**FreeRTOS学习之四：消息队列**

前提：默认已经装好MDK V5和STM32CubeMX，并安装了STM32F1xx系列的支持包。

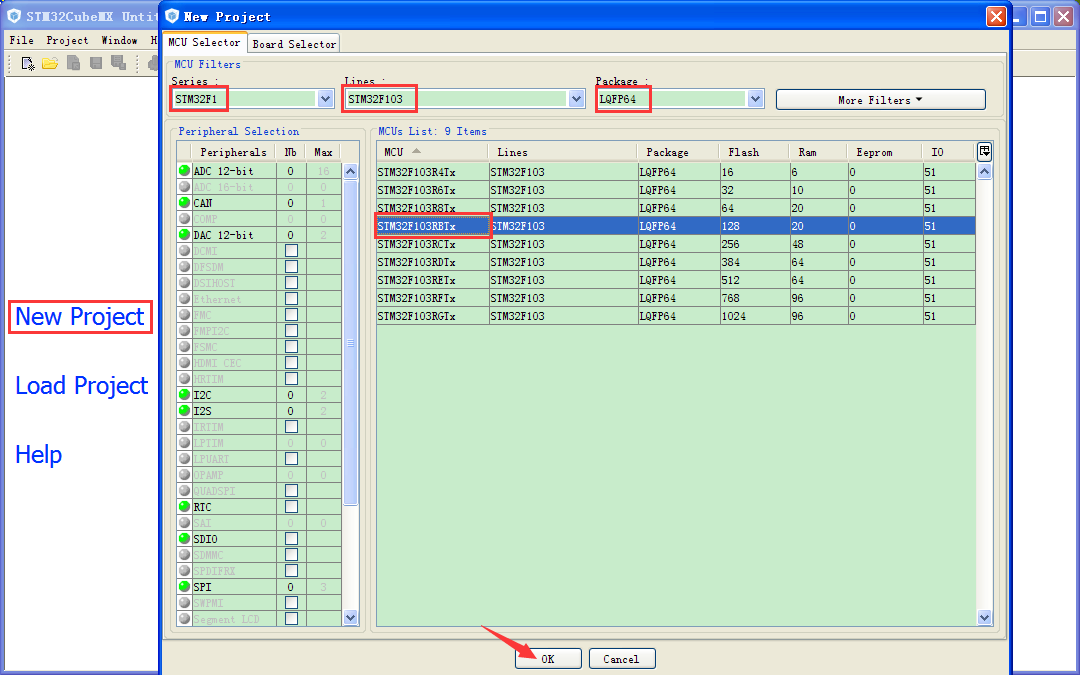
硬件平台：STM32F1xx系列。

目的：学习消息队列的使用。

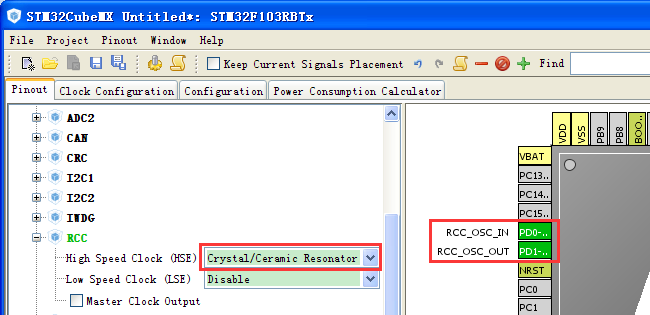
队列可以保存有限个具有确定长度的数据单元。通常情况下，队列被作为 FIFO(先进先出)使用，即数据由队列尾写入，从队列首读出。

本文例子使用STM32CubeMX配置创建两个任务，一个任务每隔一定时间发送一个消息到队列，另一个等待消息并根据消息的内容控制LED的闪烁次数。

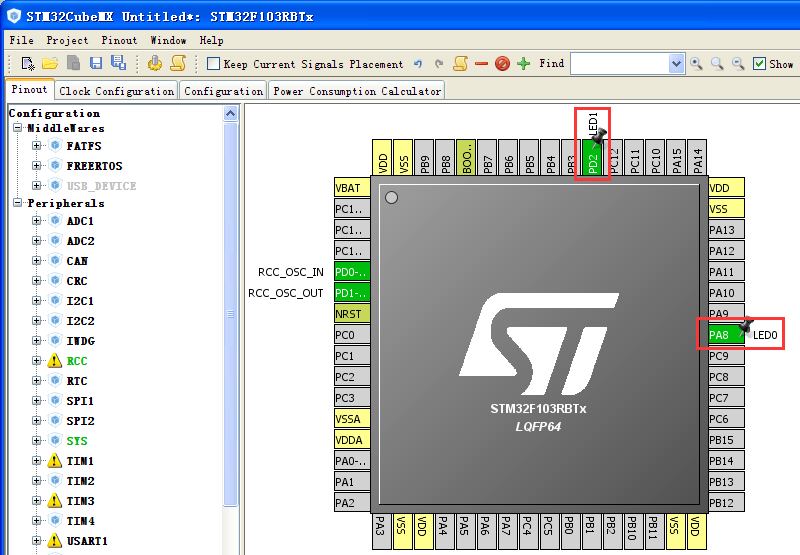
Step1.打开STM32CubeMX，点击“New Project”，选择芯片型号，STM32F103RBTx。



