Universidade Tecnológica Federal do Paraná
Engenharia da Computação
Lógica Reconfigurável

Relatório 7 - Timer

Aluno: Eduardo Yuji Yoshida Yamada
Professor orientador: Marcelo de Oliveira

Janeiro
2025

Conteúdo

1 Introdução

Este relatório apresenta o desenvolvimento de um *timer* regressivo utilizando *displays* de sete segmentos (SSDs) para exibir o tempo em segundos. O sistema conta o tempo de forma decrescente até atingir zero, momento em que um LED é acionado como indicação visual. Para a correta exibição dos números nos SSDs, foi implementada uma função de conversão de valores inteiros para *Binary-Coded* Decimal (BCD), permitindo a representação adequada dos dígitos.

A conversão para BCD foi baseada em algoritmos eficientes, como o *double dabble*, garantindo a correta tradução dos valores binários para a forma decimal apresentada nos SSDs. Além disso, o projeto inclui funcionalidades adicionais para aprimorar a usabilidade, como um botão de *reset* assíncrono, um botão para pausar ou desabilitar a contagem e diferentes opções de tempos predefinidos para a contagem regressiva.

O desenvolvimento deste sistema envolve conceitos fundamentais de eletrônica digital, como contadores, multiplexação de *displays* de sete segmentos e manipulação de sinais binários. Assim, o projeto não apenas atende aos requisitos técnicos propostos, mas também proporciona uma experiência prática no trabalho com hardware digital e sistemas embarcados.

2 Códigos

Foi implementado o seguinte código para a realização da atividade:

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

-- Definicao da entidade
entity projeto7 is
    Port (
        clk          : in  STD_LOGIC;
           -- Entrada de clock (sinal de clock)
        reset        : in  STD_LOGIC;
           -- Entrada de reset
        pause        : in  STD_LOGIC;
           -- Entrada de controle de pausa
        time_select_1 : in  STD_LOGIC;
           -- Switch para selecao de tempo (SW8)
        time_select_2 : in  STD_LOGIC;
           -- Switch para selecao de tempo (SW9)
        led          : out STD_LOGIC;
           -- LED indicador (acionado quando o tempo se
           esgota)
        ssd_unit     : out STD_LOGIC_VECTOR (6 downto 0)
           ; -- Display de 7 segmentos para as unidades
        ssd_tens     : out STD_LOGIC_VECTOR (6 downto 0)
           ; -- Display de 7 segmentos para as dezenas
        ssd_hundreds : out STD_LOGIC_VECTOR (6 downto 0)
           ; -- Display de 7 segmentos para as centenas
        ssd_thousands: out STD_LOGIC_VECTOR (6 downto 0)
             -- Display de 7 segmentos para os milhares
    );
end projeto7;

architecture projeto7 of projeto7 is

    -- Sinal para o contador de tempo, que ira de 0 ate
       9999 (usado para contagem regressiva)
    signal time_left     : INTEGER range 0 to 9999 :=
       60;
    -- Sinal para o contador de clock que ira dividir o
       clock de entrada para gerar os ciclos de contagem
```

```
signal clk_div_count : INTEGER := 0;
-- Sinal de controle de pausa
signal paused       : STD_LOGIC := '0';

-- Definicao de um tipo para representar os digitos
   BCD (Unidade, Dezena, Centena e Milhar)
type bcd_digits is array (0 to 3) of INTEGER;
-- Sinal que armazena os valores BCD correspondentes
    ao tempo
signal bcd_values : bcd_digits := (others => 0);

-- Funcao para converter o valor inteiro para BCD
function to_bcd(val : INTEGER) return bcd_digits is
    variable temp   : INTEGER := val;    -- Variavel
        temporaria para armazenar o valor em
       processamento
    variable result : bcd_digits := (others => 0);
       -- Vetor que armazena o valor em BCD
begin
    -- Converte o valor para as posicoes de Unidade,
        Dezena, Centena e Milhar
    result(0) := temp mod 10;       -- Unidade
    temp := temp / 10;
    result(1) := temp mod 10;       -- Dezena
    temp := temp / 10;
    result(2) := temp mod 10;       -- Centena
    temp := temp / 10;
    result(3) := temp;              -- Milhar
    return result;  -- Retorna o valor em formato
       BCD
end function;

-- Funcao para mapear os digitos BCD para o formato
   do display de 7 segmentos
function ssd_map(bcd_digit : INTEGER) return
   STD_LOGIC_VECTOR is
begin
    case bcd_digit is
        when 0 => return "1000000";  -- 0 no display
            de 7 segmentos
        when 1 => return "1111001";  -- 1 no display
            de 7 segmentos
```

