xSemaphoreGive

[Semaphores]

semphr. h

```
xSemaphoreGive( SemaphoreHandle_t xSemaphore )
```

Macro to release a semaphore. The semaphore must have previously been created with a call to xSemaphoreCreateBinary(), xSemaphoreCreateMutex() or xSemaphoreCreateCounting(), and obtained using sSemaphoreTake().

This must not be used from an ISR. See xSemaphoreGiveFromISR() for an alternative which can be used from an ISR.

This macro must also not be used on semaphores created using xSemaphoreCreateRecursiveMutex().

Parameters:

xSemaphore	A handle to the semaphore being released. This is the handle returned when the semaphore was created.

Returns:

pdTRUE if the semaphore was released. pdFALSE if an error occurred. Semaphores are implemented using queues. An error can occur if there is no space on the queue to post a message - indicating that the semaphore was not first obtained correctly.

Example usage:

```
SemaphoreHandle_t xSemaphore = NULL;
void vATask( void * pvParameters )
{
   // Create the semaphore to guard a shared resource.  As we are using
   // the semaphore for mutual exclusion we create a mutex semaphore
   // rather than a binary semaphore.
   xSemaphore = xSemaphoreCreateMutex();

   if( xSemaphore != NULL )
   {
       if( xSemaphoreGive( xSemaphore ) != pdTRUE )
       {
           // We would expect this call to fail because we cannot give
           // a semaphore without first "taking" it!
       }
       // Obtain the semaphore - don't block if the semaphore is not
       // immediately available.
       if( xSemaphoreTake( xSemaphore, ( TickType_t ) 0 ) )
       {
           // We now have the semaphore and can access the shared resource.
           // ...
           // We have finished accessing the shared resource so can free the
           // semaphore.
           if( xSemaphoreGive( xSemaphore ) != pdTRUE )
           {
               // We would not expect this call to fail because we must have
               // obtained the semaphore to get here.
           }
       }
   }
}
```

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


freescale
Alliance Member
infineon
Texas Instruments
MCU
Developer Network
CYPRESS
FUJITSU
Microsemi
atollic
IAR
SYSTEMS
KEIL
Tools by ARM
Embedded
Artists