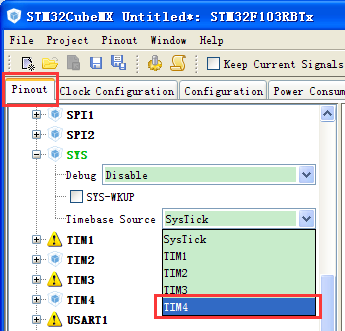
Step2.配置时钟引脚。



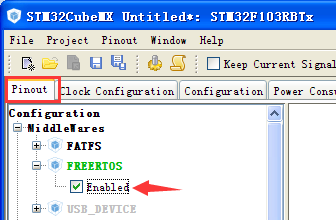
Step3.配置PA8和PD2为Output，并把用户标签分别改为LED0，LED1。



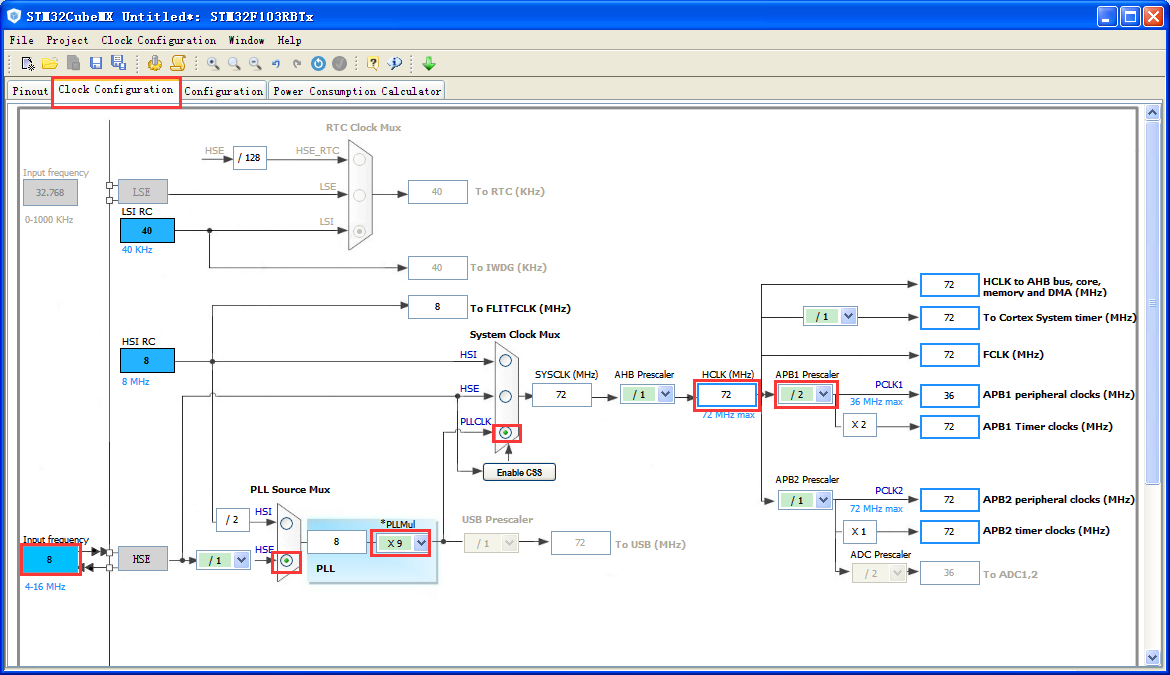
Step4.将系统时基源改为TIM4。



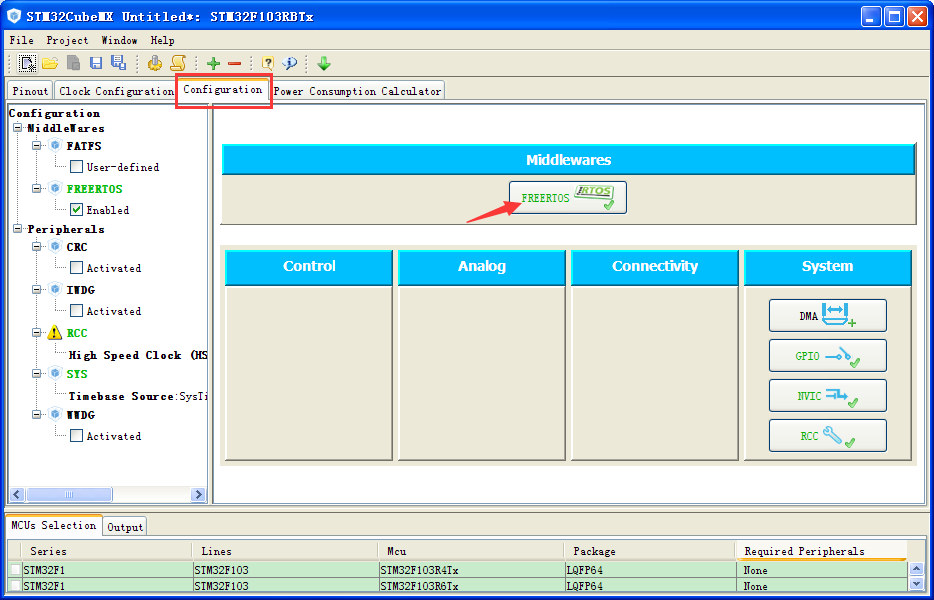
Step5.使能FreeRTOS。



Step6.配置时钟树。8M输入时，通过PLL得到72M内部时钟。

