LABORATÓRIO 3 – MULTIPLEXADORES

EFRAIN MARCELO PULGAR PANTALEON[1]
FERNANDO LUCAS SOUSA SILVA[2]
MATHEUS GOMES DINIZ ANDRADE[3]
TEÓPHILO VITOR DE CARVALHO CLEMENTE[4]

Universidade Federal do Rio Grande do Norte, Departamento de Engenharia de Computação e Automação

1. INTRODUÇÃO

O presente relatório tem por objetivo analisar o funcionamento dos Multiplexadores, visto que, estes são construídos a partir de combinação de portas lógicas, então teremos as entradas no Mux e mediante um seletor teremos a saída indicada. Assim, apresentaremos os resultados obtidos de acordo a cada item esclarecido no roteiro, de modo a mostrar os circuitos lógicos e os códigos de descrição comportamental usados para a emulação dos projetos e respectivos resultados adquiridos via o software Quartus II.

2. METODOLOGIA

Para a realização deste laboratório foi necessário o conhecimento teórico a respeito do funcionamento das portas lógicas e multiplexadores. Desse modo, para a experimentação prática foi utilizado o software Quartus II, ele possibilita através de sua interface a construção de circuitos seja através do uso de portas lógicas ou codificação comportamental em VHDL, desse modo foram construídos os circuitos pedidos no roteiro e posteriormente simulação e assim obtidos os resultados que serão mostrados nas seções a seguir.

3. RESULTADOS

Nesta seção serão apresentados os resultados obtidos nas simulações realizadas, como também os meios utilizados para as obter no software.

3.1. MUX 2×1 COM CIRCUITOS LÓGICOS

No primeiro tópico foi pedida a construção de um Mux 2×1 com o uso de portas lógicas, desse modo, foi construído na interface do Quartus a combinação das portas e assim a construção do circuito lógico. Posterior a isso foi feita a execução da simulação mediante a imposição de valores lógicos e com isso foram obtidas as respostas que veremos após a imagem que descreve o circuito.

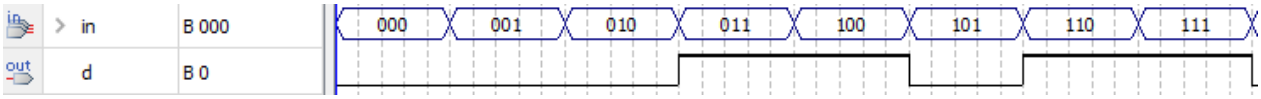

Figura 1 – Respostas do Mux 2×1 com circuitos lógicos.

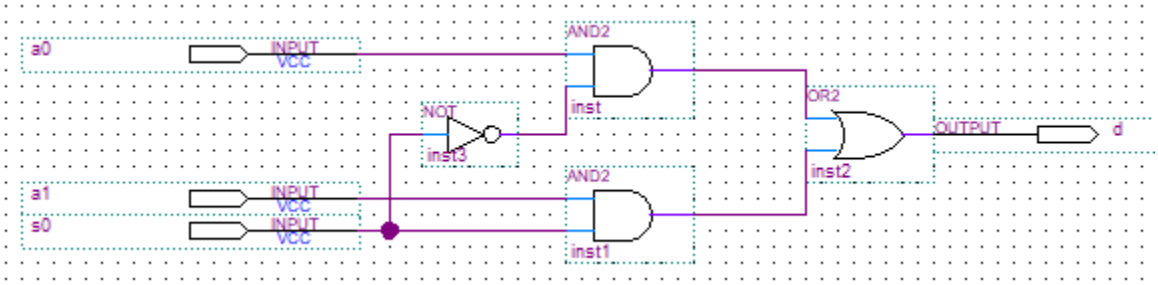

Figura 2 – Mux 2×1 com circuitos lógicos.

3.2. MUX 2×1 COM DESCRIÇÃO COMPORTAMENTAL

No segundo tópico foi pedida a construção do mesmo Mux 2×1 só que agora utilizando a descrição comportamental. Desse modo, foi construído o código em VHDL contendo a descrição comportamental a partir da arquitetura que compõe o Mux. Posterior a isso foi feita a execução da simulação mediante a imposição de valores lógicos e com isso foram obtidas as respostas que foram iguais às do circuito lógico, como era esperado e veremos nas imagens a seguir.

```
1  ENTITY Mux2x1 IS
2  PORT(I0,I1,s0 : IN BIT; -- No qual s0 é a porta de seleção
3  d : OUT BIT);
4  END;
5
6  ARCHITECTURE behav OF Mux2x1 IS
7  BEGIN
8  WITH s0 SELECT
9   d <= I0 WHEN '0',
10 I1 WHEN '1';
11 END;
12
```

Figura 3 – Mux 2×1 com descrição comportamental.

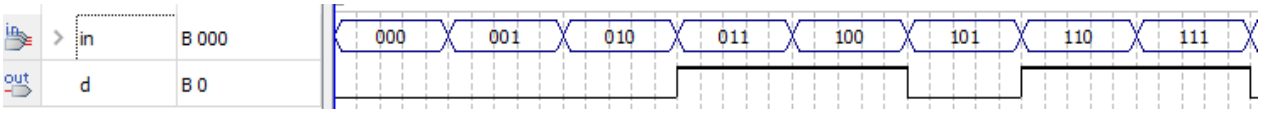

Figura 4 – Respostas do Mux 2×1 com descrição comportamental.

