**TMOS使用说明**

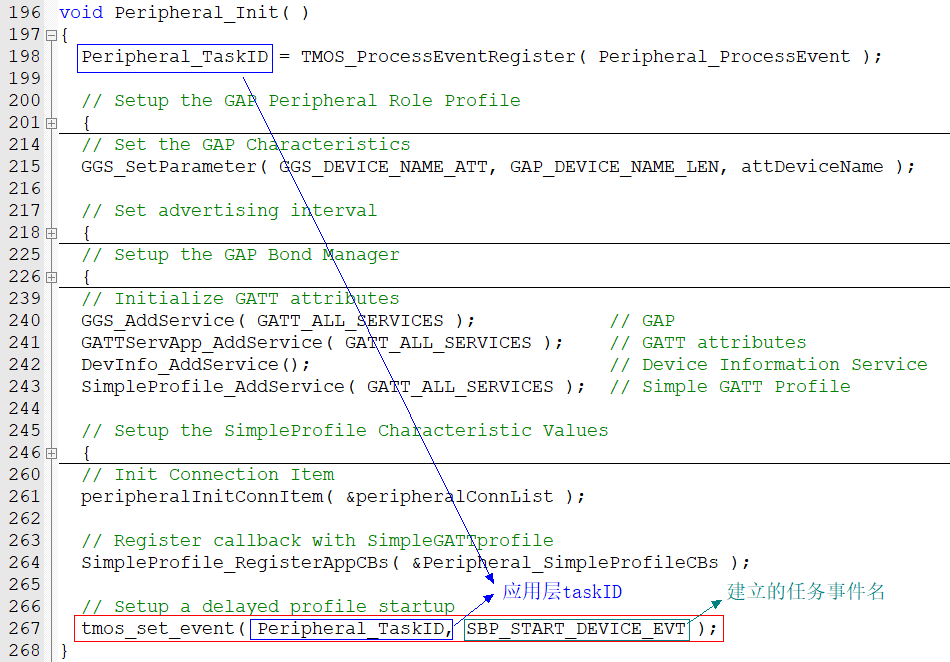
V1.0

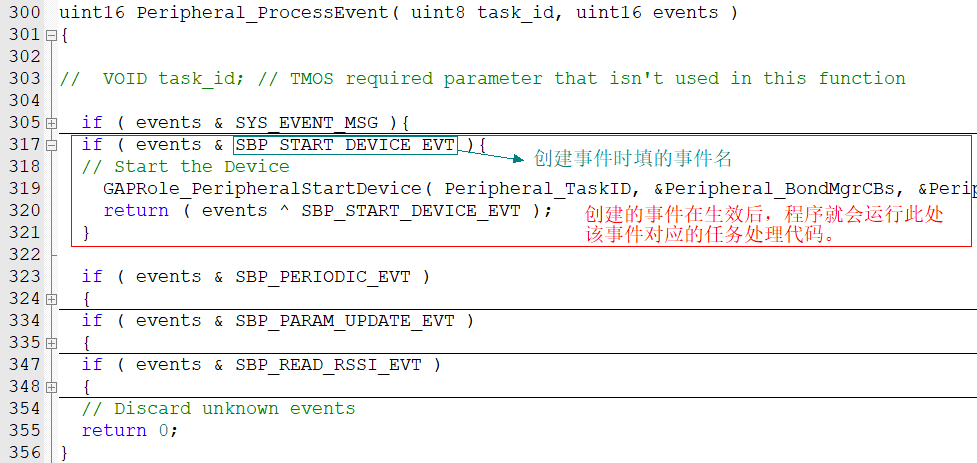
TMOS 系统时钟单位为 625us，以 RTC 为基准得到所有需要系统的时间。

**任务管理**  — 多任务管理方式实际上只有一个任务在运行，但是可以使用任务调度的策略将多个任务进行调度，每个任务占用一定的时间，所有的任务通过时间分片的方式处理。

extern bStatus\_t tmos\_set\_event( tmosTaskID taskID, tmosEvents event );

