

# Installing/Configuring Foxpro 2.6 for Unix and Linux by Dennis Allen

## Table of Contents (01/09/04)

[Disclaimer \\*](#)

[Installing/Configuring SCO Foxpro 2.6 for Unix and Linux \\*](#)

[Programming Changes \\*](#)

[Running SCO binaries on Linux \\*](#)

[Installing iBCS module \\*](#)

[Fixing iBCS module \\*](#)

[Fixing \(iBCS: SCO tape error\) \\*](#)

[Installing linux-abi \\*](#)

[RedHat \\*](#)

[Installing the linux-abi patch \\*](#)

[Recompiling the kernel \\*](#)

[Fixing \(abi: SCO tape error\) \\*](#)

[Locating Terminfo Database \\*](#)

[Disk Duplication \\*](#)

[Manual Installation \\*](#)

[Foxpro Startup Script \\*](#)

[Installing my FPU runtime \\*](#)

[Installing FPU on FreeBSD \\*](#)

[Installing FPU on RedHat 8.0 \\*](#)

[Installing FPU on slackware 7.x/8.x \\*](#)

[Installing FPU on United Linux \\*](#)

[Installing FPU on wyse60 terminals \\*](#)

[Terminfo Files \\*](#)

[Fonts \\*](#)

[Compiling your own terminfo files \\*](#)

[Terminfo Definitions \\*](#)

[Using Scancode Mode \\*](#)

[Using Keystroke Equivalents \\*](#)

[Keystroke Equivalents \\*](#)

[File Names/Directories \\*](#)

[Mounting Windows drive \\*](#)

[File Permissions \\*](#)

[Installation Example \\*](#)

[Sticky Bit On \\*](#)

[Killing User Process \\*](#)

[Command Line Switches \\*](#)

[CONFIG.FPU \\*](#)

[Error Messages \\*](#)

[Purchasing SCO Foxpro 2.6 for Unix \\*](#)

[Summary \\*](#)

[Index \\*](#)

Disclaimer

All information contained on this sheet is informational only - There are no warranties, expressed or implied. You should always double check any information before acting upon it.

This sheet contains references and/or links to other sites or to third party suppliers of goods and/or services. Such references or links should not be considered as an endorsement by the sponsors of this web site.

## Installing/Configuring SCO Foxpro 2.6 for Unix and Linux

Installing and configuring Foxpro for Unix and Linux systems is a little different than a DOS system. This tip sheet was created to cover the some of the basics.

Before I begin, I'd like to thank everyone on the CompuServe Unix and MS Developer forums, plus all the people on the Internet newsgroup <news://microsoft.public.fox.programmer.exchange>. Special thanks to John Sandell and George Carden for helping me get over the major hurtles. Also thanks to Malcolm Gambles for help on the linux-abi project.

Now using SCO binaries on Linux, such as Foxpro, requires several steps. First, you need to install the necessary modules to handle SCO binaries (see "Using SCO binaries on Linux"). You need to define where your terminfo database is located (see "Locating Terminfo database") and add your Linux compatible terminfo definition (see "Terminfo Files"). Since it is a Linux system, you must have the correct file names (see "File Names/Directories") and correct file/directory permissions (see "File Permissions").

The next step involves installing and configuring your copy of Foxpro (see "Manual Installation"). You need to set your CONFIG.FPU to the Unix/Linux environment (see "CONFIG.FPU"). Finally, you'll need to adjust your Foxpro 2.6 code for the Unix/Linux environment (see "Programming Changes").

## Programming Changes

Compiled Foxpro 2.6 for DOS \*.APP and \*.FXP files will run on Foxpro 2.6 for Unix (development or runtime). You may, however, have to adjust your code to handle Unix and Linux. The following describes some of these changes:

### CHR(7)

On Linux kernel 2.2 systems using iBCS, bell output results in a "SIG: sigpending lied" message. To reduce these messages use:

```
IF .NOT. _UNIX
?? CHR(7)
ENDIF
```

Oh, Foxpro produces it's own bell output. Typically when [Ctrl]-[F] searches result in 'Not found'. While you may not be able to avoid them, use [Ctrl]-"]" to clear (refresh) the screen of these SIG messages.

Note: You can eliminate these SIG messages by fixing the underlining Linux bug. See fixing iBCS module.

### Clear/Refresh Screen

As just stated bell output, file not found, RUN command executing a shell script, print spooler, all can produce various messages. Messages not part of the regular Foxpro screen. Annoying as these messages are, the ONLY way to remove them is to use [Ctrl]-"]" to clear (refresh) the screen. If you use Foxpro for Unix, you will be refreshing the screen a lot.

### DATE()

See TIME().

### DIR

In Unix, you have to be explicit with your DIR command. In other words, the command [DIR /dmail4] will show nothing. You have to use the command [DIR /dmail4/\*.dbf].

### DRIVESPACE()

On my RedHat 6.0 Linux system, DRIVESPACE() gives a huge negative number. Even it's ABS() value is strange. Don't trust it.

### Esc

On Unix, [Esc] is the escape character. When you press [Esc], Unix presumes that something is going to follow. Only after a pause in time will LASTKEY() register 27.

Like [Esc], [Ctrl]-[Q] will also terminate a READ. But [Ctrl]-[Q] returns LASTKEY() = 17. I suggest using ON KEY LABEL CTRL-Q KEYBOARD CHR(27) PLAIN. Simply tell people to use [Ctrl]-[Q] as well as [Esc]. Then you don't have to chase down [LASTKEY() = 27] code and appending [.OR. LASTKEY() = 17].

## FILE()

Remember that Foxpro simply cannot reference any Unix name/directory containing an uppercase character. So even if the file "/home/README" exists, the function FILE("/home/README") will always return false. See File Names/Directories.

## FOPEN()

Found this problem while doing a demonstration. Had a FOPEN(SYS(2019)) with SYS(2019) empty (no CONFIG.FPU file defined). Well, Foxpro returned a FOPEN value of 7. FEOF(7) returned a value of .f. and FGETS(7) just started to read random stuff off the hard drive. Until Foxpro shutdown completely.

To avoid this problem, simply make sure you never call FOPEN with an empty variable.

## GETENV()

Always define your environment variables in uppercase. The Unix shell commands:

```
foxterm = at386a ; export foxterm  
FOXTERM = at386a ; export FOXTERM
```

...define two different GETENV() variables, foxterm and FOXTERM. In other words, Unix is case sensitive.

## GETFILE()

Depending on what directory you start, and what directory you select your file, GETFILE() might return a strange directory. As an example, I got the following from my GETFILE():

```
D = "C:\\MNT\\WIN98\\DMAIL4\\MAL.FXP"
```

I found the easiest way to clean up this path is using:

```
D = FULLPATH(SYS(2027,D))
```

This will convert the file path to Unix, then back to the MS-DOS file naming convention.

## MODIFY COMMAND

Some editing keys might work a little different on a Unix system. Instead of [Shift] arrow keys, use [Ctrl]-[P], arrow keys, then [Ctrl]-[P] again. Instead of [Ctrl]-[Home], use [Ctrl]-[Y] [B]. For window zoom, use [Ctrl]-[Y] [U]. See keystroke equivalents section.

On my RedHat 6.0 Linux system, I have my FAT32 Windows 98 partition mounted as a (vfat) table (see file names/directories). I can actually run a Foxpro application from the vfat table. I have full read/write access. Even tried packing and reindexing. Seemed to work fine.

Well, one problem. If I do a MODIFY COMMAND, for some reason a [Save] will result in 'Cannot create file'. The solution is to [Save As] to a temporary file, then [Save As] to the original file. To avoid the problem, I suggest storing your Foxpro application either on a FAT16 partition or a Linux table.

### Mouse Cursor

The mouse cursor does not work on Linux. I've talked to the author of the Linux ncurses. To fix the mouse cursor, he needs to know how the SCO mouse drivers were designed. If anyone has this information, or knows who does, please contact me at [dennis@dennisallen.com](mailto:dennis@dennisallen.com). I'd greatly appreciate it.

### ON KEY LABEL F10 KEYBOARD CHR(23) PLAIN

There's nothing wrong with using OKL F10. Just remember, [F10] is the Unix system menu wakeup key. A Unix version of the DOS [Alt] key. Don't use it if you need access to the system menu (SET SYSMENU OFF).

### ON KEY LABEL ALT+P DO TEST\_PRINT

In Unix, the [F10] key is used instead of the [Alt] key. Therefore, OKL ALT won't work. You have two choices: You can change your OKL ALT code to OKL SHIFT or OKL CTRL. Another choice is to redirect your keyboard stroke, as in ON KEY LABEL CTRL+S KEYBOARD "{ALT+S}". Just don't use keystrokes like OKL CTRL+Y or OKL CTRL+P. Avoid conflicts with alternate keyboard assignments (see keyboard assignments).

### PACK, DELETE TAG ALL/INDEX ON

Commands like PACK or combinations like DELETE TAG ALL and INDEX ON will create new files, and thus change file permissions. Make sure the user who performs these tasks has the correct file permission rights.

For example, [umask 000] in your Foxpro startup script will allow you to

recreate files without locking out the other users in the same group. See file permissions section.

### PRINTSTATUS()/SYS(13)

As far as I know, every available printer returns a PRINTSTATUS() of true. Whether a printer is actually hooked up or not.

### PRTINFO()

Unlike other OS platforms, where PRTINFO(1) = -1, my RedHat 6.0 Linux returns PRTINFO(1) = 1. Prefix any test of PRTINFO() with \_WINDOWS.

