Aula 4: Processador - Caminho de Dados e Controle - Parte IV

Professor(a): Virgínia Fernandes Mota
http://www.dcc.ufmg.br/∼virginiaferm

OCS (TEORIA) - SETOR DE INFORMÁTICA

Caminho de Dados e Controle

- Exceções
- Pipelining Avançado: Extraindo Mais Desempenho
- O pipelining do Pentium 4
- Falácias e Armadilhas
- Considerações Finais

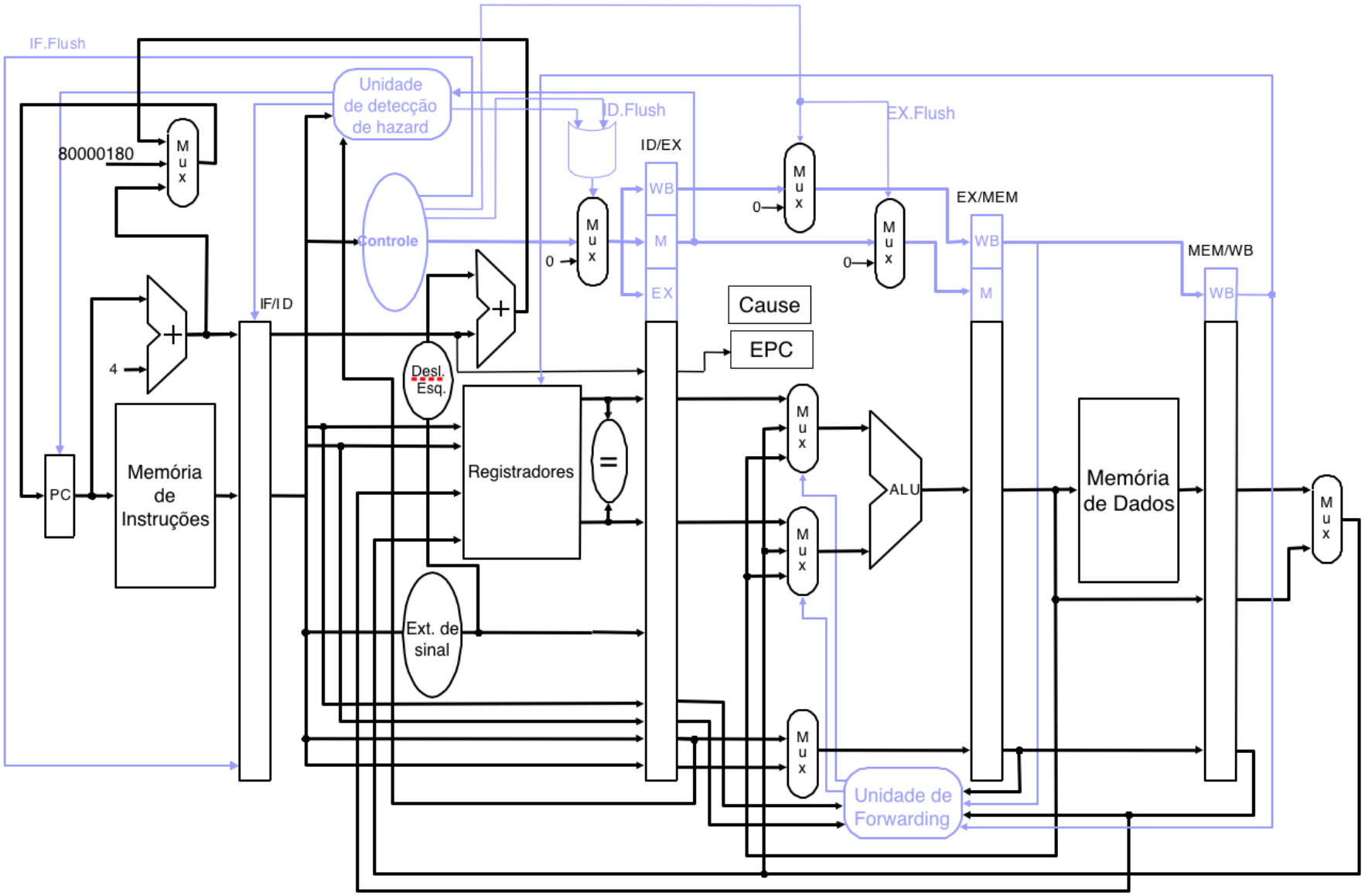
Exceções

- **Outra forma de hazard**
- Controle deve ser transferido para rotina de exceção imediatamente após instrução
  - Entrada adicional (80000180HEX) no multiplexador que fornece novo PC
- Necessário flush nas instruções que vêm após instrução que causou exceção
  - Estágio ID: fazemos OR com o sinal de stall da unidade de detecção de hazard
  - Estágio EX: novo sinal, EX.Flush

Exceções

- Suponha que add $1, $2, $1 cause overflow
- Se não pararmos execução no meio da instrução, programador não verá valor $1 que ajudou a causar exceção
  - Sinal EX.Flush pode ser usado para impedir escrita do resultado no estágio WB
- Etapa final: salvar EPC

Exceções


IF.Flush
Unidade de detecção de hazard
ID.Flush
EX.Flush
ID/EX
EX/MEM
MEM/WB
80000180
M u x
Controle
WB
M
EX
0
IF/ID
4
Desl. Esq.
Cause
EPC
PC
Memória de Instruções
Registradores
=
ALU
Memória de Dados
Ext. de sinal
Unidade de Forwarding

Exceções

- Cinco instruções ativas em qualquer ciclo de clock
  - Desafio: associar exceção à instrução correta
  - Estágio do pipeline ajuda: instrução desconhecida (ID), chamada ao SO (EX), ...
- Várias exceções podem ocorrer no mesmo ciclo de clock
  - Priorizar exceções
  - Instrução mais antiga interrompida
  - Flexibilidade para solicitações de E/S e defeitos de hardware: Não estão associados a instrução específica

Exceções

- Dificuldade de associar exceção correta à instrução correta levou projetistas a relaxarem esse requisito em casos não críticos
  - Interrupções (ou exceções) imprecisas
  - SO determina que instrução causou o problema
- Interrupções (ou exceções) precisas: associadas as instruções corretas

Especulação

