_exit - Man Page

terminate the calling process

Synopsis

```
#include <unistd.h>

noreturn void _exit(int status);

#include <stdlib.h>

noreturn void _Exit(int status);
```

    Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

_Exit():

    _ISOC99_SOURCE || _POSIX_C_SOURCE >= 200112L

Description

_exit() terminates the calling process "immediately". Any open file descriptors belonging to the process are closed. Any children of the process are inherited by init(1) (or by the nearest "subreaper" process as defined through the use of the prctl(2) **PR_SET_CHILD_SUBREAPER** operation). The process's parent is sent a **SIGCHLD** signal.

The value *status & 0xFF* is returned to the parent process as the process's exit status, and can be collected by the parent using one of the wait(2) family of calls.

The function **_Exit**() is equivalent to **_exit**().

Return Value

These functions do not return.

Conforming to

POSIX.1-2001, POSIX.1-2008, SVr4, 4.3BSD. The function **_Exit**() was introduced by C99.

Notes

For a discussion on the effects of an exit, the transmission of exit status, zombie processes, signals sent, and so on, see exit(3).

The function **_exit**() is like exit(3), but does not call any functions registered with atexit(3) or on_exit(3). Open stdio(3) streams are not

_exit - Man Page

the delay is undesired, it may be useful to call functions like tcflush(3) before calling **_exit**(). Whether any pending I/O is canceled, and which pending I/O may be canceled upon **_exit**(), is implementation-dependent.

C library/kernel differences

In glibc up to version 2.3, the **_exit**() wrapper function invoked the kernel system call of the same name. Since glibc 2.3, the wrapper function invokes exit_group(2), in order to terminate all of the threads in a process.

The raw **_exit**() system call terminates only the calling thread, and actions such as reparenting child processes or sending **SIGCHLD** to the parent process are performed only if this is the last thread in the thread group.

See Also

execve(2), exit_group(2), fork(2), kill(2), wait(2), wait4(2), waitpid(2), atexit(3), exit(3), on_exit(3), termios(3)

Colophon

This page is part of release 5.13 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/.

Referenced By

atexit(3), clone(2), daemon(3), exit(3), exit_group(2), expect(1), fork(2), gprof(1), kill(2), on_exit(3), pagesend.8c(8), persistent-keyring(7), pmGetConfig(3), pmNoMem(3), prctl(2), ptrace(2), seccomp(2), setsid(2), shmop(2), signal-safety(7), socket(7), stress-ng(1), syscalls(2), system(3), user-keyring(7), user-session-keyring(7), vfork(2), wait(2).

The man pages exit(2) and _Exit(2) are aliases of _exit(2).

2021-03-22 Linux Programmer's Manual

_exit - Man Page