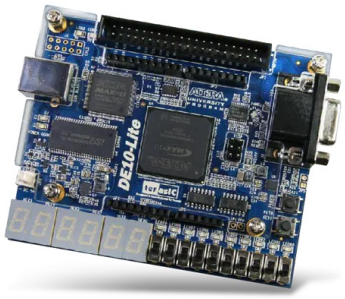
ISEL

Departamento de Engenharia Eletrónica e Telecomunicações e de Computadores

Licenciatura em Engenharia Informática e de Computadores



RELATÓRIO DO 1^0 TRABALHO

Circuitos Combinatórios

Trabalho realizado por:

(52718) Alexandre Silva

(52599) Duarte Rodrigues

Turma: 12D

Docente: Pedro Miguens, Sérgio André e Diogo Bastos

Lógica e Sistemas Digitais

2024 / 2025 Inverno

Conteúdo

1 Objetivo

Este trabalho tem como objetivo descrever e simular um circuito combinatório em VHDL e fazer a implementação deste mesmo circuito na placa de desenvolvimento *DE10-lite* da *Intel* de forma a extrair como outputs a função de paridade ímpar e a função de zero.

2 Descrição do Circuito a Projetar

Pretende-se implementar um circuito que verifique se o número de entradas ativas é ímpar (Paridade ímpar) e se todas as entradas estão inativas (Zero).

O diagrama do circuito (PZ10) faz uso de 10 inputs, correspondentes a 10 botões da placa *DE10-Lite* e 2 saídas, 2 LEDs para indicação dos sinais de Paridade e Zero. Este circuito é feitos a partir de blocos menores (PZ4) constituídos por 4 inputs e 2 outputs.

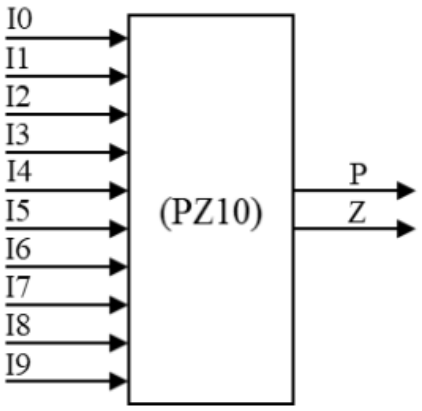


Figura 1: Diagrama PZ10.

Por este motivo, vamos fazer uso de 3 blocos de PZ4, aos quais designamos PZ4_1, PZ4_2 e PZ4_3, sendo o PZ4_1 e o PZ4_2, 2 das entradas de PZ4_3.

3 Desenvolvimento do Projeto

3.1 Funções lógicas dos circuitos

Na equação (1) podemos obter os mintermos de P (PZ4).

$$f(A^+, B, C, D^-) = \sum m(1, 2, 4, 7, 8, 11, 13, 14) \tag{1}$$

Na equação (2) podemos obter os mintermos de Z (PZ4).

$$f(A^+, B, C, D^-) = m(0) = \overline{ABCD} \tag{2}$$

3.2 Simplificação lógica

$f(A^+, B, C, D^-) =$

A \ C \ B \ D				
	$^{0}0$	$^{1}1$	$^{5}0$	$^{4}1$
	$^{2}1$	$^{3}0$	$^{7}1$	$^{6}0$
	$^{10}0$	$^{11}1$	$^{15}0$	$^{14}1$
	$^{8}1$	$^{9}0$	$^{13}1$	$^{12}0$

Não existe simplificação da função P, pelo que se descreveu a função em VHDL através dos mintermos associados.

A função Z é feita através de uma interseção entre os mintermos 0 de cada PZ4.

3.3 Diagrama de blocos

Na Figura 2 encontra-se o esquema do PZ10.

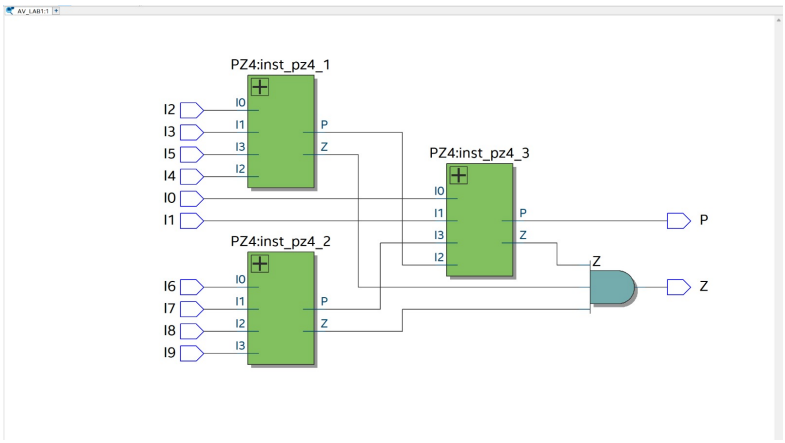


Figura 2: Diagrama de blocos do PZ10.

