

Laboratório 3 – Multiplexadores

Objetivos:

1. Experimentar a descrição em VHDL de circuitos na forma comportamental;
2. Reforçar os conceitos de multiplexadores e demultiplexadores.
3. Pôr em prática conceitos aprendidos na disciplina Circuitos Digitais - Teoria.

Introdução:

Na aula de hoje iremos rever dois conceitos importantes de circuitos digitais: multiplexadores e demultiplexadores. Também serão apresentadas duas formas de implementações deste tipo de circuito combinacional.

Multiplexadores:

Multiplexadores (ou MUX) são um bloco construtivo de nível mais elevado usado em circuitos digitais. Um multiplexador $M \times 1$ tem M entradas e 1 saída, permitindo que apenas uma das entradas seja passada para a saída, através de bits de seleção. Pode-se chamar um multiplexador de **seletor** pois seleciona uma das entradas para ser passada para a saída.

Um multiplexador 2×1 tem duas entradas de dados I_0 e I_1 , e um bit de seleção de entrada S_0 . Quando $S_0 = 0$, a entrada I_0 é replicada na saída, quando $S_0 = 1$, o valor que passará para a saída será o da entrada I_1 .

O multiplexador 2×1 pode ser construído usando portas lógicas NOT, AND e OR. Na Figura 1(c), é ilustrado o projeto de um multiplexador usando essas portas, apresentando também o fluxo de dados (Figura 1(b)) quando o bit de entrada se altera.

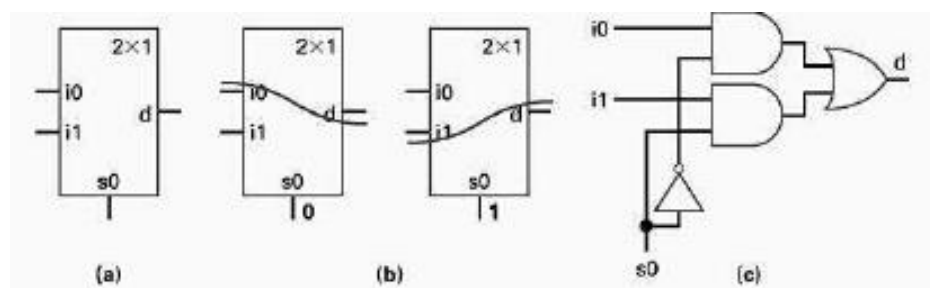


Figura 1: Estrutura de um Multiplexador

Analisando o circuito mostrado na Figura 1(c), pode-se obter a expressão:

$$D = I_0 \cdot S_0' + I_1 \cdot S_0 \quad (1)$$

A utilização de multiplexadores são as mais diversas, por exemplo como seletores de canais em televisões, seletores de canais de áudio, geradores de sinais. A representação gráfica de multiplexadores, está apresentada na Figura 2, bem como a tabela verdade.

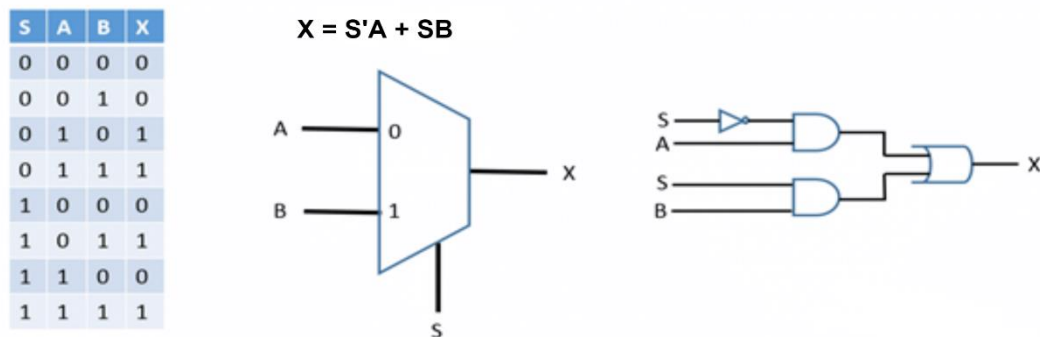


Figura 2: Representação dos multiplexadores e tabela verdade.

Normalmente um MUX pode ter até 2^N entradas e necessita para a seleção do sinal de saída, que N sinais de seleção estejam presentes. Portanto, para dois sinais, é necessário um sinal de seleção, para quatro sinais, é necessário dois sinais para seleção, e assim por diante.

Um multiplexador 4x1 tem quatro entradas de dados A_0 , A_1 , A_2 e A_3 , duas entradas de seleção S_1 e S_0 e uma saída de dados X (um multiplexador sempre terá apenas uma saída de dados, não importando quantas entradas). Um diagrama de blocos de um multiplexador 4x1 está mostrado na Figura 3.

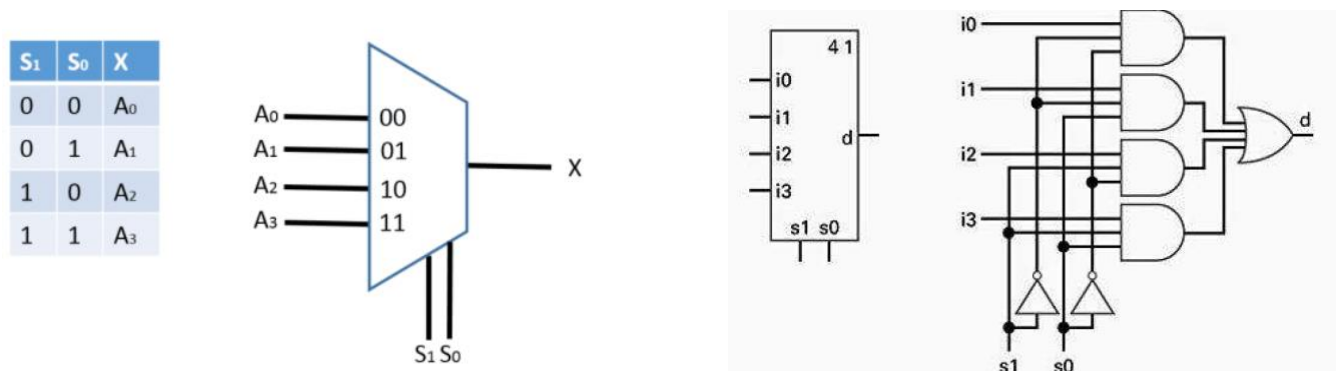


Figura 3: Diagrama de blocos de um multiplexador 4x1.

