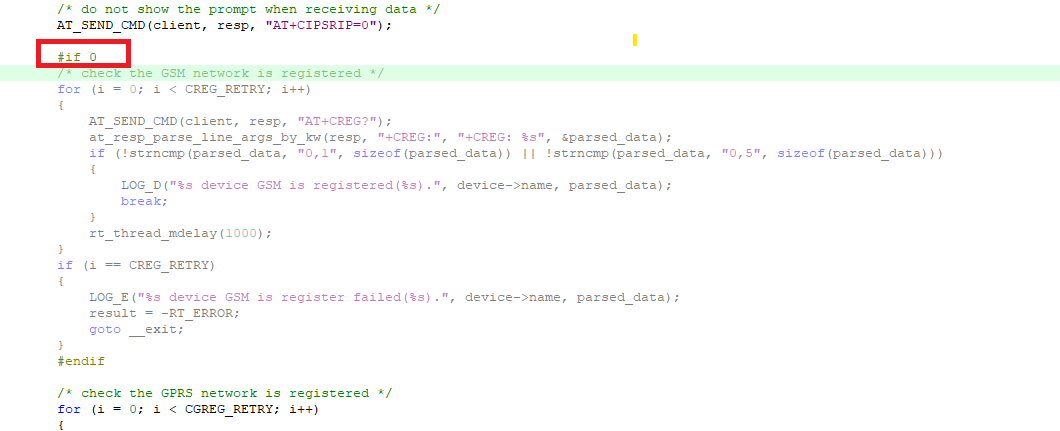
通讯串口需要和实际使用的串口保持一直，关键注意串口通讯的参数比如**波特率**等。事实上，STM32CubeMX生成的串口通讯参数，BSP中实际没有并使用，因此我们尝试修改STM32CubeMX中串口参数是没有作用的。那么修改串口正确的位置其实是在RTT操作系统**硬件初始化的函数**中，由于初始化函数里面默认是将所有的串口参数都配置成一样的，这样实际使用时需要根据不同的串口对参数进行配置。



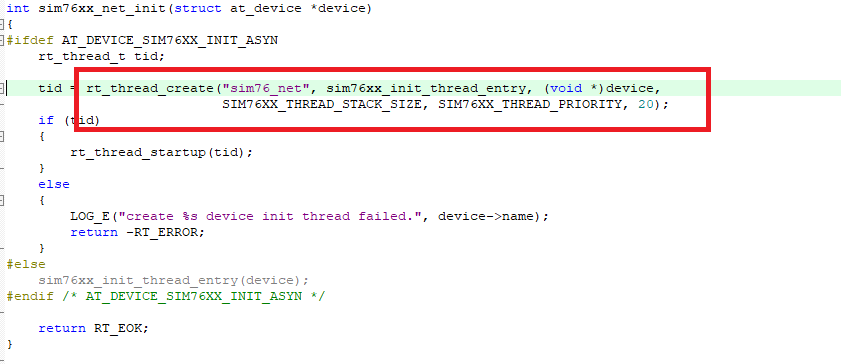


SIM7600串口通讯参数可以通过AT指令进行设置，具体参照《SIM7500\_SIM7600 Series\_AT Command Manual \_V1.12》的3.12-3.14小节。

