Universidade de Itaúna - Faculdade de Engenharia - Ciência da Computação
Arquitetura e Organização de Computadores II - Profª. Adriana

Atividade – Revisão de conteúdo

1. Marque F para as alternativas Falsas e V para as alternativas Verdadeiras:

(v) Thread é um fluxo de execução com suas próprias instruções e dados. Parte de um programa paralelo, é independente e possui estado próprio (instruções, dados, PC, estados dos registradores etc.).

(v) Processamento Superescalar permite a execução de mais de uma instrução simultaneamente (no mesmo ciclo de clock) obtido através da implementação de múltiplas unidades funcionais que são as unidades onde as instruções são executadas.

(v) Superpipeline consiste em se colocar um grande número de estágios, no caso, sendo mais que 6 estágios e tem como vantagens maior número de instruções sendo processadas ao mesmo tempo e maior freqüência de Clock.

(v) O paralelismo em nível de instrução (ILP - Instruction Level Parallelism) é uma família de técnicas de desenvolvimento de processadores e compiladores que aumentam o desempenho dos computadores, fazendo com que operações como leitura e escrita de memória, adição de inteiros, e multiplicações de ponto flutuante sejam executadas em paralelo.

(v) O paralelismo em nível de instrução (pipeline) aumenta o desempenho dos computadores, fazendo com que operações como leitura e escrita de memória, adição de inteiros, e multiplicações de ponto flutuante sejam executadas em paralelo.

(v) O paralelismo de instrução é baseado na execução simultânea de mais de uma instrução pela CPU, sendo cada instrução em um estágio diferente do ciclo de instruções.

(v) No Paralelismo por Software é possível executar um programa com capacidade de dividir uma tarefa em pequenas partes e executá-las em paralelo. Como por exemplo diversos cálculos matemáticos ou até mesmo atender a requisição de vários clientes, sem que seja necessário formar uma fila de processamento.

(v) A técnica denominada renomeação de registradores implica em registradores que são alocados dinamicamente pelo hardware do processador e são associados com valores usados pelas instruções em vários pontos do tempo.

2. O Paralelismo a nível de Processador é uma técnica que busca o ganho de desempenho associando duas ou mais CPUs organizados para executar uma determinada tarefa. São usadas as seguintes técnicas exceto:
   a) Computadores matriciais (Matriz de processadores dedicados)
   b) Multiprocessadores (Conjunto de processadores independentes)
   c) Multicomputadores
   d) Sistema NUMA (Acesso não uniforme à memória)

Universidade de Itaúna - Faculdade de Engenharia - Ciência da Computação
Arquitetura e Organização de Computadores II - Profª. Adriana

Atividade – Revisão de conteúdo

3. Em relação às arquiteturas paralelas, considere:

I. vários processos podem se comunicar apenas lendo e escrevendo na memória;
II. todas as CPUs veem a mesma imagem de memória e apenas um mapa de páginas e uma tabela de processos;
III. primitivas de software send e receive costumam ser utilizadas na comunicação entre processos;
IV. subdividir os dados corretamente e posicioná-los em localizações ótimas não é tão importante, visto que o posicionamento não afeta a correção ou a programabilidade.

Os itens I a IV referem-se a:

a) multicomputador, multiprocessador, multiprocessador e multiprocessador, respectivamente.
b) multiprocessador, multiprocessador, multicomputador e multiprocessador, respectivamente.
c) multicomputador.
d) multiprocessador.
e) multiprocessador, multiprocessador, multiprocessador e multicomputador, respectivamente.

4. Esclareça a diferença entre:

a. Processamento superescalar e processamento superpipeline
b. Paralelismo a nível de instrução (Pipeline) e Paralelismo a nível de Processador

a) **Superescalar:** permitem a execução de mais de uma instrução simultaneamente (no mesmo ciclo de clock) obtido atravez da implemetaçao de múltiplas unidades funcionais que são as unidades onde as instruções são executadas.

Superpipeline: Consiste em se colocar um grande número de estágios, no caso, sendo mais que 6 estágios.

Vantagens: Maior número de instruções sendo processadas ao mesmo tempo e maior freqüência de Clock.

