Xterm Control Sequences

Edward Moy
University of California, Berkeley

Revised by
Stephen Gildea
X Consortium (1994)

Thomas Dickey
XFree86 Project (1996-2005)

Definitions

\[ c \] The literal character \( c \).

\( C \) A single (required) character.

\( P_s \) A single (usually optional) numeric parameter, composed of one of more digits.

\( P_m \) A multiple numeric parameter composed of any number of single numeric parameters, separated by \( ; \) character(s). Individual values for the parameters are listed with \( P_s \).

\( P_t \) A text parameter composed of printable characters.

C1 (8-Bit) Control Characters

The \textit{xterm} program recognizes both 8-bit and 7-bit control characters. It generates 7-bit controls (by default) or 8-bit if S8C1T is enabled. The following pairs of 7-bit and 8-bit control characters are equivalent:

\[
\text{ESC} \, D \quad \text{Index (IND is 0x84)}
\]

\[
\text{ESC} \, E \quad \text{Next Line (NEL is 0x85)}
\]

\[
\text{ESC} \, H \quad \text{Tab Set (HTS is 0x88)}
\]

\[
\text{ESC} \, M \quad \text{Reverse Index (RI is 0x8d)}
\]

\[
\text{ESC} \, N \quad \text{Single Shift Select of G2 Character Set (SS2 is 0x8e): affects next character only}
\]

\[
\text{ESC} \, O \quad \text{Single Shift Select of G3 Character Set (SS3 is 0x8f): affects next character only}
\]

\[
\text{ESC} \, P \quad \text{Device Control String (DCS is 0x90)}
\]
Xterm Control Sequences

C1 (8-Bit) Control Characters

```
ESC [ V
Start of Guarded Area (SPA is 0x96)

ESC [ W
End of Guarded Area (EPA is 0x97)

ESC [ X
Start of String (SOS is 0x98)

ESC [ Z
Return Terminal ID (DECID is 0x9a). Obsolete form of CSI c (DA).

ESC [ [ Control Sequence Introducer (CSI is 0x9b)

ESC \ String Terminator (ST is 0x9c)

ESC ] Operating System Command (OSC is 0x9d)

ESC \ Privacy Message (PM is 0x9e)

ESC _ Application Program Command (APC is 0x9f)
```

These control characters are used in the vtXXX emulation.

**VT100 Mode**

Most of these control sequences are standard VT102 control sequences, but there is support for later DEC VT terminals (i.e., VT220, VT320, VT420), as well as ISO 6429 and aixterm color controls. The only VT102 features not supported is auto-repeat, since the only way X provides for this will affect all windows. There are additional control sequences to provide xterm-dependent functions, such as the scrollbar or window size. Where the function is specified by DEC or ISO 6429, the code assigned to it is given in parentheses. The escape codes to designate and invoke character sets are specified by ISO 2022; see that document for a discussion of character sets.

**Single-character functions**

```
BEL
Bell (Ctrl-G)

BS
Backspace (Ctrl-H)

CR
Carriage Return (Ctrl-M)

ENQ
Return Terminal Status (Ctrl-E). Default response is an empty string, but may be overridden by a resource answerbackString.

FF
Form Feed or New Page (NP) (Ctrl-L) same as LF

LF
Line Feed or New Line (NL) (Ctrl-J)

SO
Shift Out (Ctrl-N) → Switch to Alternate Character Set: invokes the G1 character set.

SP
Space.

TAB
Horizontal Tab (HT) (Ctrl-I)

VT
Vertical Tab (Ctrl-K) same as LF

SI
Shift In (Ctrl-O) → Switch to Standard Character Set: invokes the G0 character set (the default).
```
Controls beginning with ESC (other than those where ESC is part of a 7-bit equivalent to 8-bit C1 controls), ordered by the final character(s).

| ESC | SP | F | 7-bit controls (S7C1T). |
| ESC | SP | G | 8-bit controls (S8C1T). |
| ESC | SP | L | Set ANSI conformance level 1 (dpANS X3.134.1). |
| ESC | SP | M | Set ANSI conformance level 2 (dpANS X3.134.1). |
| ESC | SP | N | Set ANSI conformance level 3 (dpANS X3.134.1). |
| ESC | # | 3 | DEC double-height line, top half (DECDHL) |
| ESC | # | 4 | DEC double-height line, bottom half (DECDHL) |
| ESC | # | 5 | DEC single-width line (DECSWL) |
| ESC | # | 6 | DEC double-width line (DECDWL) |
| ESC | # | 8 | DEC Screen Alignment Test (DECALN) |
| ESC | % | @ | Select default character set, ISO 8859-1 (ISO 2022) |
| ESC | % | G | Select UTF-8 character set (ISO 2022) |
| ESC | % | C | Designate G0 Character Set (ISO 2022) |
| ESC | % | I | Designate G1 Character Set (ISO 2022) |
| ESC | % | * | Designate G2 Character Set (ISO 2022) |
| ESC | % | + | Designate G3 Character Set (ISO 2022) |

Final character C for designating character sets [0, A] and [B] apply to VT100 and up, the remainder to VT220 and up:

- C = [0] → DEC Special Character and Line Drawing Set
- C = [A] → United Kingdom (UK)
- C = [B] → United States (USASCII)
- C = [C] → Dutch
- C = [D] → Finnish
- C = [E] → French
- C = [Q] → French Canadian
- C = [K] → German
- C = [Y] → Italian
- C = [Z] → Spanish
- C = [H] or [7] → Swedish
- C = [H] → Swiss

| ESC | 7 | Save Cursor (DECSC) |
| ESC | 8 | Restore Cursor (DECRC) |
| ESC | = | Application Keypad (DECPAM) |
| ESC | > | Normal Keypad (DECPNM) |
| ESC | F | Cursor to lower left corner of screen (if enabled by the hpLowerleftBugCompat resource) |
| ESC | C | Full Reset (RIS) |
Xterm Control Sequences

**ESC** l
Memory Lock (per HP terminals). Locks memory above the cursor.

**ESC** m
Memory Unlock (per HP terminals)

**ESC** n
Invoke the G2 Character Set as GL (LS2).

**ESC** o
Invoke the G3 Character Set as GL (LS3).

**ESC** |
Invoke the G3 Character Set as GR (LS3R).

**ESC** }
Invoke the G2 Character Set as GR (LS2R).

**ESC** ˜
Invoke the G1 Character Set as GR (LS1R).

**Application Program-Control functions**

**APC** P s t
*xterm* implements no **APC** functions; **P** is ignored. **P** need not be printable characters.

**Device-Control functions**

**DCS** P s ; P t
User-Defined Keys (DECUDK). The first parameter:

- **P** = 0 → Clear all UDK definitions before starting (default)
- **P** = 1 → Erase Below (default)

The second parameter:

- **P** = 0 → Lock the keys (default)
- **P** = 1 → Do not lock.

The third parameter is a ‘;’-separated list of strings denoting the key-code separated by a ‘/’ from the hex-encoded key value. The key codes correspond to the DEC function-key codes (e.g., F6=17).

**DCS** $ q P t
Request Status String (DECRQSS). The string following the "q" is one of the following:

- "q" → DECSCA
- "p" → DECSCL
- "r" → DECTBMB
- "m" → SGR

*xterm* responds with **DCS** 1 $ r P t for valid requests, replacing the **P** with the corresponding **CSI** string, or **DCS** 0 $ r P t for invalid requests.

**DCS** + q P t
Request Termcap/Termino String (*xterm*, experimental). The string following the "q" is a list of names encoded in hexadecimal (2 digits per character) separated by ';' which correspond to termcap or terminfo key names.

*xterm* responds with **DCS** 1 + r P t for valid requests, adding to **P** an '=' and the value of the corresponding string that *xterm* would send, or **DCS** 0 + r P t for invalid requests. The strings are encoded in hexadecimal (2 digits per character).