- Técnica que permite que o compilador ou processador adivinhem as propriedades de uma instrução, para removê-la como uma dependência na execução de outras instruções
  - Ex: buscamos, emitimos e executamos instruções, como se previsão de desvio estivesse sempre correta
- Como pode estar errada, temos de verificar se a escolha foi certa
  - Caso tenha sido errada, temos de retroceder os efeitos

Especulação

- Pode ser feita por compilador ou hardware
- Recuperação diferente
  - Compilador
    - Adiciona instruções para verificar precisão da especulação
    - Rotina de reparo chamada quando especulação incorreta
  - Hardware
    - Resultados especulativos armazenados em buffer
    - Se resultados corretos, resultados escritos em registradores ou na memória
  - Especular pode introduzir exceções que antes não estavam presentes
    - Load com endereço inválido quando especulação incorreta

Pipelining Avançado

- Pipelining explora paralelismo em potencial entre instruções
  - Paralelismo em nível de instrução
- Dois métodos para aumentar quantidade de paralelismo
  - Aumentar profundidade do pipelining
    - Sobrepondo mais instruções
    - Ciclo de clock encurtado
  - Replicar componentes (despacho múltiplo)
    - Iniciar várias instruções em cada estágio do pipeline
    - Permite que CPI seja menor que 1 $\rightarrow$ IPC (instrução por ciclo)

Exemplo

- Processador com despacho múltiplo quádruplo de 6GHz
  - Velocidade de pico: 24 bilhões de instruções por segundo
  - CPI = 0,25 ou IPC = 4 (melhor dos casos)
  - Considerando pipeline de 5 estágios: 20 instruções em execução em determinado momento
  - Processadores mais potentes atuais: de 3 a 8 instruções despachadas por ciclo de clock

Despacho Múltiplo

- **Dois modos de implementar despacho múltiplo**
- Despacho estático: muitas decisões tomadas pelo compilador
- Despacho dinâmico: muitas decisões tomadas em tempo de execução pelo processador
- Duas responsabilidades principais e distintas
  - Empacotar instruções em slots de despacho
    - Quantas e quais instruções serão despachadas em determinado ciclo de clock?
  - Lidar com hazards de dados e de controle

