

**CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS –  
CEFET/MG**

**DIOGO EMANUEL ANTUNES SANTOS  
GUILHERME AUGUSTO DE OLIVEIRA  
LUIZ EDUARDO LEROY SOUZA**

**Multiplexadores de 2 e 4 bits**  
Relatório 4

**BELO HORIZONTE  
2021**

## **Sumário**

<b>Objetivo</b>	<b>2</b>
<b>Desenvolvimento</b>	<b>2</b>
2.1 Tabela verdade	2
2.2 Expressões lógicas	3
2.3 Descrição em Verilog HDL	4
2.4 Circuito lógico	5
2.5 Simulação do circuito no ModelSim:	7
<b>Análise dos resultados:</b>	<b>8</b>

## 1. Objetivo

O presente relatório possui dois objetivos principais, tendo em vista que foi desenvolvido em duas partes. O primeiro objetivo é desenvolver um circuito digital em Verilog HDL de um multiplexador de dois canais. Após isso, foi preciso utilizar o módulo criado como parte de um multiplexador de quatro canais, implementado por meio da estrutura hierárquica.

## 2. Desenvolvimento

Nesta parte será apresentado o passo a passo utilizado para desenvolver o projeto.

### 2.1 Tabela verdade

A tabela verdade para ambas as partes foi disponibilizada previamente tanto na vídeo-aula disponibilizada no YouTube quanto no documento que detalha o projeto.

Para a primeira parte, bastava observar que, quando o bit *enable* estava em nível lógico alto, a saída seria representada pela entrada digital I1. Em caso contrário, pelo endereço de entrada I0.

I0	I1	sel	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Tabela 01: Tabela verdade da primeira parte

Para a segunda parte, bastava novamente observar a disposição dos bits de sinal A e B. A tabela completa não será apresentada, pois tratam-se de 6 entradas distintas. Resumidamente, caso o bit de sinal B esteja em nível lógico alto, o mesmo acionará ou a entrada digital I1 ou I3, tendo em vista a configuração mostrada na tabela anterior. Prosseguindo, caso o bit de sinal A esteja em nível alto, o mesmo selecionará a entrada I3 como resultado, tendo em vista que se trata do segundo valor passado pelos dois primeiros módulos. A mesma lógica ocorre de forma inversa caso o bit B seja 0.

A	B	S
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Tabela 02: Tabela verdade da segunda parte

## 2.2 Expressões lógicas

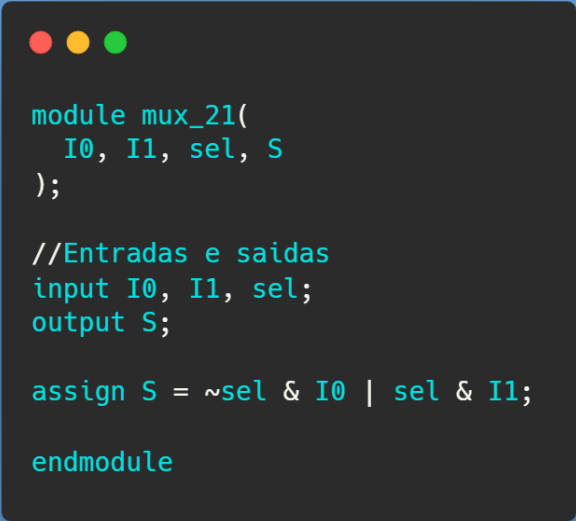
A expressão booleana do MUX de 2 canais foi fornecida na apresentação do projeto prático. Trata-se da seguinte expressão:

$$saida = \sim selI0 + selI1$$

Também seria possível obter essa expressão utilizando a tabela verdade anterior e aplicando o mapa de karnaugh para simplificar o resultado.

A segunda parte do trabalho, relativo ao MUX de 4 canais, não utilizou nenhuma expressão lógica, tendo em vista que se trata de uma implementação hierárquica que utiliza o módulo anterior como componente.

## 2.3 Descrição em Verilog HDL



```
module mux_21(  
    I0, I1, sel, S  
);  
  
    //Entradas e saidas  
    input I0, I1, sel;  
    output S;  
  
    assign S = ~sel & I0 | sel & I1;  
  
endmodule
```

Imagem 01: Representação em Verilog HDL do módulo 1

Acima é possível observar a implementação do circuito solicitado. O módulo do multiplexador de dois canais possui duas entradas digitais: *I0* e *I1*, somadas ao bit de sinal *sel*. Como saída possui o *Wire S*.

