UTFPR
Engenharia de Computação
Lógica Reconfigurável

Atividade 4: Multiplexador genérico

Aluno: Deivid da Silva Galvão
Professor orientador: Marcelo de Oliveira

Outubro
2024

UTFPR
Engenharia de Computação
Lógica Reconfigurável

Relatório

Relatório do Trabalho Prático Disciplinar apresentado como requisito parcial à obtenção de nota na disciplina de Lógica Reconfigurável do Curso Superior de Engenharia de Computação da Universidade Tecnológica Federal do Paraná.

Aluno: Deivid da Silva Galvão

Professor orientador: Marcelo de Oliveira

Outubro
2024

Conteúdo

1 Introdução

Na atividade 4, focamos no desenvolvimento e implementação de um multiplexador genérico, permitindo a configuração tanto do número de entradas quanto da quantidade de bits por entrada. A ênfase foi na flexibilidade e adaptabilidade do design proposto. Utilizando uma placa FPGA, concretizamos uma versão específica do multiplexador com 4 entradas de 2 bits cada. As entradas foram associadas às chaves da FPGA, enquanto as saídas foram mapeadas para LEDs ou SSDs.

2 Implementação

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

entity prog4 is
  generic (
        S : integer := 2; -- Numero de bits de selcao
        M : integer := 8  -- Numero de bits por entrada
    );
    port (
        entradas : in  std_logic_vector((2**S)*M-1 downto 0); -- Entradas
        selecao  : in  std_logic_vector(S-1 downto 0); -- Bits de selcao
        saida    : out std_logic_vector(M-1 downto 0)  -- Saida
    );
end prog4;

architecture prog4 of prog4 is
  type input_array is array (0 to (2**S)-1)...
  of std_logic_vector(M-1 downto 0);
  signal inputs : input_array;
begin
  -- Mapeamento das entradas para o vetor de sinais
  gen_map: for i in 0 to (2**S)-1 generate
    inputs(i) <= entradas((i+1)*M-1 downto i*M);
  end generate gen_map;

