**1. Defina localidade de referência.**
R:
É uma regra que se divide em 2 princípios diferentes de localidade: temporal e espacial.
(Quando e porquê um determinado dado vai ser levado para a memória cache)

- Temporal: estabelece que se um item é referenciado, ele tende a ser referenciado novamente dentro de um curto espaço de tempo.
  *(Exemplo: se definirmos uma variável "a=4", em um futuro próximo, a mesma será chamada em outra linha de código.*)
- Espacial: estabelece que se um item é referenciado, outros itens cujos endereços estejam próximos dele tendem a ser referenciados rapidamente.
  (Exemplo: *se definirmos uma variável "a=4;b=5;" , as duas estarão , provavelmente, em locais próximos da memória.)*

**2. Considere as seguintes referências a endereços de palavras na memória principal:**
**1, 4, 8, 5, 20, 17, 19, 56, 9, 11 e 4.**
**Considere que a cache está inicialmente vazia e que possui 16 blocos. Para cada um dos mapeamentos, a seguir, identifique cada referência na lista como uma falha ou um acerto na cache.**
**Mostre o conteúdo da cache após o processamento de todas as referências.**
**• Mapeamento direto (uma palavra);**
**1, 4, 8, 5, 20, 17, 19, 56, 9, 11 e 4.**

| ENDEREÇO REAL | CONTEÚDO PARA O RÓTULO | (ENDEREÇO REAL)%16 = ÍNDICE | Cache Hit |
|---|---|---|---|
| 01 =000001 | 00 | 01 = 0001 | F |
| 04 = 000100 | 00 | 04 = 0100 | F |
| 08 = 001000 | 00 | 08 = 1000 | F |
| 05 = 000101 | 00 | 05 = 0101 | F |
| 20 = 010100 | 01 | 04 = 0100 | F |
| 17 = 010001 | 01 | 01 = 0001 | F |
| 19 = 010011 | 01 | 03 = 0011 | F |
| 56 = 111000 | 11 | 08 = 1000 | F |
| 09 = 001001 | 00 | 09 = 1001 | F |
| 11 = 001011 | 00 | 11 = 1011 | F |
| 04 = 000100 | 00 | 04 = 0100 | F |

| I | CONTEUDO | RÓTULO |
|---|---|---|
| 0 | | |
| 1 | ~~MEM[01]~~,MEM[17] | ~~00~~,01 |
| 2 | | |
| 3 | MEM[19] | 01 |

| | | |
|---|---|---|
| 4 | ~~MEM[04]~~,~~MEM[20]~~, MEM[04] | ~~00~~,~~01~~,00 |
| 5 | MEM[05] | 00 |
| 6 | | |
| 7 | | |
| 8 | ~~MEM[08]~~,MEM[56] | ~~00~~,11 |
| 9 | MEM[09] | 00 |
| 10 | | |
| 11 | MEM[11] | 00 |
| 12 | | |
| 13 | | |
| 14 | | |
| 15 | | |

**• Mapeamento associativo por conjunto (8 conjuntos com 2 elementos);**

| **ENDEREÇO REAL** | **CONTEÚDO PARA O RÓTULO** | **(ENDEREÇO REAL)%8 = ÍNDICE** | **Cache Hit** |
|---|---|---|---|
| 01 =000001 | 001 | 01 = 001 | F |
| 04 = 000100 | 000 | 04 = 100 | F |
| 08 = 001000 | 001 | 00 = 000 | F |
| 05 = 000101 | 000 | 05 = 101 | F |
| 20 = 010100 | 010 | 04 = 100 | F |
| 17 = 010001 | 010 | 01 = 001 | F |
| 19 = 010011 | 010 | 03 = 011 | F |
| 56 = 111000 | 111 | 00 = 000 | F |
| 09 = 001001 | 001 | 01 = 001 | F |
| 11 = 001011 | 001 | 03 = 011 | F |
| 04 = 000100 | 000 | 04 = 100 | A |

