Universidade Federal de Santa Catarina, INE/CTC
INE 5411 – Organização de Computadores
**Avaliação P2, 2008-2**

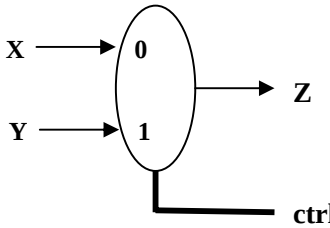
**Nome**:_____**Matrícula**:_____

**Instruções critério de avaliação:** A interpretação das questões é parte integrante desta avaliação. As respostas devem ser fornecidas no espaço para elas reservado. Nas questões que solicitam justificativa, a resposta não será pontuada sem a devida justificativa, nem se esta última estiver incorreta. A pontuação de cada item ("a", "b", etc.) de uma questão é indivisível.

## Parte I - Compreensão de conceitos básicos [3,0 pontos]

**1. [valor: 4 × 0,5 +1,0 = 3,0]** Responda **sucintamente** as perguntas abaixo:

**a)** Para suportar a instrução `jr` no datapath do Anexo III, foi preciso nele incluir o MUX abaixo. Identifique a entrada (E) ou a saída (S) de cada componente daquele datapath que precise ser conectada ao novo MUX. Se uma entrada ou saída de um componente C não tiver um nome, denote-a por E(C) ou S(C), respectivamente. (Entradas, saídas e componentes devem ser identificados de forma não-ambígua).




**X:**

**Y:**

**Z:**

**b)** Mostre os valores dos sinais de controle para implementar a seguinte instrução (codificada sob o formato R) no datapath do Anexo III: `lw rd, (rs+rt) # rd ← Memória[rd+rt].`

| RegDst | ALUSrc | MemtoReg | RegWrite | MemWrite |
|---|---|---|---|---|
| | | | | |

**c)** No código abaixo, onde R denota um registrador arbitrário, a instrução `add` pode ser executada legalmente antes de `beq` em duas condições distintas. Identifique precisamente quais são essas condições:

```
beq $s1, $s2, L
add $t1, $t2, $t3
 j exit
sub $t0, R, $t3
```

**Condição 1:**

**Condição 2:**

**d)** Uma máquina com escalonamento dinâmico pode emitir até 2 instruções por ciclo, desde que uma delas seja aritmética ou desvio e a outra de acesso à memória. Afirmação: "Nestas condições, todas as instruções do segmento de código abaixo poderiam ser emitidas dentro de dois ciclos." A afirmação é verdadeira ou falsa? **Justifique argumentando com as dependências de dados e/ou anti-dependências.**

```
lw $t1, 4($t0)
add $s2, $s1, $t1
sub $s1, $s3, $s4
sw $t3, $t4, $t5
```

**Resposta:** **Justificativa:**

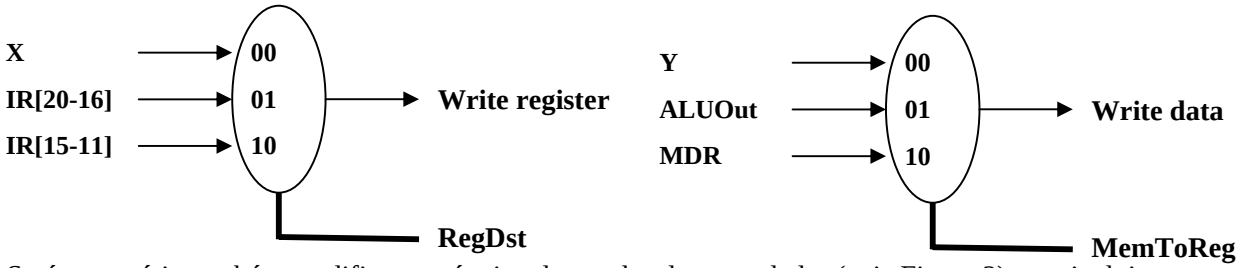
**e)** Mostre a seqüência de 5 instruções nativas do MIPS que implementa o código-fonte abaixo, sabendo-se que as instruções de desvio condicional têm atraso de um ciclo (branch delay = 1). **Restrição**: use esta alocação de registradores: (a,b,c,x,y,z) →($s1,$s2, $s3,$t0,$t1,$s4).

