NAME

aio_read - asynchronous read

SYNOPSIS

#include <aio.h>

int aio_read(struct aiocb ***_aiocbp_);**

Link with _-lrt_.

DESCRIPTION

The **aio_read**() function queues the I/O request described by the buffer pointed to by _aiocbp_. This function is the asynchronous analog of read(2). The arguments of the call

read(fd, buf, count)

correspond (in order) to the fields _aio_fildes_, _aio_buf_, and _aio_nbytes_ of the structure pointed to by _aiocbp_. (See aio(7) for a description of the _aiocb_ structure.)

The data is read starting at the absolute file offset _aiocbp->aio_offset_, regardless of the current file offset. After the call, the value of the current file offset is unspecified.

The asynchronous means that this call returns as soon as the request has been enqueued; the read may or may not have completed when the call returns. One tests for completion using aio_error(3). The return status of a completed I/O operation can be obtained by aio_return(3). Asynchronous notification of I/O completion can be obtained by setting _aiocbp->aio_sigevent_ appropriately; see sigevent(7) for details.

If **_POSIX_PRIORITIZED_IO** is defined, and this file supports it, then the asynchronous operation is submitted at a priority equal to that of the calling process minus _aiocbp->aio_reqprio_.

The field _aiocbp->aio_lio_opcode_ is ignored.

No data is read from a regular file beyond its maximum offset.

RETURN VALUE

On success, 0 is returned. On error, the request is not enqueued, -1 is returned, and _errno_ is set appropriately. If an error is detected only later, it will be reported via aio_return(3) (returns status -1) and aio_error(3) (error status—whatever one would have gotten in _errno_, such as **EBADF**).

ERRORS

EAGAIN
Out of resources.

EBADF
_aio_fildes_ is not a valid file descriptor open for reading.

EINVAL
One or more of _aio_offset_, _aio_reqprio_, or _aio_nbytes_ are invalid.

ENOSYS
aio_read() is not implemented.

EOVERFLOW
The file is a regular file, we start reading before end-of-file and want at least one byte, but the starting position is past the maximum offset for this file.

VERSIONS

The **aio_read**() function is available since glibc 2.1.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008.

NOTES

It is a good idea to zero out the control block before use. The control block must not be changed while the read operation is in progress. The buffer area being read into must not be accessed during the operation or undefined results may occur. The memory areas involved must remain valid.

Simultaneous I/O operations specifying the same *aiocb* structure produce undefined results.

EXAMPLE

See aio(7).

SEE ALSO

aio_cancel(3), aio_error(3), aio_fsync(3), aio_return(3), aio_suspend(3), aio_write(3), lio_listio(3), aio(7)

COLOPHON

This page is part of release 3.74 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 http://www.kernel.org/doc/man-pages/.