NAME

mq_send, **mq_timedsend** - send a message to message queue (REALTIME)

LIBRARY

POSIX Real-time Library (librt, -lrt)

SYNOPSIS

#include <mqueue.h>

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

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

DESCRIPTION

The **mq_send**() system call adds the message pointed to by the argument *msg_ptr* to the message queue specified by *mqdes*. The *msg_len* argument specifies the length of the message, in bytes, pointed to by *msg_ptr*. The value of *msg_len* should be less than or equal to the *mq_msgsize* attribute of the message queue, or **mq_send**() will fail.

If the specified message queue is not full, **mq_send**() will behave as if the message is inserted into the message queue at the position indicated by the *msg_prio* argument. A message with a larger numeric value of *msg_prio* will be inserted before messages with lower values of *msg_prio*. A message will be inserted after other messages in the queue, if any, with equal *msg_prio*. The value of *msg_prio* should be less than {MQ_PRIO_MAX}.

If the specified message queue is full and O_NONBLOCK is not set in the message queue description associated with *mqdes*, **mq_send**() will block until space becomes available to enqueue the message, or until **mq_send**() is interrupted by a signal. If more than one thread is waiting to send when space becomes available in the message queue and the Priority Scheduling option is supported, then the thread of the highest priority that has been waiting the longest will be unblocked to send its message. Otherwise, it is unspecified which waiting thread is unblocked. If the specified message queue is full and O_NONBLOCK is set in the message queue description associated with *mqdes*, the message will not be queued and **mq_send**() will return an error.

The **mq_timedsend**() system call will add a message to the message queue specified by *mqdes* in the manner defined for the **mq_send**() system call. However, if the specified message queue is full and O_NONBLOCK is not set in the message queue description associated with *mqdes*, the wait for

sufficient room in the queue will be terminated when the specified timeout expires. If O_NONBLOCK is set in the message queue description, this system call is equivalent to **mq_send**().

The timeout will expire when the absolute time specified by *abs_timeout* passes, as measured by the clock on which timeouts are based (that is, when the value of that clock equals or exceeds *abs_timeout*), or if the absolute time specified by *abs_timeout* has already been passed at the time of the call.

The timeout is based on the CLOCK_REALTIME clock.

RETURN VALUES

Upon successful completion, the **mq_send**() and **mq_timedsend**() system calls return a value of zero. Otherwise, no message will be enqueued, the system calls return -1, and the global variable *errno* is set to indicate the error.

ERRORS

The **mq_send**() and **mq_timedsend**() system calls will fail if:

[EAGAIN]	The O_NONBLOCK flag is set in the message queue description associated with *mqdes*, and the specified message queue is full.
[EBADF]	The *mqdes* argument is not a valid message queue descriptor open for writing.
[EINTR]	A signal interrupted the call to **mq_send**() or **mq_timedsend**().
[EINVAL]	The value of *msg_prio* was outside the valid range.
[EINVAL]	The process or thread would have blocked, and the *abs_timeout* parameter specified a nanoseconds field value less than zero or greater than or equal to 1000 million.
[EMSGSIZE]	The specified message length, *msg_len*, exceeds the message size attribute of the message queue.
[ETIMEDOUT]	The O_NONBLOCK flag was not set when the message queue was opened, but the timeout expired before the message could be added to the queue.

SEE ALSO

mq_open(2), mq_receive(2), mq_setattr(2), mq_timedreceive(2)

STANDARDS

The **mq_send**() and **mq_timedsend**() system calls conform to IEEE Std 1003.1-2004 ("POSIX.1").

HISTORY

Support for POSIX message queues first appeared in FreeBSD 7.0.

COPYRIGHT