【Linux Programming】Day21

Tags	
Date	@June 14, 2022
Summary	The termios Struc

【Ch5】Terminals

5.2 Talking to the Terminals

Linux provides a special device, `/dev/tty`, which is always the current terminal.

See `menu3.c` in the Code folder for code.

5.3 The Terminal Driver and the General Terminal Interface

Linux provides a set of interfaces(the General Terminal Interface, or GTI) that allow us to control the behavior of the terminal driver.

The main features that we can control are:

- Line editing: Choose whether to allow Backspace for editing
- Buffering: Choose whether to read characters immediately, or read them after a configurable delay
- Echo: Allows us to control echoing, such as when reading passwords
- CR/LF: Determine mapping for input and output: what happens when we print a line feed character

5.4 The termios Structure

`termios` is the standard interface specified by POSIX. The terminal interface is controlled by setting values in a structure of type `termios` and using a small set of function calls.

The values that can be manipulated to affect the terminal are grouped into various modes: Input, Output, Control, Local, Special Control Characters

A minimum `termios` is typically declared as below:

```
#include <termios.h>

struct termios {
  tcflag_t c_iflag;
  tcflag_t c_oflag;
  tcflag_t c_cflag;
  tcflag_t c_lflag;
  cc_t c_cc[NCCS};
};
```

We can initialize a `termios` structure by calling the function `tcgetattr`, which has the following prototype:

```
#include <termios.h>

int tcgetattr(int fd, struct termios *termios_p);
```

This system call writes the current values of the terminal interface variables into the structure pointed by `termios_p`.

We can reconfigure the terminal interface with the `tcsetattr` function:

```
#include <termios.h>

int tcsetattr(int fd, int actions, const struct termios *termios_p);
```

The actions field control how any changes are applied. The three possibilities are:

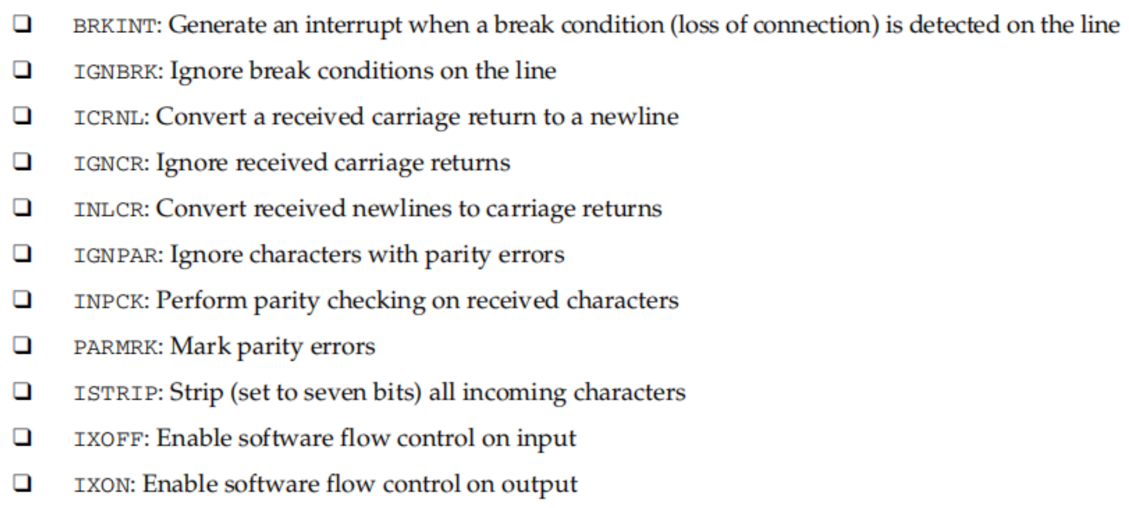
- `TCSANOW`: Changes values immediately
- `TCSADRAIN`: Changes values when current output is complete

- `TCSAFLUSH`: Changes values when current output is complete, but discards any input currently available and not yet returned in a read call.

5.4.1 Input Modes

The input modes control how input(characters received by the terminal driver) is processed before being passed on to the program.

We can control them by setting flags in the `c_iflag` member of the `termios` structure. All the flags are macros and can be combined with a bitwise OR.



