Organização de Computadores

Prof. Robson de Souza

Pipeline (paralelismo)

Arquitetos de computadores estão sempre se esforçando para melhorar o desempenho das máquinas que projetam. Aumentar as velocidades de clock é um modo, mas, para cada novo projeto, há um limite para o que é possível fazer por força bruta naquele momento.
Grande parte dos arquitetos de computadores busca o paralelismo (fazer duas ou mais coisas ao mesmo tempo) como um meio de conseguir desempenho ainda melhor para dada velocidade de clock.

O paralelismo tem duas formas gerais, a saber, no nível de instrução e no nível de processador. Na primeira, o paralelismo é explorado dentro de instruções individuais para obter da máquina mais instruções por segundo. Na última, várias CPUs trabalham juntas no mesmo problema.

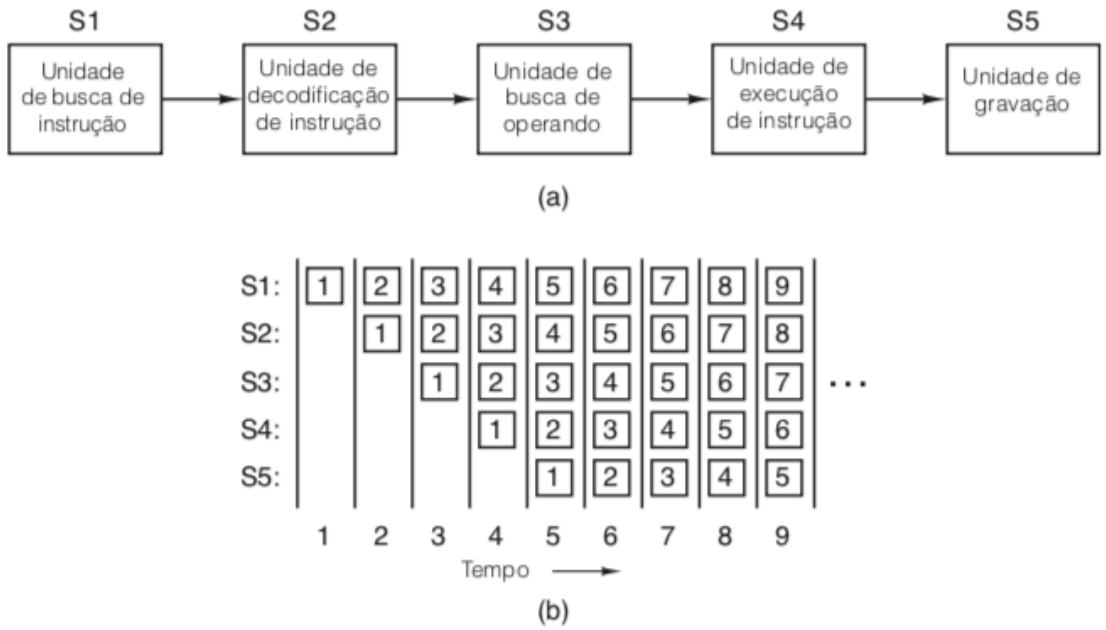
Paralelismo no nível de instrução

Há anos sabe-se que o processo de buscar instruções na memória é um grande gargalo na velocidade de execução da instrução. Para amenizar esse problema, foi criada a **busca antecipada**, que divide a execução da instrução em duas partes: a busca e a execução propriamente dita.

A busca antecipada funcionava de forma que as instruções buscadas antecipadamente eram armazenadas em um conjunto de registradores denominado **buffer de busca antecipada (ou prefetch buffer)**. Desse modo, quando necessária, uma instrução podia ser apanhada no buffer de busca antecipada, em vez de esperar pela conclusão de uma leitura da memória.

O conceito de pipeline amplia muito mais essa estratégia. Em vez de dividir a execução da instrução em apenas duas partes, muitas vezes ela é dividida em muitas partes, cada uma manipulada por uma parte dedicada do hardware, e todas elas podem executar em paralelo.

A figura abaixo ilustra um pipeline com cinco unidades, também denominadas estágios. A parte (b) da figura, mostra os estados de cada estágio como uma função do tempo. São ilustrados nove ciclos de clock. Os quadrados numerados representam as instruções.



