NAME

ioctl − control device

SYNOPSIS

```
#include <sys/ioctl.h>

int ioctl (fildes, request, argp)
int fildes, request;
struct *argp;
```

DESCRIPTION

Ioctl manipulates the file or device indicated by *fildes* as specified by *request*. The requests and the kinds of things they can access are:

TIOCGETD, TIOCSETD
Get/set line discipline. *Argp* points to a structure containing an integer with a valid line discipline indicator integer.

TIOCHPCL Hang up on last close. *Argp* indicates whether this feature should be turned on or off.

TIOCSETO, TIOCGETO
Get/set "other" bits. *Argp* contains a word with bits indicating which "other" bits are to be set/reset or interrogated. This request is essentially an extension of the old *stty/gtty* system call that allows transmission/response to xon/xoff, half duplex line, no-hangup, excluding future device opens, no sleeping if not ready, and non-standard tty escapes and kills.

TIOCGETP, TIOCSETP
These are equivalent to **gtty(fildes, argp)** and **stty(fildes, argp)**. They allow terminal (tty) characteristics to be set and examined. These include terminal input and output speed, the erase character and kill character, and mode flags. The allowed mode flags include hangup on last close, map tabs to spaces, upper case only, character echo, cr/lf mode, raw character input, parity, and delay on tabs, new lines, backspace, carriage return, and vt delay. Note that setting input speed to zero on a dh or dz line will disable the line by dropping the Data Terminal Ready(DTR) bit for the line.

TIOCSETN Equivalent to old *stty* with *noflush*.

TIOCEXCL, TIOCNXCL
Get/clear the exclude bit, which disallows future opens on the device.

TIOCTSTP Stop toggle transmit.

DIOCGETT, DIOCSETT
Get/set terminal parameters. These include terminal type, current cursor row and column (get only), variable row, last row, and terminal flags. The flags include special newline, auto newline on column 80, last column of last row special, echo of terminal cursor control, and not sending escape sequences to the user. It is used primarily for CRT terminals.

DIOCSETS Set spy mode. All output directed to the terminal specified by *fildes* will be copied to the terminal of the process performing the *ioctl*. Only one spy operation may be active in the entire system at any time. The spy continues until explicitly turned off. Currently, spy is only effective on lines using the **STD_LTYPE** line discipline and is restricted to the super-user.

FIOCLEX, FIONCLEX

Set/clear auto close for a file. If auto close is set, then the file will not be passed to children across an *exec*.

FIOSPIPE, FIOGPIPE

Get/set pipe sleep flags. This enables/disables sleeping on reads/writes to a pipe, to avoid roadblocking. Normally, reads are blocking and writes are not.

VIOCGETD, VIOCSETD

Get/set versatec parameters.

There are also requests for the multiplexor (see *mpx*(2), *mpxio*(5) and **<sys/mx.h>**). In general, each line discipline has a unique header file which defines the line discipline number and format of the structure to be used with **DIOCGETP** and **DIOCSETP** requests.

