Computadores com um conjunto reduzido de instruções

Prof. Tiago Gonçalves Botelho

RISC

- Ênfase na otimização do uso de pipelines.
- Favorece o uso da técnica de atraso de instruções de desvio *(delayed branch)*.

1. Características da execução de instruções

- Custo de software é alto, pouco confiável;
- Linguagens de programação de alto nível mais poderosa e complexa;
- Problema: gap semântico – grande diferença entre operações disponíveis em linguagem de alto nível e as disponibilizadas pelo hardware dos computadores. Sintomas dessa distância incluem ineficiência na execução de programas, tamanho excessivo dos programas e grande complexidade dos algoritmos.

Objetivos de complexos conjuntos de instruções

- Facilitar a tarefa do desenvolvedor de compiladores;
- Melhorar a eficiência da execução de programas, com a implementação de sequências de operações em microcódigo.
- Oferecer suporte para LPs de alto nível cada vez mais complexas.

Operações e operandos

- Aspectos da computação que devem ser examinados com relação as características de instruções:
  - **Operações realizadas:** determinam as funções que devem ser realizadas pelo processador e sua interação com a memória.
  - **Operandos usados:** os tipos e a frequência de uso de cada um determinam como a memória deve ser organizada para armazená-los e os modos de endereçamento que devem ser disponíveis para acessá-los.
  - **Organização das instruções para execução:** determina o controle e a organização do pipeline.

1.1 Operações

- Determinação do número médio de instruções de máquina e referências à memória causadas por um tipo de comando em linguagem de alto nível.

	Ocorrência dinâmica		Avaliação das instruções de máquina		Avaliação de referências de memória	
	Pascal	C	Pascal	C	Pascal	C
ATRIBUIÇÃO	45%	38%	13%	13%	14%	15%
LOOP	5%	3%	42%	32%	33%	26%
CHAMADA	15%	12%	31%	33%	44%	45%
IF	29%	43%	11%	21%	7%	13%
GOTO	—	3%	—	—	—	—
OUTROS	6%	1%	3%	1%	2%	1%

(Fonte: Patterson e Sequin, 1982)

1.2 PROCEDIMENTOS

- De acordo estudo realizado por TANENBAUM, 1978 , os parâmetros são inferiores a 6 estão em 98% das chamadas, e 92% dessas chamadas são usados menos que 6 variáveis locais.

Porcentagem de execução de chamadas de procedimentos com	Compiladores, interpretadores e editores de texto	Pequenos programas não-numéricos
> 3 argumentos	0 - 7%	0 - 5%
> 5 argumentos	0 - 3%	0%
> 8 argumentos e variáveis escalares locais	1 - 20%	0 - 6%
> 12 argumentos e variáveis escalares locais	1 - 6%	0 - 3%