由于笔者的板子GSM没有注册网络，因此在实际调试过程中，在SIM7600的源码中暂时**屏蔽掉了GSM**。



加载SIM7600驱动包后，软件自动增加线程，我们**无须再建线程**。



Q2: SIM7600如何启动？

## 四、外设驱动添加

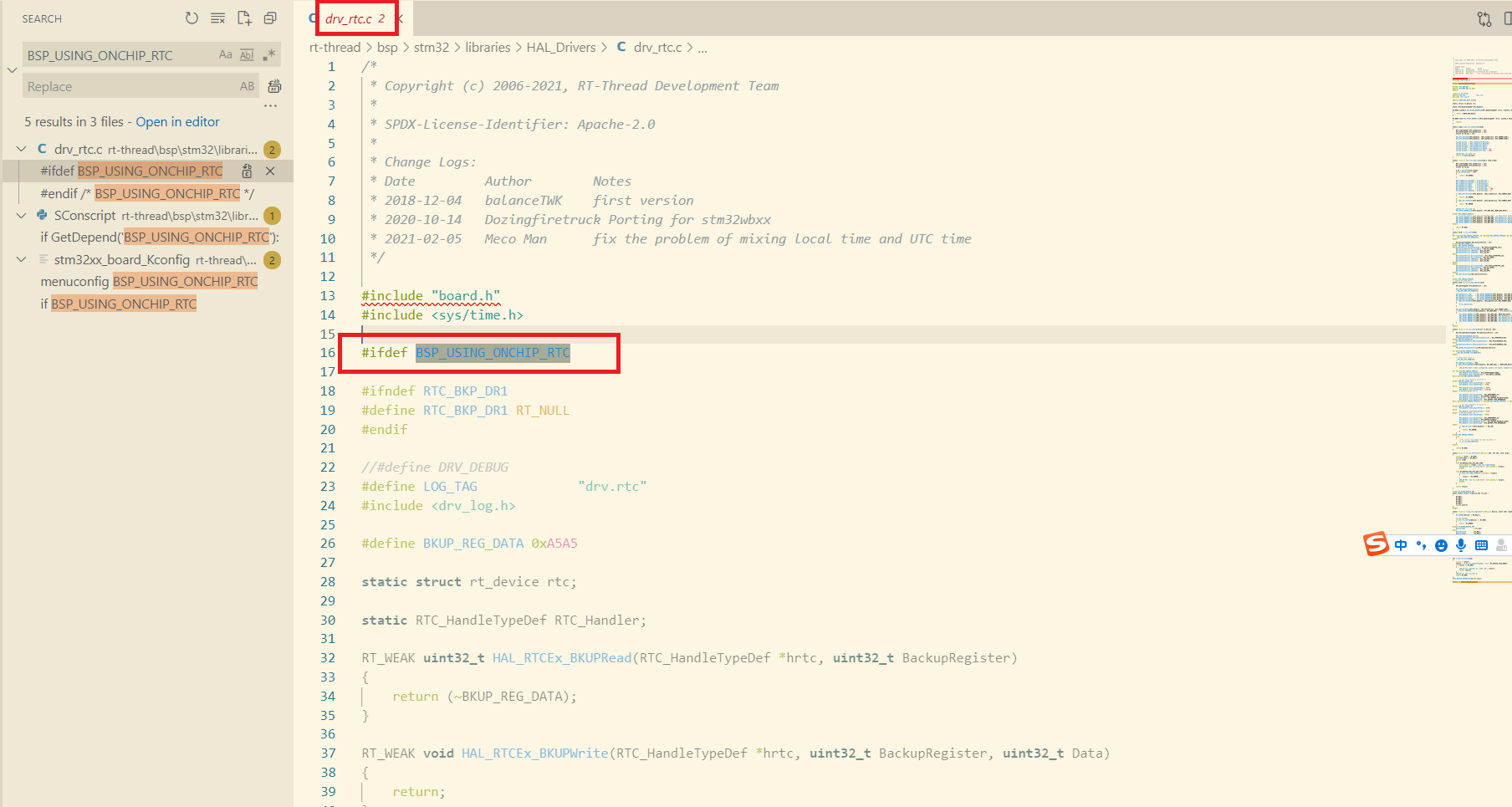
#### 4.1 外设驱动添加步骤

1）STM32CubeMX进行相关外设配置

2）修改 Kconfig 文件

强烈建议参考rt-thread\bsp\stm32\libraries\templates目录下的stm32xx\_board\_Kconfig文件，这里面基本的外设存在。我们在编写涉及的外设Kconfig时，如果对外设的宏定义名称及外设所包含的文件没有足够的把握，建议**直接复制里面的源码**。

Kconfig文件中外设涉及的宏定义的开启，最后结果在rtconfig.h中定义。这个宏定义其实直接决定了RTT编写的外设驱动是否被加入工程中。



因此，宏定义的名称**不能随意命名，必须在准备添加外设的源文件里面寻找。**这些外设源文件目录为：\rt-thread\bsp\stm32\libraries\HAL\_Drivers

当然，对于某些外设而言，依赖的文件比较多。仅仅开启/定义一个宏定义不够，此时就比较麻烦。这也就是建议直接复制stm32xx\_board\_Kconfig文件的原因。

