IFSP – SÃO JOÃO DA BOA VISTA
CIÊNCIA DA COMPUTAÇÃO

Sistemas Operacionais

SEMANA 2

Prof.: Ederson Borges

Sistemas Operacionais

Estrutura e Arquitetura de SO

- Estrutura do SO
  - Elementos do SO
    - Software
    - Hardware
  - Arquitetura
  - Chamadas de Sistema

Estrutura do SO

- Estrutura do SO

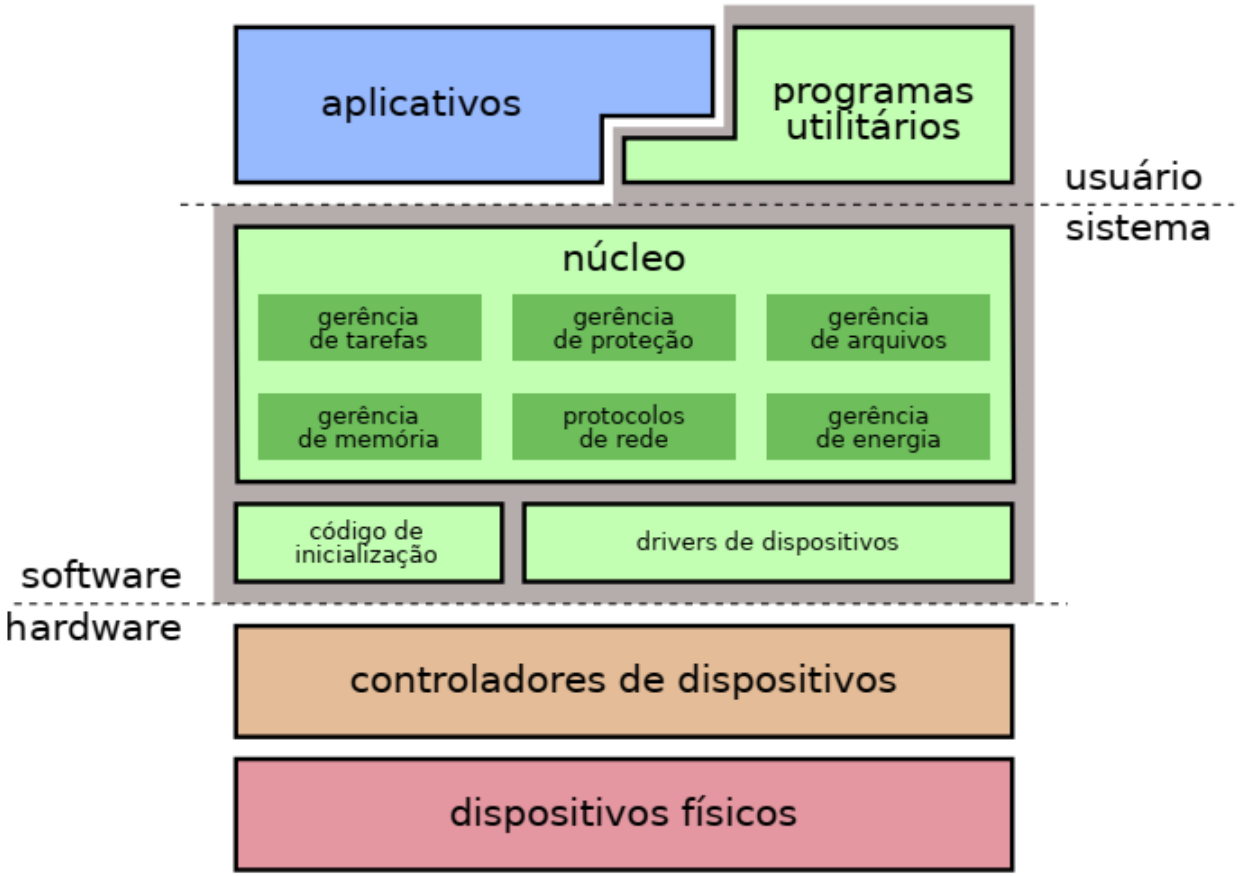


Figura 2.1: Estrutura de um sistema operacional típico

Sistemas Operacionais

Estrutura do SO

- Elementos (Software)
  - Núcleo
  - Código de inicialização (boot code)
  - Drivers
  - Programas Utilitários

Sistemas Operacionais

Estrutura do SO

- Núcleo
  - Parte central do SO (Kernel)
  - Gerencia recursos do hardware usados por aplicações
  - Implementa as principais abstrações utilizadas por aplicativos e programas utilitários
  - Executado em modo *privilegiado (modo sistema)*

Estrutura do SO

- Código de Inicialização
  - *Boot code*
  - Tarefas iniciais para carregar o núcleo do SO
    - Reconhecer dispositivos instalados
    - Testes
    - Configurações
  - Carrega o núcleo do sistema em memória
    - Inicia sua execução

Estrutura do SO

- Drivers
  - Códigos específicos a acessos a dispositivos físicos
    - Acesso a disco SATA
    - Porta USB
    - Placas Gráficas
    - Impressoras
    - Etc...
  - Construído (normalmente) pelo próprio fabricante
    - Fornecido já compilado para plataforma específica
      - Linux, Windows, ...

