NAME

picocom − minimal dumb−terminal emulation program

SYNOPSIS

picocom [*options*] *device*

DESCRIPTION

As its name suggests, **picocom(1)** is a minimal dumb−terminal emulation program. It is, in principle, very much like **minicom(1)**, only it's "pico" instead of "mini"! It was designed to serve as a simple, manual, modem configuration, testing, and debugging tool. It has also served (quite well) as a low−tech serial communications program to allow access to all types of devices that provide serial consoles. It could also prove useful in many other similar tasks.

When picocom starts it opens the terminal (serial device) given as its non−option argument. Unless the **−−noinit** option is given, it configures the device to the settings specified by the option−arguments (or to some default settings), and sets it to "raw" mode. If **−−noinit** is given, the initialization and configuration is skipped; the device is just opened. Following this, picocom sets the standard−input and standard−output to raw mode. Having done so, it goes in a loop where it listens for input from stdin, or from the serial port. Input from the serial port is copied to the standard output while input from the standard input is copied to the serial port. Picocom also scans its input stream for a user−specified control character, called the *escape character* (being by default **C−a**). If the escape character is seen, then instead of sending it to the serial−device, the program enters "command mode" and waits for the next character (which is called the "function character"). Depending on the value of the function character, picocom performs one of the operations described in the **COMMANDS** section below.

COMMANDS

Commands are given to picocom by first keying the *espace character* which by default is **C−a** (see **OPTIONS** below for how to change it), and then keying one of the function (command) characters shown here.

escape character
: Send the escape character to the serial port and return to "transparent" mode. This means that if the escape character (**C−a**, by default) is typed twice, the program sends the escape character to the serial port, and remains in transparent mode.

C−x Exit the program: if the **−−noreset** option was not given then the serial port is reset to its original settings before exiting; if it was given the serial port is not reset.

C−q Quit the program *without* reseting the serial port, regardless of the **−−noreset** option.

C−p Pulse the DTR line. Lower it for 1 sec, and then raise it again.

C−t Toggle the DTR line. If DTR is up, then lower it. If it is down, then raise it.

C−backslash
: Generate a break sequence on the serial line. A break sequence is usually generated by marking (driving to logical one) the serial Tx line for an amount of time coresponding to several character durations.

C−u Baud up. Increase the baud−rate. The list of baud−rates stepped−through by this command is: 50, 75, 110, 134, 150, 200, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. If `HIGH_BAUD` support is compiled−in, then the following baud−rates are also added to the list: 230400, 460800, 500000, 576000, 921600, 1000000, 1152000, 1500000, 2000000, 2500000, 3000000, 3500000, 4000000. Depending on you system, any of the higher baud rates may be missing.

C−d Baud down. Decrease the baud−rate. The list of baud−rates stepped−through by this command is the same as for the "baud−up" command.

C−f Cycle through flow−control settings (RTS/CTS, XON/XOFF, none).

C−y Cycle through parity settings (even, odd, none).

C−b Cycle through databits−number settings (5, 6, 7, 8).

C−j Cycle through stopbits−number settings (1, 2).

C−c Toggle local−echo mode.

C−v Show program options (like baud rate, data bits, etc) as well as the actual serial port settings. Only the options and port settings that can be modified online (through commands) are shown, not those that can only be set at the command−line. See **DISPLAY OF OPTIONS AND PORT SETTINGS** for details.

C−h or C−k
Show help or show keys. Prints a short description of all available function (command) keys.

C−s Send (upload) a file. See **SENDING AND RECEIVING FILES** below.

C−r Receive (download) a file. See **SENDING AND RECEIVING FILES** below.

After performing one of the above operations, the program leaves the command mode and enters transparent mode. Example: To increase the baud−rate by two steps, you have to type:

C−a, **C−u**, **C−a**, **C−u**

assuming of−course that **C−a** is the escape character.

OPTIONS

Picocom accepts the following command−line options.

−−baud | −b
Defines the baud−rate to set the serial−port (terminal) to.

