INSTITUTO FEDERAL DO RIO GRANDE DO NORTE
DIRETORIA ACADÊMICA DE GESTÃO E TECNOLOGIA DA INFORMAÇÃO
CURSO SUPERIOR DE TECNOLOGIA EM REDES DE COMPUTADORES
DISCIPLINA: ORGANIZAÇÃO DE COMPUTADORES

Construção de uma via de dados, abordagem monociclo, multiciclo e pipeline

Discente: Arley Willer Neves da Silva

Construção de uma via de dados

Ponto fixo – Overflow

Overflow significa inundação e, quando aplicada como termo da informática, significa um transbordamento que causa uma inundação.

Construção de uma via de dados

Ponto fixo – Solução MIPS

O MIPS é o nome de uma arquitetura de processadores baseados no uso de registradores. As suas instruções tem à disposição um conjunto de 32 registradores para realizar as operações. Entretanto, alguns destes registradores não podem ser usados por programadores, pois são usados pela própria máquina para armazenar informações úteis.

Processadores MIPS são do tipo RISC (Reduced Instruction Set Computer - ou seja, Computadores com Conjunto de Instruções Reduzidas). Isso significa que existe um conjunto bastante pequeno de instruções que o processador sabe fazer. Combinando este pequeno número, podemos criar todas as demais operações.

Pela sua elegância e simplicidade, processadores MIPS são bastante usados em cursos de arquiteturas de muitas universidades. Ele é considerado um processador bastante didático.

Projetos MIPS são atualmente bastante usados em muitos sistemas embarcados como dispositivos Windows CE, roteadores Cisco e video-games como o Nintendo 64, Playstation, Playstation 2 e Playstation Portable.

Construção de uma via de dados

Ponto fixo – Causam e não causam exceções

Algumas instruções aritméticas geram exceções quando ocorre overflow.

Em relação à solução MIPS, causam exceções no overflow a Adição, a Adição imediata e a subtração. Em contrapartida, não causam exceções a adição sem signal, a adição imediata sem signal e a subtração sem signal.

```
add     $t0, $s1, $s2    # $t0 ← $s1 + $s2
addi    $t0, $s1, 123    # $t0 ← $s1 + 123
sub     $t0, $s1, $s2    # $t0 ← $s1 - $s2
addu    $t0, $s1, $s2    # $t0 ← $s1 + $s2
addiu   $t0, $s1, 123    # $t0 ← $s1 + 123
```

Construção de uma via de dados

Ponto fixo – Operações com ponto fixo

Multiplicação é mais complexa que a divisão, pois se um dos números é negativo, a multiplicação direta não resolve.

Contudo, pode-se dizer que a divisão é ainda mais complexa justamente porque trabalhar com números negativos dá trabalho.

Temos somente os algarismos 0 e 1 para representar todos os números inteiros.

> Inteiros positivos são transformados em binário:
> 41 = 0010 1001
> 1 = 0000 0001
> 64 = 0100 0000

Construção de uma via de dados

Ponto fixo – Par de registradores

Ocorre quando o produto de uma multiplicação de 64 bits é colocado em um par de registradores de 32 bits.

Hi – armazena a parte mais significativa

Lo – armazena a parte menos significativa

Construção de uma via de dados

Ponto flutuante

A representação em aritmética de ponto flutuante é muito utilizada na computação digital. Um exemplo é a caso das calculadoras científicas.