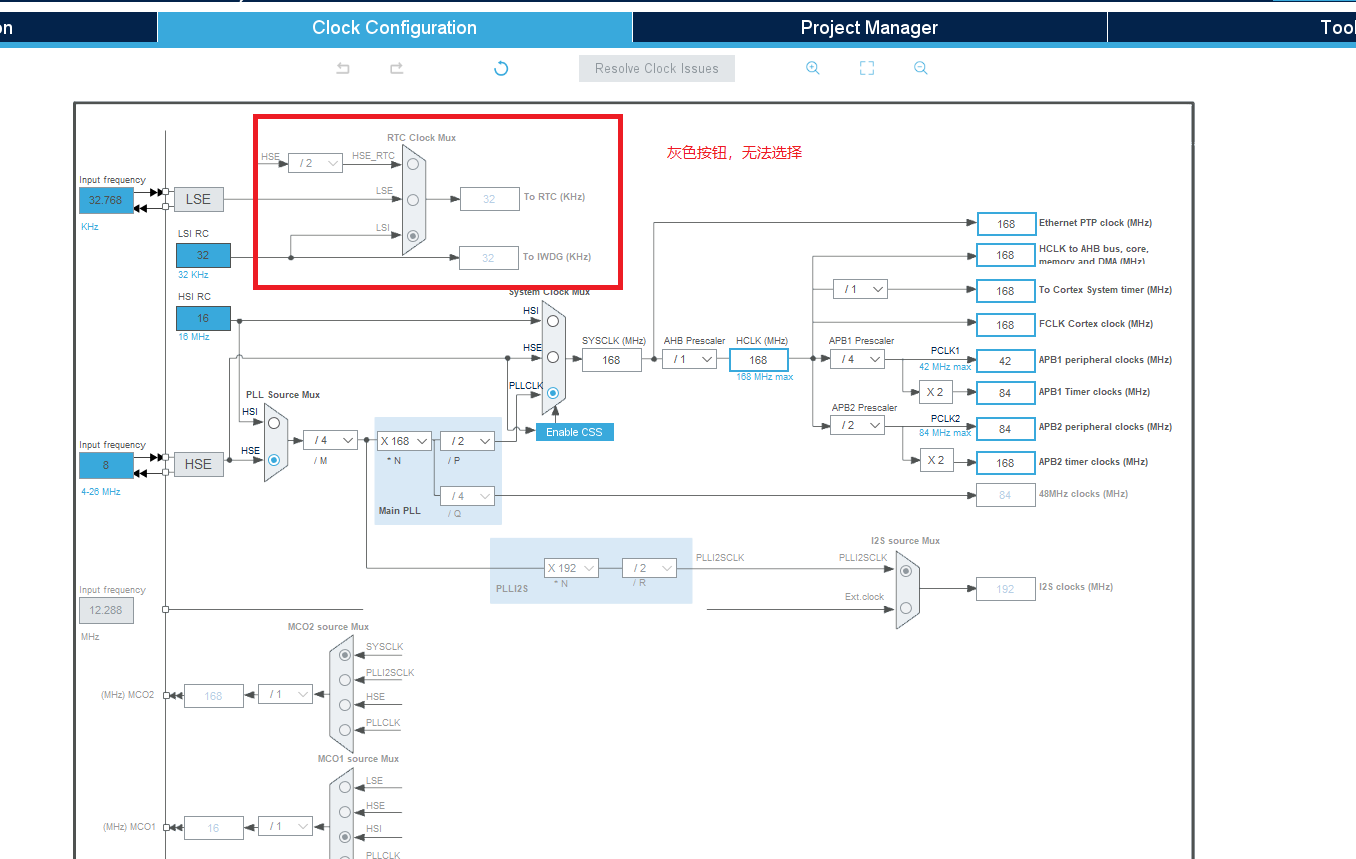
3）编译修改和调试

主要对MDK编译提示的错误修改即可，一般都是手动添加必要函数或者变量定义。

#### 4.2 RTC外设

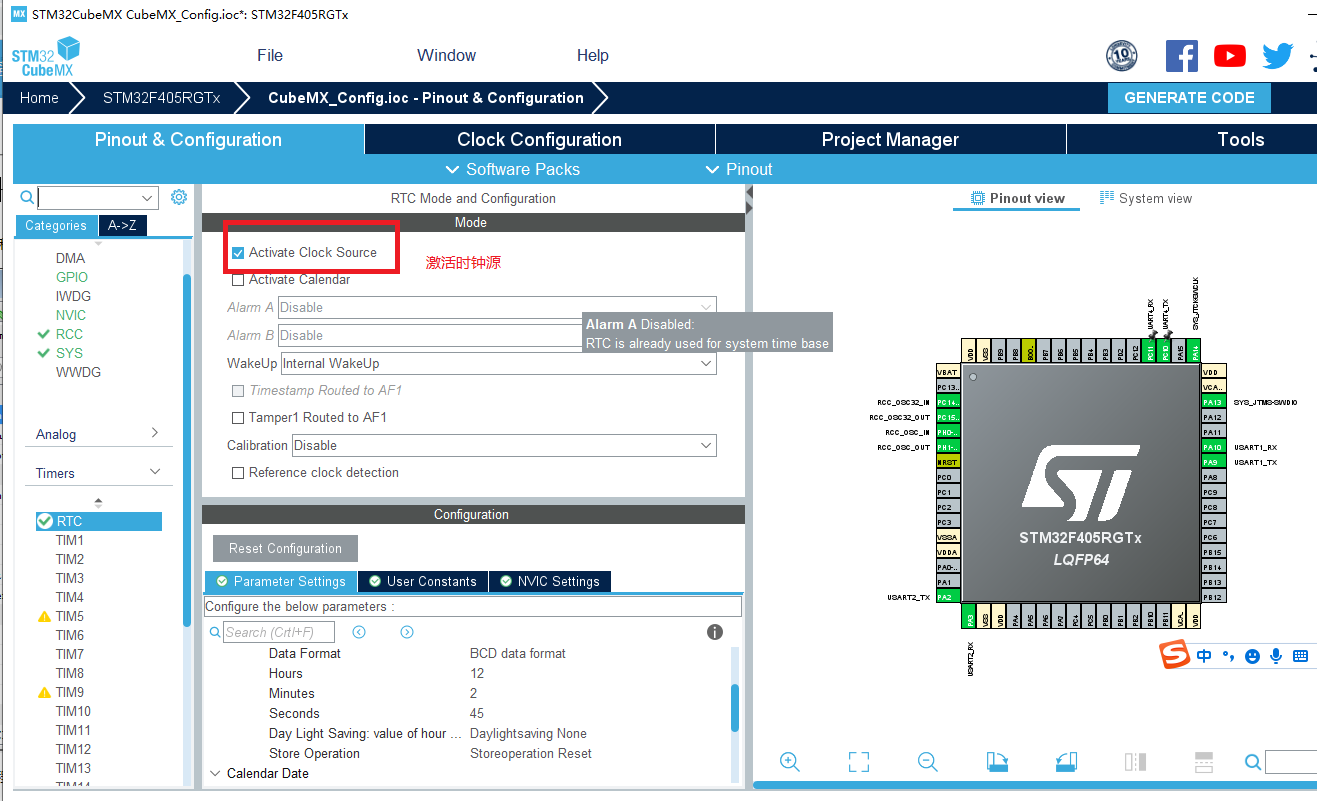
本小节详细说明RTC外设添加及Kconfig文件中宏定义的作用，其他外设的添加可以参考理解。

RTC外设是**单独时钟源**，一般选择**32.768KHz**的外部时钟，因此我们需要在RCC中开启LSE时钟。RTC时钟源开启后，**不要立马在Clock Configure界面去配置RTC时钟**，需要首先在Timers/RTC中激活时钟源，否则，Clock Configure界面无法配置RTC时钟。