Na Figura 3 encontra-se o esquema de um dos PZ4 utilizados.

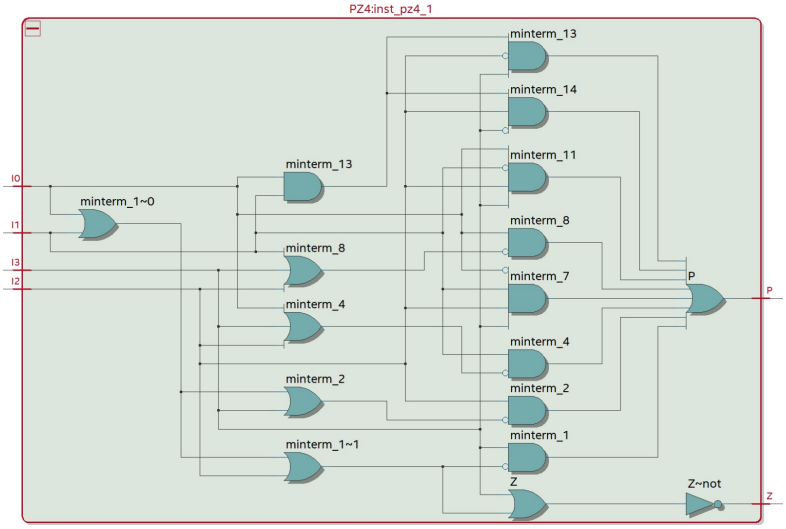


Figura 3: Diagrama de blocos do PZ4.

4 Resultados e Discussão

Quando o número total de inputs ativos (I1, I2, I3, I4, I5, I6, I7, I8, I9) é ímpar, o output P, apresentado pelo LED do pino A8 aparece ativo, e quando o número total de inputs ativos é par, o LED aparece inativo.

Quando o número de inputs ativos é zero, o output Z, apresentado pelo LED do pino A9 aparece ativo, permanecendo inativo em qualquer outra situação.

Estes resultados estão de acordo com a ideia que deu origem ao circuito e são os mesmos esperados após a utilização do testbench fornecido pelo professor, garantindo assim, que o circuito construído e implementado segue corretamente o funcionamento esperado.

No *Testbench* fizemos o teste para as situações com as entradas I0, I2, I4, I8 e I9 ativas, com as entradas I0, I2, I6, I8 ativas, com todas as entradas ativas e com todas as entradas inativas.

4.1 Simulação

Realizou-se a simulação do circuito projetado conforme o *Testbench* A.3.

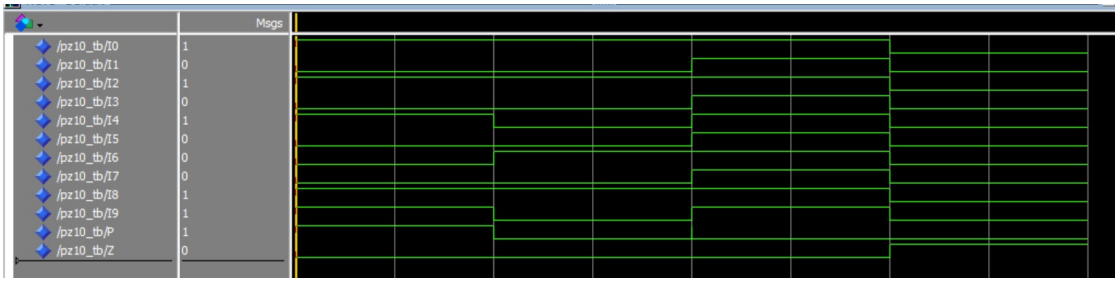


Figura 4: Waves da *Testbench* para os casos anteriormente referidos.

4.2 Teste

Teste (1) Placa *DE10-Lite*

- Inputs (I0, I2, I4, I8 e I9) ativos;
- Inputs (I1, I3, I5, I6, I7) inativos;
- A placa: *DE10-Lite* apresenta o LED A8 (output P) ativo;
- A placa: *DE10-Lite* apresenta o LED A9 (output Z) inativo.

Teste (2) Placa *DE10-Lite*

- Inputs: (I0, I2, I6, I8) ativos;
- Inputs: (I1, I3, I4, I5, I7, I9) inativos;
- A placa *DE10-Lite* apresenta o LED A8 (output P) inativo;
- A placa *DE10-Lite* apresenta o LED A9 (output Z) inativo.