### RENAME

Found this problem while doing a demonstration. I was trying to run a Foxpro 2.6 application stored on my Windows 98 FAT32 drive. Yes, both the application and data were stored on the vfat partition. Well, RENAME would return error 1153 [Attempt to Move file to a different drive].

I didn't see this problem with a 2Gig vfat table. Of course now it's a 4Gig table. I can offer only two solutions. Either change your [RENAME old TO new] to [COPY FILE old TO new / DELETE old], or move your application to a Linux partition.

### SET CLOCK ON

See TIME().

### SET PRINT TO

You can SET PRINT TO either a standard print device (PRN, LPT1, etc.), or a Unix/Linux print device (/dev/lp0).

If you perform a SET PRINT TO PRN, the public variable \_PRN needs to be set to the Unix/Linux print spooler. You can define \_PRN in either code or in your CONFIG.FPU. In Linux, \_PRN needs to equal 'lpr -s' or 'lpr -Pprinter -h'.

Couple of \_PRN notes: Setting the Linux \_PRN to either 'lp -s' or 'lpr -s' could result in 'lpstat command failed'. Since \_PRN works with spooled output, issuing SET PRINT TO closes the spooler. Blank pages might get ejected between spool opening and closings.

Printer note: On my Linux system, printer output produces a SIG message. Use

[Ctrl]-"]" to clear (refresh) the screen of this SIG message.

## SET SCOREBOARD OFF

If you SET SCOREBOARD ON, an input error might prompt with (Press SPACEBAR to continue). On my RedHat 6.0 Linux, I can't see this response message. To avoid the problem, I've added a new feature to my applications. The user can now define an environment variable STATUS=OFF, which does a SET SCOREBOARD OFF.

### Shell script

I have a couple of applications that create and execute batch files. While I'm in the process of converting these routines to handle Unix shell scripts, I have already learned a couple of things:

First, a MODIFY FILE or a FCREATE/FPUTS will create a file with linefeeds and carriage returns. Carriage returns are ok for source code, but not for scripts. Use the following Unix command to remove carriage returns:

```
cat temp.bat | tr -d '\r' > temp2.bat
```

Second, you need to be able to execute this script. Use the Unix command:

```
chmod a+x temp2.bat
```

Which brings one last point. Executing the Unix command [temp2.bat] does NOT execute the script. Unlike MS-DOS, Unix doesn't search the current directory for a specified file. You need to be explicit, as in [/home/temp/temp2.bat], or at least [./temp2.bat].

### SYS(30)

SYS(30) is suppose to return the Unix login name. Doesn't work on my RedHat 6.0 Linux. Don't count on it. If available, use getenv('USER') instead. Some systems might use getenv('LOGNAME').

### SYS(31)

If the process (RUN command) terminates normally, the function will return the exit code (in string format). If the process was killed, the signal number of the signal that killed the process is returned.

### SYS(32)

SYS(32) detects whether scancode is ON or OFF. See scancode.

SYS(2000,'')

There are some SYS() functions that will force a shutdown. Especially true for SYS(2000,''). Use FILE() instead of !SYS(2000,'')== "".

Oh, if you use SYS(2000,'',1), use ADIR() instead. For example, if you have DO WHILE !EMPTY(SYS(2000,'',1)), use =ADIR(FLD,'') / FOR D = 1 TO ALEN(FLD,1).

SYS(2027,'')

Foxpro, in general, wants to work with file paths using MS-DOS conventions. Certain instances, however, require Unix naming conventions. Use SYS(2027,'') to convert a MS-DOS path to a Unix path.

Warning: Foxpro cannot reference any Unix name/directory containing an uppercase character. See File Names/Directories.

TIME()

On a non-SCO Unix machine the DATE() and TIME() maybe four to five hours ahead of the computer's clock. Depends on your machine's Greenwich Meridian Time (GMT). The solution is simple. If your shell is /bin/sh, you need to modify the .profile (or .bash\_profile) in your home directory to include:

```
TZ=CST6CDT ;export TZ
```

...where CST is the timezone, 5 is the GMT offset, and CDT is the daylight savings time zone. On my machine, for example, I had to use TZ=EST5EDT.

If your shell is /bin/csh, you need to modify .cshrc in your home directory, to include:

```
setenv TZ CST6CDT
```

USE

If you get 'File too large' or files cannot be written beyond a certain size, you need to increase Unix's ulimit. System administrators, see ulimit.

Luis Alberto Reyna of Argentina sent me a tip. If you get 'too many files open', edit the file /proc/sys/fs/file-max (might be /proc/sys/kernel/file-max in RH 5.1). Increase the file count. If the count is 4800, increase it to 6000. No

need to reboot, the solution is immediate.

## Running SCO binaries on Linux

Using SCO binaries on Linux, such as Foxpro, depends on the version of the kernel. For kernel versions 2.2 and before, you need to install the iBCS module. For kernel versions 2.4.15 or later, you need to install the linux-abi kernel patch. For kernel versions in between, sorry. You're out of luck.

For people still using the Linux kernel versions 2.2 and prior, the following section will explain how to install the iBCS module. If you are using the Linux kernel version 2.4.15 or later, you can skip directly to the linux-abi section.

Note: To determine your kernel version, simply execute `[uname -a]`.

### Installing iBCS module

If you're running the Linux kernel 2.2, need to install the iBCS component of the kernel. If a `[lsmod]` command fails to show this component, you need to add it. The easiest way is to create a `/etc/rc.d/rc.modules` file (executed upon boot) containing the following two lines:

```
#!/bin/sh
```

```
insmod /lib/modules/2.2.9-19mdk/misc/iBCS.o
```

Note: The location of your `iBCS.o` module shouldn't be too much different. Also remember to set `rc.modules` to the correct file permission `[chmod 755 rc.modules]`.

### Fixing iBCS module

In the section on programming changes, I discussed several Foxpro coding methods to reduce the occurrence of the message "SIG: sigpending lied". This message is from a bug in the iBCS component of the Linux kernel. Well, Piotr Kasprzyk actually found this bug and showed me how to fix it.

First, check for a `/usr/src/linux/ibcs` directory. If this directory doesn't exist, you might want to reinstall your Linux. In the list of installation components, add kernel development.

If you can't reinstall Linux, you can get the source file (229 KB) for `ibcs2`

(latest version 2 of kernel) by downloading:

<ftp://tsx-11.mit.edu/pub/linux/BETA/ibcs2/ibcs-2.1-981105.tar.gz>

You would unpack this compressed tar file in /usr/src/linux:

```
mkdir /usr/src/linux
cd /usr/src/linux
tar zxvf /tmp/ibcs-2.1-981105.tar.gz
```

Once you have the iBCS source code, change to the ibcs/iBCSemul directory:

```
cd /usr/src/linux/ibcs/iBCSemul
```

Next, run "vi" to edit the file signal.c:

```
vi /usr/src/linux/ibcs/iBCSemul/signal.c
```

Type "/" and "function deactivate\_signal()", then [Enter]. You should find "function deactivate\_signal()" around line 95. Enter the [Insert] mode. Before the spin\_unlock\_irq command, add the following line:

```
recalc_sigpending(task);
```

This function should then look like:

```
Void
deactivate_signal(struct task_struct *task, int signum)
{
spin_lock_irq(&task->sigmask_lock);
sigdelset(&task->signal, signum);
recalc_sigpending(task);
spin_unlock_irq(&task->sigmask_lock);
}
```

Press [Esc] to exit the insert mode, then type ":wq" to save your change. Change to the ibcs directory:

```
cd /usr/src/linux/ibcs
```

You'll need to edit the CONFIG file. If one doesn't exist, create one (example for i386 platform):

```
cp CONFIG.i386 CONFIG
```

Run "vi" (or another editor) to edit the file "CONFIG". On the first page, find KERNEL and make sure KERNEL=/usr/src/linux. After much frustration, I found out you can't compile the iBCS module unless KERNEL has been defined.

From the ibcs directory, you can now compile a new iBCS module with the following two commands:

```
make
make install
```

During the compile, you'll see a lot of messages. If either make command bombs right away, something is missing or wasn't set right. A missing /include/linux/modversions.h file, for example, means you didn't set the CONFIG file's KERNEL option.

Note: Depending on your system, the original iBCS CONFIG file may not be configured at all. If that is the case, you may have to copy CONFIG.i386 over CONFIG. You'd then have to tweak the CONFIG file until it suits your needs. If you have only one Pentium, for example, change "SMP=yes" to "SMP=no". Oh, don't forget to set KERNEL=/usr/src/linux.

To check if you did create a new iBCS module, do the following:

```
cd /lib/modules/2.2.9-19mdk/misc
ls -l iBCS*
```

Depending on your system, your misc directory may be different. You should see two files. The file iBCS.o is the original component, the file iBCS is the newest. You can load and test your new module:

```
rmmod iBCS # Remove any loaded module
insmod iBCS # Install a new one
```

Now run FoxPro and test it. If you no longer see any "SIG: sigpending lied" message, change the file /etc/rc.d/rc.modules to reference this new iBCS component.

Note: Rebuilding a Linux module is not easy. Sometimes only the source code that comes with your Linux will work. If you're having trouble installing the iBCS module, check the Informix iBCS Howto web site, [http://www.iiug.org/resources/linux/Howto\\_IBCS.html](http://www.iiug.org/resources/linux/Howto_IBCS.html). Another great source of help is the Internet newsgroup <news://comp.os.linux.setup>.

Fixing (iBCS: SCO tape error)

When launching Foxpro under iBCS, you might see the message "iBCS: SCO tape ioctl func=0 arg=0". This message either gets displayed or logged in the file /var/log/messages. This problem can also occur when your terminal goes unused for some period of time and somebody wants remote access to Foxpro.