不能配置LSE

根据参考文献中的《STM32CubeMX系列教程13:实时时钟(RTC)》配置RTC的参数后，最后再去配置时钟源。



RTC的Kconfig文件中的编码如下:

menuconfig BSP\_USING\_ONCHIP\_RTC

bool "Enable RTC"

select RT\_USING\_RTC

select RT\_USING\_LIBC

default n

if BSP\_USING\_ONCHIP\_RTC

choice

prompt "Select clock source"

default BSP\_RTC\_USING\_LSE

config BSP\_RTC\_USING\_LSE

bool "RTC USING LSE"

config BSP\_RTC\_USING\_LSI

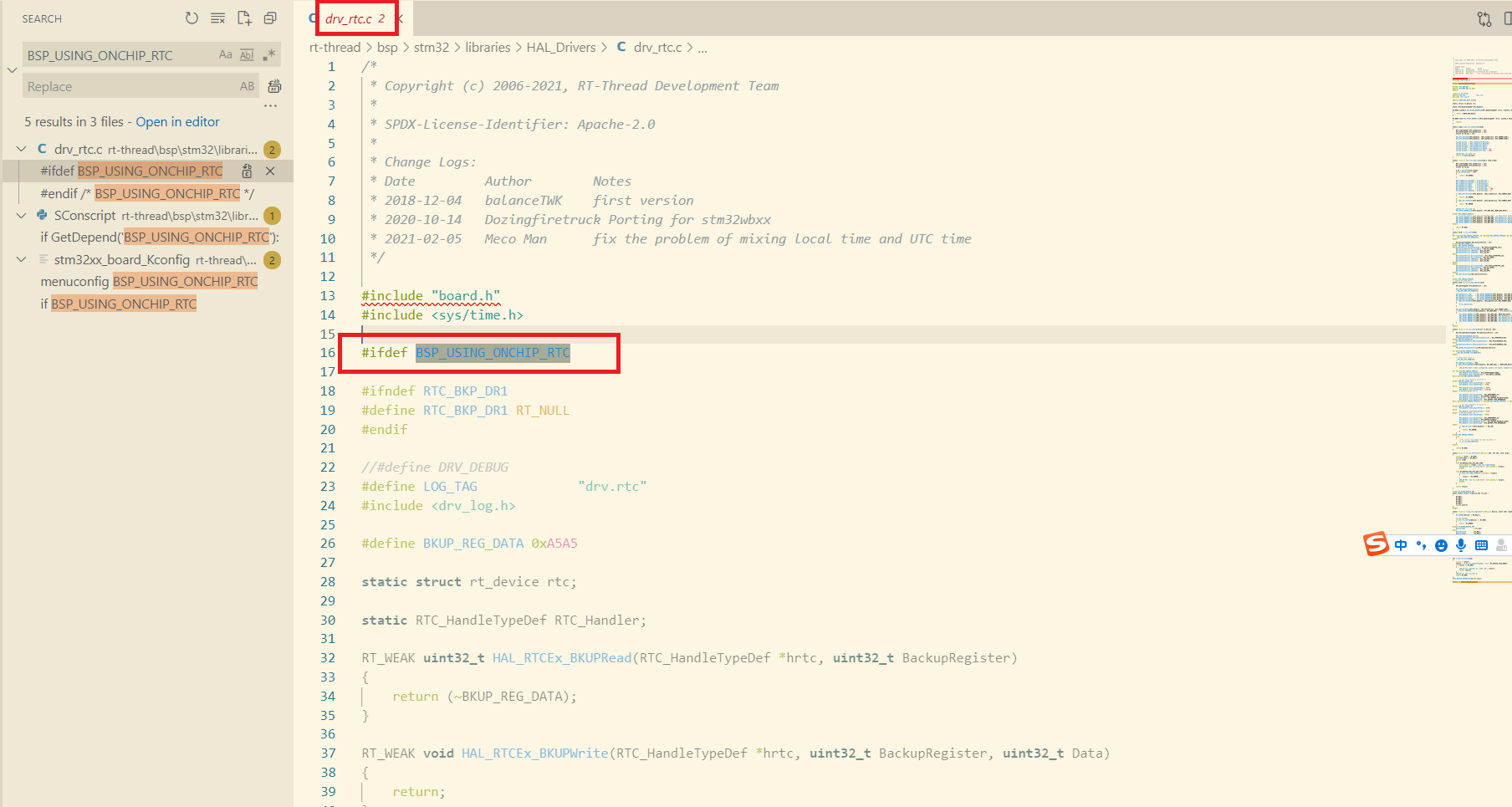
bool "RTC USING LSI"

