Arquitetura de Computadores

Turma LI21N/LT21N

AULA 8
UNIDADES DE CONTROLO MICROPROGRAMADAS

TÓPICOS DE REVISÃO

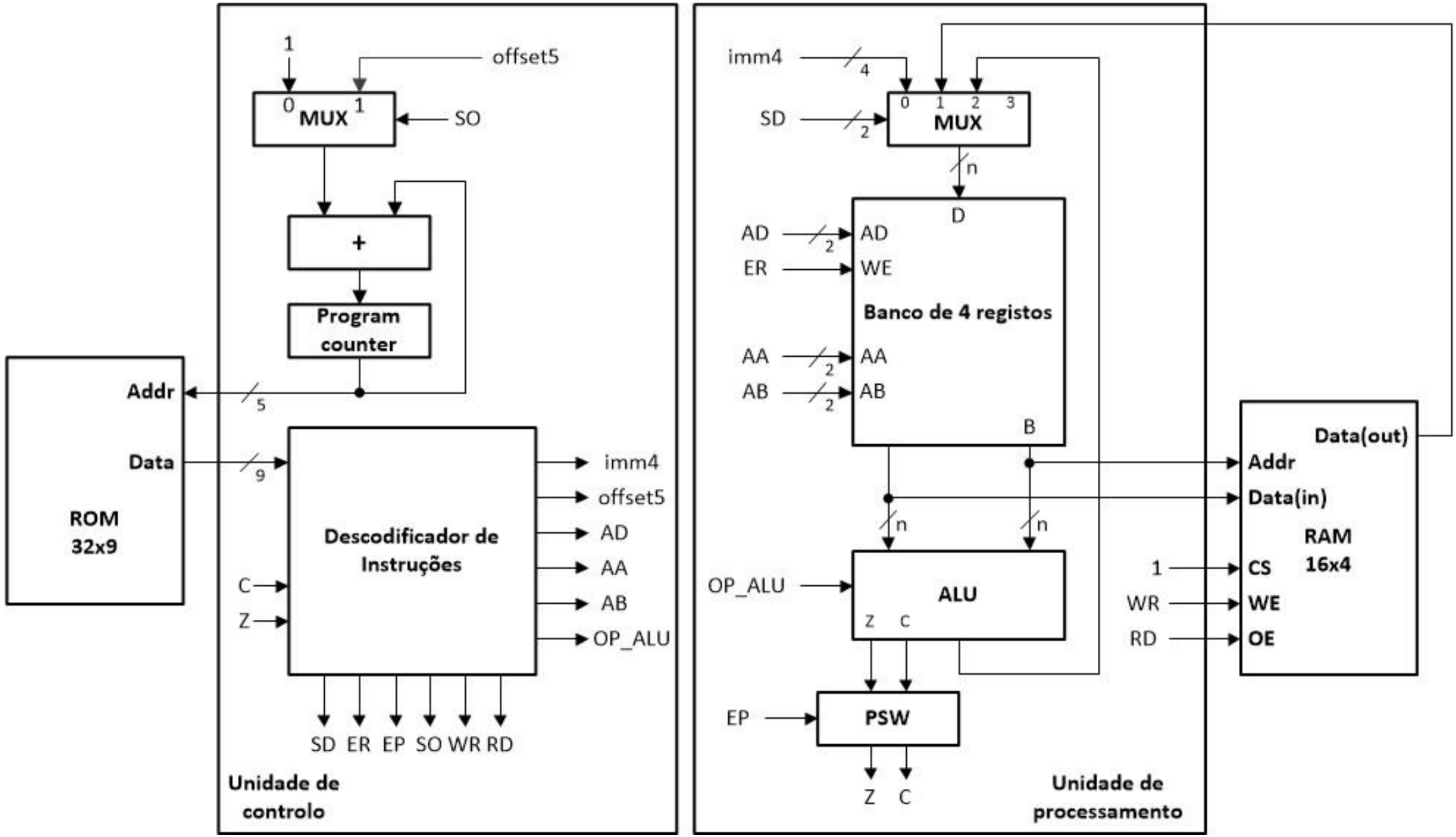
- *Definição de conjunto de instruções*
  - *Noção de mnemónica*
  - *Operandos de uma instrução*
- *Tipos de instrução*
  - *Transferência de dados*
  - *Processamento de dados*
  - *Controlo de fluxo de execução*
- *Definição / caracterização do ISA (Instruction Set Arquitecture)*

Ano Lectivo 2019/2020

2º Semestre

Prof. Jorge Fonseca

DIAGRAMA DE BLOCOS DO CORE3



ISA	OPCODE			AD		AA		AB	
	8	7	6	5	4	3	2	1	0
Instruções de Transferência									
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
Instruções de processamento de dados									
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
Instruções de controlo de fluxo									
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				

Numa instrução é preciso codificar:

- OPCODE: código único que distingue uma instrução de outra
- Operandos: pode ser operação da ALU, constante, Offset e/ou registos
- a codificação da operação da ALU pode pertencer ao OPCODE

Características:

- A constante para expressar um literal do tipo inteiro fica limitada a 4 bits, portanto ao valor 15.
- O offset é codificado com 5 bits e representa um inteiro com sinal ficando limitado a um salto baseado na instrução corrente para +15 ou -16 instruções.

Exercício1 : Escrever em Código Máquina um programa para determinar o valor de A + B – C. Considere o operando A na posição de memória 0H, o operando B na posição de memória 1H e o operando C na posição de memória 2H. O resultado da função deve ser colocado na posição de memória 3H.

	Code Address	OPCODE			AD		AA		AB		HEX
		8	7	6	5	4	3	2	1	0	
MOV R1,0	00	0	0	0	0	1	0	0	0	0	010
LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
MOV R1,1	02	0	0	0	0	1	0	0	0	1	011
LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
ADD R2, R0, R2	04	1	0	0	1	0	0	0	1	0	122
MOV R1,2	05	0	0	0	0	1	0	0	1	0	012
LD R0, [R1]	06	0	0	1	0	0	-	-	0	1	041
SUB R2, R2, R0	07	0	1	1	1	0	1	0	0	0	0E8
MOV R1,3	08	0	0	0	0	1	0	0	1	1	013
ST R2, [R1]	09	0	1	0	-	-	1	0	0	1	089
B 0	10	1	1	0	-	0	0	0	0	0	180

