UFSM Universidade Federal de Santa Maria
Centro de Tecnologia – DELC
ELC1011 – Organização de Computadores

Nome: _____

2ª Prova (01/02/2023)

Gabarito

Responda a cada uma das questões de forma clara e organizada. Apresente todos os passos da solução.

Questão 01 (1,5 ponto, 10 min.) – Converta as seguintes instruções em *assembly* para instruções em linguagem de máquina. Considere que loop é um rótulo para o endereço $0x0040\ 002C$.

Endereço	Instrução	Comentário
$0x0040\ 0004$	j loop	# (opcode = 0x02)
$0x0040\ 0008$	$\text{bne \$s0, \$t3, loop}$	# (opcode = 0x05, bne rs, rt, imm)

0x0040 0004 j loop

Da figura 2, encontramos que esta instrução tipo **J** possui dois campos: o campo opcode com 6 bits e um valor imediato de 26 bits. O opcode $0x02 = 00\ 0010_2$ foi dado no comentário do problema. Os 4 bits mais significativos do valor de PC+4 e do endereço loop são iguais: o desvio pode ser executado. O valor imediato de 26 bit é igual aos 28 bits menos significativos de loop ($0x0040\_002C = 0000\ 0000\ 0100\ 0000\ 0000\ 0000\ 0010\ 1100_2$), deslocados de 2 bits para a direita ($00\ 0001\ 0000\ 0000\ 0000\ 0000\ 1011_2$). Concatenamos os campos opcode e o valor imediato:

$$\underset{\text{opcode}\quad\text{valor imediato}}{\text{j loop} = \left|\ 0000\ 10\ \right|\ 00\ 0001\ 0000\ 0000\ 0000\ 0000\ 1011\ \left|\right.} = \quad 0x0810\ 000B$$

bne $s0, $t3, loop

Esta é uma instrução tipo **I**. Os campos desta instrução estão apresentados na figura 2: opcode, registrador rs, registrador rt e valor imediato. O opcode e o formato da instrução foi apresentado no comentário junto à instrução. O opcode é $0x05 = 00\ 0101_2$, rs = $\$s0 = \$16 = 10000$ e rt = $\$t3 = \$11 = 01011$. O campo imediato é igual a $(\text{loop - } (\text{PC}+4))>>2$ = $(0x0040\_002C - 0x0040\_000C)>>2 = 0x0020>>2 = (0000\ 0000\ 0010\ 0000_2)>>2 = 0000\ 0000\ 0000\ 1000_2$. Concatenamos os campos:

$$\underset{\text{opcode}\quad\text{rs}\quad\text{rt}\quad\text{valor imediato}}{\text{bne \$s0, \$t3, loop} = \left|\ 0001\ 01\ \right|\ 10\ 000\ \left|\ 0\ 1011\ \right|\ 0000\ 0000\ 0000\ 1000\ \left|\right.} = \quad 0x160B\ 0008$$

Questão 02 (1,5 ponto, 10 min.) – Represente o número decimal $-117{,}625$ em ponto flutuante, precisão simples. Apresente e comente os passos realizados.

Tomamos o número decimal sem sinal e convertemos para binário, no formato em ponto fixo, usando o método da divisão longa para a parte inteira do número e o método da multiplicação longa para a parte fracionária do número:

$$\begin{aligned} 117{,}625 &= 1110101,101 \\ &= 1110101,101 \cdot 2^0 \\ &= 1,\underbrace{110101101}_{\text{fração}} \cdot 2^{6 \leftarrow \text{ expoente}} \end{aligned}$$

Temos os seguintes campos para o número:

Sinal	1	O número é negativo.
Expoente polarizado	6+127 = 133 = 10000101_2	Somamos o peso 127 ao expoente.
Fração	$11010110100000000000000_2$	A fração deve ter 23 bits.

