2019 TencentOS tiny 物联网操作系统

学习永无止境~

——杰杰

关于我

一个走在物联网路上的小菜鸟~

博客：https://jiejietop.cn

CSDN：https://blog.csdn.net/jiejiemcu

GitHub：https://github.com/jiejieTop

个人公众号

学习交流群：783234154

目录

01.队列基本概念

队列是一种常用于任务间通信的数据结构，队列可以在任务与任务间、中断和任务间传递消息，实现了任务接收来自其他任务或中断的不固定长度的消息，任务能够从队列里面读取消息，当队列中的消息是空时，读取消息的任务将被阻塞，用户还可以指定任务等待消息的时间timeout，在这段时间中，如果队列为空，该任务将保持阻塞状态以等待队列数据有效。当队列中有新消息时，被阻塞的任务会被唤醒并处理新消息；当等待的时间超过了指定的阻塞时间，即使队列中尚无有效数据，任务也会自动从阻塞态转为就绪态，消息队列是一种异步的通信方式。

通过队列服务，任务或中断服务例程可以将一条或多条消息放入队列中。同样，一个或多个任务可以从队列中获得消息。当有多个消息发送到队列时，通常是将先进入队列的消息先传给任务，也就是说，任务先得到的是最先进入队列的消息，即先进先出原则（FIFO），其实TencentOS tiny暂时不支持后进先出原则LIFO操作队列，但是支持后进先出操作消息队列。

可以参考博客：https://blog.csdn.net/jiejiemcu/article/details/80563422

- 消息支持先进先出方式排队，支持异步读写工作方式。
- 读消息队列支持超时机制
- 可以允许不同长度的任意类型消息
- 一个任务能够从任意一个消息队列接收和发送消息
- 多个任务能够从同一个消息队列接收和发送消息

02.队列阻塞机制

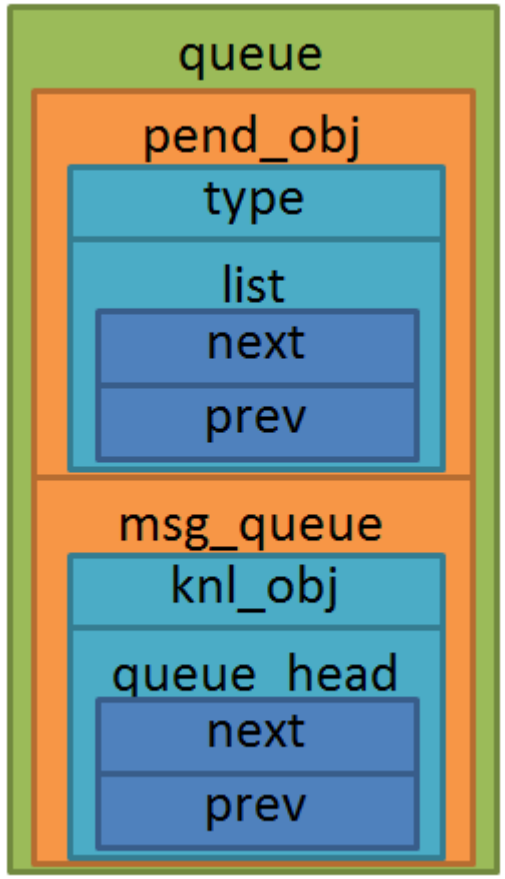
假设有一个任务A对某个队列进行读操作的时候（出队），发现它此时是没有消息的，那么此时任务A有3个选择：

1. 第一个选择，任务A扭头就走，既然队列没有消息，那我也不等了，干其它事情去，这样子任务A不会进入阻塞态；
2. 第二个选择，任务A还是在这里等等吧，可能过一会队列就有消息，此时任务A会进入阻塞状态，在等待着消息的到来，而任务A的等待时间就由我们自己指定，当阻塞的这段时间中任务A等到了队列的消息，那么任务A就会从阻塞态变成就绪态；假如等待超时了，队列还没消息，那任务A就不等了，从阻塞态中唤醒；
3. 第三个选择，任务A死等，不等到消息就不走了，这样子任务A就会进入阻塞态，直到完成读取队列的消息。