RAM	
ADDR	
0	A
1	B
2	C
3	R

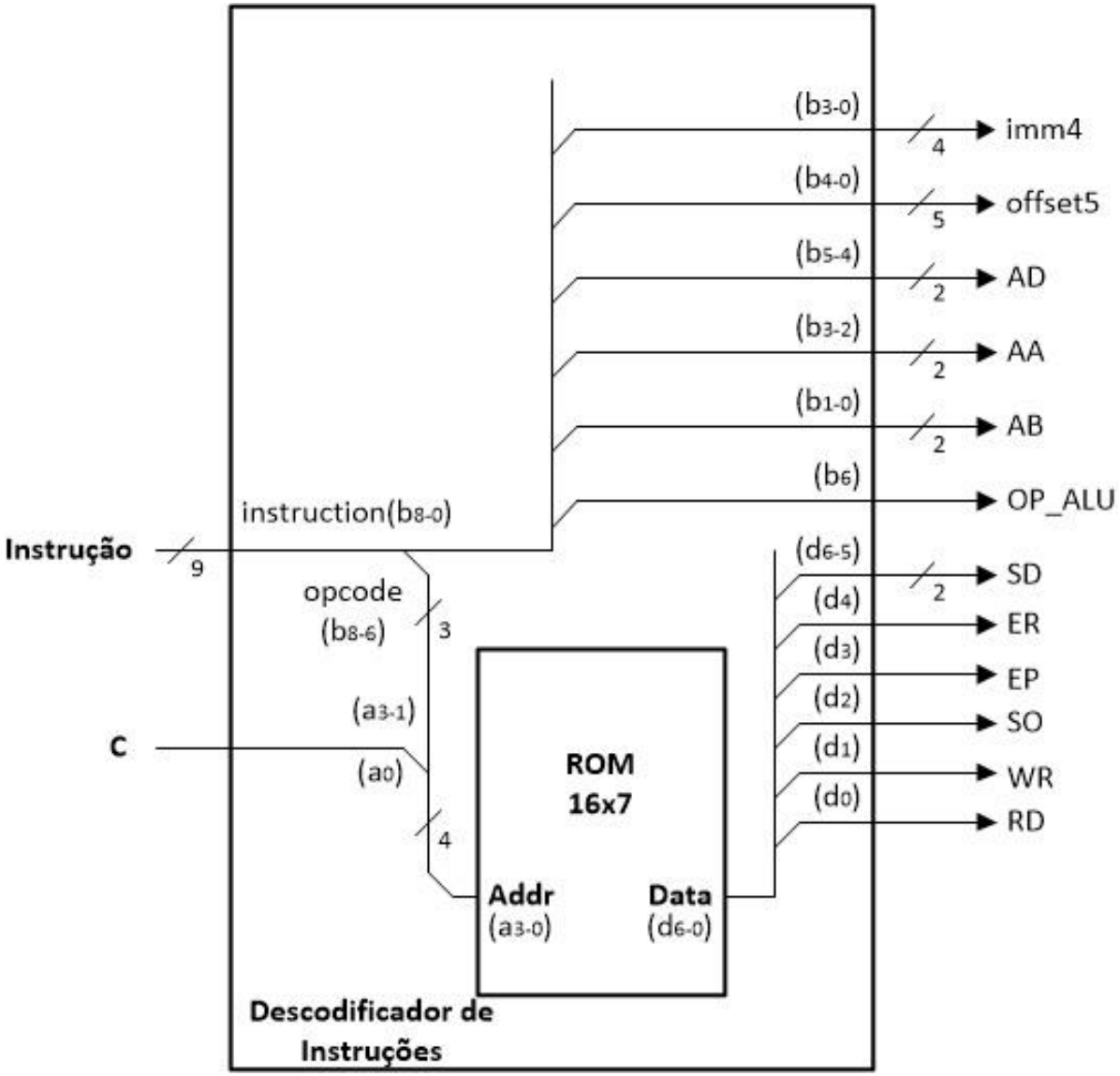


Imagem 1 – Implementação do módulo descodificador

Do ISA retira-se o controlo dos seguintes sinais:

- código da operação: B8..6
- operação da ALU: B6
- imm4: B3..0
- offset5: B4..0
- AA: B3..2
- AB: B1..0
- AD: B5..4

Estrutura interna do Core3

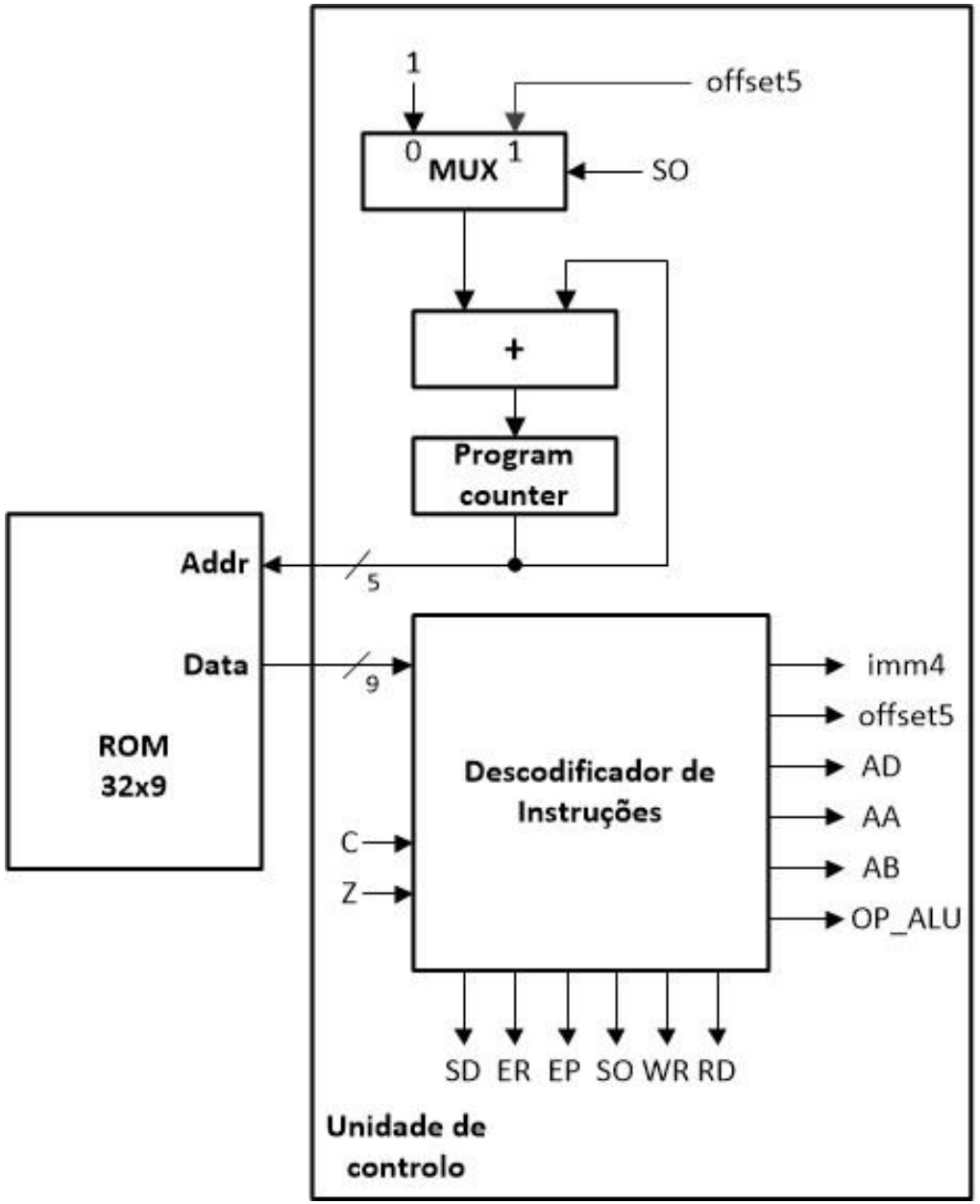
- Um programa é uma sequência de Códigos Máquina em memória para serem processados pelo CPU.
- Aspeto característico de arquitetura RISC: as instruções ocupam todas o mesmo espaço em memória.
- Processador de ciclo único → Arquitetura Harvard: separa a memória de código da memória de dados, ou seja, espaço de endereçamento distintos.

