NAME

mq_send, mq_timedsend - send a message to a message queue

SYNOPSIS

#include <mqueue.h>

int mq_send(mqd_t *mqdes*, **const char ***_msg_ptr_,
size_t *msg_len*, **unsigned int** *msg_prio*);

#include <time.h>
#include <mqueue.h>

int mq_timedsend(mqd_t *mqdes*, **const char ***_msg_ptr_,
size_t *msg_len*, **unsigned int** *msg_prio*,
**const struct timespec ***_abs_timeout_);

Link with *-lrt*.

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

mq_timedsend():
_XOPEN_SOURCE >= 600 || _POSIX_C_SOURCE >= 200112L

DESCRIPTION

mq_send() adds the message pointed to by *msg_ptr* to the message queue referred to by the descriptor *mqdes*. The*msg_len* argumen t specifies the length of the message pointed to by *msg_ptr*; this length must be less than or equal to the queue's *mq_msgsize* attribute. Zero-length messages are allowed.

The *msg_prio* argument is a nonnegative integer that specifies the priority of this message. Messages are placed on the queue in decreasing order of priority, with newer messages of the same priority being placed after older messages with the same priority.

If the message queue is already full (i.e., the number of messages on the queue equals the queue's *mq_maxmsg* attribute), then, by default, **mq_send**() blocks until sufficient space becomes available to allow the message to be queued, or until the call is interrupted by a signal handler. If the **O_NONBLOCK** flag is enabled for the message queue description, then the call instead fails immediately with the error **EAGAIN**.

mq_timedsend() behaves just like **mq_send**(), except that if the queue is full and the **O_NONBLOCK** flag is not enabled for the message queue description, then *abs_timeout* points to a structure which specifies a ceiling on the time for which the call will block. This ceiling is an absolute timeout in seconds and nanoseconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC), and it is specified in the following structure:

```
struct timespec {
time_t tv_sec; /* seconds */
long tv_nsec; /* nanoseconds */
};
```

If the message queue is full, and the timeout has already expired by the time of the call, **mq_timedsend**() returns immediately.

RETURN VALUE

On success, **mq_send**() and **mq_timedsend**() return zero; on error, -1 is returned, with *errno* set to indicate the error.

ERRORS

EAGAIN
The queue was full, and the **O_NONBLOCK** flag was set for the message queue description referred to by *mqdes*.

EBADF
The descriptor specified in *mqdes* was invalid.

EINTR
The call was interrupted by a signal handler; see signal(7).

EINVAL
The call would have blocked, and *abs_timeout* was invalid, either because *tv_sec* was less than zero, or because *tv_nsec* was less than zero or greater than 1000 million.

EMSGSIZE
msg_len was greater than the *mq_msgsize* attribute of the message queue.

ETIMEDOUT
The call timed out before a message could be transferred.

ATTRIBUTES

Multithreading (see pthreads(7))
The **mq_send**() and **mq_timedsend**() functions are thread-safe.

CONFORMING TO

POSIX.1-2001.

NOTES

On Linux, **mq_timedsend**() is a system call, and **mq_send**() is a library function layered on top of that system call.

SEE ALSO

mq_close(3), mq_getattr(3), mq_notify(3), mq_open(3), mq_receive(3), mq_unlink(3), mq_overview(7), time(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/.