Sistemas Operacionais

Estrutura do SO

- Programas utilitários
  - Programas para uso do sistema operacional de forma cotidiana
    - Editores de texto, navegadores, calculadora, etc
  - Podem fornecer funcionalidades complementares
    - Formatação de disco, configuração de dispositivo, manipulação de arquivo, gerência de janelas, etc

Sistemas Operacionais

Estrutura do SO

- Modelo de estrutura real (Android – Linux)

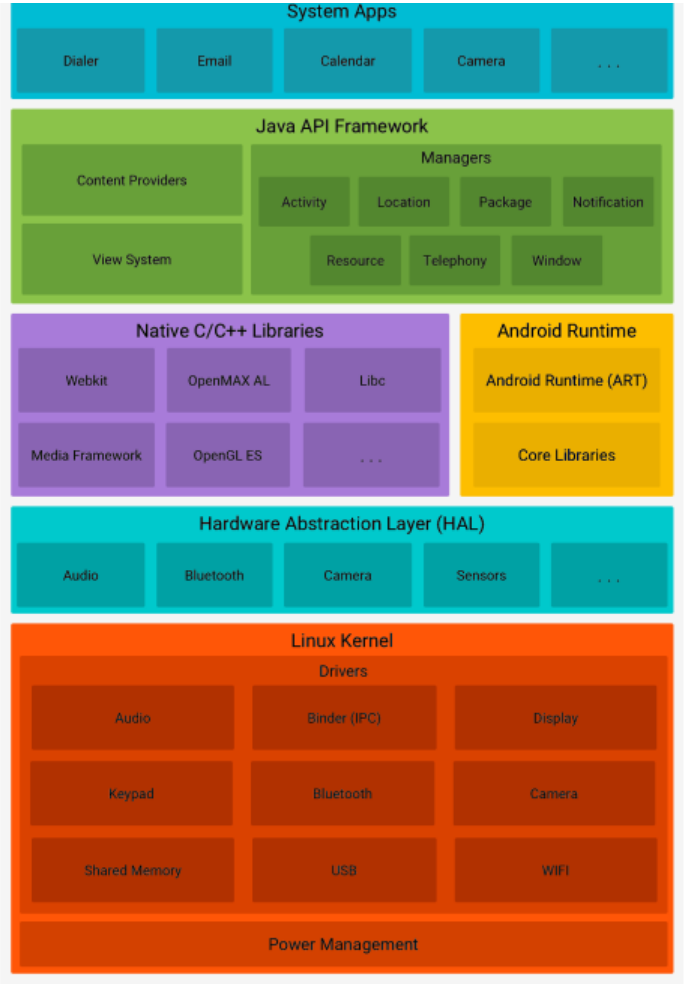


Figura 2.2: Estrutura de um sistema operacional Android [Google, 2018].

Sistemas Operacionais

Estrutura do SO

- Arquitetura de um computador
  - János (John) Von Neumann
    - Arquitetura Von Neumann
    - Anos 40
  - O programa reside na memória com os dados
  - Ligação por meio de barramentos
    - Controle, endereços, transferência de dados

