UNIVERSIDADE FEDERAL DO RECÔNCAVO DA BAHIA

CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS

GCET231 CIRCUITOS DIGITAIS II

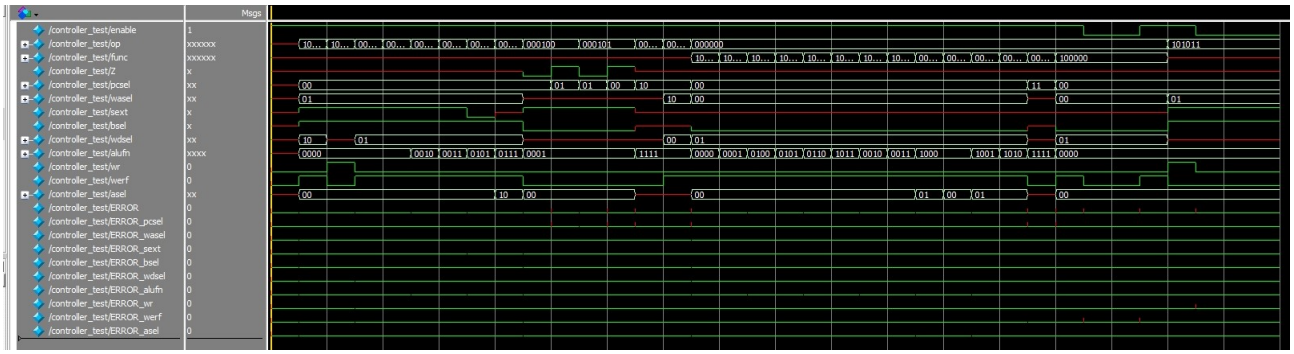
PROJETO FINAL PARTE 1: CONSTRUÇÃO DE UNIDADE DE CONTROLE PARA O PROCESSADOR MIPS231

Ejziel Sampaio Santos, Fabrício Velôso de Jesus, Mateus Rocha Velame, Rodrigo Bastos Suzart

Professor: João Carlos Nunes Bittencourt

1 WAVEFORM DA SIMULAÇÃO

Figura 1 – Waveform da Simulação



Fonte: Autoria Própria

2 TABELA DE DECODIFICAÇÃO DE INSTRUÇÕES

Type	Instr	werf	wdsel	wasel	asel	bsel	sext	wr	alufn	pcsel	
I-TYPE	LW	1	10	01	00	1	1	0	0000	00	
	SW	0	XX	01	00	1	1	1	0000	00	
	ADDI	1	01	01	00	1	1	0	0000	00	
	ADDIU	1	01	01	00	1	1	0	0000	00	
	SLTI	1	01	01	00	1	1	0	0010	00	
	SLTIU	1	01	01	00	1	1	0	0011	00	
	ORI	1	01	01	00	1	0	0	0101	00	
	LUI	1	01	01	10	1	X	0	0111	00	
	ANDI	1	01	01	00	1	0	0	0100	00	
	XORI	1	01	01	00	1	0	0	0110	00	
	BEQ	0	XX	XX	00	0	1	0	0001	01	Z=1 | Z=0
	BNE	0	XX	XX	00	0	1	0	0001	01	Z=1 | Z=0
J-TYPE	J	0	XX	XX	XX	X	X	0	1111	10	
	JAL	1	00	10	XX	X	X	0	1111	10	
R-TYPE	ADD	1	01	00	00	0	X	0	0000	00	
	ADDU	1	01	00	00	0	X	0	0000	00	
	SUB	1	01	00	00	0	X	0	0001	00	
	AND	1	01	00	00	0	X	0	0100	00	
	OR	1	01	00	00	0	X	0	0101	00	
	XOR	1	01	00	00	0	X	0	0110	00	
	NOR	1	01	00	00	0	X	0	1011	00	
	SLT	1	01	00	00	0	X	0	0010	00	
	SLTU	1	01	00	00	0	X	0	0011	00	
	SLL	1	01	00	01	0	X	0	1000	00	
	SLLV	1	01	00	00	0	X	0	1000	00	
	SRL	1	01	00	01	0	X	0	1001	00	
	SRA	1	01	00	01	0	X	0	1010	00	
	JR	0	XX	XX	XX	X	X	0	1111	11	