1.3 IMPLICAÇÕES

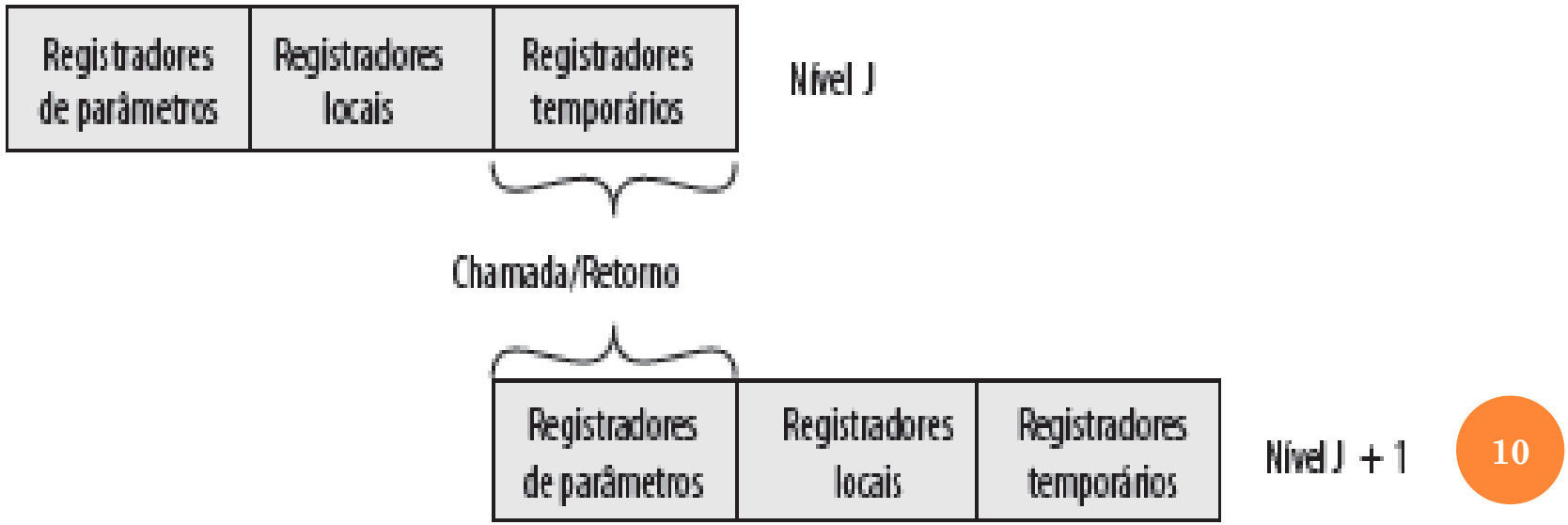
- O projeto de uma arquitetura com um conjunto de instruções próximo das linguagens de alto nível não era efetiva.
- Generalizando o trabalho dos pesquisadores, percebemos que as arquiteturas RISC se caracterizam por três elementos:
  - 1. Uso de certo número de registradores ou de técnicas de compilação que visam otimizar sua utilização.
  - 2. Atenção ao projeto de pipelines.