3.3. MUX 4×1 COM DESCRIÇÃO COMPORTAMENTAL

No terceiro tópico foi pedido a construção de um Mux 4×1 utilizando a descrição comportamental, como apresentado nas figuras 5 e 6.

```
1   ENTITY Mux4x1 IS
2   PORT(s : IN BIT_VECTOR(0 to 1);
3     I0, I1, I2, I3 : IN BIT;
4     d : OUT BIT);
5     END;
6
7   ARCHITECTURE behav OF Mux4x1 IS
8   BEGIN
9     WITH s SELECT
10     d <= I0 WHEN "00",
11    I1 WHEN "01",
12    I2 WHEN "10",
13    I3 WHEN "11";
14    END;
```

Figura 5 – Mux 4×1 com descrição comportamental.

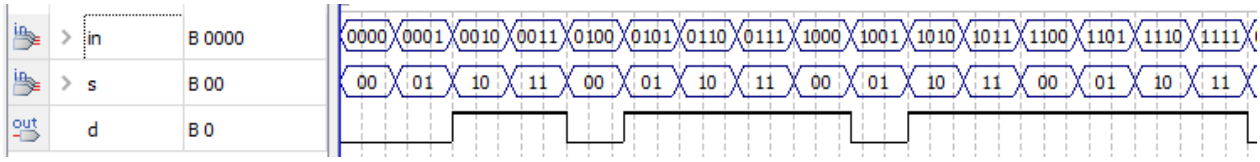

Figura 6 – Respostas do Mux 4×1 com descrição comportamental.

3.4. MUX 4×1 UTILIZANDO MUX 2×1

No quarto tópico foi pedido a construção de um Mux 4x1 em VHDL utilizando tanto a descrição comportamental quanto componentes. Para esse tópico ainda foi solicitado que, para a construção, deveria ser feito o uso de Mux 2x1.

```
1  entity mux_2x1 is
2      port(
3          i0_mux_2x1, i1_mux_2x1, s_mux_2x1: in bit;
4          d_mux_2x1: out bit
5      );
6  end;
7
8  architecture behav of mux_2x1 is
9  begin
10     with s_mux_2x1 select
11         d_mux_2x1 <= i0_mux_2x1 when '0',
12                      i1_mux_2x1 when '1';
13 end;
```

Figura 7 – Código do Mux 2×1.

```
1  entity mux_4x1 is
2      port(
3          i: in bit_vector(0 to 3);
4          s: in bit_vector(0 to 1);
5          d: out bit
6      );
7  end;
8
9  architecture behav of mux_4x1 is
10     signal mux_2x1_1: bit;
11     signal mux_2x1_2: bit;
12
13     component mux_2x1 is
14         port(
15             i0_mux_2x1, i1_mux_2x1, s_mux_2x1: in bit;
16             d_mux_2x1: out bit
17         );
18     end component;
19
20 begin
21
22     u1: mux_2x1 port map(i0_mux_2x1 => i(0), i1_mux_2x1 => i(1), s_mux_2x1 => s(1), d_mux_2x1 => mux_2x1_1);
23     u2: mux_2x1 port map(i0_mux_2x1 => i(2), i1_mux_2x1 => i(3), s_mux_2x1 => s(1), d_mux_2x1 => mux_2x1_2);
24
25     u3: mux_2x1 port map(i0_mux_2x1 => mux_2x1_1, i1_mux_2x1 => mux_2x1_2, s_mux_2x1 => s(0), d_mux_2x1 => d);
26
27 end;
```

Figura 8 – Código do Mux 4×1.

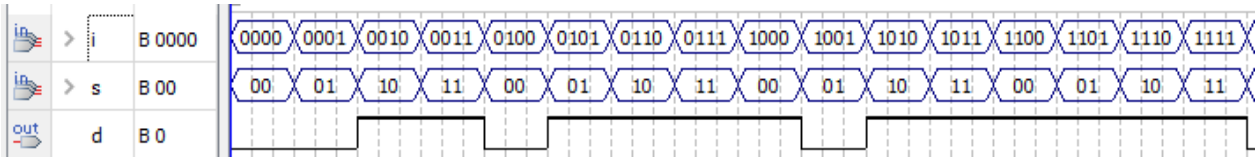

Figura 9 – Respostas do Mux 4×1.

4. CONCLUSÕES

A realização deste laboratório permitiu o melhor entendimento a respeito dos assuntos vistos nas aulas teóricas, colocando em prática a montagem de multiplexadores a partir de circuitos lógicos, vendo as variações de Mux que podemos obter e análises do funcionamento dos mesmos. Além disso, utilizamos da linguagem VHDL e em específico a descrição comportamental e variáveis do tipo bit vector nas simulações do ModelSim através do Quartus II.

5. REFERÊNCIAS

[1] QUARTUS II. Software de simulação.

[2] Vahid, Frank. Digital Design with RTL Design, VHDL, and Verilog Solution Manual. 2° Edição. 2010.