UFSM
Universidade Federal de Santa Maria

Departamento de Eletrônica e Computação
ELC 1011 – Organização de Computadores

15/02/2021

Exame

Nome:_____

1 – (1,0 ponto) Considere duas máquinas, P1 e P2, que usam o mesmo conjunto de instruções mas possuem implementações diferentes. O conjunto de instruções possui 4 classes de instruções: A, B, C e D. A frequência de relógio e o CPI em cada uma das implementações é dada pela seguinte tabela:

	Frequência do relógio (GHz)	CPI das classes de Instruções			
		A	B	C	D
P1	2,0	1	4	4	2
P2	2,5	2	2	2	2

Executamos um programa X, com 10^6 instruções, dividida nas classes da seguinte forma: 10% na classe A, 20% na classe B, 50% na classe C e 20% na classe D. Qual das máquinas possui melhor desempenho? Por quê?

2 – (4,0 pontos) Traduza o seguinte programa de C para o *assembly* do processador MIPS.

```c
int valor1 = 10;

int procedimento3(int x) {
  int tmp;

  if (x > 5) {
    tmp = x;
  } else {
    tmp = x + valor1;
  }

  return x;
}

int procedimento2(int v1[], int n) {
  int i;

  int acc;
  acc = 0;

  for (i = 0; i < n; i++) {
    acc = acc + procedimento3(v1[i]);
  }

  return acc;
}

int procedimento1(int x, int y) {
  int vetorA[10];
  int vetorB[10];
  int i;
  int resultado;

  resultado = 0;
```

```
  for (int i = 0; i < 10; i++) {
    vetorA[i] = x + i;
    vetorB[i] = y + i;
  }
  resultado = procedimento2(vetorA, 10) - procedimento2(vetorB, 10);
  return resultado;
}

int main(void) {
  int n;
  int m;
  int resultado;

  n = 5;
  m = 3;

  resultado = procedimento1(n, m);
  return 0;
}
```

3 – (1,0 ponto) Converta os números A = 20,3 e B = 30,4 para ponto fixo. Use 12 bits: 6 bits para a parte inteira e 6 bits para a parte fracionária. Os números são representados sem sinal. Faça a soma binária de (A+B) e a subtração (A-B). Mostre as operações com os vem-uns. Houve estouro (overflow) na operação?

4 – (2,0 pontos) Converta os números A = 20,3 e B = 30,4 para ponto fixo. Use 12 bits: 6 bits para a parte inteira e 6 bits para a parte fracionária. Os números são representados em complemento de 2. Faça a soma binária de (A+B) e a subtração (A-B). Mostre as operações com os vem-uns. Houve estouro (overflow) na operação? Escreva um trecho de código em assembly MIPS, para verificar se houve o estouro na operação de soma.

4 – (2,0 pontos) (a) Converta a instrução em linguagem de montagem **beq $s0**, **$s1**, **loop** para linguagem de máquina e (b) explique detalhadamente, com a ajuda da figura 2 e as tabelas 7 e 8, como a instrução em linguagem de máquina é executada pelo processador monociclo. A instrução **beq** está no endereço 0x0040000C. **Loop** é um rótulo para o endereço 0x00400000.

Tabela A. Algumas instruções e pseudoinstruções do MIPS.

Nome	Sintaxe	Significado	Nota
Add	add $1,$2,$3	$1 = $2 + $3	adds two registers, extends sign to width of register
Subtract	sub $1,$2,$3	$1 = $2 – $3	subtracts two registers
Add immediate	addi $1,$2,CONST	$1 = $2 + CONST (signed)	Used to add constants (and also to copy one register to another "addi $1, $2, 0"), with sign extension
Load word	lw $1,CONST($2)	$1 = Memory[$2 + CONST]	loads the word stored from: MEM[$s2+CONST] and the following 3 bytes.
Load byte	lb $1,CONST($2)	$1 = Memory[$2 + CONST] (signed)	loads the byte stored from: MEM[$2+CONST]
Store word	sw $1,CONST($2)	Memory[$2 + CONST] = $1	stores a word into: MEM[$2+CONST] and the following 3 bytes.
Store byte	sb $1,CONST($2)	Memory[$2 + CONST] = $1	stores the first fourth of a register (a byte) into: MEM[$2+CONST]
And	and $1,$2,$3	$1 = $2 & $3	R Bitwise and
And immediate	andi $1,$2,CONST	$1 = $2 & CONST	R Bitwise or

