CIRCUITOS CONTADORES

Júlia Pessoa Souza

Teoria de Eletrônica Digital-Turma C
Faculdade Gama - Universidade de Brasília
15/0133294
Laboratório 7

João Pedro Vergara Coneglian

Teoria de Eletrônica Digital-Turma C
Faculdade Gama - Universidade de Brasília
14/0145974
Laboratório 7

RESUMO

Para este projeto, foi criado um circuito contador crescente/decrescente de módulo 10 utilizando VHDL no programa VIVADO com o objetivo de mostrar na placa Basys 3 as saídas deste mesmo contador, dependendo apenas da entrada predefinida. Se a entrada fosse 1, o contador realizaria uma contagem crescente. Se 0, a contagem seria decrescente.

1. INTRODUÇÃO

Os circuitos contadores são circuitos digitais sequenciais que evoluem sob o comando de um clock, de forma que seus estados reproduzam uma sequência pré-determinada. Contadores digitais são utilizados principalmente para contagens, geração de palavras, divisão e medição de frequências e tempo. São divididos em duas categorias: contadores síncronos e assíncronos.

2. EXPERIMENTO

2.1. Vivado

Foi criado um novo arquivo no Vivado para fazer o projeto. Dividiu-se o código em 4 partes: uma para o contador, outra para o codificador do display, outra para o divisor do clock e a última para a junção de todos. O contador tem o reset de entrada, este serve para zerar a contagem. Também tem a entrada enable, que vai determinar se o contador é crescente ou decrescente. A entrada clock faz com que o contador espere a próxima subida para mudar a contagem, e a saída vai ser mostrada no display. O divisor de clock foi feito de uma maneira que torna possível perceber a mudança de números no contador. Com este código a frequência do clock da placa foi diminuída para 3Hz. Foi utilizado um contador que vai até o número calculado para a frequência. Este número foi calculado dividindo-se a frequência da placa (100MHz) pela frequência desejada (3Hz), e, dividindo-se o resultado por dois. Toda vez que o contador atinge este resultado ele zera e o novo clock troca de bit, e assim o processo continua. O código do codificador foi criado com a mesma base já feita em laboratório anteriormente. Este, determina como o display deve ficar dependendo do número que será mostrado. Já o código principal junta todos utilizando seus componentes e o port map.

2.2. Implementação

Utilizando o arquivo Basys 3 Master, foram retirados os comentários da parte do clock. Na parte do switch foram utilizadas duas chaves para o enable e o reset. E, na parte do display foram colocadas as saídas S0 a S6. Após feita a síntese, implementação e o bitstream, a placa recebeu os comandos e começou a contagem.

2.3. Figuras e Tabelas

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.STD_LOGIC_ARITH.ALL;
4
5
6  entity lab7 is
7      Port ( enable : in STD_LOGIC;
8             clock : in STD_LOGIC;
9             reset : in STD_LOGIC;
10            q : out STD_LOGIC_VECTOR(3 DOWNTO 0));
11 end lab7;
12
13 architecture Behavioral of lab7 is
14 begin
15
16  process (clock, reset)
17  variable contagem: integer range 0 to 15;
18  begin
19  if reset = '1' then
20     contagem:= 0;
21  elsif clock'event and clock='1' then
22     if enable='1' then
23        contagem := contagem + 1;
24        elsif enable = '0' then
25        contagem := contagem - 1;
26     end if;
27  end if;
28  q <= conv_std_logic_vector(contagem, 4);
29 end process;
30 end Behavioral;
```

Figura 1: Código para contador.

C:/Users/joped/project_14/project_14.srcs/sources_1/new/lab73.vh

```
library ieee;
use IEEE.STD_LOGIC_1164.ALL;
--use IEEE.STD_LOGIC_ARITH.ALL;
--use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity clock_div is
port(
  Clk  : in std_logic;
  X : out std_logic);
end clock_div;

architecture teste of clock_div is
signal clk_dividido: STD_LOGIC:='0';
signal count: INTEGER:=0;
begin
  process(clK)
  begin

    if RISING_EDGE(clk) then
      if count = 14999999 then
      count <= 0;
      clk_dividido <= not clk_dividido;
      else
      count <= count + 1;
    end if;
    end if;
    X <= clk_dividido;
    end process;

end teste;
```

Figura 2: Código do divisor do clock.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity lab71 is
    Port ( q : in STD_LOGIC_VECTOR (3 DOWNTO 0);
           S : out STD_LOGIC_VECTOR (6 DOWNTO 0));
end lab71;

architecture Behavioral of lab71 is

begin
process(q)
begin
    case q is
        when "0000" => S <= "1000000";
        when "0001" => S <= "1111001";
        when "0010" => S <= "0100100";
        when "0011" => S <= "0110000";
        when "0100" => S <= "0011001";
        when "0101" => S <= "0010010";
        when "0110" => S <= "0000010";
        when "0111" => S <= "1111000";
        when "1000" => S <= "0000000";
        when "1001" => S <= "0011000";
        when "1010" => S <= "0001000";
        when "1011" => S <= "1000110";
        when "1100" => S <= "1000110";
        when "1101" => S <= "0100001";
        when "1110" => S <= "0000110";
        when "1111" => S <= "0001110";
        when others => S <= "1111111";
     end case;
end process;


end Behavioral;
```

