

IOQUEUEM(a)

IOQUEUEM(a)

NAME

ioqueuem – send an I/O message

SYNOPSIS

(ioqueuem = 50.)
ioqueuem(msgbuf)
int *msgbuf; /* pointer to message */

DESCRIPTION

Ioqueuem sends an I/O message from the current process to a kernel device driver process. The message to be sent starts at *msgbuf* and must be 16 words long. A message consists of a six word header defined by the following structure:

```
struct msghdr {
    int  *mslink;          /* pointer to next input message */
    int  msfrom;          /* sending process number */
    int  msto;            /* receiving process number */
    char mssize;         /* message size in words */
    char mstype;         /* message type */
    int  msident;        /* message identifier */
    char msstat;         /* message status word */
    char msseqnum;       /* message sequence number */
};
```

and the sender's data. The sender need only fill in the *msto*, *mstype* and *mssize* fields of the message. The *mssize* word is the size of the sender's message in words exclusive of the header. The *mstype* byte may be any one of IOREAD, IOWRITE, IOOPEN or IOCLOSE and the message data should be in standard I/O format. The message body must contain:

```
int  msiosid;          /* segment ID */
int  msioba;           /* word offset */
int  msiobc;           /* I/O byte count */
char msiodev;          /* logical device number */
char msiob0;           /* 8-bit high order block number */
char *msiob1;          /* 16-bit low order block number */
```

A negative word offset indicates offset is from top of segment. Upon return word 1 of the message body will contain the 16 least significant bits of the physical address. The user status byte in *msstat* of the message contains the extension bits. This routine also increments the segment lock count and sets the *iolock* bit in the *mssize* byte so that the segment will be unlocked by *mssink* if the message type is IOREAD or IOWRITE. The sender may fill in *msident* in order to identify a particular acknowledgement message, as this word is never modified during the life of this message. The message is verified and copied into a kernel address space message buffer area. Here the *msfrom* word is filled in by the kernel as well as the message sequence number. The message is put on the input queue of messages for the *msto* process using the *mslink* word. A programmed interrupt request is enabled by sending a message event to the *msto* process. The message sequence number *msseqnum* is used only for debugging purposes. The *msstat* byte is filled in by the receiver of this message in its acknowledgement to this message. It contains the error code if non-zero. The value of -1 is reserved by the system for the case where the intended receiver process does not exist or is aborted abnormally.

If the input message queue for the receiver is overloaded or no message buffers exist in the kernel message buffer pool area, an error is passed back to the library routine which then road-blocks the process. Normally a value of 1 is returned from C.

IOQUEUEM(a)

IOQUEUEM(a)

SEE ALSO

sendmsgfrom(a), getmsg(a), gettype(a), sendmsg(a).

DIAGNOSTICS

If the message is too big, a bad EMT is indicated by a fault code of 10.