

LAB IVERILOG 1

Getting started

Version: Oct 26th 2016

I. Install

- a. Windows
 - Cài với quyền admin.
 - Chọn tất cả option.
- b. Linux

II. First counter

File Verilog có extension .v và có thể edit bằng Notepad, Notepad++, hoặc Cygwin.

```
//-----  
// This is my second Verilog Design  
// Design Name : first_counter  
// File Name : first_counter.v  
// Function : This is a 4 bit up-counter with  
// Synchronous active high reset and  
// with active high enable signal  
//-----  
module first_counter (  
    clock , // Clock input of the design  
    reset , // active high, synchronous Reset input  
    enable , // Active high enable signal  
    counter_out // 4 bit vector output);  
  
// End of port list  
//-----Input Ports-----  
input clock ;  
input reset ;  
input enable ;  
//-----Output Ports-----  
output [3:0] counter_out ;  
  
wire clock ;  
wire reset ;  
wire enable ;  
reg [3:0] counter_out ;  
  
//-----Code Starts Here-----  
  
always @ (posedge clock)  
begin : COUNTER // Block Name  
    // At every rising edge of clock we check if reset  
    is active  
    // If active, we load the counter output with  
    4'b0000  
    if (reset == 1'b1) begin  
        counter_out <= #1 4'b0000;  
    end  
    // If enable is active, then we increment the
```

// là comment, không được biên dịch

Định nghĩa một “module” với tên first_counter
Module có 4 tham số là clock, reset, enable,
counter_out

Định nghĩa các tham số nào là “input”
Các tham số này là đường dây (1 bit tín hiệu)

Định nghĩa tham số nào là “output”, tham số này là
bus 4 bits.
wire là “đường dây”

reg là thanh ghi, tức là biến đầu ra này giữ nguyên
giá trị trước đó nếu như nó không được thay đổi.

Khi tín hiệu clock thay đổi từ 0 lên 1 (positive edge)

Nếu reset = 1
Thì counter_out = 0000

<pre> counter else if (enable == 1'b1) begin counter_out <= #1 counter_out + 1; end end // End of Block COUNTER endmodule // End of Module counter </pre>	<p>Nếu không (reset = 0) thì Nếu enable = 1 thì counter tăng 1.</p> <ul style="list-style-type: none"> TH enable = 0 không được code, khi đó counter_out giữ nguyên giá trị vì nó được định nghĩa là "reg".
---	--

Dịch đoạn code trên bằng lệnh:

C:\WORK>iverilog -o test1 first_counter_tb.v

- File first_counter.v đặt trong thư mục WORK.
- -o test1 là sinh ra obj test1

Nếu biên dịch thành công, trong thư mục WORK sẽ có thêm obj test1

III. Test Bench

File first_counter.v đã hoàn tất nhưng để kiểm tra, chúng ta cần sinh ra các testcase, là bộ các giá trị đầu vào, để quan sát đầu ra xem module đã thực hiện đúng chức năng chưa.

<pre> `include "first_counter.v" module first_counter_tb(); // Declare inputs as regs and outputs as wires reg clock, reset, enable; wire [3:0] counter_out; // Initialize all variables initial begin \$dumpfile ("test1.vcd"); \$dumpvars(1, clock, reset, enable, counter_out); \$display ("time\t clk reset enable counter"); \$monitor ("%g\t %b %b %b %b", \$time, clock, reset, enable, counter_out); clock = 1; // initial value of clock reset = 0; // initial value of reset enable = 0; // initial value of enable #5 reset = 1; // Assert the reset #10 reset = 0; // De-assert the reset #10 enable = 1; // Assert enable #100 enable = 0; // De-assert enable #50 \$finish; // Terminate simulation end // Clock generator always begin #5 clock = ~clock; // Toggle clock every 5 ticks end // Connect DUT to test bench </pre>	<p>Bao gồm file first_counter.v đã viết. Định nghĩa module</p> <p>Tạo ra file .vcd sử dụng trong GTKWave với các biến quan sát: clock, reset, enable, counter_out</p> <p>Hiện thị theo thời gian với 4 cột có các tiêu đề là clock, reset, enable, counter_out</p> <p>Giá trị khởi tạo ban đầu.</p> <p>Sau 5 giây, đặt tín hiệu reset = 1 Sau 10 giây, đặt tín hiệu reset = 0 Sau 10 giây, đặt tín hiệu enable = 1 Sau 100 giây, đặt tín hiệu enable = 0 Sau 50 giây, kết thúc quan sát.</p> <p>Luôn luôn thay đổi tín hiệu clock sau mỗi 5 giây, thay đổi nghịch đảo</p> <p>Kết nối đến test bench</p>
--	---

```

first_counter U_counter (
clock,
reset,
enable,
counter_out
);

endmodule

