Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Computação
027359 - Arquitetura e Organização de Computadores 1
Prof. Luciano Neris

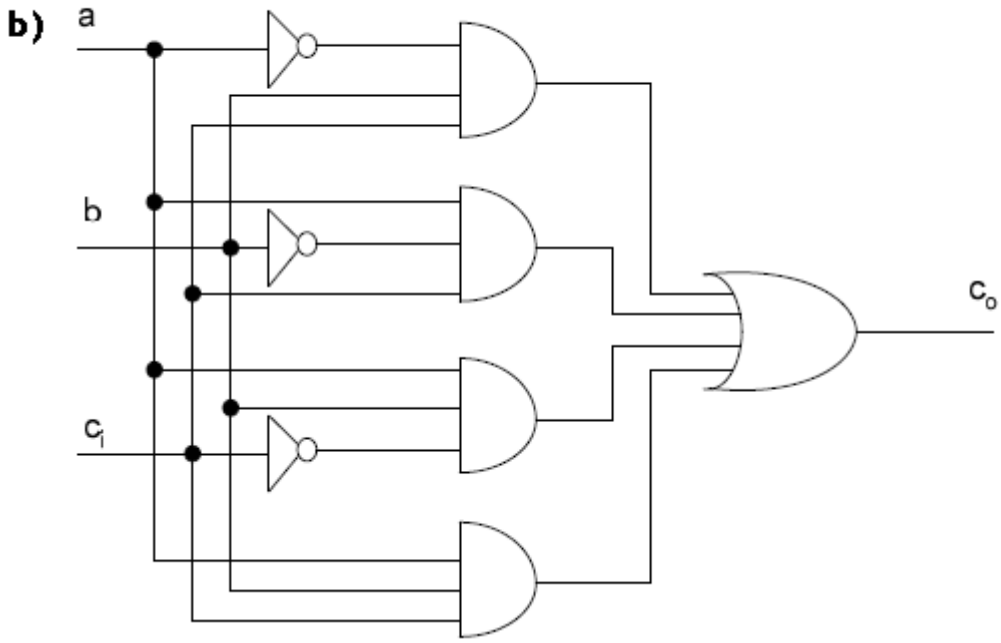
Respostas - Exercícios 02

1. Para executar o desvio dentro de um programa, o contador de programa PC deve ser atualizado com um novo valor de endereço que não seja o endereço sequencial, que no caso do MIPS é o endereço contido no PC acrescido de 4. O novo valor de endereço é calculado somando o valor contido no PC, com os 16 bits contidos na instrução, no campo "constante", multiplicados por 4. A multiplicação por 4 é necessária para apontar a uma palavra na memória.

2. O circuito serve para converter os 16 bits da instrução para um número de 32 bits. Caso o número seja negativo, o número estendido deve continuar sendo negativo. E caso o número seja positivo, o número estendido deve ser positivo.

3. **a)** $c_o = a'bc_i + ab'c_i + abc_i' + abc_i$

b)



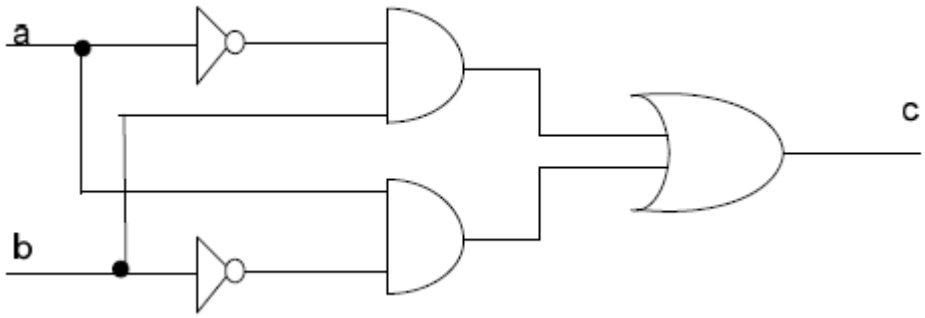
4.

a)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

b) a'b + ab'

c)



5. $7 = 1.

6. Endereço L1.

7.

a) Considerando-se a variável a em $t0,
addi $t0, $t0, -1

b) Considerando-se a variável a em $t0,
addi $t0, $0, $0

c) sw $0, 40($4) ; 40($4) = v[10], pois cada elemento do vetor é uma palavra de 32 bits, ou 4 bytes.

d) Considerando-se a variável a em $2, e a variável b em $3,
slt $7,$2,$3 ; ou seja, $2 <$3 então $7 = 1
bne $7, $0, L1 ; portanto desvia se $7 diferente de 0, como $7 =1.

e) Considerando-se a variável a em $3,
slt $7,$0,$3 ; ou seja se $0 < $3 então $7 =1
bne $7, $0, L1 ; portanto desvia se $7 diferente de 0, como $7 =1.

