CHAPTER 7

Changing Terminal Parameters

This chapter explains how to change terminal and line settings locally. The local settings temporarily override those made by the system administrator, remaining in effect only until you exit the system.

Refer to the following sections for descriptions of local changes to the terminal and line settings:

- Display the Commands that Set Local Terminal Parameters
- Display Help for All User-Level Commands
- Specify the Terminal Type
- Change the Terminal Screen Length
- Change the Terminal Screen Width
- Change the Terminal Escape Character
- Change the Terminal Hold Character
- Specify a Keyboard Type
- Change the Terminal Hold Character
- Change the Terminal Parity Bit
- Change the Terminal Line Speed
- Change the Data Bits
- Change the Stop Bits
- Set Terminal Flow Control
- Change the Start Character
- Change the Stop Character
- Set Character Padding
- Set the Packet Dispatch Character
- Establish Pending Output Notification
- Select File Download Mode
- Select the Preferred Terminal Transport Protocol
- Set the Number of Data Bits Per Character
- Locally Specify the International Character Set
- Set the Character Dispatch Timer

- Change ASCII Character Widths
- Set the Terminal Receive Speed
- Set the Terminal Transmit Speed
- Generate a Hardware Break Signal for a Reverse Telnet Connection
- Set the Line to Refuse Full-Duplex, Remote Echo Connections
- Allow Transmission Speed Negotiation
- Synchronize the BREAK Signal
- Change the End-of-Line Character
- Display Debug Messages on the Console and Terminals

You can perform all but the last task at the user-level EXEC prompt. You display system debugging messages at the privileged-level EXEC prompt. (For more information about privileged-level EXEC mode, refer to the configuration guide or command reference publication for your server product.)

Display the Commands that Set Local Terminal Parameters

To see a list of the commands for setting terminal parameters, use the **terminal ?** command:

terminal ?

Display Help for All User-Level Commands

To get help for the full set of user-level commands, use the **terminal full-help** EXEC command. The **terminal full-help** command enables (or disables) a display of all help messages available from the terminal. It is used with the **show** command. This command has the following syntax:

terminal full-help

Example

The following example is output for **show?** with **terminal full-help** enabled.

```
cs> terminal full-help
cs> show?
access expression List access expression
access lists List access lists
apollo Apollo network information
appletalk AppleTalk information
arp ARP table
async Information on terminal lines used as communication server interfaces
...
```

Specify the Terminal Type

To specify the type of terminal connected to the current line, use the **terminal terminal-type** command. Indicate the terminal type if it is different from the default of VT100. This name is used by TN3270 for display management, and by Telnet and rlogin to inform the remote host of the terminal type. This command has the following syntax:

terminal terminal-type *terminal-type*

Syntax Description

terminal-type	Defines the terminal name and type and permits terminal negotiation by hosts that provide that type of service. The default is VT100.

Example

The following example defines the terminal on line 7 as a VT220:

```
cs> terminal terminal-type VT220
```

Change the Terminal Screen Length

Use the **terminal length** *screen-length* command to set the number of lines on the current terminal screen. The screen length specified can be learned by remote hosts. For example, the rlogin protocol uses the screen length to set up terminal parameters on a remote UNIX host. The command has the following syntax:

terminal length *screen-length*

Syntax Description

screen-length	Your desired number of lines on the screen. The remote access server uses this value to determine when to pause during multiple-screen output. The default length is 24 lines. A value of zero prevents the server from pausing between screens of output. When the output exceeds the screen length, it scrolls past.

Example

The following example prevents the remote access server from pausing between multiple screens of output:

```
cs> terminal length 0
```

Change the Terminal Screen Width

By default, the server product provides a screen display width of 80 characters. You can reset this value if it does not meet the needs of your terminal. The width specified can be learned by remote hosts. To set the number of character columns on the terminal screen for the current line, use the **terminal width** EXEC command. This command has the following syntax:

terminal width *characters*

Syntax Description

characters	Number of character columns displayed on the terminal.