Desvantagens: Aumenta a complexidade, dependências e desvios.

b) **O paralelismo em nível de instrução (pipeline)** aumenta o desempenho dos computadores, fazendo com que operações como leitura e escrita de memória, adição de inteiros, e multiplicações de ponto flutuante sejam executadas em paralelo.

O paralelismo de instrução é baseado na execução simultânea de mais de uma instrução pela CPU, sendo cada instrução em um estágio diferente do ciclo de instruções.

Paralelismo a nível de Processador: Esse tipo de paralelismo é uma técnica que busca o ganho de desempenho associando duas ou mais CPUs organizados para executar uma determinada tarefa. São usadas as seguintes técnicas:

•Computadores matriciais (Matriz de processadores dedicados)
•Multiprocessadores (Conjunto de processadores independentes)
•Multicomputadores

Universidade de Itaúna - Faculdade de Engenharia - Ciência da Computação
Arquitetura e Organização de Computadores II - Profª. Adriana

Atividade – Revisão de conteúdo

5. É o tipo de arquitetura paralela que consiste em máquinas formadas por milhares de CPUs padronizadas que apresentam bom desempenho pela quantidade de processadores e que, geralmente, utilizam uma rede de interconexão proprietária de desempenho muito alto. Trata-se de:

   a) Symmetric MultiProcessor - SMP
   b) Massively Parallel Processor - MPP
   c) Parallel Vector Processor - PVP
   d) Distributed Shared Memory - DSM
   e) Cluster of Workstation - COW

6. O paralelismo em nível de instruções é caracterizado pela execução em paralelo por sobreposição de instruções de uma sequência que são independentes. Explique o conceito apresentado no exemplo abaixo:

```
Load R1 ← R2        Add R3 ← R3, "1"
Add R3 ← R3, "1"    Add R4 ← R3, R2
Add R4 ← R4, R2     Store [R4]  ← R0
```

   As instruções apresentadas no lado esquerdo são independentes, portanto, em teoria poderiam ser executadas em paralelo, já as do lado direito não são dependentes e, portanto, não podem ser executadas em paralelo, visto que a segunda instrução depende do resultado da primeira e a terceira instrução depende do resultado da segunda. A frequência da dependência de dados verdadeira e das dependências procedurais no código definem o grau de paralelismo em nível de instruções, esses fatores por sua vez são dependentes da arquitetura do conjunto de instruções. O paralelismo em nível de instruções é também definido pela latência de operação, que representa o tempo necessário para que o resultado de uma operação esteja disponível como operando em uma instrução subsequente, ou seja, ela determina quanto atraso será causado por uma dependência de dados ou procedural.

7. Para otimizar a utilização de vários elementos do pipeline, o processador precisa alterar uma ou mais dessas ordens respeitando a ordem a ser encontrada em uma execução estritamente sequencial.
   Com isso podemos classificar as políticas de emissão de instruções Superescalares em categorias, quais são elas?

   - **Emissão em-ordem com conclusão em-ordem**. - É a política mais simples de emissão de instruções, é a ordem exata que seria alcançada pela execução sequencial (emissão em-ordem) e pela escrita de resultados na mesma ordem (conclusão em-ordem). Por conta de sua simplicidade não é utilizada nem mesmo pelos pipelines escalares.
     Duas instruções são obtidas ao mesmo tempo e passadas para a unidade de decodificação. Como as instruções são obtidas em pares, as duas próximas instruções precisam esperar até que o par de estágios de decodificação do pipeline esteja limpo. Para garantir a conclusão em-ordem, quando ocorre um conflito por uma unidade funcional ou quando uma unidade funcional requer mais do que um ciclo para gerar um resultado, a emissão da instrução para temporariamente.
   - **Emissão em-ordem com conclusão fora-de-ordem**. - A conclusão fora-de-ordem é usada em processadores RISC escalar para melhorar o desempenho das instruções que requerem múltiplos ciclos.