| Código Fonte | Código em linguagem de montagem | |
|---|---|---|
| | **Labels:** | **Instruções** |
| `a = x + y;` | | |
| `if (b = = 0) b = 5;` | | |
| `c = a + 6;` | | |
| `z = b + c;` | | |
| | | |

(Critério de avaliação**:** A questão só será pontuada se os códigos forem semanticamente equivalentes. Cada omissão ou erro de sintaxe será penalizado com -0,5 ponto).

## Parte II – Aplicação de conceitos básicos [4,0 pontos]

**2. [valor: 0,25 + 0,25 + 2 x 0,25 + 15 x 0,1 = 2,5]** Vamos incluir a instrução `jal` ao datapath multiciclo do Anexo IV, através de duas modificações: os multiplexadores controlados pelos sinais RegDst e MemToReg são substituídos por dois novos multiplexadores de 3 entradas conectados conforme ilustra a figura abaixo:




Será necessário também modificar a máquina de estados do controlador (veja Figura 2) para incluir um novo estado: o estado 12. Nestas condições, responda às perguntas e complete as tabelas abaixo: **(Penalidade: 7 ou mais sinais errados anulam totalmente os itens de "d" até "h").**

**a)** Qual o valor de X em binário? **X =**

**b)** A saída de qual componente do datapath é conectada à entrada Y?
**Y conectado com a saída de:**

**c)** Complete as transições que incidem e emergem do novo estado:
**Transição incidente: ( , 12); Transição emergente: (12, )**

**d) Estado 1**

| ALUSrcA | ALUSrcB | | IorD | MemRead | MemWrite | RegDst | | MemtoReg | | RegWrite |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | **X** | **X** | | **X** | **X** | **X** | **X** | |

**e) Estado 3**

| ALUSrcA | ALUSrcB | | IorD | MemRead | MemWrite | RegDst | | MemtoReg | | RegWrite |
|---|---|---|---|---|---|---|---|---|---|---|
| **X** | **X** | **X** | | | | **X** | **X** | **X** | **X** | **0** |

**f) Estado 4**

| ALUSrcA | ALUSrcB | | IorD | MemRead | MemWrite | RegDst | | MemtoReg | | RegWrite |
|---|---|---|---|---|---|---|---|---|---|---|
| **X** | **X** | **X** | **X** | **X** | **0** | | | | | |

**g) Estado 7**

| ALUSrcA | ALUSrcB | | IorD | MemRead | MemWrite | RegDst | | MemtoReg | | RegWrite |
|---|---|---|---|---|---|---|---|---|---|---|
| **X** | **X** | **X** | **X** | **X** | **0** | | | | | |

**h) Estado 12**

| ALUSrcA | ALUSrcB | | IorD | MemRead | MemWrite | RegDst | | MemtoReg | | RegWrite |
|---|---|---|---|---|---|---|---|---|---|---|
| **X** | **X** | **X** | **X** | **X** | **0** | | | | | |

## Convenções: possíveis extensões para o datapath do Anexo V

Nas próximas questões envolvendo pipeline, o datapath básico do Anexo V pode ser estendido com diferentes combinações de recursos adicionais. Para simplificar o enunciado das próximas questões, as extensões estão numeradas e listadas abaixo:

- **Extensão 1**: Para resolver o hazard estrutural resultante do acesso simultâneo ao banco de registradores nos estágios ID e WB, a escrita de um registrador ocorre no primeiro semi-ciclo de relógio; a leitura, no segundo semi-ciclo.
- **Extensão 2**: A operação relacional "≠" do desvio condicional é realizada em uma unidade funcional dedicada, denominada TEST, que faz parte do estágio ID.
- **Extensão 3:** Há um atalho (Forwarding 1) permitindo que o valor à saída da ALU (EX) seja disponibilizado, no ciclo de relógio seguinte, como uma das entradas da unidade TEST (ID).

