NAME

open – open for reading or writing

SYNOPSIS

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int open(const char **path*, **int** *oflag*, **/* mode_t** *mode* ***/ ...);**

MT-LEVEL

Async-Signal-Safe

DESCRIPTION

open( ) opens a file descriptor for the file with the path name pointed to by *path*, and sets the file status flags according to the value of *oflag*. *oflag* values are constructed by OR-ing Flags from the following list (only one of the first three flags below may be used):

O_RDONLY Open for reading only.

O_WRONLY Open for writing only.

O_RDWR Open for reading and writing.

O_NDELAY or **O_NONBLOCK** These flags may affect subsequent reads and writes (see **read**(2) and **write**(2)). If both **O_NDELAY** and **O_NONBLOCK** are set, **O_NONBLOCK** will take precedence.

When opening a FIFO with **O_RDONLY** or **O_WRONLY** set:

If **O_NDELAY** or **O_NONBLOCK** is set: An **open** for reading-only will return without delay; an **open** for writing-only will return an error if no process currently has the file open for reading.

If **O_NDELAY** and **O_NONBLOCK** are clear: An **open** for reading-only will block until a process opens the file for writing; an **open** for writing-only will block until a process opens the file for reading.

When opening a file associated with a terminal line:

If **O_NDELAY** or **O_NONBLOCK** is set: The open will return without waiting for the device to be ready or available; subsequent behavior of the device is device specific.

If **O_NDELAY** and **O_NONBLOCK** are clear: The open will block until the device is ready or available.

O_APPEND If set, the file pointer will be set to the end of the file prior to each write.

O_DSYNC Write I/O operations on the file descriptor complete as defined by synchronized I/O data integrity completion.

O_RSYNC Read I/O operations on the file descriptor complete at the same level of integrity as specified by the **O_DSYNC** and **O_SYNC** flags. If both **O_DSYNC** and **O_RSYNC** are set in *oflag,* all I/O operations on the file descriptor complete as defined by synchronized I/O data integrity completion. If both **O_SYNC** and **O_RSYNC** are set in *oflag,* all I/O operations on the file descriptor complete as defined by synchronized I/O file integrity completion.

O_SYNC When opening a regular file, this flag affects subsequent writes. If set, each **write**(2) will wait for both the file data and file status to be physically updated. Write I/O operations on the file descriptor complete as defined by synchronized I/O file integrity completion.

O_NOCTTY If set and the file is a terminal, the terminal will not be allocated as the calling process's controlling terminal.

O_CREAT If the file exists, this flag has no effect, except as noted under **O_EXCL** below. Otherwise, the file is created and the owner ID of the file is set to the effective user ID of the process, the group ID of the file is set to the effective group ID of the process, or if the S_ISGID bit is set in the directory in which the file is being created, the file's group ID is set to the group ID of its parent directory. If the group ID of the new file does not match the effective group ID or one of the supplementary groups IDs, the **S_ISGID** bit is cleared. The access permission bits of the file mode are set to the value of *mode*, modified as follows (see **creat**(2)):

All bits set in the file mode creation mask of the process are cleared (see **umask**(2)).

The "save text image after execution bit" of the mode is cleared (see **chmod**(2)). **O_SYNC** Write I/O operations on the file descriptor complete as defined by synchronized I/O file integrity completion (see **fcntl**(5) definition of **O_SYNC .)**

O_EXCL If **O_EXCL** and **O_CREAT** are set, **open( )** will fail if the file exists. The check for the existence of the file and the creation of the file if it does not exist is atomic with respect to other processes executing **open( )** naming the same filename in the same directory with **O_EXCL** and **O_CREAT** set.

O_TRUNC If the file exists, its length is truncated to 0 and the mode and owner are unchanged. **O_TRUNC** has no effect on FIFO special files or directories.