Instruction Decoder	OPCODE			C	SD1..0		ER	EP	SnA	SO	WR	RD	HEX	PRG
	3	2	1	0	7	6	5	4	3	2	1	0		
mov rx, immediate4	0	0	0	-	0	0	1	0	-	0	0	-	020	2*020
ld rx, [ry]	0	0	1	-	0	1	1	0	-	0	0	1	061	2*061
st rx, [ry]	0	1	0	-	-	-	0	0	-	0	1	0	002	2*002
sub rx, ry, rz	0	1	1	-	1	0	1	1	1	0	0	-	0B8	2*0B8
add rx, ry, rz	1	0	0	-	1	0	1	1	0	0	0	-	0B0	2*0B0
bae offset5	1	0	1	0	-	-	0	0	-	1	0	-	004	004
bae offset5	1	0	1	1	-	-	0	0	-	0	0	-	000	000
b offset4	1	1	0	-	-	-	0	0	-	1	0	-	004	2*004
cmp rx, ry	1	1	1	-	-	-	0	1	1	0	0	-	018	2*018

Tabela 2 – Módulo descodificador

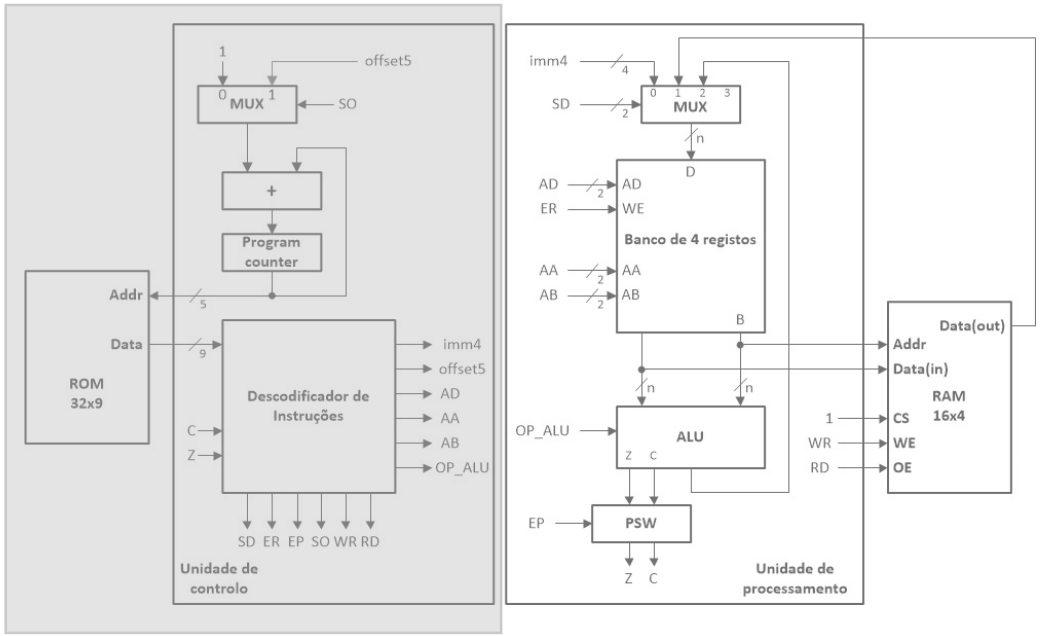
Descodificador de instruções:

- Produz os sinais de controlo da unidade de processamento baseado no OPCODE e *flags C e Z* e controla a evolução do programa
- Micro-código implementado tipicamente numa ROM

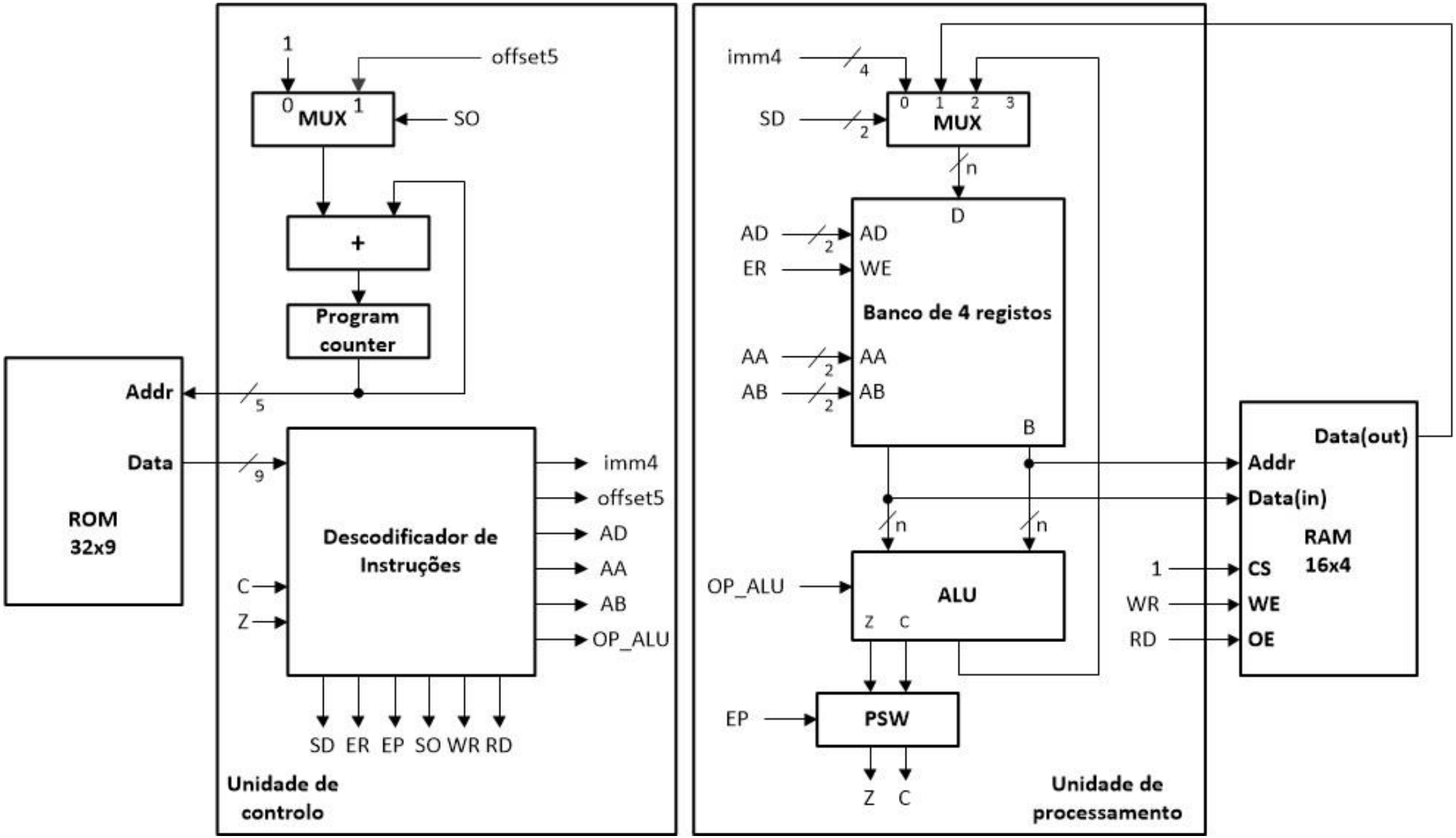


Aspectos relevantes:

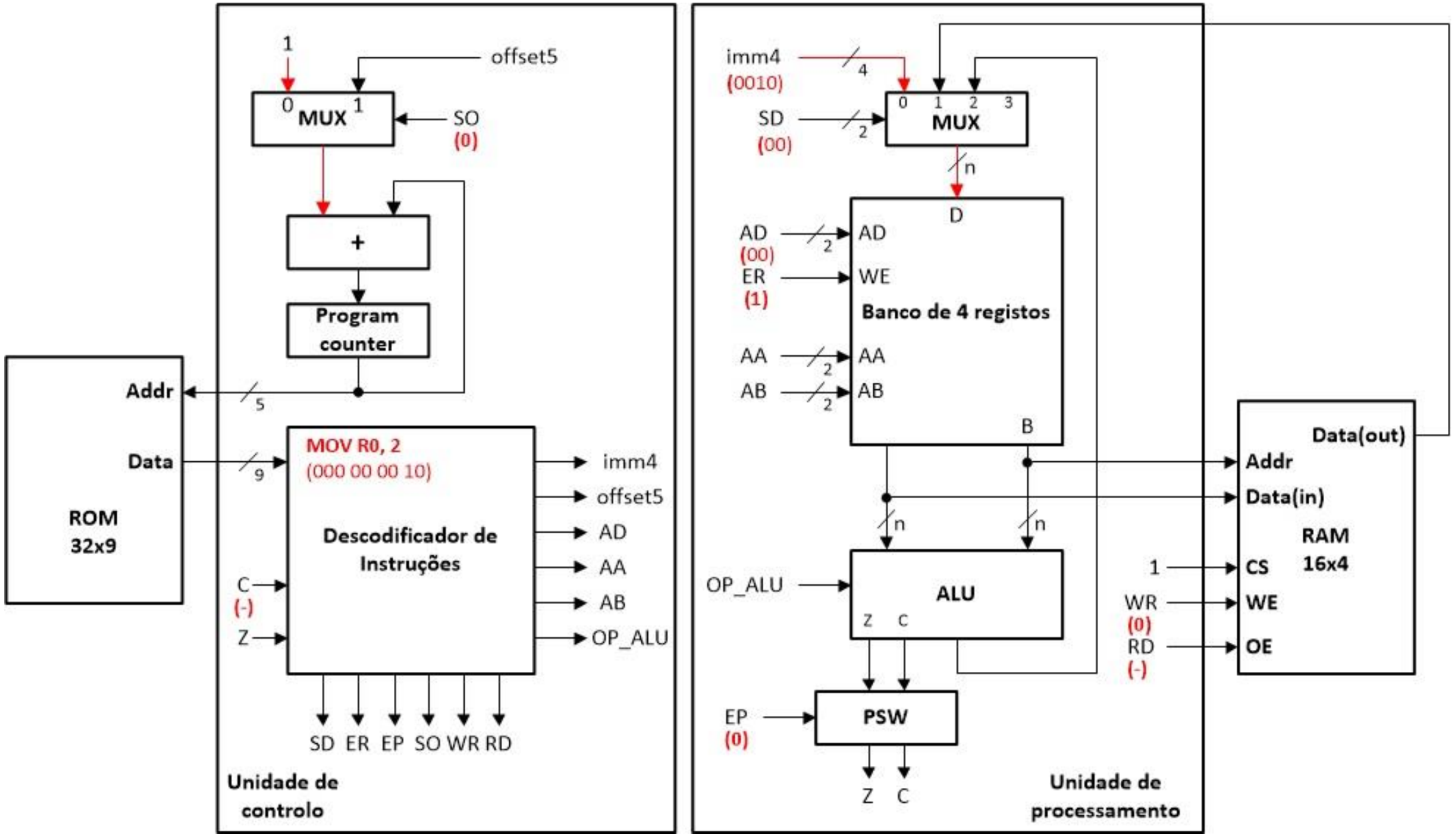
- Registo PC (Program Counter): retém a posição da instrução em execução.
- SO: controla o valor de incremento do PC.
- Clock da unidade de controlo em oposição de fase do clock da unidade de processamento.