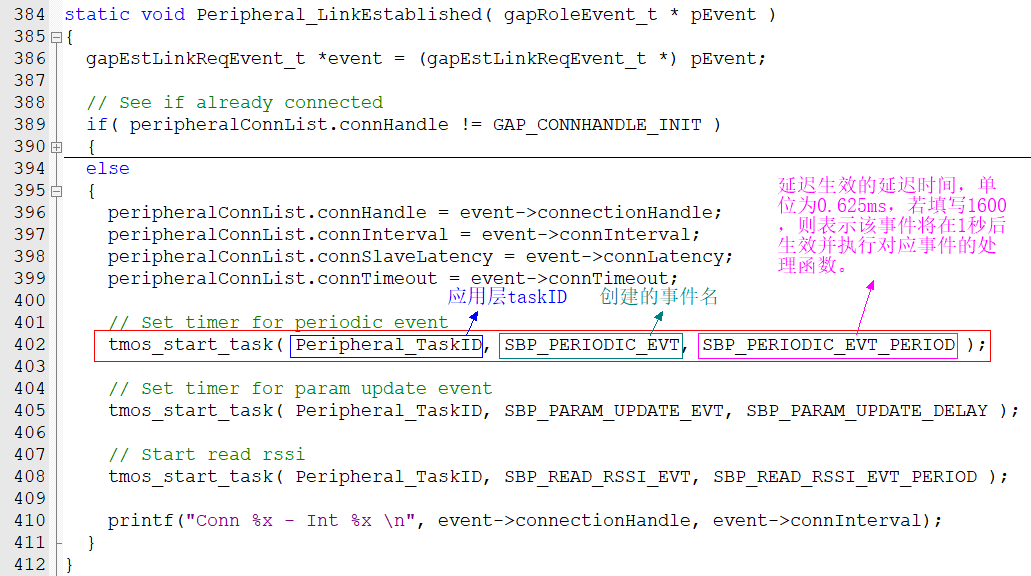
此函数将建立一个在taskID层生效的，名为event的任务，并立即生效

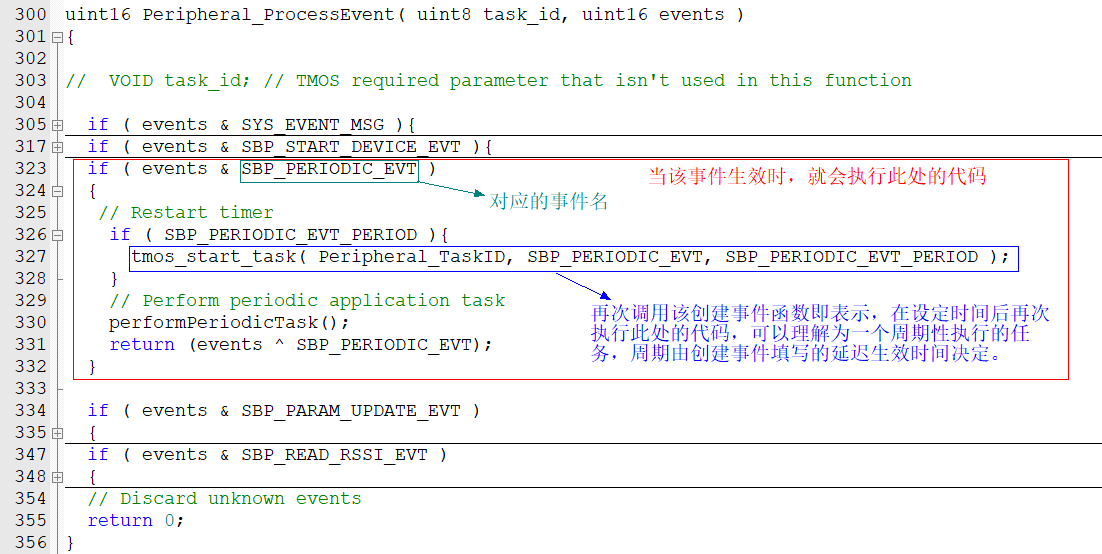




extern bStatus\_t tmos\_start\_task( tmosTaskID taskID, tmosEvents event, tmosTimer time );

此函数将建立一个在taskID层生效的，名为event的任务，并延迟time\*625us后生效。



 extern bStatus\_t tmos\_stop\_task( tmosTaskID taskID, tmosEvents event );

此函数将停止一个会在taskID层生效的，名为event的任务，调用此函数后，该事件任务将不会生效。

**任务调度函数使用注意事项：**

1. 禁止在中断中调用
2. 建议不要在单个任务中执行超过连接间隔一半时长的任务，否则将影响蓝牙通讯
3. 同理，在中断中建议不要执行超过连接间隔一半时长的任务，否则将影响蓝牙通讯
4. 在事件生效执行的代码中调用延时执行函数时，延时时间以当前事件生效时间点为基准偏移，所以对调用延时执行函数在生效执行的代码中摆放的位置没有要求。
5. 任务存在优先级，根据在xxx\_ProcessEvent函数中判断的先后顺序决定，同时生效的任务，先执行先判断，后执行后判断。注意，执行完先判断的事件任务后，要等到任务调度系统轮巡一遍后，才会执行后判断的事件任务。
6. 事件名按位定义，每一层taskID最多包含1个消息事件和15个任务事件（共16位）

**消息管理** — 消息是一个带有数据的事件，用于协议栈各层之间传递数据，支持同时添加多个消息。

extern u8 \*tmos\_msg\_allocate( u16 len );

申请内存函数，发送消息之前需要先给消息申请内存空间。如果返回为NULL，则申请失败。

extern bStatus\_t tmos\_msg\_send( tmosTaskID taskID, u8 \*msg\_ptr );

