NAME

chpoll – poll entry point for a non-STREAMS character driver

SYNOPSIS

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

int *prefix***chpoll(dev_t** *dev***, short** *events***, int** *anyyet***, short** **reventsp***,**
struct pollhead ***phpp***);**

INTERFACE LEVEL

This entry point is *optional*.
Architecture independent level 1 (DDI/DKI).

ARGUMENTS

dev The device number for the device to be polled.

events The events that may occur. Valid events are:

POLLIN	Data other than high priority data may be read without blocking.
POLLOUT	Normal data may be written without blocking.
POLLPRI	High priority data may be received without blocking.
POLLHUP	A device hangup has occurred.
POLLERR	An error has occurred on the device.
POLLRDNORM	Normal data (priority band = 0) may be read without blocking.
POLLRDBAND	Data from a non-zero priority band may be read without blocking
POLLWRNORM	The same as **POLLOUT**.
POLLWRBAND	Priority data (priority band > 0) may be written.

anyyet A flag that is non-zero if any other file descriptors in the **pollfd** array have events pending. The **poll**(2) system call takes a pointer to an array of **pollfd** structures as one of its arguments. See the **poll**(2) reference page for more details.

reventsp A pointer to a bitmask of the returned events satisfied.

phpp A pointer to a pointer to a **pollhead** structure.

DESCRIPTION

The **chpoll( )** entry point routine is used by non-STREAMS character device drivers that wish to support polling. The driver must implement the polling discipline itself. The following rules must be followed when implementing the polling discipline:

1. Implement the following algorithm when the **chpoll( )** entry point is called:

```
if (events_are_satisfied_now) {
    *reventsp = mask_of_satisfied_events;
} else {
    *reventsp = 0;
    if (!anyyet)
        *phpp = &my_local_pollhead_structure;
}
return (0);
```

2. Allocate an instance of the **pollhead** structure. This instance may be tied to the per-minor data structure defined by the driver. The **pollhead** structure should be treated as a ‘‘black box’’ by the driver. None of its fields should be referenced. However, the size of this structure is guaranteed to remain the same across releases.
3. Call the **pollwakeup( )** function whenever an event of type **events** listed above occur. This function should only be called with one event at a time.

RETURN VALUES

chpoll( ) should return **0** for success, or the appropriate error number.

SEE ALSO

poll(2), **pollwakeup**(9F)

Writing Device Drivers

NOTES

Driver defined locks should not be held across calls to this function.