```
            when 2 => return "0100100";  -- 2 no display
                de 7 segmentos
            when 3 => return "0110000";  -- 3 no display
                de 7 segmentos
            when 4 => return "0011001";  -- 4 no display
                de 7 segmentos
            when 5 => return "0010010";  -- 5 no display
                de 7 segmentos
            when 6 => return "0000010";  -- 6 no display
                de 7 segmentos
            when 7 => return "1111000";  -- 7 no display
                de 7 segmentos
            when 8 => return "0000000";  -- 8 no display
                de 7 segmentos
            when 9 => return "0010000";  -- 9 no display
                de 7 segmentos
            when others => return "1111111";  -- Valor
               invalido, todos os segmentos acesos
        end case;
    end function;

begin
    -- Processo principal que gerencia a contagem do
       tempo, controle de pausa e reset
    process(clk, reset)
    begin
        if reset = '1' then
            -- Se o reset for acionado, configura o
               tempo inicial conforme a selecao dos
               switches
            if (time_select_2 = '0' and time_select_1 =
               '0') then
                time_left <= 30;  -- Tempo inicial de 30
                    segundos
            elsif (time_select_2 = '0' and time_select_1
                = '1') then
                time_left <= 60;  -- Tempo inicial de 60
                    segundos
            elsif (time_select_2 = '1' and time_select_1
                = '0') then
                time_left <= 90;  -- Tempo inicial de 90
                    segundos
            else
```

```
            time_left <= 120; -- Tempo inicial de
                120 segundos
        end if;
        paused <= '0';  -- Reseta o controle de
            pausa
        clk_div_count <= 0;  -- Reseta o contador de
            divisao do clock
    elsif rising_edge(clk) then
        -- Processo executado a cada borda de subida
            do clock
        if pause = '1' then
            paused <= not paused;  -- Alterna o
                estado de pausa
        elsif clk_div_count = 49999999 then
            -- A cada 50 milhoes de ciclos de clock,
                atualiza a contagem
            clk_div_count <= 0;
            if paused = '0' and time_left > 0 then
                time_left <= time_left - 1;  --
                    Decrementa o tempo restante
            end if;
        else
            clk_div_count <= clk_div_count + 1;  --
                Incrementa o contador de divisao de
                clock
        end if;
    end if;
end process;

-- Atualiza o LED: acende quando o tempo chega a
    zero, caso contrario apaga
led <= '1' when time_left = 0 else '0';

-- Converte o valor do tempo restante para o formato
    BCD
bcd_values <= to_bcd(time_left);

-- Mapeia os valores BCD para os displays de 7
    segmentos
ssd_unit     <= ssd_map(bcd_values(0)); -- Unidades
ssd_tens     <= ssd_map(bcd_values(1)); -- Dezenas
ssd_hundreds <= ssd_map(bcd_values(2)); -- Centenas
ssd_thousands<= ssd_map(bcd_values(3)); -- Milhares
```

```
end projeto7;
```

3 Diagrama RTL

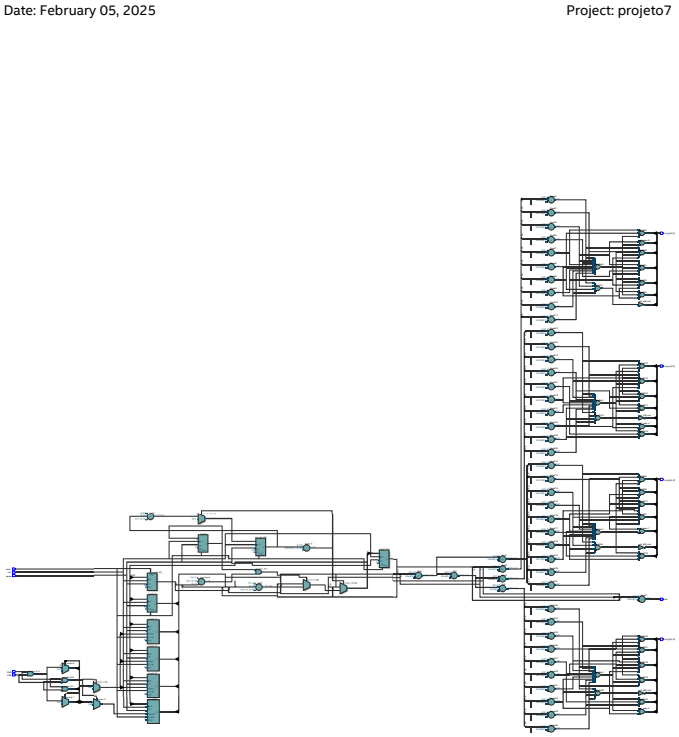

Figura 1: RTL Viewer