阿熊的FreeRTOS教程系列！

哈喽大家好！我是你们的老朋友阿熊！STM32教程系列更新完结已经有一段时间了，视频反馈还是不错的，从今天开始我们将会更新我们的FreeRTOS的教程

由于东西真的太多了，也纠结了很久要不要讲这个系列，毕竟难度真的很大，怕在难以做到那么通俗易懂，经过一段时间的考虑，还是决定好了给大家做一个入门级的讲解使用，由于FreeRTOS的内容真的很多，作为还是学生的我使用的也相对较少，操作系统层面的东西，我会用最大的能力去让大家理解，主要讲述主要功能，学完以后保证大伙可以理解80%以上的FreeRTOS的使用场景，好了废话不多说，开始我们的课程吧！

第九章：事件组（**event group**）

前面我们使用二值信号量进行数据的同步，但是我们在使用中会遇到这样一种情况，就比如说我们一个任务，他执行之前需要经过多个条件进行判断，当这多个条件全部满足或者多个条件中的某一个条件满足他才会执行

这种情况下我们使用二值信号量和互斥量好像都不靠谱，所以我们的FreeRTOS就有了另外一种东西也就是我们的事件组

壹：事件组的功能简介

我们的二值信号量就很像我们的标志位，而我们的事件组它就是24个标志位组合在一起，高八位保留

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
高8位保留																															

这样的话我们可以判断的附加条件就有很多元化，我们就可以完成或操作或者与操作，这样说可能没有什么概念，这里简单举一个例子：

就比如说我们发射核弹的任务，他必须要经过国家主席还有总司令同时确认，才可以发射出去，也就是他们俩的标志位都为一才可以执行，这个是与操作

就比如说你客厅的锁，只需要你家里人任何一个人有钥匙就都可以将它打开，这就是或操作

相对来说它的概念是比较容易理解的，然后我们这里直接讲一下它的常用函数

贰：事件组的基本函数

创建：

动态创建：

```
EventGroupHandle_t xEventGroupCreate( void );
//返回句柄或者NULL
```

静态创建：

```
EventGroupHandle_t xEventGroupCreateStatic( StaticEventGroup_t *
pxEventGroupBuffer );
```

删除：

```
void vEventGroupDelete( EventGroupHandle_t xEventGroup );
//传入句柄
```

设置事件组：

正常任务使用：

```
EventBits_t xEventGroupSetBits( EventGroupHandle_t xEventGroup,
                                const EventBits_t uxBitsToSet );
//传入句柄以及需要设置的标志位
//标志位为0不改变标志位
//标志位为1对应标志位置1
```

中断中使用：

```
BaseType_t xEventGroupSetBitsFromISR( EventGroupHandle_t
xEventGroup,
                                      const EventBits_t uxBitsToSet,
                                      BaseType_t *
pxHigherPriorityTaskWoken );
//传入句柄、标志位、判断是否需要切换任务
```

等待事件组：

```
EventBits_t xEventGroupWaitBits( EventGroupHandle_t xEventGroup,
                                 const EventBits_t uxBitsToWaitFor,
                                 const BaseType_t xClearOnExit,
                                 const BaseType_t xWaitForAllBits,
                                 TickType_t xTicksToWait );
//xEventGroup:任务句柄
//uxBitsToWaitFor：判断标志位
//xClearOnExit：是否清楚标志位中的uxBitsToWaitFor这几位
//xWaitForAllBits：判断模式
//              pdTRUE：等待的位，全部为 1;
//              pdFALSE：等待的位，某一个为 1 即可
//xTicksToWait：等待时间
//              0：立即返回
//              portMAX_DELAY：知道接收到为止
//返回值：事件组的值
```

其他：

可查阅相关手册或资料，这里不做赘述

叁：事件组的使用

实验一：事件组与操作判断模拟

创建一个事件组，以及三个任务，实现三个按键同时按下，串口打印对应消息

现象：

三个按键都最少按下一次才可以发射导弹

分析：

我们将判断模式设置为了全部符合，所以才会有此效果

实验二：事件组与操作判断模拟

创建一个事件组，以及三个任务，实现三个按键只要有一个按下，串口打印对应消息

现象：

无论按下哪一个按键都会发射导弹

分析：

我们将判断模式设置为了局部符合，所以才会有此效果