**Functions using **CSI**, ordered by the final character(s)**

- **CSI** P s @
  Insert **P** (Blank) Character(s) (default = 1) (ICH)

- **CSI** P s A
  Cursor Up **P** Times (default = 1) (CUU)

- **CSI** P s B
  Cursor Down **P** Times (default = 1) (CUD)

- **CSI** P s C
  Cursor Forward **P** Times (default = 1) (CUF)
Xterm Control Sequences

```
CSI \[D\]            Cursor Backward \(P\) Times (default = 1) (CUB)
CSI \[E\]            Cursor Next Line \(P\) Times (default = 1) (CNL)
CSI \[F\]            Cursor Preceding Line \(P\) Times (default = 1) (CPL)
CSI \[G\]            Cursor Character Absolute [column] (default = [row,1]) (CHA)
CSI \[H;P\]           Cursor Position [row;column] (default = [1,1]) (CUP)
CSI \[I\]            Cursor Forward Tabulation \(P\) tab stops (default = 1) (CHT)
CSI \[J\]            Erase in Display (ED)
                      \(P\) = [0] \(\rightarrow\) Erase Below (default)
                      \(P\) = [1] \(\rightarrow\) Erase Above
                      \(P\) = [2] \(\rightarrow\) Erase All
                      \(P\) = [3] \(\rightarrow\) Erase Saved Lines (xterm)
CSI \[K\]            Erase in Line (EL)
                      \(P\) = [0] \(\rightarrow\) Erase to Right (default)
                      \(P\) = [1] \(\rightarrow\) Erase to Left
                      \(P\) = [2] \(\rightarrow\) Erase All
CSI \[L\]            Insert \(P\) Line(s) (default = 1) (IL)
CSI \[M\]            Delete \(P\) Line(s) (default = 1) (DL)
CSI \[P\]            Delete \(P\) Character(s) (default = 1) (DCH)
CSI \[S\]            Scroll up \(P\) lines (default = 1) (SU)
CSI \[T\]            Scroll down \(P\) lines (default = 1) (SD)
CSI \[;P;;P;;P;;T\]    Erase \(P\) Character(s) (default = 1) (ECH)
CSI \[Z\]            Cursor Backward Tabulation \(P\) tab stops (default = 1) (CBT)
CSI \[\_\]            Character Position Absolute [column] (default = [row,1]) (HPA)
CSI \[b\]            Repeat the preceding graphic character \(P\) times (REP)
CSI \[c\]            Send Device Attributes (Primary DA)

\(P\) = [0] or omitted \(\rightarrow\) request attributes from terminal. The response depends on the decTerminalID resource setting.

\(\rightarrow\) \CSI \[1;2\] ("VT100 with Advanced Video Option")
\(\rightarrow\) \CSI \[1;0\] ("VT101 with No Options")
\(\rightarrow\) \CSI \[6\] ("VT102")
```

Initiate highlight Mouse Tracking. Parameters are [func;startx;starty;firstrow;lastrow].

See the section Mouse Tracking.

Send Device Attributes (Primary DA)

\(P\) = [0] or omitted \(\rightarrow\) request attributes from terminal. The response depends on the decTerminalID resource setting.
The VT100-style response parameters do not mean anything by themselves. VT220 parameters do, telling the host what features the terminal supports:

- 1: 132-columns
- 2: Printer
- 6: Selective erase
- 8: User-defined keys
- 9: National replacement character sets
- 15: Technical characters
- 2:2: ANSI color, e.g., VT525
- 2:9: ANSI text locator (i.e., DEC Locator mode)

Send Device Attributes (Secondary DA)

\[ \text{CSI} \rightarrow P \] or omitted → request the terminal’s identification code. The response depends on the \text{decTerminalID} resource setting. It should apply only to VT220 and up, but \textit{xterm} extends this to VT100.

\[ \text{CSI} > P_p ; P_v ; P_c \]

where \( P_p \) denotes the terminal type

- 0 (“VT100”)
- 1 (“VT220”)

and \( P_v \) is the firmware version (for \textit{xterm}, this is the XFree86 patch number, starting with 95). In a DEC terminal, \( P_c \) indicates the ROM cartridge registration number and is always zero.

- \text{CSI} P_m d → Line Position Absolute [row] (default = [1,column]) (VPA)
- \text{CSI} P_s ; P_t f → Horizontal and Vertical Position [row;column] (default = [1,1]) (HVP)
- \text{CSI} P_s g → Tab Clear (TBC)
- 0 → Clear Current Column (default)
- 3 → Clear All

Set Mode (SM)

- 2 → Keyboard Action Mode (AM)
- 4 → Insert Mode (IRM)
- 1 2 → Send/receive (SRM)
- 2 0 → Automatic Newline (LNM)

DECSCLM)

\[ \text{CSI} > P_m h \]

DEC Private Mode Set (DECSET)

- 1 → Application Cursor Keys (DECCKM)
- 2 → Designate USASCII for character sets G0-G3 (DECANM), and set VT100 mode.
- 3 → 132 Column Mode (DECCOLM)
- 4 → Smooth (Slow) Scroll (DECSCLM)
- 5 → Reverse Video (DECSNM)
- 6 → Origin Mode (DECOM)
Xterm Control Sequences

\[ P_s = 7 \rightarrow \text{Wraparound Mode (DECAWM)} \]
\[ P_s = 8 \rightarrow \text{Auto-repeat Keys (DECARM)} \]
\[ P_s = 9 \rightarrow \text{Send Mouse X & Y on button press. See the section \textbf{Mouse Tracking}.} \]
\[ P_s = 10 \rightarrow \text{Show toolbar (rxvt)} \]
\[ P_s = 12 \rightarrow \text{Start Blinking Cursor (att610)} \]
\[ P_s = 18 \rightarrow \text{Print form feed (DECPFF)} \]
\[ P_s = 19 \rightarrow \text{Set print extent to full screen (DECPEX)} \]
\[ P_s = 25 \rightarrow \text{Show Cursor (DECTCEM)} \]
\[ P_s = 30 \rightarrow \text{Show scrollbar (rxvt).} \]
\[ P_s = 35 \rightarrow \text{Enable font-shifting functions (rxvt).} \]
\[ P_s = 38 \rightarrow \text{Enter Tektronix Mode (DECTEK)} \]
\[ P_s = 40 \rightarrow \text{ Allow 80 → 132 Mode} \]
\[ P_s = 41 \rightarrow \text{More(1) fix (see \textit{curses} resource)} \]
\[ P_s = 42 \rightarrow \text{Enable Nation Replacement Character sets (DECNRCM)} \]
\[ P_s = 44 \rightarrow \text{Turn On Margin Bell} \]
\[ P_s = 45 \rightarrow \text{Reverse-wraparound Mode} \]
\[ P_s = 46 \rightarrow \text{Start Logging (normally disabled by a compile-time option)} \]
\[ P_s = 47 \rightarrow \text{Use Alternate Screen Buffer (unless disabled by the \textit{titeInhibit} resource)} \]
\[ P_s = 66 \rightarrow \text{Application keypad (DECNKM)} \]
\[ P_s = 67 \rightarrow \text{Backarrow key sends backspace (DECBKM)} \]
\[ P_s = 1000 \rightarrow \text{Send Mouse X & Y on button press and release. See the section \textbf{Mouse Tracking}.} \]
\[ P_s = 1001 \rightarrow \text{Use Hilite Mouse Tracking.} \]
\[ P_s = 1002 \rightarrow \text{Use Cell Motion Mouse Tracking.} \]
\[ P_s = 1003 \rightarrow \text{Use All Motion Mouse Tracking.} \]
\[ P_s = 1010 \rightarrow \text{Scroll to bottom on tty output (rxvt).} \]
\[ P_s = 1011 \rightarrow \text{Scroll to bottom on key press (rxvt).} \]
\[ P_s = 1035 \rightarrow \text{Enable special modifiers for Alt and NumLock keys.} \]
\[ P_s = 1036 \rightarrow \text{Send ESC when Meta modifies a key (enables the \textit{metaSend-sEscape} resource).} \]
\[ P_s = 1037 \rightarrow \text{Send DEL from the editing-keypad Delete key} \]
\[ P_s = 1047 \rightarrow \text{Use Alternate Screen Buffer (unless disabled by the \textit{titeInhibit} resource)} \]
\[ P_s = 1048 \rightarrow \text{Save cursor as in DECSC (unless disabled by the \textit{titeInhibit} resource)} \]
\[ P_s = 1049 \rightarrow \text{Save cursor as in DECSC and use Alternate Screen Buffer, clearing it first (unless disabled by the \textit{titeInhibit} resource). This combines the effects of the 1047 and 1048 modes. Use this with terminfo-based applications rather than the 47 mode.} \]
\[ P_s = 1051 \rightarrow \text{Set Sun function-key mode.} \]
Xterm Control Sequences

