AXEL Platine Terminal

Asynchronous AX3000 Models

Installing Platine Terminals on UNIX Systems

Dec. 1996 - Ref.: UNXE105/648-1

- 1 -
SETTINGS
FOR MOST UNIX VERSIONS

This guide provides the information needed to install the AX3000 Platine Terminal in a UNIX system environment.

This chapter (1) provides general information for setting-up the AX3000 serial models under UNIX, and introduces the Platine Terminal local printing feature.

Chapter 2 deals with installing AX3000 serial terminals under SCO XENIX, SCO UNIX 3.2v2, SCO UNIX 3.2v4 and SCO UNIX Release 5.

Chapter 3 deals with installing AX3000 serial terminals under AIX RS/6000.

Chapter 4 deals with the Platine Terminal Multiple Screens feature.

1.1 - SETTING TERMINAL EMULATION

Terminal installation under UNIX comprises two stages:

During the first stage the terminal mode and parameters are set using the terminal Set-Up. The terminal personality (emulation) is one of the most important parameters.

During the second stage the UNIX system is made aware of this terminal, by modifying various system files and setting the TERM environment variable.

AX3000 Platine Terminals offer:

- ANSI colour emulation: these emulations are the most suitable for full colour displays in all mainstream UNIX environments.
  Where the main UNIX console operates with ANSI personality, there is an added benefit that the **AXEL Platine keyboard and display exhibit the same behaviour as the main console**.

- VT220 emulation: this basic emulation is found on all versions of UNIX.
  Although VT220 supports only monochrome mode, the built-in colouring feature of the AXEL Platine allows colour display of monochrome applications. The AX3000 performs the colouring by generating a background and a character colour for each monochrome character attribute or semi-graphic character. See the User's Guide for further details).

The following table shows recommended predefined set-up and TERM values for the various operating systems:

UNIX system	Predefined set-up	TERM
SCO UNIX 3.2v2	SCO UNIX 3.2.2	ansi
SCO UNIX 3.2v4	SCO UNIX 3.24	ansi
SCO UNIX Release 5	SCO OPENSERVER	ansi
SCO XENIX	SCO XENIX	ansi
IBM AIX 3.1	ANSI RS/6000	hft-c
IBM AIX 3.2	ANSI RS6000	hft-c-old
UNIX Ware	UNIX SVR4	AT386
Interactive UNIX	ANSI INTERACTIVE	AT386
Solaris 2.3	SVR4 UNIX	AT386
Other UNIX	ANSI	ansi
	VT220	VT220

The next two sections give detailed instructions for setting terminal emulation.

1.1.1 - Selecting the Emulation on the AX3000 Platine Terminal

The built-in **predefined set-up <F10>** option provides **automatic, safe** settings for all standard terminal parameters to match the selected operating system (number of lines, function key values, etc.).

To use the predefined set-up, enter the following keystrokes:

CTRL ALT ESC	(enter terminal set-up)
F10	(enter predefined set-up)
↑ ↓ then ENTER	(select one of the predefined set-ups)
F12 then ENTER	(save selected predefined set-up)

When you select a predefined set-up it automatically initialises the main communication serial line parameters to the factory-default setting (38.4 KBaud, 8 data bits, no parity, 1 stop bit). If you need to modify any parameters, work through the set-up screen and update parameters as necessary. For details about main port parameter settings, refer to the "*Asynchronous AX3000 Models - User's Guide*".

1.1.2 - Setting the Emulation on the UNIX System.

The TERM variable should now be set to match the physical terminal personality selected above.

The UNIX TERM environment variable defines the terminal characteristics (numbers of lines and columns, function key values, etc.) by referencing an entry in `/etc/termcap` or a file in `/usr/lib/terminfo`. On some systems these files may be in other locations (e.g. `/usr/share/lib/termcap` and `/usr/share/lib/terminfo`).

It is customary to set the TERM environment variable, for the entire system, in the file `/etc/profile` or, for individual users, in their `$HOME/.profile`. In some UNIX versions (e. g. SCO UNIX), it is also possible to declare the TERM value in a file associating a device with a TERM variable. For detailed information please consult the documentation for the host computer.

1.2 - LOCAL PRINTER

The AX3000 Platine Terminal provides a serial and a parallel auxiliary ports as standard (Models 35E and 40B). Both ports may be used to connect peripherals such as printers, scanners, etc.

1.2.1 - Connecting a Local Printer

The printer may be connected to:

- the DB9 auxiliary serial port (AUX1) available on all models or
- the DB25 female auxiliary parallel port (available on Models 35 and 40B).

The printer should be declared during the Platine Terminal set-up procedure:

CTRL ALT ESC	(enter terminal set-up)
F4	(select auxiliary port option)

a - Setting a Parallel Printer

During the set-up procedure, enter the following key sequence:

SPACE	(set parallel printer port as 'installed')

ENTER	(set the 'default' parallel port parameter)
F12 then ENTER	(save current setting)

Parallel port setting is now complete.

b - Setting a Serial Printer

↓ ↑	(select 'serial port' parameter)
SPACE	(select serial port parameter as 'printer')
ENTER	(set the 'default' serial port parameter)
F12 then ENTER	(save current setting)

This procedure sets the auxiliary serial port parameters automatically to factory-default values (9.6 KBaud, 8 data bits, no parity, 1 stop bit). These parameters can then be altered through the set-up procedure.

1.2.2 - Using the Local Printer

The Platine can locally print either:
- the current screen display or
- files from the host system.

a - Print Screen

This feature is performed locally, by the Platine, without host intervention. No operating system setting is required.

To print the screen, press the **<PrtSc>** key.

b - Printing Files

Data received from the host, via the main terminal serial port, may include a mixture of display data (to appear on the screen) and printer data (to be printed locally).The terminal identifies the data as printer information when it is preceded with a 'start local printing escape sequence' and ended by a 'stop local printing escape sequence'. (‘Escape sequences’ are special groups of data characters that do not appear on the screen but control the terminal’s behaviour).

For ANSI or VT220 emulations, the AX3000 uses the following sequences (Esc stands for 1B hexadecimal value):

- start local printing escape sequence: `Esc [5i`
- stop local printing escape sequence: `Esc [4i`

Note: these two sequences are specified by '`mc4`' and '`mc5`' capabilities in a UNIX terminfo file.