The proper names for all these flags and other requests not currently used are contained in **<sys/ioctl.h>**, which is included here:

```
/*              @(#)ioctl.h     3.5             */
/*
 * structure of arg for ioctl TIOCSETP and TIOCGETP
 */
struct          ttiocb {
                char            ioc_ispeed;
                char            ioc_ospeed;
                char            ioc_erase;
                char            ioc_kill;
                short           ioc_flags;
};

/*
 * structure for old stty and gtty system calls.
 */
struct          sgttyb          {
                char            sg_ispeed;      /* input speed */
                char            sg_ospeed;      /* output speed */
                char            sg_erase;       /* erase character */
                char            sg_kill;        /* kill character */
                short           sg_flags;       /* mode flags */
};

/*
 * tty ioctl commands
 */
#define         TIOCGETD        (('t'<<8)|0)    /* get line discipline */
#define         TIOCSETD        (('t'<<8)|1)    /* set line discipline */
#define         TIOCHPCL        (('t'<<8)|2)    /* hangup on last close */
#define         TIOCMODG        (('t'<<8)|3)
#define         TIOCMODS        (('t'<<8)|4)
#define         TIOCSETO        (('t'<<8)|6)    /* set other bits */
#define         TIOCGETO        (('t'<<8)|7)    /* get other bits */
#define         TIOCGETP        (('t'<<8)|8)    /* gtty */
#define         TIOCSETP        (('t'<<8)|9)    /* stty */
#define         TIOCSETN        (('t'<<8)|10)   /* stty - no flush */
#define         TIOCEXCL        (('t'<<8)|13)   /* set exclude */
#define         TIOCNXCL        (('t'<<8)|14)   /* clr exclude */
#define         TIOCHMOD        (('t'<<8)|15)
#define         TIOCTSTP        (('t'<<8)|16)   /* toggle transmit stop */
#define         DIOCGETP        (('d'<<8)|8)    /* get discipline parameters */
#define         DIOCSETP        (('d'<<8)|9)    /* set discipline parameters */
#define         DIOCSETT        (('d'<<8)|10)   /* set terminal info */
```

```
#define     DIOCGETT    (('d'<<8)|11)   /* get terminal info */
#define     DIOCSETS    (('d'<<8)|12)   /* set spy mode */
#define     FIOCLEX                     (('f'<<8)|1)    /* set auto close */
#define     FIONCLEX    (('f'<<8)|2)    /* clr autoclose */
#define     FIOSPIPE    (('p'<<8)|1)    /* set pipe sleep flags */
#define     FIOGPIPE    (('p'<<8)|2)    /* get pipe sleep flags */
#define     VIOCGETD    (('v'<<8)|0)    /* Versatec */
#define     VIOCSETD    (('v'<<8)|1)    /* Versatec */

/*
 * Define standard line discipline for TIOCSETD and TIOCGETD
 */
#define     STD_LTYPE   (short)0
/*
 * Define half duplex line discipline for TIOCSETD and TIOCGETD
 */
#define HF_LTYPE        (short)4
/*
 * Format of third argument for TIOCSETD and TIOCGETD
 */
struct sgldisc {
            short       sgl_type;
};


/*
 * Following ioctl.h commands are used within the system only.
 */
#ifdef KERNEL
#define     OLDSGTTY    (('i'<<8)|1)
#define     GETRFP                  (('i'<<8)|2)
#define     GETWFP                  (('i'<<8)|3)
#endif

/*
 * Modes
 */
#define     HUPCL       01          /* hangup on last close */
#define     XTABS       02          /* map tabs to spaces on output */
#define     LCASE       04          /* upper case only terminal */
#define     ECHO        010         /* echo all received characters */
#define     CRMOD       020         /* map CR->LF;echo CR or LF as CR-LF */
#define     RAW         040         /* raw character input */
#define     ODDP        0100        /* odd parity rcvd/xmtd */
#define     EVENP       0200        /* even parity rcvd/xmtd */
#define     ANYP        0300        /* any parity mask */
#define     NLDELAY     001400
#define     TBDELAY     002000
#define     CRDELAY     030000
#define     VTDELAY     040000
#define     BSDELAY     0100000
#define     ALLDELAY    0173400