Estrutura do SO

- Arquitetura de um computador

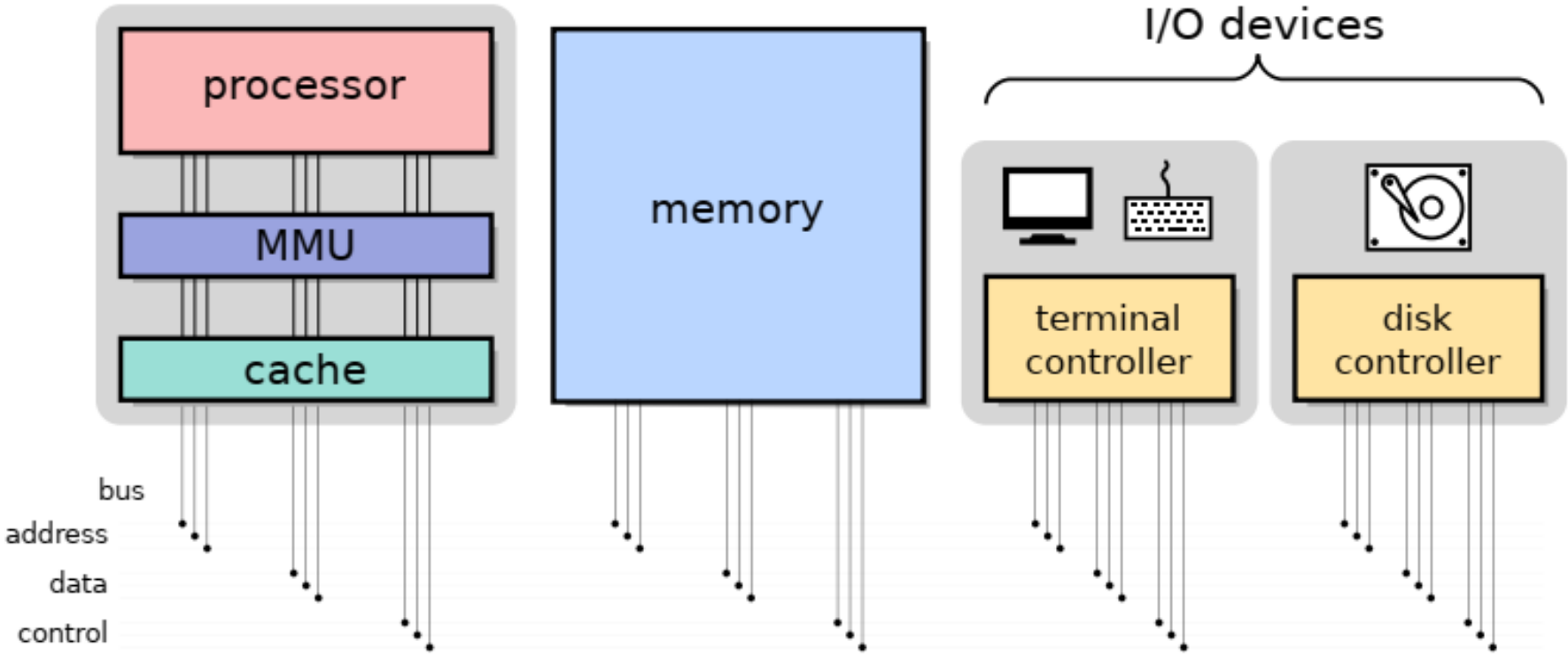
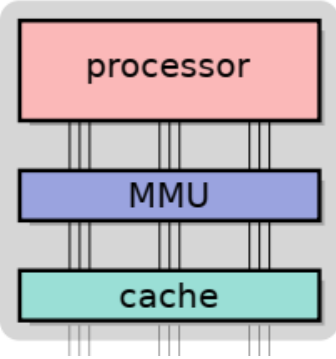


Figura 2.3: Arquitetura de um computador típico

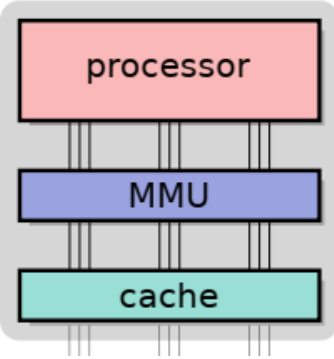
Estrutura do SO

- Arquitetura de um computador
  - CPU: Unidade Central de Processamento
  - Processador
    - Responsável por ler instruções
    - Processar
    - Enviar resultados
    - Forma tradicional
      - ULA (Unidade lógica e aritmética)
      - Registradores de dados
      - Registrador de funções (contador de programa, ponteiro de pilha)



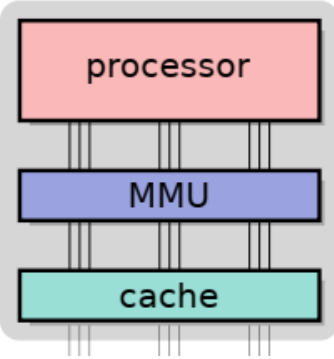
Estrutura do SO

- Arquitetura de um computador
  - Processador Moderno
    - Diversos núcleos (cores)
    - Processadores lógicos (*hyperthreading*)



