LABORATÓRIO 4 – REGISTRADORES

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 consolidar os conhecimentos adquiridos a respeito dos Latches, Flip-Flops e dos Registradores, dessa forma faremos o desenvolvimento e análise do comportamento desses circuitos digitais de memória. Assim, apresentaremos os resultados obtidos de acordo a cada item esclarecido no roteiro, de modo a mostrar os códigos usados para a emulação dos projetos e respectivos resultados adquiridos via o software Quartus.

2. METODOLOGIA

Para a realização deste laboratório foi necessário o conhecimento teórico a respeito do funcionamento dos Latches Flip-Flops junto também do conhecimento de registradores e sua construção. Desse modo, para a experimentação prática foi utilizado o software Quartus II, ele possibilita a construção de diversos circuitos digitais a partir do uso de portas lógicas, Latches, Flip-Flops e outros componentes que podem ser simulados com seu uso, com isso foi possível a construção dos códigos em VHDL utilizando a lógica dos códigos vista nas aulas.

3. RESULTADOS

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

3.1. LATCH D DE 1 BIT

Para o primeiro tópico foi pedida a construção de um Latch D de 1 bit, que é um dispositivo de armazenamento temporário que funcionará a partir de um clock que irá controlar a sua mudança de estado a cada pulso, assumindo o valor do sinal D por exemplo quando o clock está em nível lógico 1, como veremos na implementação a seguir do código e seu respectivo resultado.

```
1  entity latch_D is
2  port(
3      D, clk: in bit;
4      Q, Q_linha : out bit
5  );
6  end;
7
8  architecture behav of latch_D is
9  begin
10 process(D, clk)
11 begin
12    if(clk = '1' and D = '1') then
13       Q <= '1';
14       Q_linha <= '0';
15    elsif(clk = '1' and D = '0') then
16       Q <= '0';
17       Q_linha <= '1';
18    end if;
19
20 end process;
21 end;
```

Figura 1 - Código VHDL para o Latch D.

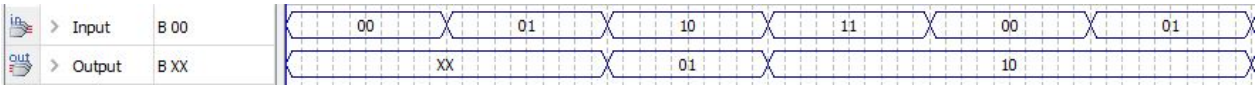

Figura 2 - Resultado da execução para o Latch D.

3.2. FLIP-FLOP DE 1 BIT

Para o segundo tópico foi pedida a construção de um Flip-Flop de 1 bit, esse dispositivo funciona de forma análoga ao Latch, a diferença aqui observada no Flip-Flop é que ele assumirá o valor de um sinal D por exemplo, somente na borda de subida do sinal do clock, sendo assim, ele não mudará de estado mediante as mudanças de estado de D durante o pulso de clock atual, ele só atualizará quando houver uma nova borda de subida do pulso de clock. A seguir mostraremos como foi feito o código em VHDL e sua execução.

```
1  ENTITY FlipFlopD IS
2  PORT (D, clk : IN BIT;
3             Q : OUT BIT);
4  END;
5  ARCHITECTURE behav OF FlipFlopD IS
6  BEGIN
7  PROCESS (D, clk)
8  BEGIN
9     IF(clk ' EVENT and clk='1' and D='1') THEN
10       Q <= '1';
11    ELSIF(clk ' EVENT and clk='1' and D='0') THEN
12       Q <= '0';
13    END IF;
14 END PROCESS;
15 END;
```

Figura 3 - Código VHDL para o Flip-Flop.

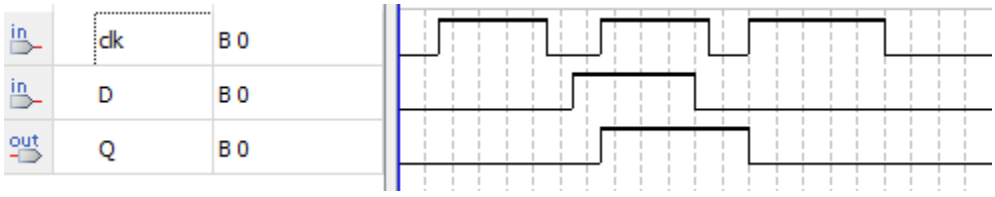

Figura 4 - Resultado da execução para o Flip-Flop.

3.3. REGISTRADOR DE 4 BITS BASEADOS EM LATCHES D

Para este tópico foi pedido a representação de um registrador de 4 bits a partir do uso de latches D, para tal como já havíamos desenvolvido o código para o Latch D de 1 bit utilizamos o comando COMPONENT e fizemos a instanciação utilizando o PORT MAP para a construção do registrador a partir do uso de 4 Latches já que cada um é de 1 bit. A seguir mostraremos o código utilizado para a implementação e os respectivos resultados da execução que vamos dividir em 4 figuras para a melhor visualização dos mesmos.

```
1   entity registrador is
2   port(
3       I0, I1, I2, I3, clk_2: in bit;
4       Q0, Q1, Q2, Q3: out bit
5   );
6   end;
7
8   architecture behav of registrador is
9       component latch_D is
10          port(
11              D, clk: in bit;
12              Q, Q_linha : out bit
13          );
14      end component;
15
16  begin
17      u1: latch_D port map(D => I0, clk => clk_2, Q => Q0);
18      u2: latch_D port map(D => I1, clk => clk_2, Q => Q1);
19      u3: latch_D port map(D => I2, clk => clk_2, Q => Q2);
20      u4: latch_D port map(D => I3, clk => clk_2, Q => Q3);
21  end;
```

Figura 5 – Código VHDL para o Registrador com Latches.

