**NAME**

msgop, msgsnd, msgrcv – message operations

**SYNOPSIS**

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
int msgsnd(int msqid, const void *msgp, size_t msgsz, int msgflg);
int msgrcv(int msqid, void *msgp, size_t msgsz, long msgtyp, int msgflg);
```

**DESCRIPTION**

**msgsnd( )** sends a message to the queue associated with the message queue identifier specified by *msqid*. *msgp* points to a user defined buffer that must contain first a field of type long integer that will specify the type of the message, and then a data portion that will hold the text of the message. The following is an example of members that might be in a user defined buffer.

```
long mtype;   /* message type */
char mtext[]; /* message text */
```

**mtype** is a positive integer that can be used by the receiving process for message selection. **mtext** is any text of length *msgsz* bytes. *msgsz* can range from **0** to a system imposed maximum.

*msgflg* specifies the action to be taken if one or more of the following are true:

The number of bytes already on the queue is equal to **msg_qbytes** (see **intro**(2)).

The total number of messages on all queues system-wide is equal to the system-imposed limit.

These actions are as follows:

If (*msgflg*&**IPC_NOWAIT**) is true, the message is not sent and the calling process returns immediately.

If (*msgflg*&**IPC_NOWAIT**) is false, the calling process suspends execution until one of the following occurs:

- The condition responsible for the suspension no longer exists, in which case the message is sent.
- *msqid* is removed from the system (see **msgctl**(2)). When this occurs, **errno** is set to **EIDRM**, and a value of –1 is returned.
- The calling process receives a signal that is to be caught. In this case the message is not sent and the calling process resumes execution in the manner prescribed in **signal**(3C).

**msgrcv( )** reads a message from the queue associated with the message queue identifier specified by *msqid* and places it in the user defined structure pointed to by *msgp*. The structure must contain a message type field followed by the area for the message text (see the structure **mymsg** above). **mtype** is the received message's type as specified by the sending process. **mtext** is the text of the message. *msgsz* specifies the size in bytes of **mtext**.

The received message is truncated to *msgsz* bytes if it is larger than *msgsz* and (*msgflg*&**MSG_NOERROR**) is true. The truncated part of the message is lost and no indication of the truncation is given to the calling process.

*msgtyp* specifies the type of message requested as follows:

If *msgtyp* is 0, the first message on the queue is received.

If *msgtyp* is greater than 0, the first message of type *msgtyp* is received.

If *msgtyp* is less than 0, the first message of the lowest type that is less than or equal to the absolute value of *msgtyp* is received.

*msgflg* specifies the action to be taken if a message of the desired type is not on the queue. These are as follows:

If (*msgflg*&**IPC_NOWAIT**) is true, the calling process returns immediately with a return value of −1 and sets **errno** to **ENOMSG**.

If (*msgflg*&**IPC_NOWAIT**) is false, the calling process suspends execution until one of the following occurs:

- A message of the desired type is placed on the queue.
- *msqid* is removed from the system. When this occurs, **errno** is set to **EIDRM**, and a value of −1 is returned.
- The calling process receives a signal that is to be caught. In this case a message is not received and the calling process resumes execution in the manner prescribed in **signal**(3C).

## RETURN VALUES

If **msgsnd( )** or **msgrcv( )** return due to the receipt of a signal, a value of −1 is returned to the calling process and **errno** is set to **EINTR**. If they return due to removal of *msqid* from the system, a value of −1 is returned and **errno** is set to **EIDRM**.

Upon successful completion, the return value is as follows:

**msgsnd( )** returns a value of 0.

**msgrcv** returns the number of bytes actually placed into *mtext*.

Otherwise, a value of −1 is returned and **errno** is set to indicate the error.

## ERRORS

**msgsnd( )** fails and sends no message if one or more of the following are true:

**EACCES** Operation permission is denied to the calling process (see **intro**(2)).

**EAGAIN** The message cannot be sent for one of the reasons cited above and (*msgflg*&**IPC_NOWAIT**) is true.

**EFAULT** *msgp* points to an illegal address.

**EINVAL** *msqid* is not a valid message queue identifier.

**EINVAL** *mtype* is less than 1.

**EINVAL** *msgsz* is less than zero or greater than the system-imposed limit.

Upon successful completion, the following actions are taken with respect to the data structure associated with *msqid* (see **intro**(2)).

**msg_qnum** is incremented by 1.

**msg_lspid** is set to the process ID of the calling process.

**msg_stime** is set to the current time.

**msgrcv( )** fails and receives no message if one or more of the following are true:

**E2BIG** The length of *mtext* is greater than *msgsz* and (*msgflg*&**MSG_NOERROR**) is false.

**EACCES** Operation permission is denied to the calling process.

**EFAULT** *msgp* points to an illegal address.

**EINVAL** *msqid* is not a valid message queue identifier.

**EINVAL** *msgsz* is less than 0.

**ENOMSG** The queue does not contain a message of the desired type and (*msgtyp*&**IPC_NOWAIT**) is true.

Upon successful completion, the following actions are taken with respect to the data structure associated with **msqid** (see **intro**(2)).

**msg_qnum** is decremented by 1.

**msg_lrpid** is set to the process ID of the calling process.

**msg_rtime** is set to the current time.

## SEE ALSO

**intro**(2), **msgctl**(2), **msgget**(2), **signal**(3C)