−−flow | −f
Defines the flow−control mode to set the serial−port to. Must be one of: **x** for xon/xoff (software) mode, **h** for hardware flow control (RTS/CTS), **n** for no flow control. (Default: **n**)

−−parity | −y
Defines the parity mode to set the serial−port to. Must be one of: **o** for odd parity mode, **e** for even parity mode, **n** for no parity mode. (Default: **n**)

−−databits | −d
Defines the number of data bits in every character. Must be one of:**5**, **6**, **7**, **8**. (Def ault: **8**)

−−stopbits | −p
Defines the number of stop bits in every character. Must be one of:**1**, or **2**. (Def ault: **1**)

−−esacpe | −e
Defines the character that will make picocom enter command−mode (see description above). If**x** is given, then **C−x** will make picocom enter command mode. (Default: **a**)

−−echo | −c
Enable local echo. Every character being read from the terminal (standard input) is echoed to the terminal (standard output) subject to the echo−mapping configuration (see **−−emap** option). (Default: Disabled)

−−noinit | −i
If given, picocom will not initialize, reset, or otherwise meddle with the serial port at start−up. It will just open it. This is useful, for example, for connecting picocom to already−connected modems, or already configured ports without terminating the connection, or altering the settings. If required, serial port parameters can then be adjusted at run−time by commands. (Default: Disabled)

−−noreset | −r
If given, picocom will not reset the serial port when exiting. It will just close the filedes and do nothing more. This is useful, for example, for leaving modems connected when exiting picocom. Regardless whether the **−−noreset** option is given, the user can exit picocom using the "Quit" command (instead of "Exit"), which never resets the serial port. If **−−noreset** is given then "Quit"

and "Exit" behave essentially the same. (Default: Disabled)

−−nolock | −l
: If given, picocom will *not* attempt to lock the serial port before opening it. Normally, depending on how it's compiled, picocom attempts to get a UUCP−style lock−file (e.g. '/var/lock/LCK..ttyS0') before opening the port, or attempts to lock the port device−node using **flock(2)**. Failing to do so, results in the program exiting after emitting an error−message. It is possible that your picocom binary is compiled without support for locking. In this case the **−−nolock** option is accepted, but has no effect. (Default: Disabled)

−−send−cmd | −s
: Specifies the external program (and any arguments to it) that will be used for transmitting files. If the argument to **−−send−cmd** is the empty string (''), the send−file command is disabled. See **SENDING AND RECEIVING FILES**. (Default: **sz −vv**)

−−receive−cmd | −v
: Specifies the external program (and any arguments to it) that will be used for receiving files. If the argument to **−−receive−cmd** is the empty string (''), the receive−file command is disabled. See **SENDING AND RECEIVING FILES**. (Default: **rz −vv**)

−−imap
: Specifies the input character map (i.e. special characters to be replaced when read from the serial port). See**INPUT , OUTPUT, AND ECHO MAPPING**. (Defaul: Empty)

−−omap
: Specifies the output character map (i.e. special characters to be replaced before being written to serial port). See **INPUT, OUTPUT, AND ECHO MAPPING**. (Defaul: Empty)

−−emap
: Specifies the local−echo character map (i.e. special characters to be replaced before being echoed−back to the terminal, if local−echo is enabled). See **INPUT, OUTPUT, AND ECHO MAPPING**. (Defaul: **delbs,crcrlf**)

−−help | −h
: Print a short help message describing the command−line options. Picocom's version, ompile−time options, and enabled features are also shown.

DISPLAY OF OPTIONS AND PORT SETTINGS

The "show program options" command (**C−v**), as well as the commands that change program options (**C−u**, **C−d**, **C−f**, etc) print messages showing the current values (or the new values, if they were changed) for the respective options. If picocom determines that an actual serial−port setting differs from the current value of the respective option (for whatever reason), then the value of the option is shown followed by the value of the actual serial−port setting in parenthesis. Example:

```
*** baud: 115200 (9600)
```

This means that a baud rate of 115200bps has been selected (from the command line, or using commands that change the baudrate) but the serial−port is actually operating at 9600bps (the driver may not support the higher setting, and has silently replaced it with a safe default, or the setting may have been changed from outside picocom). If the option and the corresponding serial−port setting are the same, only a single value is shown. Example:

```
*** baud: 9600
```

This behavioir was intriduced in picocom 2.0. Older releases displayed only the option values, not the actual serial−port settings corresponding to them.