Concatenamos os campos:

$$\text{-}117{,}625 = \underbrace{\boxed{1}\,\boxed{100\ 0010\ 1}\,\boxed{110\ 1011\ 0100\ 0000\ 0000\ 0000}}_{\text{S}\quad\text{EP}\qquad\qquad\text{f}} = \text{0xC2EB4000}$$

Questão 03 (2,0 pontos, 20 min.) – Escreva um procedimento $\mathrm{P(x)}$, em linguagem de montagem para o processador **MIPS**, para calcular a função 1. O argumento x é passado para a função $\mathrm{P(x)}$ pelo registrador $\mathrm{\$f12}$. O resultado é retornado pelo registrador $\mathrm{\$f0}$. Os números x e o retorno de $\mathrm{P(x)}$ são representados em ponto flutuante, precisão dupla. Neste problema considere que não é necessário restaurar nenhum registrador. Não precisa usar a pilha neste problema.

$$P(x) = \begin{cases} -3,0 & \text{se } x \leqslant 0,0 \\ 3,0 \cdot x & \text{se } x > 0,0 \end{cases} \tag{1}$$

Segue uma solução:

```
1  .text
2  P:              # procedimento P
3                  sub.d    $f4 , $f4 , $f4          # $f4 = 0, criamos a constante 0
4                  li       $t0 , 3                  # $t0 = 3, criamos a constante 3,0
5                  mtc1     $t0 , $f0                # $f0 = palavra 3
6                  cvt.d.w  $f0 , $f0                # $f0 = 3,0 (ponto flutuante)
7                  c.le.d   $f12 , $f4               # flag 0 = 1 se x <= 0
8                  bc1t     x_menor_igual_0          # se x <= 0 desvie para
       x_menor_igual_0
9  x_maior_1:
10                 mul.d    $f0 , $f0 , $f12         # $f0 <- 3,0*x, calculamos P(x) =
       3,0 * x
11                 j fim_P                           # desvie para o fim do procedimento
12 x_menor_igual_1:
13                 neg.d    $f0 , $f0                # $f0 <- -3,0, calculamos P(x) =
       -3,0
14 fim_P:
15                 jr $ra                            # retorna ao procedimento chamador
```

Listagem 1: Operação de atribuição com várias variáveis

Questão 04 (3,0 pontos, 20 min.) – Seja a instrução lw \$t1, 24(\$t0) (lw rt, imm(rs), opcode 0x23) no endereço 0x0040 001C. **(A)** Converta a instrução para linguagem de máquina, apresentando os campos. **(B)** Explique detalhadamente com esta instrução é executada pelo processador monociclo (use as tabelas 1 e 2 e o diagrama de blocos do processador na figura 3). Apresente na figura os sinais de controle. Escreva um texto explicando como a instrução é executada. **(C)** Quais os valores na saída dos somadores e da UAL? Considere que o registrador \$i possui o valor i. Considere que toda a memória de dados é preenchida com zeros.

(A)

$$\text{lw \$t1, 24(\$t0)} = \underbrace{\boxed{1000\ 11}}_{\text{opcode}}\ \underbrace{\boxed{01\ 000}}_{\text{rs}}\ \underbrace{\boxed{0\ 1001}}_{\text{rt}}\ \underbrace{\boxed{0000\ 0000\ 0001\ 1000}}_{\text{imediato = 0x18 = 24}} = \text{0x8D09 0018}$$

(B) Na figura 1, apresentamos os sinais no processador monociclo, na execução da instrução lw \$t1, 24(\$t0).