Very few emulations include a standard UNIX command to invoke this local printing feature. A specific script has to be created as follows:

i) Become superuser (using the UNIX `su` command). You will need to enter the superuser password.

ii) Change to the `/usr/bin` directory.

iii) Using the `vi` text editor, create the following file and name it `localprint`:

```
if [ -z "$1" ]; then
   echo "Usage : $0 file"
   exit
   fi
echo $1" printing in progress..."
echo "\033[5i\c"
cat $1
echo "\014\c"
echo "\033[4i\c"
echo "OK"
```

iv) After you have quit the `vi` editor, give execute permission for the newly generated file. At the UNIX command prompt, enter:

```
# chmod +x /usr/bin/localprint <RETURN>
```

v) Exit superuser (type <`Ctrl`><D> or `exit`)

To print a file on the Platine local printer, enter the following command at the terminal to which the printer is connected (where filename is the full or relative UNIX pathname of the file to be printed):

```
$ localprint filename <RETURN>
```

The file will not be printed at the Platine Terminal's local printer if the command is entered at the system console or at another terminal. To initiate local printing from a remote terminal, enter the following command (where `/dev/platine` is the Platine Terminal's device file):

```
$ localprint filename > /dev/platine <RETURN>
```

Note: it is recommended that you do not use the Platine Terminal's keyboard during local printing.

- 2 - SETTINGS FOR SCO UNIX/XENIX

This chapter describes AXEL Platine terminal settings for SCO XENIX and SCO UNIX (3.2v2, 3.2v4 and Release 5).

2.1 - SOFTWARE INSTALLATION

To configure an AXEL Platine terminal entails two distinct steps:
- Setting terminal parameters,
- Setting operating system.

2.1.1 - Setting Terminal Parameters

The built-in AXEL set-up automatically sets all standard terminal parameters for the selected operating system. Enter the following keystrokes:

CTRL ALT ESC	(enter terminal set-up)
F10	(select predefined set-up)
↑ ↓ then ENTER	(select a UNIX predefined set-up: SCO UNIX 3.2.2, SCO UNIX 3.2.4, SCO OPENSERVER or SCO XENIX)
F12 then ENTER	(save selected predefined set-up)

The AX3000 Platine is now ready to operate under SCO UNIX/XENIX at 38.4 Kbaud. If necessary, terminal parameters (e.g. baud rate) can be further modified using the terminal set-up sequence. For additional set-up information, refer to the "*Asynchronous AX3000 Models - User's Guide"*.

2.1.2 - Setting the Operating System

With SCO UNIX/XENIX, 4 steps are necessary to perform software installation:
- serial port selection and configuration,
- serial line setting,
- terminal description,
- keyboard mapfile optional assignment (`mapchan`).

The host computer's serial port must be configured independently of the connected terminal. The following instructions assume that the main serial port on the Platine has already been set.

The standard COM1 serial port is used as an example throughout this chapter. The procedure will be similar for any other serial port.

On UNIX, every serial port is associated with one or more dedicated files called device files. These files are used to ″read from″ the serial port or ″write to″ the serial port.

On SCO UNIX the following two device files are associated with COM1:
- `/dev/tty1a` and
- `/dev/tty1A`.

In most cases, device names that include an upper-case character are reserved for modems (for example `/dev/tty1A`).

Device names with only lower-case characters are dedicated to terminals or printers (for example `/dev/tty1a`).

a - Setting a Serial Line

The file `/etc/gettydefs` defines capabilities (such as communication parameters) of miscellaneous serial peripherals: printers, terminals, modems, etc.

This file associates a variable (usually a single character) with each peripheral feature. For example:

2 is used for a modem at 1.2 Kbaud,
6 is used for a terminal or modem at 9.6 Kbaud,
m is used for a terminal at 9.6 Kbaud,
n is used for a terminal at 19.2 Kbaud,
o is used for a terminal at 38.4 Kbaud.

By default, the `/dev/tty1a` device is associated with the **m** parameter (data transmission at 9.6 Kbaud).

The `/etc/inittab` file associates the devices with their parameters (on XENIX, this file is called `/etc/ttys`).

In this file each port is represented by a single line. Find the line that refers to the port being used. The line for COM1 should appear as follows:

```
se1a:23:off:/etc/getty tty1a m
```

In this example, replace the '`m`' by an '`o`' to modify transmission speed from 9.6 Kbaud to 38.4 Kbaud. The line then becomes:

```
se1a:23:off:/etc/getty tty1a o
```

Save the modified file.

The `/etc/inittab` (or `/etc/ttys`) file is created by the concatenation of the device driver init files in the directory `/etc/conf/init.d`. Therefore, any modifications to the `/etc/inittab` file must also be reflected in the correct device driver init file. However, this is performed automatically by the enable command.

Example: the COM1 device driver init file is: `/etc/conf/init.d/sio`.

b - Activating the Serial Port

The serial port is activated with the `enable` command. This command must specify the device name for the port to which the terminal is connected.

i) Become superuser (using the UNIX `su` command)

ii) Enter the following at the UNIX prompt (we are using a connection to port COM1 as an example):

```
# enable tty1a <RETURN>
```

This command automatically updates the `/etc/inittab` and `/etc/conf/init.d/sio` files.

iii) Exit superuser (type `<Ctrl><D>` or `exit`).

The serial line is now on. It can be turned off at any time with the `disable` command.

c - Declaring the Terminal Type

The AX3000 Platine can emulate an **ANSI** terminal, using the **ANSI** terminal definition in the `termcap` and `terminfo` files. Under SCO UNIX/XENIX the main console also emulates an ANSI terminal using this definition. This allows the Platine terminal to be **directly used without any modification to terminfo or termcap files**.

There are two ways to describe the terminal emulation:

- Set and export `TERM` variable:

If you are running the Bourne or Korn shell:

```
$ TERM=ansi <RETURN>
$ export TERM <RETURN>
```

If you are running the C shell:

```
% setenv TERM ansi <RETURN>
```

These commands can be executed also from a script (for example the user's `.profile` file, if you are running the Bourne or Korn shell).

- SCO UNIX allows a device to be "dedicated" to a specified terminal emulation. Add the following line in `/etc/ttytype` file to match the device to the emulation type:

```
ansi     tty1a
```

d - Keyboard Mapfile Assignment

Under SCO UNIX/XENIX, a terminal (or the main console) can be assigned to a keyboard mapfile. This mapfile acts like a filter to allow additional features, such as the use of 'compose characters' (for example **^** and **e** for **ê**).

The mapfile assignment can be done in two ways:

- Either use the `mapchan` command at the UNIX prompt:

```
$ mapchan /usr/lib/mapchan/cons.ibm <RETURN>
```

  This command can be executed, either from a script (for example the user's `.profile` file), or from the shell.

- or add the following line to the `/etc/default/mapchan` mapfile to match the device file to the mapfile:

```
cons.ibm       tty1a
```

The `cons.ibm` parameter mapfile is only used above as an example. The `/usr/lib/mapchan` directory contains all available mapfiles.

Note: to get identical capabilities on the AXEL Platine and the main console, you must assign the same mapfile to the Platine and to the main console.

For additional information, refer to the following SCO Manuals:

User's Reference(C)	: ENABLE, DISABLE, STTY
User's Reference(M)	: GETTY, UNGETTY, LOGIN, MAPCHAN
User's Reference(F)	: GETTYDEFS, MAPCHAN
User's Reference(ADM)	: MKDEV

2.2 - THE MULTISCREEN FEATURE

Multiple screens are a standard feature on the main console of a SCO system. However, the multiscreen must be explicitly declared to operate on a terminal. This terminal multiscreen feature can be added with the mscreen software included with SCO UNIX/XENIX.

