**ORGANIZAÇÃO DE COMPUTADORES I/INTRODUÇÃO À ARQUITETURA DE COMPUTADORES – PROVA 1 – 2017/2 – 25 pontos**

1 – Represente os seguintes números: **(2 pontos cada)**

1. -120 em complemento de dois, considerando um registrador de 16 bits (Represente todos os bits!).
2. 321 em ponto flutuante de precisão simples. Lembre-se que no ponto flutuante temos 8 bits de expoente, 23 bits para o significando, 1 bit de sinal e um desvio no expoente igual a 127.
3. Apresente uma conta com números inteiros que gera overflow.

2 – Dado somadores de um bit com as entradas *a0*, *b0* e *carry*, e as saídas *s0* e *carry*, mostre como implementar a subtração de dois números inteiros de X bits usando X somadores. **(4 pontos)**

3 – A máquina UPS (Ultra Power Superblaster) executa loads e stores em 5 ciclos, operações lógico aritméticas em 4 e ponto flutuante em 10. Seu clock é de 1GHz. Já a UPM (Ultra Power Megablaster) executa stores em 6 ciclos, operações lógico-aritméticas em 4 e ponto flutuante em 12, mas seu clock é de 1.5GHz.

1. Calcule a CPI média de ambas, considerando um programa que 20% de operações são load-store, 50% são lógico-aritméticas, e 30% são operações de ponto flutuante. **(2 pontos)**
2. Qual é a vazão (*throughput*) de ambas? **(2 pontos)**
3. Qual seria o speedup da UPM sobre a UPS? **(1 ponto)**

4- Faça um programa em MIPS que calcule o enésimo valor da série de Fibonacci (Fn), para n >= 1. O valor N estará na posição apontada pelo registrador $t0, e o resultado final deve estar armazenado no registrador $v0. Para homogeneizar, assuma que F0 = 1 e F1 = 1. **(Valor: 4 pontos)**

5 – Explique também a função dos registradores $t0, $t1, $t2 e $t3 no código abaixo. Em seguida, coloque comentários linha a linha. Finalmente, explique em uma linha qual é o resultado final do programa, armazenado em $v1. (Dica: o programa começa no main). **(Valor 2 + 2 + 2)**

label3: addi $sp, $sp, 4

sw $t0, 4($sp)

andi $t0, $a0, 0x1

add $v0, $a0, $t0

muli $v0, $v0, 0x2

lw $t0, 4($sp)

addi $sp, $sp, 4

jr $ra

main: lw $t1, 0($t2)

lw $t0, 4($t2)

add $t3, $zero, $zero

label2: beq $t0, $t1, label1

add $a0, $t0, $zero

jal label3

addi $t0, $t0, 1

j label2

label1: add $v1, $t0, $zero