Universidade Tecnológica Federal do Paraná
Campus Apucarana
Engenharia de Computação

Luis Henrique Ferracciu Pagotto Mendes, RA: 2272016

Relatório 4

Multiplexador Genérico

LRCO7A - Lógica Reconfigurável
Professor: Marcelo de Oliveira

Apucarana - PR
2024

1 Introdução:

Para este projeto, foi desenvolvido um multiplexador genérico usando a linguagem VHDL. O objetivo era explorar como esse componente essencial é utilizado em sistemas digitais. O multiplexador é um circuito que, a partir de sinais de controle, escolhe uma das entradas e a direciona para a saída. Ele é fundamental em aplicações digitais, incluindo a seleção de dados em barramentos, comunicação e processamento de informações.

O projeto foi criado para ser configurável, utilizando a estrutura generic do VHDL, o que permitiu ajustar tanto o número de entradas quanto a quantidade de bits por entrada diretamente na placa. Essa abordagem torna o multiplexador extremamente adaptável a diversas aplicações sem a necessidade de modificações substanciais no projeto.

Especificamente para esta atividade, o multiplexador foi configurado para operar com 4 entradas de 2 bits cada, controladas por 2 bits de seleção. Essa configuração foi implementada no FPGA, onde as entradas e os sinais de controle foram representados por chaves e a saída foi exibida em LEDs. Esta configuração prática proporcionou uma demonstração clara e tangível do funcionamento do circuito.

2 Código VHDL comentado:

A imagem abaixo mostra o código para a implementação do Multiplexador Genérico, com comentários para permitir um melhor entendimento e facilitar a leitura do projeto e sua execução.

```vhdl
----------------PACOTES----------------
library IEEE; -- pacote padrão para trabalhar com lógica digital
use IEEE.STD_LOGIC_1164.ALL; -- pacote para sinais em std_logic
use IEEE.NUMERIC_STD.ALL; -- pacote para operações matemáticas com vetores binários
----------------ENTIDADES----------------
entity projeto4_placa is

  port(
    sw: in std_logic_vector(9 downto 0); -- vetor que armazena as combinações dos switches para entradas e seleção
    ledr: out std_logic_vector(1 downto 0)  -- vetor que controla os LEDs para exibir os dois bits da saída
  );

end entity;

----------------PROJETO----------------
architecture projeto4_placa of projeto4_placa is -- utiliza a arquitetura do projeto para definir a lógica de funcionamento entre os sinais

  -- Declaração de constantes para configurar o multiplexador
  constant M : integer := 2; -- número de bits por entrada
  constant S : integer := 2; -- número de bits para seleção
  constant N : integer := 4; -- número total de entradas (2^S)

  -- Declaração de sinais internos para processar as entradas e seleção
  signal entradas : std_logic_vector((N*M) - 1 downto 0); -- vetor para armazenar as entradas concatenadas
  signal selecao  : std_logic_vector(S-1 downto 0);       -- vetor para armazenar os bits de seleção
  signal saida    : std_logic_vector(M-1 downto 0);       -- vetor para armazenar a saída selecionada

begin
  -- Mapear as entradas a partir das chaves (switches) da placa
  -- Os primeiros 8 switches (sw[7:0]) são usados como entradas concatenadas
  entradas <= sw(7 downto 0);

  -- Os dois últimos switches (sw[9:8]) são usados para selecionar uma das 4 entradas
  selecao  <= sw(9 downto 8);

  -- Lógica do multiplexador
  process (entradas, selecao) -- processa as entradas e os bits de seleção
    variable auxiliar : integer; -- variável auxiliar para calcular o índice da entrada selecionada
  begin
    -- Converter os bits de seleção para um índice numérico
    auxiliar := to_integer(unsigned(selecao));
    -- A saída é definida pelos bits correspondentes à entrada selecionada
    saida <= entradas(auxiliar*M + M-1 downto auxiliar*M);
  end process;

  -- Mapear a saída do multiplexador para os LEDs da placa
  -- Os LEDs (ledr[1:0]) exibem os dois bits da saída selecionada
  ledr <= saida;

end architecture;
```

Figura 1: Código VHDL comentado.

3 Simulação:

Para compreensão mais profunda do funcionamento de um multiplexador, foi gerado um Waveform com entradas (nos switches) geradas aleatoriamente para poder demonstrar a saída dos sinais.