An AX3000 Platine can manage up to 8 simultaneous screens. Each screen is stored and updated inside the Platine itself, instead of being stored in the host's memory. This Platine capability allows an instantaneous switch between screens.

Each screen feature includes:

- the screen display (characters and their attributes),
- the current attributes (colour, cursor position, display control, etc.).

Note: AXEL has designed a special multiscreen software for the AX3000 Platine terminal. This software is called `axmscreen` and offers additional features to `mscreen`, such as the multishell and initial commands. For additional information, refer to Chapter 4*.*

2.2.1 - Declaring Mscreen

To operate multiscreen features, SCO UNIX/XENIX requires:

- information about the selected terminal emulation (TERM variable) and
- data from the `/etc/mscreencap` file to describe the multiscreen terminal command sequences.

As SCO UNIX/XENIX does not perform the multiscreen function on an ANSI terminal as a standard feature, the `/etc/mscreencap` file must be modified to include an ansi entry.

i) Become superuser using the `su` command.

ii) Use the `vi` editor:

```
# vi /etc/mscreencap
```

iii) Add the following lines at the end of the file:

```
ansi:\
:who,Alt-F9,\E[<i,:\
:help,Alt-F10,\E[<j,:\
:stop,Alt-F11,\E[<k,\E[0z:\
:quit,Alt-F12,\E[<l,\E[0z:\
:,Alt-F1,\E[<a,\E[1z:\
:,Alt-F2,\E[<b,\E[2z:\
:,Alt-F3,\E[<c,\E[3z:\
:,Alt-F4,\E[<d,\E[4z:\
:,Alt-F5,\E[<e,\E[5z:\
:,Alt-F6,\E[<f,\E[6z:\
:,Alt-F7,\E[<g,\E[7z:\
:,Alt-F8,\E[<h,\E[8z:
```

Note: The above characters strings are available in the `axcap` file which is included on the AXEL multiscreen diskette. As an alternative to editing the file you can copy this file to your hard disk, then enter the following command at the UNIX prompt to update the mscreencap file:

```
# cat axcap >> /etc/mscreencap <RETURN>
```

WARNING: Make sure you type the > character twice or you will overwrite the `mscreencap` file!

iv) Save the `/etc/mscreencap` file when it has been modified.

v) Exit superuser (type <`Ctrl`><D> or `exit`).

2.2.2 - Setting Mscreen

The multiscreen uses a "pseudo-terminal" for each screen. As a pseudo-terminal is considered to be a device, UNIX will treat every multiscreen session as if it were linked to a physically independent serial line.

The pseudo-terminals used by `mscreen` have the following generic names:

```
/dev/ttypx,
/dev/ttyqx,
/dev/ttyrx,
/dev/ttysx
```

(where x stands for any hexadecimal character from 0 to f)

a - Set-Up on SCO UNIX 3.2v2 or SCO XENIX

You set up a pseudo-terminal in the same manner as you set up a normal serial port. Use the `enable` command, with the pseudo device filename, **for each device** used, as follows:

```
# enable ttyp0 <RETURN>
# enable ttyp1 <RETURN>
# enable ...
```

Note: if you need more than 64 pseudo-terminal devices, you will need to create additional devices. See the `mkdev` command, in the SCO UNIX handbooks, for instructions (the maximum number of pseudo-terminals supported by SCO UNIX is 256).

b - Set-Up on SCO UNIX 3.2v4 and Release 5

No pseudo-terminals are configured in the default SCO UNIX configuration. However, there is provision for creating pseudo-terminal devices and linking the device drivers to the kernel. Use the following command:

```
# mkdev ptty <RETURN>
```

It is then necessary to reboot the system (`shutdown` command) before the `mscreen` software can be used.

2.2.3 - Using Mscreen

To use multiscreen, run the command:

```
$ mscreen -n x <RETURN>
```

where ***x*** is the required number of screens (from 1 to 8).

This command can be invoked by a script (e.g. `.profile`) or from the UNIX shell.

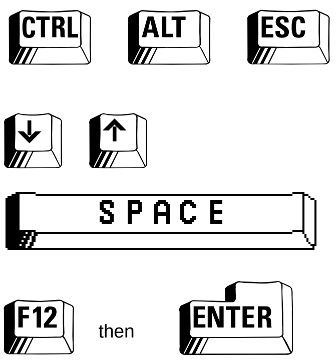
When multiscreen is set, the operational key-stokes are:

- **<Alt><F1>** to **<Alt><F8>**: move to another screen. The screen number corresponds to the function key number. Thus, **<Alt><F4>** moves you to screen 4.
- **<Alt><F9>**: display the current pseudo-terminal name
(similar to the `who` command).
- **<Alt><F10>**: display available commands (help),
- **<Alt><F11>**: exit from the `mscreen` software with a non zero return code.
- **<Alt><F12>**: exit from the `mscreen` software with a zero return code.
- **<Ctrl><PrtSc>**: move to the next screen (view 1 to view 2, view 2 to view 3, ... view 8 to view 1).

By default, the `mscreen` software supplies 8 views. It is possible to limit the number of views to less than this by setting the `-n` option to less than 8. For example, set only 4 screens with the following command:

```
$ mscreen -n 4 <RETURN>
```

The **<Ctrl><PrtSc>** keystroke invokes an automatic move to the next screen (from view 1 to view 2, ... from view 8 to view 1). For this to function correctly with less than 8 views, the number of views actually used must be declared using the terminal set-up procedure:



(enter terminal set-up)

(select the number of views multiscreen parameter)

(select the number of views)

(save current setting)

For additional information, refer to SCO UNIX handbooks:

User's Reference(C)	: ENABLE, DISABLE
User's Reference(M)	: MSCREEN
User's Reference(HW)	: SCREEN

2.3 - LOCAL PRINTER

A local printer is one that is directly connected to the terminal. It is convenient for the operator and avoids the need for an additional printer serial cable from the host computer.

Note: Printers usually accept data more slowly than terminals (9.6 Kbaud vs. 38.4 Kbaud) and the printer may send an XOFF signal to prevent data overflowing its buffer. As the local printer and the terminal are sharing the same serial line, this signal will shut off the data flow for **both** peripherals. This does not impair the data flow to and from the terminal but slows it down. **The maximum transfer rate of both peripherals is reduced to the maximum transfer rate of the slower one: the printer.**

AXEL can provide a proprietary solution to this problem: the intelligent AXEL multi I/O controller board. This **optimises the data flow** handling and allows **the local printer to be declared as a system printer** (and to be used with the `lp` command). Only a single serial connection is required. Configuration of this device is described in the *Installation Guide for V605/V610/V810 controllers for SCO UNIX*.

The following section describes the standard connection between the terminal and a local printer, when used **without** the Axel proprietary solution.

