2012年度Newグループ新B4課題

2012/4/2
池田 騰，宮﨑 清人

1　はじめに

本資料では，2012年度のNewグループ新B4課題を示す．

2　課題一覧

新B4は以下の課題をこなすこと．

(課題1) Fedora14のインストール

(課題2) 電卓プログラムと仕様書の作成

(課題3) Linuxカーネルの再構築

(課題4) Linuxカーネルへのシステムコールの実装

(課題5) 仮想アドレスから物理アドレスへの変換

(課題6) 64bit Mintの構築

3　期限

各課題について自身で設定し，Newグループ宛にメールする．

4　課題の前準備

課題に取り組むための前準備として，各自に1台割り当てられる実験用計算機の設置を行う．また，(課題1)で使用するFedora14(64bit)のインストール用DVDを受け取る．これらを用いて課題に取り組む．

5　各課題の詳細

課題1，課題5，および課題6は，動作を先輩に確認してもらうこと．課題2は，仕様書とプログラムの完成後，各自で指導教員にご指導いただくこと．課題3と課題4は，手順書を作成し，New打ち合わせに提出すること．各課題について不明な点は，Newグループの先輩にたずねるか，LastNoteにアップロードされている過去の資料を参考にする．

LastNote
http://lastnote.swlab.cs.okayama-u.ac.jp/document/list

(課題 1) Fedora14 のインストール

Fedora14(64bit) のインストール用 DVD を用いて，各自の実験用計算機に Fedora14(64bit) をインストールする．ただし，インストールの設定時には，OS の構成として「Minimum」を選択すること．また，インストール後にネットワークの設定を行い，ネットワークに接続可能にすること．

(課題 2) 電卓プログラムと仕様書の作成

各自で仕様を決定し，電卓プログラムと仕様書を作成する．この際，使用できるライブラリ関数は printf のみとし，この他にシステムコールは使用してもよい．仕様書の内容については，各自で調査する．仕様書の内容の例を以下に示す．

(1) 概要
(2) 機能
(3) 動作環境 (実装環境とは異なる)
(4) 使用方法
(5) エラー処理と保証しない動作

(課題 3) Linux カーネルの再構築

実験用計算機で Linux カーネル 2.6.39 の再構築を行う．再構築するカーネルのソースコードは，バージョン管理システム Git を用いて以下の Git リポジトリから入手する．なお，Git の詳しい使い方については，4 月上旬に New グループと GN グループが合同で行う Git 勉強会で紹介する．

`TwinOS git` リポジトリ
`git://newgroup.swlab.cs.okayama-u.ac.jp/var/git/TwinOS26.git`

(課題 4) Linux カーネルへのシステムコールの実装

(課題 3) のカーネルに自作のシステムコールを実装する．実装するシステムコールの名前には自身の名前を含めることとし，機能は自由とする．システムコールの実装後，実装したシステムコールを呼び出す応用プログラムを作成し，動作を確認すること．

(課題 5) 仮想アドレスから物理アドレスへの変換

仮想アドレスを物理アドレスに変換する以下のプログラムを (課題 3) のカーネルに実装する．x86_64 CPU のページング機構の仕組みについては，6 章の (2) を参照する．

x86_64 CPU のページング機構では，ページの大きさとして，4KB と 2MB の両方が用いられる．ページの大きさによって，アドレス変換の方法が異なる．そこで，まず 4KB のページのみに対応したプログラムを作成し，その後，プログラムを拡張して 2MB のページにも対応させること．

```
--- システムコール部 ---
unsigned long int user_syscall(unsigned long int *virt)
{
    unsigned long int phys;

    /*****************************************************************
     * 'virt' の示す (仮想) アドレスを物理アドレスに変換する.
     * 変換結果は, 'phys' に格納する.
     *****************************************************************/

    /*****************************************************************
     * 'phys' に格納された物理アドレスをカーネル空間における
     * 仮想アドレスに変換する (注：変換不可能な場合もある).
     * 変換後, 変換した仮想アドレスにアクセスし, int 型の値を得る.
     * 得た値を printk により出力する.
     * (printk は, カーネル内で利用できる printf のようなもの.
     *  ただし, メッセージの出力先が異なる)
     *****************************************************************/

    return phys;
}


--- アプリケーション部 ---
int main(void)
{
    int a;
    unsigned long int phys;

    /*****************************************************************
     * ユーザ AP で定義した変数'a' の (仮想) アドレスをシステムコールに
     * 引数として渡し, 返り値から物理アドレスへの変換結果を受け取る.
     * その後, 結果確認のために, 'a' のアドレスと'phys' の値を 16 進数で
     * 画面に出力する.
     *****************************************************************/

    return 0;
}
```

(課題 6) 64bit Mint の構築

実験用計算機に 64bit Mint を構築し，先行 OS と後続 OS の 2 つの OS を同時に走行させる．構築の完了後，動作確認として各 OS 上で同時にカーネルのコンパイルを行えることを確認する．Mint の構築に必要なものは，TwinOS の git リポジトリと Texec の git リポジトリから入手す

る．また，Mint の構築手順は，New グループ wiki を参照する．

Texec git リポジトリ
git://newgroup.swlab.cs.okayama-u.ac.jp/var/git/Texec.git

New グループ wiki
http://newgroup.swlab.cs.okayama-u.ac.jp/TwinOS/wiki/index.php

6 Tips

課題を解くために必要なことを以下に示す．不明な点は New グループの先輩に尋ねること．

(1) (課題 5) では，CPU のコントロールレジスタを参照するためにアセンブリ言語を用いる必要がある．C 言語のコード中でアセンブリ言語を使用するためには，インラインアセンブリを使用する．インラインアセンブリについては，次の URL を参考のこと．

gcc のインラインアセンブリに関して
http://sci10.org/on_gcc_asm.html

(2) x86_64 CPU におけるページングの仕組みについては,「インテル R エクステンデッド・メモリ 64 テクノロジ・ソフトウェア・デベロッパーズ・ガイド (第 1 巻)」の「1.6. オペレーティング・システムに関する注意事項」の節を参照すること．(マニュアルは次の URL に置いてある)

TwinOS 開発支援ページ
http://newgroup.swlab.cs.okayama-u.ac.jp/TwinOS/index.html

(3) 64bit Linux における仮想メモリ空間の構造については，カーネルに付属する Documentation /x86/x86_64 ディレクトリ以下にある mm.txt を参照する．64bit Linux において，物理メモリの先頭からの領域は，仮想メモリの 0xffff880000000000 番地からの領域にストレートマッピングされている．

(4) アドレス変換における多くのフラグは無視して構わない．(ただし，どんなフラグがあるのかは確認しておくこと)

(5) 仮想アドレス　リニアアドレス