xTimerStart

[Timer API]

timers.h

```
 BaseType_t xTimerStart( TimerHandle_t xTimer,
                         TickType_t xBlockTime );
```

Timer functionality is provided by a timer service/daemon task. Many of the public FreeRTOS timer API functions send commands to the timer service task through a queue called the timer command queue. The timer command queue is private to the RTOS kernel itself and is not directly accessible to application code. The length of the timer command queue is set by the configTIMER_QUEUE_LENGTH configuration constant.

xTimerStart() starts a timer that was previously created using the xTimerCreate() API function. If the timer had already been started and was already in the active state, then xTimerStart() has equivalent functionality to the xTimerReset() API function.

Starting a timer ensures the timer is in the active state. If the timer is not stopped, deleted, or reset in the mean time, the callback function associated with the timer will get called 'n 'ticks after xTimerStart() was called, where 'n' is the timers defined period.

It is valid to call xTimerStart() before the RTOS scheduler has been started, but when this is done the timer will not actually start until the RTOS scheduler is started, and the timers expiry time will be relative to when the RTOS scheduler is started, not relative to when xTimerStart() was called.

The configUSE_TIMERS configuration constant must be set to 1 for xTimerStart() to be available.

Parameters:

xTimer	The handle of the timer being started/restarted.
xBlockTime	Specifies the time, in ticks, that the calling task should be held in the Blocked state to wait for the start command to be successfully sent to the timer command queue, should the queue already be full when xTimerStart() was called. xBlockTime is ignored if xTimerStart() is called before the RTOS scheduler is started.

Returns:

pdFAIL will be returned if the start command could not be sent to the timer command queue even after xBlockTime ticks had passed. pdPASS will be returned if the command was successfully sent to the timer command queue. When the command is actually processed will depend on the priority of the timer service/daemon task relative to other tasks in the system, although the timers expiry time is relative to when xTimerStart() is actually called. The timer service/daemon task priority is set by the configTIMER_TASK_PRIORITY configuration constant.

Example usage:

See the example on the xTimerCreate() documentation page.

[Back to the top] [About FreeRTOS] [Sitemap] [Report an error on this page]