endchoice

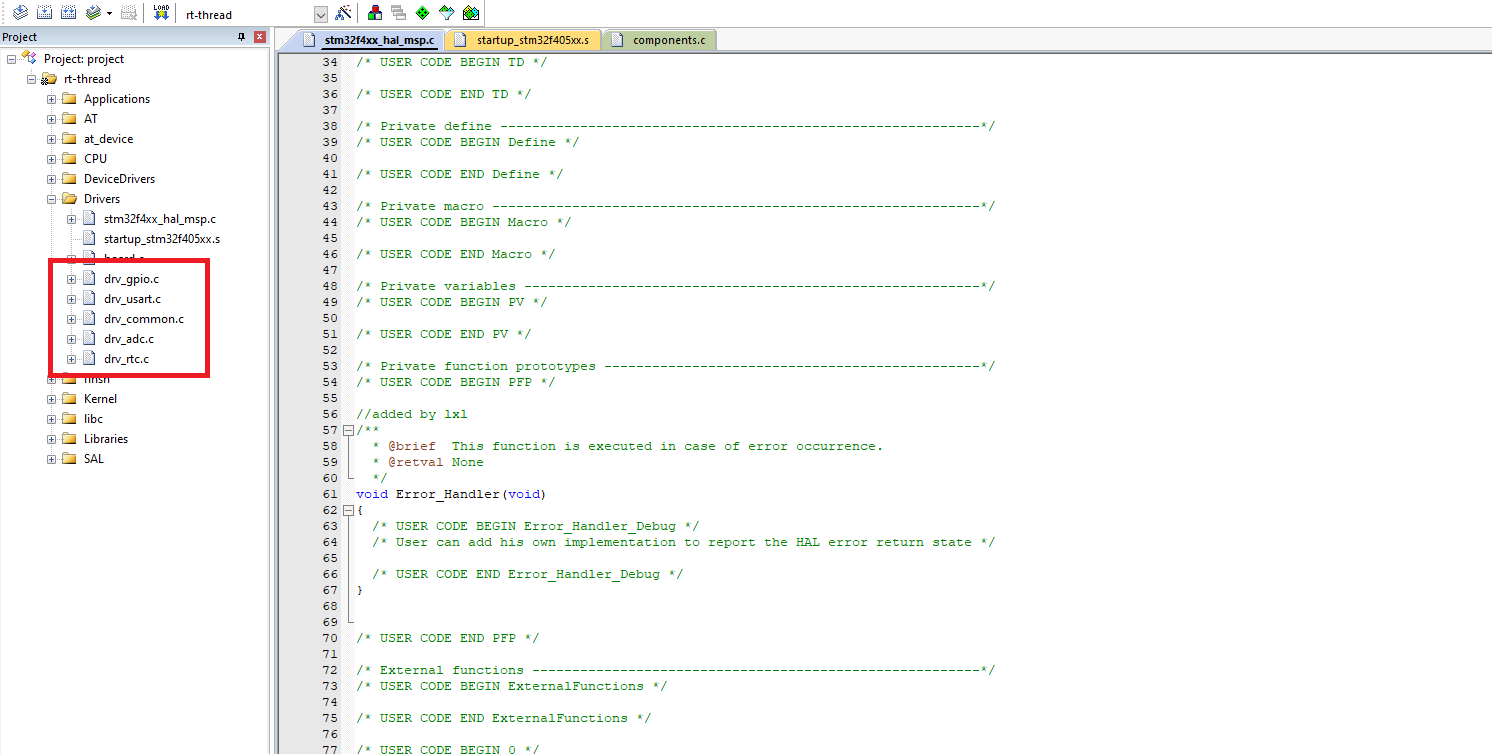
endif

关于其中的宏定义描述如下：

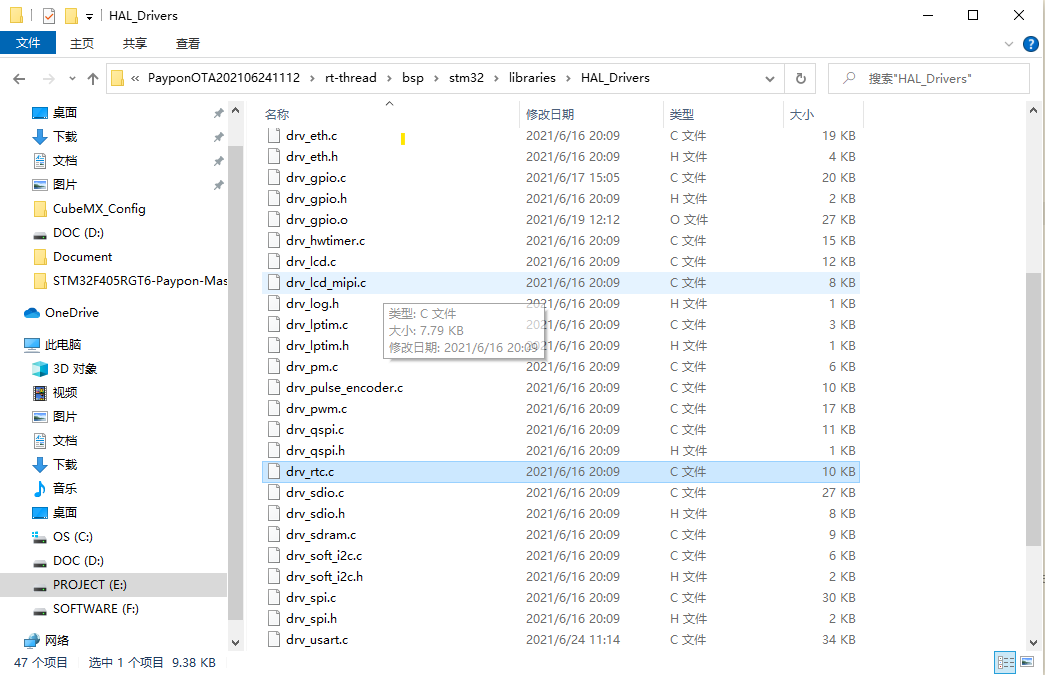
BSP\_USING\_ONCHIP\_RTC：该宏定义将把**RTT的设备底层驱动drv\_rtc.c加载**进工程中。我们在MDK编译时首先需要查看该外设驱动有没有加进工程中。