2.3.1 - Installing a Local Printer

Follow the instructions in chapter 1.3 of this guide.

2.3.2 - Setting and Using a Local Printer

A local printer can be used in two ways:

- printing the data on the terminal screen

- printing files from the host system (`lprint` command).

a - Print Screen

This is performed locally, by the Platine, without host intervention. No operating system configuration is necessary.

To use the local print screen feature, press **<PrtSc>** key.

b - Printing Files

Files are printed from the host with the `lprint` command. To use this command with a local printer, **a modification has to be made to the configuration of the SCO UNIX/XENIX system.**

The Platine manages the local printer by identifying received data that is intended for the printer, and separating it from data destined for the screen.

This identification requires 'special characters' to be added when transmitting the data to the terminal and to the printer on the same serial line. The printer data will be preceded with a 'start local printing sequence' and followed by a 'stop local printing sequence'.

For the AX3000 Platine, in SCO UNIX/XENIX personality, the respective sequences values are:

- start local printing sequence: `Esc [5i`
- stop local printing sequence: `Esc [4i`

The standard **ansi** emulation does not include theses two sequences. The section of the `/etc/termcap` file that refers to the **ansi** emulation must be modified.

i) Become superuser (using the UNIX `su` command) and edit the `/etc/termcap` file with the `vi` editor.

ii) Look for the ANSI terminal description. It begins with a line containing the word '**ansi**' and ends with the first line that does **not** end with a `\`**.**

```
li|ansi|xxxxx|xxxx:\
:xx:xxxx:xxx:\
:xxx:xx:xxx:\
:xx:xxx:xxx:
```

iii) Insert the following line in the description:

```
:PN=\E[5i:PS=\E[4i:\
```

Note: this line should be inserted anywhere in the ansi section of the file **except** before the first line or after the last line.

iv) Exit from `vi` and save the `/etc/termcap` file.

v) Exit superuser (type `<Ctrl><D> or exit`).

The local printer is now configured. To print a file on the local printer, issue the following command from the terminal where the local printer is connected (where `filename` is the full or relative UNIX pathname of the file to be printed):

```
$ lprint filename <RETURN>
```

Once printing has started, it is recommended that the terminal keyboard should not be used.

If the printer does not perform the line feeds, enter the following lines, at the UNIX prompt, before printing:

- using Bourne shell or Korn shell:

```
$ FORMS=X <RETURN>
$ export FORMS <RETURN>
```

- using C shell:

```
% setenv FORMS X <RETURN>
```

These commands can be added to the user's `.profile` file.

For additional information, see the following entries in the SCO UNIX manuals:

User's Reference(ADM)	: LPRINT
User's Reference(C)	: LP
User's Reference(F)	: TERMCAP
System Administrator's Guide	: Adding a Local Printer

2.4 - USING SCANCODE

SCO UNIX 3.2v4 and SCO UNIX Release 5 introduced the terminal keyboard scancode mode. Previous versions only used ASCII mode.

Scancode mode can be controlled in three ways:

- permanent setting: the scancode is turned on during login.
- temporary setting: the scancode mode is set during a specified period during a session.
- for the duration of software use: the scancode will only be active while the specified software is running.

Warning: scancode mode requires the terminal to use **DTR** or **XPC** handshaking (instead of XON/XOFF). Check the connected host serial port is using the same handshaking. (Handshaking is sometimes also known as 'flow control').

Note: do not use the multiscreen feature from a scancode terminal.

2.4.1 - Setting for Permanent Use

a - Platine Terminal Set-Up

The AX3000 Platine set-up must be modified to use scancode mode:

CTRL ALT ESC (enter terminal set-up)

F2	(keyboard option)
SPACE	(select scancode mode)
F12 then ENTER	(save current setting)

Note: if flow control was set to XON/XOFF, selecting scancode mode will automatically alter it to XPC flow control.

b - SCO UNIX 3.2v4 Operating System Set-Up

The `/etc/gettydefs` file must be modified to create an entry for scancode mode:

i) Become superuser and edit the `/etc/gettydefs` file with the `vi` editor. This file already contains an entry for a 9.6 Kbaud scancode terminal mode, as a standard feature (invoked by `sc_m`). This line can be copied using the `vi` editor, then edited for different baud rates.

For example, the following line must be added to `/etc/gettydefs` for a 38.4 Kbaud scancode terminal:

```
sc_o # B38400 HUPCL SCANCODE # B38400 CS8 SANE HUPCL
TAB3 ECHOE IXANY SCANCODE # \r\n@!login: # sc_o
```

ii) Configure the `/etc/inittab` file to associate a serial line with the scancode entry in `/etc/gettydefs`.

First, disable the serial port which is to be modified (use the `disable` command).

Then, with the `vi` editor, search for the appropriate line in the `/etc/inittab` file and replace the coding character (for example o) with `sc_o`:

```
se1a:23:off:/etc/getty tty1a sc_o
```

Save these modifications and exit from the editor. Then turn the serial line on again, with the enable command.

When set, the scancode mode is only operational for QWERTY keyboards. To use other keyboards (for example AZERTY), alternative encoding tables must be used. The `/usr/lib/keyboard` directory contains scancode encoding tables for every keyboard. For example, the file for use with a French keyboard is `ps.ibm.fra`.

During scancode set-up, it is possible to associate a serial port with a dedicated keyboard, by an entry in the `/etc/default/mapkey` file**.**

Each line in this file contains a serial port name and a keyboard type, separated by white space (space or tab character). For example, with a terminal connected to COM1 and using the scancode mode with a French keyboard, the following line would be added to the `/etc/default/mapkey` file:

```
tty1a     ps.ibm.fra
```

Add a line to the file for every serial port which has a terminal with a dedicated scancode keyboard.

These modifications will be registered only after the host has been rebooted (`shutdown` **command)**.

If changes to the `/etc/default/mapkey` file have still not been registered after a reboot, enter the following command:

```
$ mapkey -a <RETURN>
```

This command **must be entered before** using a scancode mode terminal.

2.4.2 - Setting for Temporary Use

In some situations it is better to have the terminal set to ASCII mode, with scancode mode set only when required.

This requires a three-stage process:

a) Modify the ANSI Emulation

The standard ANSI emulation used by the AX3000 Platine does not include the choice of keyboard mode (ASCII↔scancode) and flow control (XON/XOFF↔XPC).

`Terminfo` uses compiled binary files. Therefore, to modify this emulation, become superuser and enter the following at the UNIX prompt:

```
# cd /usr/lib/terminfo <RETURN>
# infocmp ansi > ansi.src <RETURN>
```

This creates a text file, `ansi.src`, which can be edited::

```
# vi ansi.src <RETURN>
```

Add the following line at any position except the beginning or end of the file:

```
smsc=\E[<0A, rmsc=\E[<1A, xonc=e, xoffc=g,
```

Exit from the editor and save these modifications, then enter the following command at the UNIX prompt to recompile the `terminfo` binary:

```
# tic ansi.src <RETURN>
```

Quit superuser mode (`exit` command).