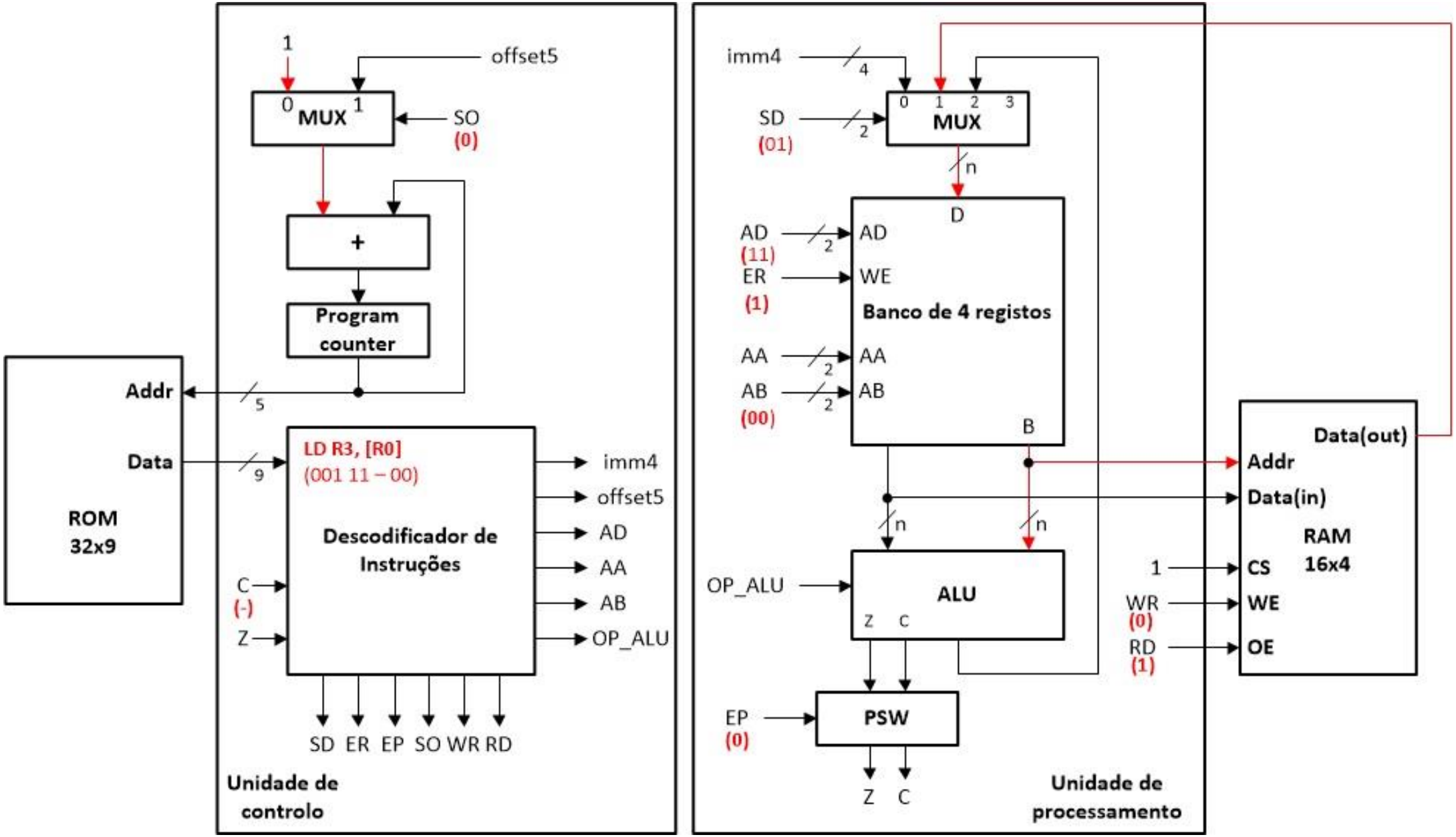
SISTEMA COMPLETO DO CORE3



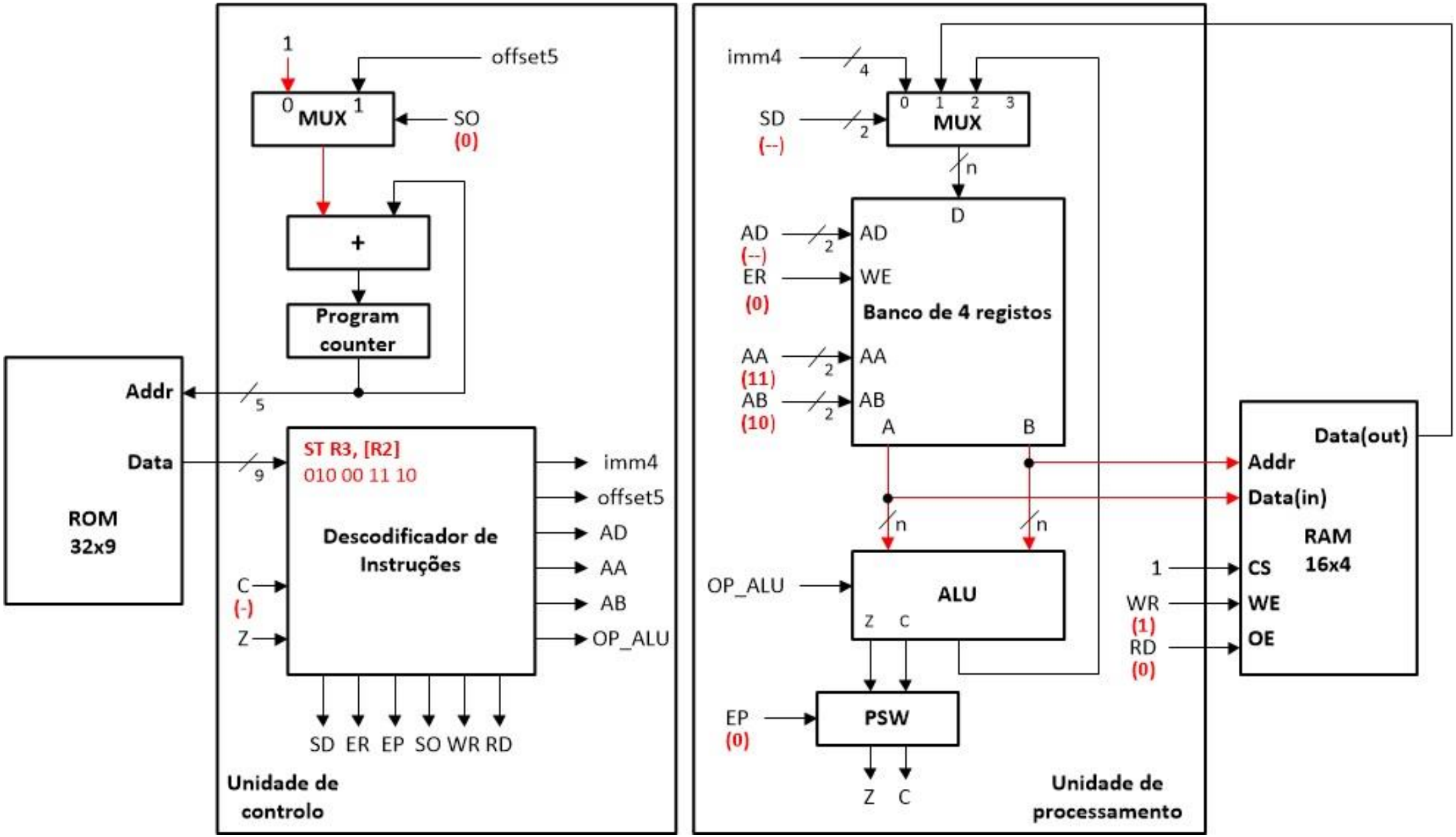
EXECUÇÃO DA INSTRUÇÃO MOV R2,0



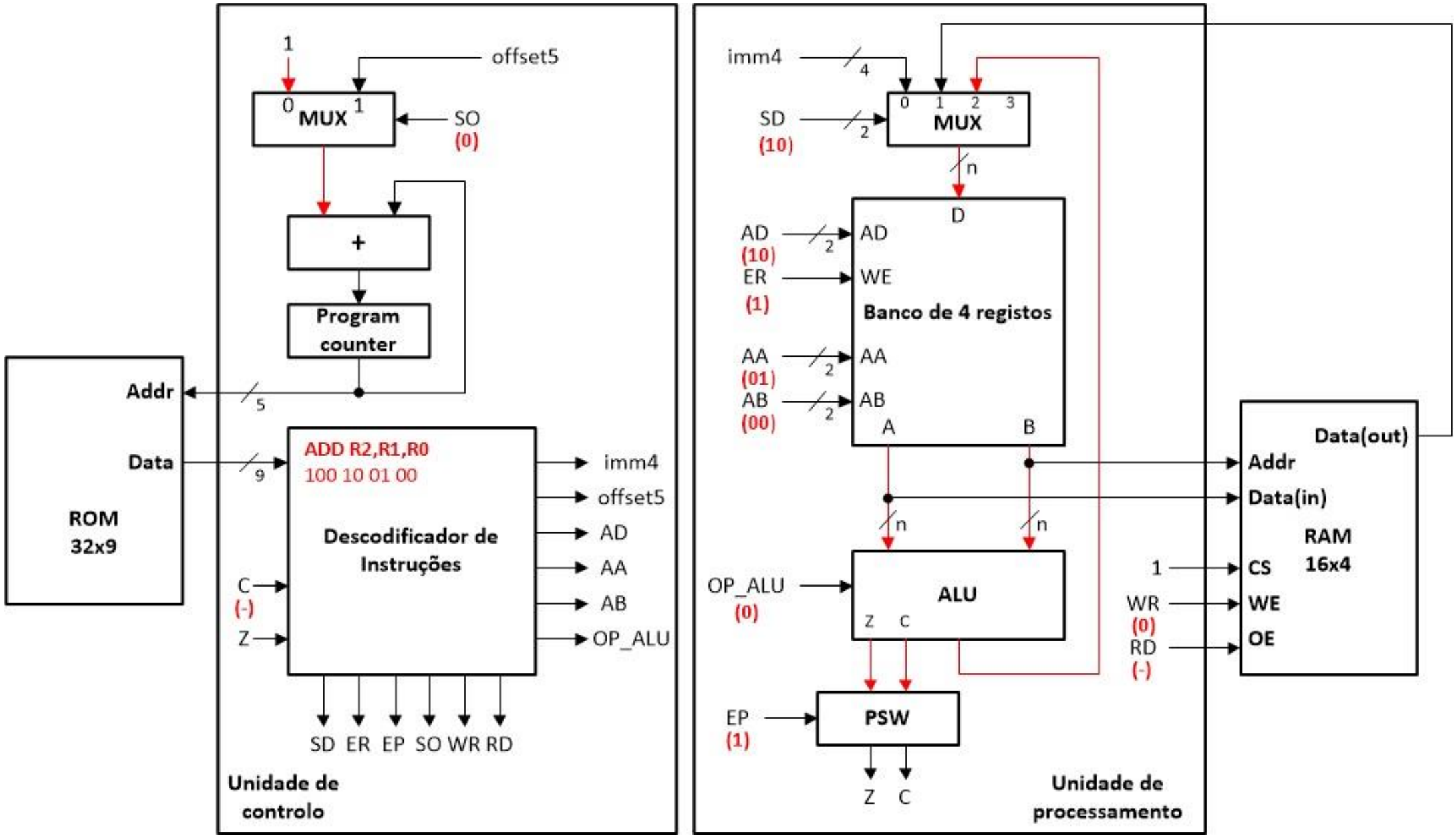
EXECUÇÃO DA INSTRUÇÃO LD R3,[R0]