Despacho Múltiplo Estático

- Compilador usado para ajudar no empacotamento de instruções
- Em alguns projetos compilador deve remover todos os hazards
  - Em outros, compilador deve evitar dependências dentro de um pacotes de despacho
  - Hardware detecta hazard de dados entre pacotes
- Pacote de despacho pode ser visto como uma grande instrução com várias operações
  - Mix de instruções que podem ser iniciadas em determinado ciclo restringidos
- Nome original da técnica: VLIW (very long instruction word?
- Nome na arquitetura IA-64: EPIC (explicitly parallel instruction computer)
  - Primeiras implementações: Itanium e Itanium-2

Exemplo

- Considere MIPS capaz de despachar duas instruções por ciclo
- Uma instrução que use a ALU e outra que acesse a memória
- Exige busca e decodificação de 64 bits de instrução
- Duas instruções retornadas da memória de instruções
- Instruções devem ser emparelhadas e alinhadas em um limite de 64 bits
- Instrução que usa a ALU aparece primeiro
- Se uma instrução do par não puder ser usada, deve ser substituída por NOP

Exemplo

- Necessárias portas extras nos registradores
- Somador separado necessário para calcular endereço de memória
  - ALU estará sendo usada pela outra instrução
- Resultado de leitura não estará disponível para as duas próximas instruções
  - Stalls
- Resultado da ALU não podem ser usados no load/store emparelhados

Escalonamento de Código Simples para Despacho Múltiplo

```
Loop: lw $t0, 0($s1) # t0: elemento do array
      addu $t0, $t0, $s2 # soma escalar
      sw $t0, 0($s1) # resultado do store
      addi $s1, $s1, -4 # decrementa pont
      bne $s1, $zero, Loop #desvia se != 0
```

- Três primeiras instruções possuem dependência de dados, bem como duas últimas

Escalonamento de Código Simples para Despacho Múltiplo

	ALU ou desvio	Memória	Ciclo
Loop:	nop	lw $t0, 0($s1)	1
	addi $s1, $s1, -4	nop	2
	addu $t0, $t0, $s2	nop	3
	bne $s1, $zero, Loop	sw $t0, 0($s1)	4

- Necessários quatro clocks para executar cinco instruções
- CPI = 4/5 = 0,8 ou IPC = 5/4 = 1,25
- Longe do melhor caso: CPI = 0,5 ou IPC = 2
- Técnica de compilador para conseguir mais desempenho: desdobramento de loop (loop unrolling)?
- Feitas várias cópias do corpo do loop

Desdobramento de Loop para Pipelines com Despacho Múltiplo

- Suponha índice múltiplo de 4 $\rightarrow$ 4 cópias do corpo do loop

	ALU ou desvio	Memória	Ciclo
Loop:	addi $s1, $s1, -16	lw $t0, 0($s1)	1
	nop	lw $t1, 12($s1)	2
	addu $t0, $t0, $s2	lw $t2, 8($s1)	3
	addu $t1, $t1, $s2	lw $t3, 4($s1)	4
	addu $t2, $t2, $s2	sw $t0, 16($s1)	5
	addu $t3, $t3, $s2	sw $t1, 12($s1)	6
	nop	sw $t2, 8($s1)	7
	bne $s1, $zero, Loop	sw $t3, 4($s1)	8

Desdobramento de Loop para Pipelines com Despacho Múltiplo

- Compilador introduziu registradores adicionais ($t0, $t1, $t2) para eliminar dependências não verdadeiras (antidependências ou dependência de nomes)
- *Renomeação de registradores*
- CPI: 8/14 = 0,57