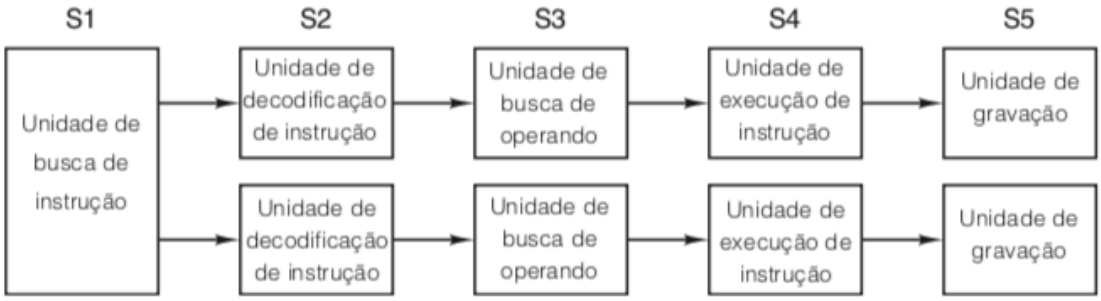
Fonte: (Tanenbaum e Austin, 2013)

Utilizando o pipeline da figura como exemplo, suponha que o tempo de ciclo dessa máquina seja 2 ut (Unidades de Tempo). Sendo assim, uma instrução leva 10 ut para percorrer todo o caminho do pipeline de cinco estágios. Vamos supor que 10 ut sejam equivalentes a 0,1 segundo.

À primeira vista, como uma instrução demora 10 ut, parece que a máquina poderia funcionar em 10 instruções por segundo, mas, na verdade, ela funciona muito melhor do que isso. A cada ciclo de clock (2 ut), uma nova instrução é concluída, portanto, a velocidade real de processamento é 50 instruções por segundo, e não 10.

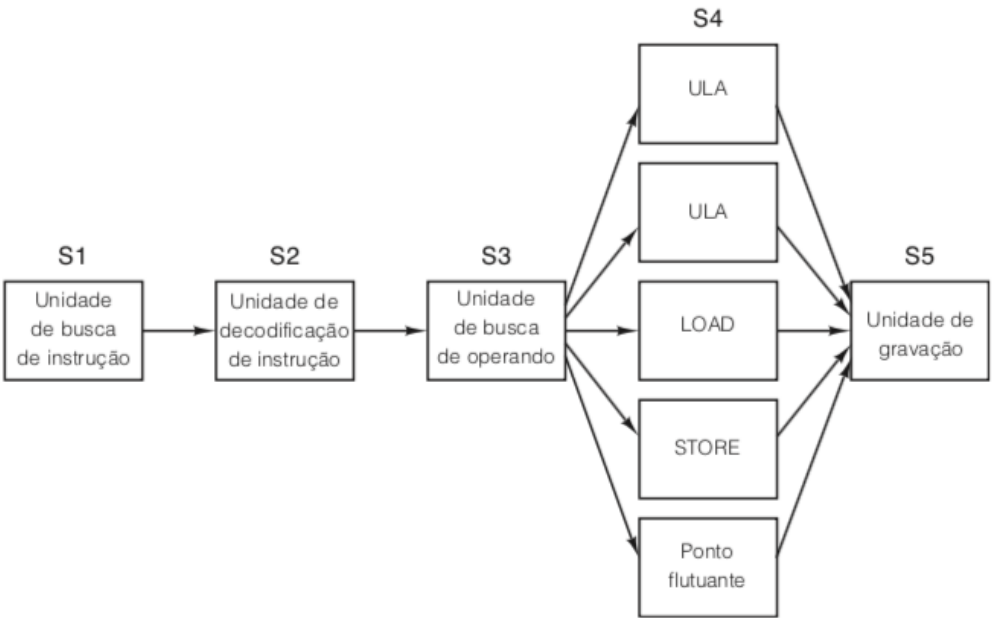
*Arquiteturas superescalares

Arquiteturas superescalares permitem a combinação de mais de um pipeline, inclusive possibilitando que algum estágio seja compartilhado. Por exemplo, na figura abaixo é mostrado um projeto de arquitetura superescalar na qual uma única unidade de busca de instruções busca pares de instruções ao mesmo tempo e coloca cada uma delas em seu próprio pipeline completo com sua própria ULA para operação paralela.



Fonte: (Tanenbaum e Austin, 2013)

Vale ressaltar que, obviamente, para poder executar em paralelo, as duas instruções não devem ter conflito de utilização de recursos (por exemplo, registradores) e nenhuma deve depender do resultado da outra. Além disso, arquiteturas superescalares com várias unidades paralelas devem possuir uma duplicação de hardware, o que nem sempre atinge às exigências de custo de um determinado projeto.
Levando isso em consideração, uma das abordagens utilizada em CPUs topo de linha segue a seguinte ideia: Ter apenas um único pipeline, mas lhe dar várias unidades funcionais, conforme mostra a figura abaixo:



Fonte: (Tanenbaum e Austin, 2013)

Paralelismo no nível do processador

A demanda por computadores cada vez mais rápidos aumenta constantemente. Astrônomos querem simular o que aconteceu no primeiro microssegundo após o Big Bang, economistas querem modelar a economia mundial e cada vez mais pessoas querem se divertir com jogos multimídia em 3D com seus amigos virtuais pela internet.

