

**CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS –
CEFETMG**

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

Comparador de 4 bits

Relatório 3

**BELO HORIZONTE
2021**

Sumário

Objetivo	2
Desenvolvimento	2
2.1 Módulos criados	2
2.3 Circuito lógico	4
2.4 Simulação do circuito no ModelSim:	5
Análise dos resultados	6

1. Objetivo

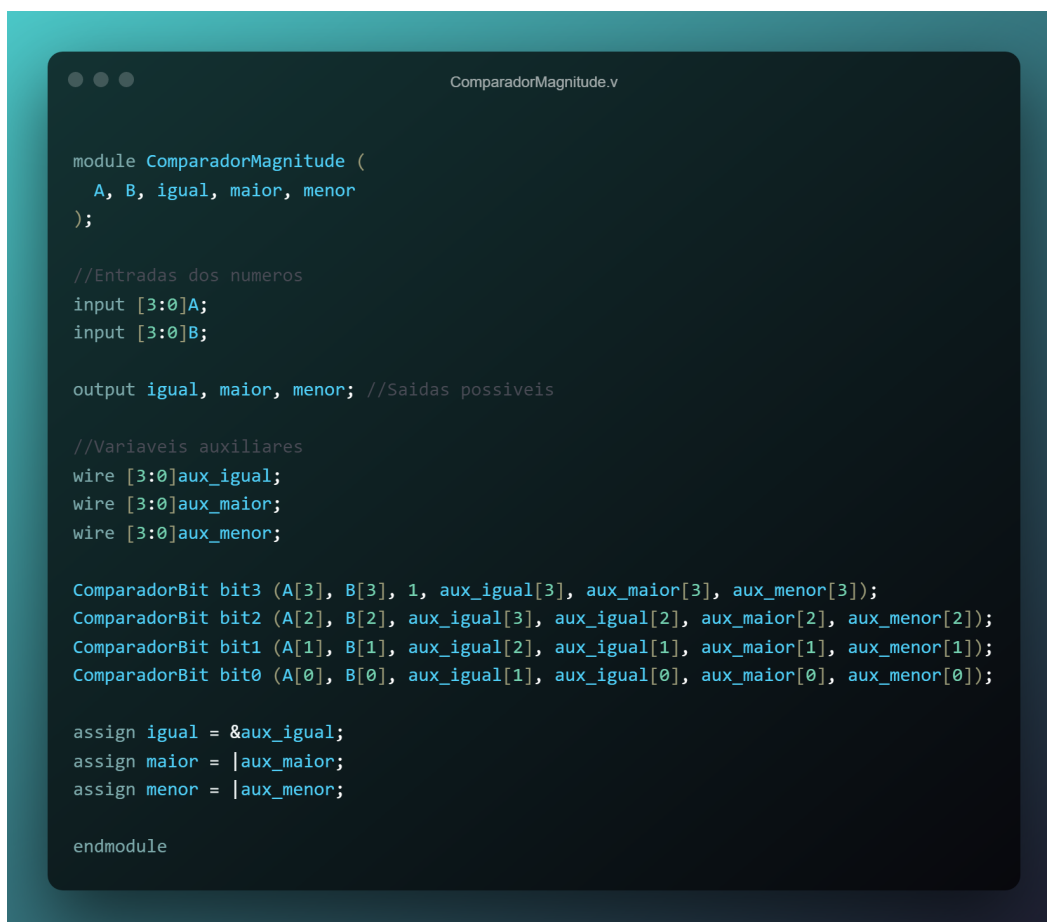
O objetivo principal deste relatório é apresentar um projeto desenvolvido utilizando a linguagem de máquina Verilog HDL e o software Quartus, juntamente com o Modelsim Altera para simulação. O projeto se trata de uma extensão do comparador de bits produzido anteriormente. Desta vez, ao invés de comparar dois bits, o circuito resultante é capaz de comparar dois valores de 4 bits.

