

オペレーティングシステム演習

第3回(2009.04.30)

ファイル構成/コンパイル/キーボード変更

MINIX実行環境の確認

□ Run.batの修正

- rem run.bat for Minix 2.0.4 on Bochs/W2K
- rem new (Bochs 2.1) 2004-02-03

- cd “U:¥My Documents¥Bochs-2.2.6¥minix204“
 - ↑ 授業中に説明したが、このパスが「アクセス権限」のあるフォルダでないと(ラボの共通マシンのC:以外にしないと)失敗するので注意
- ..¥bochs -q -f minix.bxrc

- 直接記載されている部分が、正しいか確認する。
 - 誤っている場合は、編集する。

□ 環境変数:BXSHAREの設定

- U:¥My Documents¥Bochs-2.2.6
 - U:などの個人領域に設定しないと、警告が表示される。
-

デスクトップにインストールした学生へ

- 情報ラボのコンピュータのデスクトップにインストールした学生
 - デスクトップのフォルダは、「共有領域」にマップされているようです。従って、実習終了時に「削除」しないと「ログアウト」できない。→実習継続できない。
 - My Documentにインストールしなおして下さい。
-

実行テスト

- Run.batをダブルクリックして、MINIXの起動画面が表示されることを確認する。
-

キーマップ

- 日本語のキーと、アメリカのキーは違う。
 - 「ひらがな」がない！（当たり前か？）
 - フランス語のキーボード
 - ドイツ語、ロシア語、スペイン語、……………
 - OSの言語環境で切り替える。（設定可能とする）

 - キーマップ
 - どのキーを、どのコードに割り振るか。
 - 必要とするコードそのものが違う。
 - マッピングとは？
-

システムの標準化(日本語化)

□ 教科書P201参照

- 文字コードのサポート
- ロケールとは？
- 国際化と地域化

□ 様々な言語群

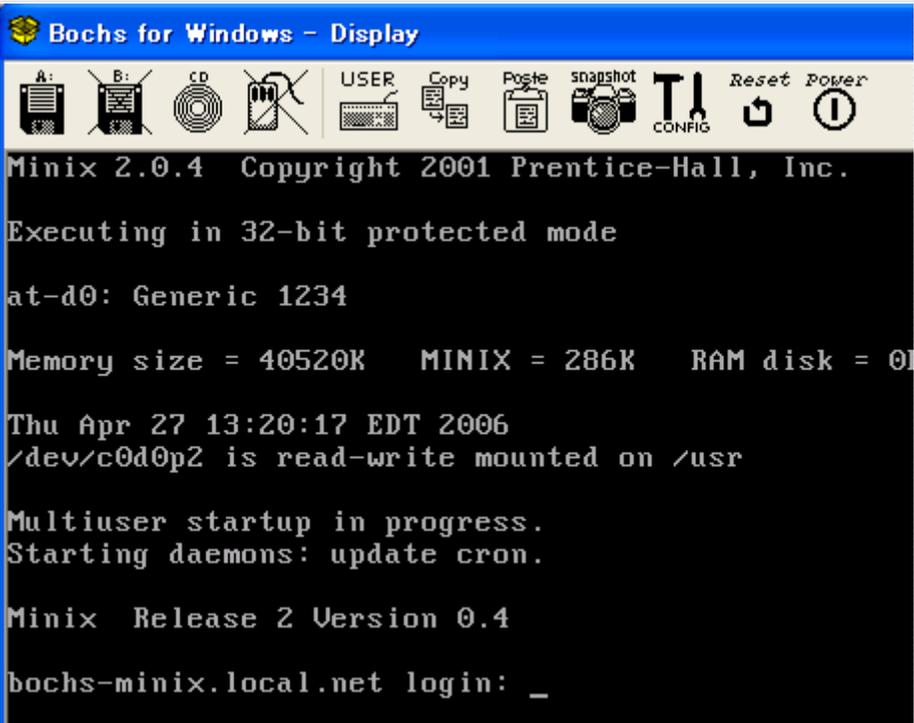
- 例:フランス語([Q]と[A]の位置が逆などかなり違う)
 - なぜ? ∵ 使用頻度が違うため
- <http://citron.maxs.jp/memo/fmemoxp.html>
- ひとつのソースコードで多くの言語に対応
 - MINIXでは、ソースコードレベルで切り替えている。
 - Bochsで対応可能だが、-jpを作っていない。
 - この演習で作って、提供する手もあるが...

MINIXの起動

- Run.batで起動する。
 - 最初の=の入力では、[^]を入力する。
 - (キーの位置が異なる。)
 - Loginプロンプトが出たら、
root
 - と入力する



```
Bochs for Windows - Display
A: B: CD USER Copy Poste snapshot
Minix boot monitor 2.19
Press ESC to enter the monitor
Hit a key as follows:
= Start Minix
n Start Networked Minix
```



```
Bochs for Windows - Display
A: B: CD USER Copy Poste snapshot CONFIG Reset Power
Minix 2.0.4 Copyright 2001 Prentice-Hall, Inc.
Executing in 32-bit protected mode
at-d0: Generic 1234
Memory size = 40520K MINIX = 286K RAM disk = 0
Thu Apr 27 13:20:17 EDT 2006
/dev/c0d0p2 is read-write mounted on /usr
Multiuser startup in progress.
Starting daemons: update cron.
Minix Release 2 Version 0.4
bochs-minix.local.net login: _
```