| **I** | **CONTEUDO** | **RÓTULO** |
|---|---|---|
| 0 | MEM[08] | 001 |

| | | |
|---|---|---|
| 0 | MEM[56] | 111 |
| 1 | MEM[01] | 001 |
| 1 | ~~MEM[17]~~, MEM[09] | ~~010~~, 001 |
| 2 | | |
| 2 | | |
| 3 | MEM[19] | 010 |
| 3 | MEM[11] | 001 |
| 4 | MEM[04] | 000 |
| 4 | MEM[20] | 010 |
| 5 | MEN[05] | 000 |
| 5 | | |
| 6 | | |
| 6 | | |
| 7 | | |
| 7 | | |

**• Mapeamento totalmente associativo.**

| **ENDEREÇO REAL** | **CONTEÚDO PARA O RÓTULO** | **(ENDEREÇO REAL)%1 = ÍNDICE** | **Cache Hit** |
|---|---|---|---|
| 01 =000001 | | | |
| 04 = 000100 | | | |
| 08 = 001000 | | | |
| 05 = 000101 | | | |
| 20 = 010100 | | | |
| 17 = 010001 | | | |
| 19 = 010011 | | | |
| 56 = 111000 | | | |
| 09 = 001001 | | | |

| | | | |
|---|---|---|---|
| 11 = 001011 | | | |
| 04 = 000100 | | | |

| I | CONTEUDO | RÓTULO |
|---|---|---|
| 0 | | |
| 1 | | |
| 2 | | |
| 3 | | |
| 4 | | |
| 5 | | |
| 6 | | |
| 7 | | |

**3. Descreva a organização de uma hierarquia de memória.**

R:

*A **hierarquia de memória*** pode ser analisada segundo suas capacidades de armazenamento, custo por bit e tempo de acesso. A hierarquia mais comum é a seguinte:

1. Memoria cache
   Memória mais rápida com um custo(R$) por bit maior
2. Memoria Principal
   Memória mais rápido e com custo(R$) por bit maior que a memória secundária
3. Memoria secundaria
   Memória mais lenta, mas com capacidade de armazenar grande quantidade de dados, seu custo por bit é o menor.

**4. Descreva um ciclo de barramento num acesso de leitura a memória principal.**

R:

1. Processador inicia o ciclo, colocando o endereço da memória no barramento de endereço e no próximo ciclo de clock ativa o barramento de controle indicando leitura;
2. Após o sinal do endereço e de controle chegarem na memória ela disponibiliza os dados no barramentos de dados, respeitando sua latência/delay;
3. No próximo ciclo de clock o processador recebe os dados e desativa os sinais de endereço e controle, finalizando o clico de barramento.

**5. Defina:**

**a) Célula de bit;**

R: É um circuito eletrônico que armazena um bit de informação.

**b) Locação de memória;**

R: É o menor conjunto de células de bits acessado pelo processador. Normalmente é de 1 byte.

**c) Byte de memória;**

R: É um nome informal para locação de memória de 8 bits.

**6. Quais as principais diferenças entre os tipos de dispositivos de memória instalados num sistema de um computador?**

R:

As diferenças ficam principalmente quanto a capacidade de memória ser escrita e quanto a ser volátil.

1. **PROM**
   (*Programmable* R*ead-Only Memory*)
   Este tipo de dispositivo permite apenas acessos de leitura. As informações armazenadas são atribuídas pelo fabricante do sistema, e não podem ser modificadas após o seu armazenamento. **Não volátil**
2. **EPROM**
   (*Erasable Programmable Read-only Memory*)
   Similar a PROM, porém seus dados podem ser apagados e regravados utilizando um aparelho específico.**Não volátil**
3. **RAM**
   (*Random Access Memory*)
   Seus dados podem ser escritos e lidos pelo processador.**Volátil**

**7. Qual a finalidade da técnica de memória virtual?**

R:

Serve para caso o programa seja maior que a memória principal instalada no computador, o mesmo ainda possa ser executado. Ela usa parte da memória secundária como memória principal.