  - 3. Mais indicado utilizar um conjunto de instruções simplificado.

2. Utilização de registradores

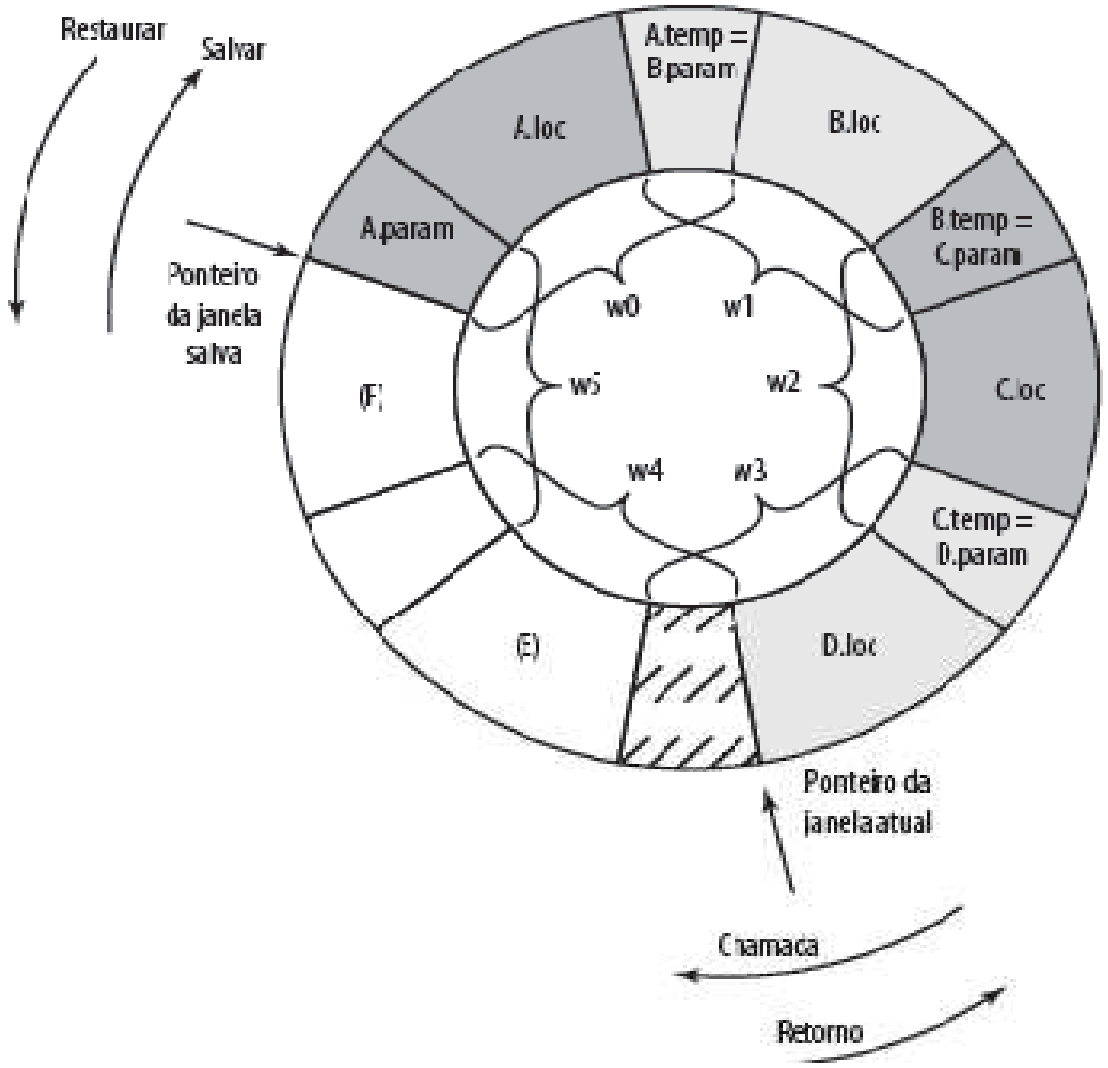
- Duas abordagens são possíveis, uma baseada em software e outra em hardware:
  - 1. Software: O compilador tenta alocar nos registradores as variáveis que serão mais usadas durante um período de tempo.
  - 2. Hardware: Consiste em usar um número maior de registradores.