- **Extensão 4:** Há um atalho (Forwarding 2) permitindo que o valor à porta de leitura da memória de dados (MEM) seja disponibilizado, no ciclo de relógio seguinte, à porta de escrita da própria memória de dados (MEM).
- **Extensão 5:** Há um atalho (Forwarding 3) permitindo que o valor à saída da ALU (EX) seja disponibilizado, no ciclo de relógio seguinte, como uma das entradas da própria ALU (EX).

Para as Questões 3 e 4, você deve completar as tabelas de ocupação do pipeline do datapath do Anexo V de acordo com as seguintes **regras obrigatórias de preenchimento:**

- Use os acrônimos IF, ID, EX, ME e WB para indicar a ocupação de um estágio por uma dada instrução.
- Uma instrução deve iniciar sua execução o mais cedo possível, mas só deve iniciá-la em um ciclo se, e somente se, a partir daquele ciclo ela puder continuar sua execução sem pausa até terminar (ou seja, se houver necessidade de pausa, ela ocorre antes da ocupação do estágio IF).
- Os ciclos em que uma instrução não ocupa um estágio (porque o estágio já está ocupado ou porque está em pausa) devem ser deixados em branco.

3. **[valor: 0,5]** Para o código abaixo, ilustre a ocupação dos estágios do pipeline, sob as seguintes hipóteses:
   - Ao datapath do Anexo V foram acrescentadas as Extensões de 1 a 5.
   - A instrução `bne` tem o comportamento tradicional (não é "delayed branch").

| **Instrução/Ciclo** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Loop: lw $t0, 0 ($t1)** | **IF** | **ID** | **EX** | **ME** | **WB** | | | | | | | |
| **lw $t2, 4 ($t1)** | | | | | | | | | | | | |
| **sw $t2, 0 ($t1)** | | | | | | | | | | | | |
| **sw $t0, 4 ($t1)** | | | | | | | | | | | | |
| **addi $t1, $t1, 8** | | | | | | | | | | | | |
| **bne $t1, $s0, loop** | | | | | | | | | | | | |

**4. [valor: 1,0]** Suponha que uma implementação da arquitetura IA-64 utilize pipelines de 5 estágios como o do Anexo V. Suponha que possam ser iniciadas até 6 instruções por ciclo e que, uma vez iniciada uma instrução, ela ocupa cada um dos estágios uma única vez (mesmo que não o utilize). Para o código abaixo, ilustre a ocupação dos estágios dos pipelines paralelos.

```
{
      .mii
      ld4   r2=[r3]           // r2 ← memória[r3]
      add   r6 = r4, r5       // r6 ← r4 + r5
      sub   r7 = r8, r9 ;;    // r7 ← r8 - r9
}

{
      .mii
      ld4   r10=[r11]         // r10 ← memória[r11]
      sub   r13 = r15, r16 ;; // r13 ← r15 - r16
      add   r12 = r2, r14     // r12 ← r2 + r14

}
```

| **Instrução/Ciclo** | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ld4 | r2=[r3] | **IF** | **ID** | **EX** | **ME** | **WB** | | | | | | | |
| add | r6 = r4, r5 | | | | | | | | | | | | |
| sub | r7 = r8, r9 **;;** | | | | | | | | | | | | |
| ld4 | r10=[r11] | | | | | | | | | | | | |
| sub | r13 = r15, r16 **;;** | | | | | | | | | | | | |
| add | r12 = r2, r14 | | | | | | | | | | | | |

(Critério de avaliação: penalidade de -0,5 ponto para cada instrução cuja temporização estiver incorreta).