Chips mais velozes, em algum momento, esbarram em alguma limitação de velocidade, além disso, vale ressaltar que, chips mais velozes também produzem mais calor, cuja dissipação é um problema.

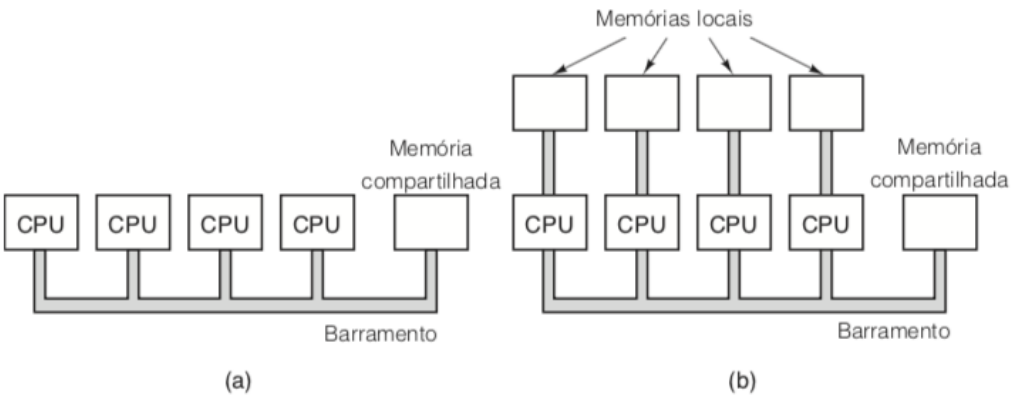
Paralelismo no nível de instrução ajuda um pouco, mas pipelining e operação superescalar raramente rendem mais do que um fator de cinco ou dez. Para obter ganhos de 50, 100 ou mais, a única maneira é projetar computadores com várias CPUs.

*Multiprocessadores

Em uma arquitetura de multiprocessadores, cada processador tem uma espécie de independência, executando tarefas que podem ser distintas dos outros. Nesse caso, as CPUs tem de trocar informações entre si para não gerarem conflitos de tarefas entre outras CPUs.

Há vários esquemas de implementação possíveis. O mais simples é um barramento único com várias CPUs e uma memória, todas ligadas nele. Não é preciso muita imaginação para perceber que, com um grande número de processadores velozes tentando acessar a memória pelo mesmo barramento, surgirão conflitos.

Projetistas de multiprocessadores apresentaram vários esquemas para reduzir essa disputa e melhorar o desempenho. Um desses esquemas, mostrado na figura abaixo na parte (b), que dá a cada processador um pouco de memória local só dele, que não é acessível para os outros. A parte (a) mostra um esquema comum com uma única memória totalmente compartilhada.



Fonte: (Tanenbaum e Austin, 2013)

Multiprocessadores têm uma vantagem sobre outros tipos de computadores paralelos: é fácil trabalhar com o modelo de programação de uma única memória compartilhada.

*Multicomputadores

Embora seja um tanto fácil construir multiprocessadores com um número modesto de processadores, construir grandes é surpreendentemente difícil. A dificuldade está em conectar todos os processadores à memória.

Para evitar esses problemas, muitos projetistas simplesmente abandonaram a ideia de ter uma memória compartilhada e passaram a construir sistemas que consistissem em grandes números de computadores

interconectados, cada um com sua própria memória privada, mas nenhuma em comum. Esses sistemas são denominados **multicomputadores**.

Costuma-se dizer que as CPUs de um multicomputador são **fracamente acopladas,** já as CPUs de um multiprocessador são consideradas **fortemente acopladas**.
As CPUs de um multicomputador se comunicam enviando mensagens umas às outras, mais ou menos como enviar e-mails, porém, com muito mais rapidez.

Uma vez que **multiprocessadores são mais fáceis de programar** e **multicomputadores são mais fáceis de construir**, há muita pesquisa sobre projetos de sistemas híbridos que combinam as boas propriedades de cada um. Esses computadores tentam apresentar a ilusão de memória compartilhada sem bancar a despesa de realmente construí-la.

Referências bibliográficas:

TANENBAUM, Andrew S. Organização Estruturada de Computadores, 2007, 5ª Edição.

TANENBAUM, Andrew S. AUSTIN, Todd; Organização Estruturada de Computadores, 2013, 6ª Edição.