2.1 JANELAS DE REGISTRADORES

- Na chamada de procedimento, o conteúdo dos registradores são salvos na memória, para que estes sejam reutilizados no procedimento, No retorno, esses valores da memória são restaurados.



Organização circular de janelas de registradores sobrepostas



2.2 VARIÁVEIS GLOBAIS

- Duas opções podem ser usadas:
  - a) Compilador alocar na memória todas as variáveis globais escritas em um programa em linguagem de alto nível.
  - b) Incorporar no processador um conjunto de registradores globais.

    Ex: 0 – 7 = Variáveis globais.

    8 – 31 = Registradores físicos na janela corrente.

2.3 GRANDE BANCO DE REGISTRADORES *VERSUS* MEMÓRIA CACHE

- Características da organização de computadores com um grande banco de registradores e com memória cache.

Grandes arquivos de registradores	Cache
Todas variáveis locais escalares	Variáveis locais recentemente usadas
Variáveis individuais	Blocos de memória
Variáveis globais assinaladas pelo compilador	Variáveis globais recentemente usadas
Salvar/restaurar baseados na profundidade de aninhamento do procedimento	Salvar/restaurar baseado em algoritmos de atualização da cache
Endereçamento de registrador	Endereço de memória

3. Otimização do uso de registradores baseada em compiladores

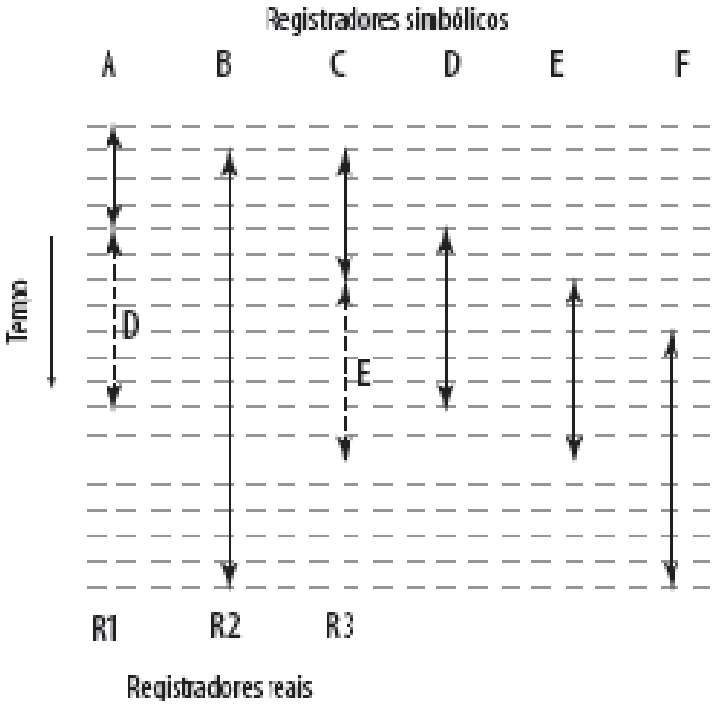
- O objetivo do compilador é manter em registradores em vez de na memória, os operandos requeridos no maior número de computações.
- Abordagem:
  - 1. Itens de dados são alocados em um registrador simbólico ou virtual;
  - 2. Compilador mapeia os registradores simbólicos em um número fixo de registradores reais;
  - 3. Registradores simbólicos cujo usos não se sobrepõem podem compartilhar o mesmo registrador real.

3. Otimização do uso de registradores baseada em compiladores

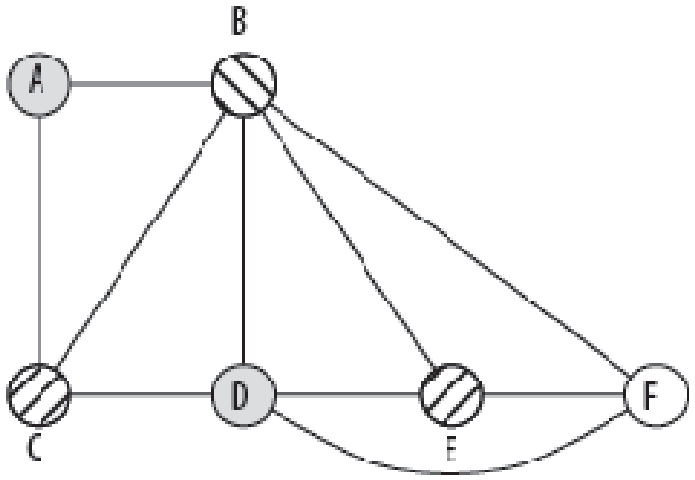
- A técnica mais utilizada em computadores RISC é conhecida como coloração de grafos.

  Exemplo: Considere um programa com seis registradores simbólicos, que deve ser compilado de modo que utilize três registradores reais.

3. Otimização do uso de registradores baseada em compiladores



(a) Sequência de tempo do uso ativo de registradores



(b) Grafo de interferência de registradores

Exercício

- Fazer o grafo de interferência para a sequência de uso de registradores, considerando quatro registradores reais:

  (Vide desenho do quadro)

4. RISC

- Motivações para uso de instruções complexas:
  - Simplificar o compilador;
  - Melhorar o desempenho.
- Outro argumento importante:
  - Produzir programas menores.

TAMANHO DO CÓDIGO RELATIVO AO RISC I

	Patterson, 1982	Katevenis, 1983	Health, 1984
RISC I	1,0	1,0	1,0
VAX-11/780	0,8	0,67	-
M68000	0,9	-	0,9
Z8002	1,2	-	1,12
PDP-11/70	0,9	0,71	-

4.1 Característica de arquiteturas RISC

- **Uma instrução por ciclo:** tempo requerido para buscar dois operandos em registradores, executar uma operação na ULA e armazenar o resultado em um registrador.
- **Operações registrador-registrador:** maioria das operações são Load/Store.

4.1 Característica de arquiteturas RISC

- Com relação ao desempenho, pode ser afirmado:
  - Compiladores otimizadores podem ser desenvolvidos.
- A maioria das instruções geradas por um compilador são simples.
- A técnica de pipeline de instruções pode ser aplicada mais efetivamente em máquinas RISC.
- São mais suscetíveis a interrupções.

