NAME

copyb – copy a message block

SYNOPSIS

#include <sys/stream.h>

mblk_t *copyb(mblk_t ***_bp_);**

INTERFACE LEVEL

Architecture independent level 1 (DDI/DKI).

ARGUMENTS

_bp_ Pointer to the message block from which data is copied.

DESCRIPTION

copyb( ) allocates a new message block, and copies into it the data from the block that _bp_ denotes. The new block will be at least as large as the block being copied. **copyb( )** uses the **b_rptr** and **b_wptr** members of _bp_ to determine how many bytes to copy.

RETURN VALUES

If successful, **copyb( )** returns a pointer to the newly allocated message block containing the copied data. Otherwise, it returns a **NULL** pointer.

CONTEXT

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

EXAMPLES

For each message in the list, test to see if the downstream queue is full with the **canputnext**(9F) function (line 21). If it is not full, use **copyb** to copy a header message block, and **dupmsg**(9F) to duplicate the data to be retransmitted. If either operation fails, reschedule a timeout at the next valid interval.

Update the new header block with the correct destination address (line 34), link the message to it (line 35), and send it downstream (line 36). At the end of the list, reschedule this routine.

```
 1 struct retrans {
 2         mblk_t          *r_mp;
 3         long            r_address;
 4         queue_t         *r_outq;
 5         struct retrans  *r_next;
 6 };
 7
 8 struct protoheader {
      . . .
 9      long            h_address;
      . . .
10 };
11
12 mblk_t *header;
13
14 void
15 retransmit(struct retrans *ret)
16 {
17         mblk_t *bp, *mp;
```

```
        struct protoheader *php;

        while (ret) {
            if (!canputnext(ret->r_outq)) {             /* no room */
                    ret = ret->r_next;
                    continue;
            }
            bp = copyb(header);                         /* copy header msg. block */
            if (bp == NULL)
                    break;
            mp = dupmsg(ret->r_mp);                     /* duplicate data */
            if (mp == NULL) {                           /* if unsuccessful */
                    freeb(bp);                          /* free the block */
                    break;
            }
            php = (struct protoheader *)bp->b_rptr;
            php->h_address = ret->r_address;            /* new header */
            bp->bp_cont = mp;                           /* link the message */
            putnext(ret->r_outq, bp);                   /* send downstream */
            ret = ret->r_next;
        }
        /* reschedule */
        (void) timeout(retransmit, (caddr_t)ret, RETRANS_TIME);
}
```

SEE ALSO

allocb(9F), **canputnext**(9F), **dupmsg**(9F)

Writing Device Drivers
STREAMS Programming Guide