  -- Selecao da entrada correta com base no seletor
  process (selecao, inputs)
  begin
    saida <= inputs(to_integer(unsigned(selecao)));
  end process;
end prog4;
```

O código VHDL apresentado realiza a implementação de um multiplexador que utiliza a diretiva generate para organizar e selecionar entradas de forma eficiente. Primeiramente, as bibliotecas necessárias são importadas, permitindo a manipulação de sinais lógicos e operações aritméticas. Na definição da entidade prog4, parâmetros genéricos são utilizados: S define o número de bits de seleção e M o número de bits por entrada. A entidade possui três portas: entradas, um vetor de entradas, seleção, um vetor de bits de seleção, e saida, que representa a saída selecionada. A arquitetura é definida com um tipo de dado "imput array", que representa um vetor de vetores de bits, e um sinal inputs deste tipo, para armazenar as entradas descomprimidas. Utiliza-se um bloco generate para mapear as entradas para o vetor inputs, dividindo o vetor de entradas em partes individuais e mapeando cada uma para um índice do vetor. Por fim, um processo é utilizado para selecionar a entrada correta com base no valor de seleção, atualizando a saída para refletir a entrada correspondente ao valor do seletor. Em resumo, o código implementa um multiplexador onde S bits de seleção determinam qual das 2 elevado a S entradas de "M"bits será passada para a saída, utilizando a diretiva generate para mapear eficientemente as entradas e selecionando a entrada correta com base no valor do seletor.

3 Resultados obtidos

Após a criação do código, foi realizado uma simulação para testar e validar o funcionamento antes de inseri-lo na placa. Abaixo, pode-se observar que o código operou conforme o esperado para o caso de 16 entradas e 8 bits, retornando os valores corretos para cada número selecionado.

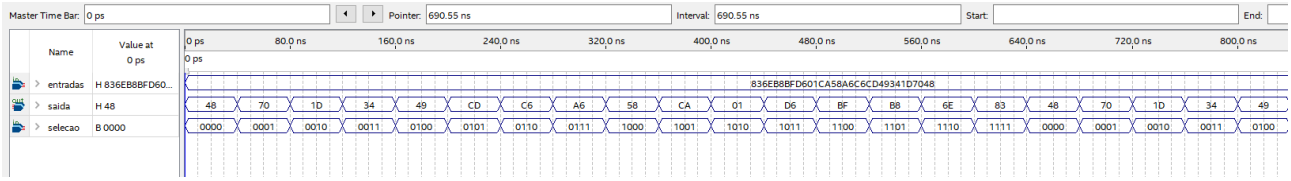

Figura 1: Simulação com 16 entradas e 8 bits

Em seguida, o código foi inserido na placa, utilizando quatro entradas de 2 bits cada e um seletor de 2 bits para se adequar ao número necessário de entradas. Cada entrada foi conectada a um switch da placa, assim como o seletor. Para verificar se a saída correspondia à entrada, utilizamos dois LEDs, conforme mostrado nas imagens abaixo. Com a implementação e o Pin Planner prontos, o código foi iniciado na placa, e os resultados foram consistentes com as entradas.

