Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática – ICEI
Arquitetura de Computadores I

ARQ1 _ Aula_09
Tema: Introdução à linguagem Verilog

Preparação

Como preparação para o início das atividades, recomendam-se

a.) leitura prévia do resumo teórico, do detalhamento na apostila e referências recomendadas
b.) estudo e testes dos exemplos
c.) assistir aos seguintes vídeos:

https://www.youtube.com/watch?v=o6kS7izbM7o

http://www.asic-world.com/verilog/art_testbench_writing2.html
http://referencedesigner.com/tutorials/verilogexamples/verilog_ex_06.php
http://www.testbench.in/TB_08_CLOCK_GENERATOR.html

Orientação geral:

Apresentar todas as soluções em apenas um arquivo com formato texto (.txt).

As implementações e testes dos exemplos em Verilog (.v) fornecidos como pontos de partida, também fazem parte da atividade e deverão ser entregues os códigos fontes separadamente. As saídas de resultados, opcionalmente, poderão ser copiadas ao final do código, como comentários.

Outras formas de solução são opcionais; e, se entregues, contarão como atividades extras (.c ou .py). Os programas com funções desenvolvidas em C ou Python (usar modelos para verificação automática de testes das respostas), se entregues, também deverão estar em arquivos separados, com o código fonte, para serem compilados e testados. As execuções deverão, preferencialmente, serem testadas mediante uso de entradas e saídas padrões e os dados/resultados usados para testes armazenados em arquivos textos. Os resultados poderão ser anexados ao código, ao final, como comentários.

Os *layouts* de circuitos deverão ser entregues no formato (.circ), identificados internamente. Figuras exportadas pela ferramenta serão aceitas como arquivos para visualização, mas não terão validade para fins de avaliação. Separar versões completas (a) e simplificadas (b).

Arquivos em formato (.pdf), fotos, cópias de tela ou soluções manuscritas serão aceitos como recursos suplementares para visualização, e não terão validade para fins de avaliação.

Atividade: Circuitos sequenciais

01.) Projetar e descrever em Verilog um módulo gerador de ***clock***.
O nome do arquivo deverá ser Exemplo_0901.v, e poderá seguir o modelo descrito abaixo.
Incluir previsão de testes e verificação da carta de tempo usando GTKWave.

```
// ---------------------------
// -- test clock generator (1)
// ---------------------------

module clock ( output clk );
reg      clk;

 initial
  begin
   clk = 1'b0;
  end

 always
  begin
   #12 clk = ~clk;
  end

endmodule // clock ( )


module Exemplo_0901;

 wire  clk;
 clock CLK1 ( clk );

 initial begin
  $dumpfile ( "Exemplo_0901.vcd" );
  $dumpvars;

  #120 $finish;
 end

endmodule // Exemplo_0901 ( )
```

02.) Projetar e descrever em Verilog módulos geradores de pulso (***pulse***) e gatilho (***trigger***). O nome do arquivo deverá ser Exemplo_0902.v, e poderá seguir o modelo descrito abaixo. Incluir previsão de testes e verificação da carta de tempo usando GTKWave.

```
// ----------------------------
// -- test clock generator (2)
// ----------------------------

module clock ( output clk );
 reg      clk;

 initial
  begin
   clk = 1'b0;
  end

 always
  begin
   #12 clk = ~clk;
  end
endmodule

module pulse ( signal, clock );
 input  clock;
 output signal;
 reg      signal;

 always @ ( clock )
  begin
        signal = 1'b1;
  #3  signal = 1'b0;
  #3  signal = 1'b1;
  #3  signal = 1'b0;
  end
endmodule // pulse

module trigger ( signal, on, clock );
 input  on, clock;
 output signal;
 reg      signal;

 always @ ( posedge clock & on )
  begin
  #60 signal = 1'b1;
  #60 signal = 1'b0;
  end
endmodule // trigger
```

```
module Exemplo_0902;

 wire   clock;
 clock clk ( clock );

 reg   p;

 wire  p1,t1;

 pulse   pulse1  ( p1, clock );
 trigger trigger1 ( t1, p, clock );

 initial begin
   p = 1'b0;
 end

 initial begin
  $dumpfile ( "Exemplo0902.vcd" );
  $dumpvars ( 1, clock, p1, p, t1 );

  #060 p = 1'b1;
  #120 p = 1'b0;
  #180 p = 1'b1;
  #240 p = 1'b0;
  #300 p = 1'b1;
  #360 p = 1'b0;
  #376 $finish;
 end

endmodule // Exemplo_0902
```