Teste (3) Placa *DE10-Lite*

- Inputs: (I0, I1, I2, I3, I4, I5, I6, I7, I8, I9) ativos;
- Inputs: Nenhum input inativo;
- A placa *DE10-Lite* apresenta o LED A8 (output P) inativo;
- A placa *DE10-Lite* apresenta o LED A9 (output Z) inativo.

Teste (4) Placa *DE10-Lite*

- Inputs: Nenhum input ativo;
- Inputs: (I0, I1, I2, I3, I4, I5, I6, I7, I8, I9) inativos;
- A placa *DE10-Lite* apresenta o LED A8 (output P) inativo;
- A placa *DE10-Lite* apresenta o LED A9 (output Z) ativo.

5 Conclusão

O trabalho consistiu em descrever um circuito combinatório que verifica a paridade ímpar e que deteta o zero.

Este projeto foi desenvolvido no decorrer da disciplina de Lógica e Sistemas Digitais com o acesso à aplicação *Quartus Prime 20.1 Lite Edition*.

Para tal, foi usado como recurso o mapa de Karnaugh, apresentado no ponto 3.2 do relatório, para determinar a função lógica que melhor descrevia a solução ao trabalho proposto.

Após a utilização do mapa de Karnaugh, os membros do grupo constataram que não havia simplificação para a expressão lógica da função de paridade ímpar, pelo que implementaram-na através do somatório dos mintermos dessa dada função. A função Z foi implementada através de um AND entre os outputs Z de cada PZ4 implementado no circuito, como é demonstrado no diagrama de blocos do PZ10.

Este circuito foi testado na placa *DE10-Lite* da *Intel* durante a aula laboratorial da disciplina de forma a ter o circuito validado.

Os resultados experimentais confirmam o bom funcionamento do circuito, de acordo com os resultados da simulação feita através da *Testbench*, entregue pelo professor encarregado da disciplina nas aulas laboratoriais.

Referências

[1] Pedro Miguens Matutino, Sérgio André *Guia de Instalação do Quartus Prime Lite Edition 20.1.1.720*, 2022, ISEL.

[2] Terasic *DE10-lite User Manual*, 2020, Terasic.

A VHDL

A.1 AV_LAB1.vhd

```
library ieee;
use ieee.std_logic_1164.all;


-- Declaration of PZ10 with inputs between I0 and I9 and outputs P and Z
entity AV_LAB1 is
        port (
                I0 : in std_logic;
                I1 : in std_logic;
                I2 : in std_logic;
                I3 : in std_logic;
                I4 : in std_logic;
                I5 : in std_logic;
                I6 : in std_logic;
                I7 : in std_logic;
                I8 : in std_logic;
                I9 : in std_logic;

                P : out std_logic;
                Z : out std_logic
        );
end AV_LAB1;

-- Declaration of the components of PZ10 -> PZ4
architecture behavioral of AV_LAB1 is


        -- Import of PZ4 circuit with inputs between I0 and I3 and
        ↪  outputs P and Z
        component PZ4
                port(
```

```
                I0 : in std_logic;
                I1 : in std_logic;
                I2 : in std_logic;
                I3 : in std_logic;

                P : out std_logic;
                Z : out std_logic
        );

    end component;

    -- Declaration of the ouputs signals of the PZ4 used on the PZ10
    -- Outputs of PZ4_1, PZ4_2 and PZ4_3
    signal out_pz4_1_p : std_logic;
    signal out_pz4_1_z : std_logic;
    signal out_pz4_2_p : std_logic;
    signal out_pz4_2_z : std_logic;
    signal out_pz4_3_p : std_logic;
    signal out_pz4_3_z : std_logic;

begin

    -- Instantiation of the PZ4_1 circuit
    inst_pz4_1 : PZ4
        port map (
                I0 => I2,
                I1 => I3,
                I2 => I4,
                I3 => I5,

                P => out_pz4_1_p,
                Z => out_pz4_1_z
        );
```

```
        -- Instantiation of the PZ4_2 circuit
        inst_pz4_2 : PZ4
                port map (
                        I0 => I6,
                        I1 => I7,
                        I2 => I8,
                        I3 => I9,

                        P => out_pz4_2_p,
                        Z => out_pz4_2_z
                );

        -- Instantiation of the PZ4_3        circuit
        inst_pz4_3 : PZ4
                port map (
                        I0 => I0,
                        I1 => I1,
                        I2 => out_pz4_1_p,
                        I3 => out_pz4_2_p,

                        P => out_pz4_3_p,
                        Z => out_pz4_3_z
                );

        -- Outputs of PZ10
        -- P is on when the number of inputs is odd
        -- Z is on when all the inputs are off
        P <= (out_pz4_3_p);
        Z <= (out_pz4_1_z and out_pz4_2_z and out_pz4_3_z);


end behavioral;
```

