ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

Arquitetura RISC X CISC X POST-RISC

Profª. Fabiana F F Peres

Apoio: Camile Bordini

Um pouco de História…

- 1964: Introdução da família de processadores System/360 da IBM
  - Primeira arquitetura comercial de computadores que usava **microprogramação** (**interpretação** de instruções de arquitetura para execução em hardware).

- O uso de linguagens de alto nível aumentava a complexidade dos compiladores

Crescimento da microprogramação

- Elevou-se o nível de abstração da linguagem de máquina
  - Utilização de **conjunto de instruções complexas**
  - Cuja implementação era muito simples devido ao emprego do **interpretador**

- Até o final da década de 70:
  - Crescimento no uso de **microprogramação** em processadores
  - Quase ninguém pensava em projetar máquinas “mais simples”

Mais um pouco de História...

- Nos anos 70 surgiram ferramentas para avaliar o desempenho dos computadores

- Provou-se que a maioria das aplicações utilizava MUITAS das instruções básicas do conjunto total de instruções na arquitetura...

- ... e POUCAS das instruções complexas disponibilizadas pela microprogramação

Mais um pouco de História...

- Início do contraponto à filosofia que privilegiava instruções complexas:
  - Grupo de pesquisa da IBM – protótipo de minicomputador (**801**)

- Apesar do **801** nunca ter sido comercializado, e dos resultados terem sido publicados somente anos depois…
- … vazaram informações sobre ele e outros grupos começaram a investigar arquiteturas similares!

Terminologias

- Em 1980, David Patterson[1] e Carlo Séquin desenvolveram processadores VLSI **que <u>não</u> utilizavam interpretação**
  - Surgiu o termo ***Reduced Instruction Set Computer* (RISC)**
    - E o chip desenvolvido foi o ***RISC I.*** Quase imediatamente seguido pelo ***RISC II***

- Em 1981, John Hennessy projetou e fabricou o chip **MIPS** (bastante diferente dos anteriores)
  - Sucesso comercial - chip SPARC e chip MIPS

1. https://en.wikipedia.org/wiki/Turing_Award

Terminologias

- **Todos esses chips**: muito diferentes dos comercializados na época
  - Não precisavam ser compatíveis
  - Projetistas livres para escolher novos conjuntos de instruções de forma a maximizar a performance

- *Obs: até então não existiam terminologias para definir as arquiteturas de computadores*

RISC – Reduced Instruction Set Computer

- Foco inicial:
  - **instruções serem simples**, que pudessem ser **EXECUTADAS rapidamente**

- Mas logo ficou clara a importância:
  - de se projetar instruções que pudessem ser **INICIADAS dentro de 1 seg**

__Tempo__ que uma instrução leva para ser executada começou a importar menos do que a __quantidade__ de instruções iniciadas antes de outras terminarem

RISC – Reduced Instruction Set Computer

- Chamou a atenção o seu pequeno número de instruções (em torno de 50 inicialmente)
  - Em contraste com as 200, 300 instruções dos processadores que lideravam o mercado na época
  - Por isso o termo **RISC**!

> *Atualmente, o tamanho do conjunto de instruções não é o mais importante, mas mesmo assim o termo RISC continua sendo usado.*

- Todas muito simples, capazes de serem executadas em um ciclo do caminho de dados

Filosofia RISC

Mesmo que uma máquina RISC precisasse de 4 ou 5 instruções pra fazer o que uma CISC faria em apenas uma, se as instruções RISC fossem 10 vezes mais rápidas (por não serem interpretadas), a máquina RISC vencia.

Além disso...

- Microprogramas
  - Eram armazenados em uma memória de controle, de somente leitura (inicialmente, mais rápidas que as memórias principais)

- Surgiram memórias principais com desempenho similar às memórias de controle
  - Interpretação perdeu uma vantagem

Microcódigo não é mágico, pois depende de um interpretador!

