NAME

io_getevents − read asynchronous I/O events from the completion queue

LIBRARY

Standard C library (*libc*, *−lc*)

SYNOPSIS

```
#include <linux/aio_abi.h>    /* Definition of *io_* types */
#include <sys/syscall.h>      /* Definition of SYS_* constants */
#include <unistd.h>

int syscall(SYS_io_getevents, aio_context_t ctx_id,
        long min_nr, long nr, struct io_event *events,
        struct timespec *timeout);
```

Note: glibc provides no wrapper for **io_getevents**(), necessitating the use of **syscall**(2).

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_getevents**() system call attempts to read at least *min_nr* events and up to *nr* events from the completion queue of the AIO context specified by *ctx_id*.

The *timeout* argument specifies the amount of time to wait for events, and is specified as a relative timeout in a **timespec**(3) structure.

The specified time will be rounded up to the system clock granularity and is guaranteed not to expire early.

Specifying *timeout* as NULL means block indefinitely until at least *min_nr* events have been obtained.

RETURN VALUE

On success, **io_getevents**() returns the number of events read. This may be 0, or a value less than *min_nr*, if the *timeout* expired. It may also be a nonzero value less than *min_nr*, if the call was interrupted by a signal handler.

For the failure return, see NOTES.

ERRORS

EFAULT
: Either *events* or *timeout* is an invalid pointer.

EINTR
: Interrupted by a signal handler; see **signal**(7).

EINVAL
: *ctx_id* is invalid. *min_nr* is out of range or *nr* is out of range.

ENOSYS
: **io_getevents**() is not implemented on this architecture.

VERSIONS

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

STANDARDS

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

NOTES

You probably want to use the **io_getevents**() 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.

BUGS

An invalid *ctx_id* may cause a segmentation fault instead of generating the error **EINVAL**.

SEE ALSO

io_cancel(2), **io_destroy**(2), **io_setup**(2), **io_submit**(2), **timespec**(3), **aio**(7), **time**(7)