When opening a STREAMS file, *oflag* may be constructed from **O_NDELAY** or **O_NONBLOCK** OR-ed with either **O_RDONLY, O_WRONLY,** or **O_RDWR.** Other flag values are not applicable to STREAMS devices and have no effect on them. The values of **O_NDELAY** and **O_NONBLOCK** affect the operation of STREAMS drivers and certain functions (see **read**(2), **getmsg**(2), **putmsg**(2), and **write**(2)). For drivers, the implementation of **O_NDELAY** and **O_NONBLOCK** is device specific. Each STREAMS device driver may treat these options differently.

When **open( )** is invoked to open a named stream, and the **connld** module (see **connld**(7M)) has been pushed on the pipe, **open( )** blocks until the server process has issued an **I_RECVFD ioctl** (see **streamio**(7I)) to receive the file descriptor.

If *path* is a symbolic link and **O_CREAT** and **O_EXCL** are set, the link is not followed.

The file pointer used to mark the current position within the file is set to the beginning of the file.

The new file descriptor is the lowest numbered file descriptor available and is set to remain open across **exec** functions (see **fcntl**(2)).

Certain flag values can be set following **open( )** as described in **fcntl**(2).

If **O_CREAT** is set and the file did not previously exist, upon successful completion **open( )** marks for update the **st_atime**, **st_ctime** and **st_mtime** fields of the file and the **st_ctime** and **st_mtime** fields of the parent directory.

If **O_TRUNC** is set and the file did previously exist, upon successful completion **open( )** marks for update the **st_ctime** and **st_mtime** fields of the file.

RETURN VALUES

Upon successful completion, the file descriptor is returned. Otherwise, a value of –**1** is returned and **errno** is set to indicate the error.

ERRORS

The named file is opened unless one or more of the following are true:

EACCES The file does not exist and write permission is denied by the parent directory of the file to be created.

O_TRUNC is specified and write permission is denied

A component of the path prefix denies search permission.

oflag permission is denied for an existing file.

EAGAIN If the file exists with enforced record locking enabled, record locks are on the file (see **chmod**(2)**),** and **O_TRUNC** is specified.

EEXIST **O_CREAT** and **O_EXCL** are set, and the named file exists.

EFAULT *path* points to an illegal address.

EINTR A signal was caught during the **open( )** function.

EIO A hangup or error occurred during the open of the STREAMS-based device.

EISDIR The named file is a directory and *oflag* is write or read/write.

ELOOP Too many symbolic links were encountered in translating *path*.

EMFILE The process has too many open files (see **getrlimit**(2)).

EMULTIHOP Components of *path* require hopping to multiple remote machines and the file system does not allow it.

ENAMETOOLONG The length of the *path* argument exceeds {**PATH_MAX**}, or the length of a *path* component exceeds {**NAME_MAX**} while {**_POSIX_NO_TRUNC**} is in effect.

ENFILE The system file table is full.

ENOENT **O_CREAT** is not set and the named file does not exist.

O_CREAT is set and a component of the path prefix does not exist or is the null pathname.

ENOLINK *path* points to a remote machine, and the link to that machine is no

	longer active.
ENOMEM	The system is unable to allocate a send descriptor.
ENOSPC	**O_CREAT** and **O_EXCL** are set, and the file system is out of inodes.
	O_CREAT is set and the directory cannot be extended.
ENOSR	Unable to allocate a stream.
ENOTDIR	A component of the path prefix is not a directory.
ENXIO	The named file is a character special or block special file, and the device associated with this special file does not exist.
	O_NDELAY or **O_NONBLOCK** is set, the named file is a FIFO, **O_WRONLY** is set, and no process has the file open for reading.
	A STREAMS module or driver open routine failed.
EROFS	The named file resides on a read-only file system and either **O_WRONLY** , **O_RDWR** , **O_CREAT** , or **O_TRUNC** is set in *oflag* (if the file does not exist).

SEE ALSO

intro(2), **chmod**(2), **close**(2), **creat**(2), **dup**(2), **exec**(2), **fcntl**(2), **getmsg**(2), **getrlimit**(2), **lseek**(2), **putmsg**(2), **read**(2), **stat**(2), **umask**(2), **write**(2), **stat**(5), **fcntl**(5), **connld**(7M), **streamio**(7I)