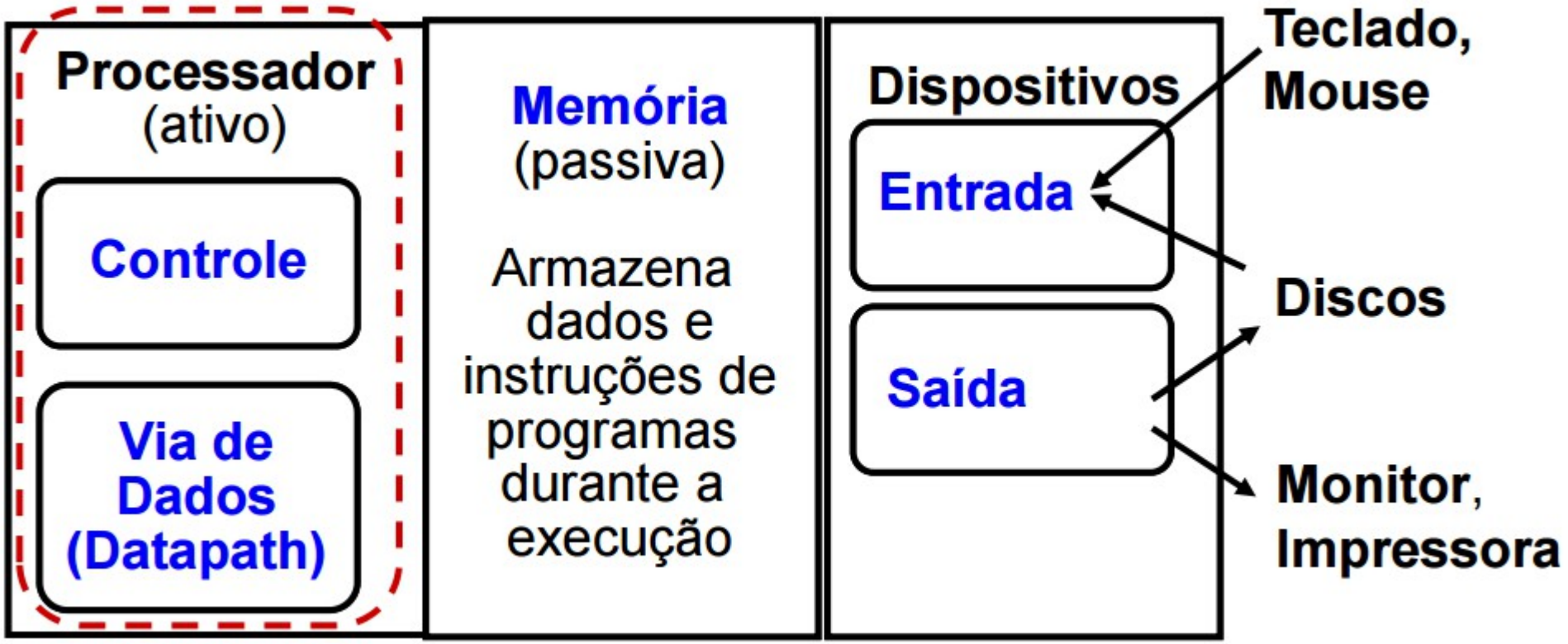
Exemplo: 2,597 –03.

Este número representa: $3\ 2{,}597\ 10^{-} \times$.

A principal vantagem da representação em ponto flutuante é que ela pode representar uma grande faixa de números se comparada a representação de ponto fixo.

Construção de uma via de dados

Componentes clássicos das vias de dados e controle



Construção de uma via de dados

Implementação de instruções no processador

Arquitetura tipo Ciclo Único

- Cada instrução é executada em um 1 ciclo de clock
- Ciclo de clock deve ser longo o suficiente para executar a instrução mais longa
- Desvantagem: velocidade global limitada à velocidade da instrução mais lenta

Construção de uma via de dados

Implementação de instruções no processador

Arquitetura tipo Multi-ciclo

- Quebra o ciclo de execução em vários passos
- Executa cada passo em um ciclo de clock
- Vantagem: cada instrução usa apenas o número de ciclos que ela necessita

Construção de uma via de dados

Linhas de controle

As unidades de estado visíveis ao programador (PC, memória, banco de registradores) e IR precisarão de sinais de controle de escrita

