แนะนำ VHDL **อย่างง่ายด้วย** Xilinx ISE Design Suite

เครื่องมือ

* Xilinx ISE Design Suite 13.2, ดาวน์โหลดจาก: http://www.xilinx.com/support/download/index.htm.

* เอกสารแนะนำ Xilinx ISE อย่างเป็นทางการดาวน์โหลดได้จาก:
http://www.xilinx.com/support/documentation/dt_ise13-2.htm.

$1. เริ่มต้น!

1. เปิด ISE Project Navigator: Start > All Programs > Xilinx ISE Design Suite > ISE Design Tools > Project Navigator.
Project Navigator แสดงดังรูปที่ 1.

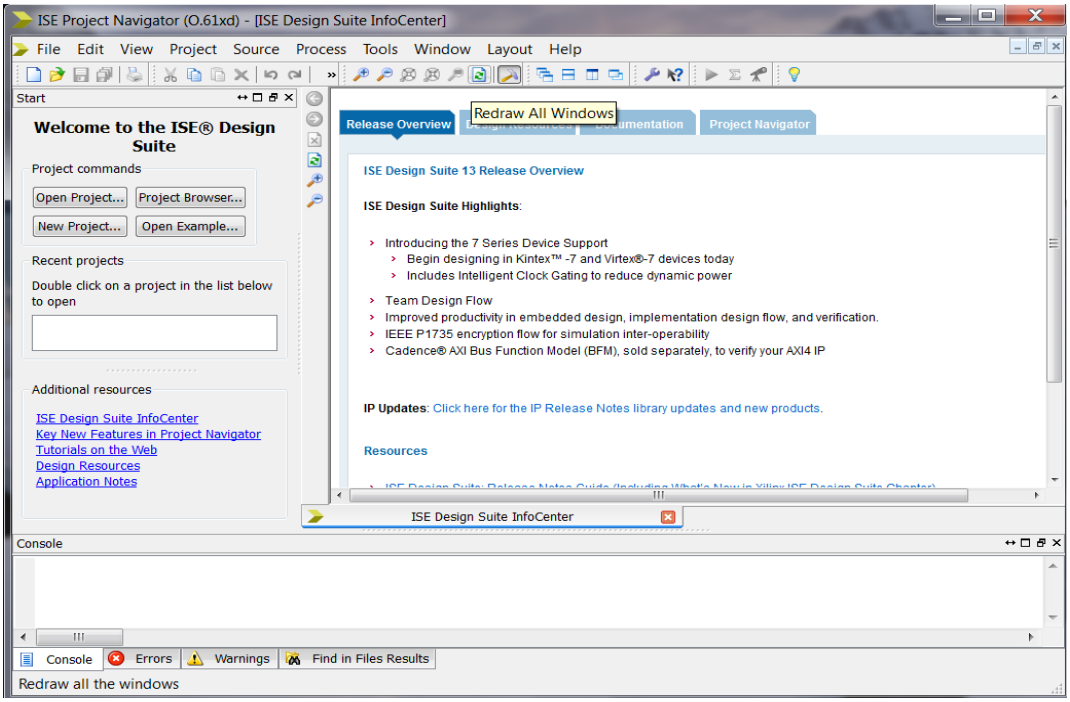


Figure 1: ISE Project Navigator

$2. สร้างโปรเจคใหม่.

a) **`File > New Project`**.
หน้าต่าง New Project Wizard จะเปิดขึ้นมา.

b) ในช่อง Location, เลือกที่ ที่จะเก็บไฟล์ต่างๆของโปรเจคนี้ลงไป. ตัวอย่าง เช่น ถ้าต้องการเก็บไฟล์งานไว้ใน `c:\xilinx_tutorial` ก็ให้ใส่ `c:\xilinx_tutorial`.

c) ในช่อง Name, ใส่ชื่อโปรเจคลงไป เช่น ในรูปที่ 2 ตั้งชื่อโปรเจคว่า **`courage01`**.

d) ตรวจดูว่าช่อง Top-Level Source Type เลือก **`HDL`** อยู่ แล้วคลิก **`Next`**.