Example

The following example sets the terminal character columns to 132:

```
terminal width 132
```

Change the Terminal Escape Character

Use the **terminal escape-character** command to set the escape character for the current terminal line. This is useful, for example, if you have the default escape character defined for a different purpose in your keyboard file. Entering the escape character followed by the X key returns you to EXEC mode when you are connected to another computer. The default escape characters are Ctrl-^ (Ctrl-Shift-6). The command has the following syntax:

terminal escape-character *ASCII-number*

Syntax Description

ASCII-number Either the ASCII decimal representation of the desired escape character or a control sequence (Ctrl-P, for example). The default is Ctrl-^. Typing the escape character followed by the X key returns you to the EXEC when you are connected to another computer. See the "ASCII Character Set" appendix later in this publication, for a list of ASCII characters.

Note The Break key cannot be used as an escape character on the console terminal because the operating software interprets BREAK as an instruction to halt the system.

Example

The following example sets the escape character to Ctrl-P (ASCII decimal 16):

```
cs> terminal escape-character 16
```

Specify a Keyboard Type

Use the **terminal keymap-type** command to specify the current keyboard type. This action is necessary when you are using a keyboard other than the default of VT100. The system administrator can define other keyboard types and give you their names. This command has the following syntax:

terminal keymap-type *keymap-name*

Syntax Description

keymap-name The name defining the current keyboard type. The default is VT100.

Example

The following example specifies a VT220 keyboard as the current keyboard type:

```
cs> terminal keymap-type vt220
```

Change the Terminal Hold Character

You can define a local hold character that temporarily suspends the flow of output on the terminal. When information is scrolling by too fast, you can type the hold character to pause the screen output, then type any other character to resume the flow of output.

Use the **terminal hold-character** command to set or change the hold character. Use the **terminal no hold-character** command to delete the hold character. You cannot suspend output on the console terminal. To send the hold character to the host, precede it with the escape character. This command has the following syntax:

terminal hold-character *ASCII-number*

Syntax Description

ASCII-number — Either the ASCII decimal representation of the hold character or a control sequence (for example, Ctrl-P). By default, no local hold character is set. The Break character is represented by zero; NULL cannot be represented.

Example

The following example removes the previously set hold character:

```
cs> terminal no hold-character
```

Change the Terminal Parity Bit

Use the **terminal parity** command to define the generation of the parity bit for the current terminal line. Communication protocols provided by devices such as terminals and modems often require a specific parity bit setting. The default is no parity. This command has the following syntax:

terminal parity {**none** | **even** | **odd** | **space** | **mark**}

Syntax Description

none — No parity. This is the default.

even — Even parity.

odd — Odd parity.

space — Space.

mark — Mark.

Example

The following example shows how to set the parity bit to odd:

```
cs> terminal parity odd
```

Change the Terminal Line Speed

To set the transmit and receive speeds of the current terminal line, use the **terminal speed** command. The default speed is 9600 bits per second (bps). This command has the following syntax:

terminal speed *bps*

Syntax Description

bps The baud rate in bits per second (bps). The default is 9600 bits per second.

Example

The following example sets the current auxiliary line transmit and receive speed to 2400 bps.

```
cs> terminal speed 2400
```

Change the Data Bits

To change the number of data bits per character for the current terminal line, use the **terminal databits** command. Communication protocols provided by devices such as terminals and modems often require a specific data bit setting. The default is 8 data bits per character. You can change to 5, 6, or 7 (or back to 8). This command has the following syntax:

terminal databits {**5** | **6** | **7** | **8**}

Syntax Description

5 Five data bits per character.

6 Six data bits per character.

7 Seven data bits per character.

8 Eight data bits per character. This is the default.

The **terminal databits** command can be used to mask the high bit on input from devices that generate 7 data bits with parity. If parity is being generated, specify 7 data bits per character. If no parity generation is in effect, specify 8 data bits per character. The other keywords (5 and 6) are supplied for compatibility with older devices and are generally not used.