/*
 * Delay algorithms
 */
#define     CR0         0
#define     CR1         010000
#define     CR2         020000
#define     CR3         030000
#define     NL0         0
#define     NL1         000400
#define     NL2         001000
```

```
#define     NL3         001400
#define     TAB0        0
#define     TAB1        002000
#define     NOAL        004000
#define     FF0         0
#define     FF1         040000
#define     BS0         0
#define     BS1         0100000

/*
 * Speeds
 */
#define B0      0
#define B50     1
#define B75     2
#define B110    3
#define B134    4
#define B150    5
#define B200    6
#define B300    7
#define B600    8
#define B1200   9
#define B1800   10
#define B2400   11
#define B4800   12
#define B9600   13
#define EXTA    14
#define EXTB    15

/*
 * Character length and stop bits.
 * Character length does not include parity or stop bits.
 * Ored with ioc_ospeeed.
 */
#define     SETSTOP     0200        /* set to change stop or length bits */
#define     ONESTOP     0000
#define     TWOSTOP     0100        /* 1.5 stop bits at 75 baud */
#define     BITS5       0000
#define     BITS6       0020
#define     BITS7       0040
#define     BITS8       0060
#define     SLBITS      0160        /* Mask of stop and length bits */


/*
 * structure of arg for ioctl TIOCSETO and TIOCGETO
 */
struct ttiothcb {
            short       ioth_flags;
};

/*
 * Definition of "other" bits
 */
#define     TANDEMO     01          /* enable transmission of xon/xoff */
#define     HDPLX       0400        /* Half duplex line */
#define     NOHUP       01000       /* not dial device flag */
#define     XCLUDE      02000       /* disallow future opens */
#define     NOSLEEP     04000       /* dont sleep if nothing is ready */
#define     TANDEMI     040000      /* enable response to xon/xoff */
#define     STDTTY      0100000     /* non-standard tty escapes and kills */
```

```
/*
 * struct of arg for ioctl FIOSPIPE and FIOGPIPE
 */
struct          pipcb           {
                char            pip_rflg;       /* read flag; 0=>nosleep */
                char            pip_wflg;       /* write flag; 0=>nosleep */
};


/*
 * structure of ioctl arg for DIOCGETT and DIOCSETT
 */
struct          termcb          {
                char            st_flgs;        /* term flags */
                char            st_termt;       /* term type */
                char            st_crow;        /* gtty only - current row */
                char            st_ccol;        /* gtty only - current col */
                char            st_vrow;        /* variable row */
                char            st_lrow;        /* last row */
};

/*
 * Terminal types
 */
#define         TERM_NONE       0               /* tty */
#define         TERM_TEC        1               /* TEC Scope */
#define         TERM_V61        2               /* DEC VT61 */
#define         TERM_V10        3               /* DEC VT100 */
#define         TERM_TEX        4               /* Tektronix 4023 */
#define         TERM_D40        5               /* TTY Mod 40/1 */
#define         TERM_H45        6               /* Hewlitt-Packard 45 */
#define         TERM_D42        7               /* TTY Mod 40/2B */
#define TERM_C100               8               /* Concept 100*/

/*
 * Terminal flags
 */
#define TM_NONE                                 0000            /* use default flags */
#define TM_SNL                                  0001            /* special newline flag */
#define TM_ANL                                  0002            /* auto newline on column 80 */
#define TM_LCF                                  0004            /* last col of last row special */
#define TM_CECHO        0010            /* echo terminal cursor control */
#define TM_CINVIS       0020            /* do not send esc seq to user */
#define TM_SET          0200            /* must be on to set/res flags */
```

