Lista 2 - AOC

Ewelly Fabiane

1) Quais as vantagens de um processador multiciclo em relação a um uniciclo?

No **Processador uniciclo** cada instrução é executada em um ciclo de clock, e o tempo do ciclo é medido de acordo com a instrução mais lenta.

Na execução do **Processador multiciclo** as instruções são divididas em 5 etapas: Busca de instrução, Decodificação, Busca de operandos, Execução e Armazenamento do resultado. Em um ciclo de clock é executada uma etapa, cada ciclo faz menos coisas isso quer dizer que o multiciclo executa conjuntos de instruções em menor tempo de execução já que não precisa que todos as trilhas sejam acionadas para executar a instrução e seu tempo de ciclo é definido pela unidade funcional mais lenta. A implementação do multiciclo é semelhante ao uniciclo, porém com uma menor quantidade de componentes no caminho de dados, mas ainda sim sua implementação possui uma lógica de complexidade maior. Outra vantagem do multiciclo é que ainda pode ser implementado pipeline que é uma técnica em que permite a execução das instruções em paralelismo.

2) Quais as modificações necessárias em um processador multiciclo simples para que se introduza a função de pipeline?

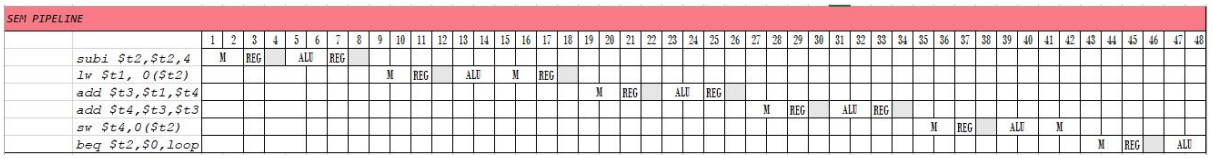
Para a implementação do pipeline no multiciclo são necessárias as seguintes modificações no multiciclo:

- Registradores que funcionarão como buffers temporários, onde serão armazenados os resultados dos componentes e o estado em que a instrução se encontra.
- Flags para a UC ter o controle dos registradores adicionais.
- Tratamento de stalls evitando a parada do pipeline e das instruções, garantindo um funcionamento eficiente do pipeline.

3) Considerando o pipeline do MIPS (simples com MEM compartilhada para instrução e dados) e uma iteração de loop conforme o trecho de programa abaixo, relacione os conflitos que podem ocorrer e seus consequentes stalls. Qual o speedup (por iteração) para o programa e relação à versão sem pipeline?

Nessa questão será considerado que para reduzir os conflitos de dados já estejam implementadas as técnicas de Bypassing (barramento que corta caminho através do pipeline). A técnica de bypassing consiste em transferir um dado via hardware para a próxima instrução ao invés de esperar um estágio do pipeline terminar de executar para então preparar os dados para a próxima instrução que depende deles.

Ciclo de clock sem pipeline = 48 ns



SEM PIPELINE		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	subi $t2,$t2,4	M		REG			ALU		REG																																								
	lw $t1, 0($t2)										M		REG			ALU			M		REG																												
	add $t3,$t1,$t4																				M		REG			ALU			REG																			
	add $t4,$t3,$t3																												M		REG			ALU			REG											
	sw $t4,0($t2)																																				M		REG			ALU			M				
	beq $t2,$0,loop																																													M		REG		ALU

Ciclo de clock com pipeline = 18 ns

COM PIPELINE		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
	subi $t2,$t2,4	M		REG		ALU		REG											
Conflito 1	Dados																		
	lw $t1,0($t2)			M		REG			ALU		M		REG						
Conflito 2	Dados																		
	add $t3,$t1,$t4					M		REG			ALU		REG						
Conflito 3	Dados																		
	add $t4,$t3,$t3							M		REG			ALU		REG				
Conflito 4	Dados																		
Conflito 5	Estrutural										Stall								
	sw $t4,0($t2)											M		REG			ALU		M
Conflito 6	Dados																		
	beq $t2,$0,loop													M		REG			ALU

Loop:

subi $t2, $t2, 4

lw $t1, 0($t2)	Conflito de dados 1
add $t3, $t1, $t4	Conflito de dados 2.
add $t4, $t3, $t3	Conflito de dados 3.
sw $t4, 0($t2)	Conflito de dados 4 e estrutural.
beq $t2, $0, loop	Conflito de dados 5.