VT100 Mode

$P_s = \begin{bmatrix} 1 & 0 & 5 & 2 \end{bmatrix} \rightarrow \text{Set HP function-key mode.}$

$P_s = \begin{bmatrix} 1 & 0 & 5 & 3 \end{bmatrix} \rightarrow \text{Set SCO function-key mode.}$

$P_s = \begin{bmatrix} 1 & 0 & 6 & 0 \end{bmatrix} \rightarrow \text{Set legacy keyboard emulation (X11R6).}$

$P_s = \begin{bmatrix} 1 & 0 & 6 & 1 \end{bmatrix} \rightarrow \text{Set Sun/PC keyboard emulation of VT220 keyboard.}$

$P_s = \begin{bmatrix} 2 & 0 & 0 & 4 \end{bmatrix} \rightarrow \text{Set bracketed paste mode.}$

**CSI $P_m$ 1**

Media Copy (MC)

$P_s = \begin{bmatrix} 0 \end{bmatrix} \rightarrow \text{Print screen (default)}$

$P_s = \begin{bmatrix} 4 \end{bmatrix} \rightarrow \text{Turn off printer controller mode}$

$P_s = \begin{bmatrix} 5 \end{bmatrix} \rightarrow \text{Turn on printer controller mode}$

**CSI $P_m$ i**

Media Copy (MC, DEC-specific)

$P_s = \begin{bmatrix} 1 \end{bmatrix} \rightarrow \text{Print line containing cursor}$

$P_s = \begin{bmatrix} 4 \end{bmatrix} \rightarrow \text{Turn off autoprint mode}$

$P_s = \begin{bmatrix} 5 \end{bmatrix} \rightarrow \text{Turn on autoprint mode}$

$P_s = \begin{bmatrix} 1 & 0 \end{bmatrix} \rightarrow \text{Print composed display, ignores DECPEX}$

$P_s = \begin{bmatrix} 1 & 1 \end{bmatrix} \rightarrow \text{Print all pages}$

**CSI $P_m$ 1**

Reset Mode (RM)

$P_s = \begin{bmatrix} 2 \end{bmatrix} \rightarrow \text{Keyboard Action Mode (AM)}$

$P_s = \begin{bmatrix} 4 \end{bmatrix} \rightarrow \text{Replace Mode (IRM)}$

$P_s = \begin{bmatrix} 1 & 2 \end{bmatrix} \rightarrow \text{Send/receive (SRM)}$

$P_s = \begin{bmatrix} 2 & 0 \end{bmatrix} \rightarrow \text{Normal Linefeed (LNM)}$

**CSI $P_m$ i**

DEC Private Mode Reset (DECRST)

$P_s = \begin{bmatrix} 1 \end{bmatrix} \rightarrow \text{Normal Cursor Keys (DECCKM)}$

$P_s = \begin{bmatrix} 2 \end{bmatrix} \rightarrow \text{Designate VT52 mode (DECANM).}$

$P_s = \begin{bmatrix} 3 \end{bmatrix} \rightarrow \text{80 Column Mode (DECCOLM)}$

$P_s = \begin{bmatrix} 4 \end{bmatrix} \rightarrow \text{Jump (Fast) Scroll (DECSCLM)}$

$P_s = \begin{bmatrix} 5 \end{bmatrix} \rightarrow \text{Normal Video (DECSM)}$

$P_s = \begin{bmatrix} 6 \end{bmatrix} \rightarrow \text{Normal Cursor Mode (DECOM)}$

$P_s = \begin{bmatrix} 7 \end{bmatrix} \rightarrow \text{No Wraparound Mode (DECAWM)}$

$P_s = \begin{bmatrix} 8 \end{bmatrix} \rightarrow \text{No Auto-repeat Keys (DECARM)}$

$P_s = \begin{bmatrix} 9 \end{bmatrix} \rightarrow \text{Don’t Send Mouse X & Y on button press}$

$P_s = \begin{bmatrix} 1 & 0 \end{bmatrix} \rightarrow \text{Hide toolbar (rxvt)}$

$P_s = \begin{bmatrix} 1 & 2 \end{bmatrix} \rightarrow \text{Stop Blinking Cursor (att610)}$

$P_s = \begin{bmatrix} 1 & 8 \end{bmatrix} \rightarrow \text{Don’t print form feed (DECPFF)}$

$P_s = \begin{bmatrix} 1 & 9 \end{bmatrix} \rightarrow \text{Limit print to scrolling region (DECPEX)}$

$P_s = \begin{bmatrix} 2 & 5 \end{bmatrix} \rightarrow \text{Hide Cursor (DECTCEM)}$

$P_s = \begin{bmatrix} 3 & 0 \end{bmatrix} \rightarrow \text{Don’t show scrollbar (rxvt).}$

$P_s = \begin{bmatrix} 3 & 5 \end{bmatrix} \rightarrow \text{Disable font-shifting functions (rxvt).}$

$P_s = \begin{bmatrix} 4 & 0 \end{bmatrix} \rightarrow \text{Disallow 80 ↔ 132 Mode}$

$P_s = \begin{bmatrix} 4 & 1 \end{bmatrix} \rightarrow \text{No more(1) fix (see curses resource)}$

$P_s = \begin{bmatrix} 4 & 2 \end{bmatrix} \rightarrow \text{Disable Nation Replacement Character sets (DECNRCM)}$

$P_s = \begin{bmatrix} 4 & 4 \end{bmatrix} \rightarrow \text{Turn Off Margin Bell}$
Xterm Control Sequences

VT100 Mode

\[ P_s = \begin{array}{c|c} 4 & 5 \\ \end{array} \] → No Reverse-wraparround Mode
\[ P_s = \begin{array}{c|c} 4 & 6 \\ \end{array} \] → Stop Logging (normally disabled by a compile-time option)
\[ P_s = \begin{array}{c|c} 4 & 7 \\ \end{array} \] → Use Normal Screen Buffer
\[ P_s = \begin{array}{c|c} 6 & 6 \\ \end{array} \] → Numeric keypad (DECNKM)
\[ P_s = \begin{array}{c|c} 6 & 7 \\ \end{array} \] → Backarrow key sends delete (DECBKM)
\[ P_s = \begin{array}{c|c|c} 1 & 0 & 0 \\ \end{array} \] → Don’t Send Mouse X & Y on button press and release. See the section **Mouse Tracking**.
\[ P_s = \begin{array}{c|c|c} 1 & 0 & 1 \\ \end{array} \] → Don’t Use Hilite Mouse Tracking
\[ P_s = \begin{array}{c|c|c} 1 & 0 & 2 \\ \end{array} \] → Don’t Use Cell Motion Mouse Tracking
\[ P_s = \begin{array}{c|c|c} 1 & 0 & 3 \\ \end{array} \] → Don’t Use All Motion Mouse Tracking
\[ P_s = \begin{array}{c|c} 1 & 0 \\ \end{array} \] → Don’t scroll to bottom on tty output (rxvt).
\[ P_s = \begin{array}{c|c} 1 & 1 \\ \end{array} \] → Don’t scroll to bottom on key press (rxvt).
\[ P_s = \begin{array}{c|c|c} 1 & 3 & 5 \\ \end{array} \] → Disable special modifiers for Alt and NumLock keys.
\[ P_s = \begin{array}{c|c|c} 1 & 3 & 6 \\ \end{array} \] → Don’t send ESC when Meta modifies a key (disables the `metaSendsEscape` resource).
\[ P_s = \begin{array}{c|c|c|c} 1 & 3 & 7 \\ \end{array} \] → Send VT220 Remove from the editing-keypad Delete key
\[ P_s = \begin{array}{c|c|c} 1 & 4 & 7 \\ \end{array} \] → Use Normal Screen Buffer, clearing screen first if in the Alternate Screen (unless disabled by the `titeInhibit` resource)
\[ P_s = \begin{array}{c|c|c} 1 & 4 & 8 \\ \end{array} \] → Restore cursor as in DECRC (unless disabled by the `titeInhibit` resource)
\[ P_s = \begin{array}{c|c|c} 1 & 4 & 9 \\ \end{array} \] → Use Normal Screen Buffer and restore cursor as in DECRC (unless disabled by the `titeInhibit` resource). This combines the effects of the \[ \begin{array}{c} 1 \\ \end{array} \] \[ \begin{array}{c} 4 \\ \end{array} \] \[ \begin{array}{c} 7 \\ \end{array} \] and \[ \begin{array}{c} 1 \\ \end{array} \] \[ \begin{array}{c} 4 \\ \end{array} \] \[ \begin{array}{c} 8 \\ \end{array} \] modes. Use this with terminfo-based applications rather than the \[ \begin{array}{c} 4 \\ \end{array} \] \[ \begin{array}{c} 7 \\ \end{array} \] mode.
\[ P_s = \begin{array}{c|c} 1 & 5 \\ \end{array} \] → Reset Sun function-key mode.
\[ P_s = \begin{array}{c|c} 1 & 6 \\ \end{array} \] → Reset HP function-key mode.
\[ P_s = \begin{array}{c|c} 1 & 7 \\ \end{array} \] → Reset SCO function-key mode.
\[ P_s = \begin{array}{c|c} 1 & 8 \\ \end{array} \] → Reset legacy keyboard emulation (X11R6).
\[ P_s = \begin{array}{c|c} 2 & 0 \\ \end{array} \] → Reset Sun/PC keyboard emulation of VT220 keyboard.
\[ P_s = \begin{array}{c|c} 2 & 4 \\ \end{array} \] → Reset bracketed paste mode.

