MC404: Organização de Computadores e Linguagem de Montagem

1ª Prova (29/9/2011)

Questão	Valor	Nota
1	2,0	
2	1,5	
3	1,5	
4	2,5	
5	2,5	
Total	10,0	

Nome:

RA:

Instruções: A duração da prova é de uma hora e quarenta minutos. *Consulta exclusivamente às folhas de resumos de instruções do Faíska.* Comente seu código! Qualquer tentativa de fraude será punida com zero para todos os envolvidos.

Questão 1. (2,0 pontos) Um projetista de *hardware* decidiu adicionar a instrução `cmp rd, [rs]` ao Faíska. Esta instrução compara o conteúdo do registrador `rd` com o conteúdo da memória apontado pelo registrador `rs`, modificando as *flags* de acordo com o resultado. Descreva os passos necessários para se executar esta instrução no Faíska. Lembre-se que o Faíska possui os registradores `IP` (*Instruction Pointer*) e `IR` (*Instruction Register*). (Utilize o verso desta folha para esta resposta).

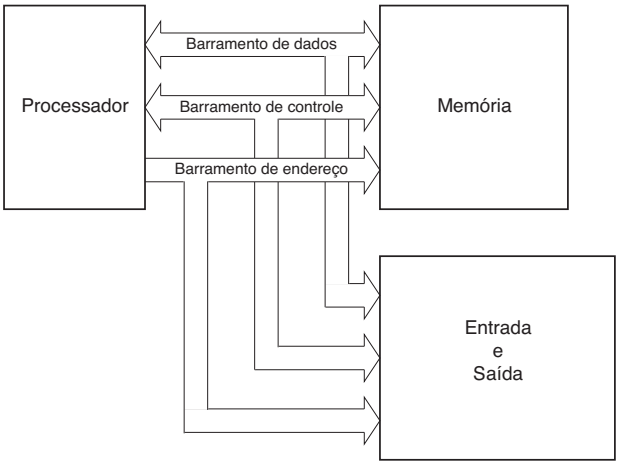

Figura 1: Diagrama do processador.

Questão 2. (1,5 ponto) Descreva o que computa o trecho de programa em linguagem de montagem abaixo, colocando comentários no código.

```
misterio:
      set  r0,0
volta:
      cmp  r1,0
volta1:
      jz   final
      shl  r1,1
      jnc  volta1
      add  r0,1
      jmp  volta
final:
```

Questão 3. (1,5 pontos) O que são exceções? Como funcionam (o que acontece quando uma exceção ocorre)? Cite dois exemplos de exceções típicas de processadores.

Questão 4. (2,5 pontos) Escreva um procedimento em linguagem de montagem do Faiska que receba no registrador `r1` o endereço de uma cadeia de caracteres '0's e '1's e devolva no registrador r0 o valor que essa cadeia de caracteres representa em binário. A cadeia tem no máximo 32 carateres e é terminada por um *byte* de valor 0 (como padrão em C). Por exemplo, se a cadeia de caracteres apontada por `r1` é '0','0','1','0','0','1','\0', o valor de `r0` ao retornar deve ser 9. (Utilize o verso desta folha para esta resposta).

Questão 5. (2,5 pontos) Traduza o procedimento em C abaixo para linguagem de montagem do Faíska. Considere que os parâmetros são passados pela pilha, empilhados na ordem inversa da declaração (isto é, o parâmetro `valor1` é empilhado por último).

```
void dif_abs(int valor1, int valor2, int *result)
{
   if (valor1 > valor2)
       *result = valor1 - valor2;
   else
       *result = valor2 - valor1;
}
```