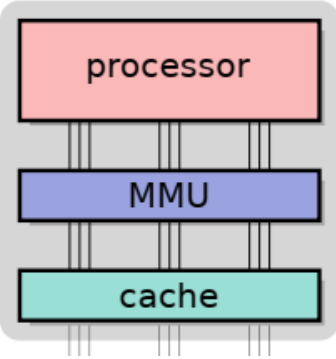
Estrutura do SO

- Arquitetura de um computador
  - Barramentos
    - Barramento de endereço
      - Posição de memória (ou dispositivo) a acessar
    - Barramento de controle
      - Operação a efetuar (leitura e escrita)
    - Barramento de dados
      - Transporte de dados entre processador e memória (ou controlador de dispositivo)



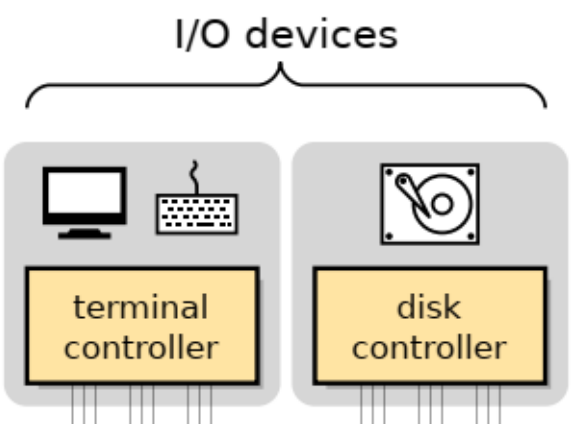
Estrutura do SO

- Arquitetura de um computador
  - Unidade de Gerência de Memória (MMU)
    - Mediadora de acesso do processador a memória
    - Pode estar fisicamente dentro do chip do processador
    - Responsável por analisar cada endereço de memória acessado pelo processador
    - Executar operação solicitada pelo processador
      - Leitura ou escrita



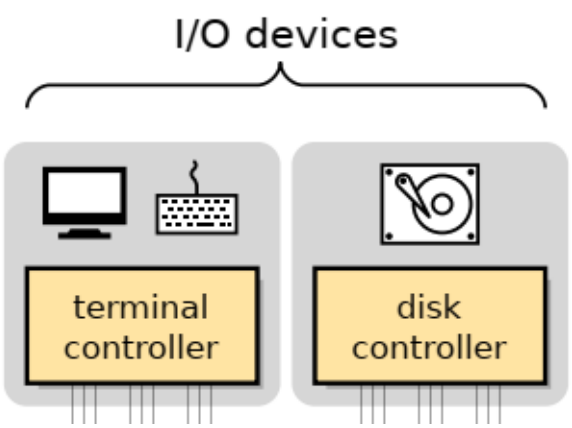
Estrutura do SO

- Arquitetura de um computador
  - Controladores
    - Circuitos eletrônicos específicos
    - Placa de vídeo
      - Monitor
    - Placa de rede
      - Rede
    - Controlador USB
      - Mouse, teclado, ...



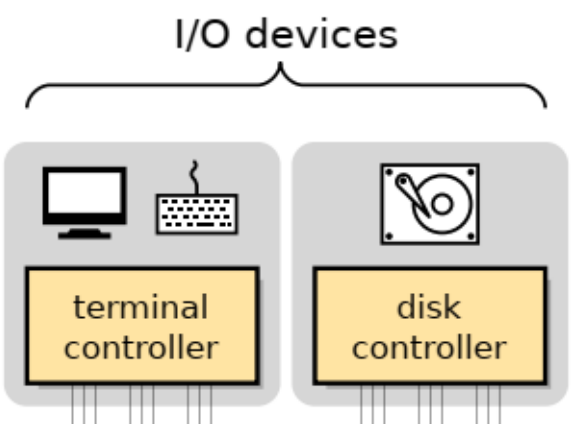
Estrutura do SO

- Arquitetura de um computador
  - Controladores
    - Cada dispositivo é representado pelo seu controlador
      - Acessos pelas *portas de entrada/saída*
        - » *Endereços*
        - » Faixas de endereços atribuídas a cada controlador



Estrutura do SO

- Arquitetura de um computador
  - Controladores
    - Toda comunicação é feita pelas porta de entrada/saída



Dispositivo	Endereços de acesso
temporizador	0040-0043
teclado	0060-006F
porta serial COM1	02F8-02FF
controlador SATA	30BC-30BF
controlador Ethernet	3080-309F
controlador	3000-303F

Tabela 2.1: Endereços de acesso a dispositivos (em hexadecimal).