03.队列实现的数据结构

队列控制块

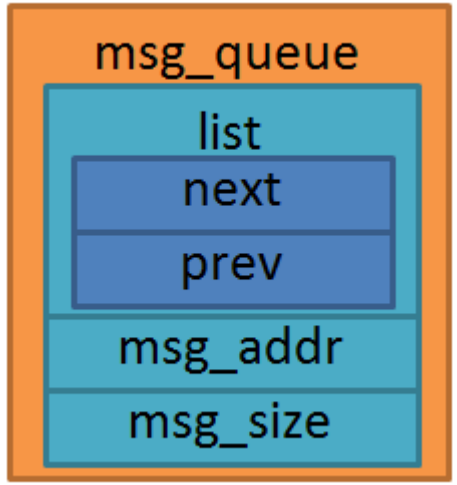
TencentOS tiny 通过队列控制块操作队列，其数据类型为k_queue_t，队列控制块由多个元素组成，主要有 pend_obj_t 类型的pend_obj以及k_msg_queue_t类型的msg_queue。其实整个队列的实现非常简单，主要靠msg_queue中的queue_head成员变量（其实是一个链表（列表）），所有的消息都会被记录在这个消息列表中，当读取消息的时候，会从消息列表读取消息。



03.队列实现的数据结构

消息控制块

除了上述的队列控制块外，还有消息队列控制块，这是因为TencentOS tiny中实现队列是依赖消息队列的，既然队列可以传递数据（消息），则必须存在一种可以存储消息的数据结构，我称之为消息控制块，消息控制块中记录了消息的存储地址msg_addr，以及消息的大小msg_size，此外还存在一个list成员变量，可以将消息挂载到队列的消息列表中。



03.队列实现的数据结构

任务控制块中的消息成员变量

假设任务A在队列中等待消息，而中断或其他任务往任务A等待的队列写入（发送）一个消息，那么这个消息不会被挂载到队列的消息列表中，而是会直接被记录在任务A的任务控制块中，表示任务A从队列中等待到这个消息，因此任务控制块必须存在一些成员变量用于记录消息相关信息（如消息地址、消息大小等）

与消息相关的宏定义

在tos_config.h中，使能队列组件的宏定义TOS_CFG_QUEUE_EN，使能消息队列组件宏定义TOS_CFG_MSG_EN，系统支持的消息池中消息个数宏定义TOS_CFG_MSG_POOL_SIZE。

消息池

在TencentOS tiny中定义了一个数组k_msg_pool[TOS_CFG_MSG_POOL_SIZE]作为消息池，它的数据类型是消息控制块类型k_msg_t，因为在使用消息队列的时候存取消息比较频繁，而在系统初始化的时候就将这个大数组的各个元素串初始化，并挂载到空闲消息列表中k_msg_freelist，组成我们说的消息池k_msg_pool，而池中的成员变量就是我们所说的消息。

04.创建队列

tos_queue_create()函数用于创建一个队列，队列就是一个数据结构，用于任务间的数据的传递。每创建一个新的队列都需要为其分配RAM，在创建的时候我们需要自己定义一个队列控制块，其内存是由编译器自动分配的。在创建的过程中实际上就是将队列控制块的内容进行初始化，将队列控制块的 pend_obj成员变量中的type 属性标识为PEND_TYPE_QUEUE，表示这是一个队列，然后调用消息队列中的API函数tos_msg_queue_create()将队列的消息成员变量msg_queue初始化，实际上就是初始化消息列表。

可以参考博客：https://blog.csdn.net/jiejiemcu/article/details/99687678

05.销毁队列

tos_queue_destroy()函数用于销毁一个队列，当队列不在使用是可以将其销毁，销毁的本质其实是将队列控制块的内容进行清除，首先判断一下队列控制块的类型是PEND_TYPE_QUEUE，这个函数只能销毁队列类型的控制块。然后判断是否有任务在等待队列中的消息，如果有则调用pend_wakeup_all()函数将这项任务唤醒，然后调用tos_msg_queue_flush()函数将队列的消息列表的消息全部“清空”，“清空”的意思是将挂载到队列上的消息释放回消息池（如果队列的消息列表存在消息，使用msgpool_free()函数释放消息），knl_object_deinit()函数是为了确保队列已经被销毁，此时队列控制块的pend_obj成员变量中的type 属性标识为KNL_OBJ_TYPE_NONE。最后在销毁队列后进行一次任务调度，以切换任务（毕竟刚刚很可能唤醒了任务）。

有一点要注意：因为队列控制块的RAM是由编译器静态分配的，所以即使是销毁了队列，这个内存也是没办法释放的~

06.清空队列

清空队列实际上就是将消息释放回消息池中，本质上还是调用tos_msg_queue_flush()函数。它是依赖于消息队列实现的。

07.等待队列（消息）

当任务试图从队列中的获取消息时，用户可以指定一个等待时间，当且仅当队列存在消息的时候，任务才能获取到消息。在等待的这段时间中，如果队列为空，该任务将保持阻塞状态以等待队列消息有效。当其他任务或中断服务程序往其等待的队列中写入了数据，该任务将自动由阻塞态转为就绪态。当任务等待发生超时，即使队列中尚无有效消息，任务也会自动从阻塞态转为就绪态。

