AC II - Exercício Prático 06

Vitor Lucio - 810862

Parte 1:

Questão 01:
A) um arquivo de texto que contém instruções de linguagem de programação.

Questão 02:
B) uma parte do processador que possui um padrão de bits.

Questão 03:
A) #

Questão 04:
C) 32

Questão 05:
D) parte do processador que contém o endereço da próxima instrução de máquina para ser obtida.

Questão 06:
C) 4

Questão 07:
D) uma declaração que diz ao montador algo sobre o que o programador quer, mas não corresponde diretamente a uma instrução de máquina.

Questão 08:
D) um nome usado no código-fonte em linguagem assembly para um local na memória.

Questão 09:
B) 0x00400000

Questão 10:
A) operando imediato

Questão 11:
B) operação bitwise

Questão 12:
D) Cada um dos registradores deve possuir 32 bit.

Questão 13:
B) Os dados são estendidos em zero à esquerda por 16 bits.

Questão 14:
C) ori $5, $0, 48

Questão 15:
A) Não.

Questão 16:
D)andi $8, $8, 0xFF

Questão 17:
A) Todos os bits em zero.

Questão 18:
A) Não. Diferentes instruções de máquina possuem campos diferentes.

Parte 2:

1)

```
# Associações:
#a -> $s0
#b -> $s1
#c -> $s2
#d -> $s3
#x -> $s4
#y -> $s5

#inicio
.text
.globl main
main:

addi $s0, $zero, 2 # a = 2;
addi $s1, $zero, 3 # b = 3;
addi $s2, $zero, 4 # c = 4;
addi $s3, $zero, 5 # d = 5;

# x = (a + b) - (c + d);
add $t0, $s0, $s1 # $t0 = a + b;
add $t1, $s2, $s3 # $t1 = c + d;
sub $s4, $t0, $t1 # x = t0  - t1 ;

# y = a - b + x;
sub $t0, $s0, $s1 # $t0 = a - b;
add $s5, $t0, $s4 # y = $t0 + x;

sub $s1, $s4, $s5 # b = x - y;

#fim
```

2)

```
# programa 2 (add, addi, sub, lógicas) {
#       x = 1;
#       y = 5 * x + 15;
# }

# Associações:
#x -> $s0
#y -> $s1

#inicio
.text
.globl main
main:

addi $s0, $zero, 1 # x = 1;

# y = 5 * x + 15;
add $t0, $s0, $s0 # t0 = x + x ou 2x;
add $t0, $t0, $t0 # t0 = t0 + t0 ou 4x;
add $t0, $t0, $s0 # t0 += x;
addi $t1, $t0, 15 # y = $t0 + x;

#fim
```

3)

```
#inicio
.text
.globl main
main:

addi $s0, $zero, 3 # x = 3;
addi $s1, $zero, 4 # y = 4;

# z = ((15 * x) + (67 * y)) * 4;
# $t0 = 15 * x;
add $t0, $s0, $s0 # $t0 = x + x ou 2x;
add $t0, $t0, $t0 # $t0 = $t0 + $t0 ou 4x;
add $t0, $t0, $t0 # $t0 = $t0 + $t0 ou 8x;
add $t0, $t0, $t0 # $t0 = $t0 + $t0 ou 16x;
sub $t0, $t0, $s0 # $t0 -= x;

# $t1 = 67 * y;
add $t1, $s1, $s1 # $t1 = y + y ou 2y;
add $t2, $t1, $s1 # $t2 = $t1 + y ou 3y;
add $t1, $t1, $t1 # $t1 = $t1 + $t1 ou 4y;
add $t1, $t1, $t1 # $t1 = $t1 + $t1 ou 8y;
add $t1, $t1, $t1 # $t1 = $t1 + $t1 ou 16y;
add $t1, $t1, $t1 # $t1 = $t1 + $t1 ou 32y;
add $t1, $t1, $t1 # $t1 = $t1 + $t1 ou 64y;
add $t1, $t1, $t2 # $t1 = $t1 + $t2 ou 67y;

add $t3 $t0, $t1 # $t3 = $t0 + $t1;

# z = $t0 + $t1
add $t3, $t3, $t3 # $t3 *= 2;
add $s2, $t3, $t3 # z = $t3 * 2;
```

4)

```
#inicio
.text
.globl main
main:

addi $s0, $zero, 3 # x = 3;
addi $s1, $zero, 4 # y = 4;

# z = ((15 * x) + (67 * y)) * 4;
# $t0 = 15 * x;
sll $t0, $s0, 4 # $t0 = x * 2^4
sub $t0, $t0, $s0 # $t0 -= x;

# $t1 = 67 * y;
add $t1, $s1, $s1 # $t1 = y + y ou 2y;
add $t2, $t1, $s1 # $t2 = $t1 + y ou 3y;
sll $t1, $t1, 5   # $t1 = $t1 * 2^5
add $t1, $t1, $t2 # $t1 = $t1 + $t2 ou 67y;

add $t3 $t0, $t1 # $t3 = $t0 + $t1;

# z = $t0 + $t1
sll $s2, $t3, 2 # z = $t3 * 2^2;
```