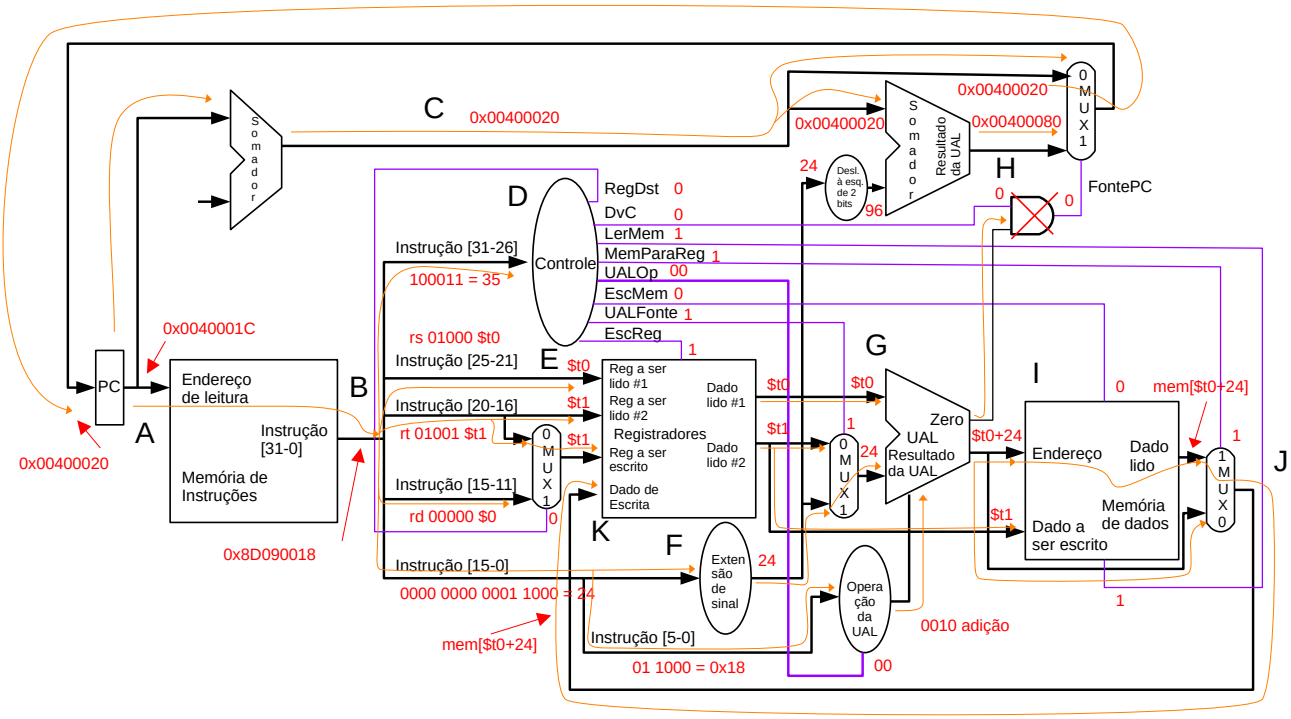

Figura 1: Diagrama do caminho de dados e de controle, apresentando os sinais na execução da instrução lw \$t1, 24(\$t0).

A seguir descrevemos o fluxo dos sinais no processador monociclo, na execução da instrução lw \$t1, 24(\$t0):

(A) Na borda de subida do sinal de relógio, o registrador PC é carregado com o endereço da próxima instrução que será executada pelo processador monociclo. Neste problema, PC é carregado com o endereço 0x0040 001C. Este endereço vai até o barramento de endereços da memória de instruções e a entrada de um somador.

(B) Na saída da memória de instruções temos a instrução 0x8D09 0018. Esta é a instrução lw \$t1, 24(\$t0). Os vários campos desta instrução são separados: o campo imediato da instrução jump (instrução[25-0]), opcode (instrução[31-26]), registrador rs (instrução[25-21]), registrador rt (instrução[20-16]), registrador rd (instrução[15-11]) e campo imediato (instrução[15-0]).

(C) Adicionamos 4 a PC em um somador. Na sua saída temos PC+4 = 0x0040 0020. O endereço de desvio incondicional é calculado. Os 26 bits menos significativos da

instrução (10 0000 1011 0000 0000 0000 1000) são deslocados de dois bits para a esquerda (1000 0010 1100 0000 0000 0010 0000 = 0x82C 0020), o que equivale a uma multiplicação por 4. Este sinal possui agora 28 bits. Os 4 bits de PC+4 = 0000 são concatenados com estes 28 bits, formando o endereço de desvio incondicional (0000 1000 0010 1100 0000 0000 0010 0000 = 0x082C_0020).

(D) O campo opcode (instrução[31-26]) é decodificado pela unidade de controle. Sinais de controle são gerados. Estes sinais estão apresentados na tabela 1: RegDst = 1, Jump = 0, UALFonte = 0, MemParaReg = 0, EscReg = 1, LerMem = 0, EscMem = 0, DvC = 0, UALOp = 10.