Luis Alberto Reyna of Argentina ([betoreyna@hotmail.com](mailto:betoreyna@hotmail.com)) ran into this problem using iBCS. His solution is to remove the offending line of code from the associated iBCS module.

First, change to the ibcs/iBCSemul directory:

```
cd /usr/src/linux/ibcs/iBCSemul
```

Next, run "vi" (or another editor) to edit the file ioctl.c:

```
vi ioctl.c
```

In the file ioctl.c, around line 1167, look for the following code:

```
#endif
default:
printk (KERN_ERR "iBCS: SCO tape ioctl func=%d arg=%x
unsupported\n", func & 0xff, (int) arg);
return -EINVAL;
}
```

Simply delete (or comment out) the printk command (the lines between default and return). Type ":wq" to save your change. Change to the ibcs directory:

```
cd /usr/src/linux/ibcs
```

The final steps will be to recompile your iBCS module (see previous section, "Fixing iBCS module").

### Installing linux-abi

If you are running the Linux kernel 2.4.15 or later, your kernel needs the linux-abi patch. iBCS is no longer compatible.

RedHat

RedHat 7.3 thru 8.0 already comes with a working linux-abi patch. Run "vi" to edit the file /etc/rc.d/rc.local:

```
vi /etc/rc.d/rc.local
```

Scroll to the end of the file, press [Ins] and add the following lines:

```
insmod /lib/modules/2.4.18-14/kernel/abi/util/abi-util.o
insmod /lib/modules/2.4.18-14/kernel/arch/i386/kernel/lcall7.o
insmod /lib/modules/2.4.18-14/kernel/abi/svr4/abi-svr4.o
insmod /lib/modules/2.4.18-14/kernel/abi/cxenix/abi-cxenix.o
insmod /lib/modules/2.4.18-14/kernel/abi/ibcs/abi-ibcs.o
insmod /lib/modules/2.4.18-14/kernel/abi/sco/abi-sco.o
insmod /lib/modules/2.4.18-14/kernel/fs/binfmt_coff.o
insmod /lib/modules/2.4.18-14/kernel/fs/binfmt_xout.o
```

(where 2.4.18-14 is the installed kernel version). Press [Esc] to exit the insert mode, then type ":wq" to save your changes.

Note: If you ever recompile your kernel, you'll need to edit these insmod commands to point to your new linux-abi modules. The command [ls /lib/modules/2.4.18-14/kernel/abi/util/abi-util.\* -l], for example, would show both the old and new versions of the module [abi-util].

Note: RedHat 8.0 has new system fonts, incompatible with Foxpro. For details, see the section 'Installing FPU on RedHat 8.0'.

### Installing the linux-abi patch

On most distributed versions of Linux, the kernel has been modified and will not accept the linux-abi patch. Therefore, to install the linux-abi patch you also need to install a clean kernel.

First, you need to download the latest linux-abi patch. Then you need to download the corresponding Linux kernel. For the linux-abi patch, search the <http://www.kernel.org> web site. I found the latest linux-abi patches at <ftp://ftp.kernel.org/pub/linux/kernel/people/hch/linux-abi/v2.4/>.

For the latest Linux kernel, search the <http://www.kernel.org> web site. Due to the long download time, I prefer using <ftp://ftp.kernel.org/pub/linux/kernel/v2.4/>. And if the main web site is down, check <http://www.kernel.org> for mirror sites.

At the time of this writing, the latest working linux-abi patch was

linux-abi-2.4.17.0.patch.gz (linux-abi-2.4.18.0.patch.gz introduced a compiler bug). For this patch, you need to download the 2.4.17 kernel.

## Recompiling the kernel

Now I'm not going to go into all the possible options of recompiling a Linux kernel. There are plenty of Linux how-to books available. I'm just going to cover the basics. To install and patch your kernel, execute the following commands:

```
cp /usr/src/linux/.config /root
cd /usr/src
rm -rf linux
tar zxvf /root/linux-2.4.17.tar.gz
mv linux linux-2.4.17
ln -s /usr/src/linux-2.4.17 /usr/src/linux
```

This set of commands saves a copy of your current configuration file (.config), and installs your new kernel 2.4.17 off in a separate directory. To install the linux-abi patch, execute the following commands:

```
cd /usr/src/linux
gzip -cd /root/linux-abi-2.4.17.0.patch.gz | patch -p1
```

If you already have the diff patch file, then execute the following commands:

```
cd /usr/src/linux
patch -p1 < /root/linux-abi-2.4.17.0.diff
```

Note: This latest patch cannot work from /usr/src, using patch -p0. It has to run from /usr/src/linux, using patch -p1.

Now you need to configure your system. Execute the following commands:

```
cd /usr/src/linux
make mrproper
make distclean
cp /root/.config .
make oldconfig
```

A typical "clean" kernel contain many more drivers then a distribution kernel. The "make oldconfig" command will show what drivers got added. When you get to the "binary support for other systems" section, press "y" on every option except "verbose debugging". Write down what else got added. Then execute the

following command:

```
make menuconfig
```

It's not enough to say "n" to a new feature, sometimes you must remove the calling section. For example, my first time recompile bombed on the new network driver. Even after I said "n" to the new driver from [make oldconfig]. So in [make menuconfig] I had to remove the whole calling section on that network driver.

To simplify things, my system would never use video tape drivers, ham-radio, or USB, so I used [make menuconfig] to remove their support altogether. Once you saved your .config file, execute the following commands:

```
make dep
make clean
make bzlilo
make modules
make modules_install
```

If you boot from a floppy, use bzdisk instead of bzlilo. If you get a compile error message, write it down. Then perform another [make menuconfig], remove the call to the offending driver, and execute these commands again. You might have to do this several times before you get a clean compile.

If you want to keep the original kernel, use bzImage instead of either bzlilo or bzdisk. There will be a few more steps involved in creating a new kernel. You will want to create a new boot directory and copy your new kernel to it. Then setup your Linux to boot from that directory.

If you are not familiar with compiling kernels, there are plenty of Linux how-to books available. You might want to check the file called <http://www.kernel.org/pub/linux/kernel/README>

If you want more information on linux-abi, visit the linux-abi project web site <https://sourceforge.net/projects/linux-abi/> or the Malcolm Gambles web site, <http://linux-abi.sourceforge.net/>. There is also an excellent linux-abi tutorial by Brian K. White on the AP Lawrence web site found at <http://aplawrence.com/Linux/linuxabi.html> or <http://pcunix.com/Linux/linuxabi.html>.

Fixing (abi: SCO tape error)

When launching Foxpro under linux-abi, you might see the message "abi: SCO

tape ioctl func=0 arg=0". This message either gets displayed or logged in the file /var/log/messages. This problem can also occur when your terminal goes unused for some period of time and somebody wants remote access to Foxpro.

Jason ([jasonp\\_hypertech@skynet.net](mailto:jasonp_hypertech@skynet.net)) ran into this problem using linux-abi. His solution is to remove the offending line of code from a couple of sco modules.

First, change to the abi/sco directory:

```
cd /usr/src/linux/abi/sco
```

Next, run "vi" (or another editor) to edit the file ioctl.c:

```
vi ioctl.c
```

Type "/" and "printk", then [Enter]. You should see the following printk command:

```
printk(KERN_DEBUG "sco: ioctl[%d, %lx[%s], 0x%p) unsupported\n", fd,  
ioctl_num, class_str, arg);
```

Simply delete (or comment out) the printk command (both lines that make up the command). Type ":wq" to save your change. Next, run "vi" (or another editor) to edit the file tapeio.c:

```
vi tapeio.c
```

Again, type "/" and "printk", then [Enter]. When you find the same printk command, delete (or comment out) the printk command and ":wq" to save the file. The final step is to recompile your kernel (see previous section, "Recompiling the kernel").

Edwin Beecher ([edwin@arnet.com.ar](mailto:edwin@arnet.com.ar)), from Argentina, offers another solution:

I've read the man pages of insmod, rmmod and lsmod and find something very interesting in the man pages of syslog about the printk function.

So this is a "cheat" to fix the abi SCO tape error without recompiling your kernel (in Red Hat 8).

What do you need?

1. The file abi-sco.o, located in /lib/modules/2.4.18-14/kernel/abi/sco/

2. A hexadecimal editor, NOT A PLAIN TEXT EDITOR. If you have one in Linux, excellent, if you don't (like me) you can copy this file to a Windows environment, make the modifications with you favorite hexadecimal editor and copy it back to Linux.

3. Of course, you must be root to do this. Important: make a copy of the original abi-sco.o file in case of trouble.

Here we go. Open the abi-sco.o file with your hexadecimal editor and find the following string:

abi: SCO tape ioctl

You'll see something like this:

```
002200 .....<7>sco: ioctl(%d, %lx[%s], 0x%p)
002240 unsupported.....<3>iBCS: SCO termios ioctl %d un
002280 supported.....<3>abi: SCO tape ioctl func=%d a
0022C0 rg=%x unsupported.....<3>vtkbd.c: vtkd ioctl 0x%02x un
002300
supported..6.....L.....N.....Y.....[.....].....c...
```

Now, change the number "3" before the "abi" word to a number "6", so it must see like this:

```
002200 .....<7>sco: ioctl(%d, %lx[%s], 0x%p)
002240 unsupported.....<3>iBCS: SCO termios ioctl %d un
002280 supported.....<6>abi: SCO tape ioctl func=%d a
0022C0 rg=%x unsupported.....<3>vtkbd.c: vtkd ioctl 0x%02x un
002300
supported..6.....L.....N.....Y.....[.....].....c...
```

Save the file with another name and put it back in Linux in the /lib/modules/2.4.18-14/kernel/abi/sco/ directory (you can use another location if you want).