5)

```
#inicio
.text
.globl main
main:

# y = 200.000;
addi $t0, $zero, 25000 # $t0 = 25.000;
sll $s1, $t0, 3 # x = $t0 * 2^3;

# x = 100.000;
#srl $s0, $s1, 1 # x = $t0 / 2;
addi $t0, $zero, 12500 # $t0 = 25.000;
sll $s0, $t0, 3 # x = $t0 * 2^2;

add $s2, $s0, $s1 # z = x + y;

#fim
```

6)

```
#inicio
.text
.globl main
main:

# y = 300.000;
addi $t0, $zero, 18750 # $t0 = 18.750;
sll $s1, $t0, 4 # x = $t0 * 2^4;

# x = 0x7FFFFFFF;
# addi $t0, $zero, -1 # $t0 = -1;
# srl $s0, $t0, 1 # x = $t0 >> 1;
ori $t0, $zero, 0x7FFF # $t0 = 0x00007FFF
sll $t0, $t0, 16 # $t0 <<= 4 ou $t0 = 0x7FFF0000
ori $s0, $t0, 0xFFFF # $x = $t0 | 0x0000FFFF

# z = x - 4 * y;
sll $t0, $s1, 2 # $t0 = y * 2^2;
sub $s2, $s0, $t0 # z = x - $t0;

#fim
```

---

7)

```
#inicio
.text
.globl main
main:

ori $t0, $zero, 0x01
sll $t0, $t0, 31 # $t0 <<= 31 ou $t0 = #0x8000000
sra $s0, $t0, 31 # $8 = $t0 >> 31 ou $s0 = 0xFFFFFFFF

#fim
```

8)

```
#inicio
.text
.globl main
main:

# $8 = 0x12345678;
ori $t0, $zero, 0x1234 # $t0 = 0x00001234
sll $t0, $t0, 16       # $t0 = 0x12340000
ori $s0, $t0, 0x5678   # $s0 = 0x12345678

# $9 = 0x12
srl $s1, $s0, 24

# $10 = 0x34
sll $t1, $s0, 8  # $t1 = 0x34567800
srl $s2, $t1, 24 # $s2 = 0x00000034

# $11 = 0x56
sll $t2, $s0, 16 # $t2 = 0x56780000
srl $s3, $t2, 24 # $s3 = 0x00000056

# $12 = 0x78
sll $t3, $s0, 24 # $t3 = 0x78000000
srl $s4, $t3, 24 # $s4 = 0x00000078

#fim
```

9)

9)

```
.data
x1: .word 15
x2: .word 25
x3: .word 13
x4: .word 17
soma: .word -1

.text
.globl main
main:

# $t0 = 0x10010000
ori $t0, $zero, 0x1001 # $t0 = 0x00001001
sll $t0, $t0, 16       # $t0 = 0x10010000

or $t1, $zero, $t0 # $t1 = $t0

# $s0 = x1
lw $s0, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s1 = x2
lw $s1, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s2 = x3
lw $s2, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s3 = x4
lw $s3, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s4 = soma
lw $s4, 0($t1)

add $t2, $zero, $s0 # $t2 = 0 + $s0;
add $t2, $t2, $s1 # $t2 += $s1;
add $t2, $t2, $s2 # $t2 += $s2;
add $t2, $t2, $s3 # $t2 += $s3;

sw $t2, 0($t1) # soma = x1 + x2 + x3 + x4
```

10)

```
.data
x: .word 5
z: .word 7
y: .word 0

.text
.globl main
main:

# $t0 = 0x10010000
ori $t0, $zero, 0x1001 # $t0 = 0x00001001
sll $t0, $t0, 16       # $t0 = 0x10010000

or $t1, $zero, $t0 # $t1 = $t0

# $s0 = x
lw $s0, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s2 = z
lw $s2, 0($t1)
addi $t1, $t1, 4 # $t1 += 4 (bytes)

# $s1 = y
lw $s1, 0($t1)

# y = 127 * x - 65 * z + 1;

# $t2 = 127 * x;
# $t2 = x * ceil(2^(log2(127)));
# $t2 = x * 2^7;
sll $t2, $s0, 7 # $t2 = x * 128;
sub $t2, $t2, $s0

# $t3 = 65 * z;
sll $t3, $s2, 6 # $t3 = z * 64;
add $t3, $t3, $s2 # $t3 += z;

# $t4 = $t2 - $t3 + 1;
sub $t4, $t2, $t3 # $t4 = $t2 - $t3;
addi $t4, $t4, 1 # $t4 += 1;

sw $t4, 0($t1)
```