Powerful 

Character Attributes (SGR)

\[ P_s = \begin{array}{c} 0 \\ \end{array} \] → Normal (default)
\[ P_s = \begin{array}{c} 1 \\ \end{array} \] → Bold
\[ P_s = \begin{array}{c} 4 \\ \end{array} \] → Underlined
\[ P_s = \begin{array}{c} 5 \\ \end{array} \] → Blink (appears as Bold)
\[ P_s = \begin{array}{c} 7 \\ \end{array} \] → Inverse
\[ P_s = \begin{array}{c} 8 \\ \end{array} \] → Invisible, i.e., hidden (VT300)
\[ P_s = \begin{array}{c} 2 \end{array} \] \[ \begin{array}{c} 2 \\ \end{array} \] → Normal (neither bold nor faint)
\[ P_s = \begin{array}{c} 2 \end{array} \] \[ \begin{array}{c} 4 \\ \end{array} \] → Not underlined
\[ P_s = \begin{array}{c} 2 \end{array} \] \[ \begin{array}{c} 5 \\ \end{array} \] → Steady (not blinking)
\[ P_s = \begin{array}{c} 2 \end{array} \] \[ \begin{array}{c} 7 \\ \end{array} \] → Positive (not inverse)
If 16-color support is compiled, the following apply. Assume that xterm’s resources are set so that the ISO color codes are the first 8 of a set of 16. Then the aixterm colors are the bright versions of the ISO colors:

\[\begin{align*}
P_s &= \text{[9 0]} \rightarrow \text{Set foreground color to Black} \\
P_s &= \text{[9 1]} \rightarrow \text{Set foreground color to Red} \\
P_s &= \text{[9 2]} \rightarrow \text{Set foreground color to Green} \\
P_s &= \text{[9 3]} \rightarrow \text{Set foreground color to Yellow} \\
P_s &= \text{[9 4]} \rightarrow \text{Set foreground color to Blue} \\
P_s &= \text{[9 5]} \rightarrow \text{Set foreground color to Magenta} \\
P_s &= \text{[9 6]} \rightarrow \text{Set foreground color to Cyan} \\
P_s &= \text{[9 7]} \rightarrow \text{Set foreground color to White} \\
P_s &= \text{[1 0 0]} \rightarrow \text{Set background color to Black} \\
P_s &= \text{[1 0 1]} \rightarrow \text{Set background color to Red} \\
P_s &= \text{[1 0 2]} \rightarrow \text{Set background color to Green} \\
P_s &= \text{[1 0 3]} \rightarrow \text{Set background color to Yellow} \\
P_s &= \text{[1 0 4]} \rightarrow \text{Set background color to Blue} \\
P_s &= \text{[1 0 5]} \rightarrow \text{Set background color to Magenta} \\
P_s &= \text{[1 0 6]} \rightarrow \text{Set background color to Cyan} \\
P_s &= \text{[1 0 7]} \rightarrow \text{Set background color to White} \\
\end{align*}\]

If xterm is compiled with the 16-color support disabled, it supports the following, from
Xterm Control Sequences

**rxvt:**

\[ P_s = \begin{array}{ccc} 1 & 0 & 0 \end{array} \rightarrow \text{Set foreground and background color to default} \]

If 88- or 256-color support is compiled, the following apply.

\[ P_s = \begin{array}{ccc} 3 & 8 & 5 \end{array} \rightarrow \text{Set foreground color to the second} \]
\[ P_s = \begin{array}{ccc} 4 & 8 & 5 \end{array} \rightarrow \text{Set background color to the second} \]

\[ \text{CSI} P_s n \]

**Device Status Report (DSR)**

\[ P_s = \begin{array}{c} 5 \end{array} \rightarrow \text{Status Report} \text{CSI} 0 n \text{ (“OK”)} \]
\[ P_s = \begin{array}{c} 6 \end{array} \rightarrow \text{Report Cursor Position (CPR) [row;column] as} \text{CSI} r ; c R \]

\[ \text{CSI} \? P_s n \]

**Device Status Report (DSR, DEC-specific)**

\[ P_s = \begin{array}{c} 6 \end{array} \rightarrow \text{Report Cursor Position (CPR) [row;column] as} \text{CSI} \? r ; c R \]

(assumes page is zero).

\[ P_s = \begin{array}{c} 1 & 5 \end{array} \rightarrow \text{Report Printer status as} \text{CSI} \? 1 0 n \text{ (ready) or} \]
\[ P_s = \begin{array}{c} 1 \end{array} n \text{ (not ready)} \]
\[ P_s = \begin{array}{c} 2 & 5 \end{array} \rightarrow \text{Report UDK status as} \text{CSI} \? 2 0 n \text{ (unlocked) or} \]
\[ P_s = \begin{array}{c} 2 \end{array} n \text{ (locked)} \]
\[ P_s = \begin{array}{c} 2 & 6 \end{array} \rightarrow \text{Report Keyboard status as} \text{CSI} \? 2 7 ; 1 ; 0 ; 0 n \text{ (North American)} \]

The last two parameters apply to VT400 & up, and denote keyboard ready and LK01 respectively.

\[ P_s = \begin{array}{c} 5 & 3 \end{array} \rightarrow \text{Report Locator status as} \text{CSI} \? 5 3 n \text{ Locator available, if compiled-in, or} \]
\[ P_s = \begin{array}{c} 5 \end{array} 0 n \text{ No Locator, if not.} \]

\[ \text{CSI} \? p \]

**Set conformance level (DECSCL)** Valid values for the first parameter:

\[ P_s = \begin{array}{c} 6 \end{array} 1 \rightarrow \text{VT100} \]
\[ P_s = \begin{array}{c} 6 \end{array} 2 \rightarrow \text{VT200} \]
\[ P_s = \begin{array}{c} 6 \end{array} 3 \rightarrow \text{VT300} \]

Valid values for the second parameter:

\[ P_s = \begin{array}{c} 0 \end{array} \rightarrow \text{8-bit controls} \]
\[ P_s = \begin{array}{c} 1 \end{array} \rightarrow \text{7-bit controls (always set for VT100)} \]
\[ P_s = \begin{array}{c} 2 \end{array} \rightarrow \text{8-bit controls} \]

\[ \text{CSI} P_s \? q \]

Select character protection attribute (DECSCA). Valid values for the parameter:

\[ P_s = \begin{array}{c} 0 \end{array} \rightarrow \text{DECSED and DECSEL can erase (default)} \]
\[ P_s = \begin{array}{c} 1 \end{array} \rightarrow \text{DECSED and DECSEL cannot erase} \]
\[ P_s = \begin{array}{c} 2 \end{array} \rightarrow \text{DECSED and DECSEL can erase} \]

\[ \text{CSI} P_s ; P_s \? r \]

Set Scrolling Region [top;bottom] (default = full size of window) (DECSTBM)

\[ \text{CSI} \? P_m r \]

Restore DEC Private Mode Values. The value of \( P_s \) previously saved is restored. \( P_s \) values are the same as for DECSCL.
Xterm Control Sequences  

VT100 Mode

\[ \text{CSI} \ P \ S \ P \ l \ P \ b \ P \ r \ P_1 S \ r \]

Change Attributes in Rectangular Area (DECCARA).