Na conclusão fora-de-ordem, qualquer número de instruções pode estar no estágio de execução em qualquer tempo até o nível máximo do paralelismo da máquina através de todas as unidades funcionais. A emissão de instruções é parada por um conflito de recurso, uma dependência de dados ou uma dependência procedural. Além dessas uma outra dependência aparece que é a chamada dependência de saída ou dependência de escrita após escrita (WAW, write after write)

- **Emissão fora-de-ordem com conclusão fora-de-ordem** - Para permitir emissão fora-de-ordem, é necessário separar os estágios de decodificação e execução do pipeline, isso é feito com um buffer conhecido como janela de instruções. Nessa configuração depois que o processador termina de decodificar uma instrução, ela é colocada na janela de instruções. Enquanto este buffer não estiver cheio, o processador pode continuar a obter e decodificar novas instruções. Quando uma unidade funcional se torna disponível no estágio de execução, uma instrução da janela de instruções pode ser emitida para o estágio de execução. Com isso o processador tem a capacidade de olhar para frente, o que permite que ele identifique instruções independentes que podem ser trazidas para o estágio de execução. As instruções são emitidas a partir da janela de instruções sem se preocupar com a ordem do programa original, tendo como única restrição que o programa se comporte corretamente. A política de emissão fora -de -ordem e conclusão fora -de -ordem é sujeita às mesmas restrições apresentadas nas outras políticas, porém aqui uma nova restrição aparece, denominada antidependência ou dependência ler após escrever (RAW, read after write)

8. Qual é a função de uma janela de instruções?

   A Janela de Instruções armazena o resultado da decodificação das instruções e isola o estágio de busca e decodificação de instruções dos estágios de execução propriamente dita das instruções. Pode ser implementada de forma Centralizada (Janela Centralizada ou “Central Window”) ou distribuída pelas unidades funcionais (Estações de Reserva ou “Reservation Stations”).

9. Uma maneira de enfrentar tipos de conflitos de armazenamento é baseada em uma solução tradicional de conflitos de recursos, a duplicação de recursos. Neste contexto, esta técnica é conhecida como renomeação de registradores. Defina o que é a renomeação de registradores e fale sobre sua função.

   Uma maneira de enfrentar esses tipos de conflitos de armazenamento é baseada em uma solução tradicional de conflitos de recursos: duplicação de recursos. Neste contexto, a técnica é conhecida como renomeação de registradores. Basicamente, registradores são alocados dinamicamente pelo hardware do processador e são associados com valores usados pelas instruções em vários pontos do tempo. Quando um novo valor de registrador é criado (por exemplo, quando uma instrução que tem um registrador como um operando de destino é executado), um novo registrador é alocado para esse valor. As instruções subsequentes que acessam esse valor como operando de origem nesse registrador têm que passar pelo processo de renomeação: as referências de registradores nessas instruções precisam ser revisadas para que se refiram ao registrador que contém o valor necessário. Assim, a mesma referência do registrador original em várias instruções diferentes pode se referir a registradores reais diferentes, se valores diferentes são pretendidos.

Atividade – Revisão de conteúdo

10. São os elementos-chaves da organização de um processador superescalar, exceto:
    a) As estratégias de busca de instrução que obtêm simultaneamente várias instruções, frequentemente prevendo os resultados das instruções de desvios condicionais. Estas funções requerem o uso de múltiplos estágios de busca e decodifi cação e lógica de previsão de desvios.
    b) Lógica para determinar dependências verdadeiras envolvendo valores de registradores e mecanismos para transferir esses valores para onde eles forem necessários durante a execução.
    c) Mecanismo para iniciar, ou emitir, múltiplas instruções em paralelo.
    d) Recursos para execução paralela de múltiplas instruções, incluindo múltiplas unidades funcionais de pipeline e hierarquias de memória capazes de atender simultaneamente várias referências de memória.
    e) Mecanismos para concluir o estado do processo na ordem correta.
    f) Do ponto de vista do programador, esse tipo de aplicação pode ser projetado através de dois modelos principais, memória compartilhada e troca de mensagens. *(resposta correta: esta falsa pois esta é a definição de processamento paralelo)*

11. Qual é a diferença entre microestrutura horizontal e vertical?

Em uma microinstrução horizontal cada bit de um campo de controle é ligado a uma linha de controle. Em uma microinstrução vertical, é usado um código para cada ação a ser efetuada e o decodificador traduz esse código em sinais de controle individuais. As vantagens de microinstruções verticais é que elas são mais compactas que microinstruções horizontais, ao custo de uma pequena lógica e tempo de atraso adicional.