Figura 3: Código do codificador.

```
1
2  library IEEE;
3  use IEEE.STD_LOGIC_1164.ALL;
4  entity toplevel is
5   Port ( enable : in STD_LOGIC;
6          clock : in STD_LOGIC;
7          reset : in STD_LOGIC;
8          S : out STD_LOGIC_VECTOR(6 DOWNTO 0));
9  end toplevel;
10
11 architecture Behavioral of toplevel is
12
13 signal q: STD_LOGIC_VECTOR(3 DOWNTO 0);
14 signal X: STD_LOGIC;
15 component lab7 is
16  Port ( enable : in STD_LOGIC;
17           clock : in STD_LOGIC;
18           reset : in STD_LOGIC;
19           q : out STD_LOGIC_VECTOR(3 DOWNTO 0));
20 end component;
21
22 component lab71 is
23     Port ( q : in STD_LOGIC_VECTOR (3 DOWNTO 0);
24        S : out STD_LOGIC_VECTOR (6 DOWNTO 0));
25 end component;
26
27 component clock_div is
28 port(
29   Clk  : in std_logic;
30   X : out std_logic);
31 end component;
32
33 begin
34
35 A1: lab71 port map ( S => S, q => q);
36 A2: lab7 port map(enable => enable, clock => X, reset => reset, q => q);
37 A3: clock_div port map(Clk => clock, X => X);
38
39 end Behavioral;
```

Figura 4: Código principal.

```
6  ## Clock signal
7  set_property PACKAGE_PIN W5 [get_ports clock]
8      set_property IOSTANDARD LVCMOS33 [get_ports clock]
9      create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clock]
0
```

Figura 5: Código de clock para Basys 3.

```
9  #    set_property IOSTANDARD LVCMOS33 [get_ports {q[3]}]
0  set_property PACKAGE_PIN W15 [get_ports {reset}]
1      set_property IOSTANDARD LVCMOS33 [get_ports {reset}]
2  set_property PACKAGE_PIN V15 [get_ports {enable}]
3      set_property IOSTANDARD LVCMOS33 [get_ports {enable}]
4  #set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
5      #set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
```

Figura 6: Switches na Basys 3.

```
1  ##7 segment display
2  set_property PACKAGE_PIN W7 [get_ports {S[0]}]
3      set_property IOSTANDARD LVCMOS33 [get_ports {S[0]}]
4  set_property PACKAGE_PIN W6 [get_ports {S[1]}]
5      set_property IOSTANDARD LVCMOS33 [get_ports {S[1]}]
6  set_property PACKAGE_PIN U8 [get_ports {S[2]}]
7      set_property IOSTANDARD LVCMOS33 [get_ports {S[2]}]
8  set_property PACKAGE_PIN V8 [get_ports {S[3]}]
9      set_property IOSTANDARD LVCMOS33 [get_ports {S[3]}]
0  set_property PACKAGE_PIN U5 [get_ports {S[4]}]
1      set_property IOSTANDARD LVCMOS33 [get_ports {S[4]}]
2  set_property PACKAGE_PIN V5 [get_ports {S[5]}]
3      set_property IOSTANDARD LVCMOS33 [get_ports {S[5]}]
4  set_property PACKAGE_PIN U7 [get_ports {S[6]}]
5      set_property IOSTANDARD LVCMOS33 [get_ports {S[6]}]
6
```

Figura 7: Display na Basys 3.

3. RESULTADOS

Na figura 1, mostra-se o código em VHDL utilizado para criar-se um contador no programa VIVADO que atenda às necessidades do projeto (contador crescente/decrescente de módulo 10). Na segunda imagem, percebe-e como o clock da placa foi dividido para se ter uma frequência menos para a contagem poder ser observada no display. Já na terceira imagem, tem-se o codificador que passa o número da contagem para o display de 7 segmentos já utilizado anteriormente.Os códigos se juntam na figura 4. Na placa Basys 3 foram utilizadas duas chaves para o enable e o reset, foi necessário utilizar o clock, já que este foi modificado, e o display de 7 segmentos. Tem-se o resultado nas figuras 5, 6 e 7.

4. DISCUSSÃO E CONCLUSÕES

O experimento obteve os resultados esperados ao mostrar na placa Basys 3 o display sendo modificado de acordo com a contagem. Quando o enable foi ativado a contagem era crescente e quando o enable foi desativado a contagem era decrescente. Quando o reset foi ativado a contagem foi zerada.Os números no display foram modificados de 3 em 3 segundos. E, assim, pode-se concluir que o experimento obteve sucesso.

5. REFERÊNCIAS

[1] I. Idoeta e I. Valeije, “Elementos de Eletrônica Digital”, Érica Ltda, São Paulo, 2008.

[2] R. Tocci, N.Widmer e G. Moss, “Sistemas Digitais: Princípios e aplicações” Pearson.

[3] G. Bezerra, “Apostila Prática de Eletrônica Digital I”, 2017

[4] G. Bezerra, “Tutorial de Simulação e Implementação no Vivado”, 2017