NAME

putctl – send a control message to a queue

SYNOPSIS

#include <sys/stream.h>

int putctl(queue_t **q*, **int** *type***);**

INTERFACE LEVEL

Architecture independent level 1 (DDI/DKI).

ARGUMENTS

q Queue to which the message is to be sent.

type Message type (must be control, not data type).

DESCRIPTION

putctl( ) tests the *type* argument to make sure a data type has not been specified, and then attempts to allocate a message block. **putctl** fails if *type* is **M_DATA**, **M_PROTO**, or **M_PCPROTO**, or if a message block cannot be allocated. If successful, **putctl( )** calls the **put**(9E) routine of the queue pointed to by *q* with the newly allocated and initialized messages.

RETURN VALUES

On success, **1** is returned. If *type* is a data type, or if a message block cannot be allocated, **0** is returned.

CONTEXT

putctl( ) can be called from user or interrupt context.

EXAMPLE

The **send_ctl** routine is used to pass control messages downstream. **M_BREAK** messages are handled with **putctl**( ) (line 11). **putctl1**(9F) (line 16) is used for **M_DELAY** messages, so that *parm* can be used to specify the length of the delay. In either case, if a message block cannot be allocated a variable recording the number of allocation failures is incremented (lines 12, 17). If an invalid message type is detected, **cmn_err**(9F) panics the system (line 21).

```
 1  void
 2  send_ctl(wrq, type, parm)
 3     queue_t *wrq;
 4     unchar type;
 5     unchar parm;
 6  {
 7          extern int num_alloc_fail;
 8
 9          switch (type) {
10          case M_BREAK:
11                  if (!putctl(wrq->q_next, M_BREAK))
12                          num_alloc_fail++;
13                  break;
14
15          case M_DELAY:
16                  if (!putctl1(wrq->q_next, M_DELAY, parm))
17                          num_alloc_fail++;
```

```
                break;

        default:
                cmn_err(CE_PANIC, "send_ctl: bad message type passed");
                break;
        }
}
```

SEE ALSO

put(9E), **cmn_err**(9F), **datamsg**(9F), **putctl1**(9F), **putnextctl**(9F)

Writing Device Drivers
STREAMS Programming Guide