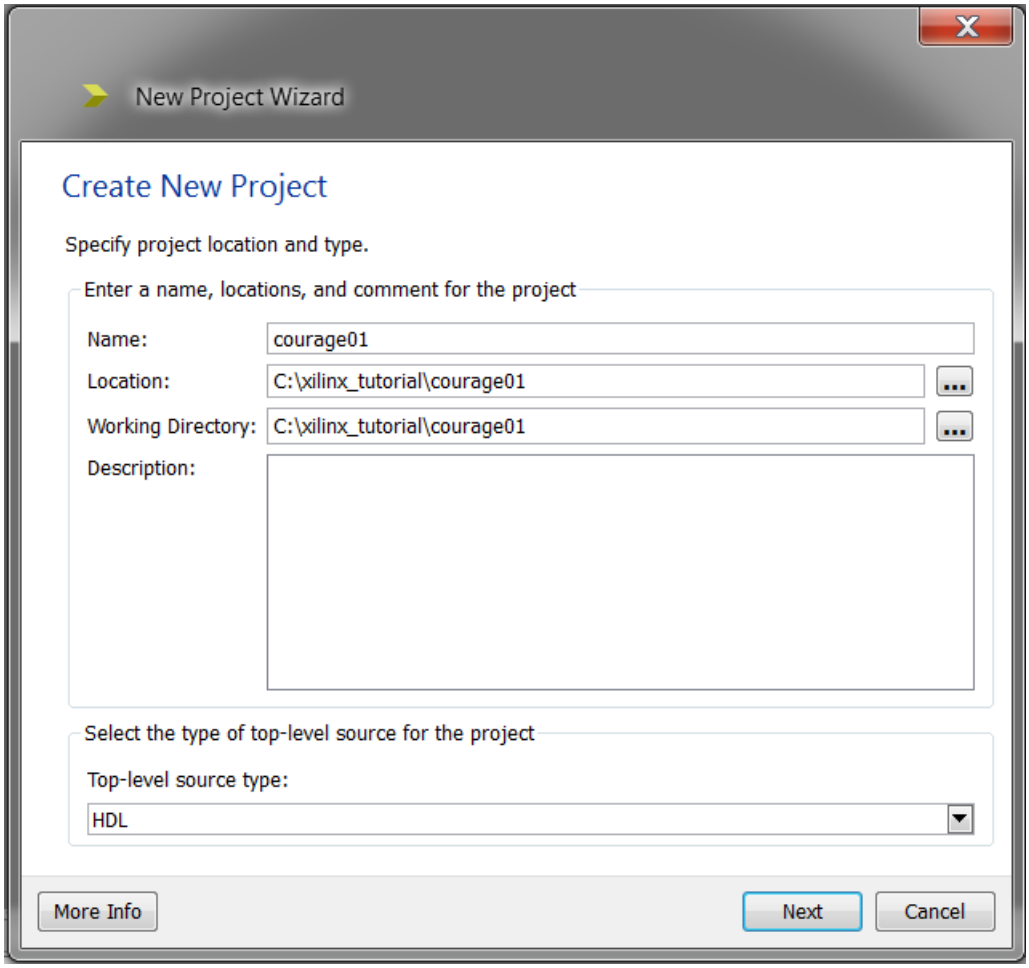


Figure 2: New Project Wizard

e) ในหน้าต่าง Project Settings, เลือกค่าต่อไปนี้:

Product Category: **`All`**
Family: **`Spartan3A and Spartan3AN`**
Device: **`XC3S700A`**
Package: **`FG484`**

Speed: **-4**
Synthesis Tool: **XST (VHDL/Verilog)**
Simulator: **ISim (VHDL/Verilog)**

f) คลิก **Next**, แล้ว **Finish** เพื่อสร้างโปรเจคใหม่.

$3. เพิ่ม ไฟล์ VHDL เข้าไปในโปรเจค.

a) **Project > New Source ….**
หน้าต่าง New Source Wizard จะเปิดขึ้นมา.

b) เลือก **VHDL Module** สำหรับ source type.

c) ใส่ **Logics00** เป็นชื่อ file name.

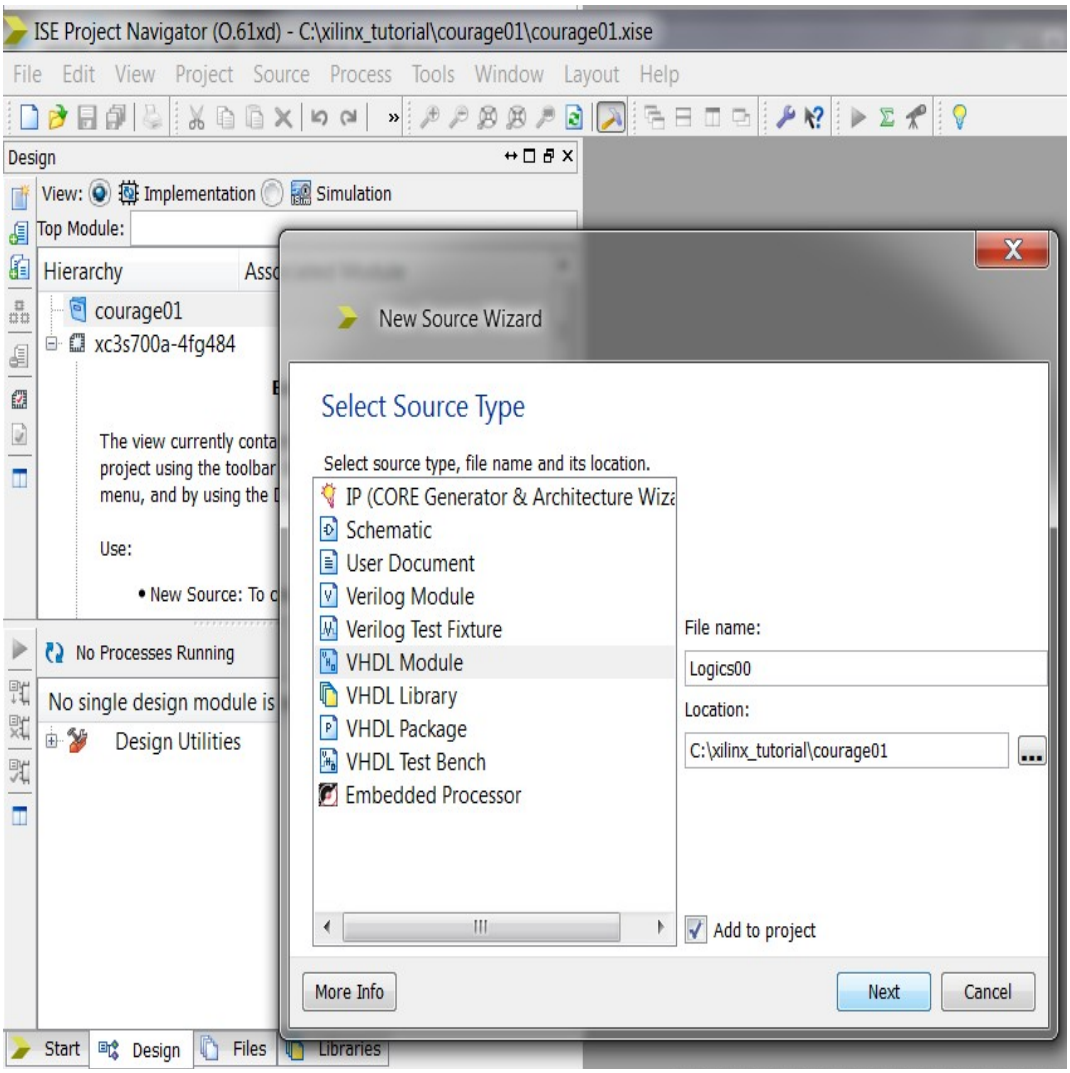


Figure 3: New Source Wizard

d) ใส่ **x0**, **x1**, และ **x2** สำหรับ **in** และ **y0** และ **y1** สำหรับ **out** ดังแสดงในรูปที่ 4 และคลิก **Next**.
ไฟล์ vhdl (Logics00.vhd) จะถูกสร้างขึ้นมาและเพิ่มเข้าไปในโปรเจค.