\( P_1 \ P_2 \ P_3 \ P_4 \ P_5 \) denotes the rectangle.

\( P_5 \) denotes the SGR attributes to change: 0, 1, 4, 5, 7

Save cursor (ANSLSYS)

\[ \text{CSI} \ P \]

Save DEC Private Mode Values. \( P_1 \) values are the same as for DECSET.

\[ \text{CSI} \ P \ P_2 \ P_3 \]

Window manipulation (from dtterm, as well as extensions). These controls may be disabled using the allowWindowOps resource. Valid values for the first (and any additional parameters) are:

\( P_5 = 1 \) → De-iconify window.

\( P_5 = 2 \) → Iconify window.

\( P_5 = 3 \); \( x \); \( y \) → Move window to \([x, y]\).

\( P_5 = 4 \); \( \text{height} \); \( \text{width} \) → Resize the xterm window to height and width in pixels.

\( P_5 = 5 \) → Raise the xterm window to the front of the stacking order.

\( P_5 = 6 \) → Lower the xterm window to the bottom of the stacking order.

\( P_5 = 7 \) → Refresh the xterm window.

\( P_5 = 8 \); \( \text{height} \); \( \text{width} \) → Resize the text area to \([\text{height};\text{width}]\) in characters.

\( P_5 = 9 \); \( 0 \) → Restore maximized window.

\( P_5 = 9 \); \( 1 \) → Maximize window (i.e., resize to screen size).

\( P_5 = 10 \) → Report xterm window state. If the xterm window is open (non-iconified), it returns \( \text{CSI} \ 10 \). If the xterm window is iconified, it returns \( \text{CSI} \ 20 \).

\( P_5 \geq 24 \) → Resize to \( P_5 \) lines (DECSLPP)

\[ \text{CSI} \ P \ P_2 \ P_3 \ P_4 \ P_5 \ P_6 \ P_7 \ P_8 \]

Reverse Attributes in Rectangular Area (DECRARA).

\( P_1 \ P_2 \ P_3 \ P_4 \ P_5 \) denotes the rectangle.

\( P_5 \) denotes the attributes to reverse. 1, 4, 5, 7

Save cursor (ANSLSYS)

\[ \text{CSI} \ u \]

Copy Rectangular Area (DECCRA)

\( P_1 \ P_2 \ P_3 \ P_4 \ P_5 \) denotes the rectangle.

\( P_5 \) denotes the source page.

\( P_1 \ P_1 \) denotes the target location.

\( P_1 \) denotes the target page.
Enable Filter Rectangle (DECEFR)
Parameters are [top;left;bottom;right].
Defines the coordinates of a filter rectangle and activates it. Anytime the locator is
detected outside of the filter rectangle, an outside rectangle event is generated and the rec-
tangle is disabled. Filter rectangles are always treated as "one-shot" events. Any param-
ters that are omitted default to the current locator position. If all parameters are omitted,
any locator motion will be reported. DECELR always cancels any previous rectangle def-
inition.

Request Terminal Parameters (DECREQTPARM)
if $P_s$ is a "0" (default) or "1", and xterm is emulating VT100, the control sequence elicits
a response of the same form whose parameters describe the terminal:

$P_s \rightarrow$ the given $P_x$ incremented by 2.

1 $\rightarrow$ no parity
1 $\rightarrow$ eight bits
1 2 8 $\rightarrow$ transmit 38.4k baud
1 2 8 $\rightarrow$ receive 38.4k baud
1 $\rightarrow$ clock multiplier
0 $\rightarrow$ STP flags

Select Attribute Change Extent (DECSACE).

$P_s = 0 \rightarrow$ from start to end position, wrapped
$P_s = 1 \rightarrow$ from start to end position, wrapped
$P_s = 2 \rightarrow$ rectangle (exact).

Fill Rectangular Area (DECFRA).

$P_t$ is the character to use.

$P_t; P_l; P_b; P_r$ denotes the rectangle.

Enable Locator Reporting (DECELR)
Valid values for the first parameter:

$P_s = 0 \rightarrow$ Locator disabled (default)
$P_s = 1 \rightarrow$ Locator enabled
$P_s = 2 \rightarrow$ Locator enabled for one report, then disabled

The second parameter specifies the coordinate unit for locator reports.

Valid values for the second parameter:

$P_u = 0$ or omitted $\rightarrow$ default to character cells
$P_u = 1 \rightarrow$ device physical pixels
$P_u = 2 \rightarrow$ character cells

Erase Rectangular Area (DECERA).

$P_t; P_l; P_b; P_r$ denotes the rectangle.
Select Locator Events (DECSLE)
Valid values for the first (and any additional parameters) are:
\[ P_s = 0 \rightarrow \text{only respond to explicit host requests (DECRQLP)} \]
\[ \text{(default) also cancels any filter rectangle} \]
\[ P_s = 1 \rightarrow \text{report button down transitions} \]
\[ P_s = 2 \rightarrow \text{do not report button down transitions} \]
\[ P_s = 3 \rightarrow \text{report button up transitions} \]
\[ P_s = 4 \rightarrow \text{do not report button up transitions} \]

Selective Erase Rectangular Area (DECSERA).
\[ P_{10}; P_{11}; P_{12}; P_{13} \text{ denotes the rectangle.} \]

Request Locator Position (DECRQLP)
Valid values for the parameter are:
\[ P_s = 0, 1 \text{ or omitted} \rightarrow \text{transmit a single DECLRP locator report} \]

If Locator Reporting has been enabled by a DECELR, xterm will respond with a DECLRP Locator Report. This report is also generated on button up and down events if they have been enabled with a DECSLE, or when the locator is detected outside of a filter rectangle, if filter rectangles have been enabled with a DECEFR.

\[ \rightarrow \text{csi } P_{\text{0}} ; P_{\text{1}} ; P_{\text{2}} ; P_{\text{3}} ; P_{\text{e}} ; P_{\text{b}} ; P_{\text{p}} \& w \]

Parameters are [event;button;row;column;page].
Valid values for the event:
\[ P_e = 0 \rightarrow \text{locator unavailable - no other parameters sent} \]
\[ P_e = 1 \rightarrow \text{request - xterm received a DECRQLP} \]
\[ P_e = 2 \rightarrow \text{left button down} \]
\[ P_e = 3 \rightarrow \text{left button up} \]
\[ P_e = 4 \rightarrow \text{middle button down} \]
\[ P_e = 5 \rightarrow \text{middle button up} \]
\[ P_e = 6 \rightarrow \text{right button down} \]
\[ P_e = 7 \rightarrow \text{right button up} \]
\[ P_e = 8 \rightarrow \text{M4 button down} \]
\[ P_e = 9 \rightarrow \text{M4 button up} \]
\[ P_e = 10 \rightarrow \text{locator outside filter rectangle} \]

“button” parameter is a bitmask indicating which buttons are pressed:
\[ P_b = 0 \rightarrow \text{no buttons down} \]
\[ P_b & 1 \rightarrow \text{right button down} \]
\[ P_b & 2 \rightarrow \text{middle button down} \]
\[ P_b & 4 \rightarrow \text{left button down} \]
\[ P_b & 8 \rightarrow \text{M4 button down} \]
“row” and “column” parameters are the coordinates of the locator position in the xterm window, encoded as ASCII decimal.
The “page” parameter is not used by xterm, and will be omitted.

Operating System Controls

Set Text Parameters. For colors and font, if $P_t$ is a “?”, the control sequence elicits a response which consists of the control sequence which would set the corresponding value. The dtterm control sequences allow you to determine the icon name and window title.

- $P_s = 0$ → Change Icon Name and Window Title to $P_t$
- $P_s = 1$ → Change Icon Name to $P_t$
- $P_s = 2$ → Change Window Title to $P_t$
- $P_s = 3$ → Set X property on top-level window. $P_t$ should be in the form "prop=value", or just "prop" to delete the property
- $P_s = 4$; $c$; spec → Change Color Number $c$ to the color specified by spec, i.e., a name or RGB specification as per XParseColor. Any number of $c$ name pairs may be given. The color numbers correspond to the ANSI colors 0-7, their bright versions 8-15, and if supported, the remainder of the 88-color or 256-color table.