SENDING AND RECEIVING FILES

Picocom can send and receive files over the serial port using external programs that implement the respective protocols. In Linux typical programs for this purpose are:

- **rx(1)** − receive using the X−MODEM protocol

- **rb(1)** − receive using the Y−MODEM protocol
- **rz(1)** − receive using the Z−MODEM protocol
- **sx(1)** − send using the X−MODEM protocol
- **sb(1)** − send using the Y−MODEM protocol
- **sz(1)** − send using the Z−MODEM protocol
- **ascii−xfr(1)** − receive or transmit ASCII files

The name of, and the command−line options to, the program to be used for transmitting files are given by the **−−send−cmd** option. Similarly the program to receive files, and its argumets, are given by the **−−receive−cmd** option. For example, in order to start a picocom session that uses **sz(1)** to transmit files, and **rz(1)** to receive files, you have to say something like this:

```
picocom --send-cmd "sz -vv" --receive-cmd "rz -vv" ...
```

If the argument to the **−send−cmd** option, or the argument to the **−−receive−cmd** option is the empty string, then the respective command is disabled. For example, in order to disable both the "send" and the "receive" commands you can invoke picocom like this:

```
picocom --send-cmd '' --receive-cmd '' ...
```

A picocom session with both, the send− and the receive−file commands disabled does not **fork(2)** and does not run any external programs.

During the picocom session, if you key the "send" or "receive" commands (e.g. by pressing **C−a**, **C−s**, or **C−a**, **C−r**) you will be prompted for a filename. At this prompt you can enter one or more file−names, and any additional arguments to the transmission or reception program. Command−line editing and rudimentary pathname completion are available at this prompt, if you have compiled picocom with support for the linenoise library. Pressing**C−c** at this prompt will cancel the file transfer command and return to normal picocom operation. After entering a filename (and / or additional transmission or reception program arguments) and assuming you have not canceled the operation by pressing **C−c**, picocom will start the the external program as specified by the **−−send−cmd**, or **−−receive−cmd** option, and with any filenames and additional arguments you may have supplied. The standard input and output of the external program will be connected to the serial port. The standard error of the external program will be connected to the terminal which−−−while the program is running−−−will revert to canonical mode. Pressing **C−c** while the external program is running will prematurely terminate it (assuming that the program itself does not ignore SIGINT), and return control to picocom. Pressing **C−c** at any other time, has no special effect; the character is normally passed to the serial port.

INPUT, OUTPUT, AND ECHO MAPPING

Using the **−−imap**, **−−omap**, and **−−emap** options you can make picocom map (tranlate, replace) certain special characters after being read from the serial port (with **−−imap**), before being written to the serial port (with **−−omap**), and before being locally echoed to the terminal (standard output) if local echo is enabled (with **−−emap**). These mapping options take, each, a single argument which is a comma−separated list of one or more of the following identifiers:

- **crlf** (map CR to LF),
- **crcrlf** (map CR to CR + LF),
- **igncr** (ignore CR),
- **lfcr** (map LF to CR),
- **lfcrlf** (map LF to CR + LF),
- **ignlf** (ignore LF),
- **bsdel** (map BS to DEL),
- **delbs** (map DEL to BS)

For example the command:

```
picocom --omap crlf,delbs --imap inglf,bsdel --emap crcrlf ...
```

will:

- Replace every CR (carriage return, 0x0d) caracter with LF (line feed, 0x0a) and every DEL (delete, 0x7f) character with BS (backspace, 0x08) before writing it to the serial port.
- Ignore (not write to the terminal) every LF character read from the serial port, and replace every BS character read from the serial port with DEL.
- Replace every CR character with CR and LF when echoing to the terminal (if local−echo is enabled).

AUTHOR

Written by Nick Patavalis <npat@efault.net>

AVAILABILITY

Download the latest release from: <https://github.com/npat-efault/picocom/releases>

COPYRIGHT