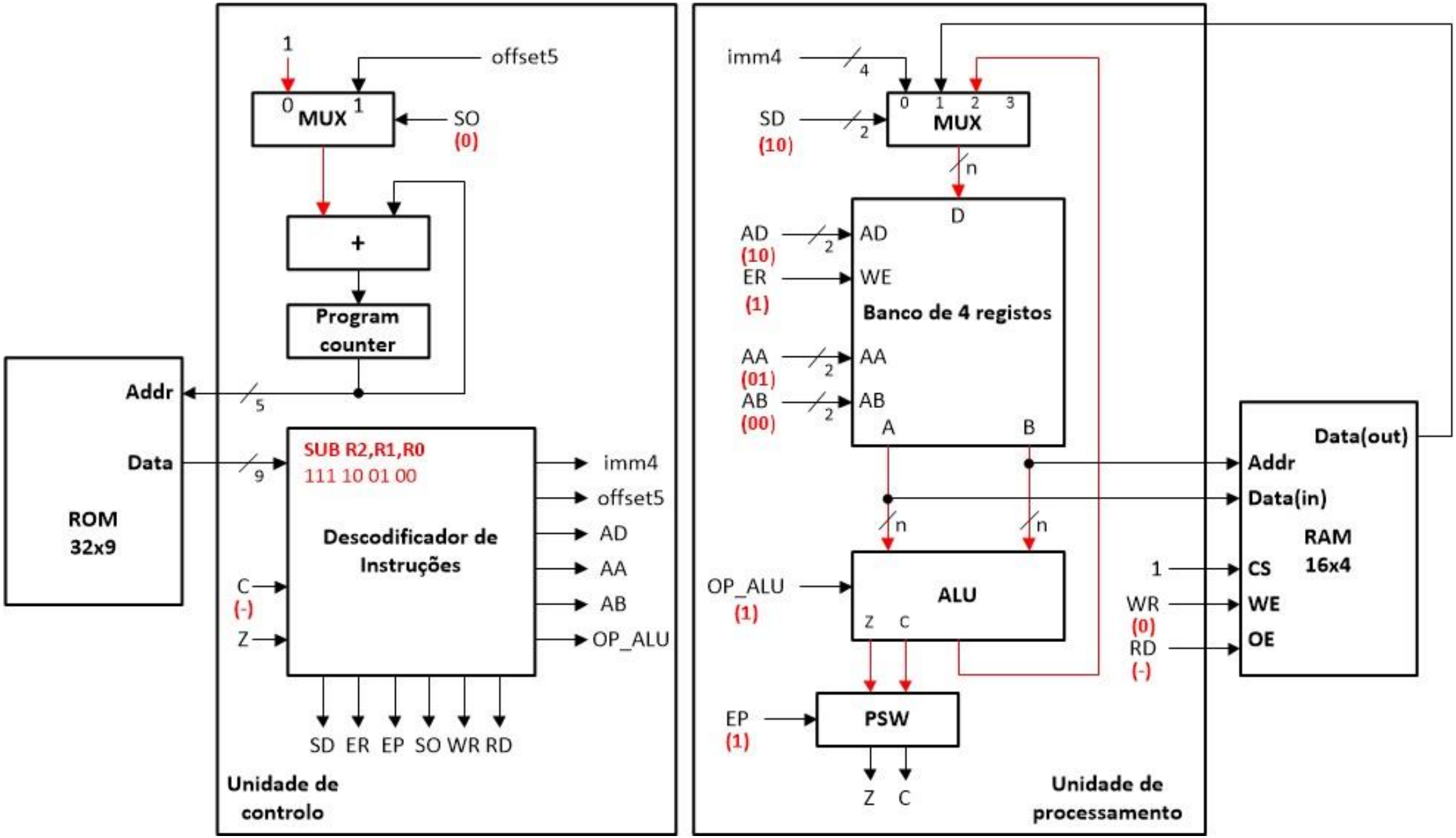
EXECUÇÃO DA INSTRUÇÃO ST R3,[R2]