This modified ANSI emulation will now support switching, between ASCII and scancode modes, when using suitable flow control.

b) Updating the `/etc/ttytype` *File*

You <u>must</u> now associate an emulation with a device.

For example, in the case of an AX3000 Platine (ansi emulation) connected to COM1, you must add the following line to the `/etc/ttytype` file (unless it already exists):

```
ansi     tty1a
```

c) Changing the Keyboard Mode

To change from ASCII mode to scancode mode, use the `scanon` command.

By default, the scancode is set for a QWERTY keyboard. To modify the keyboard type (for example to AZERTY), use the `mapkey` command, followed by the corresponding encoding filename (located in the `/usr/lib/keyboard` directory).

For example, to set the scancode mode for a French AZERTY keyboard, enter the following command:

```
$ mapkey /usr/lib/keyboard/ps.ibm.fra <RETURN>
```

To simplify the procedure, both the `scanon` and `mapkey` commands can be consolidated into a single script.

Enter the `scanoff` command to return to ASCII mode.

2.4.3 - Setting for Use with Microsoft WORD

Microsoft WORD software can modify the keyboard mode by using a (`/usr/lib/word/termscan`) parameters file. The content of this file is similar to `/etc/termcap`.

To use Word from a terminal scancode keyboard, the **ansi** entry of the `termscan` file must be modified. This entry begins with a line containing the word ansi and ends at the next line that does not end with a \.

Edit the `/usr/lib/word/termscan` file and add the following line at any position (except the beginning or end of the ansi description):

```
ZO=\E[<0A:ZF=\E[<1A:XN=\145:XF=\147:\
```

Exit from the editor and save the modifications.

The terminal will operate with the keyboard in ASCII mode for all other applications. When Microsoft WORD is loaded, it will automatically set the keyboard to scancode mode and will restore ASCII mode when it terminates.

- 3 - SETTINGS FOR IBM AIX RS/6000

This chapter describes various steps to configure AXEL Platine terminal capabilities for IBM AIX RS/6000.

3.1 - SOFTWARE INSTALLATION

The AXEL Platine terminal set-up includes two major steps:

- Setting terminal parameters,
- configuring the operating system.

3.1.1 - Setting Terminal Parameters

The built-in AXEL set-up automatically sets all standard terminal parameters for the selected operating system.

CTRL ALT ESC	(enter terminal set-up)
F10	(select predefined set-up)
↑ ↓ then ENTER	(select the ANSI RS6000 predefined set-up)
F12 then ENTER	(save selected predefined set-up)

The AX3000 Platine is now ready to operate under IBM AIX RS/6000 at 38.4 Kbaud. If necessary, terminal parameters can be modified through the terminal set-up sequence (e.g. baud rate). For additional set-up information refer to the "*Asynchronous AX3000 Models - User's Guide*".

3.1.2 - Setting the Operating System

To declare the AX3000 Platine as a standard serial terminal, **set the** `TERM` **environment variable to a value that matches the selected Platine terminal emulation.**

With the RS/6000 platform, the Platine terminal uses a **colour ANSI** emulation**.** The TERM value must be set to:

- `hft-c` for IBM AIX 3.1,
- `hft-c-old` for IBM AIX 3.2 or 4.x.

To set the terminal parameters, use the System Management Interface Tool (`smit`).

Become superuser (`su` command) and enter the following command:

```
# smit tty <RETURN>
```

Select the `'Add a TTY'` option (or `'Change/Show characteristics of a TTY'` to modify features of a line which has already been declared).

Select the serial port type (RS232 or RS422), the parent adapter and the port number. The main set-up screen is now displayed.

On this screen, the default AX3000 terminal parameters for the ANSI RS/6000 predefined set-up are:

- Transfer rate:	38.4 Kbaud
- Parity:	none
- Bits per character:	8
- Stop bits:	1
- Terminal type:	hft-c (or hft-c-old)

To save these settings and return to the `smit` menu, enter `<RETURN>.`

The AIX IBM RS/6000 system set-up is now completed.

Note: The communication parameters (for example, the transfer rate) can be modified. Such modifications must be done in the AX3000 Platine terminal set-up and in the `smit` utility on the host.

3.1.3 - Using Compose Characters

A compose character is formed by the combination of several (usually two or three) keystrokes. For example, the '**â**' character is displayed by pressing the **'^'** key followed by the **'a'** key.

With **ANSI** emulation, the compose character capability **is not performed by the terminal itself**. Compose character keystroke sequences must be defined in the operating system. The compose character scheme uses a parameters file (mapfile). This mapfile must be created in the `/etc/nls/termmap` directory and has an `'in'` suffix in most cases

With the `vi` editor, create the `ax3000.in` mapfile in the `/etc/nls/termmap` directory. This mapfile contains the following characters strings:

```
# â ê î ô û
\xb0\x61:\x83
\xb0\x65:\x88
\xb0\x69:\x8c
\xb0\x6f:\x93
\xb0\x75:\x96
# à è ì ò ù
\x60\x61:\x85
\x60\x65:\x8a
\x60\x69:\x8d
\x60\x6f:\x95
\x60\x75:\x97
# ä ë ï ö ü ÿ Ä Ö Ü
\xb1\x61:\x84
\xb1\x65:\x89
\xb1\x69:\x8b
\xb1\x6f:\x94
\xb1\x75:\x81
\xb1\x79:\x98
\xb1\x41:\x8e
\xb1\x4f:\x99
\xb1\x55:\x9a
# ñ Ñ
\x7e\x6e:\xa4
\x7e\x4e:\xa5
```

There are two ways to use this mapfile and form compose characters.

a) The Setmaps Utility

This is the easiest way. Enter the following command at the beginning of every session:

```
$ setmaps -i ax3000.in <RETURN>
```

This command can be inserted in each user's login script (`.profile` file).

b) The SMIT Utility

This procedure assigns the `ax3000.in` mapfile to the `tty` device file used by the Platine terminal.

The `smit` tool demands listed mapfile names. As the `ax3000.in` mapfile is not listed, it must be ‘fooled’ with the name of a mapfile already known to `smit` (for example vt220.in).

To perform this function, become superuser and enter the following commands at the UNIX prompt:

```
# cd /usr/nls/termmap <RETURN>
# mv vt220.in vt220.SAV <RETURN>
# mv ax3000.in vt220.in <RETURN>
```

Then, invoke the utility:

```
# smit tty <RETURN>
```

Display the serial line setting screen used by the Platine.

Assign the vt220 value to the 'Input mapfile' item:

```
...................         ....
Input mapfile               [vt220.in]
Output mapfile              [none]
....................        ....
```

To save the current setting, press `<RETURN>`.

3.2 - LOCAL PRINTER

A local printer is one that is directly connected to the terminal. It is convenient for the operator and avoids the need for an additional printer serial cable from the host computer.