3 MÓDULO CONTROLLER

Listing 3.1 – Módulo controller.sv

```
1  // ------------------------------------------------------------------------
2  // Universidade Federal do Rec ncavo da Bahia
3  // ------------------------------------------------------------------------
4  // Author : Ejziel Santos, Fabr cio Jesus, Mateus Velame, Rodrigo Suzart
5  // File   : controller.sv
6  // Create : 2019-11-07 22:26:57
7  // Editor : Visual Studio Code
8  // ------------------------------------------------------------------------
9  // Module Purpose:
10 //              Unidade de controle para o processador MIPS231
11 // ------------------------------------------------------------------------
12 // Entradas:
13 //      enable : sinal de controle de escrita
14 //      op     : opcode da instru   o
15 //      func   : fun   o para instru   es R-type
16 //    Z        : flag zero vinda da ALU
17 // ------------------------------------------------------------------------
18 // Saidas:
19 //      pcsel  : seletor do multiplexador de PC.
20 //    wasel    : seletor do multiplexador do endere o de escrita no register file
21 //    sext     : controle do sign extend (0 zero-extends, 1 sign-extends)
22 //    bsel     : seletor do multiplexador da entrada B da ALU
23 //    wdsel    : seletor do multiplexador de dados de escrita no register file
24 //    alufn    : fun   o a ser executada pela ALU
25 //    wr       : write enable da mem ria de dados
26 //    werf     : write enable do register file
27 //    asel     : seletor do multiplexador da entrada A da ALU
28 // ------------------------------------------------------------------------
29
30 `timescale 1ns / 1ps
31 `default_nettype none
32 `include "opcode.svh"
33 `include "aluop.svh" //N o tinha isso
34
35 module controller(
36     input  wire enable,
37     input  wire [5:0] op,
38     input  wire [5:0] func,
39     input  wire Z,
40     output wire [1:0] pcsel,
41     output wire [1:0] wasel,
42     output wire sext,
43     output wire bsel,
44     output wire [1:0] wdsel,
45     output logic [3:0] alufn,
46     output wire wr,
47     output wire werf,
```

```
48      output wire [1:0] asel
49      );
50
51    assign pcsel = ((op == 6'b0) & (func == `JR)) ? 2'b11    // controla o
52    //multiplexador de 4-entradas
53                 : ((op == 6'b000100) & (Z)) || ((op == 6'b000101) & (!Z)) ?
54          2'b01  : ((op == 6'b000010) || (op == 6'b000011))  ?
55          2'b10  : 2'b00;
56
57    logic [9:0] controls;
58    wire _werf_, _wr_;
59    assign werf = _werf_ & enable;        // desativa as escritas no registrador
60    //quando o processador est  desativado
61    assign wr = _wr_ & enable;            // destiva a escrita na mem ria quando
62    //o processador est  desativado
63
64    assign {_werf_,wdsel[1:0],wasel[1:0],asel[1:0],bsel,sext,_wr_} = controls[9:0];
65
66    always_comb
67       case(op)                                          // instru   es non-R-type
68          `LW: controls <= 10'b 1_10_01_00_1_1_0;        // LW
69          `SW: controls <= 10'b 0_XX_01_00_1_1_1;        // SW
70        `ADDI,                                           // ADDI
71       `ADDIU,                                           // ADDIU
72        `SLTI: controls <= 10'b 1_01_01_00_1_1_0;          // SLTI
73       `SLTIU: controls <= 10'b 1_01_01_00_1_1_0;        // SLITIU
74          `ORI: controls <= 10'b 1_01_01_00_1_0_0;        // ORI
75          `LUI: controls <= 10'b 1_01_01_10_1_X_0;        // LUI
76        `ANDI: controls <= 10'b 1_01_01_00_1_0_0;
   // ANDI
77        `XORI: controls <= 10'b 1_01_01_00_1_0_0;       // XORI
78         `BEQ: controls <= 10'b 0_XX_XX_00_0_1_0;        // BEQ
79         `BNE: controls <= 10'b 0_XX_XX_00_0_1_0;        // BNE
80           `J: controls <= 10'b 0_XX_XX_XX_X_X_0;        // J
81         `JAL: controls <= 10'b 1_00_10_XX_X_X_0;        // JAL
82        6'b000000:
83           case(func)                                    // R-type
84               `ADD,
85              `ADDU,                                          // ADD and ADDU
86               `SUB: controls <= 10'b 1_01_00_00_0_X_0;       // SUB
87               `AND: controls <= 10'b 1_01_00_00_0_X_0;       // AND
88                `OR: controls <= 10'b 1_01_00_00_0_X_0;       // OR
89               `XOR: controls <= 10'b 1_01_00_00_0_X_0;       // XOR
90               `NOR: controls <= 10'b 1_01_00_00_0_X_0;       // NOR
91               `SLT: controls <= 10'b 1_01_00_00_0_X_0;       // SLT
92              `SLTU: controls <= 10'b 1_01_00_00_0_X_0;       // SLTU
93              `SLLV: controls <= 10'b 1_01_00_00_0_X_0;       // SLLV
94               `SLL: controls <= 10'b 1_01_00_01_0_X_0;       // SLL
95               `SRL: controls <= 10'b 1_01_00_01_0_X_0;       // SRL
96               `SRA: controls <= 10'b 1_01_00_01_0_X_0;       // SRA
97                `JR: controls <= 10'b 0_XX_XX_XX_X_X_0;       // JR
98              default:   controls <= 10'b 0_xx_xx_xx_x_x_0; // instru   o
```

```
99          //desconhecida, desative a escrita no registrador e na mem ria
100           endcase
101       default: controls <= 10'b 0_xx_xx_xx_x_x_0;          // instru   o
102       //desconhecida, desative a escrita no registrador e na mem ria
103     endcase
104
105
106     aludec alu_decoder (
107         .funct(func),
108         .opcode(op),
109         .aluop(alufn)
110     );
111
112 endmodule
```