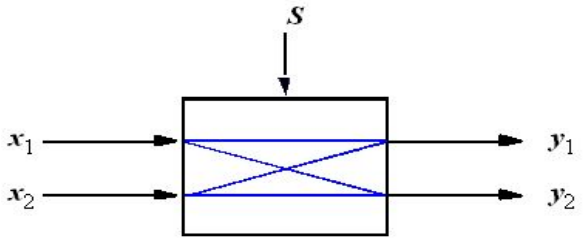
Laboratório 03

Instruções:

- Quando for demonstrar seu trabalho, tome nota do número da placa utilizada. O número da placa será utilizado para atribuir a nota ao grupo.
- A última página deste documento contém um checklist com todos os arquivos que fazem parte da entrega.
- Os nomes dos arquivos devem ser seguidos, e isso faz parte da avaliação.
- A entrega deverá estar em único arquivo .ZIP, com o nome **T_Lab03_RA.zip**, **T** é a turma, e **RA** é o RA do componente do grupo que fará a entrega. Por exemplo, B_Lab03_123456.zip é a entrega do grupo do aluno com o RA 123456, na turma B.
- Não divida ou agrupe em pastas os arquivos dentro do .ZIP.
- A entrega deve ser feita pelo Google Forms (https://forms.gle/qBnoCDXBQec8tpvE6). Você deve estar autenticado com uma conta do Google - pode ser uma conta pessoal ou da DAC.
- Apenas um integrante do grupo precisa fazer a entrega.
- Preste especial atenção aos nomes das entidades e sinais (entradas e saídas) descritos nos laboratórios. Isso também faz parte da avaliação.
- Se mais do que um arquivo for recebido para a mesma entrega, o último recebido será considerado. Utilize o mesmo RA do aluno entregando.
- Faça o download do arquivo **lab03_material_v2020.2.zip**. Esse arquivo já contém as descrições de *entity* necessárias para implementar os circuitos. Utilize elas, e não as altere.

Parte I - Crossbar switch

Seja o componente *xbar* que implementa um *crossbar switch* (a inversão só ocorre se o S estiver no no nível lógico alto, ou seja, se S = '1'). Projete os circuitos abaixo em VHDL e verifique o funcionamento de todos os projetos com simulação.



I.1. Projete este circuito usando a construção WITH, SELECT e WHEN [**sem usar processo**]. Teste por simulação. Salve como **xbar_v1.vhd**.

ENTREGAR: Arquivo **xbar_v1.vhd** e screenshot da simulação em **xbar_v1.png**.

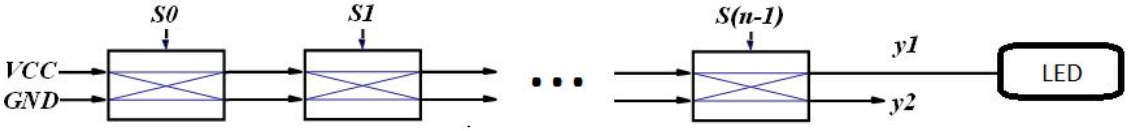
I.2. Projete este mesmo circuito usando a construção WHEN ELSE [**sem usar processo**]. Teste por simulação. Salve como **xbar_v2.vhd**.

ENTREGAR: Arquivo **xbar_v2.vhd** e screenshot da simulação em **xbar_v2.png**.

I.3. Projete este mesmo circuito em VHDL usando a construção PROCESS. Teste por simulação. Salve como **xbar_v3.vhd**.

ENTREGAR: Arquivo **xbar_v3.vhd** e screenshot da simulação em **xbar_v3.png**.

I.4. Instanciando alguma das versões do componente xbar acima, implemente o circuito abaixo, que implementa um número variável de estágios (utilize os comandos GENERIC e GENERATE). Salve como **xbar_gen.vhd**.



VCC = constante 1; GND = constante 0
ENTREGAR: Arquivo **xbar_gen.vhd**.

I.5. Crie um novo projeto instanciando o componente do item **I.4** com 5 estágios. Teste por simulação. Salve como **xbar_stage_5.vhd**. Programe a placa para verificar o funcionamento, usando 5 switches – **SW(0)** até **SW(4)** - e um LED, sinal **LEDR(0)**, como saída.

ENTREGAR: Arquivo **xbar_stage_5.vhd** e screenshot da simulação em **xbar_stage_5.png**.

I.6. Repita **I.5** para 8 estágios, ou seja, utilizando **SW(0)** até **SW(7)**. Salve como **xbar_stage_8.vhd**

ENTREGAR: Arquivo **xbar_stage_8.vhd** e screenshot da simulação em **xbar_stage_8.png**.