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Processador comanda os acessos as portas de entrada e saída
    - O que fazer quando um dispositivo precisa informar o processador da ocorrência de um evento?
      - Aguardar até que o processador faça uma consulta no controlador
      - Notificar o processador
        - » IRQ (Interrupt Request)
        - » Barramento de controle

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - O que ocorre em uma interrupção?
      - Processador faz a suspensão do fluxo de execução
      - Acessa um endereço pré-definido
        - » Rotina de tratamento de interrupção
        - » Interrupt handler

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - *Interrupt Handler*
      - Responsável por tratar a interrupção
      - Executar as ações para atender o responsável pela requisição
      - Finalizado o tratamento o processador retorna ao fluxo anterior

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções

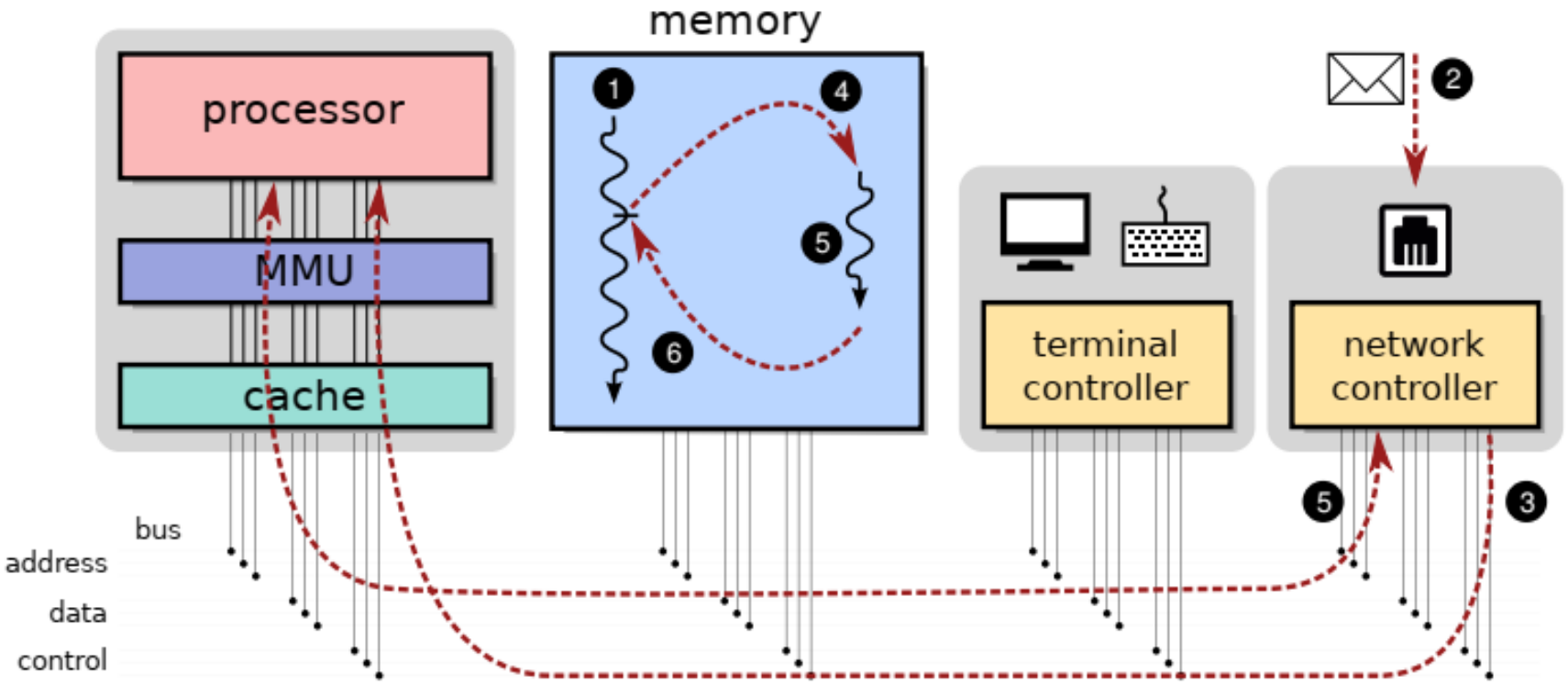


Figura 2.4: Roteiro típico de um tratamento de interrupção

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    1. O processador está em seu fluxo normal (executando tarefas)
    2. A placa ethernet recebe um pacote da rede
    3. Controlador ethernet executa IRQ
    4. Processamento vai para Interrupt Handler
    5. Interrupt Handler é executada (transfere pacote de redes para memória)
    6. Finaliza Interrupt Handler, volta ao fluxo anterior

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Possíveis interrupções
      - Clique do mouse
      - Pacote da rede
      - Fim de operação no disco (leitura, escrita)
    - Centenas (ou até milhares) de interrupções são recebidas por SEGUNDO!!!