12. A unidade de controle desempenha duas tarefas básicas, quais são elas?

- ✓ sequenciamento: a unidade de controle faz o processador executar por uma série de micro-operações na sequência correta, com base no programa que está sendo executado.
- ✓ execução: a unidade de controle faz cada micro-operação ser executada.

13. A abordagem Superescalar depende da exploração do paralelismo em nível de instrução. Para tanto, tem que lidar com algumas limitações.

    Dependência de dados verdadeira / Dependência procedural / Conflitos de recursos / Dependência de saída / Antidependência

    Complete os espaços abaixo com a definição correspondente:

    a) **Dependência procedural**: Essa dependência ocorre com a apresença de desvios em uma sequência de instruções, esse fator complica a operação do pipeline. As instruções que vêm depois de um desvio possuem uma dependência procedural com o desvio e não podem ser executadas até que o desvio seja executado.
    b) **Dependência de saída:** Quando duas ou mais instruções estão escrevendo em um mesmo local
    c) **Conflitos de recursos**: É uma competição de duas ou mais instruções pelo mesmo recurso e ao mesmo tempo. São exemplos: memória, cache, barramentos, entradas para banco de registradores e unidades funcionais.

d) **Antidependência:** É semelhante a uma dependência de dados verdadeira, porém invertida. Ao invés de a primeira instrução produzir um valor usado pela segunda instrução, a segunda destrói um valor usado pela primeira. Quando uma instrução usa um local e um operando enquanto a seguinte está escrevendo no mesmo local.

e) **Dependência de dados verdadeira:** Também denominada de WAR do inglês "Write After Read" ou dependência de fluxo, essa dependência em uma máquina superescalar ocorre quando uma instrução pode ser obtida, decodificada mas não pode ser executada até que a instrução predecessora seja executada, porque pois ela necessita de dados gerados por ela. Sem dependências, duas instruções podem ser obtidas e executadas em paralelo. Se houver uma dependência de dados entre a primeira e a segunda instrução, então a segunda é atrasada por tantos ciclos de clock quantos forem necessários para remover a dependência.

14. Analise as afirmativas a seguir sobre a técnica de paralelismo:

I. Em um pipeline, quando uma instrução depende do resultado da instrução anterior que ainda não foi concluída, dizemos que temos um exemplo de hazard de dados.
II. Uma maneira de evitar um hazard estrutural é a duplicação de um recurso para permitir todas as combinações de instruções que queremos executar em um mesmo ciclo de clock.
III. O adiantamento (Bypassing ou Forwarding) de dados é uma técnica para resolver alguns tipos de hazard de dados que consiste em utilizar o elemento de dado a partir dos buffers internos em vez de esperar que chegue nos registradores visíveis ao programador ou na memória.

Assinale:

a) se somente as afirmativas I e II estiverem corretas.
b) se somente as afirmativas I e III estiverem corretas.
c) se somente as afirmativas I, II e III estiverem corretas.
d) se somente as afirmativas II e III estiverem corretas.
e) se nenhuma afirmativa estiver correta.

15. Considerando os diferentes tipos de organização utilizadas para implementar processamento paralelo, um Analista afirma corretamente:

a) O processamento paralelo com SWAR consiste em utilizar as instruções em um arranjo do tipo MIMD para realizar tarefas em paralelo. Requer programação em baixo nível. Com SWAR é possível fazer processamento paralelo em uma máquina com um único processador.

b) O processamento paralelo com SMP requer computador com mais de um processador com as mesmas características, sendo que os processadores compartilham o barramento e a memória. Os programas podem ser desenvolvidos com o uso de *multithreading* ou múltiplos processos.

c) Beowulf é uma tecnologia de *cluster* que agrupa computadores com sistema operacional GNU/Linux para formar um supercomputador virtual usando processamento paralelo. Requer o uso de uma biblioteca de mensagens como o Mosix, que é gratuito, e o uso de *softwares* para implementação de *clustering* como PVM ou MPI.

d) Na arquitetura paralela baseada em MISD, um único fluxo de instruções opera sobre um único fluxo de dados. Apesar de os programas serem organizados através de instruções sequenciais, elas podem ser executadas em *pipelining*, de forma sobreposta em diferentes estágios.

e) A arquitetura paralela baseada em SIMD envolve o processamento de múltiplos dados por parte de múltiplas instruções. Várias unidades de controle comandam suas unidades funcionais que têm acesso a vários módulos de memória, caracterizando as arquiteturas massivamente paralelas.

