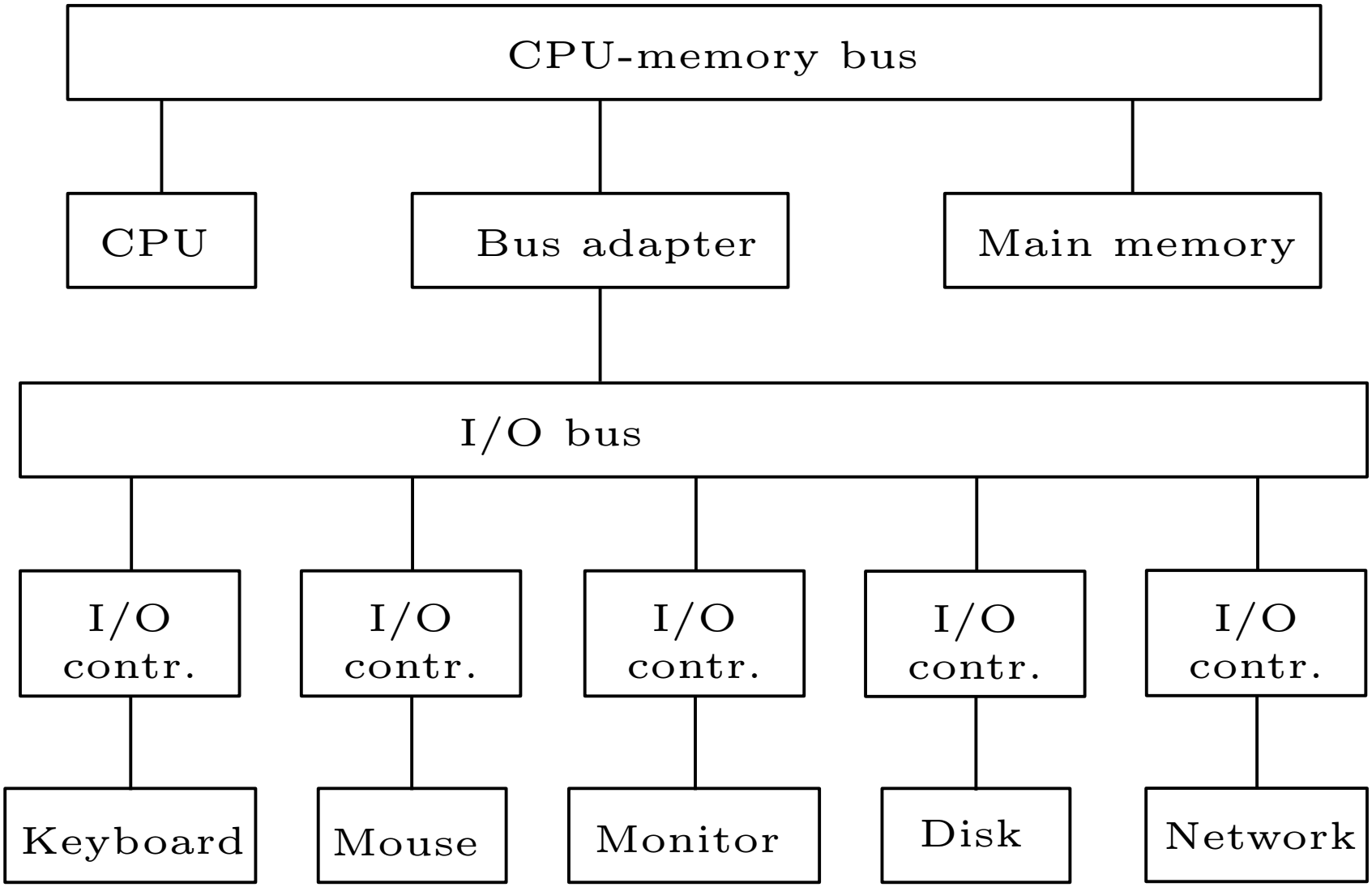
Sistemas Operativos: Introdução

February 17, 2016

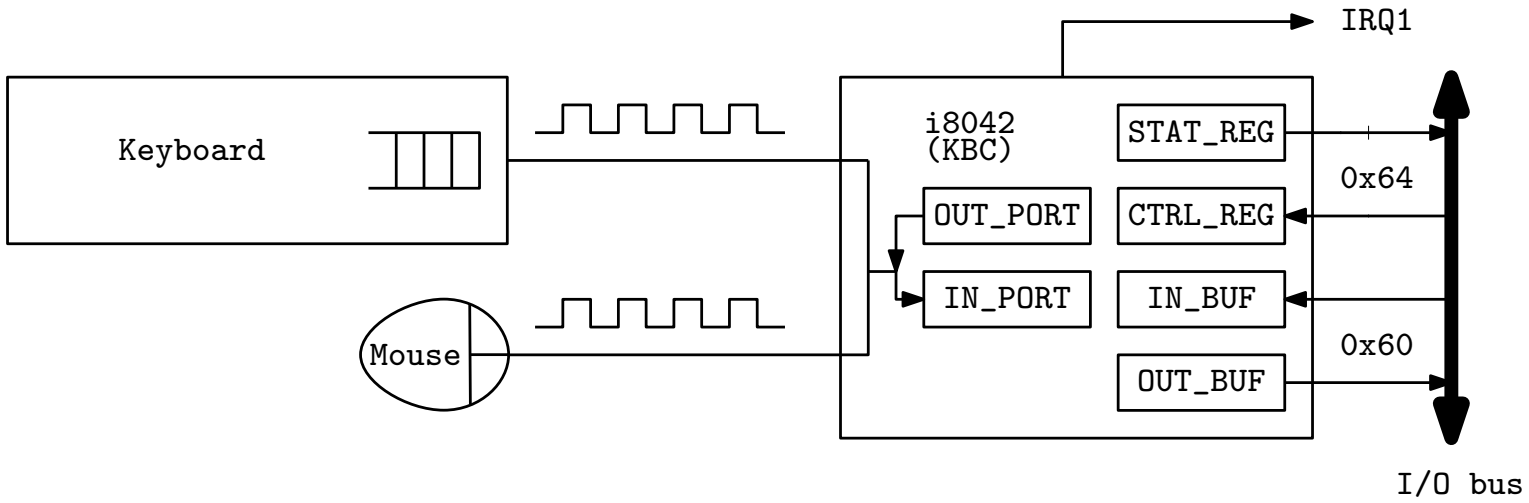
Sumário

Componentes dum Computador Pessoal (gama baixa)



- Desenvolver código directamente sobre o HW dum computador é uma tarefa hercúlea

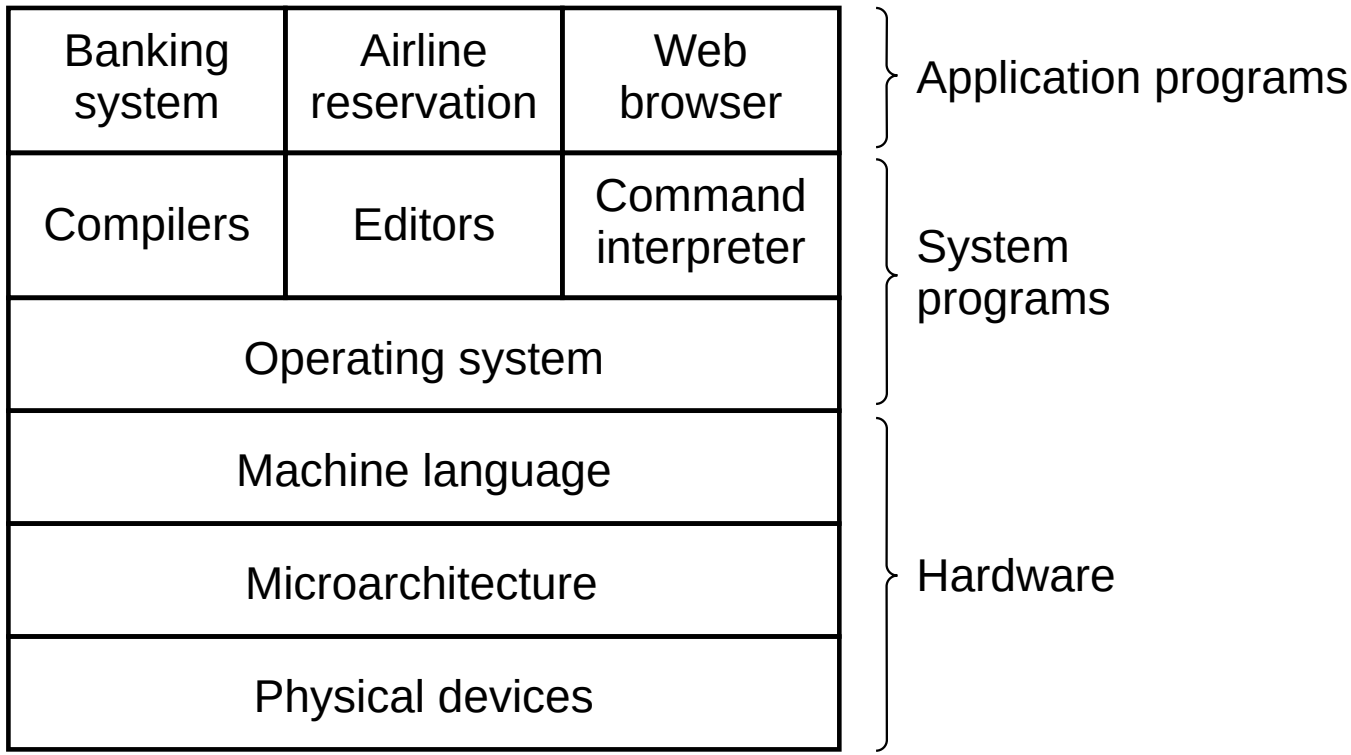
Teclado do PC



- Quando uma tecla é premida o microcontrolador do teclado gera um *scancode* que é colocado num *buffer*
  - Quando uma tecla deixa de ser premida o microcontrolador gera outro *scancode*
- Este *scancode* é posteriormente transferido para o registo `OUT_BUF` controlador do teclado (KBC)
- O KBC pode gerar uma interrupção
  - Alternativamente pode usar-se *polling*
- A identificação do carácter premido depende dos *scancodes* previamente recebidos, e do *mapa de teclado*

Modelo em camadas dum sistema computacional

- O **sistema operativo** é uma “camada” de SW que assenta sobre o HW



  - Cada nível define uma **máquina virtual**, excepto o nível mais baixo.

Parentesis: Modelo em camadas

- Estratégia típica na resolução de problemas complexos (outro exemplo são as redes de computadores)
  - Suporta **abstracção**
- Vantagens deste modelo

  Decomposição Um problema "intratável" é decomposto em problemas mais pequenos e solúveis

  Modularidade É relativamente fácil acrescentar funcionalidades ou alterar a implementação, desde que se **preservem as interfaces**

Instruction Set Architecture (ISA) Level

- Define o o HW e o conjunto de instruções visiveis para um programador de *assembly*
- Compreende um conjunto de 50 a algumas centenas de instruções para:
  - Transferir dados entre os diferentes componentes;
  - Realizar operações aritméticas e lógicas;
  - Controlar o fluxo de instruções
- A este nível operações de entrada e saída são realizadas escrevendo e lendo registos dos controladores dos dispositivos de E/S

Nível do Sistema Operativo

Define a interface tipicamente acessível a um programador.

- Oferece um conjunto de operações – chamadas ao sistema *(system calls)* – que fornecem um nível de abstracção muito mais conveniente.
  - A maioria das operações do nível ISA continua acessível.
  - Algumas contudo são escondidas, essencialmente por razões de segurança.

Abstracções Oferecidas por um SO

- Utilizador (em especial em sistemas interactivos);
- Processo;
- Ficheiro.

Utilizador

- Essencial em sistemas interactivos.
- Inclui entre outros os seguintes atributos:
  - nome (*username*);
  - identidade (*userid*);
  - grupos (*groupname* e *groupid*).
- Em sistemas multiutilizador, o conceito de utilizador é central para protecção de recursos (p.ex. ficheiros).
- O utilizador pode estar associado não só a uma pessoa como a uma função, p. ex. correio electrónico.

Processo

- Representa um programa em execução.
  - Um programa é um objeto passivo, tipicamente guardado em disco ou outros dispositivos de armazenamento de dados
  - Um processo é um objeto activo, cujo estado varia à medida que é executado
- Actualmente, quase todos os sistemas operativos são multi-processo (Linux, Windows XP, Windows Vista):
  Um computador pode executar vários programas ao "mesmo tempo".
- Um processo está associado a um utilizador, o seu dono: aquele que invoca o programa correspondente.
  A identidade do utilizador dono do processo determina os recursos a que um processo pode aceder, bem como o tipo de operações que pode realizar sobre esses recursos.

Processos: Virtualização do Processador

- Um processo oferece a um programa a ilusão de que tem ao seu dispôr todos os recursos do computador;
- Inclusivamente, pode dar a ilusão de que o computador físico tem mais recursos. P.ex.;
  - Mais memória do que a quantidade memória física existente.
  - Mais processadores do que o nº de "cores" existentes

Exemplo de Virtualização de Processador por Remzi Arpaci-Dusseau

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]){
    if (argc != 2) {
        fprintf(stderr, "usage: cpu <string>\n");
        exit(1);
    }
    char *str = argv[1];
    while (1) {
        printf("%s\n", str);
        Spin(2);
    }
    return 0;
}
```

Exemplo de Virtualização de Memória por Remzi Arpaci-Dusseau

```
int main(int argc, char *argv[]) {
    if (argc != 2) {
        fprintf(stderr, "usage: mem <value>\n");
        exit(1);
    }
    int *p;                      // memory for pointer is on "stack
    p = malloc(sizeof(int));  // malloc'd memory is on "heap"
    assert(p != NULL);
    printf("(pid:%d) addr of p:         0x%p\n",
           (int) getpid(), &p);
    printf("(pid:%d) addr stored in p: 0x%p\n",
           (int) getpid(), p);
    *p = atoi(argv[1]); // assign value to addr stored in p
    while (1) {
        usleep(1000000);
        *p = *p + 1;
        printf("(pid:%d) value of p: %d\n", getpid(), *p);
    }
    return 0;
}
```

Ficheiros

- Representam uma fonte/poço de informação
  - Para o utilizador não técnico são uma abstração do disco (e outros dispositivos de armazenamento de dados)
  - Podem contudo abstrair outros dispositivos de E/S
- Suportam três operações: leitura, escrita e execução
- Tipicamente organizados duma forma hierárquica, usando directórios:

  Ficheiros que contêm outros ficheiros.
- Cada ficheiro/directório tem um utilizador que é o seu dono
  - Usado para controlo de acesso

O SO como um Gestor de Recursos

Uma descrição alternativa do papel do SO

- Durante a sua execução, os programas fazem uso de recursos (CPU, memória, disco, . . .).
- A maioria dos “computadores” executa várias aplicações (possivelmente) de diferentes utilizadores em simultâneo:
  - O SO gere os recursos dum computador, facilitando a sua partilha entre diferentes aplicações.
  - As aplicações não podem aceder directamente aos recursos (todo o acesso é mediado pelo SO):
    - Evita-se que as aplicações interfiram na gestão dos recursos
    - Protege-se os recursos afectados a cada utilizador de acessos não autorizados por outros utilizadores

Leitura Adicional

- Secções 1.1, 1.2 e 1.4 de
  José Alves Marques e outros, *Sistemas Operativos*, FCA - Editora Informática, 2009
- Secções 1.1, 1.5 de
  Andrew Tanenbaum, *Modern Operating Systems*, 2nd Ed.
- Secções 1.1, 1.2, 1.6, 1.7, 1.8, 1.9, 2.1 e 2.2 de
  Silberschatz e outros, *Operating System Concepts*, 7th Ed.
- Ch. 2: Introduction, de *OS: Three Easy Pieces*, de Remzi Arpaci-Dusseau e Andrea Arpaci-Dusseau
- Outra documentação (transparências e enunciados dos TPs) na página da disciplina