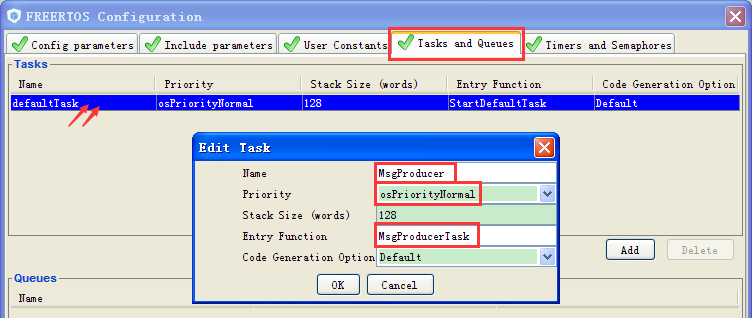


Step7.配置FreeRTOS。

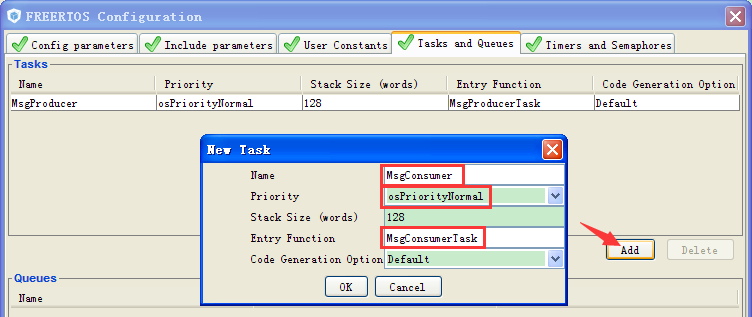


在Tasks and Queues选项卡，默认配置了一个名为defaultTask的任务，其优先级为普通，任务堆栈大小为128字，任务函数名为StartDefaultTask。

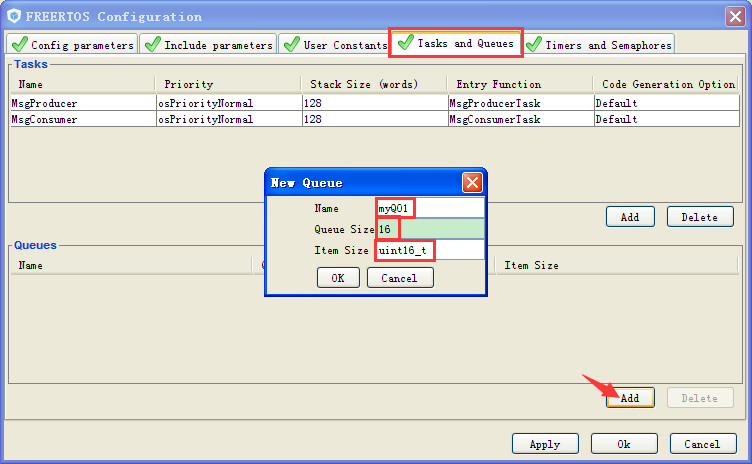
双击蓝色的地方，弹出对话框，将任务名修改为MsgProducer，将任务函数名修改为MsgProducerTask。