Parte II - Multiplexador

A figura abaixo mostra um circuito multiplexador 4 para 1 projetado utilizando-se um decodificador 2 para 4 e portas lógicas. $w_{0..3}$ são as entradas, $s_{0..1}$ os sinais de seleção de entrada, *En* sinal para ligar e desligar o circuito (significa que, quando desligado, todas as saídas do decodificador serão iguais à zero) e f a saída selecionada. Projete os circuitos abaixo em VHDL e verifique o funcionamento de todos os projetos com simulação.

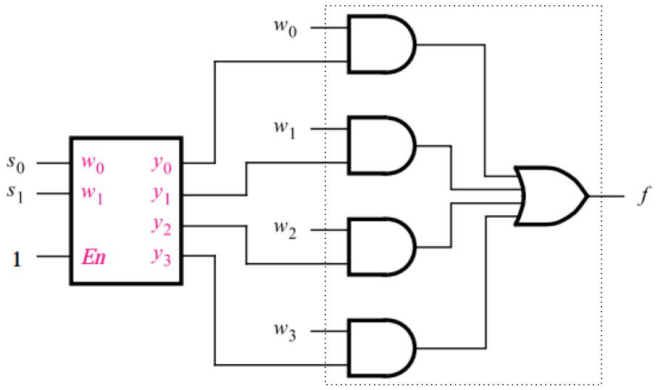


Figura 1: Mux 4-1 usando Dec2-4 e portas lógicas

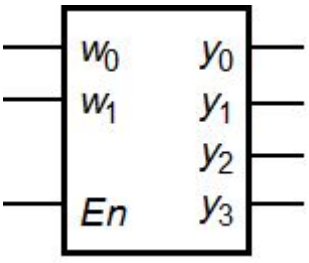


Figura 2: Dec2-4

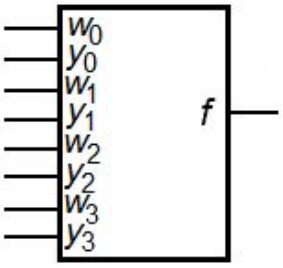


Figura 3: Lógica extra

II.1. Implemente o decodificador 2 para 4 da Figura 2 [**sem usar processo**].

ENTREGAR: Implementação em VHDL em **dec2_to_4.vhd** e screenshot da simulação em **dec2_to_4.png**.

II.2. Implemente o circuito dentro da caixa pontilhada na Figura 1, conforme o símbolo da Figura 3 [**sem usar processo**].

ENTREGAR: Implementação em VHDL em **extra_logic.vhd** e screenshot da simulação em **extra_logic.png**.

II.3. Instanciando os circuitos dos itens **II.1** e **II.2**, projete um multiplexador 4:1 como na Figura 1. Tabela de mapeamento do multiplexador:

Valor binário para a porta *sel*	**Saída esperada (porta selecionada)**
00	d0
01	d1
10	d2
11	d3

ENTREGAR: Implementação em VHDL em **mux4_to_1.vhd** e screenshot da simulação em **mux4_to_1.png**.

II.4. Instanciando o circuito do item **II.3**, implemente um multiplexador 16:1 em VHDL. Lembre-se que o quando o valor da entrada *sel* for igual a 0000, então a saída o bit menos significativo da porta *data*, ou seja *data*(0); e se sel for 1111, então a saída será o bit mais significativo, ou seja *data*(15). Não deve ser usado processo nem implementado de forma estrutural (utilizando portas lógicas).

ENTREGAR: Implementação em VHDL em **mux16_to_1.vhd** e screenshot da simulação em **mux16_to_1.png**.

- ENTREGA -

Entregue um único arquivo comprimido em formato **ZIP** de nome **T_Lab03_RA.zip**, onde **RA** é o RA do aluno entregando e **T** é a turma, contendo:

- Arquivos **xbar_v1.vhd** e **xbar_v1.png** do item **I.1**.
- Arquivos **xbar_v2.vhd** e **xbar_v2.png** do item **I.2**.
- Arquivos **xbar_v3.vhd** e **xbar_v3.png** do item **I.3**.
- Arquivo **xbar_gen.vhd** do item **I.4**.
- Arquivos **xbar_stage_5.vhd** e **xbar_stage_5.png** do item **I.5**.
- Arquivos **xbar_stage_8.vhd** e **xbar_stage_8.png** do item **I.6**.
- Arquivos **dec2_to_4.vhd** e **dec2_to_4.png** do item **II.1**.
- Arquivos **extra_logic.vhd** e **extra_logic.png** do item **II.2**.
- Arquivos **mux4_to_1.vhd** e **mux4_to_1.png** do item **II.3**.
- Arquivos **mux16_to_1.vhd** e **mux16_to_1.png** do item **II.4**.