Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'clock_nanosleep.3p' command

$ man clock_nanosleep.3p

CLOCK_NANOSLEEP(3P) POSIX Programmer's Manual CLOCK_NANOSLEEP(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

clock_nanosleep ? high resolution sleep with specifiable clock

SYNOPSIS

#include <time.h>

int clock_nanosleep(clockid_t clock_id, int flags,

const struct timespec *rqtp, struct timespec *rmtp);

DESCRIPTION

If the flag TIMER_ABSTIME is not set in the flags argument, the clock_nanosleep() function shall cause the current thread to be sus? pended from execution until either the time interval specified by the rqtp argument has elapsed, or a signal is delivered to the calling thread and its action is to invoke a signal-catching function, or the process is terminated. The clock used to measure the time shall be the clock specified by clock_id.

If the flag TIMER_ABSTIME is set in the flags argument, the clock_nanosleep() function shall cause the current thread to be sus? pended from execution until either the time value of the clock speci?

fied by clock_id reaches the absolute time specified by the rqtp argu? ment, or a signal is delivered to the calling thread and its action is to invoke a signal-catching function, or the process is terminated.

If, at the time of the call, the time value specified by rqtp is less than or equal to the time value of the specified clock, then clock_nanosleep() shall return immediately and the calling process shall not be suspended.

The suspension time caused by this function may be longer than re? quested because the argument value is rounded up to an integer multiple of the sleep resolution, or because of the scheduling of other activity by the system. But, except for the case of being interrupted by a sig? nal, the suspension time for the relative clock_nanosleep() function (that is, with the TIMER_ABSTIME flag not set) shall not be less than the time interval specified by rqtp, as measured by the corresponding clock. The suspension for the absolute clock_nanosleep() function (that is, with the TIMER_ABSTIME flag set) shall be in effect at least until the value of the corresponding clock reaches the absolute time speci? fied by rqtp, except for the case of being interrupted by a signal.

The use of the clock_nanosleep() function shall have no effect on the action or blockage of any signal.

The clock_nanosleep() function shall fail if the clock_id argument refers to the CPU-time clock of the calling thread. It is unspecified whether clock_id values of other CPU-time clocks are allowed.

RETURN VALUE

If the clock_nanosleep() function returns because the requested time has elapsed, its return value shall be zero.

If the clock_nanosleep() function returns because it has been inter? rupted by a signal, it shall return the corresponding error value. For the relative clock_nanosleep() function, if the rmtp argument is non-NULL, the timespec structure referenced by it shall be updated to con? tain the amount of time remaining in the interval (the requested time minus the time actually slept). The rqtp and rmtp arguments can point to the same object. If the rmtp argument is NULL, the remaining time is

not returned. The absolute clock_nanosleep() function has no effect on the structure referenced by rmtp.

If clock_nanosleep() fails, it shall return the corresponding error value.

ERRORS

The clock_nanosleep() function shall fail if:

EINTR The clock_nanosleep() function was interrupted by a signal.

EINVAL The rqtp argument specified a nanosecond value less than zero or greater than or equal to 1000 million; or the TIMER_ABSTIME flag was specified in flags and the rqtp argument is outside the range for the clock specified by clock_id; or the clock_id argu? ment does not specify a known clock, or specifies the CPU-time clock of the calling thread.

ENOTSUP

The clock_id argument specifies a clock for which clock_nanosleep() is not supported, such as a CPU-time clock.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Calling clock_nanosleep() with the value TIMER_ABSTIME not set in the flags argument and with a clock_id of CLOCK_REALTIME is equivalent to calling nanosleep() with the same rqtp and rmtp arguments.

RATIONALE

The nanosleep() function specifies that the system-wide clock CLOCK_RE? ALTIME is used to measure the elapsed time for this time service. How? ever, with the introduction of the monotonic clock CLOCK_MONOTONIC a new relative sleep function is needed to allow an application to take advantage of the special characteristics of this clock.

There are many applications in which a process needs to be suspended and then activated multiple times in a periodic way; for example, to poll the status of a non-interrupting device or to refresh a display device. For these cases, it is known that precise periodic activation

cannot be achieved with a relative sleep() or nanosleep() function call. Suppose, for example, a periodic process that is activated at time T0, executes for a while, and then wants to suspend itself until time T0+T, the period being T. If this process wants to use the nanosleep() function, it must first call clock_gettime() to get the current time, then calculate the difference between the current time and T0+T and, finally, call nanosleep() using the computed interval. However, the process could be preempted by a different process between the two function calls, and in this case the interval computed would be wrong; the process would wake up later than desired. This problem would not occur with the absolute clock_nanosleep() function, since only one function call would be necessary to suspend the process until the de? sired time. In other cases, however, a relative sleep is needed, and that is why both functionalities are required.

Although it is possible to implement periodic processes using the timers interface, this implementation would require the use of signals, and the reservation of some signal numbers. In this regard, the reasons for including an absolute version of the clock_nanosleep() function in POSIX.1?2008 are the same as for the inclusion of the relative nanosleep().

It is also possible to implement precise periodic processes using pthread_cond_timedwait(), in which an absolute timeout is specified that takes effect if the condition variable involved is never signaled. However, the use of this interface is unnatural, and involves perform? ing other operations on mutexes and condition variables that imply an unnecessary overhead. Furthermore, pthread_cond_timedwait() is not available in implementations that do not support threads.

Although the interface of the relative and absolute versions of the new high resolution sleep service is the same clock_nanosleep() function, the rmtp argument is only used in the relative sleep. This argument is needed in the relative clock_nanosleep() function to reissue the func? tion call if it is interrupted by a signal, but it is not needed in the absolute clock_nanosleep() function call; if the call is interrupted by

a signal, the absolute clock_nanosleep() function can be invoked again with the same rqtp argument used in the interrupted call.

FUTURE DIRECTIONS

None.

SEE ALSO

clock_getres(), nanosleep(), pthread_cond_timedwait(), sleep()

The Base Definitions volume of POSIX.1?2017, <time.h>

COPYRIGHT