- Memória precisará de um sinal de leitura

Construção de uma via de dados

Linhas de controle - Estágio 1 Instruction Fetch (IF)

Usa o registrador Program Counter (PC) para ler
a instrução da memória e armazená-la no
Registrador de Instrução (IR)

- Incrementa PC em 4 e coloca o resultado de volta em PC
- IR = *PC; // IR = valor que esta
// armazenado no endereço
// que o PC indica
- PC = PC + 4;

Construção de uma via de dados

Linhas de controle - Estágio 2 Instruction Decode (ID)

Lê os campos correspondentes aos registradores rs e rt, independentemente do tipo de instrução

- Os valores lidos são colocados nos registradores temporários A e B
- Computa o endereço de desvio, considerando que a instrução possa ser um branch

- A = Reg[IR[25-21]];
- B = Reg[IR[20-16]];
- ALUOut = PC + (sign-extend(IR[15-0]) << 2);

Construção de uma via de dados

Linhas de controle - Estágio 3 Execution (EX)

A ALU executará uma das quatro funções abaixo, dependendo do tipo da instrução:

- Referência à memória (para load/store):
  - ALUOut = A + sign-extend(IR[15-0]);
- Tipo R
  - ALUOut = A op B;
- Branch:
  - if (A==B) PC = ALUOut;
- Jump:
  - PC = PC[31-28] concat (IR(25-0) << 2)

Construção de uma via de dados

Linhas de controle - Estágio 4 Acesso à memória (MEM)

Instruções de acesso à memória:

- Load: Dados são lidos na memória e escritos no registrador de dados da memória (MDR)
  - MDR = Memory[ALUOut];
- Store: Dados são escritos na memória
  - Memory[ALUOut] = B;
  - Instruções do tipo R:
- A saída da ALU é escrita no registrador de destino
- Reg[IR[15-11]] = ALUOut;

Construção de uma via de dados

Linhas de controle - Estágio 5 Write Back (WB)

- Depende do tipo de instrução
- Load: Escrita no registrador:
- Reg[IR[20-16]]= MDR;
- Estágio final

Construção de uma via de dados

Multiciclo

Instruções MIPS levam de 3 a 5 etapas de execução
- As duas primeiras etapas (IF e ID) são comuns a todas instruções
- As células vazias na tabela indicam que a classe de instruções representada pela coluna leva menos ciclos
- No hardware multi-ciclo, esses ciclos não ficam ociosos, pois uma instrução é executada logo após a anterior

Construção de uma via de dados

Implementando controle

Os valores dos sinais de controle são dependentes de:

- Quais funções estão sendo executadas
- Quais estágios estão ocorrendo
- Baseado nos sinais de controle especificados, duas abordagens podem ser usadas para gerar os sinais de controle:

1) Especificar um máquina de estados finitos graficamente
2) Usar microprogramação

Construção de uma via de dados

Máquina de Estados Finitos (FSM)

- Finite State Machine (FSM)
- Um conjunto de estados
- Uma função que computa o próximo estado
  - Determinada pelo estado atual e pela entrada
- Uma função de saída
  - Determinada pelo estado corrente e possivelmente pela entrada
- No nosso caso, usaremos uma máquina de Moore
  - A função de saída é usada somente pelo estado Corrente

Construção de uma via de dados

Pipeline

- Pipeline é uma técnica de implementação de processadores que permite a sobreposição temporal de diversas fases de execução de instruções
- Em outras palavras, o hardware processa mais de uma instrução de cada vez, sem esperar que uma instrução termine antes de começar a outra

Construção de uma via de dados

Pipeline - Características

- Pipeline não melhora a latência de uma única tarefa, mas melhora o throughput de todo trabalho
- Tempo de execução de uma tarefa é o mesmo, com ou sem pipelining
- Ganho começa a existir a partir da segunda tarefa
- Taxa de inserção de tarefas é limitada pela tarefa mais lenta
- Pipeline explora o paralelismo entre as instruções em um fluxo de instruções sequenciais
- É uma técnica invisível ao programador, ao contrário de técnicas de multiprocessadores

Construção de uma via de dados

Ganho de desempenho com Pipelining

- Contudo:
  - Estágios não são perfeitamente balanceados para durarem a mesma quantidade de tempo T.
  - Cada estágio possui um overhead de tempo em decorrência do pipelining.

