Relatório Questão 5 - Projeto de Sistemas Digitais - P1: Primeiros passos em Verilog

Aluno: Vítor Aguiar Tavares (vat2)

Relatório Questão 5

Um Comparador é um circuito digital que executa a operação de comparação de sinais obtidos em sua entrada de dados, com isso os sinais recebidos passam por uma lógica binária digital e emitem um sinal de sua saída de acordo com o seu resultado.

Um Comparador Simples contém apenas a operação de igualdade (A = B - Uma porta XOR indica !=, uma porta XNOR indica ==), um Comparador Avançado é capaz de executar, além da operação básica de igualdade, as operações "Maior que" (A > B) e "Menor que" (A < B).

Em resumo, um Comparador de n-bits recebe duas entradas de n-bits e retorna um sinal relativo ao resultado das três operações.

Questão 5 - a)

O desenho de um Comparador de 4 bits é composto pelas portas XNOR, NOR, OR e AND. Todos os bits das entradas passam pelas portas XNOR para verificar a igualdade bit por bit e, em sequência, passam por uma AND para verificar se todas as saídas das XNOR foram 1. Caso a saída da AND seja 1, significa que as palavras de entrada A e B são iguais. Além disso, todas as entradas passam por uma sequência de ANDs seguida por uma OR para verificar se o elemento A é maior que o elemento B. Por fim, caso tanto a saída da AND (para verificar se A é igual a B) e a saída da OR (para verificar se A é maior que B) seja 0, os resultados são conectados a uma NOR que indicaria que a entrada A é menor que B.

Questão 5 - b)

Para que o Comparador de 8 bits formado através de dois Comparadores de 4 bits funcione, precisamos adicionar 3 novas entradas nas caixinhas que representam os circuitos digitais dos Comparadores, entradas: A=B_IN, A<B_IN e A>B_IN. No primeiro Comparador de 4 bits, “setamos” apenas a entrada A=B_IN como sendo 1 e as A<B_IN e A>B_IN como 0. Em sequência, conectamos as saídas A=B, A<B e A>B nas entradas A=B_IN, A<B_IN e A>B_IN do segundo Comparador de 4 bits. Dessa forma, podemos garantir que as operações irão funcionar como esperado.

Questão 5 - c)

O código em Verilog de um Comparador de 4 bits é simples e direto. Se for apenas um Comparador de 4 bits, temos as entradas: input [3:0] A, B - Vetores de palavras de 4 bits - e as saídas: output wire G, L, E. A composição do módulo são 3 linhas contendo "assign" (sinais), que representam cada uma das operações e são vinculadas a cada um dos outputs: "assign G = (A > B); assign L = (A < B); assign E = (A == B);".

Se for um Comparador de 4 bits com o objetivo de ser usado para compor outros Comparadores (um Comparador de 8 bits, por exemplo), precisamos adicionar as três entradas A=B_IN, A<B_IN e A>B_IN e alterar um pouco a estrutura das operações que serão retornadas nos sinais de output. Entradas: input G_IN, input L_IN, input E_IN - entradas novas A=B_IN, A<B_IN e A>B_IN - input [3:0] A, B - Vetores de palavras de 4 bits - e as saídas: output wire G, L, E. A composição do módulo são 3 linhas contendo "assign" (sinais), que representam cada uma das operações e são vinculadas a cada um dos outputs: "assign G = (A > B) || ((A == B) && G_IN); assign L = (A < B) || ((A == B) && L_IN); assign E = ((A == B) && E_IN);".

```
module comparador_4bits(output wire G, L, E, input wire [3:0] A, B);

   assign G = (A > B);
   assign L = (A < B);
   assign E = (A == B);

endmodule
```

ou

```
module Comparador_4bits (output wire G, L, E, input G_IN, input L_IN,
input E_IN, input wire [3:0] A, B);

   assign G = (A > B) || ((A == B) && G_IN);
   assign L = (A < B) || ((A == B) && L_IN);
   assign E = ((A == B) && E_IN);

endmodule
```

Questão 5 - d)

