NAME

termios, tcgetattr, tcsetattr, tcsendbreak, tcdrain, tcflush, tcflow, cfgetospeed, cfgetispeed, cfsetispeed, cfsetospeed, tcgetpgrp, tcsetpgrp, tcgetsid – general terminal interface

SYNOPSIS

#include <termios.h>

int tcgetattr(int *fildes*, **struct termios** **termios_p***);**

int tcsetattr(int *fildes*, **int** *optional_actions*, **const struct termios** **termios_p***);**

int tcsendbreak(int *fildes*, **int** *duration***);**

int tcdrain(int *fildes***);**

int tcflush(int *fildes*, **int** *queue_selector***);**

int tcflow(int *fildes*, **int** *action***);**

speed_t cfgetospeed(const struct termios **termios_p***);**

int cfsetospeed(struct termios **termios_p*, **speed_t** *speed***);**

speed_t cfgetispeed(const struct termios **termios_p***);**

int cfsetispeed(struct termios **termios_p*, **speed_t** *speed***);**

#include <sys/types.h>
#include <termios.h>

pid_t tcgetpgrp(int *fildes***);**

int tcsetpgrp(int *fildes*, **pid_t** *pgid***);**

pid_t tcgetsid(int *fildes***);**

MT-LEVEL

MT-Safe

tcgetattr( ), **tcsetattr( )**, **tcsendbreak( )**, **tcdrain( )**, **tcflush( )**, **tcflow( )**, **cfgetospeed( )**, **cfgetispeed( )**, **cfsetispeed( )**, **cfsetospeed( )**, **tcgetpgrp( )**, and **tcsetpgrp( )** are Async-Signal-Safe

DESCRIPTION

These functions describe a general terminal interface for controlling asynchronous communications ports. A more detailed overview of the terminal interface can be found in **termio**(7I), which also describes an **ioctl**(2) interface that provides the same functionality. However, the function interface described here is the preferred user interface.

Many of the functions described here have a *termios_p* argument that is a pointer to a **ter-**

mios structure. This structure contains the following members:

```
tcflag_t    c_iflag;      /* input modes */
tcflag_t    c_oflag;      /* output modes */
tcflag_t    c_cflag;      /* control modes */
tcflag_t    c_lflag;      /* local modes */
cc_t        c_cc[NCCS];   /* control chars */
```

These structure members are described in detail in **termio**(7I).

Get and Set Terminal Attributes

The **tcgetattr( )** function gets the parameters associated with the object referred by *fildes* and stores them in the **termios** structure referenced by *termios_p*. This function may be invoked from a background process; however, the terminal attributes may be subsequently changed by a foreground process.

The **tcsetattr( )** function sets the parameters associated with the terminal (unless support is required from the underlying hardware that is not available) from the **termios** structure referenced by *termios_p* as follows:

If *optional_actions* is **TCSANOW**, the change occurs immediately.

If *optional_actions* is **TCSADRAIN**, the change occurs after all output written to *fildes* has been transmitted. This function should be used when changing parameters that affect output.

If *optional_actions* is **TCSAFLUSH**, the change occurs after all output written to the object referred by *fildes* has been transmitted, and all input that has been received but not read is discarded before the change is made.

The symbolic constants for the values of *optional_actions* are defined in **<termios.h>**.

Line Control

If the terminal is using asynchronous serial data transmission, the **tcsendbreak( )** function causes transmission of a continuous stream of zero-valued bits for a specific duration. If *duration* is zero, it causes transmission of zero-valued bits for at least 0.25 seconds, and not more than 0.5 seconds. If *duration* is not zero, it behaves in a way similar to **tcdrain( )**.

If the terminal is not using asynchronous serial data transmission, the **tcsendbreak( )** function sends data to generate a break condition or returns without taking any action.

The **tcdrain( )** function waits until all output written to the object referred to by *fildes* has been transmitted.

The **tcflush( )** function discards data written to the object referred to by *fildes* but not transmitted, or data received but not read, depending on the value of *queue_selector*:

If *queue_selector* is **TCIFLUSH**, it flushes data received but not read.

If *queue_selector* is **TCOFLUSH**, it flushes data written but not transmitted.

If *queue_selector* is **TCIOFLUSH**, it flushes both data received but not read, and data written but not transmitted.

The **tcflow( )** function suspends transmission or reception of data on the object referred to by *fildes*, depending on the value of *action*:

If *action* is **TCOOFF**, it suspends output.

If *action* is **TCOON**, it restarts suspended output.

If *action* if **TCIOFF**, the system transmits a STOP character, which causes the terminal device to stop transmitting data to the system.