Trata-se basicamente da aplicação do método assign à expressão lógica já descoberta anteriormente.

```

module mux_41(
    I0, I1, I2, I3, A, B, S
);

//Entradas e saidas
input I0, I1, I2, I3, A, B;
output S;

//Comunicacao com o outro modulo (saida do circuito)
wire S0, S1, S2;

mux_21 saidaZero(I0, I1, B, S0);
mux_21 saidaUm(I2, I3, B, S1);
mux_21 saida(S0, S1, A, S2);

assign S = S2;

endmodule

```

Imagem 02: Descrição da segunda parte

A segunda parte do trabalho utiliza estrutura hierárquica, tendo em vista que faz uso do módulo mencionado anteriormente. Têm como entradas os sinais digitais *I0*, *I1*, *I2*, *I3* e os bits de sinal *A*, *B*. Como saída possui a variável *S*.

Além dos *inputs/outputs* anteriores, existem também 3 *Wires* *S0*, *S1* e *S2*, que servem como meio de comunicação entre o módulo de 2 bits e o módulo atual de 4 canais. Para realizar a operação, foi realizada a chamada do módulo de 2 canais duas vezes: uma para a saída zero e outra para a saída 1, pois a operação é realizada de 2 em 2 bits. Com os dois resultados anteriores, foi utilizado um outro multiplexador de 2 bits, atribuindo uma saída final *S2*.

## 2.4 Circuito lógico

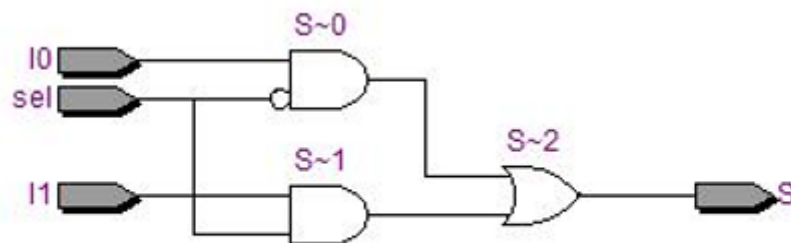


Imagem 03: Circuito da primeira parte

Descrição do circuito lógico:

No circuito acima temos três inputs, I0, I1, bits de entrada, e o bit de controle que chamamos de sel. Dessa forma, para a construção do circuito multiplexador de dois canais utilizamos duas portas lógicas AND e uma porta lógica OR, assim como um inversor NOT.

Com relação ao funcionamento do circuito, para a primeira porta AND S~0 temos as entradas I0 e o bit de controle sel, assim como na porta AND S~1 temos as entradas I1 e o bit de controle.

Para que o circuito funcione devidamente, utilizamos um inversor no bit de controle da porta S~0, para quando o mesmo for 0, o inversor torná-lo em 1 para que gere uma saída de nível lógico alto quando a entrada I0 também for 1 e, para a porta lógica S~1, o bit de controle é 0, desabilitando essa porta. Para a porta AND S~1, a situação é semelhante, assim o bit de controle assume nível lógico alto para porta S~1, fazendo com que gere uma saída de nível lógico alto quando a entrada I1 também possuir nível lógico alto e, além disso, desabilita a porta S~0 devido ao inversor presente na entrada do bit de controle. Assim é possível controlar se o sinal de nível lógico alto que chega à saída S vem da entrada I0 ou da entrada I1.

