The purpose of this assignment is to add event queues and the associated support routines to the UIKAPI. This is the last piece of the UIK we will implement - yeah!

A common requirement is for a task to wait until an “event” occurs. An event is usually some sort of externally-triggered operation, such as an I/O device requiring service or a timer going off. One possibility is for an interrupt service routine to set an event flag, then allow tasks to wait until that flag is set. Once the flag is set, the task can become ready.

An event flag group is an O/S global variable (byte) that represents events by bits. For the purpose of this exercise, only one event flag will be defined: Bit 0 represents a timer event. This flag should be set whenever an auxiliary timer goes off. For our implementation, this timer can be implemented using one of the timers built into the Atmel AVR, separate from the timer you are using for the tick. You should write in ISR that handles an interrupt from this timer by setting bit 0 in the event flags variable.

You are to implement the following event functions:

- **void UIKAssocEvent(int Event)**; will cause the task to become “associated” with the flag(s) specified by Event. Event is a value that specifies the bits representing one or more events that the task should be associated with. The task should then block until the event occurs.

- **void UIKDisassocEvent(int Event)**; causes the task to no longer be associated with the event. In other words, the task will not be scheduled when the event occurs.

In addition to implementing the semaphore functions, design an application that demonstrates the proper operation of your event functions.