



Quality RTOS & Embedded Software
[About](#) [Contact](#) [Support](#) [FAQ](#)

FreeRTOS Tutorial Books and Reference Manuals
Become an expert, while supporting the FreeRTOS project

[Quick Start](#) | [Supported MCUs](#) | [Books & Kits](#) | [Trace Tools](#) | [Ecosystem](#) | [TCP & FAT](#) | [Training](#) | [Email List](#)  

[Home](#)
[FreeRTOS Books and Manuals](#)
FreeRTOS
About FreeRTOS
Features / Getting Started...
More Advanced...
Demo Projects
Supported Devices & Demos
API Reference
PDF Reference Manual
Task Creation
Task Control
Task Utilities
RTOS Kernel Control
Direct To Task Notifications
FreeRTOS-MPU Specific
Queues
Queue Sets
Semaphore / Mutexes
xSemaphoreCreateBinary()
vSemaphoreCreateBinary()
xSemaphoreCreateCounting()
xSemaphoreCreateMutex()
xSemaphoreCreateRecursiveMutex()
vSemaphoreDelete()
xSemaphoreGetMutexHolder()
xSemaphoreTake()
xSemaphoreTakeFromISR()
xSemaphoreTakeRecursive()
xSemaphoreGive()
xSemaphoreGiveRecursive()
xSemaphoreGiveFromISR()
Software Timers
Event Groups (or 'flags')
Co-routines
Contact, Support, Advertising
FreeRTOS Interactive!

xSemaphoreTake

[Semaphores]

semphr. h

```
xSemaphoreTake(  
    SemaphoreHandle_t xSemaphore,  
    TickType_t xTicksToWait  
)
```

Macro to obtain a semaphore. The semaphore must have previously been created with a call to xSemaphoreCreateBinary(), xSemaphoreCreateMutex() or xSemaphoreCreateCounting().

This macro must not be called from an ISR. xQueueReceiveFromISR() can be used to take a semaphore from within an interrupt if required, although this would not be a normal operation. Semaphores use queues as their underlying mechanism, so functions are to some extent interoperable.

Parameters:

xSemaphore A handle to the semaphore being taken - obtained when the semaphore was created.

xTicksToWait The time in ticks to wait for the semaphore to become available. The macro portTICK_PERIOD_MS can be used to convert this to a real time. A block time of zero can be used to poll the semaphore.

If **INCLUDE_vTaskSuspend** is set to '1' then specifying the block time as portMAX_DELAY will cause the task to block indefinitely (without a timeout).

Returns:

pdTRUE if the semaphore was obtained. pdFALSE if xTicksToWait expired without the semaphore becoming available.

Example usage:

```
SemaphoreHandle_t xSemaphore = NULL;  
  
/* A task that creates a semaphore. */  
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();  
}  
  
/* A task that uses the semaphore. */  
void vAnotherTask( void * pvParameters )  
{  
    /* ... Do other things. */  
  
    if( xSemaphore != NULL )  
    {  
        /* See if we can obtain the semaphore. If the semaphore is not  
        available wait 10 ticks to see if it becomes free. */  
        if( xSemaphoreTake( xSemaphore, ( TickType_t ) 10 ) == pdTRUE )  
        {  
            /* We were able to obtain the semaphore and can now access the  
            shared resource. */  
  
            /* ... */  
  
            /* We have finished accessing the shared resource. Release the  
            semaphore. */  
            xSemaphoreGive( xSemaphore );  
        }  
        else  
        {  
            /* We could not obtain the semaphore and can therefore not access  
            the shared resource safely. */  
        }  
    }  
}
```

Quick Start Guide
Download Source

FreeRTOS+ Lab Projects
FreeRTOS+TCP:
Thread safe TCP/IP stack
FreeRTOS+FAT:
Thread aware file system

FreeRTOS+ Ecosystem
Internet of Things:
Innovative complete solution
Fail Safe File System:
Ensures data integrity
InterNiche TCP/IP:
Low cost pre-ported libraries
FreeRTOS BSPs:
3rd party driver packages
FAT SL File System:
Super lean FAT FS
UDP/IP:
Thread aware UDP stack
Trace & Visualisation:
Tracealyzer for FreeRTOS
CLI:
Command line interface
WolfSSL SSL / TLS:
Networking security protocols
Safety:
TUV certified RTOS
RTOS Training:
Delivered online or on-site
IO:
read(), write(), ioctl() interface

[\[Back to the top \]](#) | [\[About FreeRTOS \]](#) | [\[Sitemap \]](#) | [\[Report an error on this page \]](#)

Latest News:

FreeRTOS V9.0.0rc1 is now available for [download](#) and comment.

Buildable Examples

FreeRTOS+TCP



FreeRTOS+FAT

Try Them Now

Sponsored Links

Now With No Code Size Limit!

TrueSTUDIO



The best **UNLIMITED**
FREE ARM[®] development
on the planet.

DOWNLOAD
NOW
WITHOUT
REGISTRATION

NO CODE-SIZE
LIMITATION



Free Download Without Registering

USB TCP/IP File Systems



Supplied as **integrated** and
functioning FreeRTOS
projects

from the
Official FreeRTOS Partner



Download
FreeRTOS
Project

Reliance Edge
The Tiny, Open Source,
Power Fail-safe File System for FreeRTOS

Copyright (C) 2004-2010 Richard Barry. Copyright (C) 2010-2016 Real Time Engineers Ltd.
Any and all data, files, source code, html content and documentation included in the FreeRTOS™ distribution or available on this site are the
exclusive property of Real Time Engineers Ltd.. See the files license.txt (included in the distribution) and this [copyright notice](#) for more
information. FreeRTOS™ and FreeRTOS.org™ are trade marks of Real Time Engineers Ltd.

