	initFromDeviceDescription:
Getting and setting parameters	getIntValues:forParameter:count: setCharValues:forParameter:count: setIntValues:forParameter:count:
Handling the cursor	hideCursor: moveCursor:frame:token: showCursor:frame:token:
Setting screen brightness	setBrightness:token:
Setting the gamma correction table	setTransferTable:count:
Mapping the frame buffer	mapFrameBufferAtPhysicalAddress:length:
Choosing display modes	enterLinearMode revertToVGAMode

initFromDeviceDescription:

(void)enterLinearMode

Implemented by subclasses to put the display into linear frame buffer mode. This method is invoke
appropriate, such as when the window server starts running.

revertToVGAMode

(IOReturn)getIntValues:(unsigned int *)parameterArray
forParameter:(IOParameterName)parameterName
count:(unsigned int *)count

Handles NeXT-internal parameters specific to IOFrameBufferDisplays forwards the handling of al
super.

getIntValues:forParameter:count: (IODevice)

hideCursor:(int)token

Implements this method, as described in the IOScreenEvents protocol specification. You should ne
implement this method.

initFromDeviceDescription:deviceDescription

Invokes initFromDeviceDescription: on super. If successful, sets the unit number and the name (to
the unit number). Frees itself if initialization was unsuccessful.

Subclasses must implement this method so that it performs all initialization necessary to set up the
This includes setting the IODisplayInfo structure (as described in the IODisplay class description)
mapFrameBufferAtPhysicalAddress:length:. If possible, this method should also check the hardwa
the IOConfigTable. If the hardware doesn't match, the driver should do what it can to ensure that t
usable.

(vm_address_t)mapFrameBufferAtPhysicalAddress:(unsigned int)address length:(int)numByte

Maps the physical memory for this instance into virtual memory for use by the device driver. If ad
maps the physical memory corresponding to local memory range 0, and numBytes is ignored. If ad
reserved resources are overriddenÐ address is used as the physical memory address and numBytes
The mapped memory range is cached as specified in the IODisplayInfo for this instance.

token:(int)token

Implements this method, as described in the IOScreenEvents protocol specification. You should ne
implement this method.

(void)revertToVGAMode

Implemented by subclasses to remove the display from whatever advanced mode it's in and enter a
be used as a standard VGA device.

enterLinearMode

(int)selectMode:(const IODisplayInfo *)modeList count:(int)count

Invokes selectMode:count:valid:, specifying 0 for the last argument.

(int)selectMode:(const IODisplayInfo *)modeList
count:(int)count
valid:(const BOOL *)isValid

Determines which IODisplayInfo in the driver-supplied modeList matches the value of the ªDispla
device's IOConfigTable. Drivers that support multiple advanced modes should invoke this method
When the driver receives a enterLinearMode message, it should enter the mode selected by this me
doesn't find a valid mode, the driver should determine a mode that will work.

The ªDisplay Modeº key is a configuration key that can be used by drivers to support multiple mo
8-bit gray and 16-bit RGB. IODisplayInfo is defined in the header file driverkit/displayDefs.h.

The modeList argument should contain a IODisplayInfo for each advanced mode the driver suppor
should specify the number of IODisplayInfos in modeList. isValid should either be 0 (in which cas
array that corresponds to the modeList. If isValid[1] is NO, for example, then this method ignores
pointed to by modeList[1].

If this method finds a match, it returns the index of the matching IODisplayInfo in modeList. If the
is missing or its value is improperly formatted, or if a corresponding IODisplayInfo isn't found, thi

See the IODisplay class description for information on display modes and the IODisplayInfo type.

setBrightness:(int)level token:(int)token

Checks whether level is between EV_SCREEN_MIN_BRIGHTNESS and EV_SCREEN_MAX_B
(inclusive). If not, this method logs an error message. Subclasses that support brightness changes s
method and implement it as described in the IOScreenEvents protocol specification.

Returns self.

(IOReturn)setCharValues:(unsigned char *)parameterArray

(IOReturn)setIntValues:(unsigned int *)parameterArray
forParameter:(IOParameterName)parameterName
count:(unsigned int)count

Handles NeXT-internal parameters specific to IOFrameBufferDisplays forwards the handling of al
super.

setIntValues:forParameter:count: (IODevice)

setTransferTable:(const unsigned int *)table count:(int)numEntries

Specifies new gamma correction values to be used by the hardware. The default implement does n
Subclasses that support multiple gamma correction transforms must override this method so that it
reflect the values in table.

Gamma correction is necessary because displays respond nonlinearly to linear ranges of voltage. F
pixel that can have red, green, and blue values between 0 and 15. This pixel's brightness when the
might be more than 7/15 its brightness when the values are (15, 0, 0). Gamma correction lets the ha
voltage of the beamÐfor example, using 6.5/15 of maximum voltage instead of 7/15, so that the pi

Each entry in table specifies the gamma correction (a value scaled to be between 0 and 255, inclusi
corresponding pixel component values. For example, for RGB color modes, table[7] specifies the g
red value of 7, a green value of 7, and a blue value of 7 (using one byte of the entry per component
(0, 5, 15), for example, the hardware should use the red gamma correction from table[0], the green
from table[5], and the blue gamma correction from table[15]. Which bytes you use from each table
whether the transfer table is for a color or black-and-white mode you can determine the mode from
numEntries.

When numEntries is IO_2BPP_TRANSFER_TABLE_SIZE or IO_8BPP_TRANSFER_TABLE_S
header file driverkit/displayDefs.h), the table is for a black-and-white display. In this case, each tab
meaningful byte: the least significant byte.

When numEntries is IO_12BPP_TRANSFER_TABLE_SIZE, IO_15BPP_TRANSFER_TABLE_
IO_24BPP_TRANSFER_TABLE_SIZE, the table is for an RGB display, and each entry has three
most significant byte holds the red gamma correction, the next most significant byte holds the gree
and the next holds the blue gamma correction. The least significant byte holds no information.

The following example shows how to copy the correction information from the transfer table to a p
hardware.

showCursor:(Point *)cursorLoc