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Como identificar cada interrupção?
      - Número inteiro (hardware)
      - Cada IRQ tem sua própria rotina de tratamento de interrupção
      - Arquiteturas atuais utilizam uma tabela
        - » Tabela de Interrupções (IVT – Interrupt Vector Table)
        - » Cada registro na tabela representa uma rotina de tratamento de interrupção

Sistemas Operacionais

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Tabela de interrupções
      - Pentium (2005)

IRQ	Descrição
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	Intel reserved
16	floating point error
17	alignment check
18	machine check
19-31	Intel reserved
32-255	maskable interrupts (devices & exceptions)

Sistemas Operacionais

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Exceções
      - Eventos gerados pelo próprio processador
      - Ocasionam o desvio da execução usando o mesmo mecanismo das interrupções
      - Podem ser:
        - » Instruções ilegais
        - » Divisão por zero
        - » Erros de aplicações
      - Utilizam a mesma tabela de endereços de funções

Sistemas Operacionais

Estrutura do SO

- Arquitetura de um computador
  - Interrupções e Exceções
    - Utilizadas para melhorar o desempenho do processador
      - Sem interrupção o processador teria que “varrer” todos os dispositivos na busca por eventos a serem tratados
    - Permite funções entrada/saída assíncrona
      - Não é necessário aguardar sua conclusão
      - Dispositivo gera a interrupção assim que concluir a tarefa

Estrutura do SO

- Arquitetura de um computador
  - Níveis de privilégio
    - Já vimos que o núcleo tem nível privilegiado
    - Todo acesso feito pelo núcleo e por drivers deve ter esse acesso
    - Aplicativos devem ter acesso restrito
      - Evitar que corrompam espaços de memória e operações ilegais nos dispositivos
    -

Estrutura do SO

- Arquitetura de um computador
  - Níveis de privilégio
    - Intel x86

Estrutura do SO

- Arquitetura de um computador
  - Níveis de privilégio
    - Intel x86
      - Nível núcleo (privilegiado): Todas as funcionalidades do processador estão disponíveis
        - » Acesso direto a memória e registradores do processador
      - Nível usuário: subconjunto das instruções do processador podem ser utilizadas
        - » Ações como RESET e IN/OUT são negadas
        - » Ao executar, uma exceção será gerada

Estrutura do SO

- Arquitetura de um computador
  - Níveis de privilégio
    - Objetivo é proteger o sistema (inclusive dispositivos)
    - Aplicações executam de forma “isolada” umas das outras

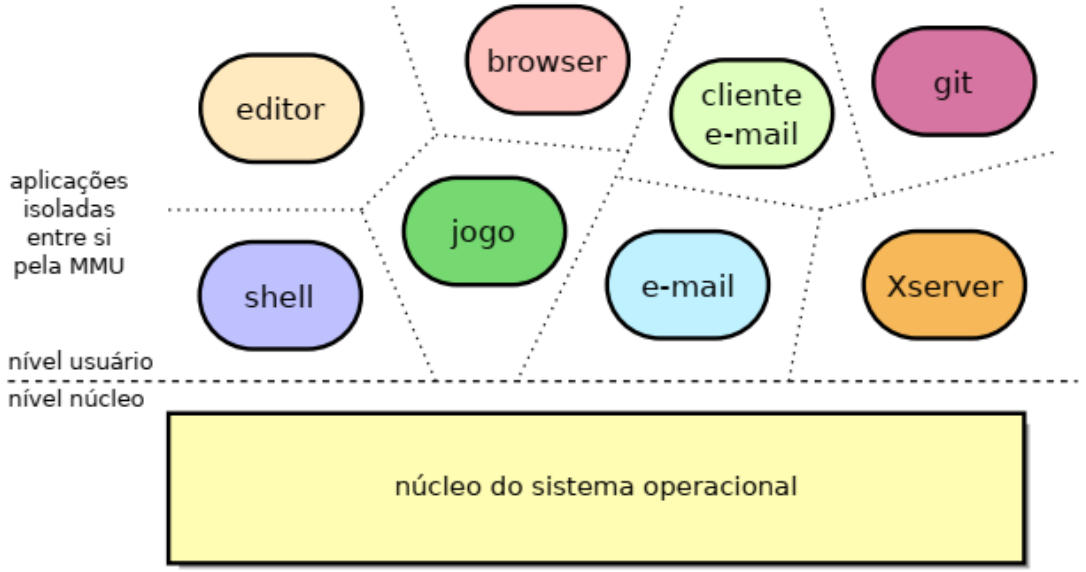


Figura 2.6: Separação entre o núcleo e as aplicações

Estrutura do SO

- Atividades para revisão
  - O que diferencia o núcleo do restante do sistema operacional?
  - A operação em modo usuário permite ao processador executar somente parte das instruções disponíveis em seu conjunto de instruções. Quais das seguintes operações não deveriam ser permitidas em nível usuário? Por quê?