Or	or $1,$2,$3	$1 = $2 | $3	
Or immediate	ori $1,$2,CONST	$1 = $2 | CONST	
Branch on equal	beq $1,$2,CONST	if ($1 == $2) go to PC+4+CONST	Goes to the instruction at the specified address if two registers are equal.
Branch on not equal	bne $1,$2,CONST	if ($1 != $2) go to PC+4+CONST	Goes to the instruction at the specified address if two registers are *not* equal.
Jump	j CONST	goto address CONST J	Unconditionally jumps to the instruction at the specified address.
Jump register	jr $1	goto address $1	Jumps to the address contained in the specified register
jump and link	jal CONST	$31 = PC + 4; goto CONST	For procedure call - used to call a subroutine, $31 holds the return address; returning from a subroutine is done by: jr $31
Set on less than	slt $1,$2,$3	$1 = ($2 < $3)	Tests if one register is less than another.
Set on less than immediate	slti $1,$2,CONST	$1 = ($2 < CONST)	Tests if one register is less than a constant.
Load Address	la $1, LabelAddr		$1 = Label Address
Load Immediate	li $1, IMMED[31:0]		$1 = 32 bit Immediate value

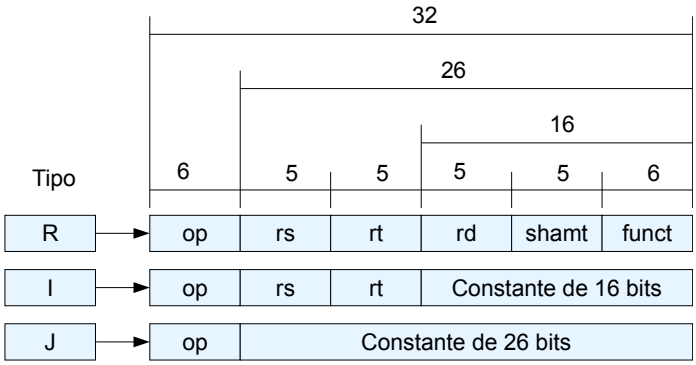

Figura 1 – Ilustração dos campos das instruções R, I e J de um processador MIPS.

Tabela 1 - Exemplos de instruções tipo R (opcode = 000000)

Instrução	Campo funct
add rd, rs, rt	0x20
addu rd, rs, rt	0x21
and rd, rs, rt	0x24
nor rd, rs, rt	0x27
sll rd, rt, shamt	0x00
sllv rd, rt, rs	0x04
sra rd, rt, shamt	0x03
sub rd, rs, rt	0x22
subu rd, rs, rt	0x23
slt rd, rs, rt	0x2A
sltu rd, rs, rt	0x2B

Tabela 2 - Exemplo de instruções tipo I

Instrução	Opcode
addi rt, rs, imediato	0x08
xori rt, rs, imediato	0x0E
slti rt, rs, imediato	0x0A
beq rs, rt, label	0x04
lb rt, imediato(rs)	0x20
lw rt, imediato(rs)	0x23
sb rt, imediato(rs)	0x28
sw rt, imediato(rs)	0x2b

Tabela 3 - Exemplo de instruções tipo J

Instrução	Opcode
j endereço-alvo	0x02

jal endereço-alvo	0x03

Tabela 4 – Nome e número dos registradores do MIPS.

$v0 a $v1	2 a 3		$gp	28
$a0 a $a3	4 a 7		$sp	29
$t0 a $t7	8 a 15		$fp	30
$s0 a $s7	16 a 23		$ra	31

Nome	**número**		**Nome**	**Número**
$zero	0		$t8 a $t9	24 a 25
$at	1		$k0 a $k1	26 a 27

