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. */
        }
    }
}
```