## Parte III – Generalização a partir de conceitos básicos [3,0 pontos]

**5.** **[0,5 + 0,5 + 1,0 = 2,0]** Dado o código abaixo da Figura 1(a), otimize-o para os 3 cenários especificados, preenchendo as Figuras 1(b), 1(c) e 1(d). Suponha que: 1) a instrução bne tem o comportamento tradicional (não é "delayed branch"); 2) cada instrução é executada em um pipeline de 5 estágios como o do Anexo 5, ao qual foram acrescentadas as Extensões de 1 a 5; 3) na implementação com emissão dual, tais extensões são duplicadas entre elementos de pipelines paralelos.

a) **Cenário 1** - Corpo do laço original escalonado para eliminar todos os ciclos de pausa em um datapath com emissão de uma única instrução por ciclo nas seguintes condições:

- As instruções do laço original devem ser preservadas sem modificações, exceto pela necessidade de compensação de "offset". Permite-se que apenas um "offset" seja compensado para fins de otimização.

b) **Cenário 2** - Corpo do laço desenrolado duas vezes (sem escalonamento) nas seguintes condições:

- Deve-se eliminar instruções de controle do laço duplicadas, preservando a semântica e a ordem das instruções do código original, através de ajustes de constante imediata e todas as compensações de "offset" que se fizerem necessárias. Quaisquer outras otimizações são proibidas.

c) **Cenário 3** – Corpo do laço desenrolado duas vezes nas seguintes condições:

- Deve-se eliminar instruções de controle do laço duplicadas, preservando a semântica do código original, através de ajustes de constante imediata e todas as compensações de "offset" que se fizerem necessárias.
- O corpo do laço deve ser escalonado de forma a obter o mínimo número de ciclos de pausa em um datapath com emissão de duas instruções por ciclo, sendo que a primeira deve ser aritmética ou desvio e a segunda deve ser de acesso à memória;
- Não deve ser aplicado renomeamento de registradores nem quaisquer outras otimizações.
- Quando duas instruções são emitidas no mesmo ciclo, se o registrador destino de uma delas é fonte da outra, deve-se supor que o valor armazenado no registrador-fonte é lido por uma instrução antes de ser modificado como registrador-destino da outra.

(Critério de avaliação: Em cada item desta questão, cada erro será penalizado com -0,25 ponto).

6. **[valor: 0,5]** Dada a atribuição x = a+b*c+d, mostre que há duas alternativas para implementá-la em assembly tais que uma delas resulta otimizada para executar em um processador com emissão múltipla. Assuma que (x,a,b,c,d) sejam alocadas em ($s0,$s1,$s2,$s3,$s4). Quaisquer outros valores devem ser alocados em registradores temporários. (Use as instruções mul e addu).

| **Não otimizada** | | **Otimizada** |
|---|---|---|
| | | |
| | | |
| | | |

(Critério de avaliação**:** Esta questão não será pontuada se você inverter as versões otimizada e não-otimizada. Ademais, cada erro de sintaxe será penalizado com -0,25 ponto).

**[valor: 0,5]** Seja o datapath original do Anexo 5, sem as extensões de 1 a 5. Suponha que a saída do somador que calcula o endereço-alvo, ao invés de ser ligada ao registrador EX/MEM, seja conectada diretamente à entrada 1 do MUX que define a entrada do PC. Afirmação: "O datapath assim modificado implementa desvios condicionais com atraso de 1 ciclo ("delayed branch slot = 1"). A afirmação é verdadeira ou falsa? **Justifique**, preenchendo o diagrama abaixo e indicando em que transição de relógio o PC é carregado com o endereço-alvo. **Resposta:** **Justificativa:**

| **Instrução/Ciclo** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **bne $s2, $zero, L** | **IF** | **ID** | **EX** | **ME** | **WB** | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |

| Código original | |
|---|---|
| loop: | lw $t0, 0 ($t1) |
| | lw $t2, 4 ($t1) |
| | sw $t2, 0 ($t1) |
| | sw $t0, 4 ($t1) |
| | addi $t1, $t1, 8 |
| | bne $t1, $s0, loop |

**( a )**

| Cenário 1 | |
|---|---|
| loop: | lw $t0, 0 ($t1) |
| | lw $t2, 4 ($t1) |
| | sw $t2, 0 ($t1) |
| | |
| | |
| | |