A.2 PZ4.vhd

```
library ieee;
use ieee.std_logic_1164.all;

-- Declaration of PZ4 with inputs between I0 and I3 and outputs P and Z
entity PZ4 is
        port
        (
                I0 : in std_logic;
                I1 : in std_logic;
                I2 : in std_logic;
                I3 : in std_logic;

                P : out std_logic;
                Z : out std_logic
        );
        end pz4;



architecture structual of pz4 is

        -- Declaration of the minterm signals where the output P is on
        signal minterm_1 : std_logic;
        signal minterm_2 : std_logic;
        signal minterm_4 : std_logic;
        signal minterm_7 : std_logic;
        signal minterm_8 : std_logic;
        signal minterm_11 : std_logic;
        signal minterm_13 : std_logic;
        signal minterm_14 : std_logic;


begin
```

```
    -- Odd input minterms
    minterm_1  <= (not (I0 or I1 or I2) and I3);
    minterm_2  <= (not (I0 or I1 or I3) and I2);
    minterm_4  <= (not (I0 or I2 or I3) and I1);
    minterm_7  <= (not I0 and (I1 and I2 and I3));
    minterm_8  <= (not (I1 or I2 or I3) and I0);
    minterm_11 <= (not I1 and (I0 and I2 and I3));
    minterm_13 <= (not I2 and (I0 and I1 and I3));
    minterm_14 <= (not I3 and (I0 and I1 and I2));

    -- P is on when either of the odd minterms is on
    -- Z is on when none of the inputs are on
    P <= (minterm_1 or minterm_2 or minterm_4 or minterm_7 or
    ↪  minterm_8 or minterm_11 or minterm_13 or minterm_14);
    Z <= (not (I0 or I1 or I2 or I3));

end structual;
```

A.3 TLAB1_tb.vhd

```
LIBRARY IEEE;
use IEEE.std_logic_1164.all;

entity PZ10_tb is
end PZ10_tb;

architecture teste of PZ10_tb is
component AV_LAB1
  port(I0, I1, I2, I3, I4, I5, I6, I7, I8, I9 : in STD_LOGIC;
           P, Z : out STD_LOGIC
  );
end component;

signal I0, I1, I2, I3, I4, I5, I6, I7, I8, I9 : STD_LOGIC;
signal P, Z : STD_LOGIC;

begin

U0 : AV_LAB1 port map (I0 => I0, I1 => I1, I2 => I2, I3 => I3, I4 => I4,
↪  I5 => I5, I6 => I6, I7 => I7, I8 => I8, I9 => I9,
                    P => P, Z => Z);

process
begin
 I0 <= '1';
 I1 <= '0';
 I2 <= '1';
 I3 <= '0';
 I4 <= '1';
 I5 <= '0';
 I6 <= '0';
 I7 <= '0';
 I8 <= '1';
```

```
I9 <= '1';
wait for 10 ns;

I0 <= '1';
I1 <= '0';
I2 <= '1';
I3 <= '0';
I4 <= '0';
I5 <= '0';
I6 <= '1';
I7 <= '0';
I8 <= '1';
I9 <= '0';
wait for 10 ns;

I0 <= '1';
I1 <= '1';
I2 <= '1';
I3 <= '1';
I4 <= '1';
I5 <= '1';
I6 <= '1';
I7 <= '1';
I8 <= '1';
I9 <= '1';
wait for 10 ns;

I0 <= '0';
I1 <= '0';
I2 <= '0';
I3 <= '0';
I4 <= '0';
I5 <= '0';
I6 <= '0';
I7 <= '0';
```

```
 I8 <= '0';
 I9 <= '0';
 wait for 10 ns;


wait;

end process;

end teste;
```

B Atribuição de Pinos

Entradas	**pino**
I0	C10
I1	C11
I2	D12
I3	C12
Saídas	**pino**
P	A8
Z	A9

Tabela 1: Atribuição de pinos para PZ4 [2].

Entradas	**pino**
I0	C10
I1	C11
I2	D12
I3	D12
I4	A12
I5	B12
I6	A13
I7	A14
I8	B14
I9	F15
Saídas	**pino**
P	A8
Z	A9

Tabela 2: Atribuição de pinos para PZ10 [2].