



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 'io_cancel.2'

\$ man io_cancel.2

IO_CANCEL(2) Linux Programmer's Manual IO_CANCEL(2)

NAME

io_cancel - cancel an outstanding asynchronous I/O operation

SYNOPSIS

```
#include <linux/aio_abi.h>            /* Defines needed types */
```

```
int io_cancel(aio_context_t ctx_id, struct iocb *iocb,  
              struct io_event *result);
```

Note: There is no glibc wrapper for this system call; see NOTES.

DESCRIPTION

Note: this page describes the raw Linux system call interface. The wrapper function provided by libaio uses a different type for the ctx_id argument. See NOTES.

The io_cancel() system call attempts to cancel an asynchronous I/O operation previously submitted with io_submit(2). The iocb argument describes the operation to be canceled and the ctx_id argument is the AIO context to which the operation was submitted. If the operation is successfully canceled, the event will be copied into the memory pointed to by result without being placed into the completion queue.

RETURN VALUE

On success, io_cancel() returns 0. For the failure return, see NOTES.

ERRORS

EAGAIN The iocb specified was not canceled.

EFAULT One of the data structures points to invalid data.

EINVAL The AIO context specified by ctx_id is invalid.

ENOSYS io_cancel() is not implemented on this architecture.

VERSIONS

The asynchronous I/O system calls first appeared in Linux 2.5.

CONFORMING TO

`io_cancel()` is Linux-specific and should not be used in programs that are intended to be portable.

NOTES

Glibc does not provide a wrapper function for this system call. You could invoke it using `syscall(2)`. But instead, you probably want to use the `io_cancel()` wrapper function provided by `libaio`.

Note that the `libaio` wrapper function uses a different type (`io_context_t`) for the `ctx_id` argument. Note also that the `libaio` wrapper does not follow the usual C library conventions for indicating errors: on error it returns a negated error number (the negative of one of the values listed in `ERRORS`). If the system call is invoked via `syscall(2)`, then the return value follows the usual conventions for indicating an error: -1, with `errno` set to a (positive) value that indicates the error.

SEE ALSO

`io_destroy(2)`, `io_getevents(2)`, `io_setup(2)`, `io_submit(2)`, `aio(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/>.