发送消息函数，参数为消息想要发送到哪一层的taskID以及消息指针。当调用此函数时，对应参数taskID层的消息事件将会立即置1生效。

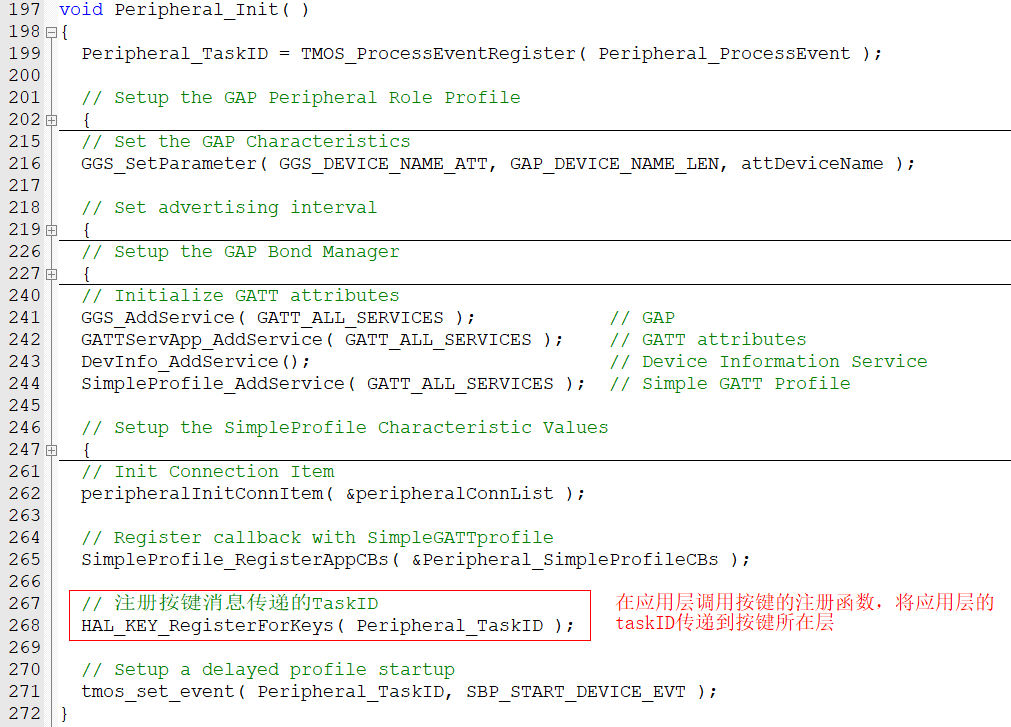
extern u8 \*tmos\_msg\_receive( tmosTaskID taskID );

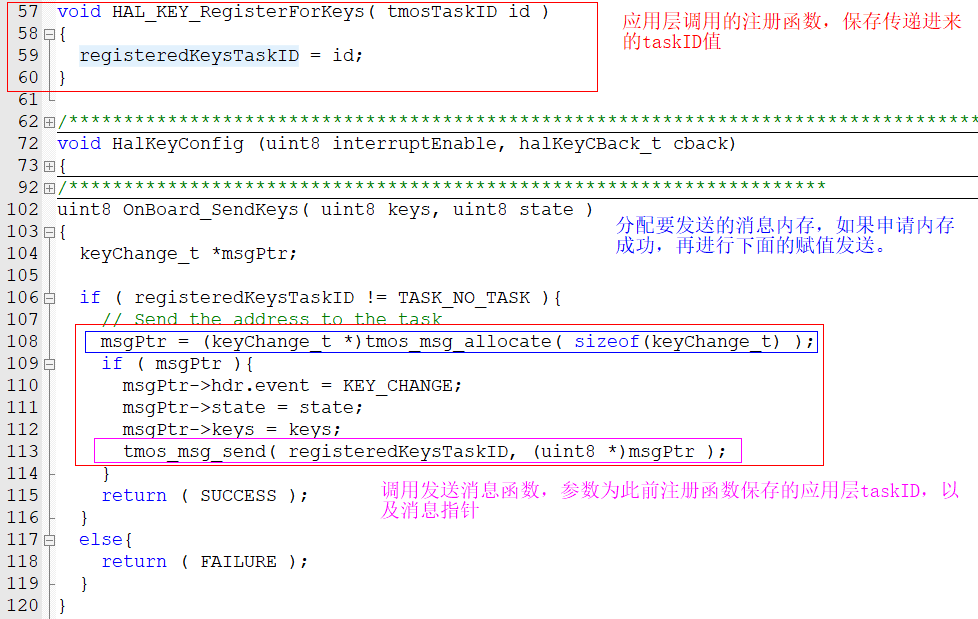
接收消息函数，参数为想要接收哪一层的taskID。

extern bStatus\_t tmos\_msg\_deallocate( u8 \*msg\_ptr );

释放消息占用内存的函数，处理完消息后需要释放内存占用。

**消息管理使用范例：**





**要发送的消息类型，下面处理消息时会用到**