2. Desenvolvimento

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

2.1 Módulos criados

Para desenvolver o projeto, foram necessários dois módulos. O módulo principal foi denominado “ComparadorMagnitude” e é a porta de entrada do circuito final. Segue o código do módulo:



```
ComparadorMagnitude.v

module ComparadorMagnitude (
    A, B, igual, maior, menor
);

    //Entradas dos numeros
    input [3:0]A;
    input [3:0]B;

    output igual, maior, menor; //Saidas possiveis

    //Variaveis auxiliares
    wire [3:0]aux_igual;
    wire [3:0]aux_maior;
    wire [3:0]aux_menor;

    ComparadorBit bit3 (A[3], B[3], 1, aux_igual[3], aux_maior[3], aux_menor[3]);
    ComparadorBit bit2 (A[2], B[2], aux_igual[3], aux_igual[2], aux_maior[2], aux_menor[2]);
    ComparadorBit bit1 (A[1], B[1], aux_igual[2], aux_igual[1], aux_maior[1], aux_menor[1]);
    ComparadorBit bit0 (A[0], B[0], aux_igual[1], aux_igual[0], aux_maior[0], aux_menor[0]);

    assign igual = &aux_igual;
    assign maior = |aux_maior;
    assign menor = |aux_menor;

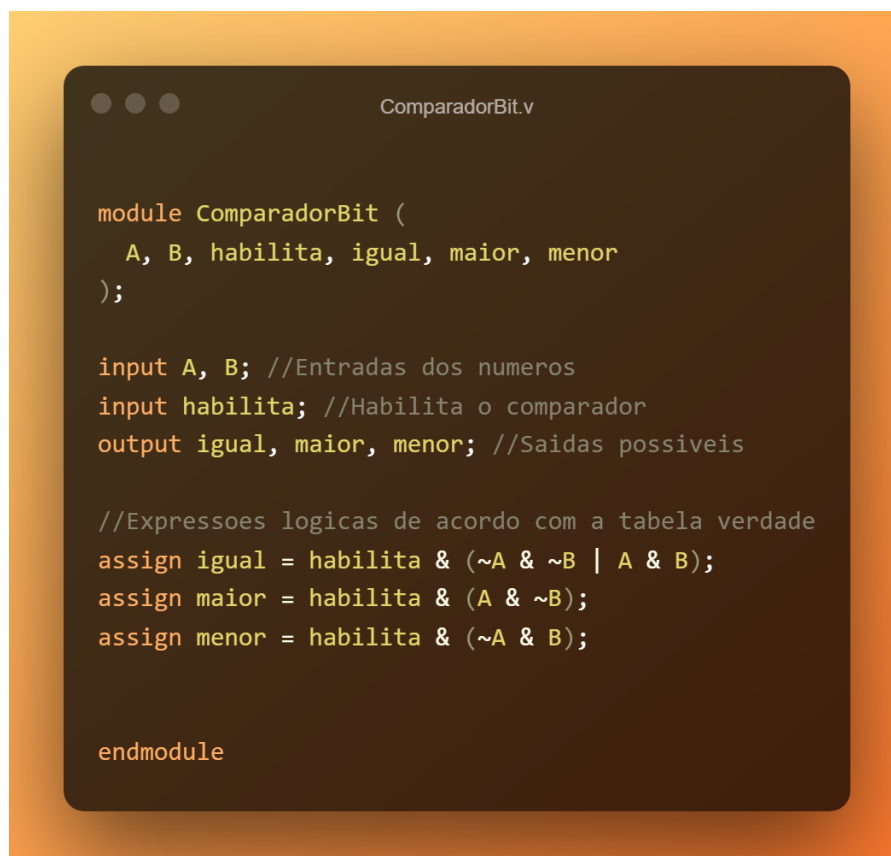
endmodule
```

Imagem 01: Módulo principal

O módulo acima foi definido de forma a possuir 2 entradas (A e B), que contam com 4 bits cada. Como saída, possui os bits de sinal *igual*, *maior* e *menor*, que representam o resultado da comparação.

Além desses *inputs/outputs*, o módulo precisa de outras 3 variáveis do tipo *wire* para realizar a conexão com o outro módulo, responsável por comparar números de um bit. Assim, estes *wires* são utilizados a fim de trafegar os sinais entre os dois componentes. É possível notar que o “ComparadorMagnitude” realiza a chamada do comparador de bits quatro vezes, tendo em vista que os números possuem essa magnitude.

