initForNetworkDevice:name:unit:type:
maxTransferUnit:flags:
finishInitialization

Passing packets from the driver up to the protocol stack
handleInputPacket:extra:

Outputting a packet outputPacket:address:

Performing a command performCommand:data:

Allocating a network buffer allocateNetbuf

Keeping statistics collisions
incrementCollisions
incrementCollisionsBy:
incrementInputErrors
incrementInputErrorsBy:
incrementInputPackets
incrementInputPacketsBy:
incrementOutputErrors
incrementOutputErrorsBy:
incrementOutputPackets
incrementOutputPacketsBy:
inputErrors
inputPackets
outputErrors
outputPackets

(netbuf_t)allocateNetbuf

This method creates and returns a netbuf to be used for an impending output.

This method doesn't always have to return a buffer. For example, you might want to limit the number of buffers your driver instance can allocate (say, 200 kilobytes worth) so that it won't use too much wired-down kernel memory. When this method fails to return a buffer, it should return NULL.

Here's an example of implementing allocateNetbuf.

(unsigned int)collisions

Returns the total number of network packet collisions that have been detected since boot time.

(int)handleInputPacket:(netbuf_t)packet extra:(void *)extra

Increments the number of input packets and passes packet to the kernel for processing. The kernel
the appropriate protocol handler, as described <<only in the OS book, for now>>.

A network device driver should invoke this method after it's processed a newly received packet. T
be zero, unless the protocol handler requires another value. For instance, token ring drivers need to
to a token ring header. This method returns EAFNOSUPPORT if no protocol handler accepts the p
returns zero.

(void)incrementCollisions

Increments by one the total number of network packet collisions that have been detected since boo

(void)incrementCollisionsBy:(unsigned int)increment

Increments by increment the total number of network packet collisions that have been detected sin

(void)incrementInputErrors

Increments by one the total number of packet input errors that have been detected since boot time.

(void)incrementInputErrorsBy:(unsigned int)increment

Increments by increment the total number of packet input errors that have been detected since boot

Increments by increment the total number of packets that have been received by the computer sinc

(void)incrementOutputErrors

Increments by one the total number of packet output errors that have been detected since boot time

(void)incrementOutputErrorsBy:(unsigned int)increment

Increments by increment the total number of packet output errors that have been detected since boo

(void)incrementOutputPackets

Increments by one the total number of packets that have been transmitted by the computer since bo

(void)incrementOutputPacketsBy:(unsigned int)increment

Increments by increment the total number of packets that have been transmitted by the computer si

initForNetworkDevice:device
name:(const char *)name
unit:(unsigned int)unit
type:(const char *)type
maxTransferUnit:(unsigned int)mtu
flags:(unsigned int)flags

Initializes and returns the IONetwork instance associated with the specified direct device driver de
connects device into the kernel's networking subsystem. It's typically called from a network driver
initFromDeviceDescription. You shouldn't invoke initForNetworkDevice:... directly. IOEthernet a
invoke this method on behalf of their subclasses and return an IONetwork object in their respective
attachToNetworkWithAddress: methods.

The name argument should be set to a constant string that names this type of network device. For e
drivers are named ªenº, and Token Ring drivers are named ªtrº. The unit is an integer greater than
unique for name. For example, the first instance of an Ethernet driver is unit 0, the second is unit 1

The type is a constant string that describes this module. For example, Ethernet drivers supply the c
IFTYPE_ETHERNET (which is defined in net/etherdefs.h to be ª10MB Ethernetº).

The mtu is the maximum amount of data your module can send or receive. For example, Ethernet d

(unsigned int)inputPackets

Returns the total number of packets that have been received by the computer since boot time.

(unsigned int)outputErrors

Returns the total number of packet output errors that have been detected since boot time.

(int)outputPacket:(netbuf_t)packet address:(void *)address

This method should deliver the specified packet to the given address. Its return value should be zer
otherwise, return an error number from the header file sys/errno.h.

If you implement this method, you need to check that [self isRunning] == YES. If so, insert the ne
addresses into the packet and check it for minimum length requirements.

(unsigned int)outputPackets

Returns the total number of packets that have been transmitted by the computer since boot time.

(int)performCommand:(const char *)command data:(void *)data

This method performs arbitrary control operations the character string command is used to select b
operations. Although you don't have to implement any operations, there are five standard operatio
your own operations.

The standard commands are listed in the following table. The constant strings listed below are dec
net/netif.h (under the bsd directory of /NextDeveloper/Headers).