Linux を使用している人のための情報

日本語訳　岡本　一幸 Kazuyuki Okamoto <ikko-@pacific.rim.or.jp>

目 次

1 XFree86 がテストされた Linux のバージョン

XFree86 は Linux のバージョン 1.1.49 でテスト済みです。 1.0 以降のバージョンでは、XFree86 に変更がなければ動くでしょう。バイナリとライブラリは 4.5.26 の DLL C ライブラリと 1.4.4 のダイナミック ・リンカー `ld.so` を基にしています (libc-4.5.26 かそれ以降の新しいものが*必須*です)。DLL X ライブラリは Eric Youngdale 氏によって 2.16 DLL-tools を使用して作成されました。

2 前のバージョンとの互換性

X11R6 は X11R5 から大きく更新したものとなっていますので、XFree86-3.1 以降の共有ライブラリは、XFree86-2.1.1 以前の古いライブラリとは互換性がありません。X11R5 のアプリケーションを実行するには、マシンの何処かに古いライブラリを保存しておかなければいけません。古いライブラリは `/usr/X386/lib` から何処かへ移動すればいいのですが、`/etc/ld.so.conf` は更新しなければいけません。全ての X11R5 のアプリケーションは問題無く X11R6 のサーバーと一緒に動作するでしょう。

XFree86-1.2、XFree86-1.3 または XFree86-2.0 ライブラリと関連のある古いバイナリは引き続き動作しますが、`/lib/libX{11,t,aw}.so.3` `/usr/X386/lib/libX{11,t,aw}.so.3` へ明確にシンボリックリンクを張る必要があります。XView3L5 配付物に対して `ld.so` をリンクする時は、XView ライブラリに ld.so が動作しないように共有イメージへの絶対パスを含ませないと、1.3 以前のバージョンの `ld.so` とは問題を生じるでしょう。暫定的対策は、/ (ルートディレクトリ) を `/etc/ld.so.config` に入れることですが、より良くするなら、XView を相対ライブラリ名とリンクするか、より新しい `ld.so` を手に入れることです。

3 XFree86 を導入するには

バージョン 3.0 から始めるなら、XFree86 を `/usr/X11R6` に導入します。X11R5 と X11R6 の両方を導入する十分なディスク容量が無い場合は、X11R5 の必要の無い部分を捨てる事でディスク容量を確保しましょう。特に、フォントと古い X11R5 のバイナリは、消去出来ます。ダイナミックリンクを使う X11R5 のバイナリを動作させるには共有ライブラリを取っておいて下さい。

XFree86 3.X のそれぞれのバイナリ配付物には、対応する版の詳細な導入に関する文書が付属しています。殆どの最新の情報がその文書にありますが、この文書に Linux で動く XFree86 3.X の基本的な機能とソースをどうやってコンパイルするかを記述しています。

4 XFree86 を動作させるには

XFree86 を動作させるためには 4MB の仮想メモリが必要ですが、快適に動作させる最小の環境としては、多分 8MB の実記憶があったほうがいいでしょう。386 プロセッサにとって 387 コプロセッサは、有効ですが、物理的なメモリを増設したほうが対話処理の速度は格段に向上します。また、高速なグラフィックカード、バスまたはメモリがサーバーの性能を向上させるでしょう。

tar ファイルを展開したら、`/etc/ld.so.conf` の中か (初期値によって既にそう定義されているはずです)、`LD_LIBRARY_PATH` 環境変数の中に `/usr/X11R6/lib` を定義する必要があります。また、構成定義ファイルである`/usr/X11R6/lib/X11/XF86Config` を `README.Config` を指針とし、`XF86Config.eg` を出発点としてホストの構成を正確に記入しなければなりません。間違った `XF86Config` を使うとハードウェアを痛める場があるので、良く文章を読んで下さい。特にオンラインマニュアルと `/usr/X11R6/lib/X11/doc` にあるその他の `README` ファイルを良く読んで下さい。

XFree86 は、X サーバーとの間で双方向に仮想端末 (VT) 切り替え (サーバーから仮想端末へ、仮想端末からサーバーへ切り替える) をする機能があります。初めて起動するときは、XFree86 が、先

ず仮想端末（使用可能な他のプロセスに開かれていないもの）を自動的に探して、その仮想端末で起動します。その仮想端末が使用不可の場合、エラーメッセージを出して XFree86 は終了するでしょう。“vt<nn>” というオプションを付けることによって、サーバーを指定する仮想端末で実行させることができます。ここで <nn> は使用可能な仮想端末番号（1 から始まります）です。空いている仮想端末がないときは、XFree86 は起動不能です。`include/linux/tty.h` の中の `NR_CONSOLES` の数を増やしてカーネルをコンパイルすれば、使用可能な仮想端末の数を増やすことができます。

一旦、XFree86 が動作してしまえば、コントロールキー <Ctrl> とオルタネイトキー <Alt> とファンクションキー <Fnn> を同時に押すことで、他の仮想端末に切り替えられます。ここの nn は切り替えたい仮想端末番号です。XFree86 サーバーに戻すときは、サーバーが標準で使っている仮想端末番号を表わす正しいキーの組み合わせを押しましょう。正しいキーの組み合わせとは、オルタネイトキー <Alt> とファンクションキー <Fmm> です。ここで、mm はサーバーが動いている仮想端末の番号です。（この番号はサーバーが起動するときに表示されます。）既に起動しているサーバーから起動した仮想端末ではないことに注意して下さい。

注意: `kbd-0.81.tar.gz` 以降のバージョンの中にある ‘`loadkeys`’ コマンドで、テキストモードのキー割当を再定義できます。これを使えば、例えば、コントロールキー <Ctrl> とオルタネイトキー <Alt> とファンクションキー <Fmm> を同時に使用して X サーバー下から移動するのと同じようなキー操作で、仮想端末のテキストモードから、移動する事が出来ます。

サーバーを終了させるとサーバーを起動した仮想端末に戻りますが、もしサーバーを起動した仮想端末が死んでしまったときは、手動で戻さなければなりません。XFree86 からテキストモードに戻る時に、未だ画面の回復に問題があるのは、グラフィックカードとマザーボードの相性が悪いのだと思われます。こんな時は、**svgalib** という配付物に入っている `runx` というスクリプトを実行すれば、解消できるでしょう。

起動時に、XFree86 サーバーはカーネルに対してキー割り付けを獲得することを要求します。この割り付けはカーネルをコンパイルしたとき設定される初期値かまたは、‘`loadkeys`’ ユーティリティによって書き替えた値になります。サーバーでは、カーネルに依存したもの、複数のキー組み合わせや無効なキーは割り付け出来ません。その他のキーは X 共通に変換されます。X サーバーはシフトしてない状態、シフト状態、モードスイッチを押していない状態と、モードスイッチを押している状態の 4 種類しか変換テーブルを持っていないことに注意して下さい。モードスイッチがどんなキーかによります（これは `XF86Config` で設定可能です。初期値はオルタネイトキー <Alt> です。）が、XFree86 はキーマップの中に変換テーブルを読み込みます。この事は、Linux のモードスイッチに左コントロールキーの様な特定のキーを割り当てられない事を意味します。

構成によっては、調子が悪くなって修正しなければならないことがあるでしょう。過去に、Linux の xload コマンドが、動作中のカーネルから、負荷平均を取り出すのに BSD の方法を使った事があります。XFree86-1.3 版では、カーネルの将来の版での互換性を考慮して、動作中のカーネルから負荷平均を取り出す代わりに、`/proc/loadavg` から取り出す方法に変えました。Xman も BSD の nroff ファミリーの代わりに GNU groff ファミリーをサポートするように変更されました。ちょっと編集して再コンパイルすれば従来の BDS の nroff の方をサポートするように戻せます。クライアントに、仮想端末で動作していたのと同等なより良いコンソール切り替えを提供するために、`/dev/console` と `/dev/tty0` の所有権と共にサーバーが変更されましたが、そのために root の setuid 属性を付けなければならなくなりました。同様に、`xterm` と `xconsole` にも root の setuid 属性を付ける必要があります。`xconsole` は root の setuid 属性を付けて動作するようには設計されてないので、これは安全上、大きな問題が生じると思われます。