Example

The following example shows how to change the databits per character to seven:

```
cs> terminal databits 7
```

Change the Stop Bits

To change the number of stop bits transmitted per byte by the current terminal line, use the **terminal stopbits** command. Communication protocols provided by devices such as terminals and modems often require a specific stopbit setting. This command has the following syntax:

terminal stopbits {**1** | **1.5** | **2**}

Syntax Description

1 One stop bit.

1.5 One and a half stop bits.

2 Two stop bits. This is the default.

Example

The following example illustrates how to change the stop bits to one:

```
cs> terminal stopbits 1
```

Set Terminal Flow Control

Flow control enables you to regulate the rate at which data can be transmitted from one point so that it is equal to the rate at which it can be received at another point. Flow control protects against loss of data because the terminal is not capable of receiving data at the rate it is being sent. You can set up data flow control for the current terminal line in one of two ways: software flow control, which you do with control key sequences, and hardware flow control, which you do at the device level. To set flow control for the current terminal line, use the **terminal flowcontrol** command. This command has the following syntax:

terminal flowcontrol {**none** | **software** [**in** | **out**] | **hardware**}

Syntax Description

none Prevents flow control.

software Sets software flow control.

[**in** | **out**] (Optional) Specifies the direction: **in** causes the remote access server to listen to flow control from the attached device, and **out** causes the remote access server to send flow control information to the attached device. If you do not specify a direction, both directions are assumed.

hardware Sets hardware flow control. For information about setting up the RS-232 line, see the hardware manual for your product.

By default, no flow control method is set. This default is returned with the **none** keyword. For software flow control, the default stop and start characters are Ctrl-S and Ctrl-Q (XOFF and XON). You can change them with the **terminal stop-character** and **terminal start-character** commands.

Example

The following example sets incoming software flow control:

```
cs> terminal flowcontrol software in
```

Change the Start Character

This character signals the start of data transmission when software flow control is in effect. You can change the flow control start character with the **terminal start-character** command. This command has the following syntax:

terminal start-character *ASCII-number*

Syntax Description

ASCII-number The ASCII decimal representation of the start character. The default is Ctrl-Q (ASCII decimal character 17).

Example

The following example changes the start character to Ctrl-O (ASCII decimal character 15):

```
cs> terminal start-character 15
```

Change the Stop Character

This character signals the end of data transmission when software flow control is in effect. You can change the flow control stop character using the **terminal stop-character** command. This command has the following syntax:

terminal stop-character *ASCII-number*

Syntax Description

ASCII-number The ASCII decimal representation of the stop character. The default is Ctrl-S (ASCII character 19).

Example

The following example changes the stop character to Ctrl-E, which is ASCII character 5.

```
cs> terminal stop-character 5
```

Set Character Padding

Character padding adds a number of null bytes to the end of the string and can be used to make a string an expected length for conformity. You can change the character padding on a specific output character using the **terminal padding** command. This command has the following syntax:

terminal padding *ASCII-number count*

Syntax Description

ASCII-number	The ASCII decimal representation of the character.
count	The number of NULL bytes sent after that character, up to 255 padding characters in length.

Example

The following example pads Ctrl-D (ASCII decimal character 4) with 164 NULL bytes:

```
cs> terminal padding 4 164
```

Set the Packet Dispatch Character

At times, you might want to queue up a string of characters until they fill a complete packet and then transmit the packet to a remote host. This can make more efficient use of a line because the remote access server normally dispatches each character as it is typed. You can define a character that causes a packet to be sent with the **terminal dispatch-character** command. This command has the following syntax:

terminal dispatch-character *ASCII-number1 [ASCII-number2 . . . ASCII-number*]

Syntax Description

ASCII-number	The ASCII decimal representation of the character, such as Return (ASCII character 13) for line-at-a-time transmissions. The command can take multiple arguments, so you can define any number of characters as the dispatch character.