Multiplexadores mais complexos podem ser construídos utilizando multiplexadores 2 x 1, conforme visto na Figura 4.

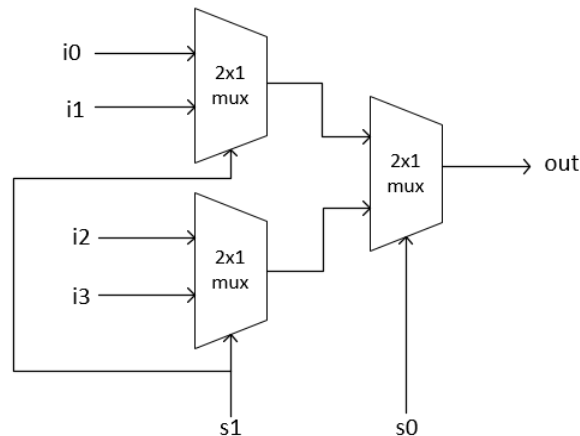


Figura 4: Multiplexador 4x1 utilizando multiplexadores 2x1

Demultiplexadores:

Os demultiplexadores são blocos construtivos combinacionais que possuem o comportamento oposto ao de um multiplexador, sendo os demultiplexadores bem menos utilizados. De uma forma geral, um demultiplexador $1 \times M$ tem uma entrada de dados e , com base nos bits de seleção, passa essa entrada para uma das M saídas, enquanto as outras permanecem em 0. Na Figura 4 é apresentado um diagrama funcional de um demultiplexador. As setas mais largas nas entradas e nas saídas podem determinar um vetor de dados, enquanto a entrada de seleção determina para qual saída o dado será transmitido.

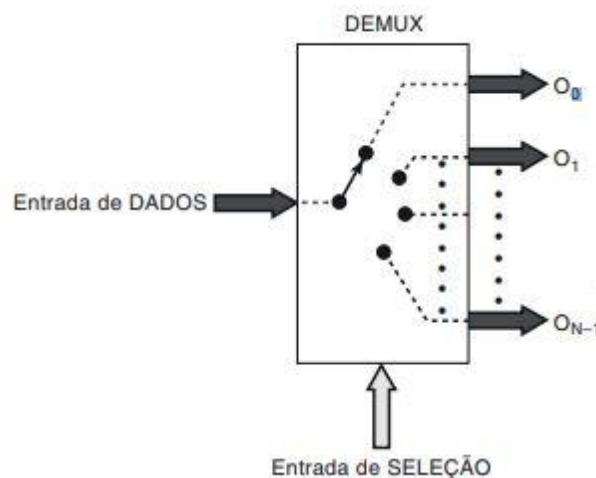


Figura 5: Estrutura de um Demultiplexador

Descrição Comportamental do VHDL:

A linguagem de descrição de hardware VHDL permite a utilização de comandos condicionais para construir uma declaração de arquitetura baseada no comportamento do módulo. Nesse tipo de descrição, não é necessário, obrigatoriamente, preocupar-se com portas lógicas. Pode-se implementar o comportamento dos circuitos lógicos por meio de algoritmos. No quadro 1, a funcionalidade da porta NOT é implementada por meio de sua descrição comportamental.

```
ENTITY PortaNot IS
PORT(A : IN BIT;
      S : OUT BIT);
END;

ARCHITECTURE behav OF PortaNot IS
BEGIN
WITH A SELECT
    S <= '1' WHEN '0',
        '0' WHEN '1';
END;
```

Quadro 1: Descrição comportamental da Porta NOT.

O comando **WITH** <variavel> **SELECT** funciona como um SWITCH...CASE da linguagem C++.

É possível implementar um MUX por meio da descrição comportamental do componente, conforme a descrição do quadro 2.

```
ENTITY Mux2x1 IS
PORT(I0,I1,s0 : IN BIT; -- No qual s0 é a porta de seleção
      d : OUT BIT);
END;
ARCHITECTURE behav OF Mux2x1 IS
BEGIN
WITH s0 SELECT
    d <= I0 WHEN '0',
        I1 WHEN '1';
END;
```

Quadro 2: Descrição comportamental do MUX 2x1

Hora de Praticar:

Considerando entradas de dados com largura de 1 bit:

1. Implemente em VHDL, utilizando circuitos lógicos, um MUX 2x1;
2. Implemente em VHDL, utilizando a descrição comportamental, um MUX 2x1;
3. Implemente em VHDL, utilizando a descrição comportamental, um MUX 4x1;
4. Implemente em VHDL, utilizando a descrição comportamental + componentes, um MUX 4x1 utilizando MUX 2x1.
5. Entregue um relatório contendo/descrevendo a execução dos itens 1 a 4.

Dica:

A descrição de sinais de seleção, para multiplexadores com mais de duas entradas, pode ser auxiliada pelo(s):

- Uso do comando “`x <= a & b`”, o qual concatena em x os valores de a e b.
- Uso de vetores de bits, declarados com variáveis do tipo “`BIT_VECTOR`”.