宏定义BSP\_USING\_ONCHIP\_RTC命名来源

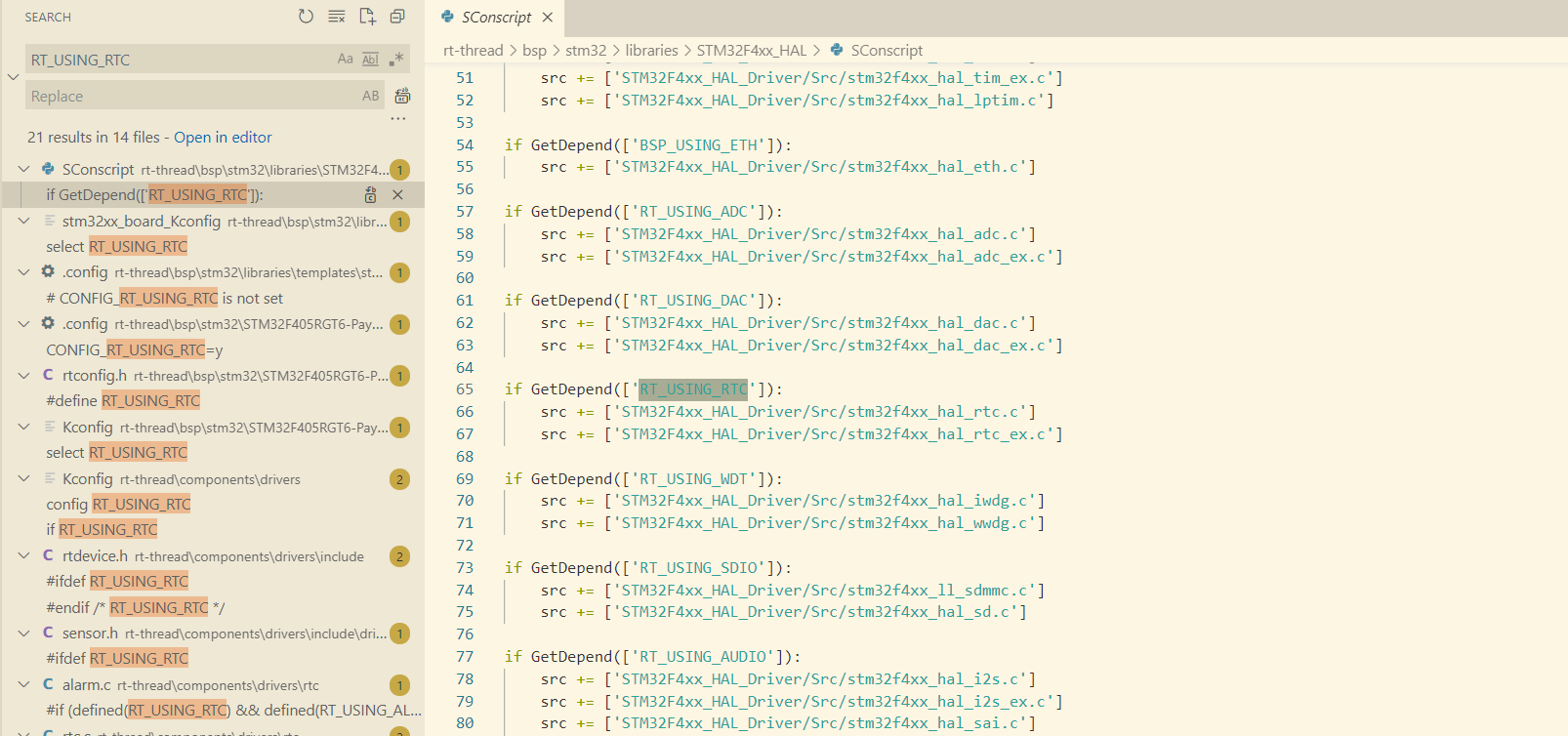


工程中已加载外设底层驱动



其他外设底层驱动

RT\_USING\_RTC：这个主要是将STM32官方提供的固件库加载进咱们的工程。STM32F4xx\_HAL\_Driver/Src/stm32f4xx\_hal\_rtc.c

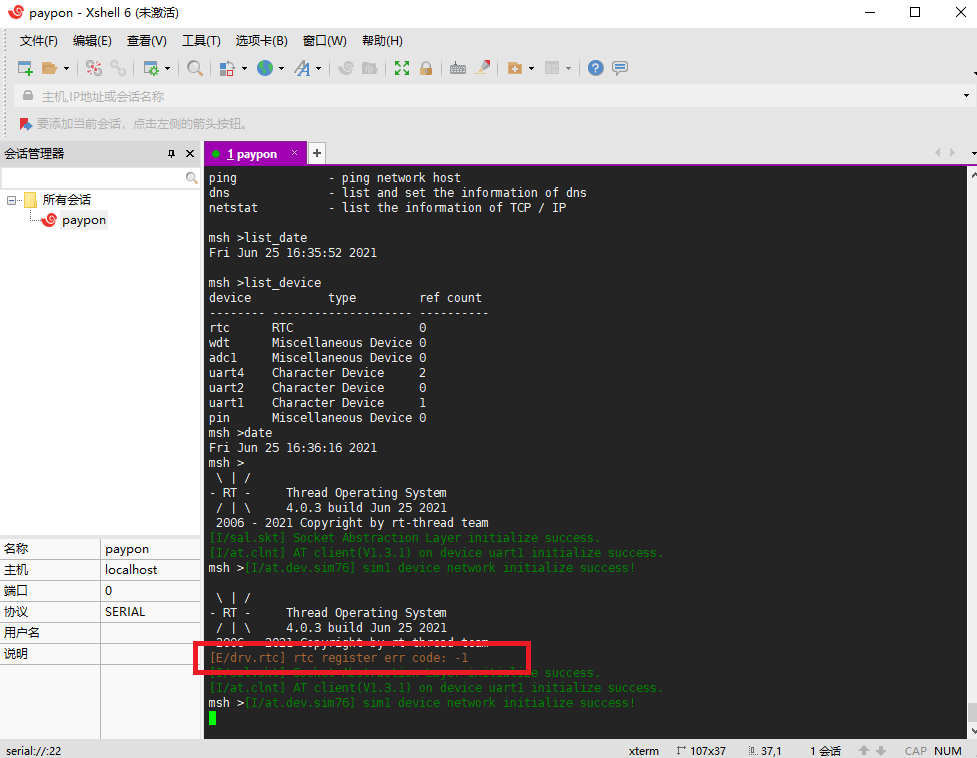


RT\_USING\_LIBC：这个主要是因为RTT系统使用RTC时使用了C标准库中的API。



【问题提出】

在调试RTC时，提示RTC注册失败，但奇怪的是RTC仍旧在正常使用。



【原因定位】

通过跟踪代码，我们发现程序在函数rt\_device\_register中的rt\_device\_find处出现错误。既在设备列表中发现了“rtc”，因此不能重复注册。