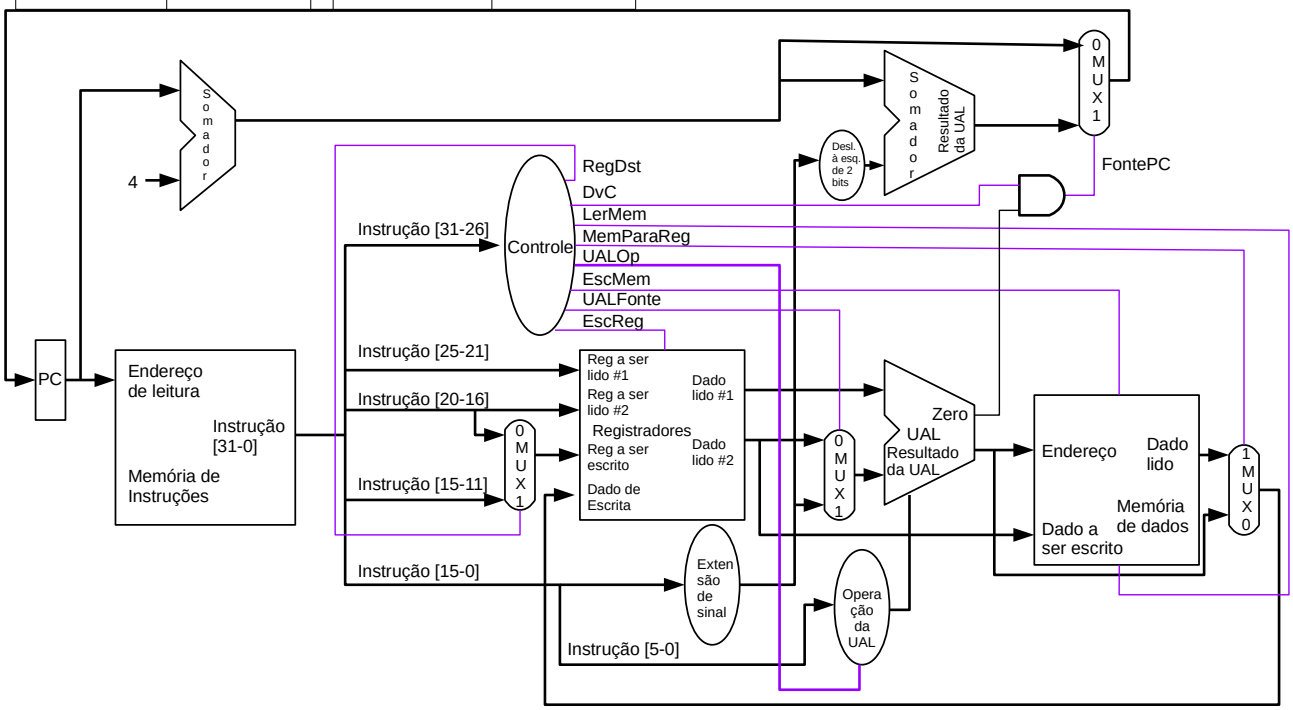

Figura 2 – Caminho de dados e de controle de um processador monociclo.

Tabela 7 – Instruções e valores dos sinais na unidade de controle da figura 2.

Controle	**Nome do Sinal**	**Formato R (0)**	**Lw (35)**	**Sw (43)**	**Beq (4)**
Entradas	OP5	0	1	1	0
	OP4	0	0	0	0
	OP3	0	0	1	0
	OP2	0	0	0	1
	OP1	0	1	1	0
	OP0	0	1	1	0
Saídas	RegDst	1	0	X	X
	UalFonte	0	1	1	0
	MemParaReg	0	1	X	X
	EscReg	1	1	0	0
	LerMem	0	1	0	0
	EscMem	0	0	1	0
	DvC	0	0	0	1
	UALOp1	1	0	0	0
	UALOp0	0	0	0	1

Tabela 8 – Operação da UAL para a combinação da UALOp e campo de função.

UALOp		Campo de Função						Operação da UAL	
UALOp1	UALOp1	F5	F4	F3	F2	F1	F0		
0	0	X	X	X	X	X	X	010	soma

X	1	X	X	X	X	X	X	110	subtração
1	X	X	X	0	0	0	0	010	soma
1	X	X	X	0	0	1	0	110	subtração
1	X	X	X	0	1	0	0	000	AND
1	X	X	X	0	1	0	1	001	OR
1	X	X	X	1	0	1	0	111	Slt