If you use the same name, rename the original file BEFORE.

Let suppose that the new file is called abi-new.o

Now we have to remove from memory the abi-sco module. Type the following:

```
rmmod abi-sco
```

If you get an error, someone is working with fox, so you can use the lsmod command:

```
lsmod | more
```

If nobody is working, you will see a "0"(zero) in the "Used by" column of the abi-sco module, otherwise you will know how many people is in fox, so "kill" them for a while :)

If you have removed the abi-sco module from memory, now you can load the new module:

```
insmod abi-new.o (or the name you've used)
```

That's all, now go to any terminal and execute fox in the command line.

You won't see the abi: SCO tape error in the console.

To finish you work you must edit the file rc.local that is located in /etc/rc.d and change the "abi-sco" line with the name of the new file.

```
cd /etc/rc.d
```

```
vi rc.local
```

Comment the old line and add a new line, like this:

```
....
```

```
....
```

```
insmod /lib/modules/2.4.18-14/kernel/abi/ibcs/abi-ibcs.o
```

```
# insmod /lib/modules/2.4.18-14/kernel/abi/sco/abi-sco.o
```

```
insmod /lib/modules/2.4.18-14/kernel/abi/sco/abi-new.o
```

```
insmod /lib/modules/2.4.18-14/kernel/fs/binfmt_coff.o
```

....

....

Save you rc.local file.

Why we changed a number "3" with a number "6"? See the man pages of the syslog command ( man syslog ) and see the DESCRIPTION SECTION. You'll find the answer.

### Locating Terminfo Database

Before you can run Foxpro on a non-SCO version of Unix, you must tell Foxpro where to find the terminfo database. On a Linux system, you must create a symlink. Execute the following command:

```
ln -s /usr/share/terminfo /usr/lib/terminfo
```

When Foxpro searches for the directory /usr/lib/terminfo, it will find the directory /usr/share/terminfo. You only need to execute this command once.

Note: Although your terminfo directory is located, your terminfo definition still needs to be defined and your terminfo file compiled. See the following sections on 'Manual Installation' and 'Terminfo files'.

### Disk Duplication

As I already learned, the first time you install Foxpro for Unix, your disks will get "branded". That is, permanently stamped with your name and organization. Tough, especially if you misspelled. To avoid branding, I suggest working with backup disks. There are a couple of methods to create backup disks. One way is to use dd:

```
dd if=/dev/fd0 of=/home/tmp1 bs=36b
```

... where /dev/fd0 is your floppy drive. Here we created a file called /home/tmp1 containing the image of disk one. To create a new disk one, simply reverse the destinations:

```
dd if=/home/tmp1 of=/dev/fd0 bs=36b
```

### Manual Installation

You can manually install the standard Microsoft Foxpro 2.6 for Unix (or associated runtime) diskettes on a Non-SCO Unix or Linux system. First, check for an uncompress command. If executing [uncompress] results in "command not found", you need to add it. The easiest way is to create a /usr/bin/uncompress file containing the following line:

```
gunzip $1
```

Note: Remember to set /usr/bin/uncompress to the correct file permission [chmod 755 /usr/bin/uncompress].

Assuming you already have either iBCS or linux-abi in your kernel and linked /usr/share/terminfo to /usr/lib/terminfo (see previous sections), execute the following commands:

```
mkdir /usr/lib/foxpro
cd /usr/lib/foxpro
tar xvf /dev/fd0 ./tmp/readme
tar xvf /dev/fd0 ./tmp/extract
./tmp/extract /dev/fd0
```

... where /dev/fd0 is your floppy drive. For the Foxpro runtime, extract the file rtheadme, not readme.

You may get a prompt to add entries to the terminfo database (see terminfo section). If these SCO terminfo definitions fail to get added, you can manually add them with the command:

```
tic /usr/lib/foxpro/terminfo.src
```

### Foxpro Startup Script

You may or may not get the startup script /usr/bin/foxpro. You should get the startup script /usr/bin/fox. The Foxpro runtime would give you the startup script /usr/bin/foxr. This startup script will need to be adjusted. Use the following script as a template:

```
PROGDIR=/usr/lib/foxpro
#
# This is the script which invokes the binary program 'foxpro.pr'.
#
umask 000

PATH=$PROGDIR:$PATH
```

```
export PATH
```

```
TZ=EST5EDT  
export TZ
```

```
FOXTERM=at386a  
export FOXTERM
```

```
exec $PROGDIR/foxpro.pr "$@"
```

Check to make sure PROGDIR on line 1 has no prefix characters. The manual installation might prefix PROGDIR with garbage characters.

For the Foxpro runtime startup script /usr/bin/foxr, use foxr.pr instead of foxpro.pr.

To the startup script, I added the umask command. The umask command specifies the default protection for newly created files. In our case, umask=000 will give new files full read, write, and execution permission (see file permission section).

To the original script, I added the TZ variable. It sets the Foxpro date/time clock to Eastern Standard Time (your time zone may differ).

To the original script, I also added the FOXTERM variable (if not found, Foxpro uses the variable TERM). FOXTERM is set to at386a, a Linux compatible terminfo definition. You can find terminfo source files on my web site, <http://www.dennisallen.com/fox.htm#foxfpu2> (the terminfo section of the Foxpro information sheet). To use at386a, download a copy of at386a.src and execute:

```
tic at386a.src
```

This tic command will compile the definition at386a, adding it to your terminfo database. Now you should be ready to try out Foxpro.

### Installing my FPU runtime

As I said in the beginning, compiled Foxpro 2.6 for DOS \*.APP and \*.FXP files will run on a Foxpro 2.6 for Unix runtime. A copy of my FPU runtime is available on the freeware section of my web site, <http://www.dennisallen.com/freeware.htm>. To manually extract this FPU runtime, called foxrun.tar, execute the command:

```
tar -xvpP -f foxrun.tar
```

This tar file will extract foxr.exe, foxr.pr and associated files into the directory /usr/lib/foxpro. The startup script file foxr should be extracted into the directory /usr/bin.

Terminfo note: The startup script foxr sets the environment variable FOXTERM to at386a, a Linux compatible terminfo definition. To use at386a, you need to execute the following command:

```
tic /usr/lib/foxpro/at386a.src
```

This tic command will compile the definition at386a, adding it to your terminfo database. Assuming you already have either iBCS or linux-abi in your kernel (see previous sections) and have linked /usr/share/terminfo to /usr/lib/terminfo (see "Locating Terminfo Database"), you should be ready. To test the FPU runtime, simply execute foxr.

I've only had one reported problem with my FPU runtime. Someone couldn't send standard output via =fopen('/dev/stdout',12).

Now my Foxpro runtimes are really stand-alone applications. Compiled one-line programs whose only purpose is to launch other Foxpro programs. In the case of the FPU runtime, foxr.exe is only needed for runtime distribution. The solution to the standard output problem is to bypass foxr.exe and call foxr.pr directly. In the /usr/bin/foxr script, simply change foxr.exe to foxr.pr.

### Installing FPU on FreeBSD

Peter Elsner ([peter@servplex.com](mailto:peter@servplex.com)) successfully installed Foxpro 2.6 for Unix on FreeBSD. In fact, a copy of his Foxpro for FreeBSD 4.0 runtime is now available on his web site, <http://www.servplex.com>.

Peter sent me a FreeBSD compatible terminfo file, available on my web site, <http://www.dennisallen.com/fox.htm#foxfpu2> (the terminfo section of the Foxpro information sheet). He also sent me the following installation script, illustrating the steps necessary to install FPU on FreeBSD:

```
#!/bin/sh
# install.sh script for installing FoxPro 2.6a for SCO Unix on FreeBSD 4.0.
# Author: Peter Elsner <peter@servplex.com>
# Date Created: 7/11/2000
# Date Last Modified: 7/12/2000
#
# The author would like to thank the following people for their contribution
# in making SCO's FoxPro 2.6a for Unix functional on FreeBSD
```

