
*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 'sigsuspend.3p' command*

***$ man sigsuspend.3p***

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

sigsuspend ? wait for a signal

SYNOPSIS

```
#include <signal.h>
int sigsuspend(const sigset_t *sigmask);
```

DESCRIPTION

The sigsuspend() function shall replace the current signal mask of the calling thread with the set of signals pointed to by sigmask and then suspend the thread until delivery of a signal whose action is either to execute a signal-catching function or to terminate the process. This shall not cause any other signals that may have been pending on the process to become pending on the thread.

If the action is to terminate the process then sigsuspend() shall never return. If the action is to execute a signal-catching function, then sigsuspend() shall return after the signal-catching function returns, with the signal mask restored to the set that existed prior to the sig? suspend() call.

It is not possible to block signals that cannot be ignored. This is en?
forced by the system without causing an error to be indicated.

RETURN VALUE

Since sigsuspend() suspends thread execution indefinitely, there is no successful completion return value. If a return occurs, -1 shall be re? turned and errno set to indicate the error.

ERRORS

The sigsuspend() function shall fail if:

EINTR A signal is caught by the calling process and control is re? turned from the signal-catching function.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Normally, at the beginning of a critical code section, a specified set of signals is blocked using the sigprocmask() function. When the thread has completed the critical section and needs to wait for the previously blocked signal(s), it pauses by calling sigsuspend() with the mask that was returned by the sigprocmask() call.

RATIONALE

Code which wants to avoid the ambiguity of the signal mask for thread cancellation handlers can install an additional cancellation handler which resets the signal mask to the expected value.

```
void cleanup(void *arg)
{
    sigset_t *ss = (sigset_t *) arg;
    pthread_sigmask(SIG_SETMASK, ss, NULL);
}
int call_sigsuspend(const sigset_t *mask)
{
    sigset_t oldmask;
    int result;
    pthread_sigmask(SIG_SETMASK, NULL, &oldmask);
```

```
        pthread_cleanup_push(cleanup, &oldmask);
        result = sigsuspend(sigmask);
        pthread_cleanup_pop(0);
        return result;
    }
```

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.4, Signal Concepts, pause(), sigaction(), sigaddset(), sigdelset(), sigemptyset(), sigfillset()

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