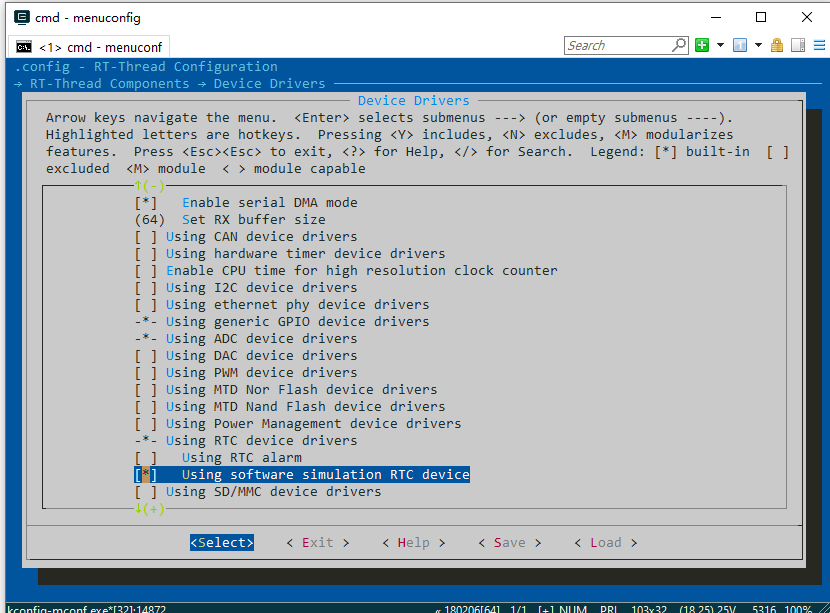
那么，是什么导致“rtc”被注册了。通过在程序中添加日志，我们发现rt\_device\_register被调用了2次。一次是被源文件soft\_rtc.c调用。