❑ BRKINT: Generate an interrupt when a break condition (loss of connection) is detected on the line
❑ IGNBRK: Ignore break conditions on the line
❑ ICRNL: Convert a received carriage return to a newline
❑ IGNCR: Ignore received carriage returns
❑ INLCR: Convert received newlines to carriage returns
❑ IGNPAR: Ignore characters with parity errors
❑ INPCK: Perform parity checking on received characters
❑ PARMRK: Mark parity errors
❑ ISTRIP: Strip (set to seven bits) all incoming characters
❑ IXOFF: Enable software flow control on input
❑ IXON: Enable software flow control on output

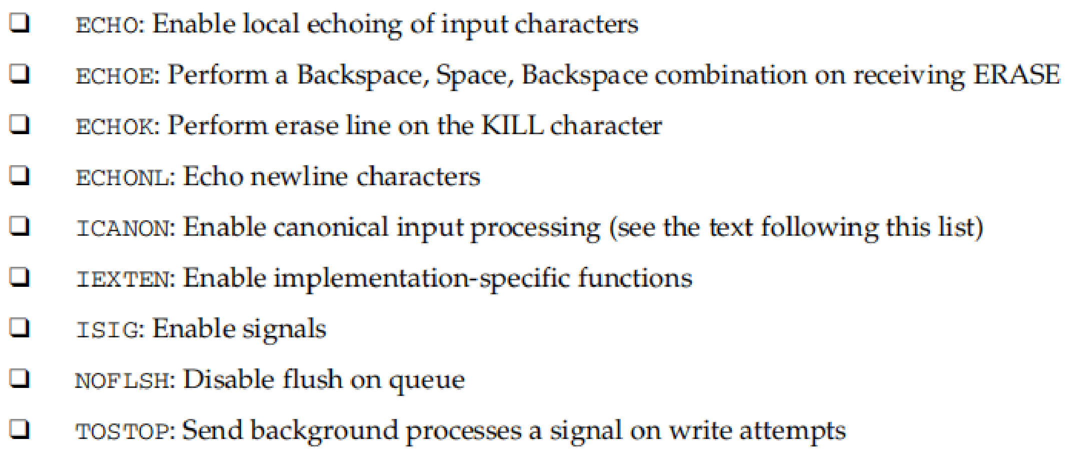
5.4.2 Output Modes

- OPOST: Turn on output processing
- ONLCR: Convert any output newline to a carriage return/line feed pair
- OCRNL: Convert any output carriage return to a newline
- ONOCR: No carriage return output in column 0
- ONLRET: A newline also does a carriage return
- OFILL: Send fill characters to provide delays
- OFDEL: Use DEL as a fill character, rather than NULL
- NLDLY: Newline delay selection
- CRDLY: Carriage return delay selection
- TABDLY: Tab delay selection
- BSDLY: Backspace delay selection
- VTDLY: Vertical tab delay selection
- FFDLY: Form feed delay selection

5.4.3 Control Modes

- CLOCAL: Ignore any modem status lines.
- CREAD: Enable the receipt of characters.
- CS5: Use five bits in sent or received characters.
- CS6: Use six bits in sent or received characters.
- CS7: Use seven bits in sent or received characters.
- CS8: Use eight bits in sent or received characters.
- CSTOPB: Use two stop bits per character, rather than one.
- HUPCL: Hang up modem on close.
- PARENB: Enable parity generation and detection.
- PARODD: Use odd parity rather than even parity.

5.4.4 Local Modes



- ECHO: Enable local echoing of input characters
- ECHOE: Perform a Backspace, Space, Backspace combination on receiving ERASE
- ECHOK: Perform erase line on the KILL character
- ECHONL: Echo newline characters
- ICANON: Enable canonical input processing (see the text following this list)
- IEXTEN: Enable implementation-specific functions
- ISIG: Enable signals
- NOFLSH: Disable flush on queue
- TOSTOP: Send background processes a signal on write attempts

`ECHO` is commonly used, which allows us to suppress the echoing of typed characters.

`ICANON` is also commonly used, which switches the terminal between canonical and non-canonical modes. When `ICANON` flag is set, the line is said to be in canonical mode; if not, the line is in non-canonical mode.

5.4.5 Special Control Characters

Special control characters are a collection of characters, like Ctrl+C, acted upon in particular ways when the user types them.

The `c_cc` array member of the `termios` structure contains the characters mapped to each of the supported functions. The position of each characters(its index into the array) is defined by a macro.

The array indices differ when the terminal is in canonical and non-canonical mode

For canonical mode, the array indices are

- VEOF: EOF character
- VEOL: EOL character
- VERASE: ERASE character
- VINTR: INTR character
- VKILL: KILL character

38

- VQUIT: QUIT character
- VSUSP: SUSP character
- VSTART: START character
- VSTOP: STOP character

For non-canonical mode, the array indices are

- VINTR: INTR character
- VMIN: MIN value
- VQUIT: QUIT character
- VSUSP: SUSP character
- VTIME: TIME value
- VSTART: START character
- VSTOP: STOP character