e) พิมพ์:

```
y0 <= NOT (x0 AND x1);
y1 <= NOT (x1 OR x2);
```

ระหว่าง **begin** และ **end** ของ **architecture** เพื่อบรรยายการเชื่อมของสัญญาณข้อมูลของ Logics00, ดังแสดงในรูปที่ 5.

f) **File** > **Save**.

g) ตรวจสอบดูว่า **`Project`** > **`Manual Compile Order`** ไม่ได้ถูกเลือก.

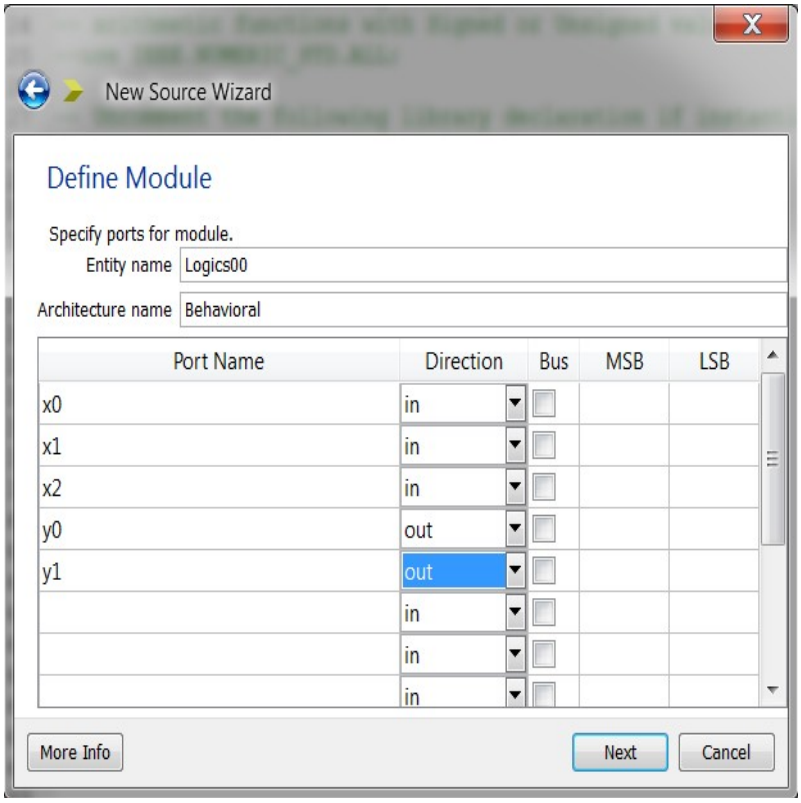


Figure 4: New Source Wizard: Define Module

```
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity Logics00 is
33      Port ( x0 : in  STD_LOGIC;
34             x1 : in  STD_LOGIC;
35             x2 : in  STD_LOGIC;
36             y0 : out  STD_LOGIC;
37             y1 : out  STD_LOGIC);
38  end Logics00;
39
40  architecture Behavioral of Logics00 is
41
42  begin
43     y0 <= NOT (x0 AND x1);
44     y1 <= NOT (x1 OR x2);
45
46  end Behavioral;
```

Figure 5: VHDL code of Logics00

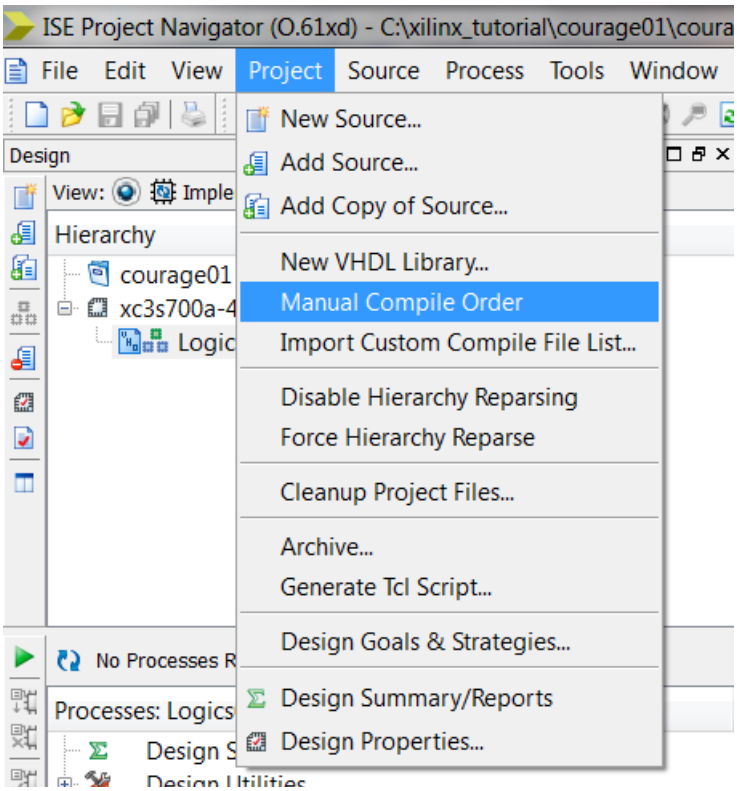


Figure 6: Verify that Manual Compile Order is unchecked.

