## Grupo I

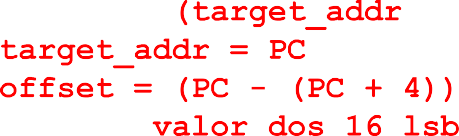
Responda às questões seguintes, **mostrando todos os passos da sua resposta**:

1. O valor dos 16 bits menos significativos, expresso em hexadecimal, do código máquina da instrução

## L1: beq $1,$1,L1 é:







1. Suponha que **$2=0xA35** e que se pretende aceder, através da instrução **LW**, ao endereço de memória **0xA31**. Para que isso aconteça, o valor dos 16 bits menos significativos, expresso em hexadecimal, do código máquina da instrução "**lw $3,??($2)**" deve ser:







1. O valor **0xAC640100** é o código máquina de uma instrução do MIPS. Apresente a instrução *Assembly* completa a que corresponde esse código (mnemónica e argumentos – consulte a tabela de códigos disponível no Grupo III):

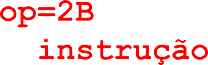
 











1. Admita que os valores indicados no *datapath* da Figura 1 correspondem à “fotografia” tirada no decurso da execução de uma dada instrução. Observe todos os sinais e valores presentes nessa figura e responda às seguintes questões:
   1. A fase de execução em que se encontra é , porque

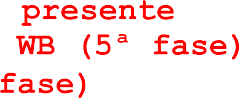


* 1. A instrução que se encontra em execução é (não necessita de colocar a instrução completa)

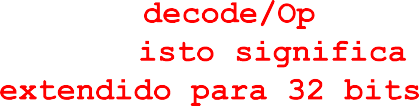
 , porque



* 1. A instrução que vai ser executada de seguida encontra-se no endereço , porque



* 1. Admita que o sinal "**RegWrite**" tinha o valor '**0**'. Podia então concluir-se que a instrução em execução estava na fase , porque





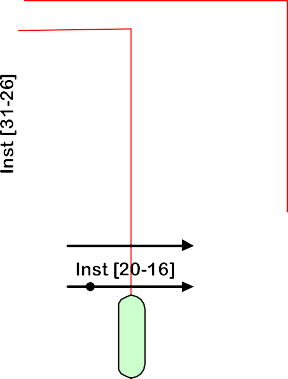


Teste prático 2 Arquitetura de Computadores I 23-01-2018

**Instruction Register**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **0** | **PCWrite** |  | | |
| **0** | **PCWriteCond** |  | **PCSource** | **01** |
| **0** | **IorD** |  | **ALUOp** | **00** |
| **0** | **MemRead** | **Control** | **ALUSelA** | **0** |
| **0** | **MemWrite** | **Unit** | **ALUSelB** | **11** |
| **0** | **IRWrite** |  | **RegWrite** | **1** |
| **1** | **MemtoReg** |  | **RegDst** |  |

Figura 1. *Datapath multi-cycle* do MIPS.



**1**

32

2

32

M

1

U

0 X **M6**

Op[5-0]

**0**

**Reset**

**Shift Left 2**

**M4**

**1**

**M1 Read Write**

**Write**

**Write**

**PC**

0M U 1X

**R Reg.#1**

32

**Address**

**R Data #1**

A

0M U 1X

Zero

32

**R Reg.#2**

**Splitter**

**Read Data**

1. M **M2**

U

1. X

32

**M5**

**W Reg.**

**R Data #2**

B

**ALU**

**Result**

**Reset**

4

0

1 M

**Alu Out**

**Write Data** **W Data**

**Registos**

2 U

**Memory**

X

3

32

**Data Register**

32

1M U

0X **M3**

16

**Sign Extend**

32

**Clock**

**Shift Left 2**

ALU

Control

32

Zona de rascunho

## Grupo II

Considere o *datapath* e a unidade de controlo fornecidos na Figura 1, sabendo que corresponde a uma implementação

*multi*-*cycle* simplificada do MIPS, sem *pipelining*.

1. Preencha a tabela seguinte com o nome de cada uma das fases de execução da instrução "**xor $1,$2,$3**" e com o valor que tomam, em cada uma delas, os sinais de controlo ali indicados. Admita que o valor lógico “1” corresponde ao estado ativo. **Assinale as situações de "don't care" com "X"**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Designação da Fase** | **Instruction Fetch/Calc. PC+4** |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PCWrite** |  |  |  |  |  |
| **MemWrite** |  |  |  |  |  |
| **IRWrite** |  |  |  |  |  |
| **ALUOp** |  |  |  |  |  |
| **ALUSelA** |  |  |  |  |  |
| **ALUSelB** |  |  |  |  |  |
| **IorD** |  |  |  |  |  |
| **PCSource** |  |  |  |  |  |
| **MemRead** |  |  |  |  |  |
| **RegWrite** |  |  |  |  |  |
| **RegDst** |  |  |  |  |  |

1. Preencha a tabela seguinte com o nome de cada uma das fases de execução da instrução "**xor $1,$2,$3**" (código máquina **0x00430826**) e com o valor que tomam, em cada uma delas, os valores do *datapath* ali indicados. Considere que os registos, no instante em que vai iniciar-se o *instruction fetch*, têm os seguintes valores: **$1=0x145**,

