LABORATÓRIO 5 – CONTADORES

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 Contadores e do seu funcionamento, dessa forma faremos o desenvolvimento e análise do comportamento dos circuitos digitais por eles compostos. Assim, apresentaremos os resultados obtidos de acordo com 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 do Contadores como também o conhecimento de registradores para a sua construção, tanto os incrementadores como os decrementadores. 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. CONTADOR CRESCENTE

Para o primeiro tópico foi solicitado a implementação em código VHDL de um contador crescente, que é um dispositivo de incrementação que funciona adicionando 1 ao seu valor a cada ciclo de clock. Como veremos na implementação a seguir do código e sua respectiva simulação.

```
1  ENTITY contador IS
2  PORT(
3    clk: IN BIT; --entrada de clock
4    ld: IN BIT; --carrega os dados
5    reset: IN BIT;
6    data: IN INTEGER RANGE 15 DOWNTO 0; -- entrada de dados
7    q: OUT INTEGER RANGE 15 DOWNTO 0); --saída de dados
8  END contador;
9  ARCHITECTURE comportamento OF contador IS
10   BEGIN PROCESS(clk,reset)
11   VARIABLE qv: INTEGER RANGE 15 DOWNTO 0; --variável para a saida
12
13   BEGIN
14   IF(reset = '1') THEN
15   qv := 0;
16   ELSIF(clk ' event and clk = '1') THEN
17   IF(ld = '1') THEN
18   qv := data;
19   ELSE
20   IF(qv >= 9) THEN
21   qv := 0;
22   ELSE
23   qv := qv + 1;
24   END IF;
25   END IF;
26   END IF;
27   q <= qv;
28   END PROCESS;
29 END;
```

Figura 1 - Código VHDL para contador crescente.

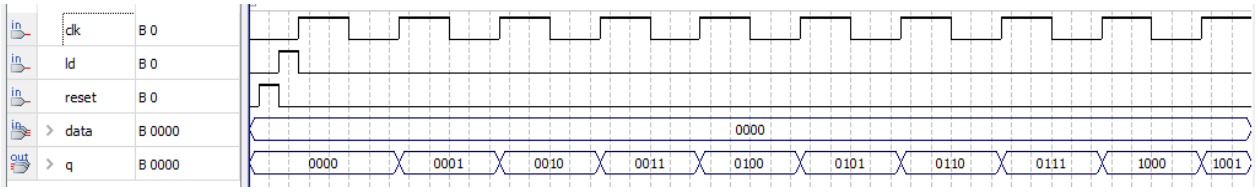

Figura 2 - Resultado da simulação para o contador crescente.

3.2. CONTADOR CRESCENTE DE 4 BITS

Para o segundo tópico foi pedida a construção de um contador crescente de 4 bits que fizesse a contagem até o último valor e retornasse ao valor inicial. Como o contador é de 4 bits, o último valor possível é 15. Além disso, foi necessário adicionar um sinal “tc” para indicar o término da contagem. A imagem a seguir mostra a implementação em VHDL do solicitado pelo tópico.

```
1  entity Contador is
2  port(
3      clk: in bit;
4      ld: in bit;
5      reset: in bit;
6      data: in integer range 15 downto 0;
7      q: out integer range 15 downto 0;
8      tc: out bit
9  );
10 end Contador;
11 architecture comportamento of Contador is
12    begin process(clk, reset)
13       variable qv: integer range 15 downto 0;
14       begin
15          if(reset = '1') then
16             qv:= 0;
17          elsif(clk ' event and clk = '1') then
18             if(ld = '1') then
19                qv := data;
20             else
21                if(qv = 14) then
22                   qv := qv + 1;
23                   tc <= '1';
24                elsif(qv <= 15) then
25                   qv := qv + 1;
26                   tc <= '0';
27                else
28                   qv := 0;
29                end if;
30             end if;
31          end if;
32       q <= qv;
33    end process;
34 end;
```

Figura 3 - Código VHDL para o contador crescente de 4 bits.