$4. Synthesize your design.

a) ใน Hierarchy pane ของ Project Navigator Design Panel, เลือกโมดูลที่เราสนใจ: **Logics00**.

b) ใน Processes pane, ให้ดับเบิ้ลคลิก **Synthesize**. ดูรูปที่ 7 (สังเกตุ เลือก Implementation View)

c) เพื่อจะดู RTL schematic จาก HDL code, ขยาย (คลิกเครื่องหมายบวกหน้า) **Synthesize** ใน Processes pane และ ดับเบิ้ลคลิก **View RTL Schematic**.

d) หน้าต่าง Set RTL/Tech Viewer Setup Mode จะเปิดขึ้นมา. เลือก **Start with the Explorer Wizard** และ คลิก **OK**.

e) ในหน้า Create RTL Schematic, เลือก **Logics00** และ **Add** เพื่อเลือกโมดูลที่จะดู.

f) คลิก **Create Schematic**.

หน้าต่าง RTL Schematic viewer แสดง schematic จาก VHDL code. เราสามารถเลือกชิ้นส่วนต่างๆและดูส่วน

ประกอบภายในของมันได้ โดย คลิกขวา และ เลือก **`Show Block Contents`**. ดูรูป 9 และ 10.

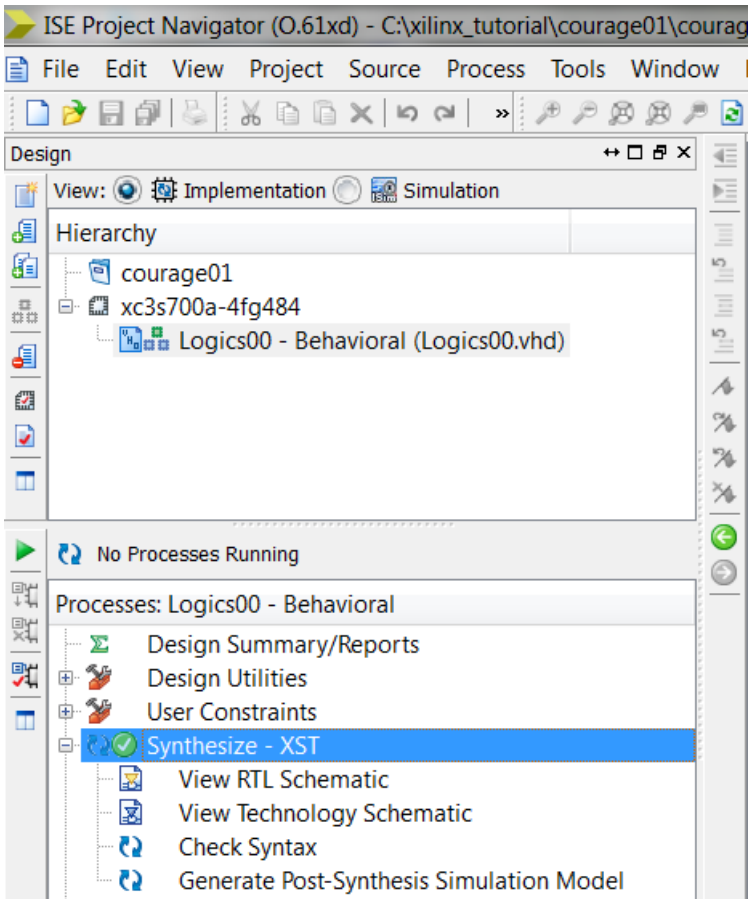


Figure 7: Hierarchy and Processes panes

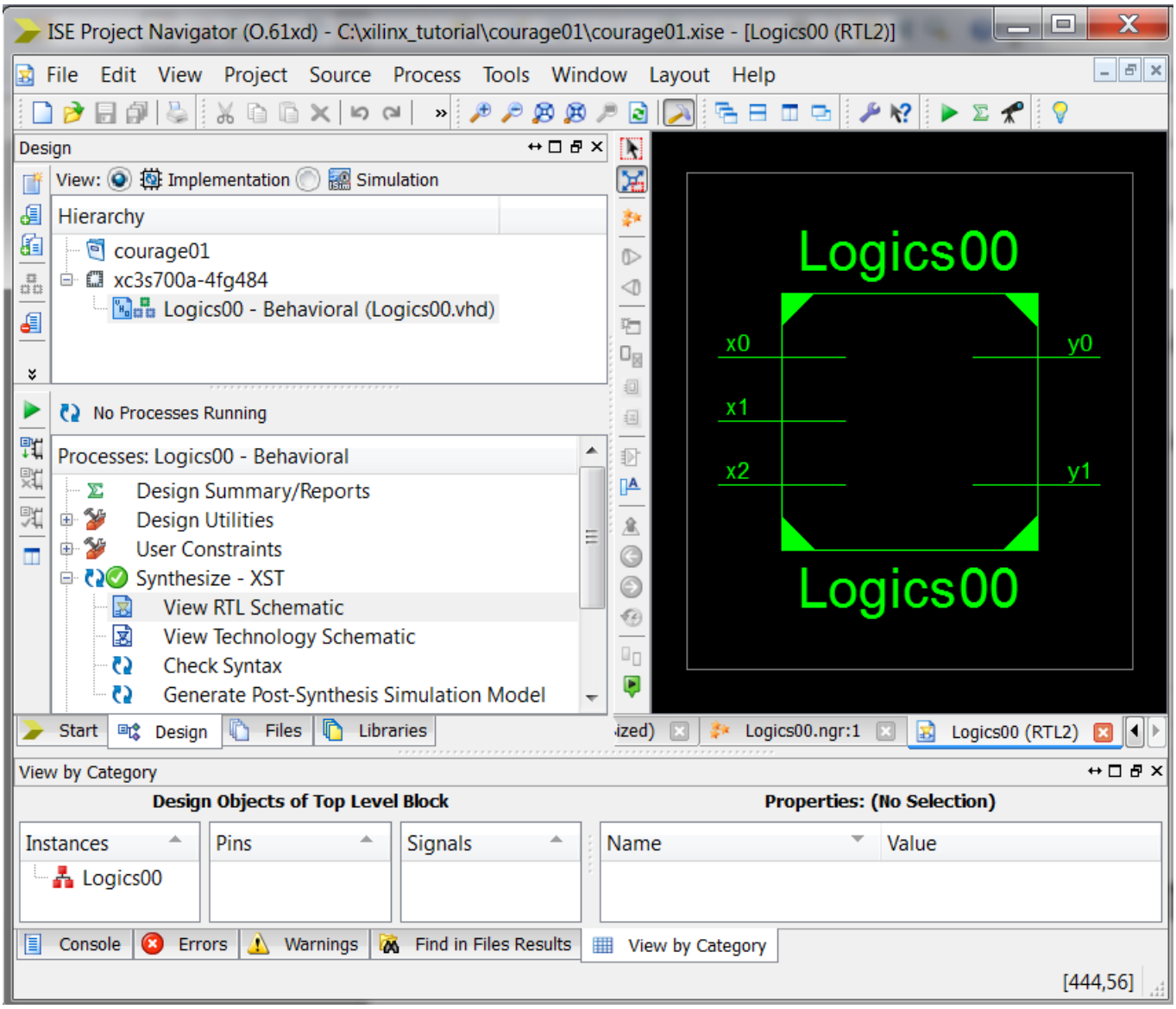


Figure 8: RTL Schematic

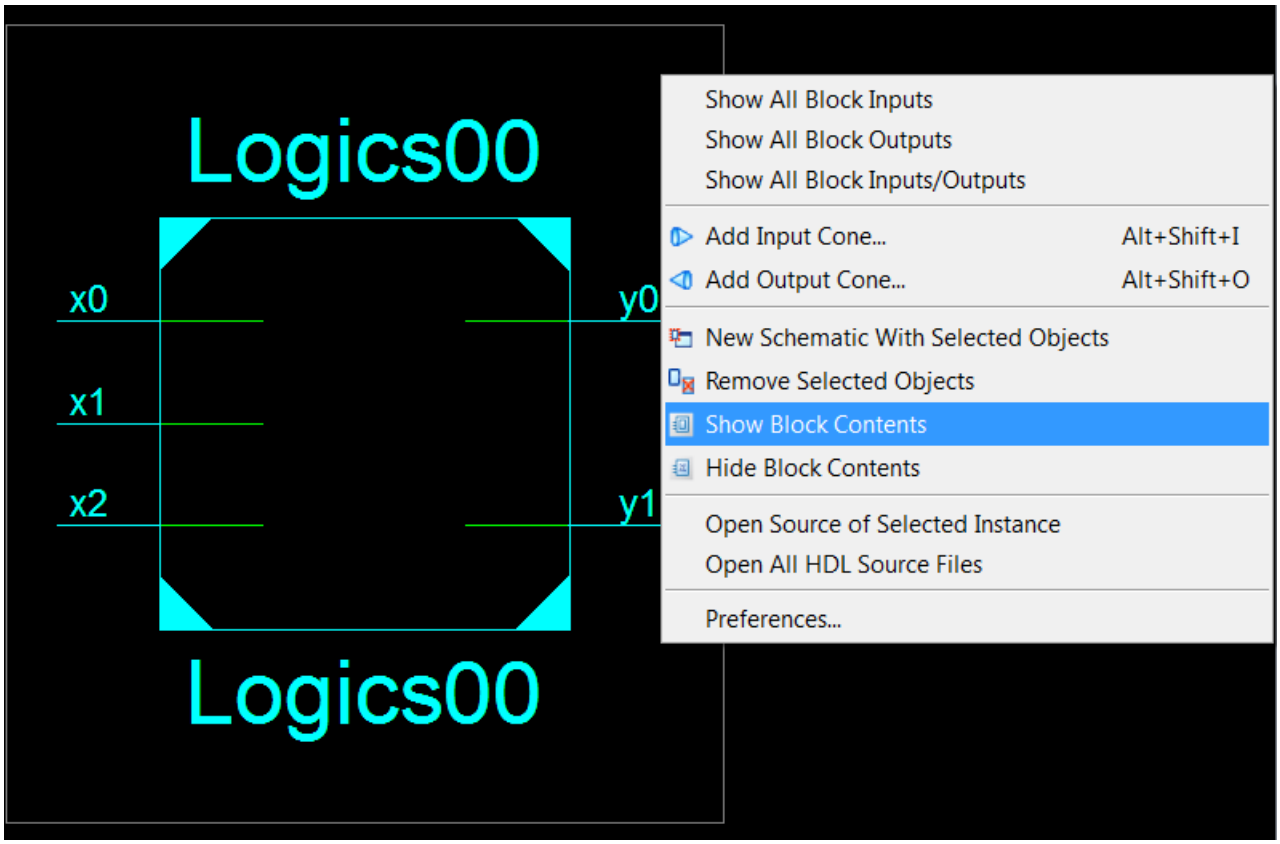


Figure 9: You can see contents inside a block in RTL Schematic viewer.

