Introdução
Instruções Atômicas
Instruções Privilegiadas
Modos de Execução
Proteção de Memória
Chamadas de Sistema
Interrupções e Exceções
Conclusões
Referências

# Suporte da Arquitetura ao Sistema Operacional

Marília Felippe Chiozo

Novembro de 2007

# Roteiro

1. Introdução
2. Instruções Atômicas
3. Instruções Privilegiadas
4. Modos de Execução
5. Proteção de Memória
6. Chamadas de Sistema
7. Interrupções e Exceções
8. Conclusões
9. Referências

# Introdução – a relação SO-hardware

- SO regula o acesso à CPU e media as operações de entrada e saída.
- Novas tecnologias de hardware podem ser inúteis até que o SO as suporte.
- Novas tecnologias de SO podem permanecer como idéias até que surja hardware no qual implementá-las.
- Funcionalidades esperadas de um SO, tais como proteção de memória e entrada e saída, possuem implementação dependente de arquitetura. Outras, como sincronização e troca de contexto, podem ter seu desempenho significantemente aumentado por suporte em hardware.

# Introdução – a relação SO-hardware

- SO regula o acesso à CPU e media as operações de entrada e saída.
- Novas tecnologias de hardware podem ser inúteis até que o SO as suporte.
- Novas tecnologias de SO podem permanecer como idéias até que surja hardware no qual implementá-las.
- Funcionalidades esperadas de um SO, tais como proteção de memória e entrada e saída, possuem implementação dependente de arquitetura. Outras, como sincronização e troca de contexto, podem ter seu desempenho significantemente aumentado por suporte em hardware.

# Introdução – a relação SO-hardware

- SO regula o acesso à CPU e media as operações de entrada e saída.
- Novas tecnologias de hardware podem ser inúteis até que o SO as suporte.
- Novas tecnologias de SO podem permanecer como idéias até que surja hardware no qual implementá-las.
- Funcionalidades esperadas de um SO, tais como proteção de memória e entrada e saída, possuem implementação dependente de arquitetura. Outras, como sincronização e troca de contexto, podem ter seu desempenho significantemente aumentado por suporte em hardware.

# Introdução – a relação SO-hardware

- SO regula o acesso à CPU e media as operações de entrada e saída.
- Novas tecnologias de hardware podem ser inúteis até que o SO as suporte.
- Novas tecnologias de SO podem permanecer como idéias até que surja hardware no qual implementá-las.
- Funcionalidades esperadas de um SO, tais como proteção de memória e entrada e saída, possuem implementação dependente de arquitetura. Outras, como sincronização e troca de contexto, podem ter seu desempenho significantemente aumentado por suporte em hardware.

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Introdução – características arquiteturais úteis ao SO

- timer
- instruções atômicas
- instruções privilegiadas e múltiplos modos de execução
- proteção de memória
- chamadas de sistema
- interrupções e exceções
- TLBs

# Instruções Atômicas

A motivação para executar código de maneira atômica é a possibilidade de uma interrupção no fluxo de execução comprometer a correção do programa.

- Projetadas principalmente para sincronização.
- Exemplos: “test-and-set”, “read-modify-write”.

# Instruções Atômicas

A motivação para executar código de maneira atômica é a possibilidade de uma interrupção no fluxo de execução comprometer a correção do programa.

- Projetadas principalmente para sincronização.
- Exemplos: “test-and-set”, “read-modify-write”.

# Instruções Privilegiadas

- Nem todas as instruções devem, idealmente, ser acessíveis a qualquer processo.
- Implementadas conjuntamente com modos de execução.
- Exemplos: instruções de entrada e saída, instruções relacionadas a cache, instruções que lidam com a configuração de interrupções.

# Instruções Privilegiadas

- Nem todas as instruções devem, idealmente, ser acessíveis a qualquer processo.
- Implementadas conjuntamente com modos de execução.
- Exemplos: instruções de entrada e saída, instruções relacionadas a cache, instruções que lidam com a configuração de interrupções.

# Instruções Privilegiadas

- Nem todas as instruções devem, idealmente, ser acessíveis a qualquer processo.
- Implementadas conjuntamente com modos de execução.
- Exemplos: instruções de entrada e saída, instruções relacionadas a cache, instruções que lidam com a configuração de interrupções.

# Modos de Execução

- Mecanismo utilizado para determinar quando é possível executar determinada instrução privilegiada.
- No mínimo dois modos (user e kernel) são implementados; a IA32 suporta quatro.
- A transição entre modos de execução é dependente de arquitetura.