Several of the modes and flags require further explanation:

LCASE Map upper case to lower case on input; map lower case to upper case on output. Map | to !; ` to '; { to (; } to); ~ to ^; \<C> to upper case input, where <C> is any upper case character.

RAW In raw mode, every character is immediately passed to the program without waiting for a full line to be typed. No input characters have a special meaning (e.g., the interrupt character DEL will not cause the program to be interrupted, but will be passed to the program as a character.). **LCASE** and **CRMOD** will still cause input mapping; output character processing is unaffected. If the transmitter has been stopped by the ESC key, setting **RAW** will release it. Note, however, that this can only be effective if the **TIOCSETP** command is utilized. Otherwise, the program will wait for the ESC key to be depressed again. Input and output data width is eight bits, but the eigth bit may be a parity bit depending upon the setting of **ODDP** and **EVENP**.

ODDP, EVENP
For the standard line discipline, a character will be rejected unless its parity matches that expected. If both bits are set, either parity is accepted and even parity is transmitted. If both bits are set and **RAW** is set, the parity is visible to and supplied by the user on input and output. If neither bit is set, no parity is expected and even parity is transmitted.

HDPLX For those communications controllers with the capability, disable reception during transmission.

XCLUDE When set, no one may open the line. Cleared upon the last close.

NOSLEEP
Return a zero if a read is performed and no characters are present. Don't wait to flush output on *close* or *ioctl*. Don't wait for carrier on the first *read* or *write* after an *open*, if carrier is not up. Normally, a process will block when waiting for carrier to come up after an *open*. This roadblock will take place in the first *read* or *write*, not the *open*.

STDTTY Change the erase character from # to _ and the delete line character from @ to $. In addition to CR and LF, wake up on / and !, and generate an interrupt upon reception of & or DEL.

TANDEMO
When set, transmission of xon/xoff is enabled. This turns off the keyboard when there are too many characters in the terminal hardware queue.

TANDEMI
When set, response to xon/xoff is enabled.

NOHUP Indicates that the line is not a dial-up line, and, therefore, will not hang up when the terminal session is completed.

DELAY For certain line speeds, a delay is desired for certain functions. Delay can be specified for CR, LF, tabs, backspaces, and formfeeds.

It is also possible for the user to set the number of data and stop bits, if the defaults are not satisfactory. The default is **TWOSTOP** at speeds B75 and B110, **ONESTOP** otherwise; **BITS5** for B75, with **BITS7** plus one bit even parity otherwise. These bits are or'd in with the *ioc_ospeed* flag. The **SETSTOP** bit must be set to change stop or length bits.

Normally, an **TIOCSETP** request will wait for output to be flushed before doing anything. This can be circumvented by using the **TIOCSETN** request.

The normal CB-UNIX line discipline is **STD_LTYPE**. Request **TIOCSETD** can be used to set the discipline to the commonly-supported half-duplex line discipline **HF_LTYPE**, and the transparent line discipline **TRANS_LTYPE**, described in <sys/**trans.h**>. Different line disciplines expect different values for certain modes. However, **STD_LTYPE** and **HF_LTYPE** require no additional information.

TRANS_LTYPE is a line discipline that allows the user full eight bit transparency on input and output with or without parity. For this line discipline, a *write* will perform no mapping. A *read* will return upon the occurrence of the first of the three conditions as specified by the user:

1) The requested number of characters have arrived.
2) The number of seconds, *ts_quanta*, has elapsed.
3) A break character has arrived.

If *ts_quanta* is zero, timing is disabled; otherwise, *ts_quanta* is the maximum wait time in seconds. If *ts_brk0* and *ts_brk1* are both zero, no break characters will awaken the process. If

ts_brk1 is 0377 then *ts_bbrk0* is taken as a single break character. Otherwise, both break characters are assumed valid. **NCDELAY, XTABS, LCASE, ECHO, CRMOD, RAW,** and **STDTTY** have no meaning for this line discipline.

The **DIOCSETT** request is used to specify the type of CRT connected to a line. **TERM_NONE** is the standard, non-CRT type. If a type other than **TERM_NONE** is specified, input and output mapping will occur for the CRT language defined in the header file **<sys/crtctl.h>**. In this case, the ESC character takes on special meaning, escaping the subsequent characters on input and output. The terminal flags *st_flgs* and modes are given a default set of values when the terminal type is set. The modes may be subsequently changed with a **DIOCSETT** request. The flags may be changed by setting the **TM_SET** bit when changing the terminal type and specifying the flag bits. The flag bits require further clarification:

TM_SNL Handle new lines specially, if the terminal driver is so equipped.

TM_ANL Provide a carriage return and new line when writing beyond column 80.

TM_LCF Immediately before placing a character in the last column and row, delete the top line, print the character in the last column of the now second-to-last row, and then move the cursor to column one of the new last line. This function is requires for terminals that move the cursor to "bad" places when printing in the last position.

TM_CECHO
Echo the control sequences, such as "cursor up", when received.

TM_CINVIS
Do not pass the cursor control characters to the user program on input.

SEE ALSO

/usr/include/sys/sgtty.h
/usr/include/sys/mx.h
/usr/include/sys/trans.h
stty:o(2), fcntl(2)

ASSEMBLER

(ioctl = 54.)
(filedes in r0)
sys ioctl; request; argp