Princípios de Projeto RISC

Princípios de Projeto RISC

- Há alguns princípios de projeto RISC que os arquitetos de processadores devem procurar seguir:

1. Analisar as aplicações para encontrar as operações (instruções) chaves

2. Projetar um caminho de dados que seja ótimo para as operações chaves, ou seja, o tempo deve ser o menor possível para executar as instruções mais utilizadas

Princípios de Projeto RISC

3. Adicionar novas instruções somente se elas não diminuirem a velocidade da máquina

4. Repetir este processo para outros recursos, como memória *cache*, gerenciamento de memória e co-processadores

RISC – Reduced Instruction Set Computer

- Exemplos de arquiteturas **RISC**
  - Alpha, da DEC
  - MIPS
    - Arquitetura simples e didática que serve como base para alguns conceitos da disciplina
    - Desenvolvida sob coordenação de J. Hennessy
  - ARM: base para smartphones e sistemas embarcados atuais
  - RISC V: design de microprocessador de código aberto

CISC – Complex Instruction Set Computers

- Modelo de arquiteturas que...
  - Contém instruções complexas, as quais realizam uma sequência de operações em baixo nível
  - Exibem um conjunto de instruções grande

- Exemplo : add #1004, BX, #1000
  - Carregar dados da memória
  - Executar operações na ULA
  - Salvar resultado na memória

CISC – Complex Instruction Set Computers

- Exemplos de arquiteturas **CISC**
  - Grandes mainframes IBM
    - System/360: pioneiro em microprogramação
  - Intel
    - 8088: primeiros processadores Intel, base para alguns conceitos da disciplina
    - Pentium
  - Família VAX, da DEC (em praticamente em todas a universidades da época)

Comparação RISC x CISC

RISC	CISC
As instruções levam em média 1 ciclo de clock para serem executadas (80% delas)	Instruções complexas levando vários ciclos de clocks
Apenas load e store referenciam a memória; poucos modos de endereçamentos	Qualquer instrução acessa a memória
Instruções com formato fixo	Instruções com formato variado
Poucas instruções	Muitas instruções
Instruções executadas pelo hardware	Instruções interpretadas pelo microprograma
Complexidade está no compilador	Complexidade esta no microprograma
Altamente pipelined	Pouco pipelined

Comparação RISC x CISC

- Com todas essas vantagens do RISC, seria possível imaginar que esse modelo seria mais usado que o CISC
- No entanto, isso não ocorreu!
- Por que?

- Por diferentes motivos comerciais e tecnológicos

1. Muitas empresas com bilhões investidos em softwares para a linha Intel

2. A Intel conseguiu desenvolver um **modelo híbrido** (ideias básicas do RISC em suas máquinas com filosofia CISC), com o modelo **80486**
   - Núcleo RISC: executa as instruções mais simples (e portanto, mais frequentes) em um único ciclo do caminho de dados
   - Equanto instruções mais complexas são interpretadas (filosofia CISC)

Resultado: instruções mais comuns executam rapidamente e as menos comuns demoram um pouco mais

Arquiteturas Post-RISC

- A esse tipo de arquitetura híbrida é chamado de Post-RISC
  - Visam melhor desempenho que CISC
  - E compatibilidade com arquiteturas anteriores

- **Post-RISC** define um modelo híbrido que inclui, por exemplo
  - Emprego de princípios RISC
  - Emprego de técnicas originadas em arquiteturas RISC: ***pipelined***
  - Interpretação de instruções mais complexas (CISC)

“Today, ***99% of the more than 16 billion microprocessors produced annually are RISC processors****, and are found in nearly all smartphones, tablets, and the billions of embedded devices that comprise the Internet of Things (IoT).”*

https://awards.acm.org/binaries/content/assets/press-releases/2018/march/turing-award-2017.pdf

A New Golden Age for Computer Architecture

https://youtu.be/kFT54hO1X8M

Referências Bibliográficas

Tanenbaum, A S “**Organização Estruturada de Computadores”** – Prentice Hall do Brasil 5ª edição, 2006; **Subseção 2.1.3: pg 33.**

Stallings, William. “**Computer Organization and Architecture – Designing for Performance**”. 8° ed. Prentice Hall, Inc., New Jersey, 2010; **Subseção 13.8: pg 517.**