LABORATÓRIO DE ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES I

AULA 13: implementação dos demais componentes do nRisc

Professor: Mateus Felipe Tymburibá Ferreira
Data: 02/09/2021

Aluno: Darmes Araujo Dias

---

1) Apresente o código fonte, em Verilog, dos módulos que descrevem os componentes do seu nRisc implementados.

2) Configure simulações que demonstrem o correto funcionamento de todos os componentes implementados. Apresente o código fonte desse(s) módulo(s) de simulação.

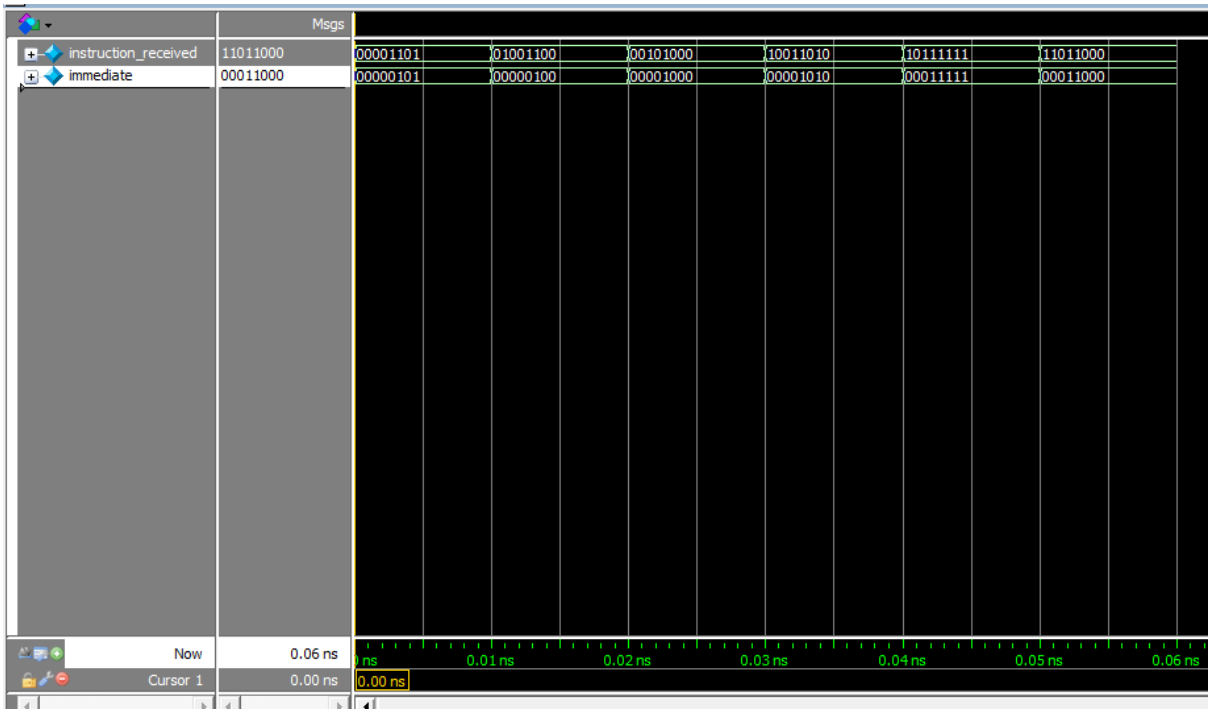
3) Demonstre e explique o funcionamento das simulações, usando fotos da tela (screenshots) da aplicação ModelSim em execução.

➢ Extensor de sinal

```
1   module extensor_de_sinal(instruction_received, immediate);
2      input [7:0] instruction_received;
3      output [7:0] immediate;
4
5      wire [2:0] opcode = instruction_received [7:5];
6
7      assign immediate =
8      (opcode == 3'b000) ?  instruction_received[2:0] : // store
9      (opcode == 3'b010) ?  instruction_received[2:0] : // load
10     (opcode == 3'b001) ?  instruction_received[3:0] ://  la
11     (opcode == 3'b100) ?  instruction_received[3:0] ://  addi
12     (opcode == 3'b101) ?  instruction_received[4:0] ://  beq
13     (opcode == 3'b110) ?  instruction_received[4:0] : 8'b00000000; // j
14
15  endmodule
16
```

```
1  module extensor_de_sinal_simulation(input a, output b);
2     reg [7:0] instruction_received;
3     wire [7:0] immediate;
4
5     wire [7:0] INSTRUCTION_RECEIVED;
6
7     assign INSTRUCTION_RECEIVED =  instruction_received;
8
9     extensor_de_sinal es(INSTRUCTION_RECEIVED, immediate);
10
11    reg clock;
12
13    always #10 clock = ~clock;
14    initial begin
15    #0 instruction_received = 8'b00001101;
16    #10 instruction_received = 8'b01001100;
17    #10 instruction_received = 8'b00101000;
18    #10 instruction_received = 8'b10011010;
19    #10 instruction_received = 8'b10111111;
20    #10 instruction_received = 8'b11011000;
21    #10 $finish;
22    end
23 endmodule
24
```

O testbench foi escrito para o teste do extensor de sinal, ele consiste em receber instruções que usam imediato e observar a saída. O que se espera é: ao receber uma instrução o extensor de sinal consulta o opcode, e depois estende o imediato de acordo com a instrução recebida. Ao receber store e load ele estende o imediato de 3 bits para 8 bits, ao receber load address e add immediate ele estende o imediato de 4 bits para 8 bits e ao receber jump e beq ele estende o imediato de 5 bits para 8 bits.