a) Ler uma porta de entrada/saída
b) Efetuar uma divisão inteira
c) Escrever um valor em uma posição de memória
d) Ajustar o valor do relógio do hardware
e) Ler o valor dos registradores do processador
f) Mascarar uma ou mais interrupções

Sistemas Operacionais

Estrutura do SO

- Chamadas de Sistema
  - Com o isolamento das aplicações em “área de memória específica” nos dá:
    - Confiabilidade ao sistema
    - Robustez (uso sem concorrência)
  - Não existe interferência entre aplicações e nem com o SO

Estrutura do SO

- Chamadas de Sistema
  - Como utilizar (invocar), por uma aplicação, rotinas que são oferecidas pelo núcleo?
    - Lembrando:
      - Núcleo está em memória protegida
      - Apenas aplicações operando no *modo privilegiado* podem acessar essas áreas

Estrutura do SO

- Chamadas de Sistema
  - Como utilizar (invocar), por uma aplicação, rotinas que são oferecidas pelo núcleo?
    - Solução: Interrupção
    - Quando ocorre uma interrupção o processador suspende a tarefa atual e executa o código de um endereço pré-definido
    - O processador está em modo privilegiado
      - Pode acessar qualquer área de memória
    - Mecanismo chamado
      - Interrupção de software
      - *Trap*

Estrutura do SO

- Chamadas de Sistema
  - Como utilizar (invocar), por uma aplicação, rotinas que são oferecidas pelo núcleo?
    - Quando uma rotina é ativada usando uma *trap*
      - Temos uma CHAMADA DE SISTEMA

Sistemas Operacionais

Estrutura do SO

- Chamadas de Sistema
  - SO definem chamadas de sistema para todos as operações com acesso a recursos de baixo nível
    - Periféricos
    - Arquivos
    - Alocação de memória
    - Etc
  - Também estão incluídas as abstrações lógicas
    - Criação e encerramento de tarefas
    - Operadores de sincronização
    - etc

Sistemas Operacionais

Estrutura do SO

- Chamadas de Sistema
  - Como executo uma chamada de sistema?
    - Bibliotecas do sistema
      - System library
      - Preparação de parâmetros
      - Faz (invoca) a chamada
      - Devolve resultado para a aplicação

Estrutura do SO

- Chamadas de Sistema
  - Processadores modernos
    - X86 (32 bits)
      - sysenter
      - Sysexit
    - X86 (64 bits)
      - syscall
      - sysret

Sistemas Operacionais

Estrutura do SO

- Chamadas de Sistema
  - Como é executada uma chamada de sistema?
    - Antes de tudo registradores do processador são carregados com valores específicos
      - Número de operação
      - Endereço dos parâmetros da chamada
      - Etc.
    - Cada sistema operacional tem seus próprios parâmetros

Estrutura do SO

- Chamadas de Sistema
  - Um exemplo em C para Linux

```
#include <unistd.h>

int main (int argc, char *argv[])
{
  write (1, "Hello World!\n", 13) ; /* write string to stdout */
  _exit (0) ;                       /* exit with no error */
}
```

Sistemas Operacionais

Estrutura do SO

- Chamadas de Sistema
  - Como ficaria em Assembly (x86)

```
; to assembly and link (in Linux 64 bit):
; nasm -f elf64 -o hello.o hello.asm; ld -o hello hello.o

section .data
msg db 'Hello World!', 0xA  ; output string (0xA = "\n")
len equ 13                  ; string size

section .text

global _start

_start:
   mov rax, 1               ; syscall opcode (1: write)
   mov rdi, 1               ; file descriptor (1: stdout)
   mov rsi, msg             ; data to write
   mov rdx, len             ; number of bytes to write
   syscall                  ; make syscall

   mov rax, 60              ; syscall opcode (60: _exit)
   mov rdi, 0               ; exit status (0: no error)
   syscall                  ; make syscall
```

Estrutura do SO

- Chamadas de Sistema
  - E no sistema operacional/computador?

