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 'sleep.3p' command

$ man sleep.3p

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

sleep ? suspend execution for an interval of time

SYNOPSIS

#include <unistd.h>

unsigned sleep(unsigned seconds);

DESCRIPTION

The sleep() function shall cause the calling thread to be suspended from execution until either the number of realtime seconds specified by the argument seconds has elapsed or a signal is delivered to the call? ing thread and its action is to invoke a signal-catching function or to terminate the process. The suspension time may be longer than requested due to the scheduling of other activity by the system.

In single-threaded programs, sleep() may make use of SIGALRM. In multi-threaded programs, sleep() shall not make use of SIGALRM and the re? mainder of this DESCRIPTION does not apply.

If a SIGALRM signal is generated for the calling process during execu? tion of sleep() and if the SIGALRM signal is being ignored or blocked

from delivery, it is unspecified whether sleep() returns when the SIGALRM signal is scheduled. If the signal is being blocked, it is also unspecified whether it remains pending after sleep() returns or it is discarded.

If a SIGALRM signal is generated for the calling process during execu? tion of sleep(), except as a result of a prior call to alarm(), and if the SIGALRM signal is not being ignored or blocked from delivery, it is unspecified whether that signal has any effect other than causing sleep() to return.

If a signal-catching function interrupts sleep() and examines or changes either the time a SIGALRM is scheduled to be generated, the ac? tion associated with the SIGALRM signal, or whether the SIGALRM signal is blocked from delivery, the results are unspecified.

If a signal-catching function interrupts sleep() and calls siglongjmp() or longjmp() to restore an environment saved prior to the sleep() call, the action associated with the SIGALRM signal and the time at which a SIGALRM signal is scheduled to be generated are unspecified. It is also unspecified whether the SIGALRM signal is blocked, unless the sig? nal mask of the process is restored as part of the environment.

Interactions between sleep() and setitimer() are unspecified.

RETURN VALUE

If sleep() returns because the requested time has elapsed, the value returned shall be 0. If sleep() returns due to delivery of a signal, the return value shall be the ``unslept'' amount (the requested time minus the time actually slept) in seconds.

ERRORS

No errors are defined.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

None.

RATIONALE

There are two general approaches to the implementation of the sleep() function. One is to use the alarm() function to schedule a SIGALRM sig? nal and then suspend the calling thread waiting for that signal. The other is to implement an independent facility. This volume of POSIX.1?2017 permits either approach in single-threaded programs, but the simple alarm/suspend implementation is not appropriate for multi-threaded programs.

In order to comply with the requirement that no primitive shall change a process attribute unless explicitly described by this volume of POSIX.1?2017, an implementation using SIGALRM must carefully take into account any SIGALRM signal scheduled by previous alarm() calls, the ac? tion previously established for SIGALRM, and whether SIGALRM was blocked. If a SIGALRM has been scheduled before the sleep() would ordi? narily complete, the sleep() must be shortened to that time and a SIGALRM generated (possibly simulated by direct invocation of the sig? nal-catching function) before sleep() returns. If a SIGALRM has been scheduled after the sleep() would ordinarily complete, it must be rescheduled for the same time before sleep() returns. The action and blocking for SIGALRM must be saved and restored.

Historical implementations often implement the SIGALRM-based version using alarm() and pause(). One such implementation is prone to infi? nite hangups, as described in pause(). Another such implementation uses the C-language setjmp() and longjmp() functions to avoid that win? dow. That implementation introduces a different problem: when the SIGALRM signal interrupts a signal-catching function installed by the user to catch a different signal, the longjmp() aborts that signal-catching function. An implementation based on sigprocmask(), alarm(), and sigsuspend() can avoid these problems.

Despite all reasonable care, there are several very subtle, but de? tectable and unavoidable, differences between the two types of imple? mentations. These are the cases mentioned in this volume of POSIX.1?2017 where some other activity relating to SIGALRM takes place, and the results are stated to be unspecified. All of these cases are

sufficiently unusual as not to be of concern to most applications.

See also the discussion of the term realtime in alarm().

Since sleep() can be implemented using alarm(), the discussion about alarms occurring early under alarm() applies to sleep() as well.

Application developers should note that the type of the argument sec? onds and the return value of sleep() is unsigned. That means that a Strictly Conforming POSIX System Interfaces Application cannot pass a value greater than the minimum guaranteed value for {UINT_MAX}, which the ISO C standard sets as 65535, and any application passing a larger value is restricting its portability. A different type was considered, but historical implementations, including those with a 16-bit int type, consistently use either unsigned or int.

Scheduling delays may cause the process to return from the sleep() function significantly after the requested time. In such cases, the re? turn value should be set to zero, since the formula (requested time mi? nus the time actually spent) yields a negative number and sleep() re? turns an unsigned.

FUTURE DIRECTIONS

A future version of this standard may require that sleep() does not make use of SIGALRM in all programs, not just multi-threaded programs.

SEE ALSO

alarm(), getitimer(), nanosleep(), pause(), sigaction(), sigsetjmp()

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

COPYRIGHT

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker?nel.org/doc/man-pages/reporting_bugs.html .