Para fazer um Comparador de 8 bits utilizando dois Comparadores de 4 bits em Verilog, temos que adicionar o módulo de Comparador de 4 bits que contém as trêns entradas A=B_IN, A<B_IN e A>B_IN, assim como foi descrito na questão acima. O novo módulo (Comparador_8bits) contém as seguintes entradas: input wire [7:0] A, B - Vetores de palavras de 8 bits - e as saídas: output wire G, L, E. Além disso, é necessário conter uma linha para adicionar os fios que vem da saída do primeiro Comparador para as entradas A=B_IN, A<B_IN e A>B_IN do segundo Comparador: "wire G_old, L_old, E_old;". Após isso, basta chamar os módulos de Comparadores de 4 bits (Dei o nome de c1 e c2) e se atentar para as entradas em cada um. Na chamada da "caixinha" c1, temos que "setar" as entradas A=B_IN, A<B_IN e A>B_IN como sendo entradas de 1bit e determinar seus valores conforme foi descrito nas questões acima (A=B_IN = 1, A<B_IN = 0 e A>B_IN = 0): "Comparador_4bits c1(G_old, L_old, E_old, 1'b0, 1'b0, 1'b1, A[3:0], B[3:0]);". Na chamada da segunda "caixinha", "caixinha" c2, basta utilizar as saídas G_old, L_old e E_old nas entradas A=B_IN, A<B_IN e A>B_IN: "Comparador_4bits c2(G, L, E, G_old, L_old, E_old, A[7:4], B[7:4]);". Dessa forma, os dois módulos ficarão conectados e o resultado será o esperado.

```
module Comparador_4bits (output wire G, L, E, input G_IN, input L_IN,
input E_IN, input wire [3:0] A, B);

   assign G = (A > B) || ((A == B) && G_IN);
   assign L = (A < B) || ((A == B) && L_IN);
   assign E = ((A == B) && E_IN);

endmodule

module Comparador_8bits (output wire G, L, E, input wire [7:0] A, B);

   wire G_old, L_old, E_old;
   Comparador_4bits c1(G_old, L_old, E_old, 1'b0, 1'b0, 1'b1, A[3:0],
B[3:0]);
   Comparador_4bits c2(G, L, E, G_old, L_old, E_old, A[7:4], B[7:4]);

endmodule
```

Questão 5 - e)

Para criar a Bancada de Testes do Comparador de 8 bits, teremos que criar o módulo “stimulos”, setar as entradas e saídas, instanciar o módulo que será usado no teste, descrever os acontecimentos durante o teste e gerar o arquivo .vcd que será utilizado no GTK Waves para visualizar as ondas resultantes da saída do arquivo teste.

-> Primeiro momento, vamos criar o módulo:

```
module stimulus;
```

- Esse módulo não recebe os parâmetros de input e output.

-> Em sequência, descrevemos as entradas e saídas que serão utilizados pelo módulo do Comparador de 8 bits:

```
// Inputs
reg [7:0] A;
reg [7:0] B;

// Outputs
wire G;
wire L;
wire E;
```

- As entradas precisam ser “setadas” como registradores e as saídas como fios.

-> Após isso, instanciamos o módulo do Comparador de 8 bits:

```
// Instantiate the Unit Under Test (UUT)
Comparador_8bits uut (
    .A(A),
    .B(B),
    .G(G),
    .L(L),
    .E(E)
);
```

- Precisamos dar um nome para essa "caixinha", nesse caso, o nome foi uut - Unit Under Test (UUT)

-> Então, após inicializar, criamos o arquivo .vcd que será usado pelo GTK:

```
initial begin
    $dumpfile ("waves.vcd");
    $dumpvars (0, stimulus);
```

- "waves" foi o nome que foi dado ao arquivo que será lido pelo GTK Waves.

-> “Setamos” a inicialização dos inputs A e B do módulo do Comparador de 8 bits e descrevemos os eventos que ocorrerão durante o teste:

```
        // Initialize Inputs
        A = 8'b10000000;
        B = 8'b10000000;

        #20 A = 8'b11111111;
        #20 B = 8'b11111111;
        #20 A = 8'b10000000;
        #20 B = 8'b10000000;
        #20 A = 8'b11111111;
        #20 B = 8'b11111111;
        #20 A = 8'b10000000;
        #20 B = 8'b10000000;
        #20 A = 8'b11111111;
        #20 B = 8'b11111111;
        #20 A = 8'b10000000;
        #20 B = 8'b10000000;
        #20 A = 8'b11111111;
        #20 B = 8'b11111111;
        #20 A = 8'b10000000;
        #20 B = 8'b10000000;

        $finish;
    end
```

- As entradas são 8 bits, portanto descrevemos 8'b seguido do valor para atribuir o valor esperado. "#20" indica o atraso em ns para a realização do evento.

-> Antes de finalizar, colocamos um monitor para printar os principais elementos descritos no teste (Tempo, Entrada A, Entrada B, Operação Maior Que G, Operação Menor Que L e a Operação Igual).

```
    initial begin
        $monitor ("T=%3d A=%d,B=%d,G=%d L=%d E=%d \n", $time, A, B, G,
L, E);
    end
endmodule
```

- Fim do módulo.

Com isso, basta compilar, executar e abrir o arquivo "waves.vcd" no GTK Waves e verificar as ondas resultantes da descrição da Bancada de Testes do Comparador de 8 bits utilizando dois Comparadores de 4 bits.

→ Comandos (Linux):

```
$ iverilog -o testBank stimulus.v Comparador_8bits.v
$ ./testBank
$ gtkwave waves.vcd
```

ou

```
$ iverilog -o testBank stimulus.v Comparador_8bits.v && ./testBank && gtkwave
waves.vcd
```