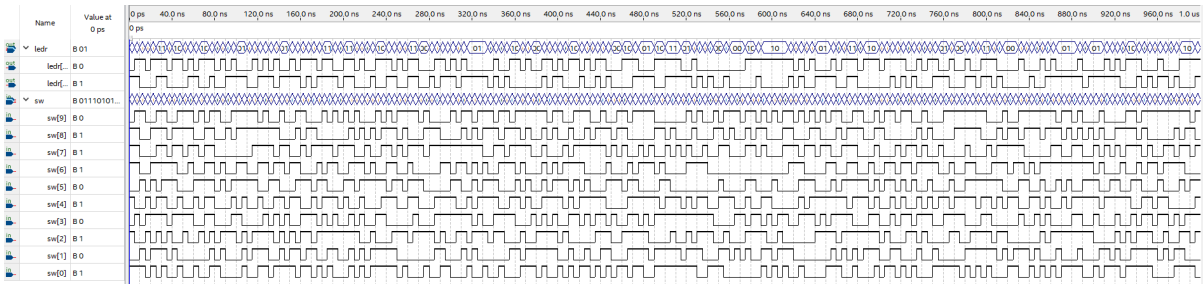

Figura 2: Simulação do projeto com sinais gerados.

4 Multiplexador Aplicado na Placa:

Após a implementação via software do projeto, foi possível enviar o código para a placa DE10-Lite, permitindo a interação usuário-placa, a fim de que o usuário tenha uma experiência prática e visual do funcionamento do multiplexador e a forma como ele aloca e trabalha com os sinais.

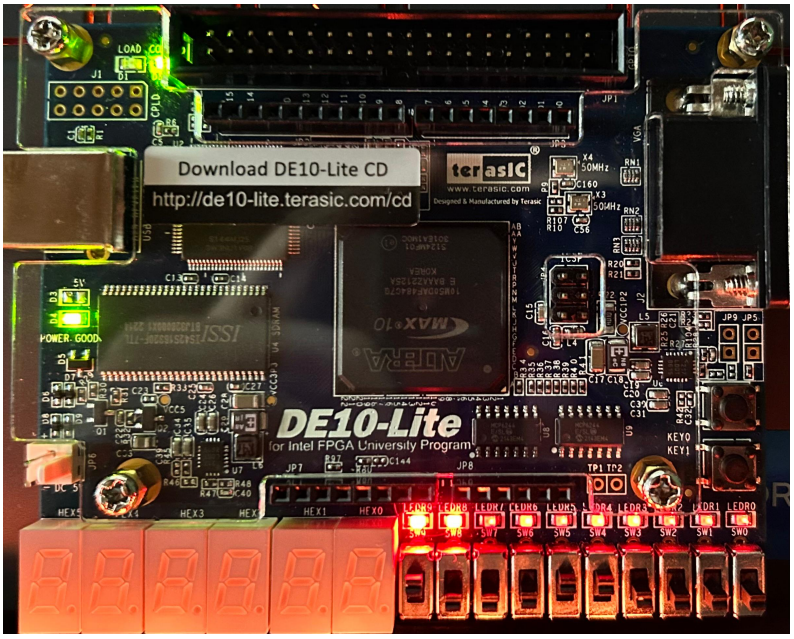

Figura 3: Foto da placa com o MUX em funcionamento.

5 Diagrama RTL:

Por fim, o diagrama RTL demonstra a construção lógica do código criado, e como ele trabalhará a partir das entradas.

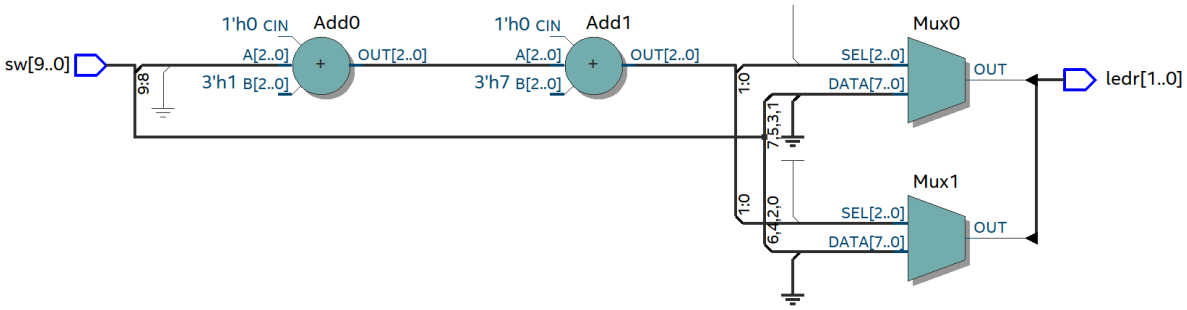

Figura 4: Diagrama RTL gerado pelo Quartus.