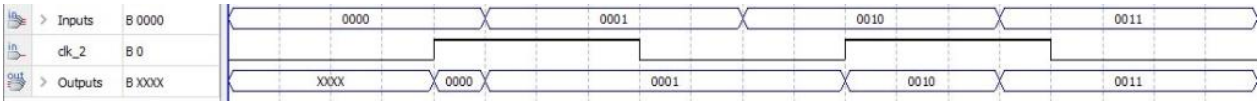

Figura 6 – Resultado da execução para o Registrador (Parte 1) .

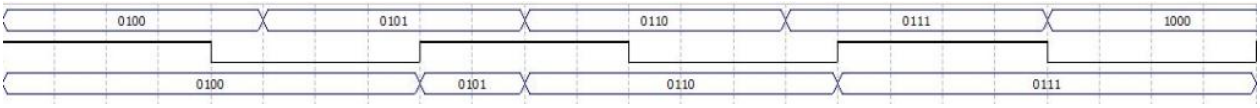

Figura 7 – Resultado da execução para o Registrador (Parte 2) .

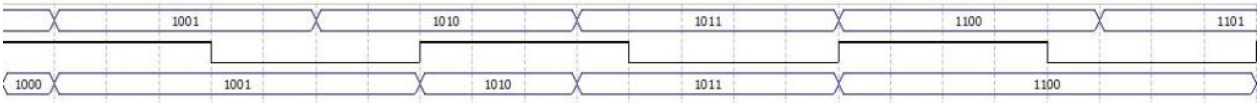

Figura 8 – Resultado da execução para o Registrador (Parte 3) .

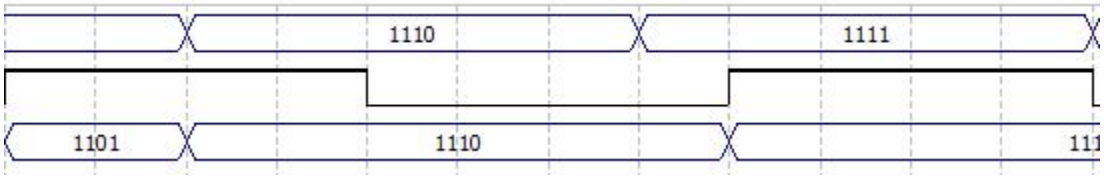

Figura 9 – Resultado da execução para o Registrador (Parte 4) .

3.4. REGISTRADOR DE 4 BITS BASEADOS EM FLIP-FLOPS D

Para o 4 e último item foi pedida a construção de um registrador de 4 bits a partir do uso de Flip-Flops, para tal como já havíamos também desenvolvido o código para Flip-Flip de 1 bit no item 3.2 então utilizamos novamente o comando COMPONENT e fizemos a instanciação utilizando o PORT MAP para a construção do registrador a partir do uso de 4 Flip-Flops de 1 bit. A seguir mostraremos o código utilizado para a implementação e os respectivos resultados da execução do mesmo.

```
1  ENTITY Registrador IS
2  PORT (clk_2, I1, I2, I3, I0 : IN BIT;
3             Q1, Q2, Q3, Q0 : OUT BIT);
4  END;
5  ARCHITECTURE behav OF Registrador IS
6     COMPONENT FlipFlopD IS
7        PORT(
8           D, clk : IN BIT;
9                Q : OUT BIT
10       );
11    END COMPONENT;
12 BEGIN
13    U1: FlipFlopD PORT MAP (D => I0, clk => clk_2, Q => Q0);
14    U2: FlipFlopD PORT MAP (D => I1, clk => clk_2, Q => Q1);
15    U3: FlipFlopD PORT MAP (D => I2, clk => clk_2, Q => Q2);
16    U4: FlipFlopD PORT MAP (D => I3, clk => clk_2, Q => Q3);
17 END;
```

Figura 10 – Código VHDL para o Registrador com Flip-Flops.

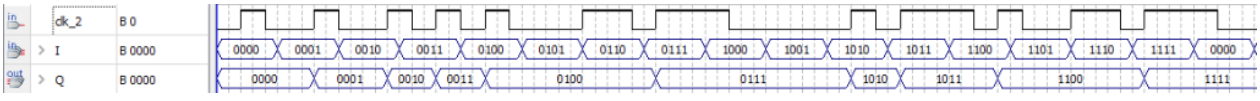

Figura 11 – Resultado da execução para o Registrador com Flip-Flops.

4. CONCLUSÕES

A realização deste laboratório permitiu o melhor entendimento a respeito dos assuntos vistos nas aulas teóricas, a partir do desenvolvimento dos Latches e Flip-Flops em linguagem VHDL foi possível observar de forma prática o seu funcionamento e a diferença entre o funcionamento entre eles, como também a possibilidade de construir Registradores com ambos os componentes mediante a necessidade dada. Para tal, utilizamos da linguagem VHDL e simulações do ModelSim através do Quartus II e os conceitos vistos em laboratórios anteriores para a definição de componentes e instanciação para obtermos os códigos necessários para a execução.

5. REFERÊNCIAS

[1] QUARTUS II. Software de simulação.

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