

Curso: Bacharelado em Sistemas de Informação
Disciplina: Introdução aos Sistemas Lógicos
2º Semestre de 2023
Professor Marcos Augusto Menezes Vieira (*mmvieira@dcc.ufmg.br*)
Monitor Filipe Pirola Santos (*filipepirolasantos@hotmail.com*)

Data de entrega: **29/11/2023**

Trabalho Prático 2: Desenvolvimento de um Contador Síncrono de 4 Bits em Verilog

1 Introdução

Com o fim da construção da Arena MRV, a artista Ariana Grande foi contratada para fazer o show de inauguração do estádio. Devido ao prazo apertado para a realização do evento, os cientistas da computação da arena foram acionados para começar a desenvolver os softwares, necessários na apresentação, o mais rápido possível. Por conseguinte, você foi escolhido para contruir o cronômetro utilizado para auxiliar a artista no intervalo entre suas músicas. Portanto, você precisa desenvolver um circuito que resolva esse problema e garanta que o show aconteça sem erros.



Figura 1: Palco montado, na arena, para a apresentação

2 Objetivo

O objetivo deste trabalho prático é projetar e implementar um contador síncrono de 4 bits em Verilog, no modelo de comportamento. O contador deve ser capaz de contar de 0 a 9 e, em seguida, reiniciar a contagem, usando um sinal de clock como referência e com capacidade de redefinir a contagem para zero através de um sinal de reset.



Figura 2: Cronômetro utilizado para auxiliar a cantora

2.1 Especificações do Trabalho

- Implemente um módulo Verilog denominado `contador_sequencial` que contenha os seguintes elementos:
 1. Uma entrada para o sinal de clock (`clk`).
 2. Uma entrada para o sinal de reset (`reset`).
 3. Uma saída de 4 bits para a contagem atual (`count`).
- O contador deve ser síncrono com o sinal de clock, ou seja, a contagem só deve ser atualizada nas transições de borda de subida do sinal de clock.
- Além disso, ele deve ser capaz de contar de 0 a 9 e, em seguida, reiniciar a contagem de 0, caso atinja o valor 9. A reinicialização deve ser acionada pelo sinal de reset.
- Ao aplicar o sinal de reset, o contador deve ser imediatamente redefinido para zero, independentemente da contagem atual.
- Crie um arquivo de teste em Verilog que instancie o módulo `contador_sequencial`, forneça sinais de clock e reset adequados e monitore a saída do contador.

3 Caso de teste

Implemente um arquivo que teste seu módulo para a seguinte situação:

1. O contador é acionado durante 5 segundos,
2. Em sequência, o reset é acionado,
3. Por fim, o contador fica ligado pelos próximos 12 segundos.

4 EDA Playground

Verilog é uma linguagem de descrição de hardware (Hardware Description Language - HDL) usada para modelar e simular sistemas eletrônicos ao nível de circuito. Apesar de possuir sintaxe semelhante a da linguagem C, Verilog é uma linguagem especificamente orientada à descrição da

estrutura e do comportamento do hardware. O **modelo estrutural** é a forma pela qual o projetista descreve a interconexão entre os componentes que fazem parte do circuito (i.e. define as relações entre entrada e saída de um circuito); o **modelo comportamental**, por sua vez, descreve o funcionamento de cada um dos componentes do circuito. Neste trabalho criaremos um **modelo comportamental** e uma **simulação lógica** para testar o modelo.

A ferramenta EDA Playground é bastante utilizada para criar e simular descrições de hardware devido à sua praticidade e facilidade de uso. A Figura 3 mostra uma implementação dos casos de teste e o log da simulação usando essa ferramenta. A Figura 4 ilustra as formas de onda relativas aos casos de teste usados na simulação.

EDA Playground

5 O que entregar?

Cada aluno deve submeter um arquivo **MATRÍCULA_NOME.zip** contendo, na raiz,

1. Documentação (em pdf) breve contendo decisões de implementação, as máquinas de estado utilizadas como referência para a implementação do código e os diagramas de onda gerados com os testes,
2. O arquivo “desing.sv”, contendo a implementação, e
3. O arquivo “testbench.sv”, contendo o caso de teste.