16. Considerando-se a taxonomia de sistemas de computação com capacidade de processamento paralelo, associe as arquiteturas de máquinas presentes na primeira coluna (sistemas de computadores) com as descrições sucintas da segunda coluna.

I. SISD
II. SIMD
III. MISD
IV. MIMD

(IV) Um conjunto de elementos processadores executa simultaneamente sequências de instruções diferentes em diferentes conjuntos de dados.
(III) Um grupo de elementos processadores executam diferentes sequências de instruções sobre um mesmo conjunto de dados.
(I) Um único processador executa uma única sequência de instruções para operar em dados armazenados em um único sistema de memória principal.
(II) Uma única instrução controla diversos elementos processadores paralelos, cada um atuando sobre o seu próprio conjunto de dados (memória).

17. O que é o barramento de tempo compartilhado e quais são seus recursos fornecidos?

A organização mais comum para computadores pessoais, estações de trabalho e servidores é o barramento de tempo compartilhado. O barramento de tempo compartilhado é o mecanismo mais simples para construir um sistema multiprocessador (Figura 17.5). As estruturas e as interfaces são basicamente as mesmas para um sistema de um processador único que usa um barramento de interconexão. O barramento consiste em linhas de controle, de endereço e de dados. Para facilitar transferências DMA pelos processadores de E/S, os seguintes recursos são fornecidos:

} **Endereçamento**: deve ser possível distinguir os módulos no barramento para determinar a origem e o destino dos dados.

} **Arbitração**: qualquer módulo de E/S pode funcionar temporariamente como “mestre”. Um mecanismo é fornecido para arbitrar requisições concorrentes para o controle do barramento, usando algum tipo de esquema de prioridade.

} **Tempo compartilhado**: quando um módulo está controlando o barramento, outros módulos são bloqueados e devem, se necessário, suspender a operação até que o acesso ao barramento seja possível. Esses recursos de uniprocessadores são utilizáveis diretamente em uma organização SMP. Nesse último caso, existem agora múltiplos processadores, assim como múltiplos

processadores de E/S, tentando obter o acesso a um ou mais módulos de memória pelo barramento. A organização de barramento possui vários recursos atraentes:

} **Simplicidade**: esta é a abordagem mais simples para a organização de multiprocessadores. A interface física e a lógica de endereçamento, a arbitração e o tempo compartilhado de cada processador permanecem os mesmos, como em um sistema de um único processador.

} **Flexibilidade**: normalmente é fácil expandir o sistema anexando mais processadores ao barramento.

} **Confiabilidade**: o barramento é basicamente um meio passivo, e uma falha de qualquer dispositivo conectado não deve causar uma falha do sistema todo.

18. O termo SMP refere-se a uma arquitetura de hardware computacional e também ao comportamento do sistema operacional que reflete essa arquitetura. Um SMP pode ser definido como um sistema de computação independente com quais características?

1. Há dois ou mais processadores semelhantes de capacidade comparável.

2. Esses processadores compartilham a mesma memória principal e os recursos de E/S, e são interconectados por um barramento ou algum outro esquema de conexão interna, de tal forma que o tempo de acesso à memória é aproximadamente igual para cada processador.

3. Todos os processadores compartilham acesso aos dispositivos de E/S, ou pelos mesmos canais ou por canais diferentes que fornecem caminhos para o mesmo dispositivo.

4. Todos os processadores desempenham as mesmas funções (daí o termo simétrico).

5. O sistema é controlado por um sistema operacional integrado que fornece interação entre os processadores e seus programas em nível de trabalhos, tarefas, arquivos ou elementos de dados.

Os itens de 1 a 4 são autoexplicativos. O item 5 ilustra um dos contrastes com um sistema com multiprocessamento fracamente acoplado, como um cluster. No último, a unidade física de interação é normalmente uma mensagem ou um arquivo completo. Em um SMP, elementos individuais de dados podem constituir o nível de interação e pode haver um alto grau de cooperação entre processos.