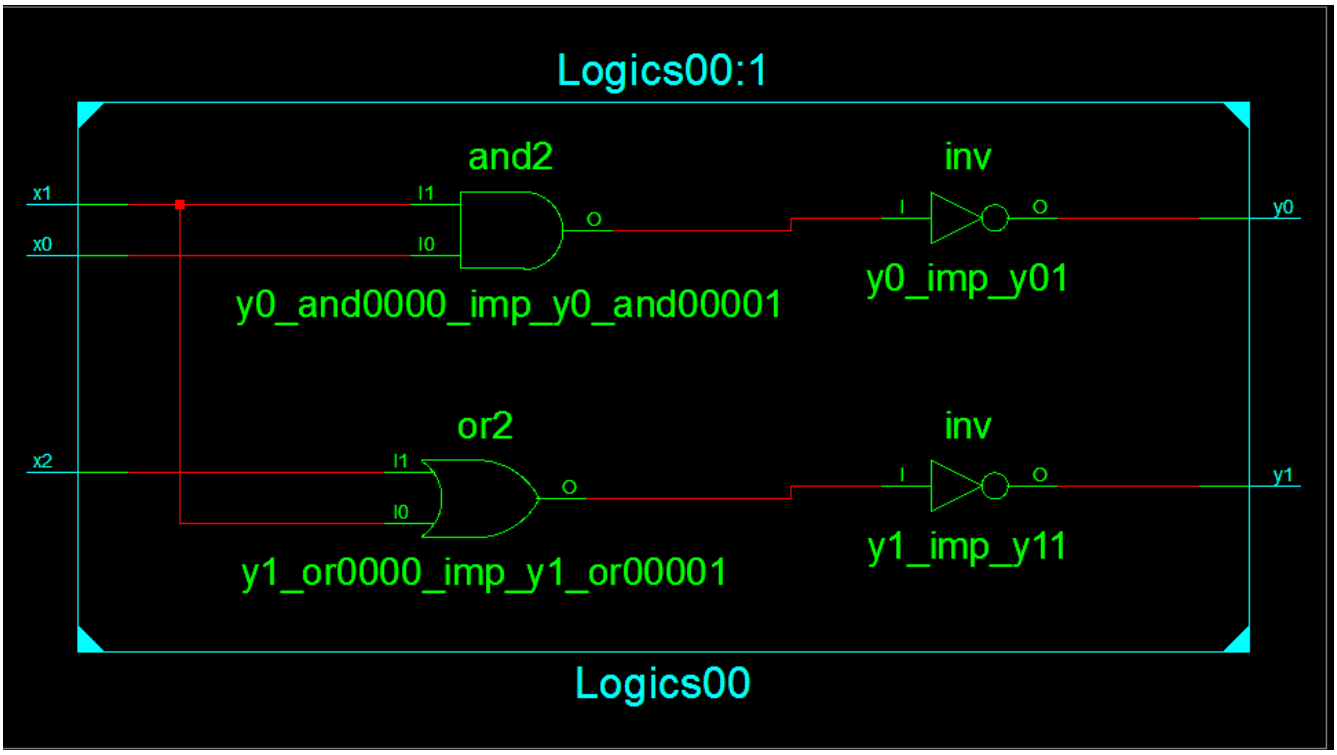
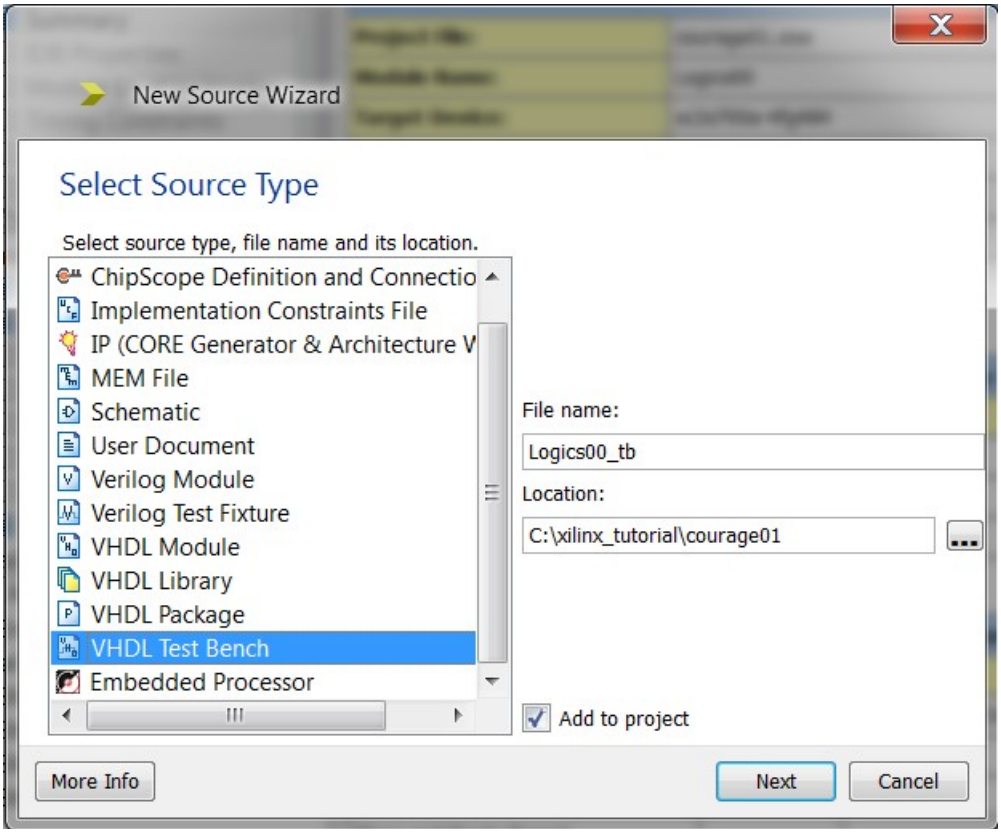


Figure 10: Logic gates inside Logics00 block.

$5. Simulate your first design.

a) **`Project > New Source …`**
หน้าต่าง New Source Wizard เปิดขึ้นมา.

b) เลือก **`VHDL Test Bench`** และ พิมพ์: **`Logics00_tb`** เพื่อเป็นชื่อไฟล์. คลิก **`Next`**, **`Next`**, และ **`Finish`**.