(E) No banco de registradores chegam os endereços dos registradores rs (instrução[25-21]) = $t0 e rt (instrução[20-16]) = $t1, na entrada dos endereços dos registradores que serão lidos. O conteúdo destes registradores aparecem nas saídas do banco de registradores "dado lido #1" e "dado lido #2". O banco de registradores recebe o sinal de controle EscReg = 1. Na próxima borda de subida do sinal de relógio, o valor na entrada "dado de escrita" será escrito no registrador "reg a ser escrito". Neste problema, o endereço do registrador é a entrada 0 do MUX. Nesta entrada temos o registrador $t1. O sinal de controle RegDst = 0 seleciona esta entrada deste MUX.

(F) Ocorre a extensão do sinal no campo imediato (instrução[15-0]) = 24. Este sinal passa de 16 para 32 bits. O campo funct(instrução[5-0]) = 0x18 é extraído do campo imediato e ligado a entrada do circuito que gera o código de operação da UAL. Este circuito também recebe o sinal de controle UALOp = 00. Usando a tabela 2, vemos que é gerado o sinal de controle para a UAL 0010, solicitando uma adição para a ULA.

(G) Em uma das entradas da ULA temos um MUX que recebe o sinal de controle UALFonte = 1. É selecionado o valor imediato estendido 24 para uma das entradas da ULA. Esta recebe em sua outra entrada o conteúdo do registrador rs = $t0. O sinal de controle 0010 do controle de operação da ULA faz com que as entradas sejam somadas. Na saída da ULA temos o valor $t0+24.

(H) Um segundo somador calcula o endereço de desvio de instruções condicionais beq. O valor imediato = 24 é deslocado por 2 bits para a esquerda, o que equivale a uma multiplicação por 4. O valor imediato multiplicado por 4, 96, é uma das entradas do somador. A outra entrada é o endereço da próxima instrução que será executada, PC+4 = 0x0040 0020. Na saída do somador temos o endereço 0x0040 0080. Este é o endereço de desvio de uma instrução beq. Na saída do somador temos um multiplexador com um sinal de controle FontePC. Uma porta **AND** gera o sinal FontePC, que é igual a 0, porque uma de suas entradas tem o sinal de controle DvC igual a 0. Neste caso, a saída da porta, o sinal FontePC será sempre 0, independente do valor do sinal zero vindo da ULA. Com FontePC = 0 selecionamos o endereço PC+4 = 0x0040 0020. Este endereço é a entrada 0 do MUX com o sinal de controle jump. Como este sinal de controle é 0, a saída deste multiplexador será o valor de PC+4, o endereço da próxima instrução e não o endereço de desvio incondicional. O endereço de PC+4 vai para a entrada do registrador PC.

(I) A memória de dados recebe a soma $t0+24 no barramento de endereços e o conteúdo do registrador $t1 na entrada "dado a ser escrito". Esta memória recebe os sinais de controle EscMem = 0 e LerMem = 1. Ocorre a leitura da memória de dados. O conteúdo do endereço de $t0+24 aparece na saída "dado lido" da memória de dados.

(J) O multiplexador com o sinal MemParaReg = 1 tem na sua saída o conteúdo do endereço $t0+24, da memória de dados. Este sinal é levado até a entrada "dado de escrita" do banco de registradores.

(K) O sinal EscReg = 1 do banco de registradores permite a escrita neste banco. Na borda de subida do sinal de relógio a entrada "dado de escrita" é escrito no registrador de endereço "reg a ser escrito": escrevemos no registrador $t1 o conteúdo do endereço

$\$t0+24$, da memória de dados. Nesta borda de subida o registrador PC também é atualizado. Gravamos neste registrador o endereço da próxima instrução que será executada pelo processador, o endereço $0x0040\ 0020$.