8. SOLUÇÃO: (lembrar que o registrador destino é o terceiro no formato).

op	rs	rt	rd	shamt	function
000000	00101	00110	00100	00000	100000

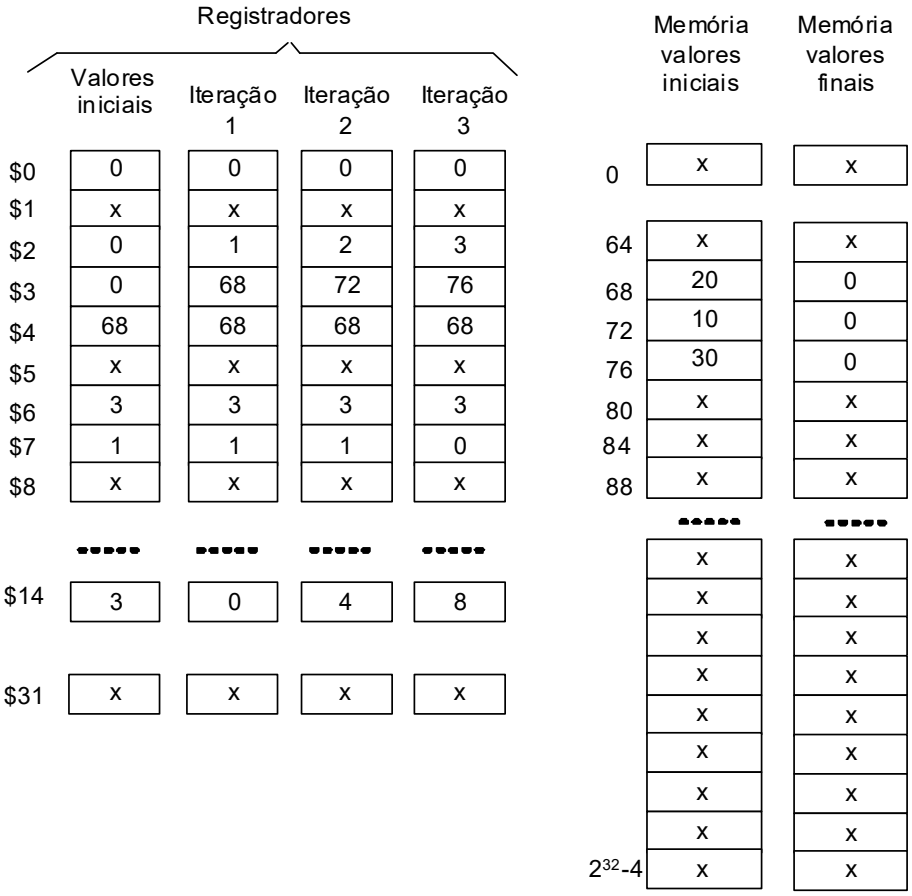
9. SOLUÇÃO:

op	rs	rt / rd	16 bits de endereço
100011	00011	00100	0000000000000101

10. SOLUÇÃO:

op	rs	rt	16 bits de desvio de endereço
000101	00011	00100	0000000000000101

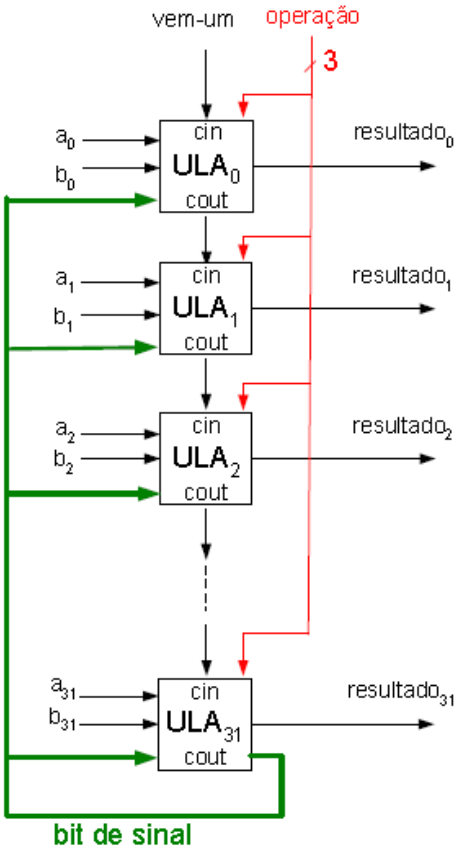
11. SOLUÇÃO:



12. SOLUÇÃO:

```
        sub   $2,$2,$2        # zera $2
L1:     muli  $14, $2, 4      # offset = i * 4
        add   $3, $5, $14     # $3 = end B[i]
        lw    $t1,  0 ($3)    # $t1 = B[i]
        add   $3,  $4, $14    # $3 = end A[i]
        lw    $t0, 0($3)      # $t0 = A[i]
        add   $t0,$t0, $t1    # $t0 = A[i] + B[i]
        sw    $t0, 0 ($3)     # A[i] = A[i]+B[i]
        addi  $2,$2,1         # i = I +1
        slt   $7, $2, $6      # se i < n $7 = 1
        bne   $7, $0, L1      # desvia se $7 <> 0
```

13. Para modificar o slt, levamos em conta que: -1 em binário é 111...11, portanto, todos os bits do resultado devem ser iguais a 1, quando a <b, e todos os bits devem ser iguais a 0, caso contrário, portanto a solução é obtida ligando o bit de sinal na entrada do multiplexador de saída de todas as 32 ULAs conforme a figura abaixo.



14. SOLUÇÃO:
    -126 em binário = 10000010
    -64 em binário = 11000000
    (-126) + (-64) em binário = 01000010

    Ocorre overflow pois a soma de dois números negativos (-126 e -64) deu positivo. O resultado em decimal é:
    $01000010 = 1x2^6 + 1x2^1 = 1x64 + 1x2 = 66$.

15.
    17 = 00010001
    18 = 00010010
    17 +18 = 00100011

16. 11111111 11000001