NAME

aio_write − asynchronous write

LIBRARY

Real-time library (*librt*, *−lrt*)

SYNOPSIS

#include <aio.h>

int aio_write(struct aiocb *****aiocbp*);**

DESCRIPTION

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

```
write(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.)

If **O_APPEND** is not set, the data is written starting at the absolute position *aiocbp−>aio_offset*, regardless of the file offset. If**O_APPEND** is set, data is written at the end of the file in the same order as **aio_write**() calls are made. After the call, the value of the file offset is unspecified.

The "asynchronous" means that this call returns as soon as the request has been enqueued; the write 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 **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 written to 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 to indicate the error. 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 writing.

EFBIG
The file is a regular file, we want to write at least one byte, but the starting position is at or beyond the maximum offset for this file.

EINVAL
One or more of *aio_offset*, *aio_reqprio*, *aio_nbytes* are invalid.

ENOSYS
aio_write() is not implemented.

VERSIONS

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

ATTRIBUTES

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

Interface	Attribute	Value
aio_write()	Thread safety	MT-Safe

STANDARDS

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 write operation is in progress. The buffer area being written out 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.

SEE ALSO

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