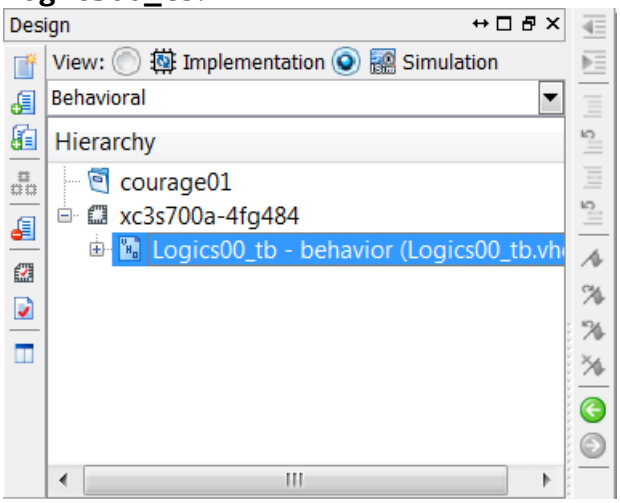
c) พิมพ์โค้ดต่อไปนี้ลงในไฟล์ **Logics00_tb.vhd**.

```
--------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:   16:28:19 08/16/2011
-- Design Name:
-- Module Name:   C:/xilinx_tutorial/courage01/Logics00_tb.vhd
-- Project Name:  courage01
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: Logics00
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic and
-- std_logic_vector for the ports of the unit under test.  Xilinx recommends
-- that these types always be used for the top-level I/O of a design in order
-- to guarantee that the testbench will bind correctly to the post-implementation
-- simulation model.
--------------------------------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
```

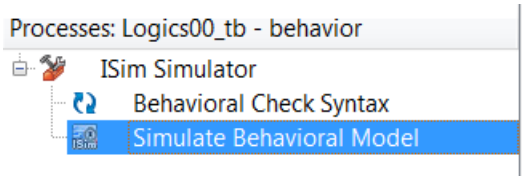
```
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY Logics00_tb IS
END Logics00_tb;

ARCHITECTURE behavior OF Logics00_tb IS

 -- Component Declaration for the Unit Under Test (UUT)

 COMPONENT Logics00
 PORT(
  x0 : IN  std_logic;
  x1 : IN  std_logic;
  x2 : IN  std_logic;
  y0 : OUT  std_logic;
  y1 : OUT  std_logic
  );
 END COMPONENT;


 --Inputs
 signal x0 : std_logic := '1';
 signal x1 : std_logic := '0';
 signal x2 : std_logic := '0';

       --Outputs
 signal y0 : std_logic;
 signal y1 : std_logic;
 -- No clocks detected in port list. Replace <clock> below with
 -- appropriate port name

 constant clk_period : time := 1000 ns;

BEGIN

       -- Instantiate the Unit Under Test (UUT)
 uut: Logics00 PORT MAP (
   x0 => x0,
   x1 => x1,
   x2 => x2,
   y0 => y0,
   y1 => y1
  );

 -- Clock process definitions
 clk_process :process
 begin
                clk_loop : loop
                        wait for clk_period/2;
                        x0 <= '0';
                        wait for clk_period/2;
                        x0 <= '1';
                end loop clk_loop;
 end process;


 -- Stimulus process
 stim_proc: process
 begin
                x1_loop : loop
```

```
                        x1 <= '0';
                        wait for clk_period/4;
                        x1 <= '1';
                        wait for clk_period/4;
                end loop x1_loop;
end process;

        x2_proc: process
        begin
                x2 <= '0';
                wait for clk_period;
                x2 <= '1';
                wait for clk_period;
        end process;

END;
```

d) **Save**. ใน View panel, ดังแสดงในรูปข้างล่าง, เลือก **Simulation**. ใน Hierarchy pane, เลือก **Logics00_tb**.



e) ใน Processes pane, ขยาย **ISim Simulator** และ เลือก **Simulate Behavioral Model** และ คลิกขวา.

