RISC

Arquitetura e Organização de Computadores

Prof. Lucas de Oliveira Teixeira

UEM

Introdução

Introdução

Grandes avanços da computação: Conceito de família

- Introduzido pela IBM no System/360 - 1964.
- Separação de arquitetura e sua implementação (organização).
- Implementações com preço/desempenho diferentes.

Introdução

Grandes avanços da computação: Unidade de controle microprogramada

- Sugerida por Wilkes - 1951.
- Produzido pela IBM no S/360 - 1964.
- Facilita a tarefa de projetar e implementar uma unidade de controle.

Introdução

Grandes avanços da computação: Memória cache

- Aumenta dramaticamente o desempenho da hierarquia de memória.
- Introduzida pela IBM no S/360 Model 85 - 1968

Introdução

Grandes avanços da computação: Pipelining

- Introduzido pela Intel no 486 - 1989.
- Trouxe paralelismo na natureza essencialmente sequencial de um programa de instruções de máquina.

Introdução

Grandes avanços da computação: Múltiplos processadores

- Introduzido pela Intel no Celeron Dual-Core em 2003.
- Introduzido pela AMD no Athlon 64 X2 em 2005.

Introdução

Grandes avanços da computação: RISC!

CISC

CISC

Força motriz do CISC:

- Custos de software são superiores aos custos de hardware.
- Linguagens de alto nível cada vez mais poderosas e complexas: High Level Languages (HLL).
- HLL gera uma lacuna semântica: Diferença entre operações fornecidas em linguagens de alto nível e aquelas fornecidas na arquitetura de computador. Isto leva a:
  - Grandes conjuntos de instruções de máquina.
  - Mais modos de endereçamento.
  - Implementações em hardware de instruções de HLL.

CISC

Objetivos CISC:

- Facilitar a tarefa do programador de compiladores.
- Melhorar a eficiência da execução com operações complexas implementadas em microcódigo.
- Admitir HLLs cada vez mais complexas e sofisticadas.

CISC

Problemas do CISC:

- Instruções de máquina complexas são difíceis de serem encontradas/compreendidas pelo compilador no código HLL, otimização é mais difícil.
- Programa ocupa menos memória, mas memória agora é barata!
- Mais instruções exigem opcodes mais longos.
- Muitos tipos de endereçamento exigem mais bits nas instruções.
- Unidade de controle fica mais complexa que precisa de um microprograma maior.
- Porém, geralmente usa-se instruções mais simples e elas acabam levando mais tempo para executar.

RISC

RISC

Na prática um programa típico contém bastante instruções de:

- Atribuição.
- Instruções condicionais (if, laço, ...).
- Chamada e retorno de procedimento (lento).
- Normalmente o acesso acontece em variáveis locais.

RISC

Assim, RISC adotou os princípios:

- Grande número de registradores para resolver as muitas referências a operandos locais.
- Projeto cuidadoso dos pipelines para resolver os problemas com muitas instruções de desvio, previsão de desvio, etc.
- Conjunto de instruções simplificado (reduzido) para facilitar a otimização pelo compilador.

RISC

Características RISC:

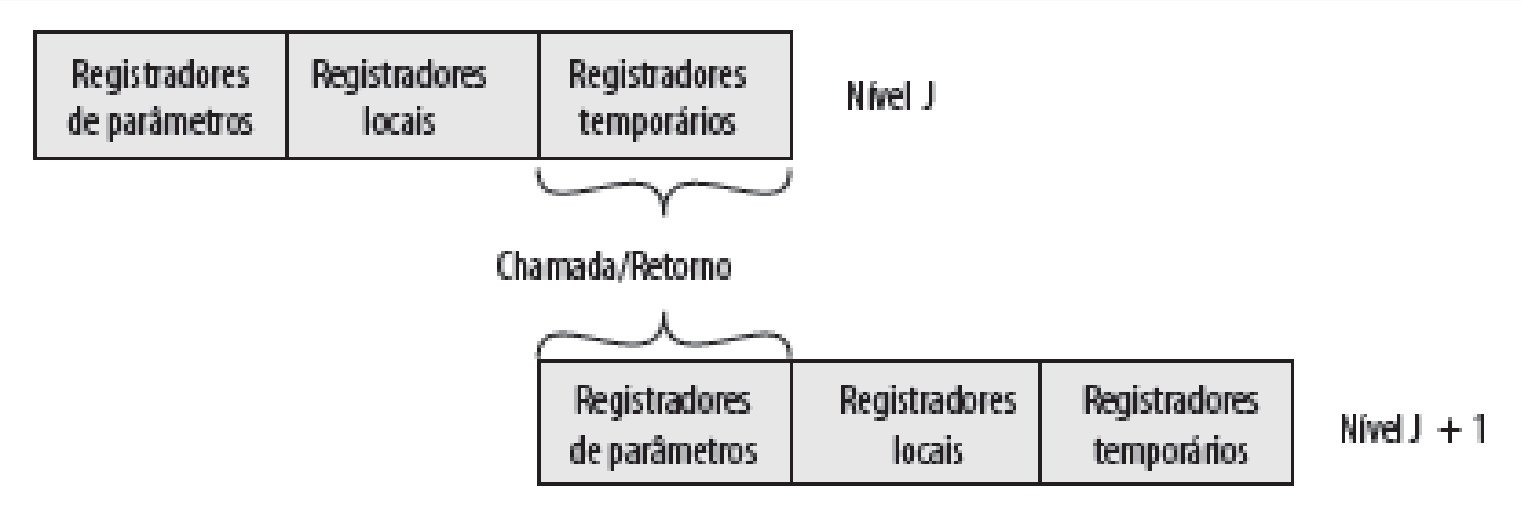
- Grande número de registradores.
- Qualquer instrução é executada em um ciclo.
- Operações são feitas de registrador a registrador.
- Poucos modos de endereçamento (geralmente apenas um).
- Poucos formatos de instrução (geralmente apenas um).
- Projeto fisicamente conectado (sem microcódigo).
- Posição dos campos na instrução sempre fixo.
- Mais tempo/esforço de compilação.

RISC

Janela de registradores:

- Múltiplos conjuntos pequenos de registradores.
- Chamadas de procedimentos trocam para um conjunto diferente de registradores, isso permite a passagem de parâmetros sem movimentar dados.
- Três tipos de registradores: parâmetros, locais e temporários (parâmetros e resultados para o nível abaixo da chamada de procedimentos).

Janela de registradores:



Janela de registradores:

- O esquema de janelas oferece uma organização eficiente para registradores com variáveis locais. Mas e para as globais?
- Duas estratégias:
  - Alocadas pelo compilador para a memória: Ineficaz para variáveis acessadas frequentemente.
  - Um conjunto de registradores específicos apenas para variáveis globais: Estes registradores seriam visíveis para todos os programas.

RISC

Pipeline:

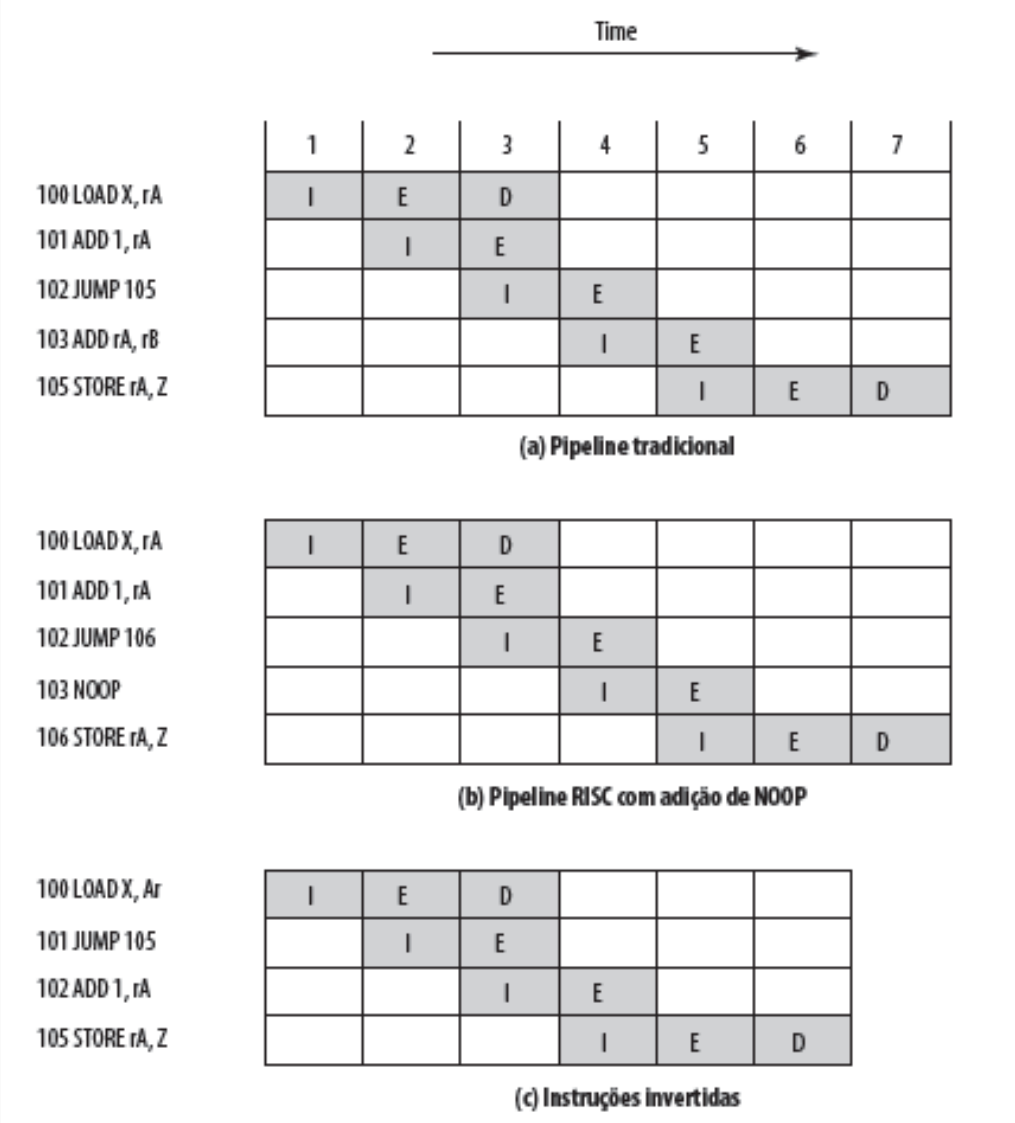
- As operações são feitas entre registradores.
- Assim, envolve duas fases: busca e execução.
- Para operações de carregar (load) e armazenar (store), envolve três fases: busca, execução (cálculo do endereço de memória) e memória.

RISC

Desvio atrasado:

- O desvio não é tomado antes da execução da instrução seguinte.
- Essa instrução seguinte é o delay slot.
- Esse delay spot é preenchido com um NOOP (ruim) ou com uma instrução válida independente anterior (reordenação).

Desvio atrasado:



RISC

Carregamento (load) atrasado:

- Registrador a alvo de LOAD é bloqueado pelo processador.
- Continue o fluxo de execução até o registrador ser exigido.
- Fica então ocioso até que LOAD seja completo.
- Rearranjar instruções permite que um trabalho útil seja feito enquanto ocorre leitura (trabalho do compilador).

RISC

Laço desenrolado:

- Replica corpo do laço em um número de vezes.
- Faz iteração do loop menos vezes.
- Aumenta paralelismo das instruções.
- Melhor localidade de registradores e cache de dados.

RISC vs CISC

RISC vs CISC

Qual arquitetura é melhor?

- Trabalhos foram feitos com dois tipos de análises:
- Quantitativa: Compara tamanhos de programa e velocidades de execução.
- Qualitativa: Examina questões de suporte para linguagem de alto nível e uso otimizado do estado atual de VLSI (Very-large-scale integration) – Uso dos transistores.

RISC vs CISC

Problemas com a comparação:

- Não há pares de máquinas RISC e CISC comparáveis diretamente.
- Não existe nenhum conjunto definitivo de programas de teste.
- É difícil separar efeitos de hardware dos efeitos provenientes da capacidade de escrever compiladores.
- A maioria das comparações são feitas sobre máquinas brinquedo ao invés de produtos comerciais. Porque? Maioria das máquinas comerciais é uma mistura.