Relógio Digital VHDL no Quartus ||

João Pedro Peres Bertoncelo RA112650

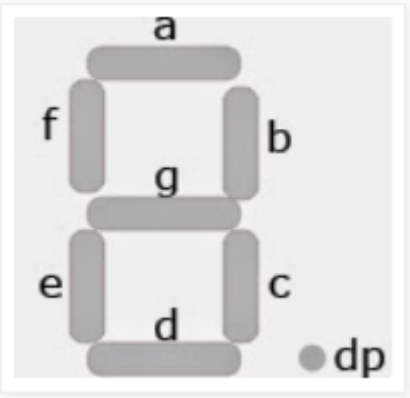
1. Funcionamento do Circuito

Este circuito foi desenvolvido na linguagem VHDL com o software "Quartus 2", de acordo com as orientações da disciplina de Circuitos Digitais 2.

a. Funcionamento Relógio

O funcionamento do circuito é baseado em um package e uma main, onde, no package é feita a conversão dos números de 0 a 9 para o binário que represente aquele número com os leds ligados

Os leds são representados da seguinte forma:



quando um led está ligado, é representado por 0, e desligado, por 1, ou seja, para fazermos o número 7, teríamos:

|G|F|E|D|C|B|A|
|1|0|1|1|0|0|0|

b. Funcionamento do Código

Na main, é declarado o número de clocks para contar um segundo, declarado como “generic”, e depois, declaramos as entradas, que são as 4 variáveis “hex”(que serão os nossos algarismos, então, são declaradas como vetores lógicos de tamanho 7, para receberem a entrada do package após a conversão), e o nosso Clock, que contará os segundos.

Na arquitetura, são declaradas as variáveis “ TEMPORAL, SS0, HH0,HH1,MM0,MM1” que serão responsáveis por contar segundos, horas e minutos respectivamente, logo após, a função do package é chamada para cada variável, para que o valor das variáveis de tempo sejam convertidas para seu valor binário, de forma que os leds sejam ativados corretamente, já a temporal, será para fazer a contagem de números de clocks para 1 segundo.

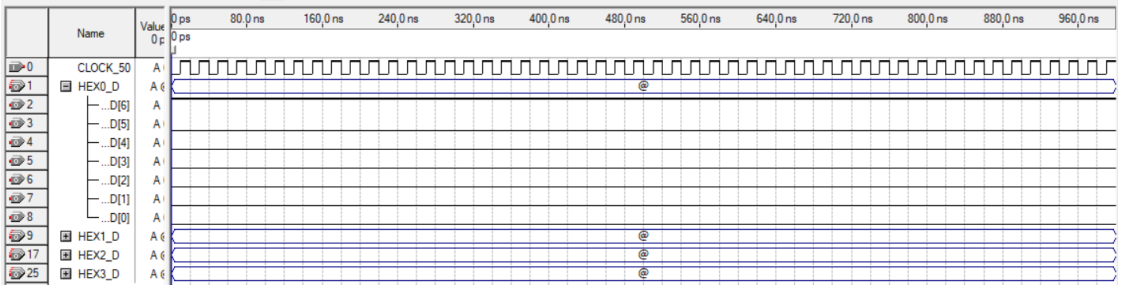
É utilizado um “process” com o clock como parâmetro, para que seja um código sequencial, e o próximo ciclo seja afetado pelo ciclo anterior, tendo uma continuidade no relógio.

Utilizamos condições para conferir se o clock já atingiu o valor para contarmos um segundo, se o valor para 1 segundo ainda não foi atingido, incrementamos 1 na variável “TEMPORAL” e o process é reiniciado, se foi, a variável “TEMPORAL” é resetada para 0 e incrementa a variável de segundo, então, caso a variável de segundo chegue em 60, ela é resetada para 0 e a de minuto é incrementada.