# Modos de Execução

- Mecanismo utilizado para determinar quando é possível executar determinada instrução privilegiada.
- No mínimo dois modos (user e kernel) são implementados; a IA32 suporta quatro.
- A transição entre modos de execução é dependente de arquitetura.

# Modos de Execução

- Mecanismo utilizado para determinar quando é possível executar determinada instrução privilegiada.
- No mínimo dois modos (user e kernel) são implementados; a IA32 suporta quatro.
- A transição entre modos de execução é dependente de arquitetura.

# Proteção de Memória

- Objetivo: evitar interações maliciosas e/ou decorrentes de bugs entre programas (inclusive o próprio SO).
- Exemplo: uso de registradores para limitar a faixa de endereços acessível ao processo atual.
- Exemplo: paginação.

# Proteção de Memória

- Objetivo: evitar interações maliciosas e/ou decorrentes de bugs entre programas (inclusive o próprio SO).
- Exemplo: uso de registradores para limitar a faixa de endereços acessível ao processo atual.
- Exemplo: paginação.

# Proteção de Memória

- Objetivo: evitar interações maliciosas e/ou decorrentes de bugs entre programas (inclusive o próprio SO).
- Exemplo: uso de registradores para limitar a faixa de endereços acessível ao processo atual.
- Exemplo: paginação.

# Chamadas de Sistema

- Comunicação entre os aplicativos comuns (modo usuário) e o SO (modo kernel).
- Exemplos: requisições de entrada e saída, gerenciamento de processos, comunicação inter-processos.
- Envolvem mudança de modo de execução e troca de contexto.

# Chamadas de Sistema

- Comunicação entre os aplicativos comuns (modo usuário) e o SO (modo kernel).
- Exemplos: requisições de entrada e saída, gerenciamento de processos, comunicação inter-processos.
- Envolvem mudança de modo de execução e troca de contexto.

# Chamadas de Sistema

- Comunicação entre os aplicativos comuns (modo usuário) e o SO (modo kernel).
- Exemplos: requisições de entrada e saída, gerenciamento de processos, comunicação inter-processos.
- Envolvem mudança de modo de execução e troca de contexto.

# Interrupções e Exceções

- Operação similar às chamadas de sistema, mas propósito e invocação diferentes.
- Resposta a eventos do sistema, tais como erros ou sinais enviados por dispositivos.
- Imprevisíveis para os processos.

# Interrupções e Exceções

- Operação similar às chamadas de sistema, mas propósito e invocação diferentes.
- Resposta a eventos do sistema, tais como erros ou sinais enviados por dispositivos.
- Imprevisíveis para os processos.

# Interrupções e Exceções

- Operação similar às chamadas de sistema, mas propósito e invocação diferentes.
- Resposta a eventos do sistema, tais como erros ou sinais enviados por dispositivos.
- Imprevisíveis para os processos.

# Conclusões

- Suporte arquitetural ao SO aumenta a performance.
- Algumas funcionalidades são impossíveis a baixo custo sem suporte no hardware.
- Problemas de projeto podem comprometer o esforço de suporte a determinada característica, como por exemplo na troca de contexto via hardware da IA32.

# Conclusões

- Suporte arquitetural ao SO aumenta a performance.
- Algumas funcionalidades são impossíveis a baixo custo sem suporte no hardware.
- Problemas de projeto podem comprometer o esforço de suporte a determinada característica, como por exemplo na troca de contexto via hardware da IA32.

# Conclusões

- Suporte arquitetural ao SO aumenta a performance.
- Algumas funcionalidades são impossíveis a baixo custo sem suporte no hardware.
- Problemas de projeto podem comprometer o esforço de suporte a determinada característica, como por exemplo na troca de contexto via hardware da IA32.

# Referências

[1] Intel *Intel Architecture Software Developer's Manual, Volume 3*. 1999.

[2] Robin, J. S. e Irvine, C. E. *Analysis of the Intel Pentium's ability to support a secure virtual machine monitor*. In *Proceedings of the 9th USENIX Security Symposium*. Denver, 2000.

[3] Ganssle, J. G. *Interrupt Predictability*. 1995. URL: http://www.avocetsystems.com/company/articles/magazine/aintrp.htm

[4] Swift, M. et al. *Computer Architecture and Operating Systems* URL: http://pages.cs.wisc.edu/c̃s537-2/lectures/02-arch.pdf