3.2.1 - Setting a Local Printer

Follow the instructions listed in section 1.2 of this guide.

3.2.2 - Using the Local Printer

On IBM **AIX 3.1**, the local printer **cannot** be used as a system printer. The use of the local printer is described in section 1.2.

On IBM **AIX 3.2**, the local printer can be assigned as a system printer and controlled through the operating system.

The local printer is declared in the `smit` utility and a local print queue is assigned to the device (`tty`) used by the Platine.

The `terminfo` emulation must include the `mc5` and `mc4` capabilities, to define start and stop local printing sequences.

As the standard `hft-c` (or `hft-c-old`) emulation does not include these sequences, the `/usr/share/lib/terminfo/ibm.ti` file (which describes the `hft-c` emulation) must be modified.

- Become superuser.

- Change to the `/usr/share/lib/terminfo` directory

- Edit the `ibm.ti` file with the `vi` editor.

- Look for the `hft-c-old` terminal description:

```
hft-c-old,xxxxx,xxxx,
xx,xxxx,xxx,
xxx,xx,xxx,
xx,xxx,xxx,
```

- Insert the following line in the description:

```
mc5=\E[5i, mc4=\E[4i,
```

This line must not be the first one or the last one in the terminal description.

- Save the modified file and exit from `vi`.

- Recompile the terminfo definition with the command:

```
# tic ibm.ti <RETURN>
```

- Exit superuser mode.

The printer is now ready to be used with the `lp` command.

- 4 -
MULTISCREEN UTILITY

This chapter describes the installation and the use of `axmscreen`*: the multiscreen software for the AXEL asynchronous AX3000 Platine terminal.*

4.1 - INTRODUCTION

The Multiscreen capability lets the user run up to 8 concurrent sessions (or screens) on one serial AX3000 serial terminal and switch between them by use of hot keys (function key combinations). The Multiscreen is handled, at the UNIX level, by AXEL `axmscreen` software, shipped with all serial Platine terminals.

The Platine Multiscreen feature can be used in two different ways:

- **multishell**: a new shell is launched at the start of each of the 8 sessions,
- **multilogin**: a login (requiring the user's name and password) is performed for each of the 8 sessions. This allows a different user account on each of the available screens.

The `axmscreen` software provides multiscreen features for most UNIX systems:

- SCO UNIX (3.2v2, 3.2v4, Release 5 and XENIX),
- IBM AIX,
- AT&T UNIX (Interactive, UNIXWARE, etc.),
- HP 9000,
- SINIX (Siemens),
- AViiON (Data General),
- MOTOROLA.

The multiscreen uses a pseudo-terminal (ptty) for each screen. For using `axmscreen`, the pseudo-terminal function has to be declared within the UNIX kernel (refer to your UNIX literature).

`axmscreen` runs only on the AXEL Platine terminal and stores details of each session in the Platine's local memory. It makes no demands on the host and does not use up **host memory.**

4.2 - INSTALLATION

Insert the floppy diskette provided. Log in as superuser and enter the following commands:

```
# cd /usr/bin
# tar xvf /dev/fdx axmscreen.OS
```

`fdx` is the local name for the floppy disk drive device and `axmscreen.OS` is one of the following, corresponding to the host operating system:

- axmscreen.AIX	(for IBM AIX)
- axmscreen.SCO3	(for SCO UNIX 3.2v2 and XENIX)
- axmscreen.SCO4	(for SCO UNIX 3.2v4 and SCO UNIX Release 5)
- axmscreen.UW	(for AT&T UNIX)
- axmscreen.DG	(for AViiON)
- axmscreen.SNX	(for SINIX)
- axmscreen.HP	(for HP 9000)
- axmscreen.MOTO88	(for MOTOROLA 8800)
- axmscreen.MOTO68	(for MOTOROLA 6800)
- axmscreen.INTER	(for UNIX INTERACTIVE)

When the required file has been copied onto your hard disk, rename it `axmscreen`. (Enter the command as follows, but with "`OS`" replaced by the operating system suffix):

```
# mv axmscreen.OS axmscreen
```

Because it is dedicated to the Platine terminal, the software is immediately operational (with no need for configuration files, nor for the system to be rebooted).

4.3 - USING AXMSCREEN

The `axmscreen` command can be invoked either at the UNIX prompt or from a script (e.g. `.profile` file), followed by possible options (see chapter 4.4).

When running `axmscreen`, the operational keystrokes to move between screens are:

- **<Alt><F1>**: move to view 1
- **<Alt><F2>**: move to view 2
- ... ...
- **<Alt><F7>**: move to view 7
- **<Alt><F8>**: move to view 8
- **<Ctrl><PrtSc>**: move to the next view (view 1 to view 2, view 2 to view3, ... last view to view 1).

The other available keystrokes are:

- **<Alt><F9>** (who): display the number of the current view (from 1 to 8) and the current pseudo-terminal name.
- **<Alt><F10>** (help): display available commands and associated keystrokes.
- **<Alt><F11>** (quit): terminate the `axmscreen` program with a non-zero return code.
- **<Alt><F12>** (exit): terminate the `axmscreen` program with a zero return code.

Notes:

- When using VT220 emulation, <Alt> must be replaced by <Ctrl> in all the above keystrokes.
- <Alt><Fx> are the default keystrokes. This setting can be modified through the AX3000 set-up

4.4 - THE AXMSCREEN OPTIONS

When invoking `axmscreen`, the option syntax is:

```
    axmscreen [-bcefhklmnpstvwx] [file] [x]
```

Each option is described below:

4.4.1 - '-b' Option: Select Pttys

The `axmscreen` software uses pseudo-terminals (pttys). A pseudo-terminal is a device file. The filenames used for pseudo terminals differ between different operating systems.

The **'-b'** option is used to set the generic name of pseudo-terminals and must be followed by a parameter. There are three possible values:

- -**'b 0'**: use with `ttypx` (where x is any decimal value from 0 to 63),
- -**'b 1'**: use with `ttypy` (where y is any hexadecimal value from 0 to 3Fhex),
- -**'b 2'**: use with `ttypz, ttyqz, ttyrz and ttysz` (where z is any hexadecimal value from 0 to F).

By default, the axmscreen uses the **'-b 0'** option.

Note: this option is not compatible with use of the **'-p'** option (see section 4.4.10) and not available for AT&T UNIX.

4.4.2 - '-c' Option: Overscan Colour

When `axmscreen` software is active, the **<Alt><F9>** keystroke may be used to identify the current view and will display its number (from 1 to 8).

Another way to identify the current view is to associate a different overscan colour with each view. This can be done using the **'-c'** option.

The following table lists views and associated overscan colours:

View	Colour
1	Blue
2	Light yellow
3	Green
4	Cyan
5	Red
6	Magenta
7	Yellow
8	White

4.4.3 - '-e' Option: Exit Disabled

<Alt><F11> or **<Alt><F12>** keystrokes are used to exit the software. The **'-e'** option can be used to disable these keystrokes and prevent accidental exit from the program.