6 Avaliação

O trabalho será avaliado com base nos seguintes critérios:

1. Implementação correta e funcional do contador síncrono de 4 bits em Verilog.
2. Teste eficaz do contador com um arquivo de teste que inclui casos que verifiquem a contagem de 0 a 9 e a reinicialização correta.
3. Compreensão dos princípios de circuitos sequenciais, incluindo a utilização de registradores e lógica de controle para gerenciar a contagem e a reinicialização.
4. Qualidade e organização do código Verilog, incluindo comentários explicativos.

7 Considerações finais

- A legibilidade do código será considerada na correção! Use nomes intuitivos, indente seu código e faça bom uso de comentários. :)
- Fique atento quanto ao uso de eventos do bloco *always*. Trocar uma borda de descida por uma de subida gerará defasagem na forma de onda e deixará sua resposta incorreta.
- É importante fornecer documentação explicativa no relatório, incluindo diagramas e explicações do circuito.
- Qualquer dúvida podem entrar em contato comigo por e-mail.

Brought to you by **DOULOS**

▼ Languages & Libraries

Testbench + Design

SystemVerilog/Verilog

UVM / OVM

None

Other Libraries

None
OVL 2.8.1
SVUnit 2.11

☐ Enable TL-Verilog

☐ Enable Easier UVM

☐ Enable VUnit

▼ Tools & Simulators

Icarus Verilog 0.10.0 11/23/14

Compile Options

-Wall -g2012

Run Options

Run Options

☒ Open EPWave after run

☐ Show output file after run

☐ Download files after run

► Examples

► Community

208

testbench.sv

```

1 module test_iluminacao;
2
3     reg clk;           // Sinal de clock
4     reg rst;           // Sinal de reset
5     wire [2:0] state;  // Sinal de cor (3 bits)
6
7     // Instanciar a bola iluminada
8     semaforo_transito u_semaforo_transito (
9         .clk(clk),
10        .rst(rst),
11        .state(state)
12    );
13
14    // Gerar um clock com um periodo de 10 unidades de tempo (ajuste conforme necessário)
15    always begin
16        #5 clk = ~clk;
17    end
18
19    // Inicializar o sinal de reset
20    initial begin
21        $dumpfile("dump.vcd"); // Gera o diagrama de ondas
22        $dumpvars;             // Requisito do EPWave
23        clk = 1;
24        rst = 0;
25        #10 rst = 1;
26        #100 rst = 0; // Simulação dura 100 unidades de tempo
27        $finish;
28    end
29
30    // Monitorar a cor da bola iluminada
31    always @(posedge clk) begin
32        case (state)
33            3'b000: $display("DESLIGADO");
34            3'b001: $display("BRANCO");
35            3'b010: $display("ROXO");
36            3'b100: $display("ROSA");
37        endcase
38    end
39
40 endmodule

```

SV/Verilog Testbench

Log Share

DESLIGADO
BRANCO
ROXO
ROSA
BRANCO
ROXO
ROSA
BRANCO
ROXO

Figura 3: Exemplo de implementação do caso de uso: em testbench.sv temos a implementação de um caso de teste e, abaixo, o log da simulação.a

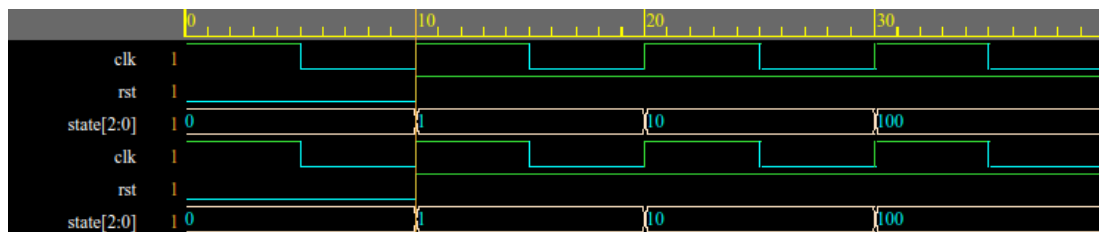


Figura 4: Formas de onda do caso de teste