NAME

ddi_peek, ddi_peekc, ddi_peeks, ddi_peekl, ddi_peekd – read a value from a location

SYNOPSIS

#include <sys/ddi.h>
#include <sys/sunddi.h>

int ddi_peekc(dev_info_t **dip*, **char** **addr*, **char** **valuep***);**

int ddi_peeks(dev_info_t **dip*, **short** **addr*, **short** **valuep***);**

int ddi_peekl(dev_info_t **dip*, **long** **addr*, **long** **valuep***);**

int ddi_peekd(dev_info_t **dip*, **longlong_t** **addr*, **longlong_t** **valuep***);**

INTERFACE LEVEL

Solaris DDI specific (Solaris DDI).

ARGUMENTS

dip A pointer to the device's **dev_info** structure.

addr Virtual address of the location to be examined.

valuep Pointer to a location to hold the result. If a null pointer is specified, then the value read from the location will simply be discarded.

DESCRIPTION

These routines cautiously attempt to read a value from a specified virtual address, and return the value to the caller, using the parent nexus driver to assist in the process where necessary.

If the address is not valid, or the value cannot be read without an error occurring, an error code is returned.

The routines are most useful when first trying to establish the presence of a device on the system in a driver's **probe**(9E) or **attach**(9E) routines.

RETURN VALUES

DDI_SUCCESS The value at the given virtual address was successfully read, and if *valuep* is non-null, **valuep* will have been updated.

DDI_FAILURE An error occurred while trying to read the location. **valuep* is unchanged.

CONTEXT

These functions can be called from user or interrupt context.

EXAMPLES

Check to see that the status register of a device is mapped into the kernel address space:

```
if (ddi_peekc(dip, csr, (char *)0) != DDI_SUCCESS) {
        cmn_err(CE_WARN, "Status register not mapped");
        return (DDI_FAILURE);
}
```

Read and log the device type of a particular device:

```
int
xx_attach(dev_info_t *dip, ddi_attach_cmd_t cmd)
{
        ...
        /* map device registers */
        ...

        if (ddi_peekl(dip, id_addr, &id_value) != DDI_SUCCESS) {
                cmn_err(CE_WARN, "%s%d: cannot read device identifier",
                   ddi_get_name(dip), ddi_get_instance(dip));
                goto failure;
        } else
                cmn_err(CE_CONT, "!%s%d: device type 0x%x\n",
                   ddi_get_name(dip), ddi_get_instance(dip), id_value);
        ...
        ...

        ddi_report_dev(dip);
        return (DDI_SUCCESS);

failure:
        /* free any resources allocated */
        ...
        return (DDI_FAILURE);
}
```

SEE ALSO

attach(9E), **probe**(9E), **ddi_poke**(9F)

Writing Device Drivers