Aluno: Miller Raycell Monteiro Correia

Matricula: 2017009560

Disciplina: Arquitetura e Organização de Computadores

**RESPOSTAS LISTA 2**

1)

Na implementação multiciclo uma instrução pode ser separada e utilizar somente o hardware necessário para aquela fase do processamento, dessa forma pode se ter um reaproveitamento do hardware, o que resulta em menos tempo de ociosidade, além disso somente tem uma única ULA, e também uma única unidade de memória.

2)

Para que seja implementado o pipeline em um processador multiciclo se faz necessário o uso de registradores intermediários entre as fases de processamento para evitar hazard de dados dessa forma não se vai perder dados evitando stalls no processador, tem que se aplicar algoritmos para tratar os problemas de controle do processador, como algoritmo de Tomasulo.

3)

Sem pipeline

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ciclo | 1 | 2 | 3 | 4 | 5 | 6 |  |
| subi $t2, $t2, 4 | IF | ID | EX | WB |  |  |  |
| lw $t1, 0($t2) |  |  |  |  | IF | ID |  |
|  | | | | | | | |
| Ciclo | 7 | 8 | 9 | 10 | 11 | 12 |  |
| lw $t1, 0($t2) | EX | MEM | WB |  |  |  |  |
| add $t3, $t1, $t4 |  |  |  | IF | ID | EX |  |
|  | | | | | | | |
| Ciclo | 13 | 14 | 15 | 16 | 17 | 18 |  |
| add $t3, $t1, $t4 | WB |  |  |  |  |  |  |
| add $t4, $t3, $t3 |  | IF | ID | EX | WB |  |  |
| sw $t4, 0($t2) |  |  |  |  |  | IF |  |
|  | | | | | | |  |
| Ciclo | 19 | 20 | 21 | 22 | 23 | 24 |  |
| sw $t4, 0($t2) | ID | EX | MEM |  |  |  |  |
| beq $t2, $0, loop |  |  |  | IF | ID | EX |  |

Com pipeline

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ciclo | 1 | 2 | 3 | 4 | 5 | 6 |  |
| subi $t2, $t2, 4 | IF | ID | EX | MEM | WB |  |  |
| lw $t1, 0($t2) |  | IF | ID | EX | MEM | WB |  |
| Conflito de dados |  |  | stall |  |  |  |  |
| add $t3, $t1, $t4 |  |  |  | IF | ID | EX |  |
| add $t4, $t3, $t3 |  |  |  |  | IF | ID |  |
| sw $t4, 0($t2) |  |  |  |  |  | IF |  |
|  | | | | | | | |
| Ciclo | 7 | 8 | 9 | 10 | 11 | 12 |  |
| add $t3, $t1, $t4 | MEM | WB |  |  |  |  |  |
| add $t4, $t3, $t3 | EX | MEM | WB |  |  |  |  |
| sw $t4, 0($t2) | ID | EX | MEM | WB |  |  |  |
| beq $t2, $0, loop | IF | ID | EX | MEM | WB |  |  |

4)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ciclo | 1 | 2 | 3 | 4 | 5 | 6 |
| div.d F1, F2, F3 | IF | ID | EX | WB |  |  |
|  | bolha | bolha | bolha | bolha | bolha | bolha |
| sub.d F4, F5, F1 |  |  |  | IF | ID | EX |
|  | | | | | | |
| Ciclo | 7 | 8 | 9 | 10 | 11 | 12 |
| sub.d F4, F5, F1 | WB |  |  |  |  |  |
|  | bolha | bolha | bolha | bolha | bolha | bolha |
| s.d F4, 4(F10) | IF | ID | EX | MEM |  |  |
| add.d F5, F6, F7 |  | IF | ID | EX | WB |  |
|  | bolha | bolha | bolha | bolha | bolha | bolha |
| div.d F4, F5, F6 |  |  |  |  | IF | ID |
|  | | | | | | |
| Ciclo | 13 | 14 | 15 | 16 | 17 | 18 |
| div.d F4, F5, F6 | EX | WB |  |  |  |  |
|  |  |  |  |  |  |  |
| div.d F1, F2, F3 | IF | ID | EX | WB |  |  |
| sub.d F4, F5, F1 |  | IF | ID | EX | WB |  |
| s.d F4, 4(F10) |  |  | IF | ID | EX | MEM |
| add.d F5, F6, F7 |  |  |  | IF | ID | EX |
| div.d F4, F5, F6 |  |  |  |  | IF | ID |
| add.d F5, F6, F7 | WB |  |  |  |  |  |
| div.d F4, F5, F6 | EX | WB |  |  |  |  |
|  |  |  |  |  |  |  |

Conflito 1: RAW: primeiro div.d e sub.d com o registrador F1

Conflito 2: RAW: sub.d e s.d com o registrador F4

Conflito 3: RAW: add.d e segundo div.d com o registrador F5

Conflito 4: WAW: sub.d e segundo div.d com o registrador F4

Conflito 5: WAR: sub.d e add.d com o registrador F5

5)

CPU tempo ideal = (CPU ciclos de clock + Ciclos de Clock de Stall de Memória) x Período de ciclo de Clock =>

CPU tempo ideal = (IC x CPI + 0) x Período de ciclo de Clock =>

CPU tempo ideal = IC x 1 x Período de ciclo de Clock

Ciclos de clock de stall de memória = IC x (1 + 0,5) x 0.02 x 25 = IC x 0,75

CPU tempo real = (IC x 1 + IC x 0,75) x Período de ciclo de Clock =>

CPU tempo real = IC x 1,75 x Período de ciclo de Clock

Comparando os desempenhos com stalls pelo ideal:

6)

a) É relativo à quando a escrita de dados é feita simultaneamente na memoria Cache, quanto no endereço correspondente na memória principal, fazendo que dessa forma não se trabalhe com dados desatualizados, pois sempre que houver uma alteração do valor a memória principal será atualizada, contudo essa técnica trás consigo um maior tempo, pois toda vez ter que atualizar o valor na memória principal tem um gasto.

b) É a técnica que o processador faz a escrita diretamente no Cache e fica como responsabilidade do sistema fazer a atualização dos dados na memória principal, dessa forma se usa menos tempo, pois não tem que ficar atualizando a memoria principal a cada alteração, mas pode acontecer que o processador acabe operando com dados desatualizados, por que o sistema não mandou a instrução para a atualização dos dados na memória principal.

c) Se trata do reconhecimento que um dado que foi acessado recentemente tem mais chance de ser usado novamente, do que um dado que já foi usado há muito tempo, dessa forma os dados recentes que estão na RAM são copiados para o Cache.

d) Se trata do reconhecimento há uma maior chance de se acessar um dado de posições próximas a um dado já acessado, por exemplo programas sequenciais tendem a acessar endereços de memória próximos e com isso quando eu acesso uma instrução a instrução com mais chance de ser acessada será a próxima, com isso as instruções próximas as que estão sendo usadas ficam armazenadas na memoria cache.