**Arquitetura de Dispositivos Móveis**

**2015/2016 - Exame - Duração: 2h**

TeSP AM - Arquitetura de Dispositivos Móveis, André Pereira

Nome:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Nº:\_\_\_\_\_\_\_\_\_

Nota 1: As perguntas assinaladas com **CBi** são as de competências básicas (“**i**” indica o número da pergunta). As que estiverem assinaladas com **CCi** são perguntas de competências complementares.

Nota 2: Para todas as respostas apresente todos os cálculos efetuados

**Nota 3: É necessário responder acertadamente a pelo menos 4 básicas**

**PARTE I**

1. **Efetue** as seguintes conversões: **(CB1)**
   1. 10110.102 para decimal (base 10)
   2. 0x3CF para decimal (base 10) e binário (base 2)
2. **Converta** o valor -187 para uma representação binária, usando 10 *bits*, nas seguintes representações: **(CB2)**
   1. Complemento para 2
   2. Excesso 2n
3. **Considere** o seguinte formato para vírgula flutuante, representados com 8 bits:

* FORMATO1:
  + o *bit* mais significativo contém o sinal
  + os 4 *bits* seguintes formam o expoente (em excesso 2n)
  + os restantes 3 *bits* formam a mantissa

Para todos os restantes casos, as regras são as mesmas que as da norma IEEE (valor normalizado, desnormalizado, representação do 0, infinito, e NaN).

* 1. **Converta** os valores 100110112 e 011100002 para decimal. **(CB3)**

1. **Indique** a informação que circula nos barramentos apenas durante a última fase de execução da instrução addl (0x100), %ebx, e **indique** quais os registos que foram alterados. Considere que a instrução ocupa 16 *bits* em memória e é executada numa máquina com as seguintes características: **(CB4)**

* Arquitetura: 16 bits
* Ordenação: *Big Endian*
* Valores em memória:
  + De 0x100 até 0x103: 1B 6C D9 11
* Valores em registos:
  + %ebx = 0x0100

1. Considere o seguinte excerto de código C, e **complete** o código *assembly* correspondente.**(CB5)**

|  |  |
| --- | --- |
| ... if( x < y) return 1; else return 0; ... | ...  cmpl %eax, %edx  j\_\_ .L3  movl $1, \_\_\_\_  jmp .L5 .L3:  movl \_, %eax .L5:  ... |

1. Considere o seguinte excerto de código *assembly* e os valores para os registos %eax e %ebx, e **indique** qual a instrução a ser executada depois do salto condicional. **(CB6)**

|  |  |
| --- | --- |
| cmpl %eax, %ebx  jge .L1  addl $10, %eax  jmp .FIM  .L1  addl $10, %ebx  .FIM  pushl %eax | **--- VALORES DE REGISTOS ---**  **%eax ->** 0x15  **%ebx ->** 0xC |

**PARTE II**

1. Considere uma representação de inteiros em Complemento para 2 com 8 bits. **Indique** o resultado, em binário, da seguinte operação: 0xC2 + 0x30 **(CC1)**
2. Considere o FORMATO1 apresentado da pergunta 3 (PARTE I), e **indique** (em decimal): **(CC2)**
   1. O maior valor positivo finito possível de ser representado
   2. O valor negativo mais próximo de zero possível de ser representado
3. **Crie** o código *assembly* da seguinte função:**(CC3)**

|  |  |
| --- | --- |
| **------- Código C -------**  int func (int x, int y){  if(x > y){  return x+1;  }else{  return y+1;  }  } | **----- Código Assembly -----**  .func |

**ANEXO I - Regras da Norma IEEE**