点击Add按钮，增加一个任务MsgConsumer，优先级设置为Normal，函数名为MsgConsumerTask。

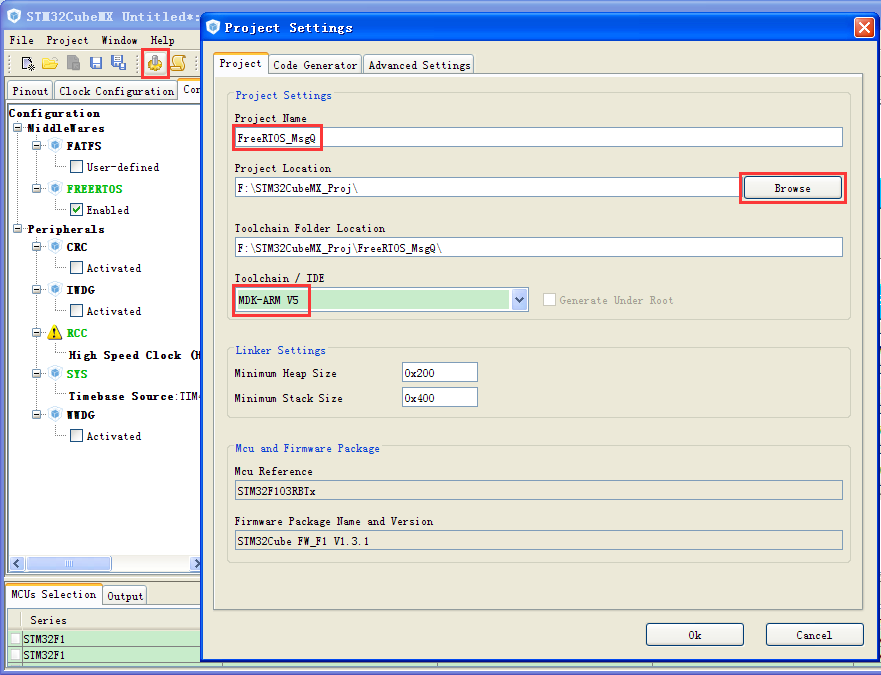


点击Queues栏的Add按钮，增加一个任务队列myQ01，深度为16，每个单元数据类型是uint16\_t。

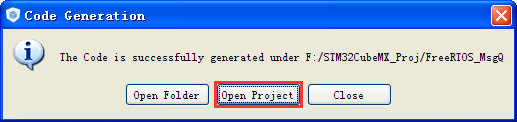


注：其他的都使用默认参数。

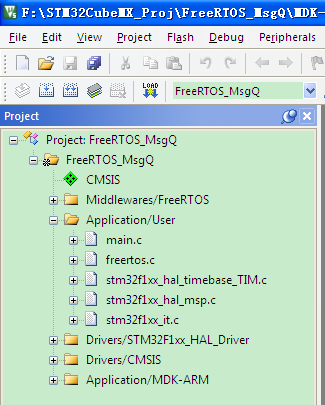
Step8.生成代码。



等完成后直接打开工程。

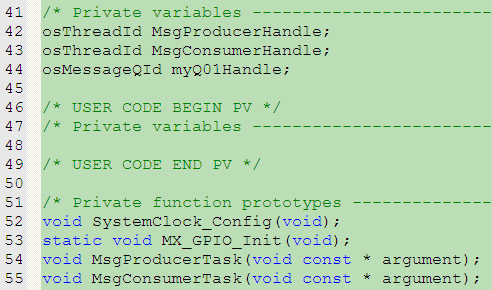


工程基本组织结构如下图，其中Application/User组中的文件是用户可以修改的，而其他组中的文件一般不进行修改。

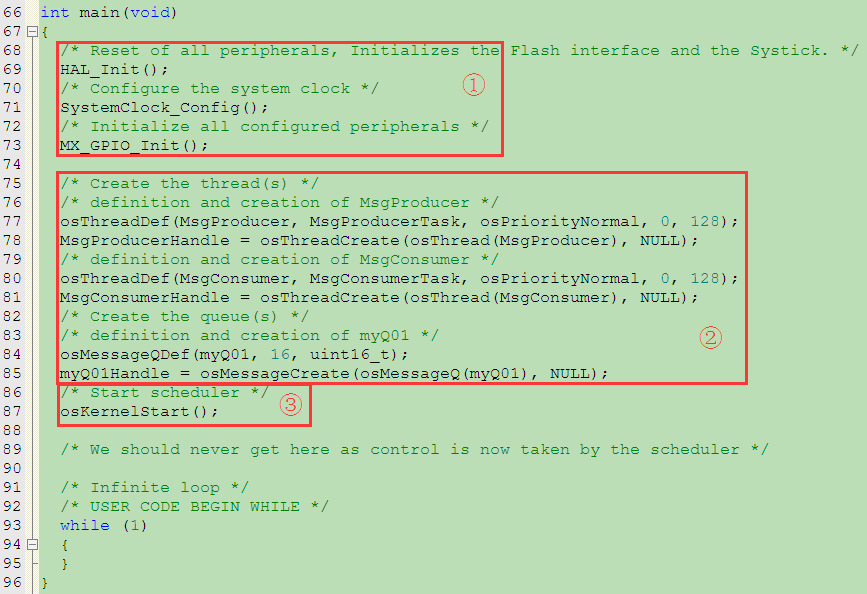


Step9.分析程序结构。

在进入main函数之前，先定义了几个变量，声明了几个函数。