A lógica funciona da seguinte forma: a cada chamada são passados como parâmetros um bit de cada número, o sinal de *enable* e os *wires* responsáveis por trazer os resultados. Nota-se que a comparação se inicia no bit mais significativo, pois uma vez que o bit de um dos números for 1 e o do outro 0, é possível acabar com a comparação naquele instante, pois se os bits forem diferentes, o sinal de habilita indicará que as próximas comparações não serão necessárias, tendo em vista que este sinal é determinado pelo *wire* que indica a igualdade dos bits.



```
module ComparadorBit (
    A, B, habilita, igual, maior, menor
);

input A, B; //Entradas dos numeros
input habilita; //Habilita o comparador
output igual, maior, menor; //Saidas possiveis

//Expressoes logicas de acordo com a tabela verdade
assign igual = habilita & (~A & ~B | A & B);
assign maior = habilita & (A & ~B);
assign menor = habilita & (~A & B);

endmodule
```

Imagem 02: Módulo secundário

O último módulo, representando acima, funciona da mesma forma que o comparador apresentando no relatório 2. A única diferença é o sinal de habilita, introduzido como uma nova porta de entrada. Como dito anteriormente, este componente recebe os dois bits e realiza a comparação para identificar se os mesmos são iguais, ou se existe um bit diferente. Percebe-se que o bit de habilita está junto da porta lógica que realiza a comparação. Desse modo, caso a comparação esteja habilitada (sinal 1), o resultado será dado normalmente. Caso contrário, o resultado será sempre 0.

Com esses dois módulos, foi possível desenvolver um circuito digital utilizando a lógica da estrutura hierárquica.

2.3 Circuito lógico

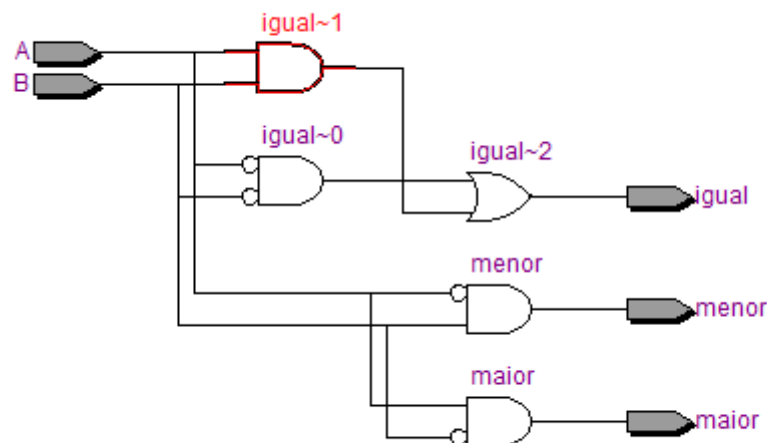


Imagem 03: Circuito gerado a partir da tabela verdade de dois bits

Descrição do circuito lógico:

- Tem-se duas entradas, cada uma representando um bit, as entradas A e B
- A porta lógica AND igual~1 recebe os bits A e B e, sua saída gera uma entrada para a porta OR igual~2 que gera a saída **igual**
- A porta lógica AND igual~0 recebe os bits A e B barrados e, sua saída gera a outra entrada da porta lógica OR igual~2 que gera a saída **igual**
- A porta lógica AND menor recebe como entrada o bit A barrado e o bit B e, gera a saída **menor**
- A porta lógica And maior recebe o bit A e o bit B barrado e, gera a saída **maior**

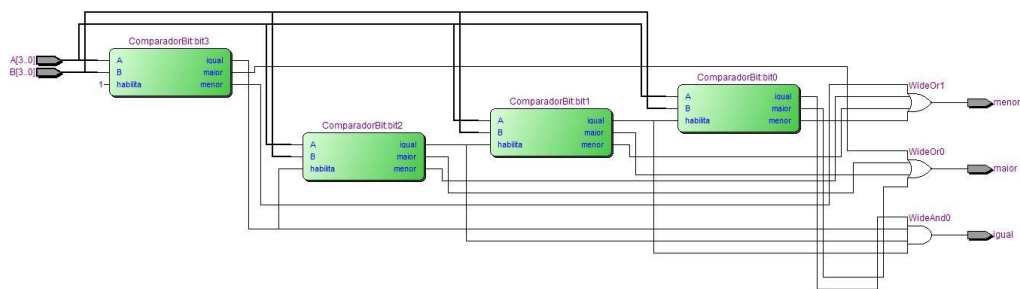


Imagem 04: Circuito complexo para comparar 8 bits

Acima temos a imagem referente ao comparador de 4 bits em que cada uma das caixas verdes presentes no circuito lógico se refere a um comparador de um bit explicado anteriormente.

Com relação ao funcionamento do circuito comparador de 4 bits, este compara o cada bit de números em binário formados por 4 bits, comparando do bit mais significativo ao ao bit menos significativos de cada número para verificar se um número é maior, menor ou igual ao outro.