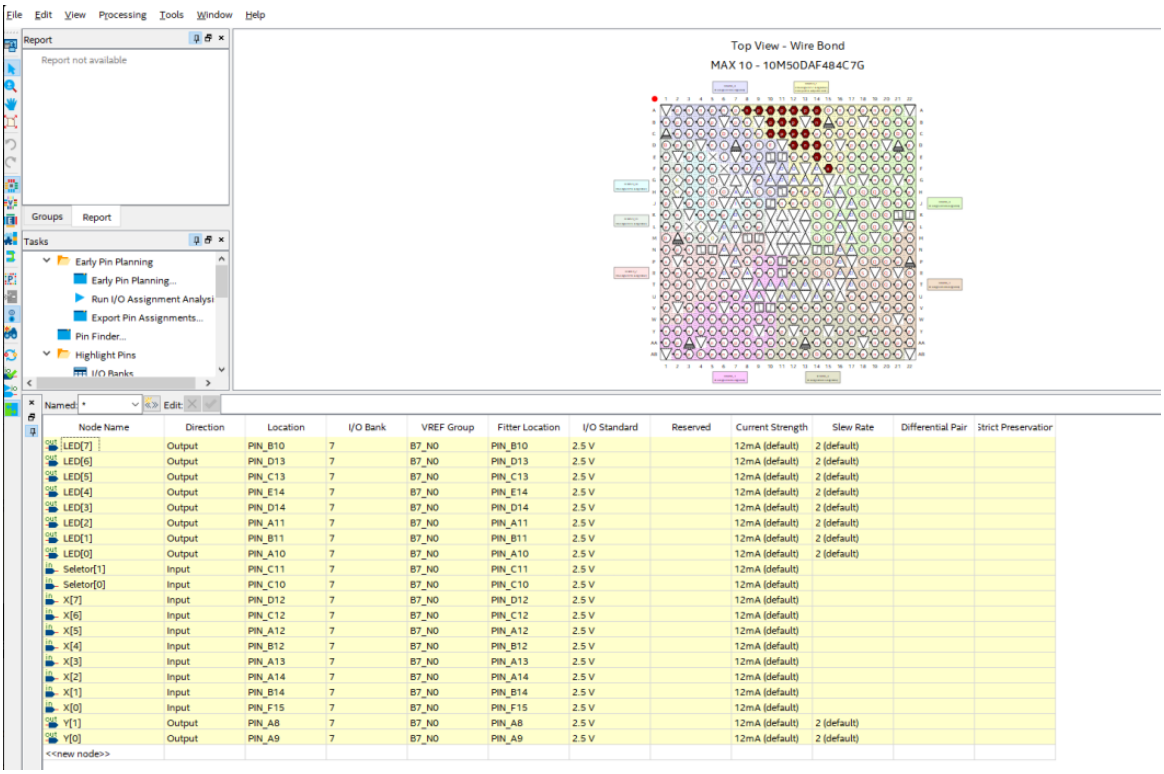

Figura 2: Pin Planner

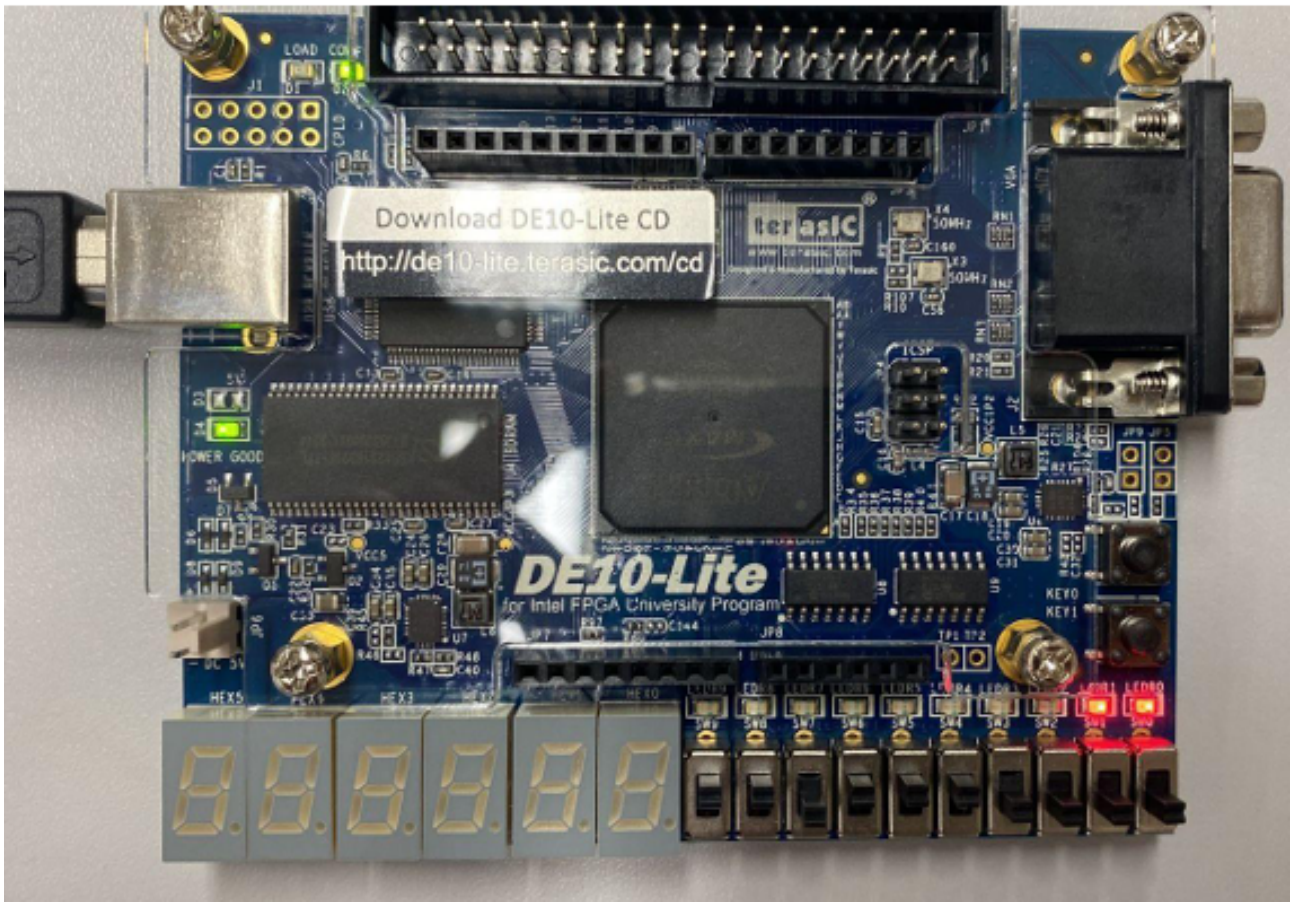

Figura 3: Foto da placa com seletor zero

Pode-se notar que o seletor está ajustado para zero, o que o transforma em um valor inteiro.

Nossa faixa de seleção é a primeira, e como podemos ver, os dois switches estão configurados em 1 e 1, o que corresponde à saída nos LEDs.

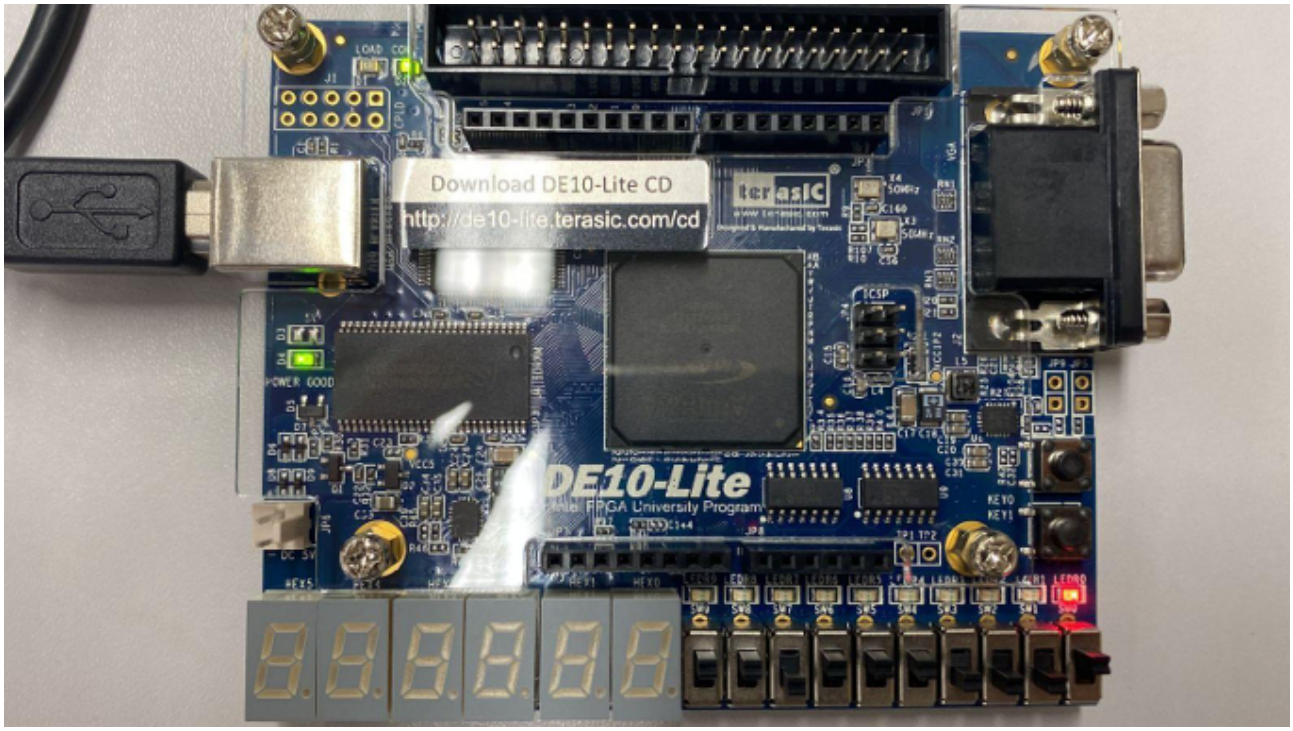

Figura 4: Foto da placa com seletor 1

Ajustando o seletor para um, também foi obtido um resultado satisfatório.

A baixo segue a imagem do diagrama rtl gerado pelo código VHDL.

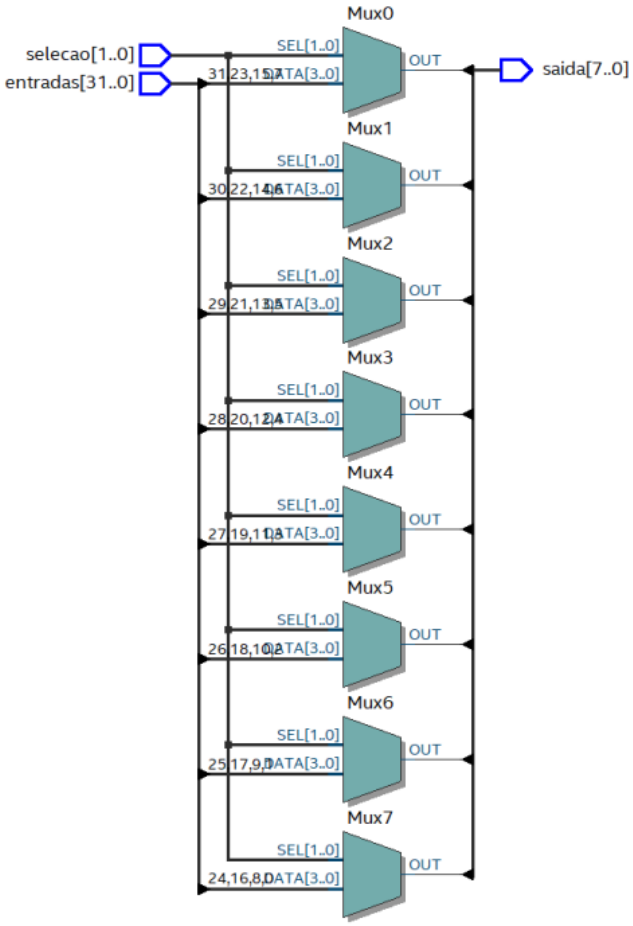

Figura 5: Diagrama RTL

Por fim, conclui-se que a implementação bem-sucedida na placa FPGA demonstrou a viabilidade prática do projeto. Utilizando a interface com componentes da placa, como chaves e LEDs ou SSDs, pudemos observar de forma concreta o desempenho do multiplexador, confirmando que ele opera conforme especificado.