## $2=0x3A4, $3=0x75D, PC=0x0040008C). Assinale as situações de "valor desconhecido" com "?".

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Designação da fase** | **Instruction Fetch/Calc. PC+4** |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PC** | **0x0040008C** |  |  |  |  |
| **Instr. Register** |  |  |  |  |  |
| **Data Register** |  |  |  |  |  |
| **A** |  |  |  |  |  |
| **B** |  |  |  |  |  |
| **ALU Result** |  |  |  |  |  |
| **ALU Out** |  |  |  |  |  |
| **ALU Zero** |  |  |  |  |  |



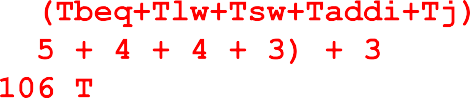
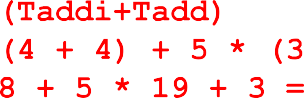
**ALUOp – 00: Add, 01: Subtract, 10: R-Type, 11: Set if Less Than**









## Grupo III

Considere o trecho de código da tabela ao lado onde o endereço representado pelo *label* **L1** é **0x0000A204**.

1. Traduza para código máquina do MIPS as instruções abaixo indicadas (expressando o resultado em hexadecimal) e indique o endereço de memória em que se encontra cada uma. Mostre todos os passos da sua resposta e, no final, **preencha a tabela**.

**L1: addi $2,$0,0x14 add** **$3,$0,$0**

**L2: beq** **$3,$2,L3**

**lw $4,0x100($3) sw $4,0x200($3) addi $3,$3,4**

**j L2**

**L3: ...**

|  |  |  |
| --- | --- | --- |
| ***Opcode*** | ***Funct*** | **Instr .** |
| **0** | **0x20** | **add** |
| **0** | **0x26** | **xor** |
| **0x04** |  | **beq** |
| **0x2B** |  | **sw** |
| **0x02** |  | **j** |
| **0x08** |  | **addi** |
| **0x0A** |  | **slti** |
| **0x23** |  | **lw** |

## sw $4,0x200($3)









## j L2



|  |  |  |
| --- | --- | --- |
| **Endereço** | **Instrução** | **Código Máquina (hexadecimal)** |
|  | **sw** **$4,0x200($3)** |  |
|  | **j** **L2** |  |

1. Calcule o número total de ciclos de relógio que demora a execução completa desse trecho de código (desde o instante inicial do *instruction fetch* da primeira instrução até ao momento em que vai iniciar-se o *instruction fetch* da instrução presente em "L3:"): i) num *datapath single-cycle*; ii) num *datapath multi-cycle*. **Apresente todos os passos que justifiquem a sua resposta (a simples apresentação de valores sem justificação adequada terá cotação 0)**.



# Single-cycle:







* 1. *Multi-cycle:*





a) O código VHDL que se apresenta de seguida corresponde a uma possível implementação do módulo "PC\_update" para a arquitetura ***single-cycle*** do MIPS que implementou nas aulas práticas, responsável pela manutenção e atualização do valor do *Program Counter*. Complete o diagrama temporal da figura seguinte, calculando o valor de saída ("**pc**") para todos os ciclos de relógio ali apresentados (note que os valores de "**jAddr**" e "**offset32**" estão representados em hexadecimal).

**entity PC\_update is**

**port(clk, reset, branch, jump, zero : in std\_logic; offset32 : in std\_logic\_vector(31 downto 0); jAddr** **: in std\_logic\_vector(25 downto 0); pc** **: out std\_logic\_vector(31 downto 0));**

**end PC\_update;**

**architecture Behavioral of PC\_update is**

**signal s\_pc, s\_offsetSL2 : unsigned(31 downto 0); signal s\_pc4 : unsigned(31 downto 0);**

**begin**

**s\_offsetSL2 <= unsigned(offset32(29 downto 0)) & "00"; s\_pc4 <= s\_pc + 4;**

**process(clk) begin**

**if(rising\_edge(clk)) then if(reset = '1') then**

**s\_pc <= (others => '0'); else**

**if(jump = '1') then**

**s\_pc <= s\_pc4(31 downto 28) & unsigned(jAddr) & "00"; elsif(branch = '1' and zero = '1') then**

**s\_pc <= s\_pc4 + s\_offsetSL2; else**

**s\_pc <= s\_pc4; end if;**

**end if; end if;**

**end process;**

**pc <= std\_logic\_vector(s\_pc); end Behavioral;**



**?**

**?**

**04EA1F3**

**FFFFA1F3**

**0640014**

**20A925A**

**1253F47**

**0000201**

**0000320**

**00000014**

**FFFF925A**

**00003F47**

**00000201**

**00000320**

**?**





Teste prático 2 Arquitetura de Computadores I 23-01-2018



a) Complete o código VHDL seguinte com a implementação de um registo de N bits com *reset* síncrono e *enable*. A ação de *reset* não deve depender do sinal *enable*.

**entity RegisterN is**

**generic (**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**);**

**port (** : **std\_logic**

: **std\_logic**

: **std\_logic**



: **std\_logic**

: **std\_logic );** **end RegisterN;**



**architecture of is**

**end behav;**

Zona de rascunho

**Cotações**: *Grupo I* (4.5 valores): 1, 1, 1, 1.5; *Grupo II* (5.5 valores)*:* 2.5, 3; *Grupo III:* (6 valores): 3, 3; *Grupo IV:* 2; *Grupo V*: 2 (8)