1. **Quais as diferenças de um processador multiciclo em relação a um uniciclo?**

O processador multiciclo, neste tipo de implementação, cada passo de execução gasta um período do clock. A implementação multiciclo permite que uma unidade funcional seja utilizada mais de uma vez por instrução, uma vez que ela está sendo usada em ciclos diferentes do clock. Estapossibilidade de compartilhamento pode ajudar a reduzir a quantidade de hardware necessário à implementação. Em resumo, as principais vantagens da implementação multiciclo são a possibilidade de fazer com que instruções sejam executadas em quantidades diferentes de períodos do clock, e a capacidade de compartilhar unidades funcionais dentro do espaço de tempo necessário à execução de uma únicainstrução.

Já o processador uniciclo, são baseados em um único ciclo de clock, grande o suficiente para acomodar todas as instruções a serem consideradas. Toda a instrução começa sua execução em uma transição ativa do clock e completa a execução na próxima transição ativa do sinal do clock. Assim sendo, todas as instruções gastam o mesmo tempo para serem executadas, tempo este que vaicorresponder ao tempo gasto na execução da instrução mais demorada, e que deve obrigatoriamente ser igual ao ciclo do clock.

1. **Quais as modificações necessárias em um processador multiciclo simples para que se introduza a função de pipeline?**

Pipeline é uma técnica de paralelismo que divide instruções em etapas menores, executando diferentes etapas de diferentes instruções simultaneamente. Desta forma, cada etapa pode ser realizada em um intervalo de tempo menor, assim como em uma implementação multiciclo, além de aumentar a vazão de instruções executadas, isto é, a quantidade de instruções executadas em um determinado intervalo de tempo, o que deverá suprir a necessidade do pipeline para funcionar no multiciclo. Assim como mostra a descrição “foi adicionado um parâmetro que permite definir, em tempo de compilação, se a instrução no branch delay slot será executada; desta forma, um código desenvolvido para uma implementação multiciclo, nos quais o conceito de branch delay slot não existe, pode ser executado nesta implementação pipeline corretamente”.

1. **No programa abaixo, relacione as dependências (dados, WAR, WAW e outros) e conflitos** **existentes para execução em um processador MIPS usando pipeline.**

**Dados**

Os conflitos estruturais e de dados são verificados na fase de ID: Assim que a instrução tem condições para ser executada, é libertada da fase de ID;

Read Operands: espera até que todos os conflitos de dados estejam resolvidos; leitura dos operandos: F|I|R|X|M|W.

**WAR**

O scoreboard regista todas as dependências/conflitos que vão ocorrendo ao longo do programa, determinando quando uma dada instrução pode: Escrever os operandos (write/store), verificando conflitos de dados do tipo WAR e introduzindo stalls sempre que necessário.

Exemplo de conflito WAR:

**div.d** F1, F2, F3

**sub.d** F4, F5, F1

**s.d** F4, 4(F10)

**add.d** F5, F6, F7

**div.d** F4, F5, F6

**WAW**

O scoreboard regista todas as dependências/conflitos que vão ocorrendo ao longo do programa, determinando quando uma dada instrução pode: Ser iniciada (issue), verificando os conflitos estruturais ou de dados do tipo WAW e introduzindo stalls sempre que necessário;

Exemplo de conflito WAW:

**div.d** F1, F2, F3

**sub.d** F4, F5, F1

**s.d** F4, 4(F10)

**add.d** F5, F6, F7

**div.d** F4, F5, F6

**RAW**

O scoreboard regista todas as dependências/conflitos que vão ocorrendo ao longo do programa, determinando quando uma dada instrução pode: Ler operandos (read operands), verificando os conflitos de dados do tipo RAW e introduzindo stalls sempre que necessário;

Exemplo de conflito RAW:

**div.d** F1, F2, F3

**sub.d** F4, F5, F1

**s.d** F4, 4(F10)

**add.d** F5, F6, F7

**div.d** F4, F5, F6

**4) Descreva os seguintes conceitos:**

**a) Write through**

É um método de armazenamento no qual os dados são gravados no cache e no local correspondente da memória principal ao mesmo tempo. Os dados em cache permitem uma recuperação rápida sob demanda, enquanto os mesmos dados na memória principal garantem que nada será perdido se ocorrer uma falha , falha de energia ou outra interrupção do sistema.

**b) Write back**

Um método de armazenamento em cache no qual as modificações nos dados no cache não são copiadas para a origem do cache até que seja absolutamente necessário. O write-back está disponível em muitos microprocessadores , incluindo todos os processadores Intel desde o 80486. Com esses microprocessadores, as modificações de dados (por exemplo, operações de gravação) nos dados armazenados no cache L1 não são copiadas para a memória principal até que seja absolutamente necessário. Por outro lado, um cache de gravação realiza todas as operações de gravação em paralelo - os dados são gravados na memória principal e no cache L1 simultaneamente.

**c) Localidade Temporal**

É quando um dado acessado recentemente tem mais chances de ser usado novamente, do que um dado usado há mais tempo. Isso é verdade porque as variáveis de um programa tendem a ser acessadas várias vezes durante a execução de um programa, e as instruções usam bastante comandos de repetição e sub-programas, o que faz instruções serem acessadas repetidamente. Sendo assim, o Sistema de Memória tende a manter os dados e instruções recentemente acessados no topo da Hierarquia de Memória.

**d) Localidade Espacial**

É quando há uma probabilidade de acesso maior para dados e instruções em endereços próximos àqueles acessados recentemente. Isso também é verdade porque os programas são sequenciais e usam de repetições. Sendo assim, quando uma instrução é acessada, a instrução com maior probabilidade de ser executada em seguida, é a instrução logo a seguir dela. Para as variáveis o princípio é semelhante. Variáveis de um mesmo programa são armazenadas próximas uma às outras, e vetores e matrizes são armazenados em sequência de acordo com seus índices. Baseado neste princípio, o Sistema de Memória tende a manter dados e instruções próximos aos que estão sendo executados no topo da Hierarquia de Memória.