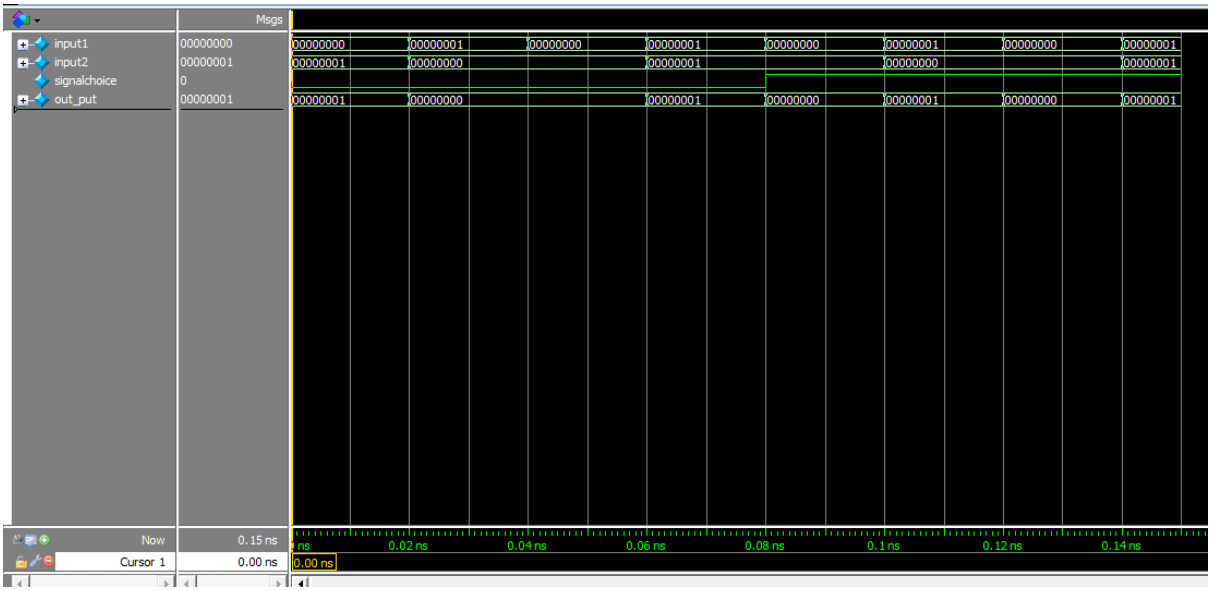
A simulação foi como o esperado.
O wave.do foi salvo na pasta extensor_de_sinal_simulation.

➢ MUX

```
1   module mux(input1, input2, signalchoice, out_put);
2       input wire [7:0] input1;
3       input wire [7:0] input2;
4       input wire signalchoice;
5       output wire [7:0] out_put;
6
7       assign out_put = (signalchoice) ? input1 : input2;
8
9   endmodule
10
```

```
module mux_simulation(input a, input b);
   reg [7:0] input1;
   reg [7:0] input2;
   reg signalchoice;
   wire [7:0] out_put;

   wire [7:0] INPUT1;
   wire [7:0] INPUT2;
   wire SIGNALCHOICE;

   assign INPUT1 = input1;
   assign INPUT2 = input2;
   assign SIGNALCHOICE = signalchoice;

   mux mux_test(INPUT1, INPUT2, SIGNALCHOICE, out_put);

   initial
   begin
      #0 signalchoice = 0;
      #0 input1 = 0;
      #0 input2 = 1;
      #20 signalchoice = 0;
      #0 input1 = 1;
      #0 input2 = 0;
      #20 signalchoice = 0;
      #0 input1 = 0;
      #0 input2 = 0;
      #20 signalchoice = 0;
      #0 input1 = 1;
      #0 input2 = 1;
      #20 signalchoice = 1;
      #0 input1 = 0;
      #0 input2 = 1;
      #20 signalchoice = 1;
      #0 input1 = 1;
      #0 input2 = 0;
      #20 signalchoice = 1;
      #0 input1 = 0;
        #0 input2 = 0;
        #20 signalchoice = 1;
        #0 input1 = 1;
        #0 input2 = 1;
        #10 $finish;
    end

endmodule

```

O testbench foi escrito para verificar o funcionamento do multiplexador, ele consiste em testar todas as entradas de input1 e input2, sendo signalchoice 0 e depois 1, e observar as saídas do mux.



A simulação foi como o esperado.

O wave.do foi salvo na pasta mux_simulation.