If a “?” is given rather than a name or RGB specification, xterm replies with a control sequence of the same form which can be used to set the corresponding color. Because more than one pair of color number and specification can be given in one control sequence, xterm can make more than one reply.

The 8 colors which may be set using $[1 0$ through $[1 7$ are denoted dynamic colors, since the corresponding control sequences were the first means for setting xterm’s colors dynamically, i.e., after it was started. They are not the same as the ANSI colors. One or more parameters is expected for $P_t$. Each successive parameter changes the next color in the list. The value of $P_t$ tells the starting point in the list. The colors are specified by name or RGB specification as per XParseColor.

If a “?” is given rather than a name or RGB specification, xterm replies with a control sequence of the same form which can be used to set the corresponding dynamic color. Because more than one pair of color number and specification can be given in one control sequence, xterm can make more than one reply.

$$P_s = [1 0] \rightarrow \text{Change VT100 text foreground color to } P_t$$
$$P_s = [1 1] \rightarrow \text{Change VT100 text background color to } P_t$$
$$P_s = [1 2] \rightarrow \text{Change text cursor color to } P_t$$
$$P_s = [1 3] \rightarrow \text{Change mouse foreground color to } P_t$$
Xterm Control Sequences

VT100 Mode

\( Ps = \begin{bmatrix} 1 & 4 \end{bmatrix} \rightarrow \text{Change mouse background color to } P_i \)
\( Ps = \begin{bmatrix} 1 & 5 \end{bmatrix} \rightarrow \text{Change Tektronix foreground color to } P_i \)
\( Ps = \begin{bmatrix} 1 & 6 \end{bmatrix} \rightarrow \text{Change Tektronix background color to } P_i \)
\( Ps = \begin{bmatrix} 1 & 7 \end{bmatrix} \rightarrow \text{Change highlight color to } P_i \)
\( Ps = \begin{bmatrix} 1 & 8 \end{bmatrix} \rightarrow \text{Change Tektronix cursor color to } P_i \)
\( Ps = \begin{bmatrix} 4 & 6 \end{bmatrix} \rightarrow \text{Change Log File to } P_i \) (normally disabled by a compile-time option)

\( Ps = \begin{bmatrix} 5 & 0 \end{bmatrix} \rightarrow \text{Set Font to } P_i \) If \( P_i \) begins with a "#", index in the font menu, relative (if the next character is a plus or minus sign) or absolute. A number is expected but not required after the sign (the default is the current entry for relative, zero for absolute indexing).

\( Ps = \begin{bmatrix} 5 & 1 \end{bmatrix} \) (reserved for Emacs shell)

\( Ps = \begin{bmatrix} 5 & 2 \end{bmatrix} \rightarrow \text{Manipulate Selection Data. These controls may be disabled using the } allowWindowOps \text{ resource. The parameter } P_i \text{ is parsed as } P_c; P_d \)
The first, \( P_c \), may contain any character from the set \( \{ c \ p \ s \ 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \} \). It is used to construct a list of selection parameters for clipboard, primary, select, or cut buffers 0 through 8 respectively, in the order given. If the parameter is empty, \textit{xterm} uses \( \{ s \ 0 \} \), to specify the configurable primary/clipboard selection and cut buffer 0.
The second parameter, \( P_d \), gives the selection data. Normally this is a string encoded in base64. The data becomes the new selection, which is then available for pasting by other applications.
If the second parameter is a \( ? \), \textit{xterm} replies to the host with the selection data encoded using the same protocol.

Privacy Message
\[ \text{Privacy Message } \]

\( PM \)

\( Ps \)

\[ \text{ST} \]

\textit{xterm} implements no \( PM \) functions; \( P_i \) is ignored. \( P_i \) need not be printable characters.

PC-Style Function Keys

If \textit{xterm} does minimal translation of the function keys, it usually does this with a PC-style keyboard, so PC-style function keys result. Sun keyboards are similar to PC keyboards. Both have cursor and scrolling operations printed on the keypad, which duplicate the smaller cursor and scrolling keypads.
X does not predefine NumLock (used for VT220 keyboards) or Alt (used as an extension for the Sun/PC keyboards) as modifiers. These keys are recognized as modifiers when enabled by the \texttt{numLock} resource, or by the DECSET \( 1 \ 0 \ 3 \ 5 \) control sequence.
The cursor keys transmit the following escape sequences depending on the mode specified via the \texttt{DECKKM} escape sequence.
### Xterm Control Sequences

<table>
<thead>
<tr>
<th>Key</th>
<th>Normal</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cursor Up</td>
<td>CSI A</td>
<td>SS5 A</td>
</tr>
<tr>
<td>Cursor Down</td>
<td>CSI B</td>
<td>SS5 B</td>
</tr>
<tr>
<td>Cursor Right</td>
<td>CSI C</td>
<td>SS5 C</td>
</tr>
<tr>
<td>Cursor Left</td>
<td>CSI D</td>
<td>SS5 D</td>
</tr>
</tbody>
</table>

The home- and end-keys (unlike PageUp and other keys also on the 6-key editing keypad) are considered "cursor keys" by xterm. Their mode is also controlled by the DECCKM escape sequence:

<table>
<thead>
<tr>
<th>Key</th>
<th>Normal</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>CSI H</td>
<td>SS5 H</td>
</tr>
<tr>
<td>End</td>
<td>CSI F</td>
<td>SS5 F</td>
</tr>
</tbody>
</table>

The application keypad transmits the following escape sequences depending on the mode specified via the DECPNM and DECPAM escape sequences. Use the NumLock key to override the application mode.

Not all keys are present on the Sun/PC keypad (e.g., PF1, Tab), but are supported by the program.

<table>
<thead>
<tr>
<th>Key</th>
<th>Numeric</th>
<th>Application</th>
<th>Terminfo</th>
<th>Termcap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>SP</td>
<td>SS5 SP</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Tab</td>
<td>TAB</td>
<td>SS5 I</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Enter</td>
<td>CR</td>
<td>SS5 M</td>
<td>kent</td>
<td>@8</td>
</tr>
<tr>
<td>PF1</td>
<td>SS5 P</td>
<td>SS5 P</td>
<td>kf1</td>
<td>k1</td>
</tr>
<tr>
<td>PF2</td>
<td>SS5 O</td>
<td>SS5 O</td>
<td>kf2</td>
<td>k2</td>
</tr>
<tr>
<td>PF3</td>
<td>SS5 R</td>
<td>SS5 R</td>
<td>kf3</td>
<td>k3</td>
</tr>
<tr>
<td>PF4</td>
<td>SS5 S</td>
<td>SS5 S</td>
<td>kf4</td>
<td>k4</td>
</tr>
</tbody>
</table>

* (multiply) | * | SS5 j | - | - |
+ (add) | + | SS5 k | - | - |
, (comma) | , | SS5 l | - | - |
- (minus) | - | SS5 m | - | - |
 Delete | - | CSI 3 ~ | - | - |
 / (divide) | / | SS5 o | - | - |
 0 (Insert) | 0 | CSI 2 ~ | - | - |
 1 (End) | 1 | SS5 F | kc1 | K4 |
 2 (DownArrow) | 2 | CSI B | - | - |
 3 (PageDown) | 3 | CSI 6 ~ | kc3 | K5 |
 4 (LeftArrow) | 4 | CSI D | - | - |
 5 (Begin) | 5 | CSI E | kb2 | K2 |
 6 (RightArrow) | 6 | CSI C | - | - |
 7 (Home) | 7 | SS5 H | ka1 | K1 |
 8 (UpArrow) | 8 | CSI A | - | - |
 9 (PageUp) | 9 | CSI 5 ~ | ka3 | K3 |
= (equal) | = | SS5 X | - | - |

They also provide 12 function keys, as well as a few other special-purpose keys.