#### 4.0-RELEASE.

```
#
# Bill W Smith Jr <bill@servplex.com>
# For his work in getting the fansi FOXTERM to work
# and getting colors.
#
# Dennis Allen <dennis@dennisallen.com>
# For his info on FoxPro running on Linux. It was very helpful.
#
# Chorny S.I. <serg@relay.dtcom.dp.ua>
# For his copy of the terminfo.db files (in particular fansi).
# NOTE: From the fansi binary file, I was able to convert it
# back to fansi.src using the tconv command with the -b option.
#
# Hampton Finger <locnar@locnar.net>
# For his knowledge of FreeBSD.
#
# Place fansi.src, fpu26_FreeBSD.tar.gz, terminfo.tar.gz and this
# install.sh file in your /tmp directory. Then run ./install.sh AS ROOT!!!
#
# THE SCRIPT STARTS HERE
#
echo 'Creating FoxPro directory under /usr/lib'
if [ ! -f /usr/lib/foxpro/foxpro.pr ]
then
mkdir /usr/lib/foxpro
echo 'Done'
else
echo '/usr/lib/foxpro already exists...'
exit 1
fi
echo 'Copying FoxPro tar file to /usr/lib/foxpro'
cp /tmp/fpu26_FreeBSD.tar.gz /usr/lib/foxpro
echo 'Done'
echo 'Untarring FoxPro tar file'
cd /usr/lib/foxpro
tar xzf fpu26_FreeBSD.tar.gz
echo 'Done'
echo 'Creating dependencies'
# Check to see if these directories exist. Create if they don't.
if [ ! -d /usr/compat/ibcs2 ]
then
mkdir /usr/compat/ibcs2
fi
if [ ! -d /usr/compat/ibcs2/dev ]
```

```
then
mkdir /usr/compat/ibcs2/dev
fi
# Move the null (empty file) from /usr/lib/foxpro to /usr/compat/ibcs2/dev/
cd /usr/lib/foxpro
mv null /usr/compat/ibcs2/dev
# Assign links...
cd /usr/compat/ibcs2/dev
ln -fs /dev/null XOR
ln -fs socksys nfsd
ln -fs /dev/null socksys
echo 'Done'
# Check for the existence of a terminfo directory in /usr/lib.
if [ ! -d /usr/lib/terminfo ]
then
mkdir /usr/lib/terminfo
cp /tmp/terminfo.tar.gz /usr/lib/terminfo
cd /usr/lib/terminfo
tar xzf terminfo.tar.gz
fi
# Check for the existence of /usr/lib/terminfo/f directory.
if [ ! -d /usr/lib/terminfo/f ]
then
mkdir /usr/lib/terminfo/f
fi
echo 'Compiling fansi.src into terminfo.db'
mv /tmp/fansi.src /usr/lib/terminfo/f
if [ ! -f /usr/bin/tic ]
then
mv /usr/lib/foxpro/tic /usr/bin
fi
cd /usr/lib/terminfo/f
tic fansi.src
# Now check to see if ibcs2 modules are already loaded in the kernel.
cd /usr/compat/ibcs2/dev
kldstat > kldstat.txt
grep 'ibcs2.ko' kldstat.txt > ibcs2.txt
if [ -s "ibcs2.txt" ]
then
echo 'ibcs2 already installed!'
else
kldload ibcs2.ko
kldload ibcs2_coff.ko
echo 'You must edit /etc/rc.conf and add IBCS2_ENABLE="YES"'
echo 'Then reboot your FreeBSD server.'
```

```

fi
rm kldstat.txt
rm ibcs2.txt
echo 'Done'
if [ ! -d /usr/local/bin ]
then
mkdir /usr/local/bin
fi
mv /usr/lib/foxpro/stfox /usr/local/bin
cp /usr/lib/foxpro/foxpro /usr/local/bin
cd /usr/lib/foxpro
echo 'FoxPro 2.6a has been successfully installed on your FreeBSD system'
echo 'To run FoxPro, type stfox at the prompt'
exit 0

```

### Installing FPU on RedHat 8.0

As I understand it, RedHat 8.0 has new system fonts. These new fonts are not compatible with older applications, like [make menuconfig] and Foxpro. You need to edit /etc/sysconfig/i18n. First save off a copy, then edit i18n:

```

cp /etc/sysconfig/i18n /etc/sysconfig/orig-i18n
vi /etc/sysconfig/i18n

```

Delete all text, then enter [insert] mode and add the following lines:

```

LANG="en_US"
LC_ALL="en_US"
LINGUAS="en_US"

```

Press [Esc] to exit the insert mode, then type ":wq" to save your changes. Changes won't take effect until you reboot.

### Installing FPU on slackware 7.x/8.x

Alejandro Fernández ( [foxlinux@alejandrofernandez.com.ar](mailto:foxlinux@alejandrofernandez.com.ar)) successfully installed Foxpro 2.6 for Unix on slackware 7.x/8.x. He also sent me a slackware 7.x/8.x compatible terminfo file, available on my web site <http://www.dennisallen.com/fox.htm#foxfpu2> (the terminfo section of the Foxpro information sheet).

To make his system compatible with both Foxpro and sco foxbase+, he changed the character set from 850 to 437. To do that, he added the following to his startup script:

```
# ....  
  
# ....  
  
# 437  
  
if [ $TERM = linux ]  
  
then  
  
echo -e -n '\033(K'  
  
fi  
  
exec $PROGDIR/foxpro.pr +pc -i -t "$@"
```

### Installing FPU on United Linux

Frank Quirk ([quirkf@swbell.net](mailto:quirkf@swbell.net)) successfully installed Foxpro 2.6 for Unix on United Linux. According to Frank:

'I brought up my AccountiX on United Linux. This is SCO Linux 4.0. I can bring the package up on F1 - F6 using the at386b emulation. The package can be brought up the X windows without a FOXTerm. I would guess that the X windows are running in ansi the same as the console on SCO Unix. You can bring up multiple windows on the windows. I would guess that you could have multiple sessions with multiple windows open. In United Linux the file to put the insmods is boot.local rather than rc.local. This also runs without the SCO Libraries so there is no problem with the SCO claims. Oh, you do not have to have the en\_US entries. And I will look into the mouse cursor as this should work.'

### Installing FPU on wyse60 terminals

John Contento ([contentj@ix.netcom.com](mailto:contentj@ix.netcom.com)) successfully installed Foxpro 2.6 for Unix on wyse60 terminals. He also sent me the following notes:

You need to set the tabs on otherwise the input will look a little weird, the cursor will jump around some. You need to set the init tabs=on under the setup screen in f2 in order for it to save the tab information (otherwise when you power off the terminal it resets the tabs to blank and you have to reset them again).

Also here's a wiring diagram to get dtr (hardware flow control) to make linux reset the terminal to a login prompt. Resets the terminal without having to

reboot the server. This works on a rocketport from control but should work with any db-25 serial connection. I know this wiring isn't standard but I've got it running on around 30 terminals total with no problems. Hope this is helpful to someone. John

wyse60 host

2 3 \

3 2 / send & receive

7 7 signal ground

20 5 hardware flow control

4 8 resets terminal to login w/power cycle

### Terminfo Files

From the manual installation section, the startup script initialized the environment variable FOXTERM to at386a. at386a is a Linux compatible terminfo definition. This definition is stored in the terminfo source file at386a.src. You can find terminfo source files on my web site, <http://www.dennisallen.com/fox.htm#foxfpu2> (the terminfo section of the Foxpro information sheet). To use at386a, download a copy of at386a.src and execute:

```
tic at386a.src
```

This tic command compiled the at386a terminfo definition, adding it to your terminfo database.

For other variants of Unix, you may need to try other terminfo files. The terminfo database itself contains a vast set of definitions from which to choose. SCO Foxpro for Unix also comes with the following set of SCO terminfo definitions, in a source file called terminfo.src:

wy60fox, wy60-25fox, wy60fox-pc, wy50fox, wy325fox, wyse325fox

wy325wfox, wy325-25fox, ansifax, ansicfox

To add these definitions to your terminfo database, execute the command:

```
tic /usr/lib/foxpro/terminfo.src
```

To try out a new terminfo definition, simply change FOXTERM. For example, to use the definition called ansifox, execute:

```
FOXTERM=ansifox ; export FOXTERM
```

Oh, if you are using C shell, you'll need to use the setenv command instead of the export command. Otherwise, you're ready to run Foxpro and test your new terminfo definition.

Note: If you plan to change FOXTERM on the fly, remember to remove any FOXTERM line from your Foxpro startup script. Oh, I did hear of a guy who couldn't get FOXTERM to change. Using the Foxpro command ?GETENV('FOXTERM'), he finally figured out he had several layers of FOXTERM commands buried in his scripts.

## Fonts

If your terminfo definition is just slightly off, it may be the system fonts. As I understand it, RedHat 8.0 has new system fonts. These new fonts are not compatible with older applications, like [make menuconfig] and Foxpro. You need to edit /etc/sysconfig/i18n:

```
cp /etc/sysconfig/i18n /etc/sysconfig/orig-i18n
vi /etc/sysconfig/i18n
```

Delete all text, then enter [insert] mode and add the following lines:

```
LANG="en_US"
LC_ALL="en_US"
LINGUAS="en_US"
```

Press [Esc] to exit the insert mode, then type ":wq" to save your changes. Changes won't take effect until you reboot. Who knows? You might find that editing /etc/sysconfig/i18n is easier than changing terminfo definitions.

## Compiling your own terminfo files

Once you find a terminfo definition that works reasonably well on your system, you will probably want to adjust it. After all, no terminfo definition is perfect.

Suppose, for example, you wanted to try adjust the terminfo definition xterm for your X Windows system. Make a copy of the source code by executing the command:

```
infocmp -l /usr/lib/terminfo/x/xterm > xtermfox.src
```

Edit this terminfo source file. The first thing you need to change is line 2. Change 'xterm' to 'xtermfox'. That way, you can compile the definition xtermfox, not overwrite the original definition xterm. Next, change the line count from 24 to 25. Foxpro needs 25 lines. Now compile xtermfox with the command:

```
tic xtermfox.src
```

Once you set FOXTERM to xtermfox, you will be ready to test the new terminfo definition.

### Terminfo Definitions

For any given terminal, Foxpro will look for input escape sequences. The following lists the basic key definitions:

- kf0 Function key 10
- kf1 - kf10 Function Keys 1..10
- kf13 - kf22 Shift Function Keys 1..10
- kf25 - kf34 Ctrl Function Keys 1..10
- kf37 - kf46 Shift-Ctrl Function Keys 1..10
- kcub1 Left Arrow
- kcud1 Down Arrow
- kcuf1 Right Arrow
- kcuu1 Up Arrow
- khome Home
- kend End
- knp Page Down
- kpp Page Up
- kich1 Insert
- kdch1 Delete
- kbs Backspace
- kcbt BackTab
- kcan Escape
- khlp Help Key (F1)
- kcpy Copy Key for Edit-Copy accelerator, (^C)
- knxt Key Next: move right one word, Ctrl-rightarrow
- kprv Key Previous: move left one word, Ctrl-leftarrow
- kfnd Key Find: Edit-Find accelerator (^F)
- krpl Key Replace: Replace and find again accelerator
- krdo Key Redo: Edit-Redo accelerator (^R)
- kund Key Undo: Edit-Undo accelerator (^Z)
- krfr Key Refresh: Re-paint the screen, (^])

kcmd Key Command: Special Key lead-in, (^Y)  
kCMD Key Shift Command: Shift Toggle, (^P)  
kLFT Key Shift Left Arrow: Selects one char to left  
kRIT Key Shift Right Arrow: Selects one char to right  
kHOM Key Shift Home: Selects to beginning of line  
kEND Key Shift End: Selects to end of line  
kNXT Key Shift Next: (Shift-Ctrl-rightarrow)  
kPRV Key Shift Previous: (Shift-Ctrl-leftarrow)

One way to find out what keystroke your terminal produces is by executing the following:

```
stty -echo ; cat -v ; stty echo
```

This command echos most keystrokes back to the console. Use [Ctrl]-[D] to exit. If you run across another way to determine keystrokes, please let me know.

My advice is to use the old error and trial method. Say, for example, that the backspace key doesn't work in xtermfox. But backspace does work in at386a. Well, a look in at386a.src shows kbs=\177. Simply find kbs in xtermfox.src, change it to \177, then compile with the tic command.

Terminfo files wanted: If you do fix an existing terminfo definition, or can create a new working terminfo (for whatever terminal or Unix variant), send me a copy. I'd be more than happy to post it on my web site. Make it available to everyone.

### Using Scancode Mode

You can put some terminals, for example, the Wyse 60, into PC scancode mode, which means that a numeric code is sent to the computer each time a key is pressed down, and each time a key is let up. If your UNIX system is running on SCO UNIX System V 3.4 or higher, scancode mode is supported.

When a terminal is in scancode mode, Foxpro is able to detect any sequence it could detect coming from the PC console. It can tell the difference between all Ctrl key combinations and special keys. With the terminal in PC scancode mode, a terminal user can use the same keystroke or combination of keystrokes as a console user.

If the terminfo entry for your terminal has the sequences for entering and exiting scancode mode (smc and rmc), Foxpro will put it into scancode mode automatically when you start the program. For information on terminal types that support scancode mode, consult your terminal documentation.

## Using Keystroke Equivalents

If your serial terminal cannot be set to go into scancode mode, you can overcome its keyboard restrictions with the system of keystroke equivalents in Foxpro. For each unreliable keystroke, Foxpro provides a key sequence that will work on any serial terminal. The system works, in general, as follows:

To simulate pressing the Shift key by itself (as used in text selection, for example), type Ctrl+P. To Foxpro, this action is equivalent to holding down the Shift key on the console.

To simulate releasing the Shift key, either type another Ctrl+P or type any other sequence to complete the action on the selected text (such as the sequences for Cut or Copy).

To simulate a lead-in to all other unreliable key sequences, type Ctrl+Y. In general, you will type Ctrl+Y, release both keys, and type another single character. For example, Ctrl+Y followed by the number 3 simulates F3.

On Unix, [Esc] is the escape character. When you press [Esc], Unix presumes that something is going to follow. If nothing follows, after a time, then [Esc] will register with Foxpro.

## Keystroke Equivalents

All keystrokes used commonly in Foxpro are summarized in the table that follows. There are three columns: Action, a description of the action that Foxpro takes; Console, the key sequence that produces that action on the console (and on serial terminals that are capable of emitting the sequence); and Terminal, the alternate key sequence that can be used on any ASCII terminal.

### Browse Window

#### Action Console Terminal

```
=====
Change active partition Ctrl+H Ctrl+Y, H
Append a blank record Ctrl+N Ctrl+N
Toggle record delete Ctrl+T Ctrl+T
```

### Dialogs

#### Action Console Terminal

```
=====
Select default text button Ctrl+Enter Ctrl+W or Ctrl+J
```

## Expression Builder

### Action Console Terminal

```
=====
```

Choose database popup Ctrl+B Ctrl+B  
 Display date functions Ctrl+D Ctrl+D  
 Verify expression Ctrl+E Ctrl+E  
 Choose fields list Ctrl+F Ctrl+F  
 Display logical functions Ctrl+L Ctrl+L  
 Display math functions Ctrl+M Ctrl+Y, M  
 Choose variables list Ctrl+R Ctrl+R  
 Display string functions Ctrl+S Ctrl+S

## Filer

### Action Console Terminal

```
=====
```

Tag all files or dirs Ctrl+A Ctrl+A  
 Copy tagged files or dirs Ctrl+C Ctrl+C  
 Delete tagged files or dirs Ctrl+D Ctrl+D  
 Edit tagged files Ctrl+E Ctrl+E  
 Find text in tagged files Ctrl+F Ctrl+F  
 Change current directory Ctrl+H Ctrl+H  
 Make a subdirectory Ctrl+K Ctrl+K  
 Display file tree panel Ctrl+L Ctrl+L  
 Untag all files or dirs Ctrl+N Ctrl+N  
 Move tagged files or dirs Ctrl+O Ctrl+O  
 Rename tagged files or dirs Ctrl+R Ctrl+R  
 Change tagged files' attrs Ctrl+T Ctrl+T  
 Invert file or dir's tags Ctrl+V Ctrl+V  
 Tag multiple files Shift+Spacebar Ctrl+P, Spacebar

## Interface

### Action Console Terminal

```
=====
```

ESC (abort current action) Ctrl+Q Ctrl+Q  
 Exit and save Ctrl+W Ctrl+W  
 Exit without saving Esc Esc  
 HELP F1 Ctrl+Y, 1  
 SET F2 Ctrl+Y, 2  
 LIST F3 Ctrl+Y, 3  
 DIR F4 Ctrl+Y, 4  
 DISP STRU F5 Ctrl+Y, 5  
 DISP STAT F6 Ctrl+Y, 6

DIP MEM F7 Ctrl+Y, 7  
 DISP F8 Ctrl+Y, 8  
 APPEND F9 Ctrl+Y, 9  
 ALT; activate/de-active sys F10 Ctrl+Y, 0  
 Activate/de-activate sys Alt Ctrl+Y, 0

## Keyboard Macros

### Action Console Terminal

=====  
 Clear all current macros Ctrl+A Ctrl+A  
 Clear selected macros Ctrl+C Ctrl+C  
 Make macro set the default Ctrl+D Ctrl+D  
 Edit existing macro Ctrl+E Ctrl+E  
 Record a macro Ctrl+M Ctrl+Y, M  
 Create new macro Ctrl+N Ctrl+N  
 Restore a set of macros Ctrl+R Ctrl+R  
 Save current macros Ctrl+S Ctrl+S  
 Shortcut to create macros Shift+F10 Ctrl+P, Ctrl+Y, 0

## Label Designer

### Action Console Terminal

=====  
 Create an expression Ctrl+E Ctrl+E  
 Preview the labels Ctrl+I Ctrl+Y, I  
 Specify label size Ctrl+L Ctrl+L  
 Save label environment Ctrl+N Ctrl+N  
 Specify style for label Ctrl+Y Ctrl+Y

## Menu Builder

### Action Console Terminal

=====  
 Delete an item Ctrl+E Ctrl+E  
 Insert a menu item Ctrl+I Ctrl+Y, I

## Program Execution

### Action Console Terminal

=====  
 Execute a program Ctrl+D Ctrl+D  
 Resume program execution Ctrl+M Ctrl+Y, M

## Project Manager

## Action Console Terminal

```
=====
```

Add file to project Ctrl+A Ctrl+A  
 Build the project Ctrl+B Ctrl+B  
 Exclude a project file Ctrl+C Ctrl+C  
 Edit a project Ctrl+E Ctrl+E  
 Display info for project Ctrl+I Ctrl+Y, I  
 Display project info Ctrl+J Ctrl+J  
 Display Build Option dialog Ctrl+O Ctrl+O  
 Show project errors Ctrl+S Ctrl+S  
 Remove file from project Ctrl+V Ctrl+V

## Record Search

### Action Console Terminal

```
=====
```

Continue finding a record Ctrl+K Ctrl+K

## Report Writer

### Action Console Terminal

```
=====
```

Create a box Ctrl+B Ctrl+B  
 Add a field Ctrl+F Ctrl+F  
 Bring object to front Ctrl+G Ctrl+G  
 Preview report Ctrl+I Ctrl+Y, I  
 Move object to back Ctrl+J Ctrl+J  
 Add a line Ctrl+N Ctrl+N  
 Remove a line Ctrl+O Ctrl+O  
 Add text Ctrl+T Ctrl+T  
 Resize object Ctrl+Spacebar Ctrl+Y, Spacebar  
 Select multiple objects Shift+Spacebar Ctrl+P, Spacebar

## RQBE

### Action Console Terminal

```
=====
```

Execute the query Ctrl+Q Ctrl+Q  
 Show SELECT command Ctrl+S Ctrl+S

## Screen Builder

### Action Console Terminal

```
=====
```

Create a box Ctrl+B Ctrl+B

Add a field Ctrl+F Ctrl+F  
 Bring object to front Ctrl+G Ctrl+G  
 Create push buttons Ctrl+H Ctrl+H  
 Create invisible buttons Ctrl+I Ctrl+Y, I  
 Move object to back Ctrl+J Ctrl+J  
 Create a check box Ctrl+K Ctrl+K  
 Create a list Ctrl+L Ctrl+L  
 Create radio buttons Ctrl+N Ctrl+N  
 Create a popup Ctrl+O Ctrl+O  
 Open all snippets Ctrl+S Ctrl+S  
 Add text Ctrl+T Ctrl+T  
 Resize object Ctrl+Spacebar Ctrl+Y, Spacebar