soft\_rtc.c提前注册了”rtc”，而soft\_rtc.c是由宏RT\_USING\_SOFT\_RTC引入的。

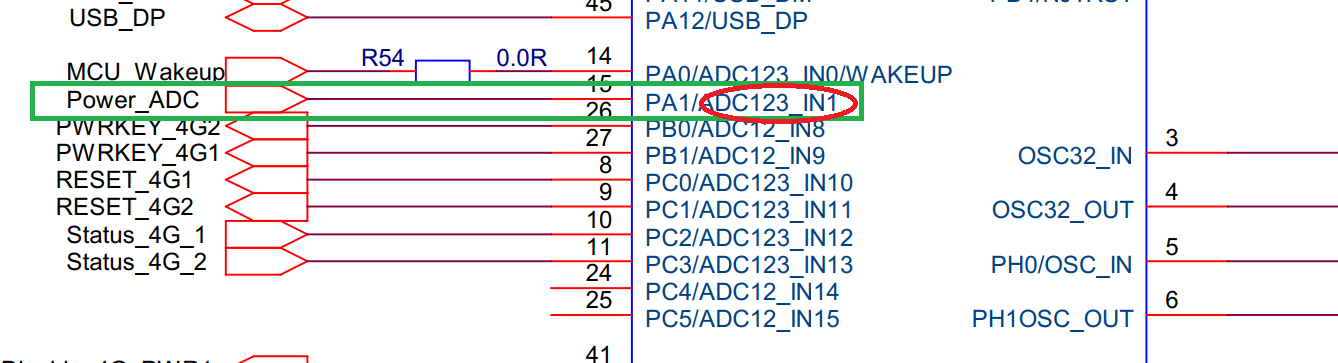
【问题解决】

关闭掉软件模拟RTC功能即可。

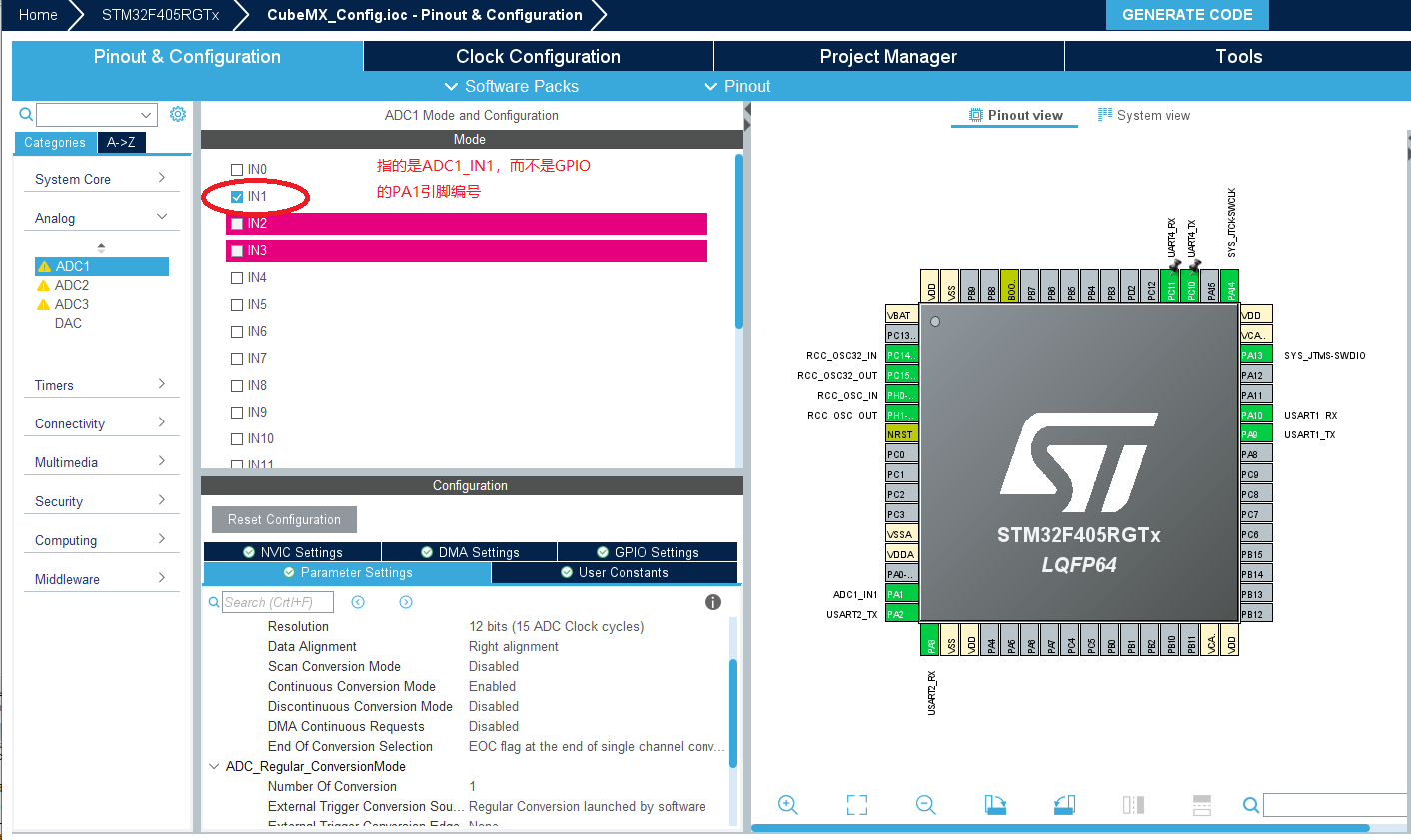


#### 4.3 ADC1外设

根据原理图，确认ADC1的实际使用的输入引脚为PA1->**ADC1\_IN1**。



ADC1是挂接在**APB2(高速外设)**上的设备，需要根据自己的实际情况如实配置涉及的参数即可。



Kconfig文件直接复制stm32xx\_board\_Kconfig里面的即可。

生成MDK后编译可能有错误提示：stm32f4xx\_hal\_msp.c文件中**hdma\_adc1 和 Error\_Handler未定义**，参考注意事项。

#### 4.4 IWDG外设

RT-Thread提供的看门狗外设为**独立看门狗**，暂不支持窗口看门狗。



测试IWDG的效果可以**直接复制**RT-Thread提供的Demo，不需要在main函数中调用。



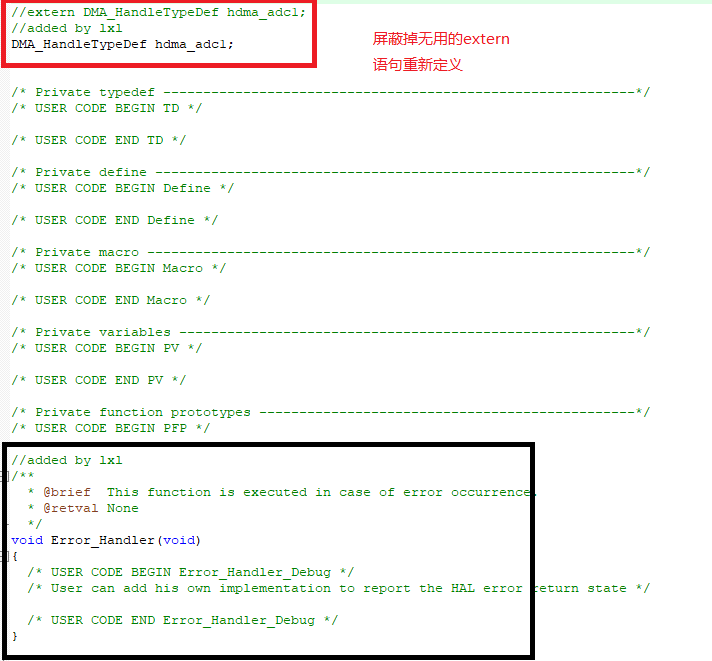
## 四、BSP裁剪

略

## 五、注意事项

1.**除**stm32f4xx\_hal\_msp.c外，我们手动在MDK工程中修改添加的代码内容，**不会**由于scons –target=mdk5重新生成而**覆盖**；

2.每次使用scons –target=mdk5重新生成工程时，唯一刷新的是stm32f4xx\_hal\_msp.c文件。因此，我们可能在添加某些外设后，编译可能有错误提示：stm32f4xx\_hal\_msp.c文件中**hdma\_adc1 和 Error\_Handler未定义**。这是因为它们是在STM32CubeMx生成的工程main函数定义的，而我们实际的工程并不使用这个main因此提示错误。我们只要在stm32f4xx\_hal\_msp.c文件**手动添加**它们即可。



3.使用STM32CubeMX时，建议先配置外设基本参数**后进行时钟配置**。比如配置RTC时，有时出现时钟源无法配置的情况；

## 六、参考文献

### 5.1 网络资源

1）stm32CubeMX如何自定义串口引脚

<https://jingyan.baidu.com/article/b0b63dbf2ba5254a483070d9.html>

2）PIN 设备

<https://www.rt-thread.org/document/site/#/rt-thread-version/rt-thread-standard/programming-manual/device/pin/pin>

3）STM32CubeMX系列教程13:实时时钟(RTC)

<https://www.waveshare.net/study/article-652-1.html>

4）RT-Thread文档中心

<https://www.rt-thread.org/document/site/#/rt-thread-version/rt-thread-standard/programming-manual/device/watchdog/watchdog>

### 5.2 书籍