To delete the processes which have been created by `axmscreen`, use the following command:

```
# kill -s 15 pid
```

where `pid` is the process ID of the `axmscreen` parent process (in most cases, the axmscreen parent is a shell).

Note: use signal 15 (and not signal 9) with the `kill` command.

4.4.4 - '-f' Option: Initial Scripts

When `axmscreen` uses multishell, an initial command script can be automatically launched within each of the views.

This list of command scripts (one per view) is stored in a file whose filename is provided as the argument to the **'-f'** option.

This command file has a maximum of 8 lines. Each line lists the command associated with one view:

1st line	view 1
... ...	...
8th line	view 8

If a view does not have an initial command, its associated line must start with a dash character (2Dhex character).

Example: the cmd file:

```
smit
-
client
```

This file is used through the following command:

```
$ axmscreen -f cmd
```

In this example, the `smit` utility is invoked within view 1 and the client application within view 3. Views 2, 4, 5, 6, 7 and 8 do not have initial commands. Instead the shell remains active.

4.4.5 - '-h' Option: On Line Help

This option displays the `axmscreen` syntax and gives a brief description of every option.

4.4.6 - '-k' Option: Comments

With `axmscreen`**,** a status line can be used to list the available views and the active view (see **'-s'** option description, section 4.4.11). The **'-k'** option specifies a character string which will be displayed on the right side of the status line.

Example:

```
NAME=`tty`
axmscreen -n 4 -s -k $NAME
```

4.4.7 - '-l' Option: Multilogin

The multiscreen feature of `axmscreen` software can be used in multishell or multilogin mode. With multilogin, a login (user's name and password) is invoked for each of the 8 sessions.

The multilogin feature is set with the **'-l'** option.

Note: it is not possible to use both login and script (multishell) views simultaneously on the same terminal The **'-l'** and **'-f'** options are not compatible on a single terminal.

The use of the multilogin varies between different UNIX operating systems.

Note: the multilogin feature of `axmscreen` is not available on SIEMENS and HP 9000 platforms

a - SCO UNIX 3.2v4 and Release 5

On SCO 3.2v4 (or SCO Release 5), setting the login is protected and does not use the `/etc/inittab` file.

Several indirections are performed and the login is set with the `/etc/utmp_getty` file. The access permissions of this file are listed in the file `/tcb/files/no_luid/cmdtable`. To allow multilogins, this file must be modified.

Edit the following line in `/tcb/files/no_luid/cmdtable`:

```
utmp_getty:/etc/utmp_getty:root
```

Replace `'root'` by '`*`' (2A hex character). Save the modified file and exit the text editor.

Check the operating system release level (0 or later) before running the multilogin program. To use multilogin with SCO UNIX **3.2v4.0** the **'-v'** option is required.

For later versions, do not use this option.

Example:

```
    axmscreen -n 4 -l -v  for SCO UNIX 3.2v4.0
    axmscreen -n 4 -l     for SCO UNIX 3.2v4.2
```

b - AT&T UNIX

Under this operating system, the login is controlled by a port monitor `ttymon`. Port monitors can only be invoked by the super-user.

Super-user authority can be obtained by executing the `su` command but, by default, this requires entry of the root password. This password request procedure can be by-passed by modifying the file `/etc/default/su`.

Edit `/etc/default/su` with `vi` and add the following line at the end of the file:

```
PROMPT=No
```

Save the file. You can now use the multilogin feature with `axmscreen` software.

Note: removal of password protection from the `su` command can make the system vulnerable to unauthorised interference. Only follow this procedure if you are sure that other safeguards are adequate.

c - Other UNIX

The login feature uses a 'background' process which waits for a user name on each view. This process is invoked by executing `/etc/getty`. A `getty` must be run for each pseudo-terminal in use. A list of every `getty` running can be obtained with the following command:

```
# ps -ef | grep getty
```

If `getty` processes are not running for any of the pseudo-terminals in use, the file `/etc/inittab` should be modified.

<u>All pseudo-terminals which require a login must be listed in this file.</u>

For each pseudo-terminal, add the following line:

```
ptyx:2:respawn:/etc/getty /dev/ttypx m
```

where x stands for the pseudo-terminal number.

Save the file, then, enter the command `init q` which will immediately initialise the required `gettys` and create logins.

Check these modifications have been correctly performed by repeating the command `ps -ef | grep getty`.

4.4.8 - '-m' Option: Silent Mode

By default, help messages (number of views, active keys, etc.) are automatically displayed when `axmscreen` is launched.

This **'-m'** option suppresses these messages.

4.4.9 - '-n' Option: Number of Views

By default, the `axmscreen` software supplies 8 views. It is possible to limit the number of views to less than this by setting the **'-n x'** option, where **x** stands for the number of views (from 1 to 8).

Example:

```
$ axmscreen -n 4
```

The **<Ctrl><PrtSc>** keystroke invokes an automatic move to the next screen (from view 1 to view 2, ... from view 8 to view 1). For this to function correctly, with less than 8 views, the revised number of views in use must be declared using the terminal set-up procedure.

Enter set-up mode and proceed as follows:

CTRL ALT ESC	(enter terminal set-up)
↓ ↑	(select the number of multiscreen views)
SPACE	(select the number of views)

F12 then ENTER (save this configuration)

4.4.10 - '-p' Option: Predefined Pseudo-Terminals

It is possible to use a fixed set of pseudo-terminals, which are listed in a parameter file which is named as the argument to the **'-p'** option. Each line of the file lists the ptty dedicated to its view (8 lines maximum):

1st line	view 1
2nd line	view 2
... ...	...
8th line	view 8

Example: the `predef` file

```
ptyp2
ptyp5
ptyp6
ptyp8
```

The command line would be as follows:

```
$ axmscreen -p predef
```

In this example, 4 views are defined by the `predef` file: ptyp2 is assigned to view 1, ptyp5 is assigned to view 2, ptyp6 is assigned to view 3 and ptyp8 is assigned to view 4.

Notes: the **'-p'** option overrides the **'-n'** option (for setting the number of views). Thus, for example, the command `'axmscreen -n 8 -p predef'` will only create 4 views.
This option is not available with the AT&T UNIX System.

4.4.11 - '-s' Option: Status Line

The `axmscreen` can use line 25, at the bottom of the screen, to display a status line which specifies the active session and the number of declared views.

Note: a comment can be added on the status line with the **'-k'** option (see section 4.4.6).

Generally, ansi entries of a terminfo file specify the behaviour of a 25-line terminal. To use the status line feature, a terminfo entry for a 24-line terminal must be created (for more information, refer to Chapter 4.5.3).

4.4.12 - '-t' Option: No tty Test

During its initialisation, the `axmscreen` tests for the type of tty on which it is executed. If this tty type is recognised as a pseudo-terminal, the software terminates.