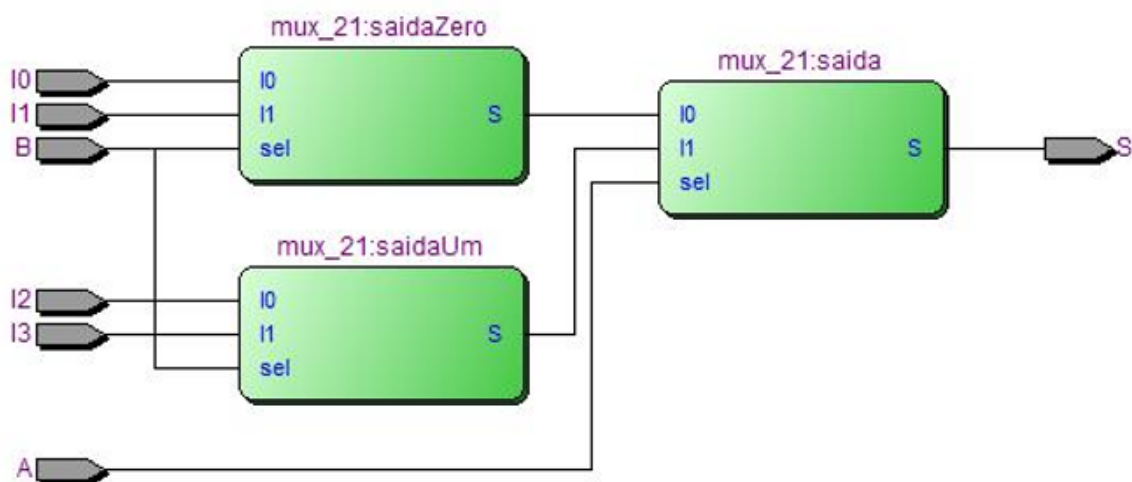


Imagem 04: Circuito da segunda parte

## Descrição do circuito lógico

Temos acima um circuito multiplexador de 4 canais no qual utilizamos para a construção 3 multiplexadores de 2 canais explicados anteriormente. O circuito acima consiste em 6 bits de entrada, sendo 4 para os canais de entrada do multiplexador e outros 2 bits, sendo que um é o bit de controle do primeiro par de multiplexadores mux2:1\_saidaZero e mux2:1\_saidaUm e o outro é bit de controle do terceiro multiplexador de 2 bits.

Com relação ao funcionamento, cada multiplexador de dois bits funciona conforme explicado anteriormente, o primeiro tem como entradas os bits I0, I1 e o bit de controle B , o segundo multiplexador tem como entradas os bits I2,I3 e também o bit de controle B, já o terceiro recebe a saída de cada multiplexador S0 e S1 como bits de entrada e utiliza A como bit de controle , assim é possível controlar de qual entrada virá o que resultará na saída S.

## 2.5 Simulação do circuito no ModelSim:

Seguem os sinais de onda gerados no software:

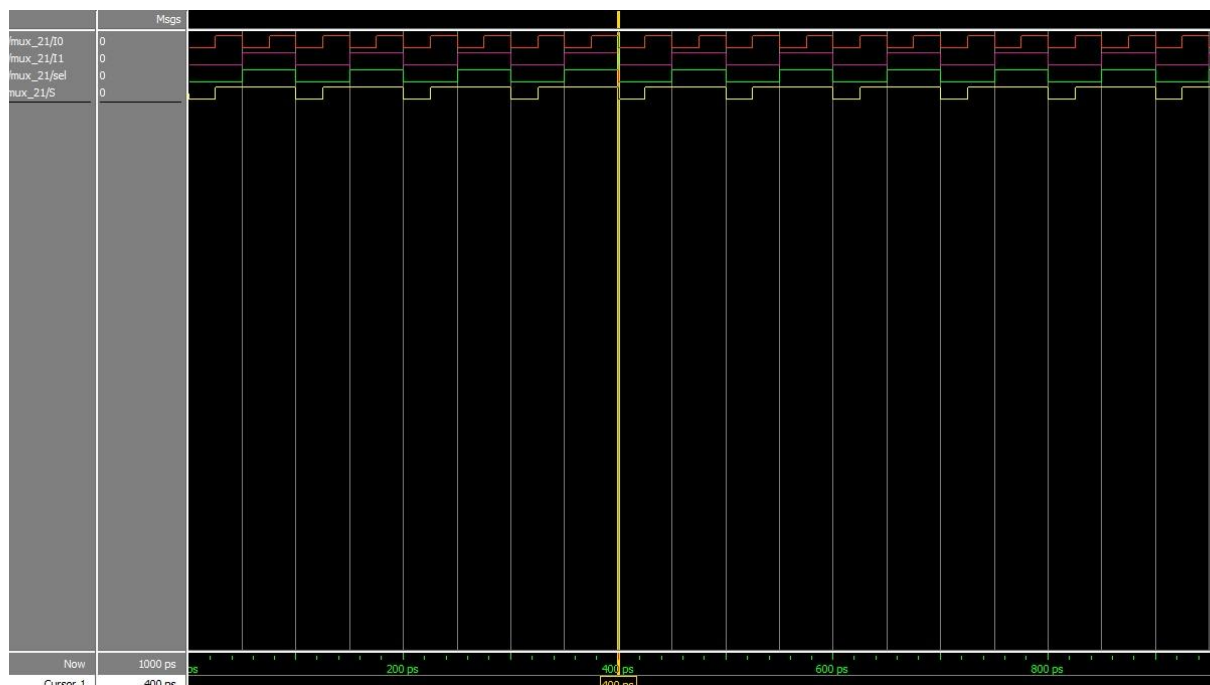


Imagem 05: Painel de ondas do Modelsim Altera para o módulo 1

Os sinais de onda foram gerados com o objetivo de testar múltiplas entradas e suas respectivas saídas através do gráfico lógico produzido pelo software de simulação. Os sinais



estão representados na seguinte ordem: entradas *I0*, *I1* e *sel*. Saída *S*. Como estímulos, foram usados sinais de onda do tipo clock com período de 50ps para *I0* e períodos de 100ps, ambos com *duty cycle* de 50ps.