参数	说明
queue	队列控制块指针
msg_addr	用于保存获取到的消息（这是输出的）
msg_size	用于保存获取到消息的大小（这是输出的）
timeout	等待时间（以k_tick_t为单位）

其中msg_addr与msg_size参数是用于保存函数返回的内容，即输出。

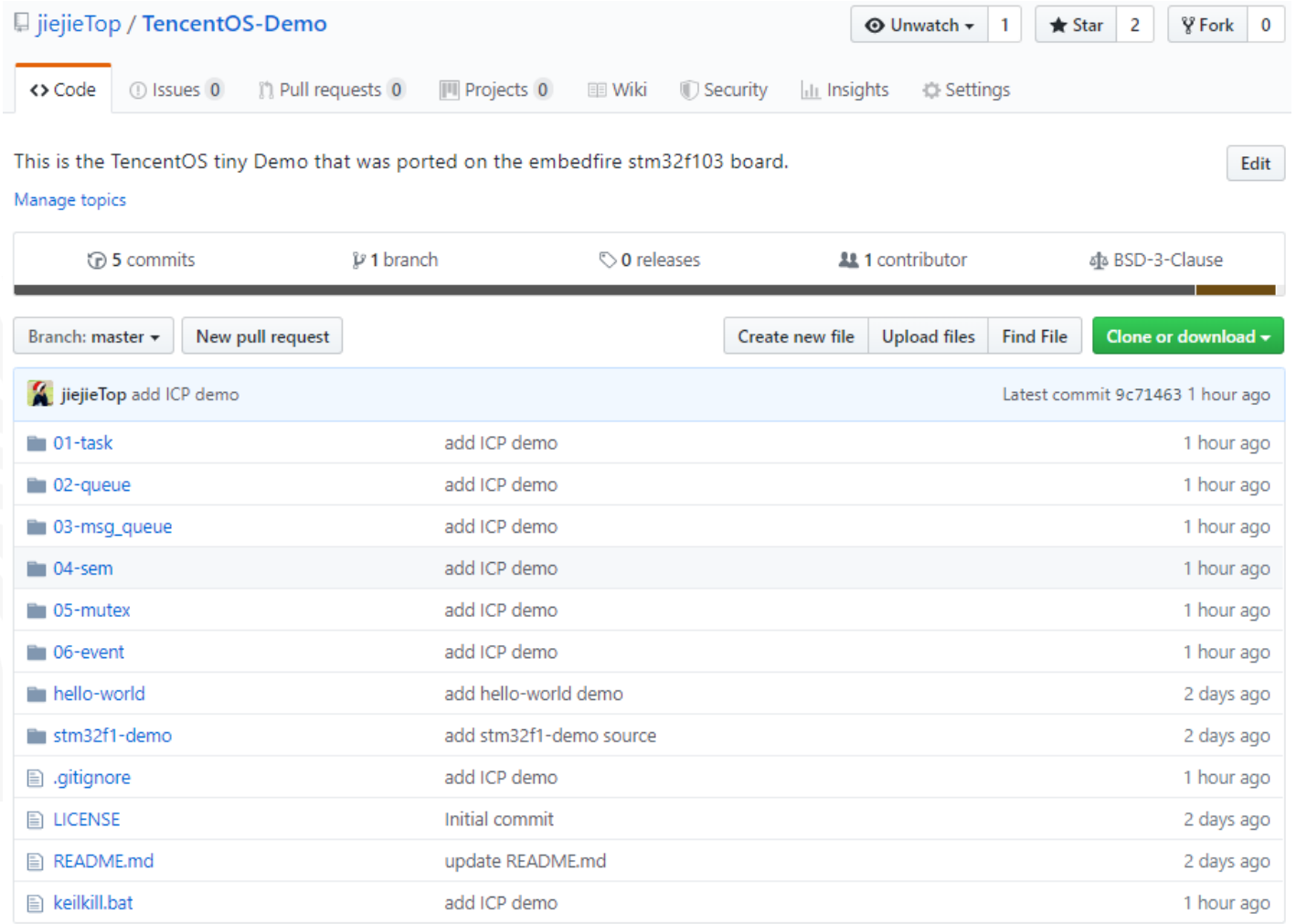
08.（消息）写入队列

任务或者中断服务程序都可以给消息队列发送消息，当发送消息时，TencentOS tiny会从消息池中取出一个消息，挂载到队列的消息列表末尾（FIFO发送方式）。tos_queue_post()是唤醒一个等待队列消息任务，tos_queue_post_all()则会唤醒所有等待队列消息的任务，无论何种情况，都是调用queue_do_post将消息写入队列中。

09. 实验

代码获取：https://github.com/jiejieTop/TencentOS-Demo

或者关注公众号，在后台回复“19”



THANKS