03.) Projetar e descrever em Verilog módulos geradores de pulso (***pulse***) com períodos diferentes. O nome do arquivo deverá ser Exemplo_0903.v, e poderá seguir o modelo descrito a seguir. O gerador de ***clock*** do Exemplo0901.v deverá ser previamente isolado em um arquivo único cujo nome deverá ser **clock.v**, para uso posterior. Incluir previsão de testes e verificação da carta de tempo usando GTKWave.

```
// ----------------------------
// -- test clock generator (3)
// ----------------------------

`include "clock.v"

module pulse1 ( signal, clock );
 input   clock;
 output signal;
 reg      signal;

 always @ ( posedge clock )
  begin
       signal = 1'b1;
  #4  signal = 1'b0;
  #4  signal = 1'b1;
  #4  signal = 1'b0;
  #4  signal = 1'b1;
  #4  signal = 1'b0;
  end
endmodule // pulse

module pulse2 ( signal, clock );
 input  clock;
 output signal;
 reg      signal;

 always @ ( posedge clock )
  begin
       signal = 1'b1;
  #5  signal = 1'b0;
  end
endmodule // pulse

module pulse3 ( signal, clock );
 input  clock;
 output signal;
 reg    signal;

 always @ ( negedge clock )
  begin
       signal = 1'b1;
  #15 signal = 1'b0;
  #15 signal = 1'b1;
  end
endmodule // pulse

module pulse4 ( signal, clock );
 input  clock;
 output signal;
 reg    signal;

 always @ ( negedge clock )
  begin
       signal = 1'b1;
  #20 signal = 1'b0;
  #20 signal = 1'b1;
  #20 signal = 1'b0;
  end
endmodule // pulse
```

```
module Exemplo_0903;

 wire  clock;
 clock clk ( clock );

 wire  p1,p2,p3,p4;

 pulse1   pls1   ( p1, clock );
 pulse2   pls2   ( p2, clock );
 pulse3   pls3   ( p3, clock );
 pulse4   pls4   ( p4, clock );

 initial begin
  $dumpfile ( " Exemplo0903.vcd" );
  $dumpvars ( 1, clock, p1, p2, p3, p4 );

  #480 $finish;
 end

endmodule // Exemplo_0903
```

04.) Projetar e descrever em Verilog um módulo gerador de pulso (***pulse***) com frequência igual ao dobro da frequência (metade do período) do gerador do Exemplo0901.v.
O nome do arquivo deverá ser Exemplo_0904.v.
Incluir previsão de testes e verificação da carta de tempo usando GTKWave.

05.) Projetar e descrever em Verilog um módulo gerador de pulso (***pulse***) com frequência igual a um terço da frequência (a três vezes o período) do gerador do Exemplo0901.v.
O nome do arquivo deverá ser Exemplo_0905.v.
Incluir previsão de testes e verificação da carta de tempo usando GTKWave.

Extra

06.) Projetar e descrever em Verilog um módulo
gerador de pulso (***pulse***) com marcação igual
a 4 unidades de tempo, sincronizado com a
borda de subida do gerador do Exemplo_0901.v.
O nome do arquivo deverá ser Exemplo_0906.v.
Incluir previsão de testes e verificação da
carta de tempo usando GTKWave.
DICA: Usar *always* @(posedge clk).

07.) Projetar e descrever em Verilog um módulo
gerador de pulso (***pulse***) com marcação igual
a 5 unidades de tempo, sincronizado com a
borda de descida do gerador do Exemplo_0901.v.
O nome do arquivo deverá ser Exemplo_0907.v.
Incluir previsão de testes e verificação da
carta de tempo usando GTKWave.
DICA: Usar *always* @(negedge clk).

08.) Projetar e descrever em Verilog um módulo
gerador de pulso (***pulse***) com marcação igual
a 6 unidades de tempo, sincronizado com
o nível alto e estável do gerador do
Exemplo_0901.v.
O nome do arquivo deverá ser Exemplo_0908.v.
Incluir previsão de testes e verificação da
carta de tempo usando GTKWave.
DICA: Usar *always* @(clk).

Instruções para ver as cartas de tempo no GTKWave:

01.) Abrir o módulo de visualização (GTKWave)

02.) Selecionar a pasta de trabalho:
File
Open
Exemplo_0901 (.vcd) (por exemplo)

03.) Selecionar os sinais desejados:
clk (sinal a ser visto)
clock (outro sinal a ser visto)
(selecionar, arrastar e soltar na coluna à direita)