NAME

sem_timedwait, **sem_clockwait_np** - lock a semaphore

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <semaphore.h>
#include <time.h>

int
sem_timedwait(*sem_t * restrict sem, const struct timespec * restrict abs_timeout*);

int
sem_clockwait_np(*sem_t * restrict sem, clockid_t clock_id, int flags, const struct timespec * rqtp, struct timespec * rmtp*);

DESCRIPTION

The **sem_timedwait**() function locks the semaphore referenced by *sem*, as in the sem_wait(3) function. However, if the semaphore cannot be locked without waiting for another process or thread to unlock the semaphore by performing a sem_post(3) function, this wait will be terminated when the specified timeout expires.

The timeout will expire when the absolute time specified by *abs_timeout* passes, as measured by the clock on which timeouts are based (that is, when the value of that clock equals or exceeds *abs_timeout*), or if the absolute time specified by *abs_timeout* has already been passed at the time of the call.

Note that the timeout is based on the CLOCK_REALTIME clock.

The validity of the *abs_timeout* is not checked if the semaphore can be locked immediately.

The **sem_clockwait_np**() function is a more flexible variant of **sem_timedwait**(). The *clock_id* parameter specifies the reference clock. If the *flags* parameter contains TIMER_ABSTIME, then the requested timeout (*rqtp*) is an absolute timeout; otherwise, the timeout is relative. If this function fails with EINTR and the timeout is relative, a non-NULL *rmtp* will be updated to contain the amount of time remaining in the interval (the requested time minus the time actually slept). An absolute timeout has no effect on *rmtp*. A single structure can be used for both *rqtp* and *rmtp*.

RETURN VALUES

These functions return zero if the calling process successfully performed the semaphore lock operation

on the semaphore designated by *sem*. If the call was unsuccessful, the state of the semaphore is unchanged, and the function returns a value of -1 and sets the global variable *errno* to indicate the error.

ERRORS

These functions will fail if:

[EINVAL] The *sem* argument does not refer to a valid semaphore, or the process or thread would have blocked, and the *abs_timeout* parameter specified a nanoseconds field value less than zero or greater than or equal to 1000 million.

[ETIMEDOUT] The semaphore could not be locked before the specified timeout expired.

[EINTR] A signal interrupted this function.

SEE ALSO

sem_post(3), sem_trywait(3), sem_wait(3)

STANDARDS

The **sem_timedwait**() function conforms to IEEE Std 1003.1-2004 ("POSIX.1"). The **sem_clockwait_np**() function is not specified by any standard; it exists only on FreeBSD at the time of this writing.

HISTORY

The **sem_timedwait**() function first appeared in FreeBSD 5.0. The **sem_clockwait_np**() function first appeared in FreeBSD 11.1.