Para realizar a comparação de cada bit, há três entradas, a entrada A que é constituída de 1 bit, a entrada B que também é constituída de um bit e, também, a entrada habilita que serve de controle para que a comparação de bits seja realizada, na comparação do primeiro bit, ou seja, o bit mais significativo de ambos os números, a entrada habilita recebe 1(nível lógico ALTO).

Com relação a saída, temos três casos, se o bit do número correspondente a entrada A for maior , a saída maior recebe 1, e as demais 0, se o bit do número correspondente a entrada B for maior , então a saída menor recebe 1, além disso, quando os bits correspondentes às entradas A e B forem iguais , a saída igual recebe 1 e a mesma funcionará como a entrada habilita do próximo comparador de bits , indicando se será necessário comparar o próximo bit de ambos os números ou não.

2.4 Simulação do circuito no ModelSim:

Seguem os sinais de onda gerados no software:

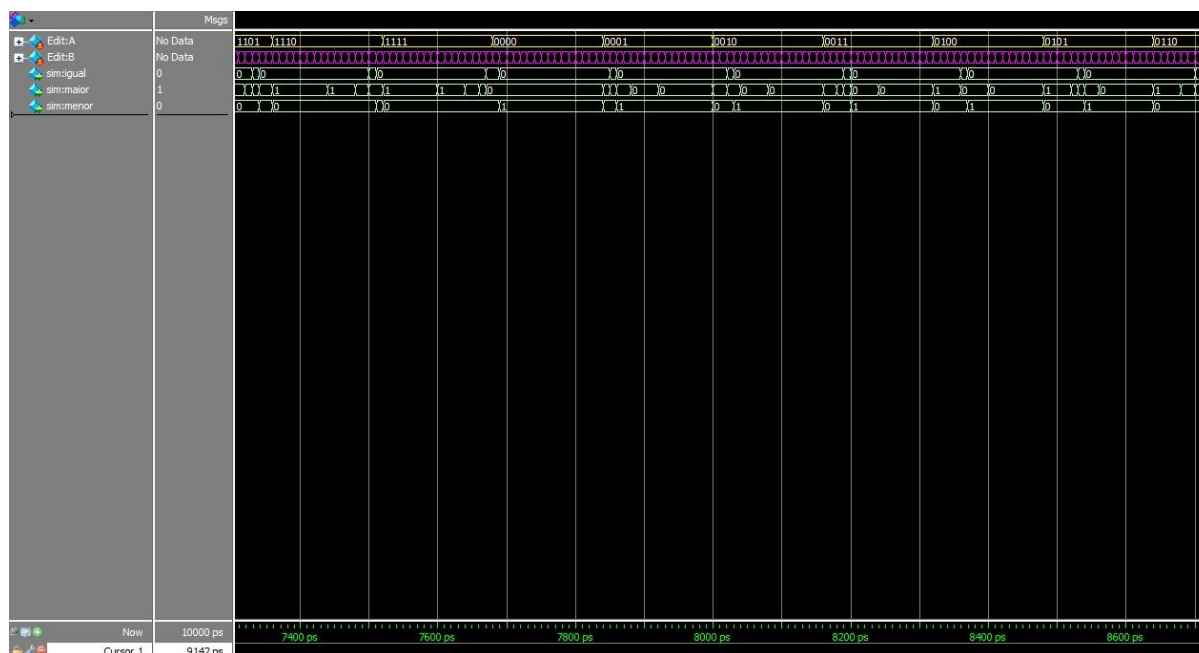


Imagem 05: Painel de ondas do Modelsim Altera

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. Para tanto, foram gerados dois sinais de ondas do tipo counter. O primeiro deles na entrada A, variando de 0 a 10k ps e com período de 160ps. O segundo na entrada B, variando também de 0 a 10k ps, porém com período de 10ps (16x menor), tendo em vista que o circuito realiza 16 comparações por cada valor assumido por B.

3. Análise dos resultados

Utilizando-se de dois módulos para comparar Magnitude e os Bits de dois números foi possível desenvolver um circuito digital utilizando a lógica da estrutura hierárquica. Seus dois números de entrada de magnitude de 4 bits, de valor 0 ou 1 resultam em 3 saídas, cada saída recebe o valor 1 se for verdade e 0 em caso negativo. Assim é possível averiguar se as entradas A, B são iguais, a entrada “A” é maior ou a entrada “A” é menor. Isso é demonstrado na simulação do circuito no ModelSim Altera. Assim, foi possível cumprir o objetivo inicial de comparar a magnitude de dois números de 4 bits proposto para o projeto.