Projeto de Arquitetura de Sistemas Digitais (FTL066)
Implementação em VHDL do Processador MIPS

No projeto da disciplina de arquitetura de sistemas digitais, você deverá implementar em uma linguagem de descrição de hardware VHSIC *"Very High Speed Integrated Circuits"* (i.e., em VHDL), o processador MIPS com *pipeline* considerando perigos de controle, dados e estrutural. A descrição completa de cada bloco funcional pode ser encontrado em [1].

Crie bibliotecas para cada um dos blocos funcionais do processador MIPS com o intuito de modularizar o projeto do processador. Além disso, o código em VHDL resultante desta implementação deve ser embarcado em um FPGA (*Field-Programmable Gate Array*) e testado usando o programa C/assembly mostrado logo abaixo (considere a varriável *v* em *$a0*, *n* em *$a1*, *i* em *$s0* e *j* em *$s1*).

Vale ressaltar que, além da implementação do processador em VHDL, você também deverá escrever um relatório técnico, descrevendo todos os blocos funcionais do processador MIPS, código VHDL implementado e os testes que foram realizados para cada bloco funcional com o intuito de garantir o correto funcionamento do sistema.

Procedimento de ordenação em C

```
void sort (int v[], int n) {
  int i, j;
  for (i = 0; i < n; i += 1) {
    for (j = i – 1;
         j >= 0 && v[j] > v[j + 1];
         j -= 1) {
      swap(v,j);
    }
  }
}
```

Procedimento de ordenação em Assembly

```
        move $s2, $a0              # salva $a0 em $s2
        move $s3, $a1          # salva $a1 em $s3
        move $s0, $zero        # i = 0
for1tst: slt  $t0, $s0, $s3    # $t0 = 0 if $s0 ≥ $s3 (i ≥ n)
        beq  $t0, $zero, exit1 # vai p/ exit1 if $s0 ≥ $s3 (i ≥ n)
        addi $s1, $s0, –1      # j = i – 1
for2tst: slti $t0, $s1, 0      # $t0 = 1 if $s1 < 0 (j < 0)
        bne  $t0, $zero, exit2 # vai p/ exit2 if $s1 < 0 (j < 0)
        sll  $t1, $s1, 2       # $t1 = j * 4
        add  $t2, $s2, $t1     # $t2 = v + (j * 4)
        lw   $t3, 0($t2)       # $t3 = v[j]
        lw   $t4, 4($t2)       # $t4 = v[j + 1]
```

```
        slt  $t0, $t4, $t3          # $t0 = 0 if $t4 ≥ $t3
        beq  $t0, $zero, exit2    # vai p/ exit2 if $t4 ≥ $t3
        move $a0, $s2             # 1st param de troca é v (old $a0)
        move $a1, $s1             # 2nd param de troca é j
        jal  swap                 # chama procedimento swap
        addi $s1, $s1, –1         # j –= 1
        j    for2tst               # jump to test of inner loop
exit2:  addi $s0, $s0, 1          # i += 1
        j    for1tst              # jump to test of outer loop
sort:   addi $sp,$sp, –20     # abrir espaço na pilha para 5 regs.
        sw $ra, 16($sp)       # salva $ra na pilha
        sw $s3,12($sp)        # salva $s3 na pilha
        sw $s2, 8($sp)        # salva $s2 na pilha
        sw $s1, 4($sp)        # salva $s1 na pilha
        sw $s0, 0($sp)        # salva $s0 na pilha
exit1: lw $s0, 0($sp)  # restaura $s0 da pilha
        lw $s1, 4($sp)        # restaura $s1 da pilha
        lw $s2, 8($sp)        # restaura $s2 da pilha
        lw $s3,12($sp)        # restaura $s3 da pilha
        lw $ra,16($sp)        # restaura $ra da pilha
        addi $sp,$sp, 20      # restaura ponteiro de pilha
        jr $ra              # retorna para o procedimento que chamou
```

Referências

[1] D. A. Patterson and J. L. Hennessy, *Computer Organization and Design - The Hardware / Software Interface (Revised 4th Edition)*. The Morgan Kaufmann Series, Academic Press, 2012

Datas das apresentações parciais

O progresso do projeto será acompanhado através de apresentação e relatório técnico, a cada duas semanas, de acordo com o seguinte planejamento:

- Primeira reunião: 23/06/15
- Segunda reunião: 07/07/15
- Terceira reunião: 21/07/15

Data de entrega: 04 de agosto de 2014 (terça-feira).
Após esta data será descontado 2 pontos por dia de atraso.
O projeto pode ser desenvolvido e entregue individualmente ou em dupla.

04/06/2015