- Portanto:
  - O tempo de execução de cada estágio no processador com pipeline será maior que no computador sem pipeline.
  - O ganho será um pouco menor que o número K de estágios.

Construção de uma via de dados

Processador RISC de cinco estágios

Cinco estágios de execução de cada instrução:

- IF Busca (fetch) da instrução
- ID Decodificação da instrução
- EX Execução ou cálculo do endereço efetivo
- MEM Acesso à memória ou desvio
- WB Atualização dos registradores (write back)

- Cada fase de uma instrução é processada por um estágio, em um ciclo de clock
- Uma nova instrução é iniciada a cada ciclo

Construção de uma via de dados

Processador RISC de cinco estágios

- Durante um mesmo ciclo de clock, temos de garantir que não ocorram duas operações com os mesmos componentes do caminho de dados:
- Utiliza-se uma memória cache de dados (DM) separada da memória cache de instruções (IM)
- O banco de registradores é utilizado duas vezes:
  - Estágio ID para leitura, na 2ª. metade do ciclo de clock
  - Estágio WB para escrita, na 1ª. metade do ciclo de clock

Construção de uma via de dados

Processador RISC de cinco estágios

- PC é incrementado e armazenado a cada ciclo, durante a fase IF
- Como tratar desvios?
- Registradores são inseridos entre cada estágio do pipeline para:
  - Garantir que operações em diferentes estágios do pipeline não interfiram uma nas outras
  - Carregar resultados intermediários entre estágios não adjacentes
- Tais registradores recebem nomes especiais, que é a junção dos nomes dos estágios que interfaceiam:
  - IF/ID
  - ID/EX
  - EX/MEM
  - MEM/WB

Construção de uma via de dados

Processador RISC de cinco estágios

- PC é incrementado e armazenado a cada ciclo, durante a fase IF
- Como tratar desvios?
- Registradores são inseridos entre cada estágio do pipeline para:
  - Garantir que operações em diferentes estágios do pipeline não interfiram uma nas outras
  - Carregar resultados intermediários entre estágios não adjacentes
- Tais registradores recebem nomes especiais, que é a junção dos nomes dos estágios que interfaceiam:
  - IF/ID
  - ID/EX
  - EX/MEM
  - MEM/WB

Projeto do conjunto de instruções

Processador RISC de cinco estágios

- PC é incrementado e armazenado a cada ciclo, durante a fase IF
- Como tratar desvios?
- Registradores são inseridos entre cada estágio do pipeline para:
  - Garantir que operações em diferentes estágios do pipeline não interfiram uma nas outras
  - Carregar resultados intermediários entre estágios não adjacentes
- Tais registradores recebem nomes especiais, que é a junção dos nomes dos estágios que interfaceiam:
  - IF/ID
  - ID/EX
  - EX/MEM
  - MEM/WB

Webgrafia

http://www.cpdee.ufmg.br/~frank/lectures/SPP/SPP-aula05-Aritmetica_Computacional .pdf

https://pt.wikibooks.org/wiki/Introdu%C3%A7%C3%A3o_%C3%A0_Arquitetura_de_Computadores/O_que_%C3%A9_o_MIPS%3F

http://www.guj.com.br/java/261493-ponto-flutuante-e-ponto-fixo

https://pt.wikipedia.org/wiki/Ponto_flutuante

http://www.cpdee.ufmg.br/~frank/lectures/SPP/SPP-aula07-Via_de_Dados_e_Controle_2.pdf

Obrigado!