再看main函数。将main函数整理，删除很多注释之后，得到下图所示内容。

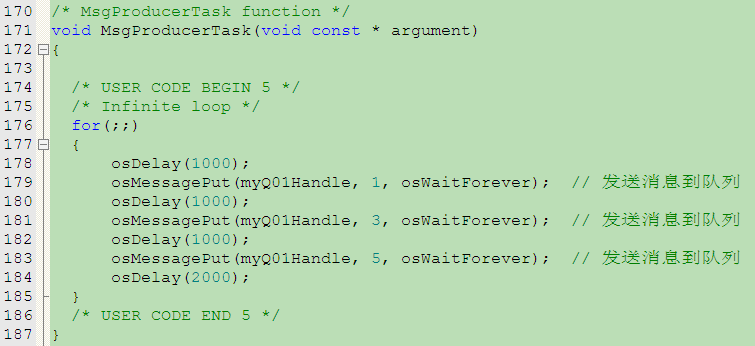


其中第①部分，是硬件配置；第②部分，创建两个任务和一个消息队列；第③部分，启动调度器。

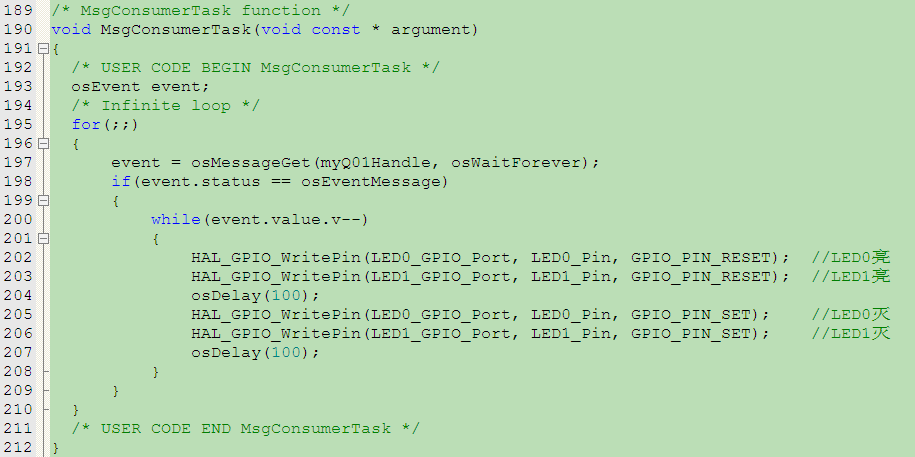
Step10.添加代码。

在main.c文件中，找到前面配置添加的两个任务函数，并在其中分别添加代码。

MsgProducerTask的功能是，发送1次消息，间隔一秒后发送1次，再间隔一秒发送1次，然后等待2秒。



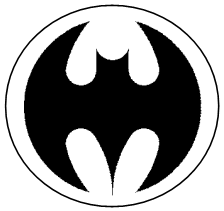
MsgConsumerTask的功能的，等待消息，然后根据消息的内容控制LED0和LED1闪烁次数。



Step11.编译下载运行。现象是，LED闪1次，一秒后闪3次，再过一秒闪5次，再等三秒，LED闪1次…如此循环。

程序分析：

消息队列和计数信号量相比，队列可以传递更多的信息。如果把消息的内容忽略，那么队列实现的实际上就是计数信号量的功能。



S.D.Lu 于 深圳

2016年8月