**( b )**

| Cenário 2 | |
|---|---|
| loop: | lw $t0, 0 ($t1) |
| | lw $t2, 4 ($t1) |
| | sw $t2, 0 ($t1) |
| | sw $t0, 4 ($t1) |
| | |
| | |
| | |
| | |
| | addi $t1, $t1, |
| | bne $t1, $s0, loop |

**( c )**

| Cenário 3 | | |
|---|---|---|
| | **aritmética ou desvio** | **load ou store** |
| loop: | nop | lw $t0, 0 ($t1) |
| | nop | lw $t2, 4 ($t1) |
| | nop | sw $t2, 0 ($t1) |
| | nop | sw $t0, 4 ($t1) |
| | nop | |
| | | |
| | | |
| | | |

**( d )**

**Figura 1 – Cenários de transformação de código**

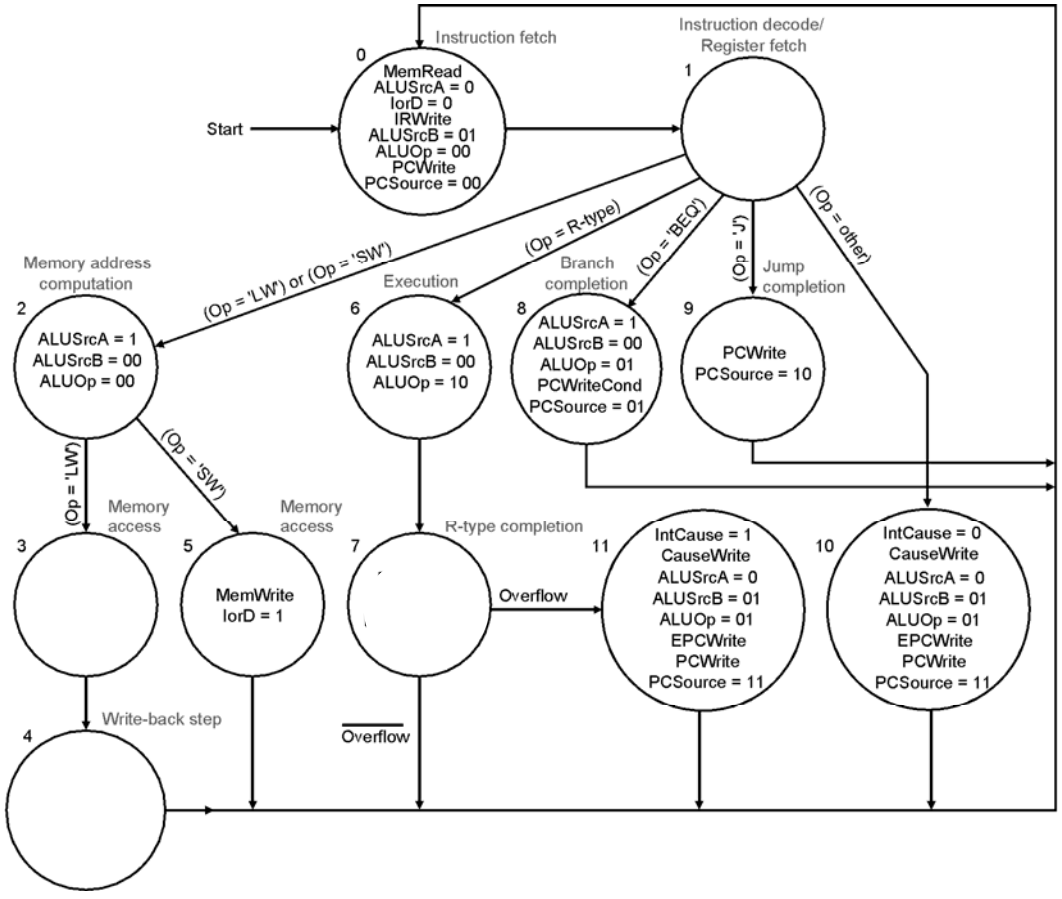



**Figura 2 – Diagrama de transição de estados para o datapath multiciclo (veja Anexo IV)**