5 XFree86 をコンパイルするには

XFree86 をコンパイルするためには特別な手順書は必要ありません。このバージョンの XFree86 は gcc-2.5.8 、DLL ライブラリの 4.5.26 と共用ライブラリの 1.4.4 、ダイナミック ・リンカ ld.so とでコンパイルしています。共有ライブラリは DLL tools-2.16 パッケージで作成し、新しい場所 (`/usr/bin`) にあるツールを使うようにサーバーのソースを更新しました。サーバーは 486 プロセッサ用に最適化する為に、`-m486` オプションを付けてコンパイルされていますが、バイナリは 386 プロセッサで動作するでしょう。(-m486 を付けてコンパイルしたものは、`-m386` を付けてコンパイルしたものより少々ファイルサイズが大きくなっていますが、性能の損失はありません。)

全ての機能拡張とクライアントは移植とテストが終わっています。標準では、サーバーは外部機能拡張 (PEX, XIE 等) 無しでコンパイルしていますが、これらの外部機能拡張のライブラリは既に生成済みです。`site.def` を変更すれば、外部機能拡張をコンパイルしてサーバーに組み込めます。若しくは、リンクキットがサーバーを変更できます。

配付物は大変大きいものですが、オンラインマニュアル、PEX や大きなクライアントを消去する等してソースを注意深く整理すれば、XFree86 は 64MB のパーティション 1 つで、コンパイル出来ます。 Makefile を作成するには、単に “`make Makefiles`” を実行してください。ファイルが自動的に作成され、実行は終了します。ディスクを節約するならば、“`make depend`” は実行する必要はないでしょう。100MB の空きディスクがあれば、もっと容易に XFree86 をコンパイルできます。サーバー全てをコンパイルするには、約 10MB の仮想メモリが必要でしょう。

パッチがライブラリに重要な変更を行なう場合は、更新した jump_xxx が必要でしょう。DLL tools パッケージに付属する手順でパッチが生成でき、XFree86 のパッチを当てるのに必要でしょう。

例えば、Xaw3d のような外部 X 共有ライブラリも X ライブラリをコンパイルするように、 X ライブラリを定義する JUMP_xxx で定義してコンパイル出来ます。詳細な手順については、X ライブラリの定義例を、`/usr/X11R6/lib/X11/config/lnxLib.rules` に提供しています。

Orest Zborowski (*orestz@eskimo.com*) 氏が開発した最新の方法で、全ての詳細説明ファイルを書き、初めて XFree86 DLL ライブラリを作成してくれた Dirk Hohndel (*hohndel@aib.com*) 氏に感謝します。

6 バグの告知について

バグ報告は *XFree86@XFree86.org* に送るか *comp.windows.x.i386unix* のニュースグループに投稿して下さい。Linux のサポートや Linux の配付物に関する質問やコメントは *orestz@eskimo.com.* で行なわれています。

```
$XConsortium: Linux.sgml,v 1.2 95/01/23 15:34:44 kaleb Exp $
$XFree86: xc/programs/Xserver/hw/xfree86/doc/sgml/Linux.sgml,v 3.5 1995/01/28 16:01:43 dawes Exp
```

このファイルは xc/programs/Xserver/hw/xfree86/doc/sgml/Linux.sgml,v 3.5 1995/01/28 16:01:43 を、岡本 一幸 (Kazuyuki Okamoto <*ikko-@pacific.rim.or.jp*> (`mailto:ikko-@pacific.rim.or.jp`)) が XFree86 3.1.2 を日本でインストールする人向けに和訳したものです。ここがおかしいとか、ここはこうしたほうがいいといったご意見がありましたら、電子メールでお知らせ下さい。原文の著作権は XFree86 プロジェクト社にあります。この和訳の著作権は XFree86 プロジェクト社と岡本 一幸にありますが、この和訳の不具合は私に、電子メールで送って下さい。

```
$XFree86: xc/programs/Xserver/hw/xfree86/doc/sgml/Japanese/Linux.sgml,v 3.3 1995/08/30 21:42:23 d
```

[EOF]