Разработване на тестове

# Тест без проверка на резултатите



### Пример

| module lab1 #(parameter WIDTH = 8) ( input clock, input reset, output logic [WIDTH-1:0] sr);always\_ff @(posedge clock, posedge reset) if(reset) sr <= ‘0; else sr <= {sr[WIDTH-2:0], ~sr[WIDTH-1]};endmodule | `timescale 1ns / 1psmodule lab1\_test; logic clock = 0; logic reset; logic [7:0] sr; lab1 uut(.\*); initial forever #50 clock = ~clock; initial $monitor("%b",sr); initial begin reset = 1; #100 reset = 0; #5000 $finish; endendmodule |
| --- | --- |

00000000

00000001

00000011

00000111

00001111

00011111

00111111

01111111

11111111

11111110

11111100

11111000

11110000

11100000

11000000

10000000

00000000

00000001

00000011

# Тест с проверка на резултатите

## Пример 1

| `timescale 10ns / 1nsmodule parity #(parameter SIZE = 8)( input [SIZE-1:0] data, output odd);  assign odd = ^data;endmodule | module parity\_test; localparam N = 4; logic [N-1:0] data; logic odd; parity #(.SIZE(N)) uut (.\*); initial $monitor("%b -> %b", data, odd); initial begin data = 3'b111; #1 assert (odd == 1); data = 3'b000; #1 assert (odd == 0); data = 3'b001; #1 assert (odd == 1); data = 3'b101; #1 assert (odd == 0); data = 3'b100; #1 assert (odd == 0); // error data = 3'b110; #1 assert (odd == 0); $finish; endendmodule |
| --- | --- |

0111 -> 1

0000 -> 0

0001 -> 1

0101 -> 0

0100 -> 1

Error: Assertion violation

Time: 50 ns Iteration: 0 Process: /parity\_test/Initial22\_1 File: C:/usr/verilog/2018/sim/parity.sv

0110 -> 0

## Пример 2 -> дом. работа

import bcd::\*;

module bcd\_adder\_1d(

 input bcd\_t a,b,

 input cin,

 output bcd\_t s,

 output logic cout);

 // TODO

endmodule

---

import bcd::\*;

module bcd\_adder\_1d\_test;

 bcd\_t a, b, s;

 logic cin, cout;

 bcd\_adder\_1d uut(.\*);

 task test\_vector (

 input bcd\_t a\_test, b\_test,

 input cin\_test,

 input cout\_expected,

 input bcd\_t s\_expected);

 a = a\_test;

 b = b\_test;

 cin = cin\_test;

 #10 assert ((s\_expected == s) && (cout\_expected == cout))

 else $error("%0d + %0d + %0d = expected %0d%0d but was %0d%0d",a\_test, b\_test, cin\_test, cout\_expected, s\_expected, cout, s);

 endtask

 initial begin

 test\_vector(0,0,0,0,0);

 test\_vector(0,0,1,0,1);

 test\_vector(0,1,0,0,1);

 test\_vector(1,0,0,0,1);

 test\_vector(1,1,0,0,2);

 test\_vector(2,2,0,0,4);

 test\_vector(2,3,0,0,5);

 test\_vector(3,2,0,0,5);

 test\_vector(1,1,1,0,3);

 test\_vector(2,2,1,0,5);

 test\_vector(2,3,1,0,6);

 test\_vector(3,2,1,0,6);

 test\_vector(5,5,0,1,0);

 test\_vector(5,5,1,1,1);

 test\_vector(7,8,0,1,5);

 test\_vector(8,7,1,1,6);

 test\_vector(9,8,0,1,7);

 test\_vector(9,9,0,1,8);

 test\_vector(9,9,1,1,9);

 $finish;

 end

 initial $monitor("%0d + %0d + %0d = %0d%0d",a, b, cin, cout, s);

endmodule

0 + 0 + 0 = 00

0 + 0 + 1 = 01

0 + 1 + 0 = 01

1 + 0 + 0 = 01

1 + 1 + 0 = 02

2 + 2 + 0 = 04

2 + 3 + 0 = 05

3 + 2 + 0 = 05

1 + 1 + 1 = 03

2 + 2 + 1 = 05

2 + 3 + 1 = 06

3 + 2 + 1 = 06

5 + 5 + 0 = 10

5 + 5 + 1 = 11

7 + 8 + 0 = 15

8 + 7 + 1 = 16

9 + 8 + 0 = 17

Warning: 9 + 8 + 0 expected 17 but was 10

Time: 1700 ns Iteration: 0 Process: /bcd\_adder\_test/test\_vector File: ...

9 + 9 + 0 = 18

9 + 9 + 1 = 19

## Пример 3

import bcd::\*;

module bcd\_adder\_3d(

 input bcd\_t [2:0] op1, op2,

 input cin,

 output bcd\_t [2:0] sum,

 output logic cout);

 // TODO

endmodule

---

import bcd::\*;

module bcd\_adder\_3d\_test;

 bcd\_t [2:0] op1, op2, sum;

 logic cin, cout;

 bcd\_adder\_3d uut(.\*);

 task test\_vector (

 input bcd\_t [2:0] op1\_test, op2\_test,

 input cin\_test,

 input cout\_expected,

 input bcd\_t [2:0] sum\_expected);

 op1 = op1\_test;

 op2 = op2\_test;

 cin = cin\_test;

 #10 assert (sum\_expected == sum);

 assert (cout\_expected == cout);

 endtask

 initial begin

 test\_vector(

 {4'd0,4'd0,4'd2}, // op1\_test

 {4'd0,4'd0,4'd2}, // op2\_test

 0, // cin\_test

 0, // cout\_expected

 {4'd0,4'd0,4'd4}); // sum\_expected

 test\_vector({'0,'0,'0}, {'0,'0,'0}, 0, 0, {'0,'0,'0});

 test\_vector({'0,'0,'0}, {'0,'0,'0}, 1, 0, {'0,'0,'1});

 test\_vector({4'd9,4'd9,4'd9}, {'0,'0,'0}, 1, 1, {'0,'0,'0});

 test\_vector({4'd5,4'd6,4'd7}, {4'd2,4'd8,4'd5}, 0, 0, {4'd8,4'd5,4'd2});

 test\_vector({4'd5,4'd6,4'd7}, {4'd2,4'd8,4'd5}, 1, 0, {4'd8,4'd5,4'd3});

 test\_vector({4'd9,4'd9,4'd9}, {4'd9,4'd9,4'd9}, 1, 1, {4'd9,4'd9,4'd9});

 $finish;

 end

 initial $monitor("%0d%0d%0d + %0d%0d%0d + %b = %0b%0d%0d%0d",op1[2], op1[1], op1[0], op2[2], op2[1], op2[0], cin, cout, sum[2],sum[1],sum[0]);

endmodule