(C)

Saída do somador 1: PC+4 = $0x0040\ 001C$ + 4 = $0x0040\ 0020$

Saída do somador 2: (PC+4) + 4 · 24 = $0x0040\ 0080$

Saída da UAL: rs + imm = 8 + 24 = 32

Questão 05 (2,0 pontos, 20 min.) – Considere os seguintes estágios individuais de um caminho de dados:

	IF (ps)	ID (ps)	EX (ps)	MEM (ps)	WB (ps)
a.	200	250	300	350	100

(A) Qual é o período mínimo do sinal de relógio para um processador monociclo e com pipeline?

Para um processador monociclo, somamos os tempos de latência individuais: $T_{CLK} = 1200\,\text{ps}$. No processador com pipeline, usamos o maior tempo de latência dos estágios individuais: $T_{CLK} = 350\,\text{ps}$.

(B) Qual a latência esperada para a execução de uma instrução sw no processador monociclo e para o processador com pipeline.

Em um processador monociclo, qualquer instrução é executada em um ciclo de relógio. A instrução sw será executada em $1200\,\text{ps}$. Em um processador com pipeline, a instrução é processada sequencialmente nos estágios individuais. Este pipeline possui 5 estágios, logo, serão necessários 5 ciclos de relógio: $5 \cdot 350\,\text{ps} = 1750\,\text{ps}$.

(C) Quanto mais rápido é o processador com pipeline em relação ao processador monociclo na execução de um programa? Considere que não há *hazards* no processador com pipeline.

No processador monociclo, cada instrução do programa será executada em um ciclo de relógio: $1200\,\text{ps}$. No processador com pipeline, considerando que não haja hazards, cada instrução é executada em um ciclo de relógio: $350\,\text{ps}$. No processador com pipeline, um programa será executado ${}^{1200\,\text{ps}}/_{350\,\text{ps}} \approx 3,4$ vezes mais rápido, em relação ao processador monociclo.

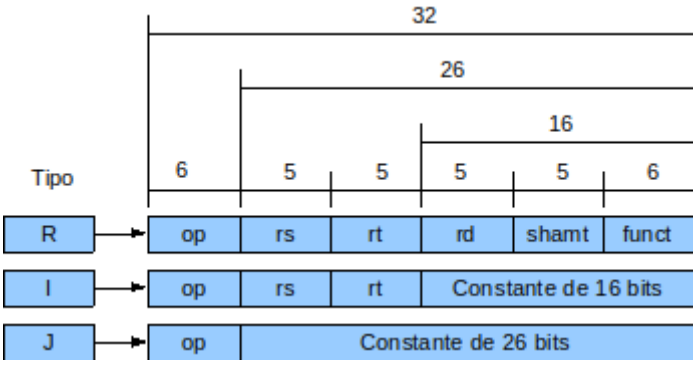

Figura 2: Ilustração dos campos das instruções R, I e J de um processador **MIPS**.

Tabela 1: Instruções e valores dos sinais na unidade de controle do processador mono-ciclo.

Controle	Sinal	Formato R (0)	lw (35)	sw (43)	beq (4)	j (2)
Entradas	OP5	0	1	1	0	0
	OP4	0	0	0	0	0
	OP3	0	0	1	0	0
	OP2	0	0	0	1	0
	OP1	0	1	1	0	1
	OP0	0	1	1	0	0
Saídas	RegDst	1	0	X	X	X
	Jump	0	0	0	0	1
	UALFonte	0	1	1	0	1
	MemParaReg	0	1	X	X	X
	EscReg	1	1	0	0	X
	LerMem	0	1	0	0	0
	EscMem	0	0	1	0	0
	DvC	0	0	0	1	0
	UALOp1	1	0	0	0	X
	UALOp0	0	0	0	1	X

Tabela 2: Operação da ULA para a combinação de UALOp e o campo de função.