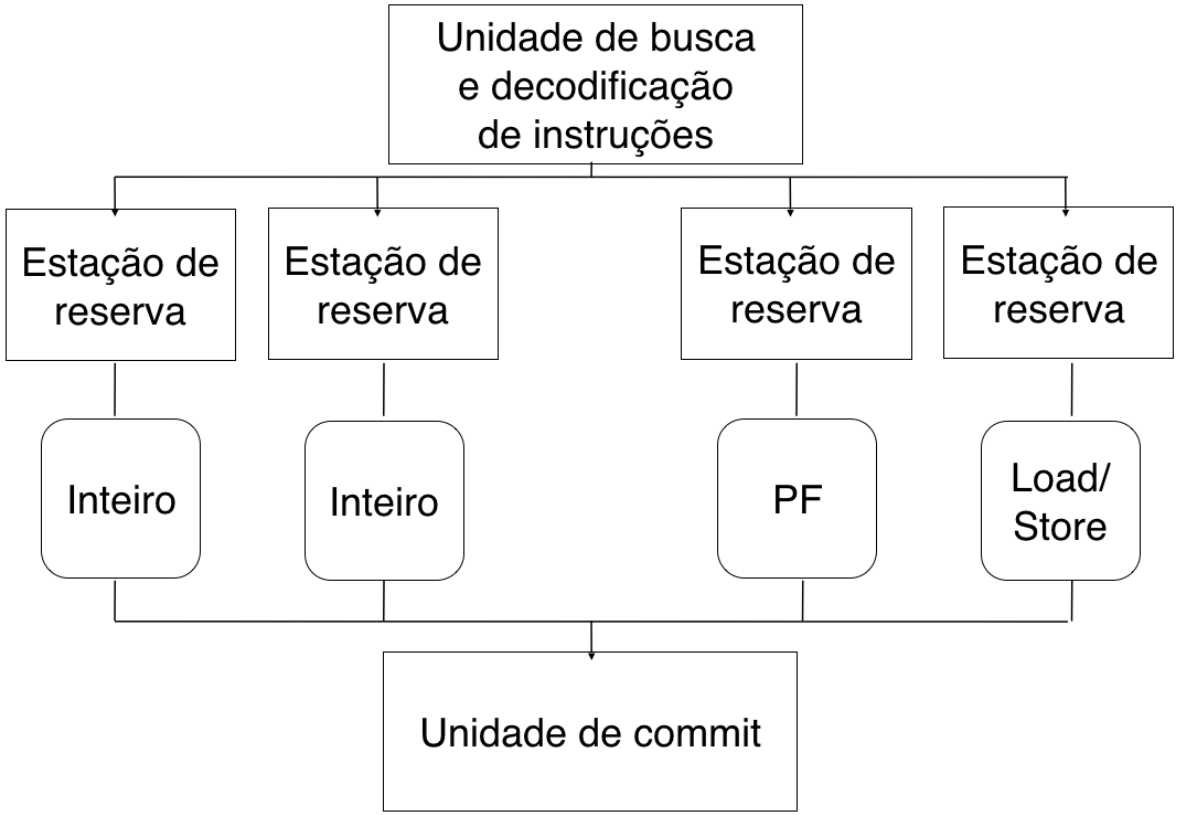
Processadores com Despacho Dinâmico

- Também conhecidos como processadores superescalares
- Instruções despachadas em ordem e processador decide se 0, 1 ou mais podem sem despachadas em determinado ciclo de clock
  - Escalonamento dinâmico em pipeline
  - Tenta evitar hazards e stalls

Processadores com Despacho Dinâmico

- **Pipeline dividido em três unidades principais**
- Busca e despacho de instruções
  - Busca, decodifica e envia instrução para a unidade funcional correspondente para execução
- Unidades funcionais (10 ou mais)
  - Possui buffers (estação de reserva), que mantêm operandos e operação
- Unidade de commit
  - Mantém resultados em buffer (buffer de reordenação)
  - Decide quando é seguro tornar resultados visíveis
  - Resultados no buffer de reordenação também usados para fornecer operandos

Processadores com Despacho Dinâmico



Processadores com Despacho Dinâmico

Como seria o pipeline de um processador atual?

Falácias e Armadilhas

- Falácia: Pipelining é fácil
- Falácia: As idéias de pipelining podem ser implementadas independente de tecnologia
- Armadilha: A falha em considerar o projeto do conjunto de instruções pode afetar o pipeline de forma adversa
  - Tamanhos de instruções e tempos de execução muito variáveis podem causar desequilíbrio entre estágios
  - Modos de endereçamento sofisticados podem levar a hazards e complicar o controle do pipeline

Próxima aula

Exercícios e Prova