Exemplo: O HD / dispositivo móvel é utilizado como memória RAM. O controle é feito pelo SO.

**8. Defina:**

**a) DAT** (*Dynamic Address Translator*);

R: É um componente do sub-sistema de memória que realiza o mapeamento entre endereços virtuais e reais.

**b) TLB** *(Translation Lookaside Buffer)***;**

R: É uma pequena memória que o DAT possui agindo como uma memória cache. Ela armazena os últimos pares de endereços acessados (virtual e real).

**9. Quais as questões mais importantes relacionadas à hierarquia de memória para tratar um bloco de memória?**

R:

*Onde colocar, como localizar, substituir e modificar um bloco.*

*RCompleta:*

Onde colocar um bloco;

Pode ser implementada com diversos esquemas, e entre eles , o mapeamento direto,

associativo por conjunto e totalmente associativo.

Como localizar um bloco;

A forma de localizar um bloco depende do esquema implementado para colocação do bloco na hierarquia de memória. A escolha do mapeamento depende do custo de uma falha em relação ao custo da implementação da associatividade.

Como substituir um bloco;

É necessário substituir um bloco de memória quando não existem mais espaços livres naquele nível da hierarquia memória. O bloco candidato à substituição depende da forma como o mapeamento foi implementado.

Como modificar um bloco;

Existem duas opções básicas para tratamento de escritas na memória: *write-through* e *write-back*.

*Write-through*:o dado é escrito tanto na memória cache quanto na memória principal; (Há menos riscos da perda de dados,porém o programa demora mais para executar).

*Write-back*:o dado é escrito somente na memória cache e é escrito na memória principal de tempos em tempos (Há mais risco de perda de dados,porém programa executa mais rápido).

**Cap 5**

**10. Qual a finalidade de uma interface de entrada e saída?**

R: A principal função de uma interface de entrada e saída é tornar transparente para o processador os detalhes de operação e controle dos dispositivos periféricos.

**11. Descreva a organização típica de uma interface de entrada e saída.**

R:

É organizada em duas partes:

1. Genérica
   Parte que interage com o processador. Tem no mínimo um registrador de dados, um registrador de controle e um registrador de estado.

2. Específica
   Parte que interage diretamente com o periférico I/O.
   Exemplos: HD, Teclado, Monitor.

**12. Quais são as técnicas mais comuns de transferência de dados? Resuma cada uma delas.**

R:

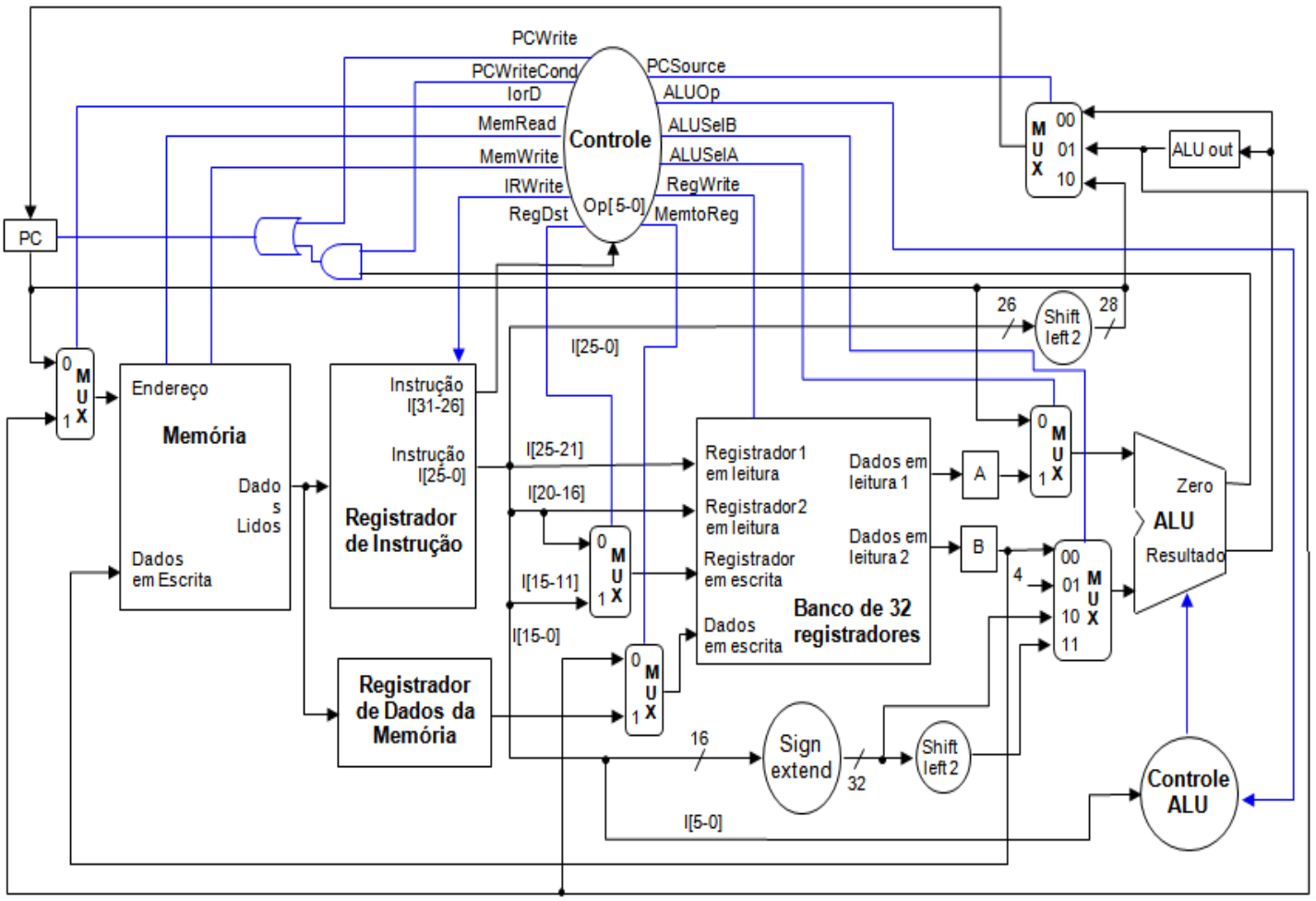
1. **Polling**
   O processador usa uma flag chamada **done bit**. Ela mostra se o dado está sendo escrito no registrador ou no setor do disco. O **polling** é quando o processador fica testando a flag, esperando que os dados tenham sido enviados. Quando a flag indica que o dado foi enviado no setor do disco, o processador verifica se tem outro dado a ser enviado. Caso existam, ele repete o polling para escrever esse o outro dado.
   Obs.: Dado enviado pode ser leitura ou escrita.