Example

The following example defines the characters Ctrl-D (ASCII decimal character 4) and Ctrl-Y (ASCII decimal character 25) as the dispatch characters:

```
cs> terminal dispatch-character 4 25
```

Establish Pending Output Notification

You can set up a line to inform a user who has multiple concurrent Telnet connections when output is pending on a connection other than the current one. You might want to know, for example, when another connection receives mail or a message. Use the **terminal notify** command, which has the following syntax:

terminal notify

This command has no arguments or keywords.

Select File Download Mode

You can temporarily set the ability of a line to act as a transparent pipe for file transfers using the **terminal download** command. You can use this feature to run a program such as KERMIT, XMODEM, or CrossTalk that downloads a file across a remote access server line. This command has the following syntax:

terminal download

This command sets up the terminal line to transmit data and is equivalent to entering all the following commands:

```
terminal telnet-transparent
terminal no escape-character
terminal no hold-character
terminal no pad 0
.
.
.
terminal no pad 128
terminal parity none
terminal databits 8
```

Select the Preferred Terminal Transport Protocol

You can use the **terminal transport preferred** command to specify the preferred protocol to use when a command does not specify one. For server products that support LAT, the default protocol is LAT. For those that do not support LAT, the default is Telnet. Other options include the UNIX rlogin and X.29 PAD protocols. This command has the following syntax:

terminal transport preferred {**telnet** | **pad** | **lat** | **rlogin** | **none**}

Syntax Description

telnet	Specifies the TCP/IP Telnet protocol.
pad	Specifies X.3 PAD, which is used most often to connect a remote access server to X.25 hosts.
lat	Specifies the LAT protocol.
rlogin	Specifies UNIX rlogin.
none	Prevents any protocol selection on the line. The system normally assumes that any unrecognized command is a host name. If the protocol is set to none, the system no longer makes that assumption. No connection will be attempted if the command is not recognized.

Example

The following example shows you how to configure the console so that it does not connect when an unrecognized command is entered:

```
cs> terminal transport preferred none
```

Set the Number of Data Bits Per Character

To set the number of data bits per character that are interpreted and generated by software for the current line, use the **terminal data-character-bits** command. This command is used primarily to strip parity from X.25 connections on IGS or Cisco 3000 routers with the protocol translation software option. The **terminal data-character-bits** command does not work on hardwired lines. This command has the following syntax:

terminal data-character-bits {7 | 8}

Syntax Description

7 Seven data character bits.

8 Eight data character bits. This is the default.

Example

The following example sets the data bits per character on the current line to 7:

```
cs> terminal data-character-bits 7
```

Locally Specify the International Character Set

To locally change the ASCII character set used in EXEC and configuration command characters, use the **terminal exec-character-bits** EXEC command.

This EXEC command overrides the **default-value exec-character-bits** global configuration command. Configuring the EXEC character width to 8 bits enables you to add special graphical and international characters in banners, prompts, and so forth.

When the user exits the system, the character width is reset to the default value established by the global configuration command. However, setting the EXEC character width to 8 bits can also cause failures. If a user on a terminal that is sending parity enters the command **help**, an “unrecognized command” message appears because the system is reading all 8 bits, although the eighth bit is not needed for the **help** command.

This command has the following syntax:

terminal exec-character-bits {7 | 8}

Syntax Description

7 Selects the 7-bit ASCII character set.

8 Selects the full 8-bit character set.

Example

The following example temporarily configures a server product to use a full 8-bit user interface for system banners and prompts, allowing the use of additional graphical and international characters.

```
terminal exec-character-bits 8
```

Set the Character Dispatch Timer

To set the character dispatch timer for the current terminal line, use the **terminal dispatch-timeout** command. Use this command to increase the processing efficiency of the remote host. This command has the following syntax:

terminal dispatch-timeout *milliseconds*

Syntax Description

milliseconds An integer that specifies the number of milliseconds the server product waits after putting the first character into a packet buffer before sending the packet. During this interval, more characters can be added to the packet, which increases processing efficiency on the remote host.

Note The server product's response might appear intermittent if the timeout interval is greater than 100 milliseconds and remote echoing is used.

Example

The following example sets the dispatch timer to 80 milliseconds:

```
cs> terminal dispatch-timeout 80
```

Change ASCII Character Widths

To change the ASCII character widths to accept special characters for the current terminal line, use the **terminal special-character-bits** EXEC command. This is useful, for example, if you want the server product to provide temporary support for international character sets. It overrides the **default-value special-character-bits** global configuration command and is used to compare character sets typed by the user with the special character available during a data connection, which includes software flow control and escape characters. This command has the following syntax:

terminal special-character-bits {**7** | **8**}

Syntax Description

7 Selects the 7-bit ASCII character set. This is the default.

8 Selects the full 8-bit ASCII character set. Configuring the width to 8 enables you to use twice as many special characters as with the 7-bit setting. This selection enables you to add special graphical and international characters in banners, prompts, and so forth.

When you exit the system, the character width is reset to the default value established by the global configuration command. However, setting the EXEC character width to eight bits can also cause failures. If a user on a terminal that is sending parity enters the command **help**, an "unrecognized command" message appears because the system is reading all eight bits, although the eighth bit is not needed for the **help** command.

Example

The following example temporarily configures your server product to use a full 8-bit user interface for system banners and prompts. When you exit the system, character width is reset to the width established by the **default-value exec-character-bits** global configuration command.

```
cs> terminal special-character-bits 8
```

Set the Terminal Receive Speed

To set the terminal receive (from terminal) speed for the current line, use the **terminal rxspeed** command. This command has the following syntax:

terminal rxspeed *bps*

Syntax Description

bps Baud rate in bits per second (bps). The default is 9600 bps. Table 7-1 lists line speeds for supported server products.

Use Table 7-1 as a guide for setting the line speeds.

Table 7-1 Remote Access Server Line Speeds in Bits per Second

Remote Access Server Model	Baud Rates
500-CS	Any speed between 50 and 38400.
ASM-CS	The standard speeds include 75, 110, 134, 150, 300, 600, 1200, 2000, 2400, 4800, 1800, 9600, and 19200. Nonstandard speeds include 11520, 12800, 14400, 16457, 23040, 28800, 38400, and 57600.
Cisco 2500	Any speed from 50 to 115200.
Cisco 7000, AGS, CGS, MGS	50, 75, 110, 134, 150, 200, 300, 600, 1050, 1200, 2000, 2400, 4800, 9600, 19200, and 38400.
Cisco 2000, Cisco 3000, Cisco 4000	75, 110, 134, 150, 300, 600, 1200, 2000, 2400, 4800, 1800, 9600, 19200, and 38400.

Example

The following example sets the current auxiliary line receive speed to 2400 bps:

```
router> terminal rxspeed 2400
```

Set the Terminal Transmit Speed

To set the terminal transmit (to terminal) on the current line, use the **terminal txspeed** command. This command has the following syntax:

terminal txspeed *bps*

Syntax Description

bps Baud rate in bits per second (bps). The default is 9600 bps. Table 7-1 lists line speeds for supported server products.

Use Table 7-1 as a guide for setting the line speeds.

Example

The following example sets the current auxiliary line transmit speed to 2400 bps:

```
cs> terminal txspeed 2400
```

Generate a Hardware Break Signal for a Reverse Telnet Connection

To cause the system to generate a hardware Break signal on the RS-232 line that is associated with a reverse Telnet connection for the current line, use the **terminal telnet break-on-ip** EXEC command.

The hardware break signal occurs when a Telnet Interrupt-Process (IP) command is received on that connection. This command can be used to control the translation of Telnet IP commands into X.25 Break indications.