Conflito de dados 1: RAW (read after write)
Conflito de dados 2: RAW (read after write)
Conflito de dados 3: RAW (read after write)
Conflito de dados 4: WAW (write after write)
Conflito de dados 5: WAR (write after read)

Speedup = 48/18 = 2,66666...

#As células coloridas (Mais claras) representam os componente que estão gerando conflito de dados.
#A célula colorida (Mais viva e com a palavra em negrito) representa os componente que estão gerando conflito estrutural.
#As células na cor cinza representam o tempo de espera depois da utilização do componente registrador.

4) No programa abaixo, relacione as dependências (dados, WAR, WAW e outros) existentes.

1 div.d F1, F2, F3
2 sub.d F4, F5, F1
3 s.d F4, 4(F10)
4 add.d F5, F6, F7
5 div.d F4, F5, F6

Dependências reais:
Instrução 1 (*div.d F1, F2, F3*) e Instrução 2 (*sub.d F4, F5, F1)*
Instrução 2 (*sub.d F4, F5, F1*) e Instrução 3 (*s.d F4, 4(F10)*)
Instrução 4 (*add.d F5, F6, F7*) e Instrução 5 (*div.d F4, F5, F6*)

Dependência de saída entre o SUB.D e DIV.D:
Instrução 2 (*sub.d F4, F5, F1)* pode terminar depois da Instrução 5 (*div.d F4, F5, F6*)
WAW: Uso do F4.

Antidependência entre SUB.D e ADD.D:
Instrução 4 (*add.d F5, F6, F7*) pode escrever em reg. que a Instrução 2 (*sub.d F4, F5, F1)* lê.
WAR: Uso de F5 por SUB.D.

5) Em relação a memória cache. Um computador tem CPI 1 quando todos os acessos à memória acertam no cache. Loads e Stores totalizam 50% das instruções. Se a penalidade por miss é de 25 ciclos e o miss rate é 2%, qual o desempenho relativo se o computador acertar todos os acessos?

CPU execution time = (CPU clock cycles + Memory stall cycles) x Clock cycle time
= (IC x CPI + 0) x Clock cycle
= IC x 1.0 x Clock cycle

Memory stall cycles = IC x $\frac{\text{Memory accesses}}{\text{Instruction}}$ x Miss rate x Miss penalty
= IC x (1 + 0.5) x 0.02 x 25
= IC x 0.75

CPU execution time (cache) = (CPU clock cycles + Memory stall cycles) x Clock cycle time
= (IC x 1.0 + IC x 0.75) x Clock cycle
= 1.75 x IC x Clock cycle

$$\frac{\text{CPU execution time (cache)}}{\text{CPU execution time}} = \frac{1.75 \times IC \times \text{Clock cycle}}{IC \times 1.0 \times \text{Clock cycle}} = 1.75$$

Resultado final: 1.75

6) Descreva os seguintes conceitos:

a) Write through: escrita ao mesmo tempo no cache e na memória.

Vantagens
- Fácil de implementar
- Um "cache-miss" nunca resulta em escritas na memória;
- A memória tem sempre a informação mais recente.

Desvantagens
- a escrita é lenta;
- cada escrita necessita de um acesso à memória;
- consequentemente usa mais largura de banda da memória.

b) Write back: escrita somente no cache e atualização na memória somente na substituição do bloco.

Vantagens
- A escrita ocorre à velocidade do cache;
- Escritas múltiplas de um endereço requerem apenas uma escrita na memória;
- Consome menos largura de banda.

Desvantagens
- Difícil de implementar;
- Nem sempre existe consistência entre os dados existentes no cache e na memória;
- Leituras de blocos de endereços no cache podem resultar em escritas de blocos de endereços "dirty" na memória

c) Localidade Temporal: Se um item é referenciado, ele tende a ser referenciado novamente dentro de um espaço de tempo curto. Se o estudante tiver trazido o livro recentemente para sua mesa, é provável que o faça em breve novamente.

d) Localidade Espacial: Se um item é referenciado, itens cujos endereços sejam próximos dele tendem a ser logo referenciados.