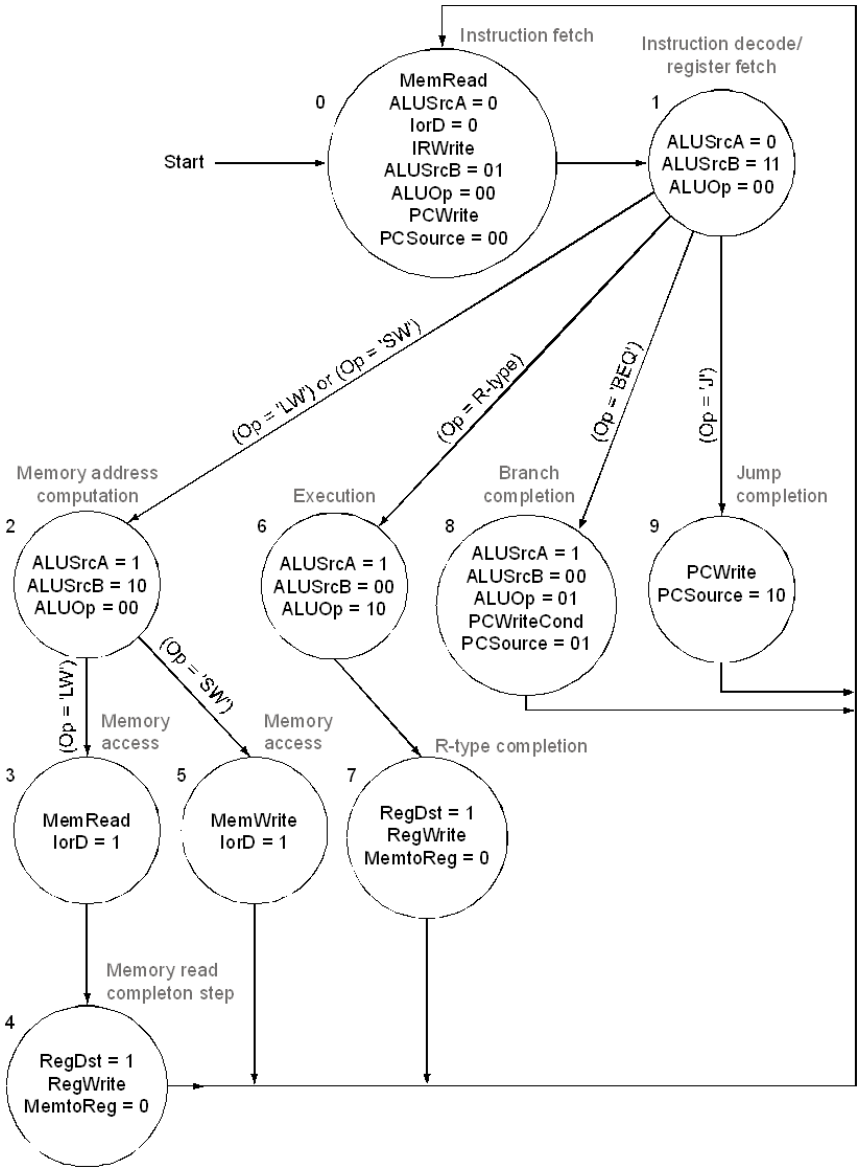
2. **Interrupção**

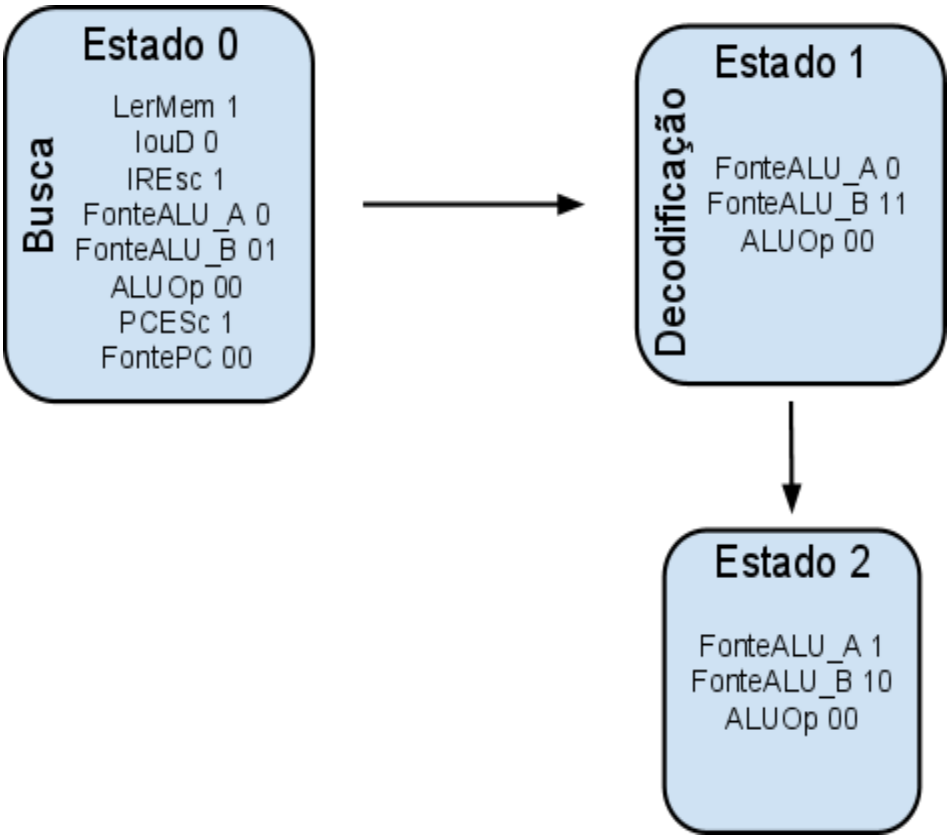
   Requer suporte de hardware. Responsável por *notificar o processador* através de um ***sinal de interrupção***, aplica uma ***rotina de serviço de interrupção***, que são controladas pelo ***controlador de interrupção***. O vetor da rotina de serviço fica na ***tabela de vetores de interrupção***.
   Há um retardo na comunicação entre o controlador e o processador que é chamado de ***tempo de latência de interrupção***.