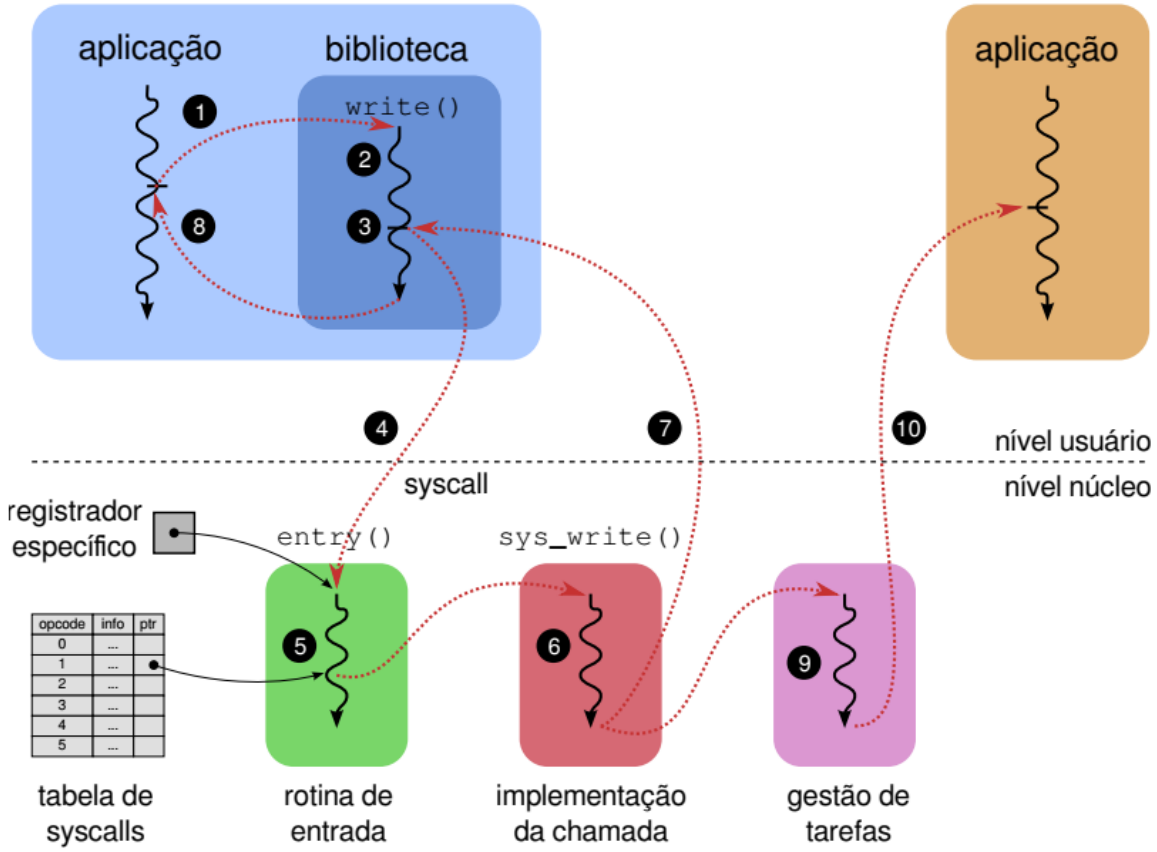


Figura 2.7: Roteiro típico de uma chamada de sistema

Sistemas Operacionais

Chamadas de Sistema

- Gestão de processos: criar, carregar código, terminar, esperar, ler/mudar atributos
- Gestão da memória: alocar/liberar/modificar áreas de memória
- Gestão de arquivos: criar, remover, abrir, fechar, ler, escrever, ler/mudar atributos
- Comunicação: criar/destruir canais de comunicação, receber/enviar dados
- Gestão de dispositivos: ler/mudar configurações, ler/escrever dados
- Gestão do sistema: ler/mudar data e hora, desligar/suspender/reiniciar o sistema

Sistemas Operacionais

Chamadas de Sistema

- Atividades
  - Quais as diferenças entre interrupções, exceções e traps?
  - O comando em linguagem C ***fopen*** é uma chamada de sistema ou uma função de biblioteca? Por quê?
  - Considerando um processo em um sistema operacional com proteção de memória entre o núcleo e as aplicações, indique quais das seguintes ações do processo teriam de ser realizadas através de chamadas de sistema, justificando suas respostas:

(a) Ler o relógio de tempo real do hardware.

(b) Enviar um pacote através da rede.

(c) Calcular uma exponenciação.

(d) Preencher uma área de memória do processo com zeros.

(e) Remover um arquivo do disco.