



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

### ***Rocky Enterprise Linux 9.2 Manual Pages on command 'sigset.3'***

**\$ man sigset.3**

SIGSET(3)                      Linux Programmer's Manual                      SIGSET(3)

#### NAME

sigset, sighold, sigrelse, sigignore - System V signal API

#### SYNOPSIS

```
#include <signal.h>

typedef void (*sighandler_t)(int);

sighandler_t sigset(int sig, sighandler_t disp);

int sighold(int sig);

int sigrelse(int sig);

int sigignore(int sig);
```

Feature Test Macro Requirements for glibc (see feature\_test\_macros(7)):

```
sigset(), sighold(), sigrelse(), sigignore():

_XOPEN_SOURCE >= 500
```

#### DESCRIPTION

These functions are provided in glibc as a compatibility interface for programs that make use of the historical System V signal API. This API is obsolete: new applications should use the POSIX signal API (sigaction(2), sigprocmask(2), etc.)

The `sigset()` function modifies the disposition of the signal `sig`. The `disp` argument can be the address of a signal handler function, or one of the following constants:

##### SIG\_DFL

Reset the disposition of `sig` to the default.

##### SIG\_IGN

Ignore `sig`.

## SIG\_HOLD

Add sig to the process's signal mask, but leave the disposition of sig unchanged.

If disp specifies the address of a signal handler, then sig is added to the process's signal mask during execution of the handler.

If disp was specified as a value other than SIG\_HOLD, then sig is removed from the process's signal mask.

The dispositions for SIGKILL and SIGSTOP cannot be changed.

The sighold() function adds sig to the calling process's signal mask.

The sigrelse() function removes sig from the calling process's signal mask.

The sigignore() function sets the disposition of sig to SIG\_IGN.

## RETURN VALUE

On success, sigset() returns SIG\_HOLD if sig was blocked before the call, or the signal's previous disposition if it was not blocked before the call. On error, sigset() returns -1, with errno set to indicate the error. (But see BUGS below.)

The sighold(), sigrelse(), and sigignore() functions return 0 on success; on error, these functions return -1 and set errno to indicate the error.

## ERRORS

For sigset() see the ERRORS under sigaction(2) and sigprocmask(2).

For sighold() and sigrelse() see the ERRORS under sigprocmask(2).

For sigignore(), see the errors under sigaction(2).

## ATTRIBUTES

For an explanation of the terms used in this section, see attributes(7).

??

?Interface ? Attribute ? Value ?

??

?sigset(), sighold(), ? Thread safety ? MT-Safe ?

?sigrelse(), sigignore() ? ? ?

??

## CONFORMING TO

SVr4, POSIX.1-2001, POSIX.1-2008. These functions are obsolete: do not use them in new programs. POSIX.1-2008 marks sighold(), sigignore(), sigpause(3), sigrelse(), and sigset() as obsolete, recommending the use of sigaction(2), sigprocmask(2), pthread\_sigmask(3), and sigsuspend(2) instead.

## NOTES

These functions appeared in glibc version 2.1.

The `sighandler_t` type is a GNU extension; it is used on this page only to make the `sigset()` prototype more easily readable.

The `sigset()` function provides reliable signal handling semantics (as when calling `sigaction(2)` with `sa_mask` equal to 0).

On System V, the `signal()` function provides unreliable semantics (as when calling `sigaction(2)` with `sa_mask` equal to `SA_RESETHAND | SA_NODEFER`). On BSD, `signal()` provides reliable semantics. POSIX.1-2001 leaves these aspects of `signal()` unspecified. See `signal(2)` for further details.

In order to wait for a signal, BSD and System V both provided a function named `sigpause(3)`, but this function has a different argument on the two systems. See `sigpause(3)` for details.

## BUGS

In versions of glibc before 2.2, `sigset()` did not unblock sig if `disp` was specified as a value other than `SIG_HOLD`.

In versions of glibc before 2.5, `sigset()` does not correctly return the previous disposition of the signal in two cases. First, if `disp` is specified as `SIG_HOLD`, then a successful `sigset()` always returns `SIG_HOLD`. Instead, it should return the previous disposition of the signal (unless the signal was blocked, in which case `SIG_HOLD` should be returned). Second, if the signal is currently blocked, then the return value of a successful `sigset()` should be `SIG_HOLD`. Instead, the previous disposition of the signal is returned. These problems have been fixed since glibc 2.5.

## SEE ALSO

`kill(2)`, `pause(2)`, `sigaction(2)`, `signal(2)`, `sigprocmask(2)`, `raise(3)`, `sigpause(3)`, `sigvec(3)`, `signal(7)`

## COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <https://www.kernel.org/doc/man-pages/>.