<table>
<thead>
<tr>
<th>Key</th>
<th>Escape Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>SS5 P</td>
</tr>
<tr>
<td>F2</td>
<td>SS5 Q</td>
</tr>
<tr>
<td>F3</td>
<td>SS5 R</td>
</tr>
<tr>
<td>F4</td>
<td>SS5 S</td>
</tr>
<tr>
<td>F5</td>
<td>CSI 1 5 ~</td>
</tr>
<tr>
<td>F6</td>
<td>CSI 1 7 ~</td>
</tr>
<tr>
<td>F7</td>
<td>CSI 1 8 ~</td>
</tr>
<tr>
<td>F8</td>
<td>CSI 1 9 ~</td>
</tr>
<tr>
<td>F9</td>
<td>CSI 2 0 ~</td>
</tr>
</tbody>
</table>
Older versions of xterm implement different escape sequences for F1 through F4. These can be activated by setting the oldXtermFKeys resource. However, since they do not correspond to any hardware terminal, they have been deprecated. (The DEC VT220 reserves F1 through F5 for local functions such as Setup).

<table>
<thead>
<tr>
<th>Key</th>
<th>Escape Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>CSI 1 1 ~</td>
</tr>
<tr>
<td>F2</td>
<td>CSI 1 2 ~</td>
</tr>
<tr>
<td>F3</td>
<td>CSI 1 3 ~</td>
</tr>
<tr>
<td>F4</td>
<td>CSI 1 4 ~</td>
</tr>
</tbody>
</table>

In normal mode, i.e., a Sun/PC keyboard when the sunKeyboard resource is false, xterm recognizes function key modifiers which are parameters appended before the final character of the control sequence.

<table>
<thead>
<tr>
<th>Code</th>
<th>Modifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Shift</td>
</tr>
<tr>
<td>3</td>
<td>Alt</td>
</tr>
<tr>
<td>4</td>
<td>Shift + Alt</td>
</tr>
<tr>
<td>5</td>
<td>Control</td>
</tr>
<tr>
<td>6</td>
<td>Shift + Control</td>
</tr>
<tr>
<td>7</td>
<td>Alt + Control</td>
</tr>
<tr>
<td>8</td>
<td>Shift + Alt + Control</td>
</tr>
</tbody>
</table>

For example, shift-F5 would be sent as CSI 1 5 : 2 ~

If the alwaysUseMods resource is set, the Meta modifier also is recognized, making parameters 9 through 16.

**VT220-Style Function Keys**

However, xterm is most useful as a DEC VT102 or VT220 emulator. Set the sunKeyboard resource to true to force a Sun/PC keyboard to act like a VT220 keyboard.

The VT102/VT220 application keypad transmits unique escape sequences in application mode, which are distinct from the cursor and scrolling keypad:

<table>
<thead>
<tr>
<th>Key</th>
<th>Numeric</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>SP</td>
<td>SS3 SP</td>
</tr>
<tr>
<td>Tab</td>
<td>TAB</td>
<td>SS3 I</td>
</tr>
<tr>
<td>Enter</td>
<td>CR</td>
<td>SS3 M</td>
</tr>
<tr>
<td>PF1</td>
<td>P</td>
<td>SS3 P</td>
</tr>
<tr>
<td>PF2</td>
<td>Q</td>
<td>SS3 Q</td>
</tr>
<tr>
<td>PF3</td>
<td>R</td>
<td>SS3 R</td>
</tr>
<tr>
<td>PF4</td>
<td>S</td>
<td>SS3 S</td>
</tr>
<tr>
<td>* (multiply)</td>
<td>*</td>
<td>SS3 j</td>
</tr>
<tr>
<td>+ (add)</td>
<td>+</td>
<td>SS3 k</td>
</tr>
<tr>
<td>, (comma)</td>
<td>,</td>
<td>SS3 I</td>
</tr>
<tr>
<td>- (minus)</td>
<td>-</td>
<td>SS3 m</td>
</tr>
<tr>
<td>. (period)</td>
<td>.</td>
<td>SS3 n</td>
</tr>
<tr>
<td>/ (divide)</td>
<td>/</td>
<td>SS3 o</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>SS3 p</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>SS3 q</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>SS3 r</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>SS3 s</td>
</tr>
</tbody>
</table>
The VT220 provides a 6-key editing keypad, which is analogous to that on the PC keyboard. It is not affected by DECKM or DECPNM/DECPAM:

<table>
<thead>
<tr>
<th>Key</th>
<th>Normal/Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>CSI 2 ^</td>
</tr>
<tr>
<td>Delete</td>
<td>CSI 3 ^</td>
</tr>
<tr>
<td>Home</td>
<td>CSI 1 ^</td>
</tr>
<tr>
<td>End</td>
<td>CSI 4 ^</td>
</tr>
<tr>
<td>PageUp</td>
<td>CSI 5 ^</td>
</tr>
<tr>
<td>PageDown</td>
<td>CSI 6 ^</td>
</tr>
</tbody>
</table>

The VT220 provides 8 additional function keys. With a Sun/PC keyboard, access these keys by Control/F1 for F13, etc.

<table>
<thead>
<tr>
<th>Key</th>
<th>Escape Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>F13</td>
<td>CSI 2 5 ^</td>
</tr>
<tr>
<td>F14</td>
<td>CSI 2 6 ^</td>
</tr>
<tr>
<td>F15</td>
<td>CSI 2 8 ^</td>
</tr>
<tr>
<td>F16</td>
<td>CSI 2 9 ^</td>
</tr>
<tr>
<td>F17</td>
<td>CSI 3 1 ^</td>
</tr>
<tr>
<td>F18</td>
<td>CSI 3 2 ^</td>
</tr>
<tr>
<td>F19</td>
<td>CSI 3 3 ^</td>
</tr>
<tr>
<td>F20</td>
<td>CSI 3 4 ^</td>
</tr>
</tbody>
</table>

VT52-Style Function Keys

A VT52 does not have function keys, but it does have a numeric keypad and cursor keys. They differ from the other emulations by the prefix. Also, the cursor keys do not change:

<table>
<thead>
<tr>
<th>Key</th>
<th>Normal/Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cursor Up</td>
<td>ESC A</td>
</tr>
<tr>
<td>Cursor Down</td>
<td>ESC B</td>
</tr>
<tr>
<td>Cursor Right</td>
<td>ESC C</td>
</tr>
<tr>
<td>Cursor Left</td>
<td>ESC D</td>
</tr>
</tbody>
</table>

The keypad is similar:

<table>
<thead>
<tr>
<th>Key</th>
<th>Numeric</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>SP</td>
<td>ESC ? SP</td>
</tr>
<tr>
<td>Tab</td>
<td>TAB</td>
<td>ESC ? I</td>
</tr>
<tr>
<td>Enter</td>
<td>CR</td>
<td>ESC ? M</td>
</tr>
<tr>
<td>PF1</td>
<td>ESC P</td>
<td>ESC P</td>
</tr>
<tr>
<td>PF2</td>
<td>ESC Q</td>
<td>ESC Q</td>
</tr>
<tr>
<td>PF3</td>
<td>ESC R</td>
<td>ESC R</td>
</tr>
<tr>
<td>PF4</td>
<td>ESC S</td>
<td>ESC S</td>
</tr>
<tr>
<td>* (multiply)</td>
<td>*</td>
<td>ESC ? I</td>
</tr>
<tr>
<td>+ (add)</td>
<td>+</td>
<td>ESC ? K</td>
</tr>
</tbody>
</table>
Sun-Style Function Keys

The xterm program provides support for Sun keyboards more directly, by a menu toggle that causes it to send Sun-style function key codes rather than VT220. Note, however, that the sun and VT100 emulations are not really compatible. For example, their wrap-margin behavior differs.

Only function keys are altered; keypad and cursor keys are the same. The emulation responds identically. See the xterm-sun terminfo entry for details.

HP-Style Function Keys

Similarly, xterm can be compiled to support HP keyboards. See the xterm-hp terminfo entry for details.

The Alternate Screen Buffer

Xterm maintains two screen buffers. The normal screen buffer allows you to scroll back to view saved lines of output up to the maximum set by the saveLines resource. The alternate screen buffer is exactly as large as the display, contains no additional saved lines. When the alternate screen buffer is active, you cannot scroll back to view saved lines. Xterm provides control sequences and menu entries for switching between the two.