```

Dịch đoạn code trên bằng lệnh:

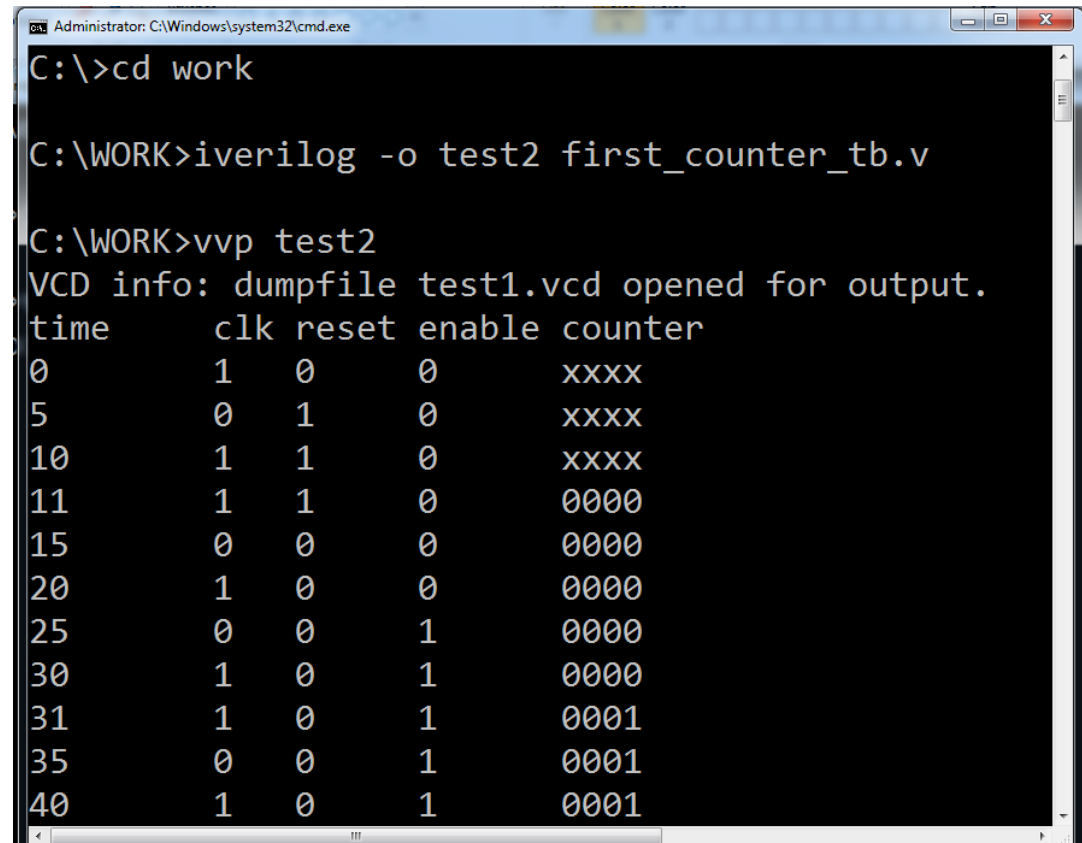
C:\WORK>iverilog -o test2 first_counter_tb.v

Do đã include file first_counter.v nên không cần biên dịch file first_counter.v

Chạy obj sinh ra bằng lệnh

C:\WORK>vvp test2

Console sẽ hiển thị bảng chân trị của mạch.



```

Administrator: C:\Windows\system32\cmd.exe
C:\>cd work

C:\WORK>iverilog -o test2 first_counter_tb.v

C:\WORK>vvp test2
VCD info: dumpfile test1.vcd opened for output.
time      clk reset enable counter
0          1    0     0      xxxx
5          0    1     0      xxxx
10         1    1     0      xxxx
11         1    1     0      0000
15         0    0     0      0000
20         1    0     0      0000
25         0    0     1      0000
30         1    0     1      0000
31         1    0     1      0001
35         0    0     1      0001
40         1    0     1      0001