This test is done by default to prevent the `axmscreen` software operating on an `axmscreen` view.

However this test can be prevented with the **'-t'** option. This option might be used, for example, when the Platine terminal is linked to a terminal server on an Ethernet network, which uses pseudo-terminal devices.

4.4.13 - '-v' Option: SCO UNIX 3.2v4.0 only

This **option is COMPULSORY when** the **'-l'** (multilogin) option is used under SCO UNIX 3.2v4.0. For additional information, see section 4.4.7.

4.4.14 - '-w' Option: No Shell Regeneration

This option is only for use with the multishell feature.

Within multishell mode, the active shell is terminated by the `exit` command or when <Ctrl><D> is pressed. This shell is automatically regenerated (and may then run an initial script) by the `axmscreen` software.

Shell regeneration can be prevented with the **'-w'** option. When the **'-w'** option is set, a terminated shell is not regenerated and the related view becomes inaccessible. The view is labelled as 'killed' when the '`who`' command is run (<Alt><F9> keystroke), and disappears from the optional status line.

4.4.15 - '-x' Option: Circular Buffers

A view is considered as inactive when it is not the current view. Within a session, when *n* views are used, there is always one active view (the current one) and *n-1* inactive views.

By default, it is impossible to display data on an inactive view. When an inactive view receives data, it is immediately locked (using XOFF) so that the flow of data is stopped. Data transmission is not resumed until it is unlocked, when it becomes active again.

The **'x'** option is used, to avoid these locks and the interruption of data transmission, by assigning a circular buffer to each view. When this option is set, data transmitted to an inactive view is not stopped but is, instead, stored in the proper circular buffer. When the view is reactivated, the buffer contents are displayed.

By default, the size of each circular buffer is 2 Kbytes. This size can be modified using the **'-x n'** option where **n** stands for the buffer size (from 1 KB to 9 KB).

The circular buffer feature saves only the latest data transmitted to an inactive view. If the volume of data transmitted exceeds the buffer limits, incoming data may overwrite the existing contents of the buffer and cause incomplete data display when the view is reactivated (see chapter 2.3).

4.5 - USER NOTES

4.5.1 - Multilogin or Multishell

The available modes of the `axmscreen` are as follows:

- **multishell**: a new shell is launched at the start of each of the 8 sessions,
- **multilogin**: a login (requiring the user's name and password) is performed for each of the 8 sessions.

The multishell mode allows the use of initial scripts. Then, different software packages can be run automatically on each screen.

The multilogin mode allows a different user account on each of the available screens. This mode does not work on all UNIX Systems because the login feature is sometimes protected.

4.5.2 - Add a Multiscreen User

The multiscreen is invoked through a simple command. This command can be inserted within a script to launch the multiscreen automatically.

For example, a new user can be created, and the `axmscreen` command can be added at the end of his `.profile` file:

```
...
axmscreen -n 4 -s -f batch
exit
```

When this user logs in, the multiscreen feature is automatically run. When the `axmscreen` is terminated (<Alt><F11> or <Alt><F12>) the user is automatically logged out.

4.5.3 - Status Line

A status line feature can be used through the **'-s'** and **'-k'** options. This option makes it possible to visualise the active session and the number of declared views.

Emulation problems can occur with the use of this status line.

This status line leaves only 24 lines available for use by applications. A `terminfo` file must therefore be used that specifies the behaviour of a 24-line terminal. Terminfo files describe the behaviour of each type of terminal. They are to be found in the `usr/lib/terminfo` directory (or the `usr/lib/terminfo` directory) and end with a `.src` or `.ti` suffix.

If your UNIX operating system does not provide a `terminfo` file with a 24-line terminal, you must modify an existing `terminfo` file.

Create a source (`.src`) file using the `infocmp` command as described above. Edit the source file with the `vi` editor. Set the proper variable (`li#` or `lines#` depending on the UNIX version) to 24 lines. Save these modifications.

The updated source file must be recompiled. Invoke the following command (where `filename.src` is the name of the modified source file):

```
# tic filename.src
```

The 25th status line option must match the ‘Number of Lines’ parameter of the Platine terminal set-up.

This parameter can take either of two values:
- 25 lines,
- 24 + 1 message line.

When the `axmscreen` software is launched with the **'-s'** option, the 'Number of Lines' parameter is set to '24+1 message line' without saving the current value.

The 'Number of Lines' parameter can be modified as follows:

CTRL ALT ESC	(enter terminal set-up)
↓ ↑	(select the 'number of lines' parameter)
SPACE	(select '25 lines' or '24 + 1 message line')
F12 then ENTER	(save this configuration)

4.5.4 - Exit Multiscreen

The standard way to exit `axmscreen` is to use the <Alt><F11> or <Alt><F12> keystrokes.

The **'-e'** option allows these keystrokes to be disabled and prevents accidental exit from `axmscreen`. Then the only way to exit `axmscreen` is to kill the corresponding process.

The following script (named `end_axmscr` on the AXEL diskette) performs this `kill` command:

```
echo "*** axmscreen terminator"
if [ -z "$DAD_AX" ]; then
   echo -n "Enter the tty name:"
   read DAD_AX
fi
PID_AX=`ps -t $DAD_AX | grep "axmscr" | awk '{print $1}' `
if [ -z "$PID_AX" ]; then
   echo "Nobody to kill !!!"
   else
   echo "kill -s 15 "$PID_AX
   kill -s 15 $PID_AX
fi
```

The `DAD_AX` variable can be set within the `.profile` file:

```
DAD_AX=`basename \`tty\``
```

Run this script from one of the multiscreen views. If the `DAD_AX` variable is not set, you must enter the name of the `tty` associated with the Platine terminal (`tty1a` for example).

IMPORTANT: Only use this script within multishell mode.

4.6 - LIMITATIONS OF THE ANSI STANDARD

If an output escape sequence is interrupted, by another output escape sequence or by any unexpected data, it may leave the terminal display in an indeterminate state. In ANSI mode, the characters of any unknown escape sequence (for example an interrupted or incomplete sequence) are displayed (and not ignored).

This ANSI characteristic may cause multiscreen display problems in the two following cases:

- **Moving between views**: a switch between screens (**<Alt><Fx>** keystroke) may interrupt an output escape sequence and disturb view contents. To prevent this problem, only **change screens when the display is stable.**

- **Using buffers ('-x' option)**: When a new screen is activated, the `axmscreen` software displays the contents of the corresponding circular buffer. If the first escape sequence within the circular buffer has been interrupted, in the course of circular data processing, the corresponding screen display is corrupted. This can happen, for example, if a circular buffer is too small, the inactive view receives too many characters and older data is overwritten. To prevent this problem, **use larger circular buffers ('-x n' option)**.

AXEL

Zone d'activité d'Orsay-Courtabœuf
16 Avenue du Québec - BP 728 - 91962 LES ULIS Cedex
Tel.: (33) 1 69 28 27 27 - Fax: (33) 1 69 28 82 04