3. **DMA**
   Nesta técnica o processador só participa do processo de transferência no começo e fim ficando em um estado chamado de *hold state*, onde não pode iniciar ciclos de barramento. Na fase de disparo o processador envia para o controlador DMA o número de dados, o endereço do começo e o sentido *(vindo ou indo pra memória por exemplo).*
   *Em seguida, o processador envia para a interface controladora de disco o número de trilha, o número de setor e o comando da operação.*
   O DMA assume o controle dos barramentos e realiza as transferências, informando ao processador quando tiver terminado.

**Cap 3**
**13. Mostre como é a máquina de estados finitos completa combinando todos os passos de execução de todas as classes de instruções abordadas.**
R:
//acho que tem que desenhar, né? quem acha?
//quem perguntou isso?
//fui eu, o Anderson ;)


PCWrite
PCWriteCond
IorD
MemRead
MemWrite
IRWrite
RegDst
Controle
Op[5-0]
PCSource
ALUOp
ALUSelB
ALUSelA
RegWrite
MemtoReg
MUX
00
01
10
ALU out
PC
0
1
MUX
Endereço
Memória
Dados
s
Lidos
Dados
em Escrita
Instrução
I[31-26]
Instrução
I[25-0]
Registrador
de Instrução
I[25-0]
26
Shift
left 2
28
I[25-21]
I[20-16]
I[15-11]
I[15-0]
Registrador 1
em leitura
Registrador 2
em leitura
Registrador
em escrita
Dados
em escrita
Dados em
leitura 1
Dados em
leitura 2
Banco de 32
registradores
A
B
4
Zero
ALU
Resultado
Registrador
de Dados da
Memória
16
Sign
extend
32
Shift
left 2
Controle
ALU
I[5-0]


Instruction fetch
Instruction decode/
register fetch
0
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
Start
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
Memory address
computation
Execution
Branch
completion
Jump
completion
2
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
6
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 10
8
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01
9
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
Memory
access
Memory
access
R-type completion
3
MemRead
IorD = 1
5
MemWrite
IorD = 1
7
RegDst = 1
RegWrite
MemtoReg = 0
Memory read
completon step
4
RegDst = 1
RegWrite
MemtoReg = 0




**14. Acrescente ao caminho de dados multiciclo a instrução addi. Verifique a necessidade de algum componente de hardware adicional e sinais de controle. Mostre as modificações necessárias na máquina de estados finitos.**
R:

**15. Considere o mesmo enunciado da questão anterior e acrescente a instrução jal (jump and link) ao projeto multiciclo.**
R:

**16. Mostre como a instrução jr (jump register) pode ser implementada simplesmente com algumas modificações na máquina de estados finitos completa (lembre-se que $zero = 0 e está mapeado no registrador $0)**
R: