NAME

esballoc – allocate a message block using a caller-supplied buffer

SYNOPSIS

#include <sys/stream.h>

mblk_t *esballoc(unsigned char **base,* **int** *size,* **int** *pri,* **frtn_t** **fr_rtnp***);**

INTERFACE LEVEL

Architecture independent level 1 (DDI/DKI).

ARGUMENTS

base	Address of user supplied data buffer.
size	Number of bytes in data buffer.
pri	Priority of allocation request (to be used by **allocb**(9F) function, called by **esballoc( )**).
fr_rtnp	Free routine data structure.

DESCRIPTION

esballoc( ) creates a STREAMS message and attaches a user-supplied data buffer in place of a STREAMS data buffer. It calls **allocb**(9F) to get a message and data block header only. The user-supplied data buffer, pointed to by *base*, is used as the data buffer for the message.

When **freeb**(9F) is called to free the message, the driver's message freeing routine (referenced through the **free_rtn** structure) is called, with appropriate arguments, to free the data buffer.

The **free_rtn** structure includes the following members:

```
void (*free_func)();        /* user's freeing routine */
char *free_arg;             /* arguments to free_func() */
```

Instead of requiring a specific number of arguments, the **free_arg** field is defined of type **char** *. This way, the driver can pass a pointer to a structure if more than one argument is needed.

The method by which **free_func** is called is implementation-specific. The module writer must not assume that free_func will or will not be called directly from STREAMS utility routines like **freeb**(9F) which free a message block.

free_func must not call another modules put procedure nor attempt to acquire a private module lock which may be held by another thread across a call to a STREAMS utility routine which could free a message block. Otherwise, the possibility for lock recursion and/or deadlock exists.

free_func must not access any dynamically allocated data structure that might no longer exist when it runs.

RETURN VALUES

On success, a pointer to the newly allocated message block is returned. On failure, **NULL** is returned.

CONTEXT

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

SEE ALSO

allocb(9F), **freeb**(9F), **datab**(9S), **free_rtn**(9S)

Writing Device Drivers
STREAMS Programming Guide

WARNINGS

The **free_func** must be defined in kernel space, should be declared **void** and accept one argument. It has no user context and must not sleep.