If *action* is **TCION**, the system transmits a START character, which causes the terminal device to start transmitting data to the system.

Get and Set Baud Rate

The baud rate functions get and set the values of the input and output baud rates in the **termios** structure. The effects on the terminal device described below do not become effective until the **tcsetattr( )** function is successfully called.

The input and output baud rates are stored in the **termios** structure. The values shown in the table are supported. The names in this table are defined in **<termios.h>**.

Name	Description	Name	Description
B0	Hang up	**B4800**	4800 baud
B50	50 baud	**B9600**	9600 baud
B75	75 baud	**B19200**	19200 baud
B110	110 baud	**B38400**	38400 baud
B134	134.5 baud	**B57600**	57600 baud
B150	150 baud	**B76800**	76800 baud
B200	200 baud	**B115200**	115200 baud
B300	300 baud	**B153600**	153600 baud
B600	600 baud	**B230400**	230400 baud
B1200	1200 baud	**B307200**	307200 baud
B1800	1800 baud	**B460800**	460800 baud
B24000	24000 baud		

cfgetospeed( ) returns the output baud rate stored in the **termios** structure pointed to by *termios_p.*

cfsetospeed( ) sets the output baud rate stored in the **termios** structure pointed to by *termios_p* to *speed*. The zero baud rate, **B0**, is used to terminate the connection. If **B0** is specified, the modem control lines are no longer be asserted. Normally, this disconnects the line.

cfgetispeed( ) returns the input baud rate stored in the **termios** structure pointed to by *termios_p*.

cfsetispeed( ) sets the input baud rate stored in the **termios** structure pointed to by *termios_p* to *speed*. If the input baud rate is set to zero, the input baud rate is specified by the value of the output baud rate. Both **cfsetispeed( )** and **cfsetospeed( )** return a value of zero if successful and **−1** to indicate an error. This refers both to changes to baud rates not supported by the hardware, and to changes setting the input and output baud rates to different values if the hardware does not support this.

Get and Set Terminal Foreground Process Group ID

tcsetpgrp( ) sets the foreground process group ID of the terminal specified by *fildes* to *pgid*. The file associated with *fildes* must be the controlling terminal of the calling process and the controlling terminal must be currently associated with the session of the calling process. *pgid* must match a process group ID of a process in the same session as the calling process.

tcgetpgrp( ) returns the foreground process group ID of the terminal specified by *fildes*. **tcgetpgrp( )** is allowed from a process that is a member of a background process group; however, the information may be subsequently changed by a process that is a member of a foreground process group.

Get Terminal Session ID

tcgetsid( ) returns the session ID of the terminal specified by *fildes*.

RETURN VALUES

On success, **tcgetpgrp( )** returns the process group ID of the foreground process group associated with the specified terminal. Otherwise, it returns −**1** and sets **errno** to indicate the error.

On success, **tcgetsid( )** returns the session ID associated with the specified terminal. Otherwise, it returns −**1** and sets **errno** to indicate the error.

On success, all other functions return a value of **0**. Otherwise, they return −**1** and set **errno** to indicate the error.

ERRORS

All of the functions fail if one of more of the following is true:

EBADF The *fildes* argument is not a valid file descriptor.

ENOTTY The file associated with *fildes* is not a terminal.

tcsetattr( ) also fails if the following is true:

EINVAL The *optional_actions* argument is not a proper value, or an attempt was made to change an attribute represented in the **termios** structure to an unsupported value.

tcsendbreak( ) also fails if the following is true:

EINVAL The device does not support the **tcsendbreak( )** function.

tcdrain( ) also fails if one or more of the following is true:

EINTR A signal interrupted the **tcdrain( )** function.

EINVAL The device does not support the **tcdrain( )** function.

tcflush( ) also fails if the following is true:

EINVAL The device does not support the **tcflush( )** function or the *queue_selector* argument is not a proper value.

tcflow( ) also fails if the following is true:

EINVAL The device does not support the **tcflow( )** function or the *action* argument is not a proper value.

tcgetpgrp( ) also fails if the following is true:

ENOTTY the calling process does not have a controlling terminal, or *fildes* does not refer to the controlling terminal.

tcsetpgrp( ) also fails if the following is true:

EINVAL *pgid* is not a valid process group ID .

ENOTTY the calling process does not have a controlling terminal, or *fildes* does not refer to the controlling terminal, or the controlling terminal is no longer associated with the session of the calling process.

EPERM *pgid* does not match the process group of an existing process in the same session as the calling process.

tcgetsid( ) also fails if the following is true:

EACCES *fildes* is a terminal that is not allocated to a session.

SEE ALSO

setpgid(2), **getsid**(2), **termio**(7I)