Tais faixas produzidas foram capazes de gerar todas as combinações da tabela verdade.

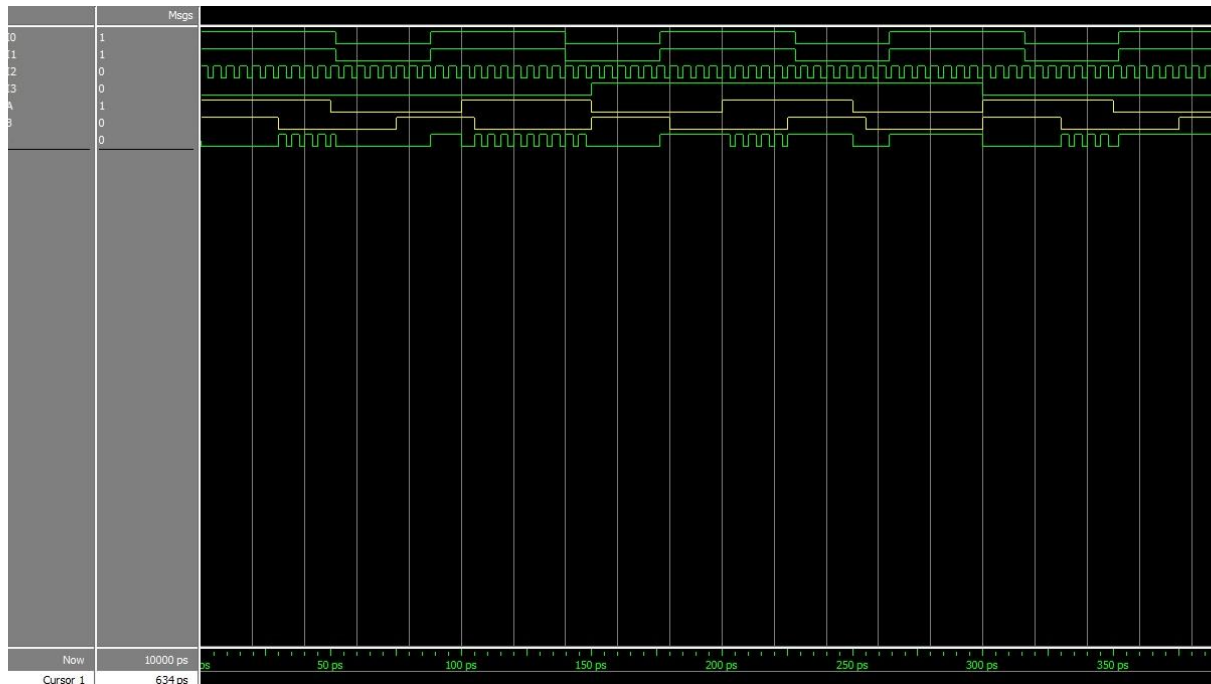


Imagem 06: Pannel de ondas do Modelsim Altera para o módulo 2

Nesta segunda imagem estão representados os sinais de entrada em ordem (*I0*...*I3*) com a cor verde, os bits de sinal com a cor amarela e a saída final em verde.

Para realizar os testes, foram utilizados os mesmos sinais de clock disponibilizados no arquivo de apresentação do projeto.

### 3. Análise dos resultados:

Levando em consideração os dois objetivos principais deste relatório, que foi desenvolvido em duas partes. No qual foi desenvolvido um circuito digital em Verilog HDL de um multiplexador de dois canais. Após isso, foi utilizado o módulo criado como parte de um multiplexador de quatro canais, implementado por meio da estrutura hierárquica. Dessa maneira, os resultados simulados no ModelSim Altera demonstram de forma correta os valores de *I0* e de *I1*, e a saída respeita a estrutura hierárquica, o que confirma que o código feito no Quartus II foi executado com sucesso.

Neste código percebemos que a comparação feita nos 2 MUX anteriores são verdadeiras. Com isso, analisamos as variáveis de controle *B*, *A* e a saída *S*, uma vez que o resultado deve

obedecer o valor controlado por elas. Como no exemplo acima, quando  $A=0$ ,  $B=1$ , então a saída deve ser o valor de  $I1$ . O que comprova o que foi demonstrado na simulação do ModelSim Altera.