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！

第五章：任务的基本操作

上节课我们已经为大家讲了任务的状态，以及如何进行转换，本节课就是来进行实操，我们将教大家如何进行创建任务删除任务以及改变任务的状态

壹：任务的句柄

这里需要给大家先介绍一下这个概念也就是我们任务的句柄，相信我们在创建任务的时候就有提到过我们的参数一共有6个，最后一个参数是任务句柄参数（简称：**TCB**）
TaskControlBlock

复杂的东西咱们也不去讲，我们每个任务都有那么一大串东西，然后肯定需要用一个东西去管理它，然后我们的任务句柄就是起到这个这个作用的，你可以把它理解为一个指针指向了我们的那个任务，然后我们可以通过他找到我们的任务，去管理我们的任务

我们创建任务的时候把它赋给了一个句柄，然后我们的这个句柄就可以管理我们的这个任务，在我们删除暂停等一系列操作的时候，我们都可以直接传入，我们的这个句柄就可以达到我们的目的

贰：任务的基本操作展示

我们将用到的基本函数：

创建任务：**xTaskCreate()**

```
BaseType_t  xTaskCreate( TaskFunction_t pxTaskCode, // 函数指针，任务函数
                         const char * const pcName, // 任务的名字
                         const configSTACK_DEPTH_TYPE usStackDepth, // 栈大小,单位为 word
                         void * const pvParameters, // 调用任务函数时传入的参数
                         UBaseType_t uxPriority, // 优先级
                         TaskHandle_t * const pxCreatedTask ); // 任务句柄，以后使用它来操作这个任务
```

删除任务：**vTaskDelete()**

```
void    vTaskDelete(TaskHandle_t    xTaskToDelete);//传入任务句柄
        句柄为NULL时，删除自己
            为自己时，删除自己
            为其他任务时，删除其他任务
```

暂停任务：**vTaskSuspend()**

```
void    vTaskSuspend(TaskHandle_t    xTaskToSuspend);//传入任务句柄
        句柄为NULL时，暂停自己
            为自己时，暂停自己
            为其他任务时，暂停其他任务
```

恢复任务：**vTaskResume()**

```
void    vTaskResume(TaskHandle_t    vTaskResume);//传入任务句柄
        无法自己恢复自己，因为被暂停的任务是无法执行的
```

其他函数：

uxTaskPriorityGet():

```
UBaseType_t uxTaskPriorityGet( const TaskHandle_t xTask )//传入任务句柄
        此函数用来获取指定任务的优先级，使用INCLUDE_uxTaskPriorityGet函数的话应该定义为1
```

vTaskPrioritySet():

```
void    vTaskPrioritySet(xTaskHandle pxTask,unsigned portBASE_TYPEuxNewPriority);
        //传入任务句柄，新的优先级
        此函数用于改变某一个任务的任务优先级，要使用此函数的话宏INCLUDE_vTaskPrioritySet应该定义为1
```

uxTaskGetSystemState():

```
UBaseType_t uxTaskGetSystemState( TaskStatus_t * const
pxTaskStatusArray,
                                  const UBaseType_t
uxArraySize,
                                  uint32_t * const
pulTotalRunTime )
```

pxTaskStatusArray： 指向 TaskStatus_t 结构体类型的数组首地址，每个任务至少需要一个TaskStatus_t 结构体，任务的数量可以使用函数 uxTaskGetNumberOfTasks()

uxArraySize: 保存任务壮态数组的数组的大小。

pulTotalRunTime: 如果 configGENERATE_RUN_TIME_STATS 为 1 的话此参数用来保存系统总的运行时间。

返回值： 统计到的任务壮态的个数，也就是填写到数组 pxTaskStatusArray 中的个数，此值应该等于函数 uxTaskGetNumberOfTasks()的返回值。如果参数uxArraySize太小的话返回值可能为0

此函数用于获取系统中所有任务的任务壮态，每个任务的壮态信息保存在一个TaskStatus_t类型的结构体里面，这个结构体里面包含了任务的任务句柄、任务名字、堆栈、优先级等信息，要使用此函数的话宏 configUSE_TRACE_FACILITY 应该定义为 1

还有很多就不一一列举了，小伙伴们可以自行查找相关的手册或者其他资料

函数	描述
uxTaskPriorityGet()	查询某个任务的优先级。
vTaskPrioritySet()	改变某个任务的任务优先级。
uxTaskGetSystemState()	获取系统中任务状态。
vTaskGetInfo()	获取某个任务信息。
xTaskGetApplicationTaskTag()	获取某个任务的标签(Tag)值。
xTaskGetCurrentTaskHandle()	获取当前正在运行的任务的任务句柄。
xTaskGetHandle()	根据任务名字查找某个任务的句柄
xTaskGetIdleTaskHandle()	获取空闲任务的任务句柄。
uxTaskGetStackHighWaterMark()	获取任务的堆栈的历史剩余最小值，FreeRTOS 中叫做“高水位线”
eTaskGetState()	获取某个任务的壮态，这个壮态是 eTaskState 类型。
pcTaskGetName()	获取某个任务的任务名字。
xTaskGetTickCount()	获取系统时间计数器值。
xTaskGetTickCountFromISR()	在中断服务函数中获取时间计数器值
xTaskGetSchedulerState()	获取任务调度器的壮态，开启或未开启。
uxTaskGetNumberOfTasks()	获取当前系统中存在的任务数量。
vTaskList()	以一种表格的形式输出当前系统中所有任务的详细信息。
vTaskGetRunTimeStats()	获取每个任务的运行时间。
vTaskSetApplicationTaskTag()	设置任务标签(Tag)值。
SetThreadLocalStoragePointer()	设置线程本地存储指针
GetThreadLocalStoragePointer()	获取线程本地存储指针

实验：

创建任务一（点亮LED1间隔1000MS闪烁 并且 串口输出传入参数）

创建任务二（点亮LED2间隔500MS闪烁并且 串口输出传入参数）

创建任务三（使用KEY1删除和创建任务一）

创建任务四（使用KEY2暂停和恢复任务二）

现象：

按下KEY1，任务一被删除，再次按下新的任务一被创建

按下KEY2，任务二被暂停，再次按下任务二恢复执行

```
[2022-08-10 16:16:01.088 R]Task2#

[2022-08-10 16:16:02.098 R]Task1#
Task2#

[2022-08-10 16:16:02.947 R]已删除任务1

[2022-08-10 16:16:03.096 R]Task2#

[2022-08-10 16:16:04.093 R]Task2#

[2022-08-10 16:16:04.316 R]任务1不存在，已创建任务
任务创建完成!

[2022-08-10 16:16:05.093 R]Task2#

[2022-08-10 16:16:05.423 R]任务2#已暂停!

[2022-08-10 16:16:06.308 R]NEW Task1#

[2022-08-10 16:16:06.816 R]任务2#已恢复!

[2022-08-10 16:16:07.310 R]Task2#

[2022-08-10 16:16:08.316 R]Task2#
NEW Task1#

```

注意事项：任务被删除句柄并不会被删除，需要手动清空