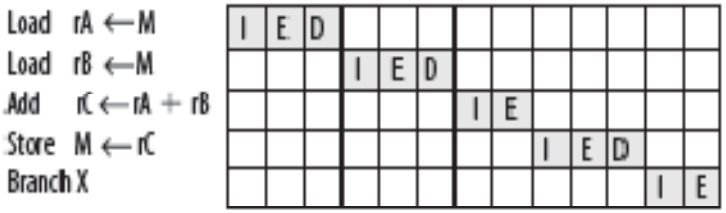
Esforço de projeto e leiaute de componentes para alguns microprocessadores (Fitzpatrick, 1981)

CPU	Transistores (milhares)	Projeto (pessoas/mês)	Leiaute de componentes (pessoas/mês)
RISC I	44	15	12
RISC II	41	18	12
M68000	68	100	70
Z8000	18	60	70
Intel iAPx-432	110	170	90

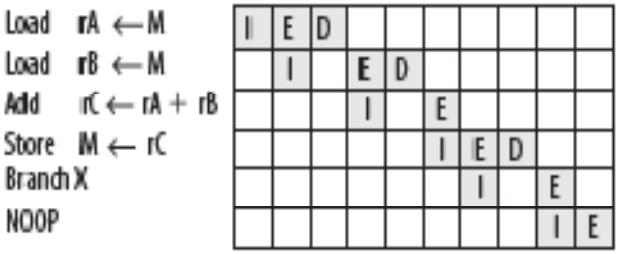
5. Pipeline de instruções RISC

- O ciclo de instrução apresenta duas fases para registrador-registrador
  - I: Busca de instrução.
  - E: Execução – Operação na ULA, com entrada e saída de registradores.
- A execução de carga e armazenamento requer três fases:
  - I: Busca de instrução.
  - E: Execução – Calcula o endereço de memória.
  - D: Memória – Operações registrador -> memória ou memória -> registrador