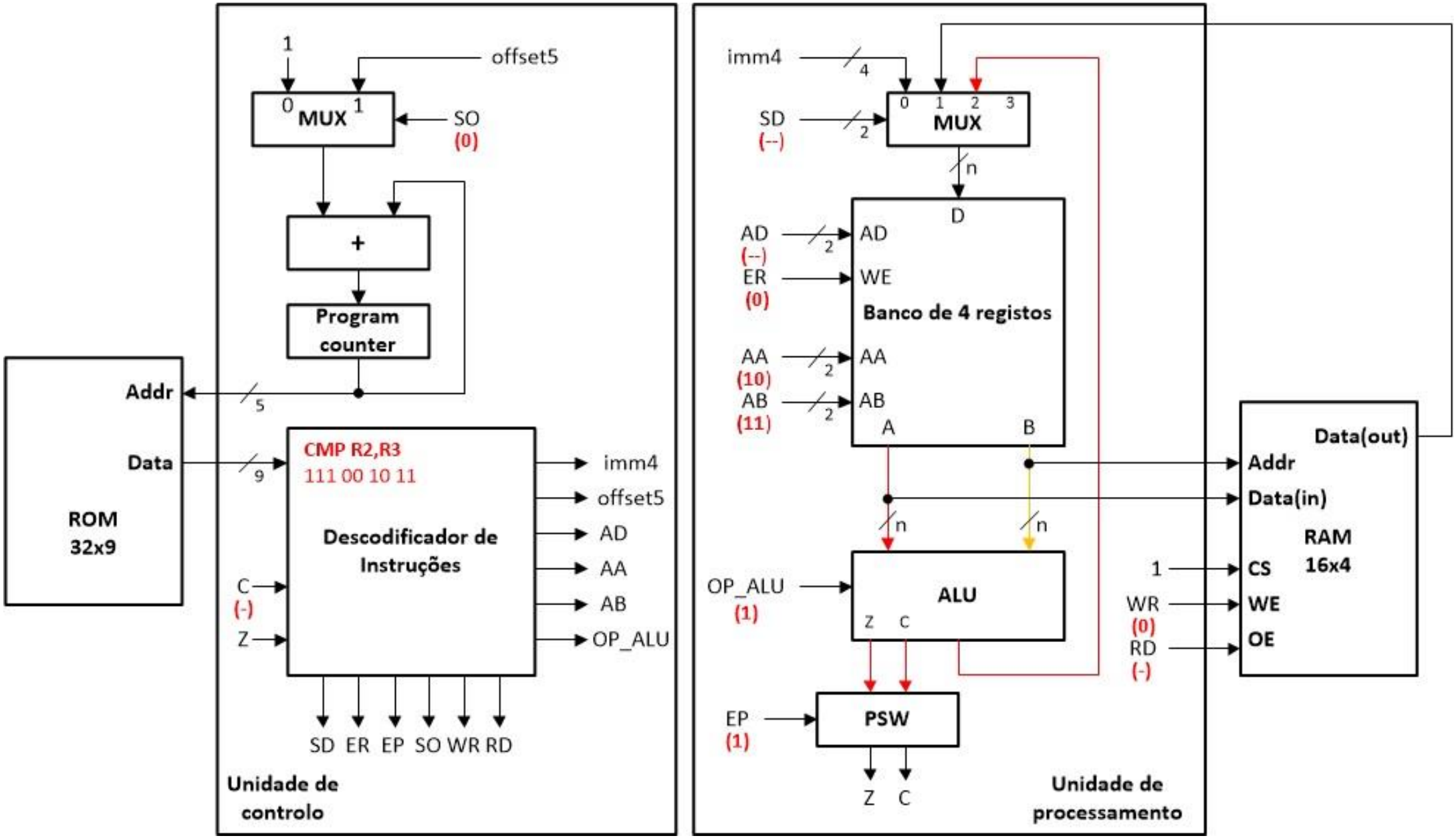
EXECUÇÃO DA INSTRUÇÃO ADD R2,R1,R0



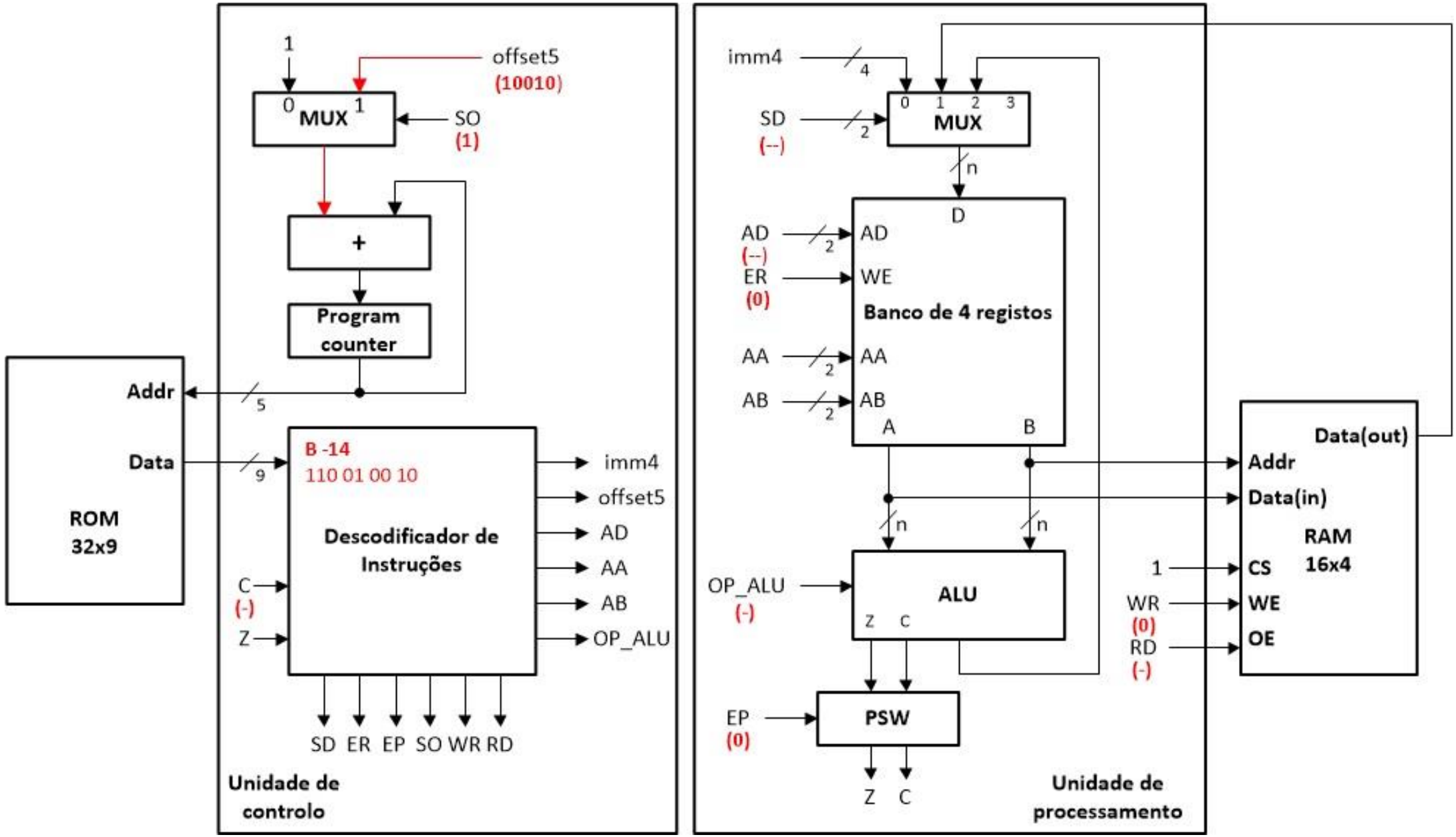
EXECUÇÃO DA INSTRUÇÃO SUB R2,R1,R0



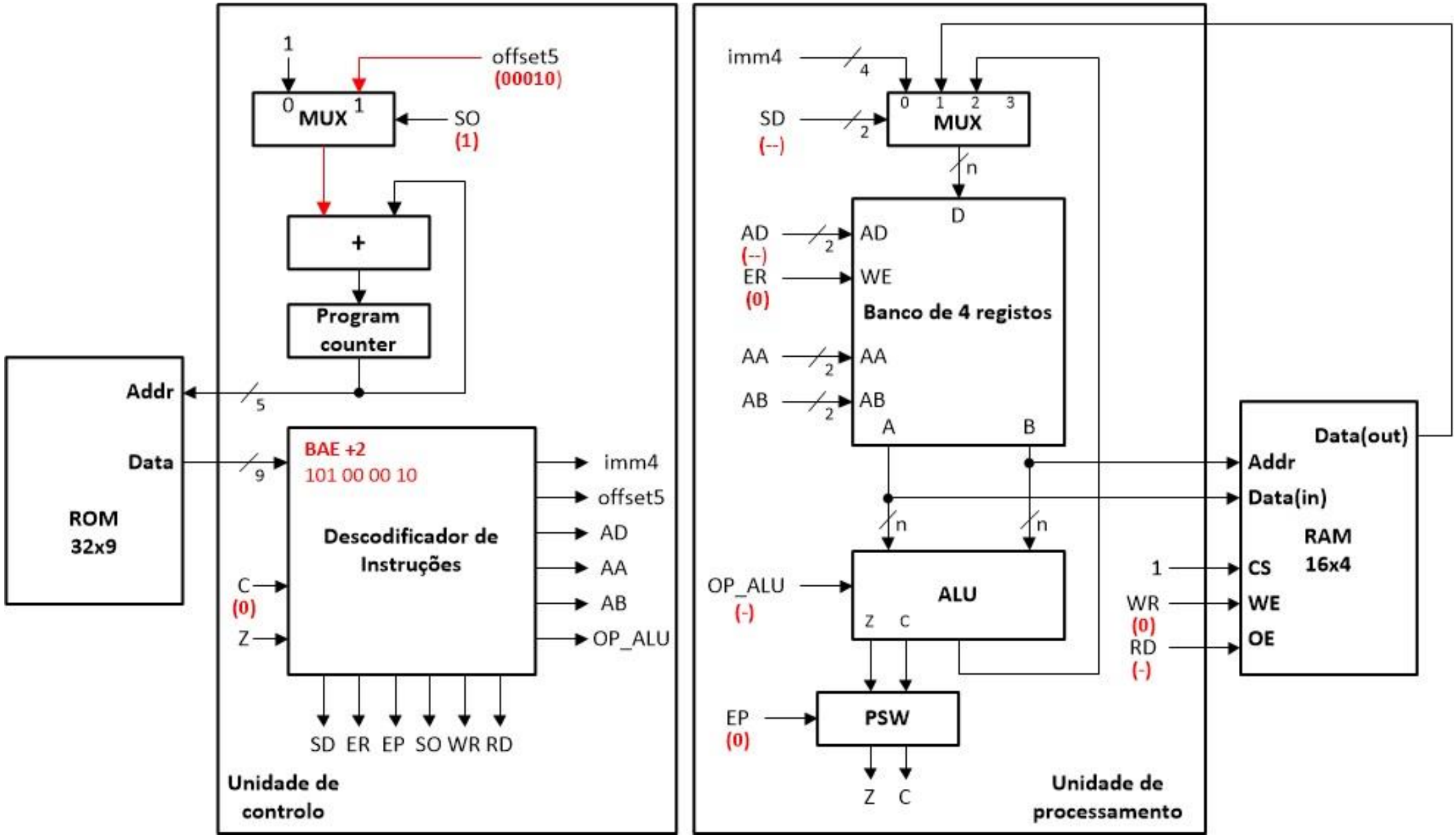
EXECUÇÃO DA INSTRUÇÃO CMP R2,R3



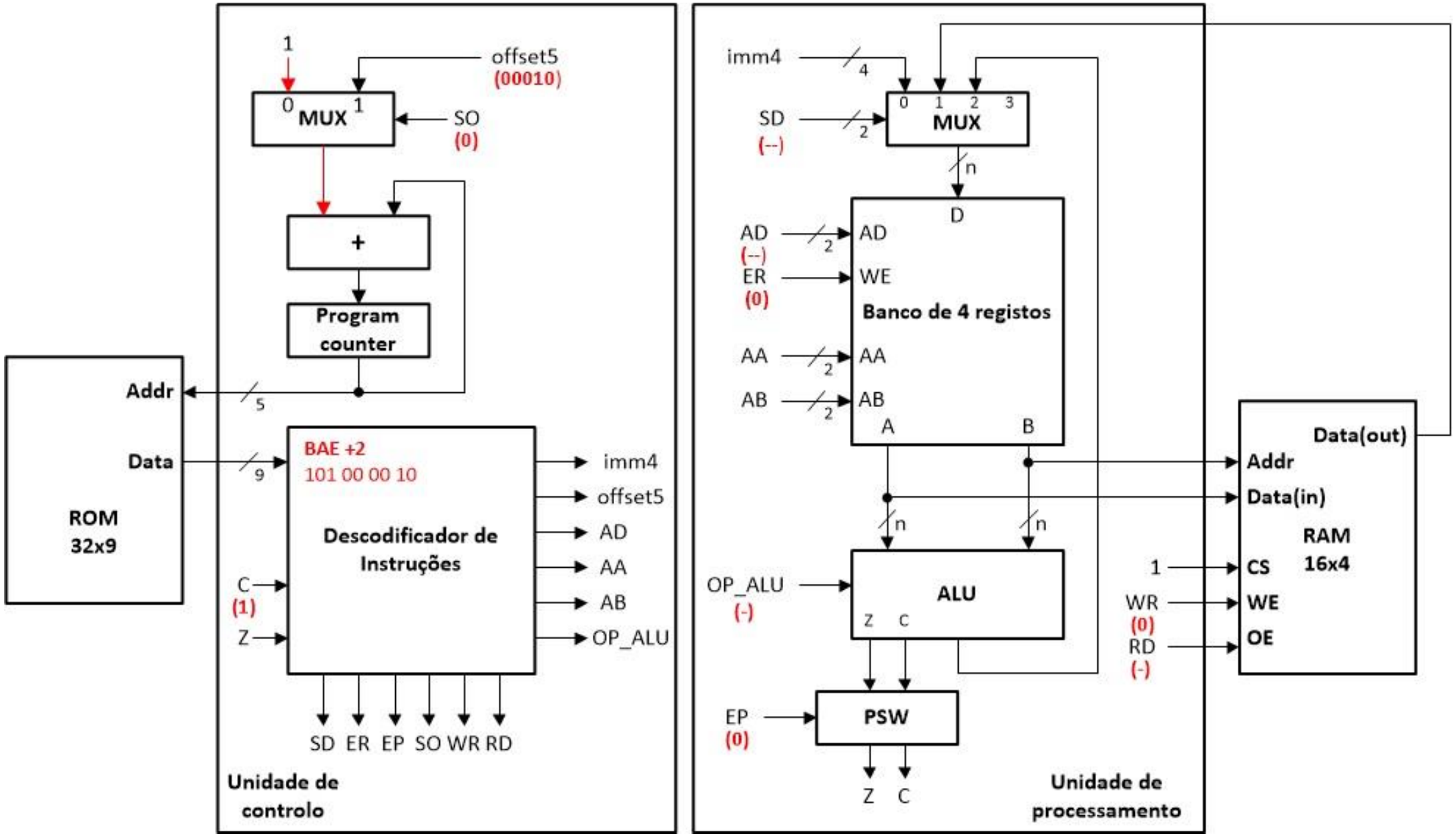
EXECUÇÃO DA INSTRUÇÃO B -14



EXECUÇÃO DA INSTRUÇÃO BAE +2 COM C=0



EXECUÇÃO DA INSTRUÇÃO BAE +2 COM C=1



PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE			AD		AA		AB		PRG (HEX)
			8	7	6	5	4	3	2	1	0	
	MOV R1, 0A	00										000
	LD R0, [R1]	01										000
	MOV R1, 0B	02										000
	LD R2, [R1]	03										000
	CMP R0, R2	04										000
	BAE +4	05										000
	MOV R1, 0C	06										000
	ST R2, [R1]	07										000
	B +3	08										000
	MOV R1, 0C	09										000
	ST R0, [R1]	0A										000
	B 0	0B										000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE			AD		AA		AB	
	8	7	6	5	4	3	2	1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01										041
	MOV R1, 0B	02										01B
	LD R2, [R1]	03										061
	CMP R0, R2	04										1C2
	BAE +4	05										143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02										01B
	LD R2, [R1]	03										061
	CMP R0, R2	04										1C2
	BAE +4	05										143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03										061
	CMP R0, R2	04										1C2
	BAE +4	05										143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04										1C2
	BAE +4	05										143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05										143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06										01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE			AD		AA		AB		PRG (HEX)
			8	7	6	5	4	3	2	1	0	
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07										089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE			AD		AA		AB	
	8	7	6	5	4	3	2	1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS - SOLUÇÃO

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07	0	1	0	-	-	1	0	0	1	089
	B +3	08										183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE			AD		AA		AB		PRG (HEX)
			8	7	6	5	4	3	2	1	0	
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07	0	1	0	-	-	1	0	0	1	089
	B +3	08	1	1	0	-	0	0	0	1	1	183