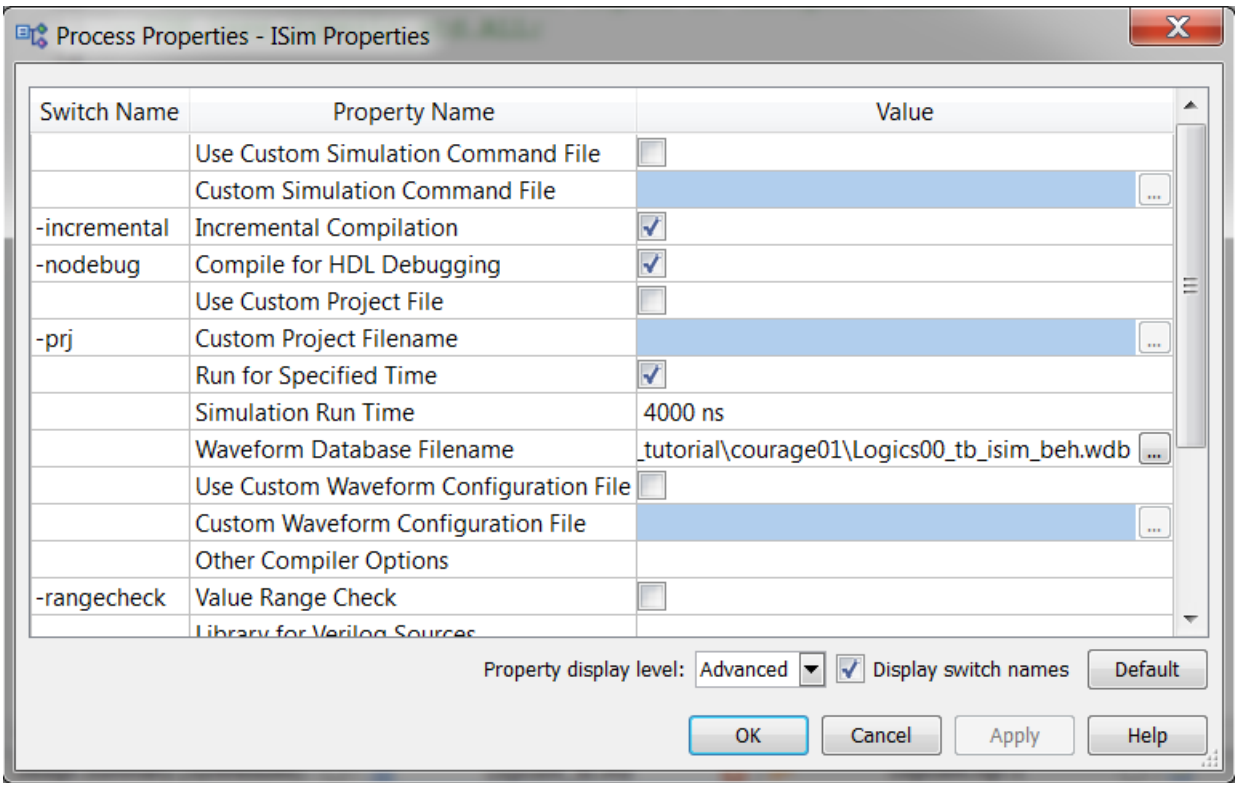


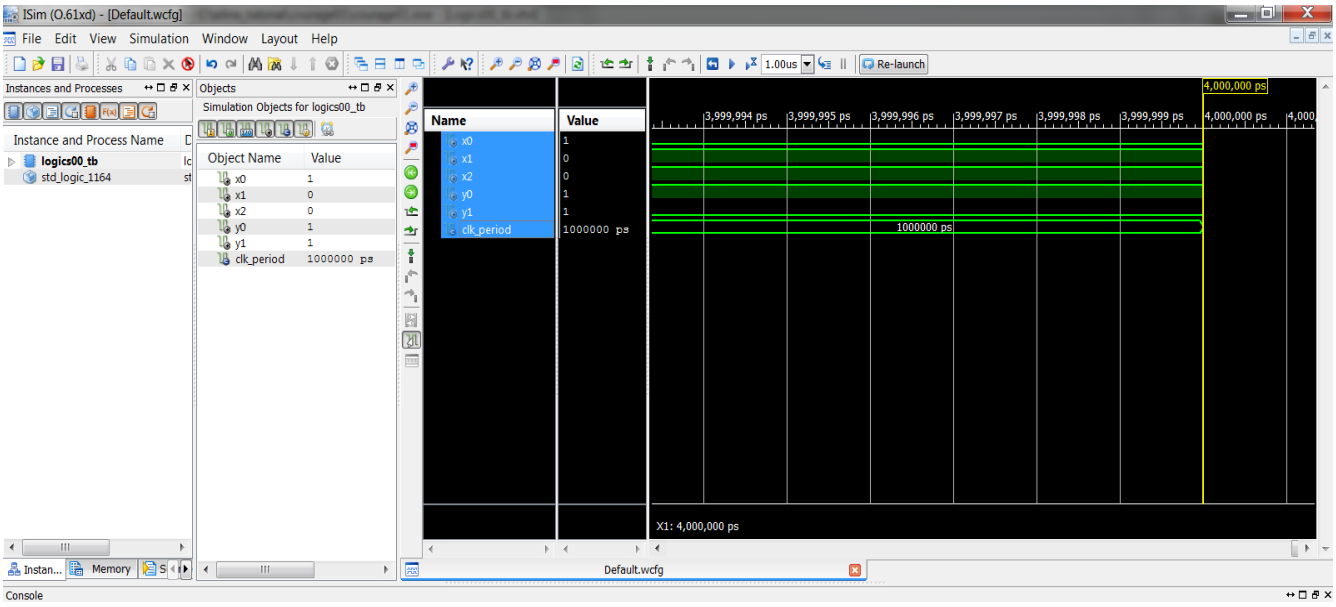
จาก pop-up menu, เลือก **Process Properties …**

เมื่อหน้าต่าง Process Properties เปิดขึ้นมา ให้เปลี่ยน **Simulation Run Time** เป็น **4000 ns**. คลิก **OK**.

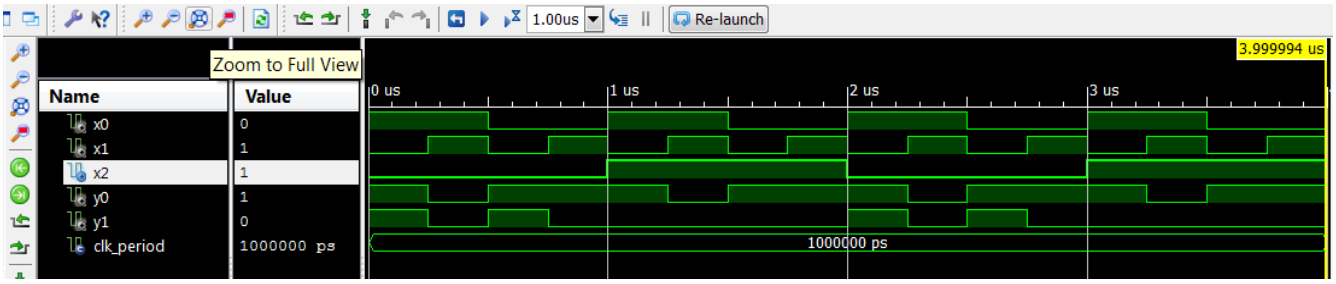
f) ดับเบิ้ลคลิก **Simulate Behavioral Model**.

หน้าต่าง ISim จะเปิดขึ้นมา แสดงค่าของสัญญาณต่างๆ ต่อเวลา (เช่น x0, x1, x2, y0, และ y1).





g) **`View > Zoom > To Full View`** เพื่อดูผลซิมมูเลชั่นในช่วงเวลาที่กำหนด.



แนะนำ VHDL รวดรัดเบื้องต้น ก็มีเท่านี้หละครับ.