## Text Editing

### Action Console Terminal

=====  
 Select all text Ctrl+A Ctrl+A  
 Copy selected text Ctrl+C Ctrl+C  
 Replace and find again Ctrl+E Ctrl+E  
 Display Find dialog Ctrl+F Ctrl+F  
 Find again Ctrl+G Ctrl+G  
 Redo text editing Ctrl+R Ctrl+R  
 Undo text editing Ctrl+U Ctrl+U  
 Paste copied text Ctrl+V Ctrl+V  
 Cut selected text Ctrl+X Ctrl+X  
 Move right Right arrow Right arrow  
 Move left Left arrow Left arrow  
 Move up one line Up arrow Up arrow  
 Move down one line Down arrow Down arrow  
 Move up one window Pg up Ctrl+Y, R  
 Move down one window Pg Dn Ctrl+Y, C  
 Move to start line Home Ctrl+Y, S  
 Move to end line End Ctrl+Y, D  
 Move one word right Ctrl+Right arrow Ctrl+Y, F  
 Move one word left Ctrl+Left arrow Ctrl+Y, A  
 Move to start file Ctrl+Home Ctrl+Y, B  
 Move to end of file Ctrl+End Ctrl+Y, E  
 Selects one character left Shift+Left arrow Ctrl+P, Lft arrw  
 Selects one character right Shift+Right arrow Ctrl+P, Rgt arrw  
 Selects one line up Shift+Up arrow Ctrl+P, Up arrow  
 Selects one line down Shift+Down arrow Ctrl+P, Dn arrow  
 Selects to start of word Shift+Ctrl+lft arw Ctrl+P, Ctrl+Y,F  
 Selects to end of word Shift+Ctrl+rgt arw Ctrl+P, Ctrl+Y,A  
 Selects to start of file Shift+Ctrl+Home Ctrl+P, Ctrl+Y,E

Selects to end of file Shift+Ctrl+End Ctrl+P, Ctrl+Y,B

## Trace Window

### Action Console Terminal

```
=====
```

Clear program breakpoints Ctrl+B Ctrl+B  
 Open a program Ctrl+E Ctrl+E  
 Toggle line numbers on/off Ctrl+L Ctrl+L  
 Specify program speed Ctrl+R Ctrl+R

## Window Manipulation

### Action Console Terminal

```
=====
```

Cycle between open windows Ctrl+F1 Ctrl+Y, O  
 Bring Command window up Ctrl+F2 Ctrl+Y, N  
 Move current window Ctrl+F7 Ctrl+Y, V  
 Size current window Ctrl+F8 Ctrl+Y, W  
 Zoom window to minimum Ctrl+F9 Ctrl+Y, L  
 Zoom window to maximum Ctrl+F10 Ctrl+Y, U  
 Hide all windows Ctrl+Alt+Shift Ctrl+Y, X

## File Names/Directories

A while back, I installed RedHat-Mandrake 7.0 Linux. This installation automatically mounted my Windows 98 C: drive as a vfat table. When I tried to test a Foxpro application on this drive, however, I would get an invalid path error. I figured out why.

My Windows 98 C: drive was mounted as /mnt/DOS\_hda1. In Foxpro, this directory is seen as C:\MNT\DOS\_HDA1. When converted to a Unix path, however, it becomes /mnt/dos\_hda1. Of course /mnt/dos\_hda1 doesn't exist. My application would try to use this non-existent path and produce an error. The solution was to unmount /mnt/DOS\_hda1 and to remount as /mnt/win\_c (see next section).

From this example we can conclude that Foxpro simply cannot reference any Unix directory containing an uppercase character. In fact, Foxpro cannot reference any Unix file with a name containing an uppercase character. Files and directories from a Windows 98 mounted drive are not a problem, however, since uppercase characters are automatically converted to lowercase.

If you are unzipping a pkzip file, use [unzip -L filename] to unzip filenames

into lowercase.

If you have a directory with uppercase or mixed upper/lowercased character file names, use the following command to convert all file names to lowercase:

```
for f in * ; do mv $f $( echo $f | tr A-Z a-z ) ; done
```

### Mounting Windows drive

As I said, my RH-MDK 7.0 didn't mount my Windows drive correctly. RH 8.0 didn't mount my Windows 98 C: drive at all. If you don't have a fsconf utility, then you'll have to edit your /etc/fstab file. Perform a [vi /etc/fstab]. Enter the [Insert] mode and add the line:

```
/dev/hda1 /mnt/win_c vfat rw,suid,umask=000,noexec 0 0
```

Press [Esc] to exit the insert mode, then type ":wq" to save your change. Remember to [mkdir /mnt/win\_c] before you reboot.

### File Permissions

In UNIX, all files and directories have a set of permissions that determine who can read, edit, and (in the case of program files) execute them. Read, write and execute privileges are extended to three types of users: the owner of the file, the owner's group, and all other users on the system.

To install a Foxpro application, a system administrator should create a separate group. All users, needing to run the Foxpro application, would belong to that group. The application's directory, and all database files within it, must be given read/write permission to the group. Foxpro, and associated startup script, must also be given read/execute permission to the group.

Note: If you have trouble modifying a database file for which you have the proper permissions, keep in mind that you need write permission to not only the database file itself, but also its associated files, such as index or memo files.

### Installation Example

Say I wanted to install the application dMAIL4. As the user root, I'd first create a group called mail. I'd define all the users, wishing to run dmail4, to belong to group mail.

I would make sure that Foxpro itself, and it's startup script, had read/execute permission by the group mail. I'd also check to make sure the Foxpro startup script had the command `umask=000`. After all, any file created by one user has to be available to all users in the group (see Foxpro startup script section).

I'd then use the following steps to install the dmail4 application:

```
mkdir /home/dmail4
chgrp mail /home/dmail4
chmod u+rw,g+srw,o+rw /home/dmail4
cp /mnt/win98/dmail4/*.prg /home/dmail4
cp /mnt/win98/dmail4/*.fxp /home/dmail4
cp /mnt/win98/dmail4/scr.* /home/dmail4
cp /mnt/win98/dmail4/*.man /home/dmail4
cd /home/dmail4
chmod g+r-wx *
```

Let me explain these steps. First, I created the directory `/home/dmail4`. Then I made the directory available to the group mail.

The `chmod` command gives everyone in the group mail the ability to read/write in our directory. In case you were wondering, the `g+s` in the `chmod` command does have a purpose. Whenever anyone creates a file in `/home/dmail4`, that file's group will be defined by our directory itself, regardless of the user's primary group.

The next set of commands simply copies our application files into our directory. The final `chmod` command will mark these program files as read-only. To prevent a user from accidentally changing or recompiling code.

### Sticky Bit On

Before proceeding, I'd like to talk about that last `chmod` command. While it does protect your program files somewhat from tampering, it cannot prevent users from deleting them. Which may or may not be a problem. After all, it's just as easy to delete a database file as it is to delete a program file.

If you're really serious about protecting your program files, however, you could use the following command:

```
chmod +t /home/dmail4
```

This command turns on the sticky bit to our directory. For any file in our directory, only the owner has permission to delete it.

The drawback to this command is that ALL files in our directory would work with way, database files included. So if user xyz was the first to run dmail4, creating the initial database files, then only user xyz could delete them. Not good. I believe the only way to use the stick bit is to store the database files in a separate directory. In my applications, it would be a simple matter to set the environment variable DATA=/home/dmail4/data.

### Killing User Process

In my applications, certain functions (reindexing, packing, initializing, etc.) must be done in single-user mode. To switch to single-user mode, everyone has to exit the application.

If you are working with programs and files on the Linux box, the system administrator can find out who hasn't exited the program with the command:

```
ps aux | grep fox
```

If the system administrator has to, he can kill the user process or processes. With everyone out of the system, he can then reindex, pack, or whatever.

A note from John G. Sandell (jsandell@att.net): The only problem I have encountered when the system administrator kills a user Foxpro process is a left-over file lock, and only then if the user was in one of the set exclusive on modules. This happens once a year or less. The file then can't be re-opened. The solution is to rename both the \*.dbf and the \*.cdx files to a new name using the UNIX mv command, then do a UNIX cp back to the original names. Takes a minute. Easier than trying to find the lock file.

### Command Line Switches

A command line switch consists of a hyphen, followed by an upper- or lower-case letter, optionally followed by any additional information that may be needed (such as file or path name). No embedded spaces are allowed. These switches are passed directly to the Foxpro shell script that starts Foxpro and override any environment variables that may configure the same Foxpro operation (for example, you can override the FOXPROCFG environment variable with -c).

The table that follows lists the startup switches and their effects.

### Startup Switches

## Switch Effect

-----

- c/<pathname>/<file> Specify configuration file
- e Disable the mouse
- i Treat input device as serial terminal
- o Treat output device as serial terminal.
- t Suppress Foxpro sign-on screen

### Specify Configuration File

You can specify the name of the configuration file to be used with the -c startup switch. If you do not include the <file>, Foxpro looks for an existing CONFIG.FPU.

This is equivalent to setting the environment variable FOXPROCFG. However, if the environment variable exists, this switch overrides the variable's effect.

### Disable the Mouse

The -e switch prevents the use of a mouse in Foxpro. Use this switch if you want to use the console as a text-based serial terminal without a mouse.

Note: The Foxpro readme file mentions that -e also disables SCO event queue management. To be used in non SCO systems.

### Treat Input Device as Serial Terminal

When you use -i, Foxpro treats the input device as a serial terminal. This disables the PC keyboard scan codes, even if they are supported by the system console or terminal. This switch can be useful in debugging and allows Foxpro to run on operating systems that cannot process keyboard scan codes.