Most full-screen applications use terminfo or termcap to obtain strings used to start/stop full-screen mode, i.e., smcup and rmcup for terminfo, or the corresponding ti and te for termcap. The titeInhibit resource removes the ti and te strings from the TERMCAPE string which is set in the environment for some platforms. That is not done when xterm is built with terminfo libraries because terminfo does not provide the whole text of the termcap data in one piece. It would not work for terminfo anyway, since terminfo data is not passed in environment variables; setting an environment variable in this manner would have no effect on the application’s ability to switch between normal and alternate screen buffers. Instead, the newer private mode controls (such as \[1 \ 0 \ 4 \ 9\]) for switching between normal and alternate screen buffers simply disable the switching. They add other features such as clearing the display for the same reason: to make the details of switching independent of the application that requests the switch.

Bracketed Paste Mode

When bracketed paste mode is set, pasted text is bracketed with control sequences so that the program can differentiate pasted text from typed-in text. When bracketed paste mode is set, the program will receive: ESC \[ 200 \ 201 \].

Mouse Tracking

The VT widget can be set to send the mouse position and other information on button presses. These modes are typically used by editors and other full-screen applications that want to make use of the mouse.

There are six mutually exclusive modes. One is DEC Locator mode, enabled by the DECSEL: \[P_3 P_5 Z\]
control sequence, and is not described here (control sequences are summarized above). The remaining five modes are each enabled (or disabled) by a different parameter in DECSET \texttt{CSI}?P_mh or DECRST \texttt{CSI}?P_ml control sequence.

Manifest constants for the parameter values are defined in \texttt{xcharmouse.h} as follows:

\begin{verbatim}
#define SET_X10_MOUSE 9
#define SET_VT200_MOUSE 1000
#define SET_VT200_HIGHLIGHT_MOUSE 1001
#define SET_BTN_EVENT_MOUSE 1002
#define SET_ANY_EVENT_MOUSE 1003
\end{verbatim}

The motion reporting modes are strictly \texttt{xterm} extensions, and are not part of any standard, though they are analogous to the DEC VT200 DECELR locator reports.

Parameters (such as pointer position and button number) for all mouse tracking escape sequences generated by \texttt{xterm} encode numeric parameters in a single character as value+32. For example, \texttt{?3} specifies the value 1. The upper left character position on the terminal is denoted as 1,1.

X10 compatibility mode sends an escape sequence only on button press, encoding the location and the mouse button pressed. It is enabled by specifying parameter 9 to DECSET. On button press, \texttt{xterm} sends \texttt{CSI M G_x C_x C_y} (6 characters). \(G_x\) is button−1. \(C_x\) and \(C_y\) are the x and y coordinates of the mouse when the button was pressed.

Normal tracking mode sends an escape sequence on both button press and release. Modifier key (shift, ctrl, meta) information is also sent. It is enabled by specifying parameter 1000 to DECSET. On button press or release, \texttt{xterm} sends \texttt{CSI M G_x C_x C_y}. The low two bits of \(G_x\) encode button information: 0=MB1 pressed, 1=MB2 pressed, 2=MB3 pressed, 3=release. The next three bits encode the modifiers which were down when the button was pressed and are added together: 4=Shift, 8=Meta, 16=Control. Note however that the shift and control bits are normally unavailable because \texttt{xterm} uses the control modifier with mouse for popup menus, and the shift modifier is used in the default translations for button events. The Meta modifier recognized by \texttt{xterm} is the mod1 mask, and is not necessarily the "Meta" key (see \texttt{xmodmap}). \(C_x\) and \(C_y\) are the x and y coordinates of the mouse event, encoded as in X10 mode.

Wheel mice may return buttons 4 and 5. Those buttons are represented by the same event codes as buttons 1 and 2 respectively, except that 64 is added to the event code. Release events for the wheel buttons are not reported.

Mouse highlight tracking notifies a program of a button press, receives a range of lines from the program, highlights the region covered by the mouse within that range until button release, and then sends the program the release coordinates. It is enabled by specifying parameter 1001 to DECSET. Highlighting is performed only for button 1, though other button events can be received. \textbf{Warning:} use of this mode requires a cooperating program or it will hang \texttt{xterm}. On button press, the same information as for normal tracking is generated; \texttt{xterm} then waits for the program to send mouse tracking information. \textit{All X events are ignored until the proper escape sequence is received from the pty: \texttt{CSI M P_s P_e P_s P_e P_s P_e T}.} The parameters are \texttt{func, startx, starty, firstrow, and lastrow}. \texttt{func} is non-zero to initiate highlight tracking and zero to abort. \texttt{startx} and \texttt{starty} give the starting x and y location for the highlighted region. The ending location tracks the mouse, but will never be above row \texttt{firstrow} and will always be above row \texttt{lastrow}. (The top of the screen is row 1.) When the button is released, \texttt{xterm} reports the ending position one of two ways: if the start and end coordinates are valid text locations: \texttt{CSI M C_x C_y}. If either coordinate is past the end of the line: \texttt{CSI T C_x C_y C_y}. The parameters are \texttt{startx, starty, endx, endy, mousex, and mousey}. \texttt{startx, starty, endx, and endy} give the starting and ending character positions of the region. \texttt{mousex and mousey} give the location of the mouse at button up, which may not be over a character.

Button-event tracking is essentially the same as normal tracking, but \texttt{xterm} also reports button-motion events. Motion events are reported only if the mouse pointer has moved to a different character cell. It is enabled by specifying parameter 1002 to DECSET. On button press or release, \texttt{xterm} sends the same codes used by normal tracking mode. On button-motion events, \texttt{xterm} adds 32 to the event code (the third character, \(C_y\)). The other bits of the event code specify button and modifier keys as in normal mode. For example, motion into cell x,y with button 1 down is
reported as $\text{CSI} \text{M} \left[ \text{button} \right] \text{C} \text{C}_x \text{C}_y \left( \text{button} = 32 + 0 \right) \text{motion indicator} \text{). Similarly, motion with button 3}

down is reported as $\text{CSI} \text{M} \left[ \text{button} \right] \text{B} \text{C} \text{C}_x \text{C}_y \left( \text{button} = 32 + 2 \right) \text{motion indicator} \text{).}

Any-event mode is the same as button-event mode, except that all motion events are reported, even if no mouse button is down. It is enabled by specifying 1003 to DECSET.

**Tektronix 4014 Mode**

Most of these sequences are standard Tektronix 4014 control sequences. Graph mode supports the 12-bit addressing of the Tektronix 4014. The major features missing are the write-through and defocused modes. This document does not describe the commands used in the various Tektronix plotting modes but does describe the commands to switch modes.
Xterm Control Sequences

**Tektronix 4014 Mode**

- `ESC i` Defocused Z Axis and Dotted Line Vectors
- `ESC j` Defocused Z Axis and Dot-Dashed Vectors
- `ESC k` Defocused Z Axis and Short-Dashed Vectors
- `ESC l` Defocused Z Axis and Long-Dashed Vectors
- `ESC p` Write-Thru Mode and Normal (solid) Vectors
- `ESC q` Write-Thru Mode and Dotted Line Vectors
- `ESC r` Write-Thru Mode and Dot-Dashed Vectors
- `ESC s` Write-Thru Mode and Short-Dashed Vectors
- `ESC t` Write-Thru Mode and Long-Dashed Vectors
- **FS** Point Plot Mode (Ctrl-\)
- **GS** Graph Mode (Ctrl-]
- **RS** Incremental Plot Mode (Ctrl-^)
- **US** Alpha Mode (Ctrl-_)  

**VT52 Mode**

Parameters for cursor movement are at the end of the `ESC Y` escape sequence. Each ordinate is encoded in a single character as `value+32`. For example, `!` is 1. The screen coordinate system is 0-based.

- `ESC A` Cursor up.
- `ESC B` Cursor down.
- `ESC C` Cursor right.
- `ESC D` Cursor left.
- `ESC F` Enter graphics mode.
- `ESC G` Exit graphics mode.
- `ESC H` Move the cursor to the home position.
- `ESC I` Reverse line feed.
- `ESC J` Erase from the cursor to the end of the screen.
- `ESC K` Erase from the cursor to the end of the line.
- `ESC Y p, p` Move the cursor to given row and column.
- `ESC Z` Identify \[ \text{→} \text{ESC} [/ Z] \] ("I am a VT52.")
- **ESCAPE** Enter alternate keypad mode.
- **ESCAPE** Exit alternate keypad mode.
- **ESCAPE** Exit VT52 mode (Enter VT100 mode).