```

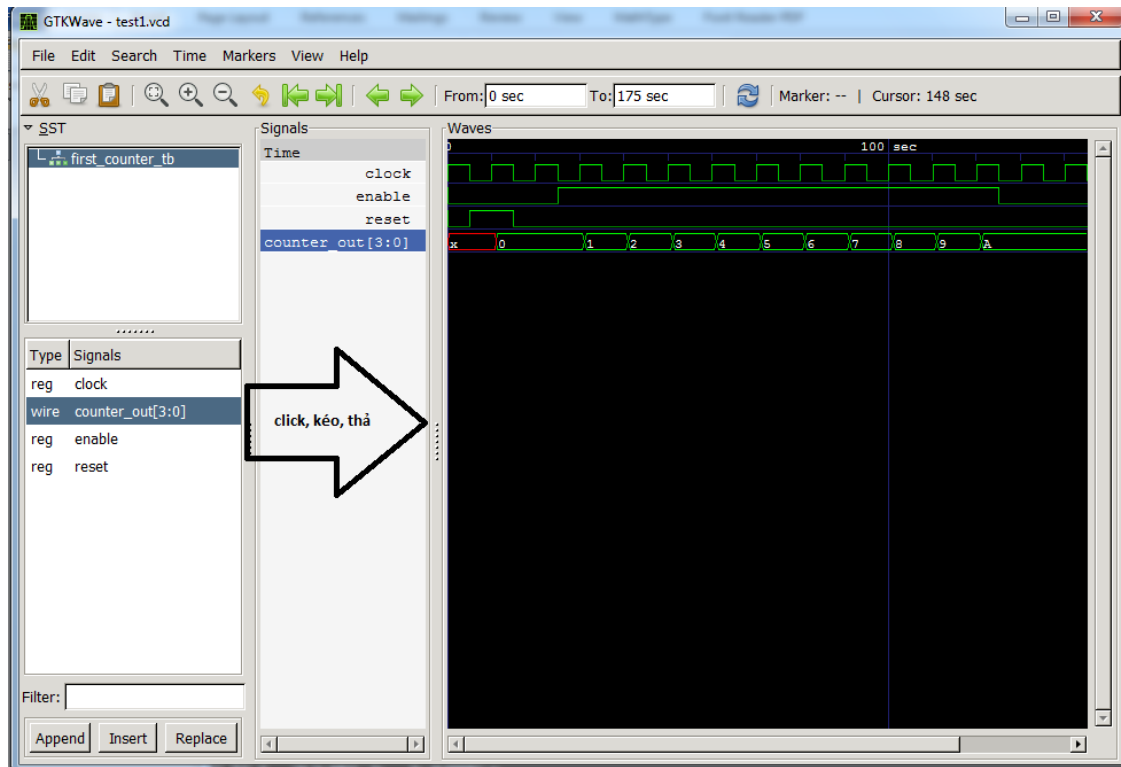
IV. GTKWave

Gõ lệnh

C:\WORK>gtkwave test1.vcd

Click lên module, click vào biến cần quan sát và kéo thả vào màn hình thời gian.

Phóng to thu nhỏ để quan sát.



V. Practice

1. Dựa vào file first_counter.v, chỉnh sửa lại để có counter 3 bits.
2. Dựa vào file first_counter.v, chỉnh sửa lại để có counter đếm số lẻ: 1, 3, 5, 7, ... sau mỗi tín hiệu positive edge của clock.
3. Viết module "NOT" có đầu ra output = ~input khi mà enable = 1 và positive edge clock. Viết module test bench cho module "NOT".
4. Viết module "BU_2" có đầu vào là 1 số 8 bits và đầu ra là 1 số bù 2 của đầu vào khi mà enable = 1 và positive edge clock. Viết module test bench cho module "NOT".

VI. Reflexive question

- a. Trong test bench của first_counter, tại sao khi gõ lệnh vvp test2 thì console liệt kê các dòng thời gian tại giây 11, 31, 41, ... bên cạnh các mốc thời gian 5 giây?
- b. Một test bench tốt được hình thành bằng cách nào?