UALOP		Campo de Função							
UALOp1	UALOp0	F5	F4	F3	F2	F1	F0	Operação da ULA	
0	0	X	X	X	X	X	X	0010	soma
X	1	X	X	X	X	X	X	0110	subtração
1	X	X	X	0	0	0	0	0010	soma
1	X	X	X	0	0	1	0	0110	subtração
1	X	X	X	0	1	0	0	0000	and
1	X	X	X	0	1	0	1	0001	or
1	X	X	X	1	0	1	0	0111	slt

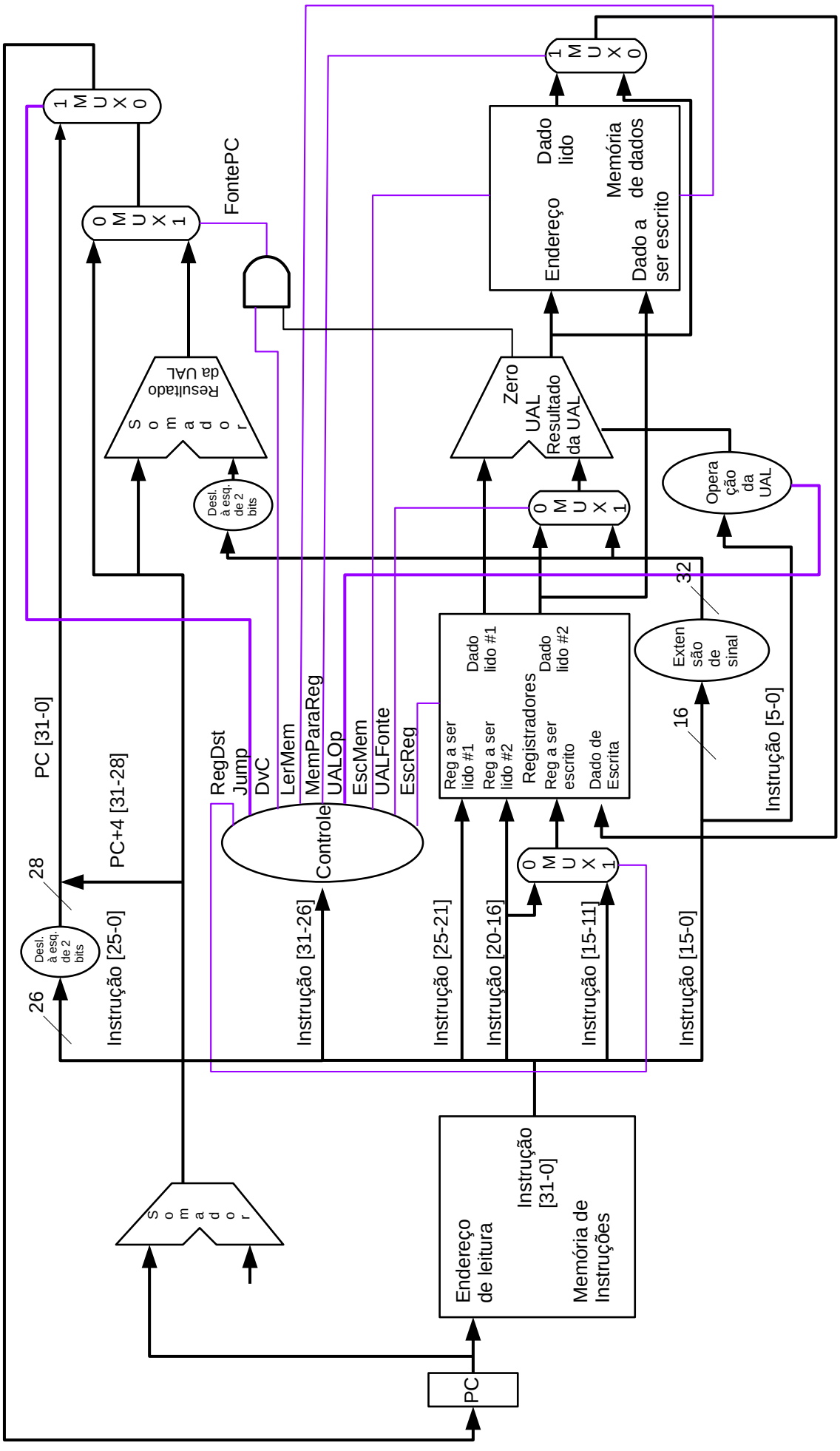

Figura 3: Diagrama de blocos do processador monociclo