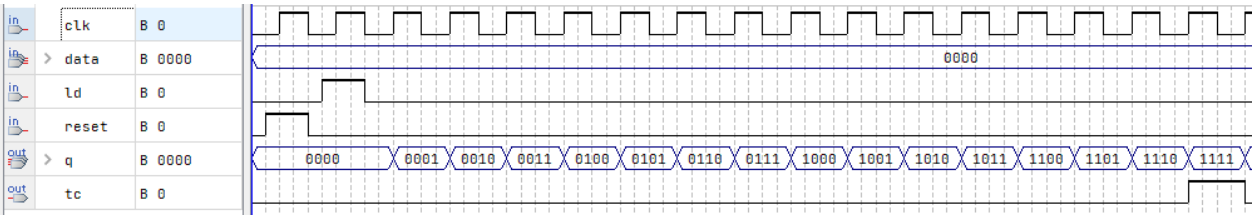

Figura 4 - Resultado da simulação para o contador crescente de 4 bits.

3.3. CONTADOR DECRESCENTE DE 4 BITS

Para este tópico foi pedido a representação de um contador decrescente de 4 bits, esse realizará a contagem partindo de 15 e irá decrescer uma vez a cada ciclo de clock até chegar ao valor mínimo e após isso retornar para seu valor inicial. Além disso, assim como no contador crescente, foi necessário adicionar um sinal “tc” para indicar o término da contagem. A seguir mostraremos o código utilizado para a implementação e os respectivos resultados da execução.

```
1  ENTITY contador IS
2  PORT(
3    clk: IN BIT; --entrada de clock
4    ld: IN BIT; --carrega os dados
5    reset: IN BIT;
6    data: IN INTEGER RANGE 15 DOWNTO 0; -- entrada de dados
7    q: OUT INTEGER RANGE 15 DOWNTO 0; --saída de dados
8    tc: OUT BIT);
9  END contador;
10 ARCHITECTURE comportamento OF contador IS
11  BEGIN PROCESS(clk,reset)
12   VARIABLE qv: INTEGER RANGE 15 DOWNTO 0; --variável para a saida
13   BEGIN
14  IF(reset = '1') THEN
15   qv := data;
16  ELSIF(clk ' event and clk = '1') THEN
17  IF(ld = '1') THEN
18   qv := data;
19  ELSE
20  IF(qv /= 1) THEN
21   qv := qv - 1;
22   tc <= '0';
23  ELSE
24   qv := 0;
25   tc <= '1';
26   END IF;
27   END IF;
28   END IF;
29   q <= qv;
30   END PROCESS;
31 END;
```

Figura 5 – Código VHDL para o contador decrescente de 4 bits.

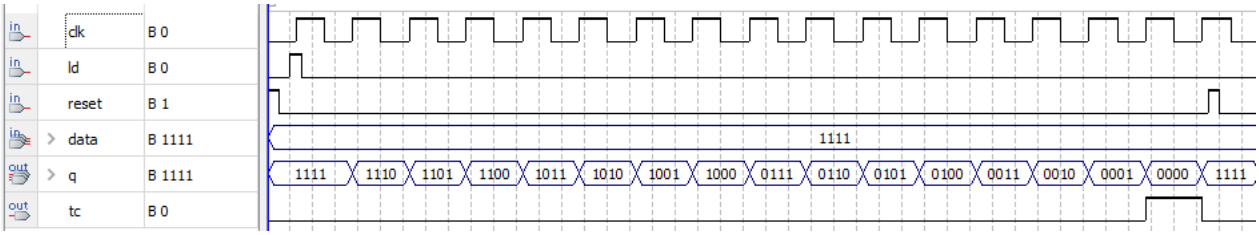

Figura 6 – Resultado da simulação para o contador decrescente de 4 bits.

4. CONCLUSÕES

A realização deste laboratório permitiu o melhor entendimento a respeito dos assuntos vistos nas aulas teóricas, a partir do desenvolvimento dos contadores crescentes e decrescentes em linguagem VHDL foi possível observar de forma prática o funcionamento dos contadores, como também a possibilidade de construí-los com o uso de registradores mediante a necessidade dada. Para tal, utilizamos da linguagem VHDL e simulações do ModelSim através do Quartus II e os conceitos adequados 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.