INTRODUCTION TO I/O PROCESS MESSAGES

Most block and record device I/O driver processes are programmed to accept messages of the following types:

IOREAD	1
IOWRITE	2
IOOPEN	3
IOCLOSE	4

However some I/O device drivers may not be programmed to accept IOOPEN and IOCLOSE messages, as they may be unnecessary for these devices. When the I/O device driver process is built using *ldp(e)* one of the specifications is whether the process will accept open and close messages. The format of a message to the device drivers is specified by the following structure:

```
struct io_msgd {
        struct   msghdr io_dhr; /* message 6-word header */
        int      msiosid;       /* ID of segment into which or from which I/O is to take place
        int      msioba;        /* word offset into segment */
        int      msiobc;        /* number of bytes to be read
        char     msiodev;       /* logical device number */
        char     msiob0;        /* high order byte of block number */
        char     *msiob1;       /* low order word of block number */
        char     msiopri;       /* priority of I/O */
        char     msiotry;       /* number of retries on error */
        int      msiocnt;       /* number of bytes transferred */
        int      msfill1;       /* scratch word, may be used by driver */
        int      msfill2;       /* scratch word, may be used by driver */
        int      msfill3;       /* scratch word, may be used by driver */
}
```

The IOOPEN and IOCLOSE messages need only specify the logical device number of the particular device driver and the number of times that the device driver file is open. This information is normally only available to the file manager process; the file manager thus will automatically send the open and close messages to the device drivers when the special files are opened and closed.

The IOREAD and IOWRITE messages must specify a total of 5 arguments besides the normal parameters in the message header as discussed in the introduction. These are the ID of the segment into or from which the I/O transfer is to take place, the word offset into this segment, the number of bytes to be transferred, the logical device number of the driver and the block number at which the transfer is to start. The segment may be either in supervisor or user address space, but it must be locked for I/O by the sender of the message before the message is sent. The word offset into the segment is from tne beginning of the segment if it is positive, otherwise it is from the end of the segment (e.g. for stack segment). Normally the file manager will make a call to the *ioqueuem* routine to determine if the transfer is within the bounds of the segment and if the segment is locked in memory. *Ioqueuem* then sends message to the appropriate driver. This routine will convert the virtual address given by *msiosid* and *msioba* into a 22-bit physical address. The lower 16 bits are returned in *msioba* and the upper 2 bits are set in the *msustat* byte of the message status byte. The total number of bytes read or written are returned in *msiocnt.* By invoking the *messink* routine in the device driver when the I/O transfer is complete, the segment is unlocked.

SEE ALSO

ioqueuem(b), open(c), close(c).

DIAGNOSTICS

An error status byte is returned if the segment is not locked for I/O, the segment does not exist, the I/O transfer would extend beyond the end of the segment or if a physical I/O error occurs in the transfer.