This command is also a useful workaround in the following situations:

- Several user Telnet programs send an IP command, but cannot send a Telnet break signal.
- Some Telnet programs implement a Break signal that sends an IP command.

Some RS-232 hardware devices use a hardware Break signal for various purposes. A hardware Break signal is generated when a Telnet Break command is received.

Note This command applies only to communication and access servers. It is not supported on standalone routers.

This command has the following syntax:

terminal telnet break-on-ip

Example

The following example shows how to generate a Break signal on the RS-232 line:

```
line aux 0
terminal telnet break-on-ip
```

Set the Line to Refuse Full-Duplex, Remote Echo Connections

You can set the line to allow the server product to refuse full-duplex, remote echo connection requests from the other end. This task suppresses negotiation of the Telnet Remote Echo and Suppress Go Ahead options. To set the current line to refuse to negotiate full-duplex, remote echo options on incoming connections, use the **terminal telnet refuse-negotiations** EXEC command.

Note This command applies only to communication and access servers. It is not supported on standalone routers.

This command has the following syntax:

terminal telnet refuse-negotiations

Example

The following example shows how to set an asynchronous interface to refuse full duplex, remote echo requests:

```
line async 1
terminal telnet refuse-negotiations
```

Allow Transmission Speed Negotiation

To allow the server product to negotiate transmission speed for the current line, use the **terminal telnet speed** EXEC command.

You can match line speeds on remote systems in reverse Telnet, on host machines hooked up to a server product to access the network, or on a group of console lines hooked up to the server product, when disparate line speeds are in use at the local and remote ends of the connection. Line speed negotiation adheres to the Remote Flow Control option, defined in RFC 1080.

Note This command applies only to communication and access servers. It is not supported on standalone routers.

This command has the following syntax:

terminal telnet speed *default-speed maximum-speed*

Syntax Description

default-speed	Line speed (in bps) that the server product will use if the device on the other end of the connection has not specified a speed.
maximum-speed	Maximum speed (in bps) that the device on the port will use.

Example

The following example enables the server product to negotiate a bit rate on the line using the Telnet option. If no speed is negotiated, the line will run at 2400 bps. If the remote host requests a speed of greater than 9600 bps, then 9600 bps will be used.

```
line async 7
terminal telnet speed 2400 9600
```

Synchronize the BREAK Signal

You can set the line to cause a reverse Telnet line to send a Telnet Synchronize signal when it receives a Telnet BREAK signal. The TCP Synchronize signal clears the data path, but interprets incoming commands. To cause the server product to send a Telnet Synchronize signal when it receives a Telnet Break signal on the current line, use the **terminal telnet sync-on-break** EXEC command.

Note This command applies only to communication and access servers. It is not supported on standalone routers.

This command has the following syntax:

terminal telnet sync-on-break

Example

The following example shows how to set an asynchronous line to cause the server product to send a Telnet synchronize signal:

```
line async 15
terminal telnet sync-on-break
```

Change the End-of-Line Character

The end of each line typed at the terminal is ended with a Return (CR). Use the **terminal telnet-transparent** command to cause the current terminal line to send a CR as a CR followed by a NULL instead of a CR followed by a LINE FEED (LF). This permits interoperability with different interpretations of end-of-line handling in the Telnet protocol specification.

Note This command applies only to communication and server products. It is not supported on standalone routers.

This command has the following syntax:

terminal telnet-transparent

Display Debug Messages on the Console and Terminals

To display **debug** command output and system error messages in EXEC mode on the current terminal, use the **terminal monitor** command. Remember that all terminal parameter-setting commands are set locally and do not remain in effect after a session is ended. You must perform this task at the privileged-level EXEC prompt at each session to see the debugging messages. This command has the following syntax:

terminal monitor

For more information about privileged-level EXEC mode, refer to the configuration guide or command reference publication for your server product.