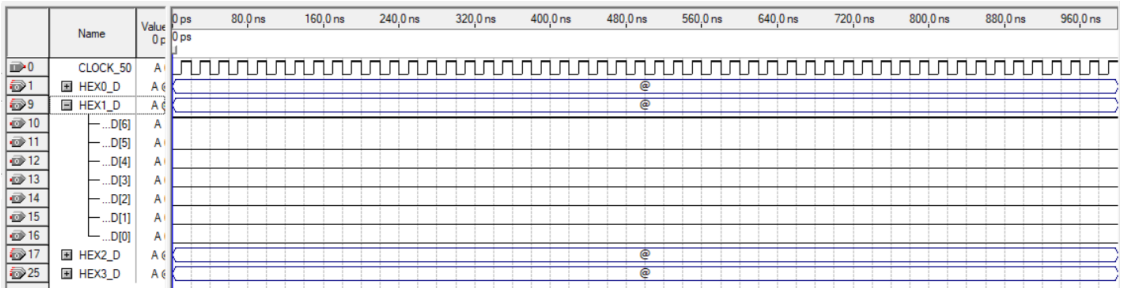
Assim como quando a variável de minutos passa de 9 e assim por diante, isso fará com que o relógio sempre vá atualizando os horários corretamente, até 99:99, finalizando o relógio

c. WaveForm(1 us)

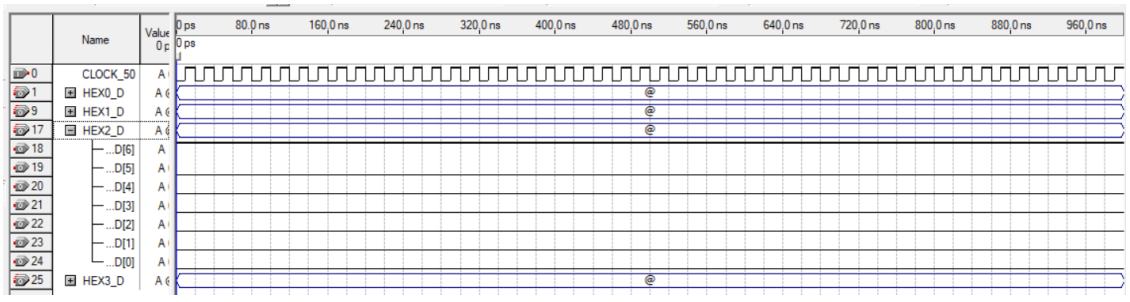
HEX0:



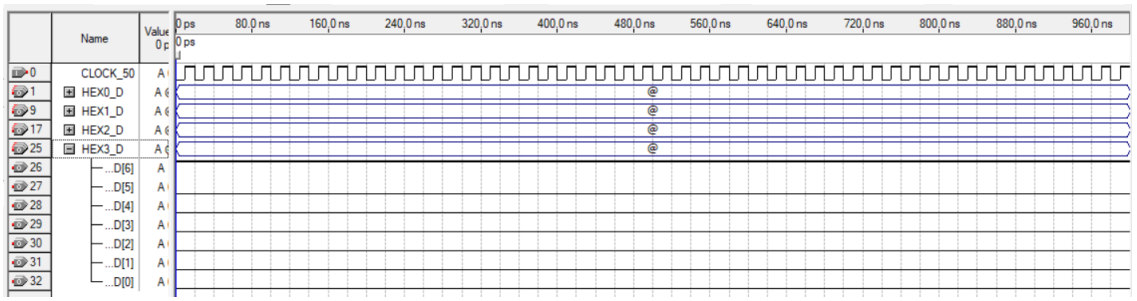
HEX1:



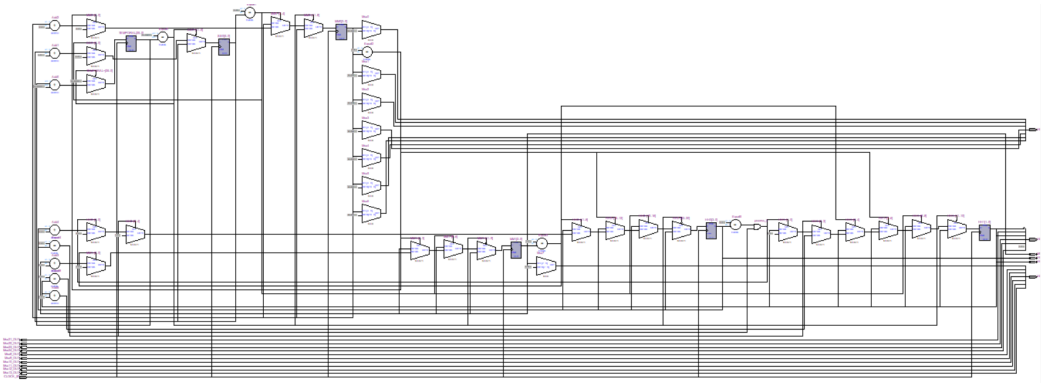
HEX2:



HEX3:



d. Circuito Final



2. Código

a. Main

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
USE work.my_package.all;

ENTITY relogio_digital IS
	GENERIC (SEG: INTEGER := 49999999);
	PORT(
		HEX0_D  :OUT STD_LOGIC_VECTOR (6 downto 0);
		HEX1_D  :OUT STD_LOGIC_VECTOR (6 downto 0);
		HEX2_D  :OUT STD_LOGIC_VECTOR (6 downto 0);
		HEX3_D  :OUT STD_LOGIC_VECTOR (6 downto 0);
		CLOCK_50 :IN STD_LOGIC
	);
END relogio_digital;

ARCHITECTURE myarch OF relogio_digital IS
 SIGNAL TEMPORAL: INTEGER RANGE 0 TO SEG;

 SIGNAL SS0: INTEGER RANGE 0 TO 59 := 0;
 SIGNAL HH0: INTEGER RANGE 0 TO 9 := 0;
 SIGNAL HH1: INTEGER RANGE 0 TO 2 := 0;
 SIGNAL MM0: INTEGER RANGE 0 TO 9 := 0;
 SIGNAL MM1: INTEGER RANGE 0 TO 5 := 0;
```

```
BEGIN
 HEX0_D <= int2seg(MM0);
 HEX1_D <= int2seg(MM1);
 HEX2_D <= int2seg(HH0);
 HEX3_D <= int2seg(HH1);

 PROCESS(CLOCK_50)
 BEGIN
  IF rising_edge(CLOCK_50) THEN
   IF(TEMPORAL /= SEG) THEN
    TEMPORAL <= TEMPORAL + 1;
   ELSE
    TEMPORAL <= 0;
    SS0 <= SS0 + 1;
    IF(SS0 = 59) THEN
     SS0 <= 0;
     MM0 <= MM0 + 1;
     IF(MM0 = 9) THEN
      MM0 <= 0;
      MM1 <= MM1 + 1;
      IF(MM1 = 5) THEN
       MM1 <= 0;
       HH0 <= HH0 + 1;
       IF(HH0 = 9) THEN
        HH0 <= 0;
        HH1 <= HH1 + 1;
        ELSIF (HH0 = 3 AND HH1 = 2) THEN
         HH1 <= 0;
         HH0 <= 0;
         MM1 <= 0;
         MM0 <= 0;
       END IF;
      END IF;
     END IF;
    END IF;
   END IF;
  END IF;
 END PROCESS;
END myarch;
```

b. Package

```
USE ieee.numeric_std.all;
PACKAGE my_package IS
        FUNCTION int2seg(A: INTEGER) RETURN STD_LOGIC_VECTOR;
END my_package;

PACKAGE BODY my_package IS
        FUNCTION int2seg(A: INTEGER) RETURN STD_LOGIC_VECTOR IS
                VARIABLE result: STD_LOGIC_VECTOR(6 downto 0);
        BEGIN
                CASE A IS
```

```
                    WHEN 0 => result := "1000000";
                    WHEN 1 => result := "1111001";
                    WHEN 2 => result := "0100100";
                    WHEN 3 => result := "0110000";
                    WHEN 4 => result := "0011001";
                    WHEN 5 => result := "0010010";
                    WHEN 6 => result := "0000010";
                    WHEN 7 => result := "1011000";
                    WHEN 8 => result := "0000000";
                    WHEN 9 => result := "0010000";
                    WHEN OTHERS => result := (OTHERS=>'0');
                END CASE;
        RETURN result;
        END int2seg;
END my_package ;
```

3. Referências

Site Blogger.com:
http://infraled.blogspot.com/2014/06/vhdl-tutorial-2-relogio-digital.html