MINIXの内容確認

- /usrディレクトリの中身を表示させる

cd /usr

cd src

- と入力する。
- 以下のように表示されていたら、OK

- kernel

- MINIXシステムの、「核」の部分のソースコード

- mm

- メモリ管理(memory management)のソースコード

- fs

- ファイルシステム(File System)のソースコード
-

MINIXハードディスク

- 「インストール済のハードディスク」
 - minix.imgを使用する。
 - minix.bxrcの記述の以下の部分を確認する。
 - ata0-master: type=disk, path=minix.img, cylinders=200, heads=16, spt=32
 - s=6, spt=17
 - boot: disk
 - 念のために、各自バックアップを作成しておいて下さい。
 - minix.imgが、演習用の「ハードディスク」
 - コピーを作成しておく、復旧が容易
 - 演習のステップごとに、コピーを作る。
-

MINIXのコマンド(ls, cd, pwd)

- rootでログインする。
 - / (ルートディレクトリ)の下を調べる。
 - ls (lsコマンド)リスト
 - ファイルの一覧を調べる。
 - cd (change directoryコマンド)
 - cd /usr
 - カレントディレクトリを変更する。
 - pwd (print working directoryコマンド)
 - 現在のワーキングディレクトリを表示
-

ファイル構成(1)

- / (ルートディレクトリの下)
 - usr
 - ユーザフォルダ
 - bin
 - システムプログラム
 - dev
 - デバイスファイル: 物理デバイスの抽象化
 - boot
 - MINIXのブートイメージ(システム本体プログラム)
 - etc
 - 様々な環境設定ファイルが記録されている。
-

ファイル構成(2)

- /usr
 - bin
 - ユーザがインストールしたプログラム
 - パスがここを読むように指定される
 - src
 - ソースファイルが格納されている。
 - include
 - C言語で#includeで指定される、ヘッダファイル群
 - システム共通のファイル
 - lib
 - C言語で、システム機能を使用する場合にリンクされるライブラリ
-

スクリプトファイル

- /.ashrc
 - /.profile
 - .で始まるファイル名群
 - /usr/astの下(MINIXの著者のユーザディレクトリ)にもある。(こちらがrootよりも一般的な設定)
 - 通常は表示されない
 - これらのファイルで、ログインした際にユーザごとに、プロンプトの表示を変えたり、パスを変えたり、エイリアス(別名)を登録したりすることができる。
 - If などの記述があるのは、シェルスクリプト
-

ファイルを探す (ls, grep, man)

- ls *.c
 - * -- ワイルドカード(任意の文字、任意の長さの文字列を当てはめることができる。)
 - ? -- 一文字だけ、任意の文字に置き換えて検索する。
 - grep strings filename.c
 - filename.cというファイルの中にある、stringsという文字列を検索し、存在していたら画面に表示する。
 - grep strings kernel/*.c
 - kernelというディレクトリ内の、拡張子がCのファイルで、ファイルの中にstringsという文字列があるものを探す。
 - man grep
 - grep コマンドのマニュアル表示
 - hのキーを押すと、ヘルプ表示
 - LINUXでは、grep --help というオプションもつけるのだが・・・
-

MINIXのコンパイル

- `cd /usr/src/tools`
 - `/usr/src/tools`のディレクトリに移動する。
 - Makeとコマンド入力する。
 - 'image' is up to dateのメッセージを確認する。
 - 「コンパイル済み」のディスクが準備できています。
 - 作成されたimageのファイルのタイムスタンプを確認する。
 - 出来上がったimageファイルを、`/minix`の下にコピーする。
 - `cp /minix/2.0.4 /minix/2.0.4.org`
 - `cp image /minix/2.0.4`
 - 2.0.4以外のファイル名を指定する場合は、boot monitorで、imageファイルを指定することが必要。
-

Shutdownとモニター

- shutdown -h now
 - 電源を切る前に、MINIXをシャットダウンする際の
コマンド
 - ダウンするとモニターに移動する。
 - モニターのコマンド
 - helpと入力する。
 - image = 2.0.4の部分を修正する。
 - Image = bootFileName
 - 起動直後にも、モニターに入れる。
-

キーボードの修正

- 現在の指定は、US-STDの指定になっている。
 - us-stdでセットアップしたMINIXカーネルを使用している。
 - /usr/src/kernel/keyboard.cのファイルを修正する。
 - Us-stdを、japaneseに修正する。
 - 修正したら、コンパイルしたimageを/minixの下にコピーする。
 - Viのコマンドエディタを使用する。
 - コンパイルして、日本語のキー配置で入力できるようにする。
-

実習課題1:コンパイル

- 各自のコンピュータで、minixをコンパイルする。
 - コンパイルしたファイルにkbfidfixと名前をつけて、自分で起動させる。
 - `cp image /minix/kbfidfix`
 - キーボードの配置が修正されていたら、OK
 - TAのチェックを受けること。
 - できないと、次の授業から配置が使いにくい。
 - できた人から帰って構いません。
-

第3週：課題2「コンパイルとmake」

- Makeコマンドの機能を説明しなさい。(5/7まで)
 - Makeコマンド本体の機能について:3点
 - Makefile(スクリプト)の指定方法についての説明:4点
 - その他(Makefileの発展的な使用方法についてなど)もしあれば、適宜加点します。
-