### Treat Output Device as Serial Terminal

When the -o command line switch is included, Foxpro treats the output device as a serial terminal and uses terminfo for screen information. When -o is included, Foxpro does not write directly to video memory, even on the system console.

### Suppress Sign-On Screen

You can prevent the display of the sign-on screen by including the optional -t switch.

This switch has no equivalent CONFIG.FPU statement.

### CONFIG.FPU

Here are a couple of CONFIG.FPU commands you may need to add to your Unix/Linux system:

```
_PRN = "lpr"
```

SET PRINT TO PRN uses the spooler defined in public variable \_PRN. The default is "lp -s". Linux needs \_PRN to equal "lpr -s" or "lpr -Pprinter -h". That goes for \_LPT1 and \_LPT2 as well.

To see if you have lp or lpr, either use the command "lpstat -t" or the command "lpc status" (not sure which).

```
VOLUME C = /home/temp
```

If you created /home/temp/dmail4, this volume command will result in FULLPATH('') = "C:\DMAIL4".

Warning: If you do use VOLUME C, remember that ALL Foxpro files must reside under drive C. For example, if you define RESOURCE = /foxrun/foxuser.dbf, Foxpro will think foxuser.dbf resides in c:\foxrun. Which translates into /home/temp/foxrun.

### Error Messages

Installing a SCO product, like Foxpro, on a non-SCO operating system is not always easy. The following is a list of error messages you might encounter, with hints on how to deal with them.

#### Error: Character Fonts

As I understand it, RedHat 8.0 has new system fonts. These new fonts are not compatible with older applications, like [make menuconfig] and Foxpro. You need to edit /etc/sysconfig/i18n. First save off a copy, then edit i18n:

```
cp /etc/sysconfig/i18n /etc/sysconfig/orig-i18n
vi /etc/sysconfig/i18n
```

Delete all text, then enter [insert] mode and add the following lines:

```
LANG="en_US"  
LC_ALL="en_US"  
LINGUAS="en_US"
```

Press [Esc] to exit the insert mode, then type ":wq" to save your changes. Changes won't take effect until you reboot.

Error: Foxpro.pr: cannot execute binary file

: Exec format error

To run SCO binaries on Linux, you need to install additional modules. If you have a kernel versions 2.2 and before, you need to install the iBCS module. For kernel versions 2.4.15 or later, you need to install the linux-abi kernel patch. For kernel versions in between, sorry. You're out of luck. To determine your kernel version, execute [uname -a] (see "Using SCO binaries on Linux").

Error: Foxpro.pr: cannot execute: Permission denied

Your file, foxpro.pr, does not have file execute permission. Try a `chmod a+rx foxpro.pr` (see "File Permissions").

Error: libc: setlocal: LC\_TYPE: unable to open /etc/default/lang

```
libc: setlocal: LC_TYPE: unable to open /etc/default/lang  
libc: setlocal: LC_NUMERIC: unable to open /etc/default/lang  
libc: setlocal: LC_TIME: unable to open /etc/default/lang  
libc: setlocal: LC_COLLATE: unable to open /etc/default/lang  
libc: setlocal: LC_MESSAGES: unable to open /etc/default/lang  
libc: setlocal: LC_MONETARY: unable to open /etc/default/lang
```

You see these messages the first time you launch Foxpro. Annoying, but not fatal. You can, however, reduce the number of messages. Use "vi" to create /etc/default/lang. Add the following lines:

```
LANG="en_US"  
LC_ALL="en_US"  
LINGUAS="en_US"
```

Press [Esc] to exit the insert mode, then type ":wq" to save your change.

Nacho Lamas [nacho@pilicarrera.com](mailto:nacho@pilicarrera.com) was able to completely remove these messages, at least under RedHat 9.0, by adding the line

LANG=en\_US.8859 (or LANG=en\_US.850) to /etc/default/lang (please

note the lack of brackets). The proposed 'english\_us.ascii', as well as C,C\_C,etc., made my copy of FoxPro complain.

Error: linux-abi-2.4.18

The linux-abi patch in linux-abi-2.4.18.0.patch.gz seems to have a svr4 compiler bug. Looking into it. Strangely enough, the linux-abi 2.4.18 patch found in RedHat 8.0 seems fine.

Error: Not enough memory to allocate name table

You have a pre-2.4.15 kernel linux-abi patch (see "Installing linux-abi").

Error: SCO tape ioctl func=0 arg=0

When launching Foxpro with either iBCS or linux-abi, you might see the message "SCO tape ioctl func=0 arg=0". Prefixed by either "iBCS:" or "abi:". This message either gets displayed or logged in the file /var/log/messages. This problem can also occur when your terminal goes unused for some period of time and somebody wants remote access to Foxpro.

The solution is to remove the offending message and recompile the associated module. For iBCS users, see "Fixing (iBCS: SCO tape error)". For linux-abi users, see "Fixing (abi: SCO tape error)".

Error: Segmentation Fault

I've been told that a segmentation fault is like a kill process that hasn't completed yet. Something like a Windows General Protection Fault.

If you're using iBCS, the problem is probably a module that didn't get recompiled correctly. I would first try using your original iBCS module (see installing iBCS module). Next, I'd try to upgrade to the latest iBCS2 code (see "Fixing iBCS module"). Finally, I would upgrade to a 2.4 kernel and use linux-abi (see "Installing linux-abi").

If you're using the linux-abi patch, the kernel and linux-abi patch must be the same version. I had one user install Foxpro on RedHat 8.0, which already had linux-abi. He added insmod calls to his bootup script. For some reason he then decided to recompile the kernel. When he recompiled, he made new linux-abi

modules. The insmod calls in his bootup script, however, were still pointing at the original modules.

Note: If you recompile your kernel, edit any insmod commands to point to the new linux-abi modules. The command `[ls /lib/modules/2.4.18-14/kernel/abi/util/abi-util.* -l]`, for example, shows both the old and new versions of the module [abi-util].

Error: System lockup, Caldera Linux

One person reported this problem. He solved it by editing `/etc/initab`, changing `ID:5` to `ID:3` (boot to the command prompt).

Error: terminfo file not found

First, you need to define where your terminfo database is located (see "Locating Terminfo database"). Next, you need to compile your Linux compatible terminfo source file (see "Terminfo Files"). Finally, your startup script needs to define `FOXTERM` for an existing terminfo definition (see "Manual installation").

Error: Wyse 60 terminal, extra tabs.

John Contento had a problem with his Wyse 60 terminal. At the command window, every 5th character typed would have an extra space in it. During program execution, the cursor wouldn't line up correctly from field to field.

John solved the problem. The `wy60fox` terminfo file that comes with Foxpro worked just fine. He needed a change in the Wyse 60 setup screens. After you enter the terminal setup you need to hit `f6` (it's labeled tabs) and then hit backspace to default the tabs.

### Purchasing SCO Foxpro 2.6 for Unix

People have already started asking how to purchase a copy of Foxpro for Unix. I know you can't get any floppy disks from Microsoft anymore. You might try [ebay.com](http://ebay.com). Copies float on and off the site all the time. Also try one of the following:

EMS Professional Software Retro Tools:  
<http://www.emsps.com/oldtools/msfoxv.htm>

CC Systems, a Florida consulting firm: (727) 596-1460

If all else fails, you can ask for a copy of FPU on the Internet newsgroup

<news://microsoft.public.fox.programmer.exchange>. This is a public newsgroup, so simply create an Outlook Express news account and define the news server as [msnews.microsoft.com](mailto:msnews.microsoft.com).

In the event you can't find a development copy, don't despair. A Foxpro 2.6 for DOS/Windows compiled FXP or APP file will work just fine in Foxpro 2.6 for DOS, Foxpro 2.6 for Windows, Foxpro 2.6 for Macintosh, or Foxpro 2.6 for Unix. All you need is a runtime, which you can find on my web site.

### Summary

I hope you have found this tip sheet useful. If you have a tip to add, drop me an email ([dennis@dennisallen.com](mailto:dennis@dennisallen.com)).

For additional information, try the Internet newsgroup: <news://microsoft.public.fox.programmer.exchange>. This is a public newsgroup, so simply create an Outlook Express news account and define the news server as [msnews.microsoft.com](mailto:msnews.microsoft.com).

You might want to check out the Microsoft Foxpro web site:

<http://msdn.microsoft.com/vfoxpro>

The Microsoft web site has a nice Foxpro 2.6 for Unix FAQ sheet:

<http://support.microsoft.com/support/vfoxpro/content/faq/fpunix/all.asp>

If you need to investigate FPU on Linux, talk to Frank Quirk ([quirkf@swbell.net](mailto:quirkf@swbell.net)). Frank converted SBT Vision Point accounting software, called AccountiX, to run under Linux. Better yet, visit his web site [http://www.accountixinc.com/foxpro\\_inst.htm](http://www.accountixinc.com/foxpro_inst.htm).

If you need to investigate FPU on FreeBSD 4.0, talk to Peter Elsner. A copy of his Foxpro for FreeBSD 4.0 runtime is now available on his web site <http://www.servplex.com>

If you need help running other SCO binaries on Linux, check out the linux-abi project web site <https://sourceforge.net/projects/linux-abi/>, the Malcolm Gambles web site <http://linux-abi.sourceforge.net/>, the AP Lawrence web site <http://aplawrence.com/Linux/linuxabi.html>, or Brian K. White's web site <http://pcunix.com/Linux/linuxabi.html>.

If you still need more help, check out my web site <http://www.dennisallen.com/> Favorite (hyper) Links section.

# Index