11)

```
.data
x: .word 100000
z: .word 200000
y: .word 0

.text
.globl main
main:

# $t0 = 0x10010000
ori $t0, $zero, 0x1001 # $t0 = 0x00001001
sll $t0, $t0, 16       # $t0 = 0x10010000

or $t1, $zero, $t0 # $t1 = $t0

lw $s0, 0($t1) # $s0 = x
addi $t1, $t1, 4 # $t1 += 4 (bytes)

lw $s1, 0($t1) # $s1 = z
addi $t1, $t1, 4 # $t1 += 4 (bytes)

lw $s2, 0($t1) # $s2 = y

# y = x - z + 300000;

sub $t2, $s0, $s1 # $t2 = x - z;
 # $t3 = 300.000
addi $t3, $zero, 18750 # $t0 = 18.750;
sll $t3, $t3, 4 # x *= 2^4;

add $t2, $t2, $t3 # $t2 += $t3;

sw $t2, 0($t1)
```

12)

```
.data
a: .word 5
b: .word 0x10010000
c: .word 0x10010004
d: .word 0x10010008

.text
.globl main
main:

# $t0 = 0x10010000
ori $t0, $zero, 0x1001 # $t0 = 0x00001001;
sll $t0, $t0, 16       # $t0 = 0x10010000;

or $t1, $zero, $t0 # $t1 = $t0;

lw $s0, 0xC($t1) # $s0 = d;
lw $s1, 0($s0) # $s1 = c;
lw $s2, 0($s1) # $s2 = b;
lw $s3, 0($s2) # $s3 = a;

add $s3, $s3, $s3 # $s3 = 2 * $s3 ou 2a;

sw $s3, 0($s2) # ***d = (***d) * 2;
```

13)

```
.data
A: .word -5

.text
.globl main
main:

# $s0 = 0x10010000
ori $t0, $zero, 0x1001
sll $t0, $t0, 16

lw $s0 0($t0) # $s0 = A;
sra $t1, $s0, 31 # $t1 = "sinal" de $s1

beq $t1, $zero end # if ($t1 == 0) goto end; else A = abs(A);
sub, $s0, $zero, $s0 # $t2 = abs($s1);

end:

sw $s0, 0($t0)
```

14)

```
.data
A:  .word   4

.text
    .globl  main

main:

    # $s0 = 0x10010000
    ori     $t0,    $zero,  0x1001
    sll     $t0,    $t0,    16

    lw      $s0,    0($t0)          # $s0 = A;
    andi    $t1,    $s0,    1       # $t1 = "sinal" de $s1

    bne     $t1,    $zero,  else    # if (isEven($s0)) $s1 = 0; else goto else;

if:
    or      $s1,    $zero,  $zero   # $s1 = 0;
    j       end                     # goto end

else:
    ori     $s1,    $zero,  1       # $s1 = 1;

end:

    sw      $s1,    4($t0)

    #fim
```

15)

```
.data
soma:   .word   0
vetor:  .word   0x8

.text
        .globl  main

main:

    # $s0 = 0x10010000
    ori     $t0,    $zero,  0x1001
    sll     $t0,    $t0,    16

    lw      $s0,    0($t0)          # $s0 = soma;
    lw      $s1,    4($t0)          # $s1 = vetor;
    add     $s1,    $s1,    $t0     # $s1 = 3? endere?o

    or      $s2,    $zero,  $zero   # $s2 = 0;

    ori     $t1,    $zero,  100     # $t1 = 100;

for:
    beq     $s2,    $t1,    end     # if ($s2 == 100) goto end;

    sll     $t2,    $s2,    2       # $t2 = 4 * i;
    add     $t2,    $t2,    $s1     # endereço de vetor[i]?

    sll     $s3,    $s2,    1       # $s3 = 2 * i;
    addi    $s3,    $s3,    1       # $s3 += 1;

    sw      $s3,    0($t2)          # vetor[i];
    add     $s0,    $s0,    $s3     # soma += $s3;

    addi    $s2,    $s2,    1       # = i++;
    j       for                     # goto for

end:

    sw      $s0,    0($t0)

    #fim
```

Perguntas Finais

Questão 01:
C) 64

Questão 02:
B) hi e lo

Questão 03:
A) mult

Questão 04:
C) mflo $8

Questão 05:
B) 32

Questão 06:
A) lo

Questão 07:
D) div

Questão 08:
A) 1110 0110

Questão 09:
A) Se o inteiro for unsigned, o shift o divide por 2. Se o inteiro for signed, o shift o divide por 2.

Questão 10:
A) ori $3,$0,3
mult $8,$3
mflo $9
addi $9,$9,7