	MOV R1, 0C	09										01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE			AD		AA		AB	
	8	7	6	5	4	3	2	1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07	0	1	0	-	-	1	0	0	1	089
	B +3	08	1	1	0	-	0	0	0	1	1	183
	MOV R1, 0C	09	0	0	0	0	1	1	1	0	0	01C
	ST R0, [R1]	0A										081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07	0	1	0	-	-	1	0	0	1	089
	B +3	08	1	1	0	-	0	0	0	1	1	183
	MOV R1, 0C	09	0	0	0	0	1	1	1	0	0	01C
	ST R0, [R1]	0A	0	1	0	-	-	0	0	0	1	081
	B 0	0B										180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS - SOLUÇÃO

Exercício2 : Escrever em Código Máquina um programa para determinar o maior valor contido nas posições de memória de endereço 0AH e 0BH.

		Code Address	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0	PRG (HEX)
	MOV R1, 0A	00	0	0	0	0	1	1	0	1	0	01A
	LD R0, [R1]	01	0	0	1	0	0	-	-	0	1	041
	MOV R1, 0B	02	0	0	0	0	1	1	0	1	1	01B
	LD R2, [R1]	03	0	0	1	1	0	-	-	0	1	061
	CMP R0, R2	04	1	1	1	-	-	0	0	1	0	1C2
	BAE +4	05	1	0	1	-	0	0	0	1	1	143
	MOV R1, 0C	06	0	0	0	0	1	1	1	0	0	01C
	ST R2, [R1]	07	0	1	0	-	-	1	0	0	1	089
	B +3	08	1	1	0	-	0	0	0	1	1	183
	MOV R1, 0C	09	0	0	0	0	1	1	1	0	0	01C
	ST R0, [R1]	0A	0	1	0	-	-	0	0	0	1	081
	B 0	0B	1	1	0	-	0	0	0	0	0	180
												000
												000
												000
												000
												000

RAM	
ADDR	
0A	X
0B	Y
0C	M

Algoritmo

```
IF X >= Y {
   M = X
} ELSE {
   M = Y
}
```

ISA	OPCODE 8	7	6	AD 5	4	AA 3	2	AB 1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	

PROGRAMAS

Exercício3 : Escrever em Código Máquina um programa para determinar o maior valor contido no array x.

		Code Address	OPCODE			AD		AA		AB		PRG (HEX)
			8	7	6	5	4	3	2	1	0	
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000
												000

RAM	
ADDR	
0	i
1	maior
2	x[0]
3	x[1]
4	x[2]
5	x[3]
6	x[4]
7	x[5]

Algoritmo

```
nibble i, maior;
nibble x[6]; /* inteiros sem sinal */
maior=x[0];
for (i=1; i < 6; i++) {
   if (x[i] > maior) maior = x[i];
}
```

ISA	OPCODE			AD		AA		AB	
	8	7	6	5	4	3	2	1	0
mov rx, imm4	0	0	0	rx		imm4			
ld rx, [ry]	0	0	1	rx		-	-	ry	
st rx, [ry]	0	1	0	-	-	rx		ry	
sub rx, ry, rz	0	1	1	rx		ry		rz	
add rx, ry, rz	1	0	0	rx		ry		rz	
bae offset5	1	0	1	-	offset5				
b offset5	1	1	0	-	offset5				
cmp rx, ry	1	1	1	-	-	rx		ry	