

NAME

lio_listio - list directed I/O (REALTIME)

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

```
#include <aio.h>
```

```
int
```

```
lio_listio(int mode, struct aiocb * const list[], int nent, struct sigevent *sig);
```

DESCRIPTION

The **lio_listio**() function initiates a list of I/O requests with a single function call. The *list* argument is an array of pointers to *aiocb* structures describing each operation to perform, with *nent* elements. NULL elements are ignored.

The *aio_lio_opcode* field of each *aiocb* specifies the operation to be performed. The following operations are supported:

LIO_READ Read data as if by a call to `aio_read(2)`.

LIO_READV
Read data as if by a call to `aio_readv(2)`.

LIO_NOP No operation.

LIO_WRITE Write data as if by a call to `aio_write(2)`.

LIO_WRITEV
Write data as if by a call to `aio_writev(2)`.

If the *mode* argument is LIO_WAIT, **lio_listio**() does not return until all the requested operations have been completed. If *mode* is LIO_NOWAIT, *sig* can be used to request asynchronous notification when all operations have completed. If *sig* is NULL, no notification is sent.

For SIGEV_KEVENT notifications, the posted kevent will contain:

Member	Value
<i>ident</i>	<i>list</i>

filter EVFILT_LIO
udata value stored in *sig->sigev_value*

For SIGEV_SIGNO and SIGEV_THREAD_ID notifications, the information for the queued signal will include SI_ASYNCIO in the *si_code* field and the value stored in *sig->sigev_value* in the *si_value* field.

For SIGEV_THREAD notifications, the value stored in *sig->sigev_value* is passed to the *sig->sigev_notify_function* as described in `sigevent(3)`.

The order in which the requests are carried out is not specified; in particular, there is no guarantee that they will be executed in the order 0, 1, ..., *nent*-1.

RETURN VALUES

If *mode* is LIO_WAIT, the `lio_listio()` function returns 0 if the operations completed successfully, otherwise -1.

If *mode* is LIO_NOWAIT, the `lio_listio()` function returns 0 if the operations are successfully queued, otherwise -1.

ERRORS

The `lio_listio()` function will fail if:

- | | |
|----------|---|
| [EAGAIN] | There are not enough resources to enqueue the requests. |
| [EAGAIN] | The request would cause the system-wide limit {AIO_MAX} to be exceeded. |
| [EINVAL] | The <i>mode</i> argument is neither LIO_WAIT nor LIO_NOWAIT, or <i>nent</i> is greater than {AIO_LISTIO_MAX}. |
| [EINVAL] | The asynchronous notification method in <i>sig->sigev_notify</i> is invalid or not supported. |
| [EINTR] | A signal interrupted the system call before it could be completed. |
| [EIO] | One or more requests failed. |

In addition, the `lio_listio()` function may fail for any of the reasons listed for `aio_read(2)` and `aio_write(2)`.

If `lio_listio()` succeeds, or fails with an error code of EAGAIN, EINTR, or EIO, some of the requests

may have been initiated. The caller should check the error status of each *aio*cb structure individually by calling `aio_error(2)`.

SEE ALSO

`aio_error(2)`, `aio_read(2)`, `aio_readv(2)`, `aio_write(2)`, `aio_writv(2)`, `read(2)`, `write(2)`, `sigevent(3)`, `siginfo(3)`, `aio(4)`

STANDARDS

The `lio_listio()` function is expected to conform to IEEE Std 1003.1-2001 ("POSIX.1"). The `LIO_READV` and `LIO_WRITEV` operations are FreeBSD extensions, and should not be used in portable code.

HISTORY

The `lio_listio()` system call first appeared in FreeBSD 3.0.