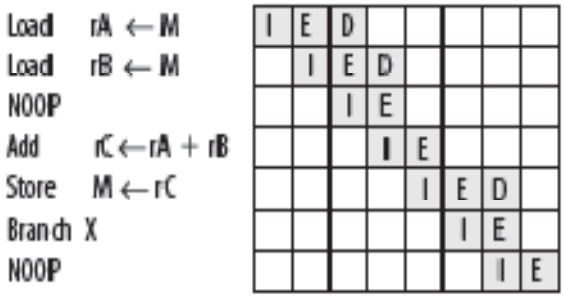
5. Pipeline de instruções RISC



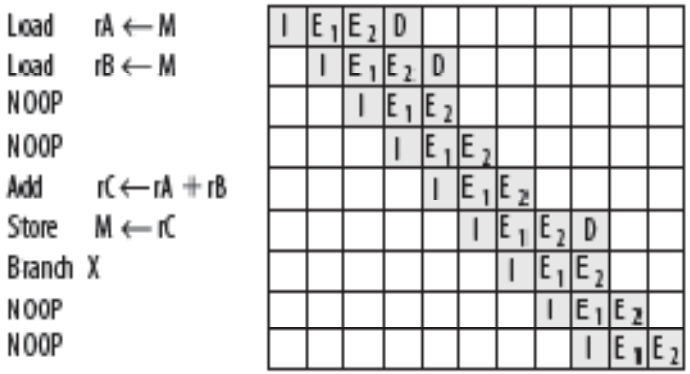
(a) Execução sequencial



(b) Tempo do pipeline de dois estágios



(c) Tempo do pipeline de três estágios



(d) Tempo do pipeline de quatro estágios

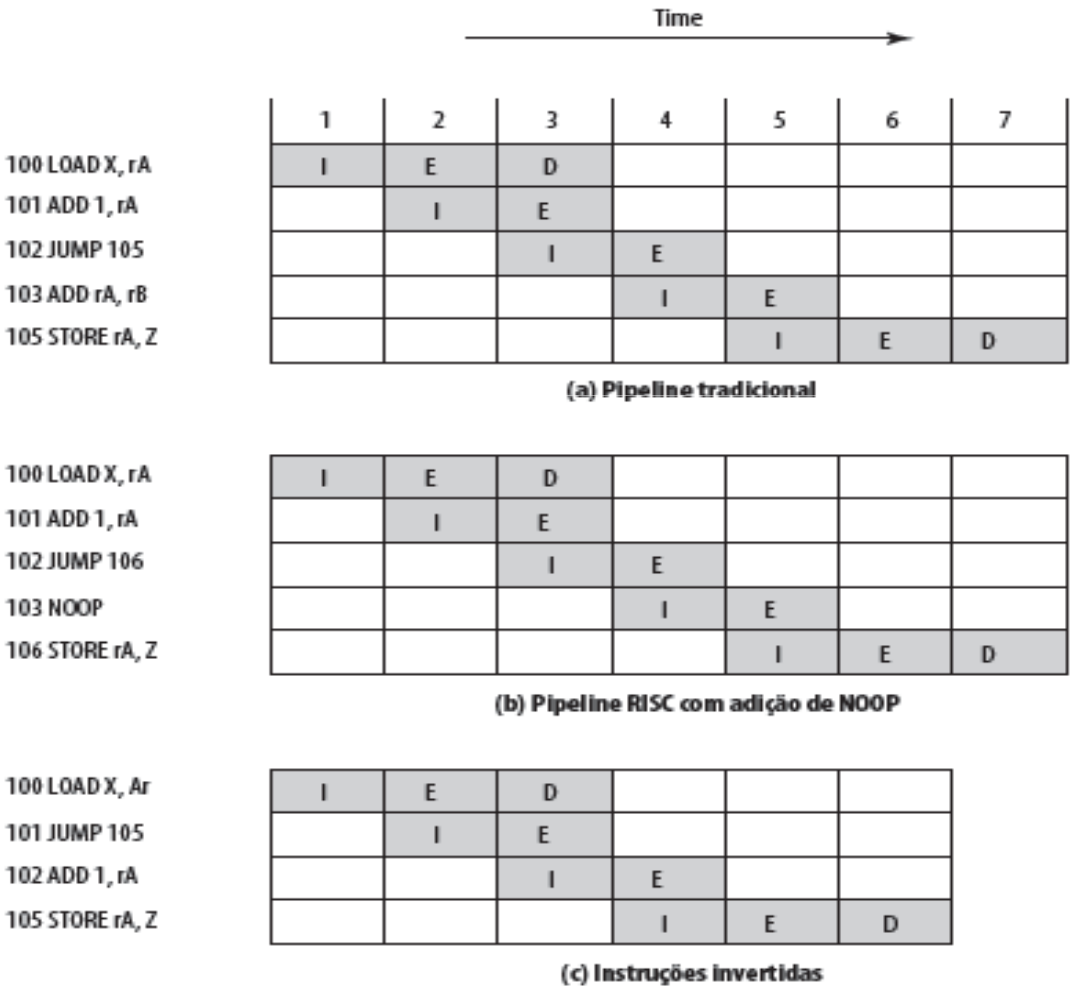
5.1 Otimizando o Pipeline

- **Desvio atrasado:** adota um desvio que não tem efeito antes de terminada a execução da instrução seguinte.

DESVIO NORMAL E ATRASADO

Endereço	Desvio normal	Desvio atrasado	Desvio atrasado otimizado
100	LOAD X, rA	LOAD X, rA	LOAD X, rA
101	ADD 1, rA	ADD 1, rA	JUMP 105
102	JUMP 105	JUMP 106	ADD 1, rA
103	ADD rA, rB	NOOP	ADD rA, rB
104	SUB rC, rB	ADD rA, rB	SUB rC, rB
105	STORE rA, Z	SUB rC, rB	STORE rA, Z
106		STORE rA, Z	

USO DO DESVIO ATRASADO



BIBLIOGRAFIA

- Stallings, W., Arquitetura e Organização de Computadores – 8ª Ed. – São Paulo: Editora Pearson, 2010.