--------------------------------------------------------------------------------

-- Company:

-- Engineer:

--

-- Create Date: 14:41:16 01/20/2015

-- Design Name:

-- Module Name: C:/Users/blhilbor/LAB1\_HILBORN\_KOPP/tb\_combinational\_switch\_logic.vhd

-- Project Name: LAB1\_HILBORN\_KOPP

-- Target Device:

-- Tool versions:

-- Description:

--

-- VHDL Test Bench Created by ISE for module: switch\_logic

--

-- 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 tb\_combinational\_switch\_logic IS

END tb\_combinational\_switch\_logic;

ARCHITECTURE behavior OF tb\_combinational\_switch\_logic IS

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

COMPONENT switch\_logic

PORT(

switches\_inputs : IN std\_logic\_vector(2 downto 0);

outputs : OUT std\_logic\_vector(2 downto 0)

);

END COMPONENT;

--Inputs

signal switches\_inputs : std\_logic\_vector(2 downto 0) := (others => '0');

--Outputs

signal outputs : std\_logic\_vector(2 downto 0);

-- No clocks detected in port list. Replace <clock> below with

-- appropriate port name

--constant <clock>\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: switch\_logic PORT MAP (

switches\_inputs => switches\_inputs,

outputs => outputs

);

-- Clock process definitions

--<clock>\_process :process

--begin

-- <clock> <= '0';

-- wait for <clock>\_period/2;

-- <clock> <= '1';

-- wait for <clock>\_period/2;

--end process;

A\_process: process

begin

switches\_inputs(0) <= '1';

--YOU DO THIS

wait for 20ns; --(pick some value, for example 20 ns)

switches\_inputs(0) <= '0';

wait for 20ns;

end process;

B\_process: process

begin

switches\_inputs(1) <= '1';

--YOU DO THIS

wait for 40ns;

switches\_inputs(1) <= '0';

wait for 40ns;

end process;

C\_process: process

begin

switches\_inputs(2) <= '1';

--YOU DO THIS

wait for 80ns;

switches\_inputs(2) <= '0';

wait for 80ns;

end process;

-- Stimulus process

stim\_proc: process

begin

-- -- hold reset state for 100 ns.

-- wait for 100 ns;

-- -- Set all inputs to 0

-- switches\_inputs(0) <= '0'; --A

-- switches\_inputs(1) <= '0'; --B

-- switches\_inputs(2) <= '0'; --C

-- wait for 50 ns;

-- -- Test an input combination

-- switches\_inputs(0) <= '1'; --A

-- switches\_inputs(1) <= '0'; --B

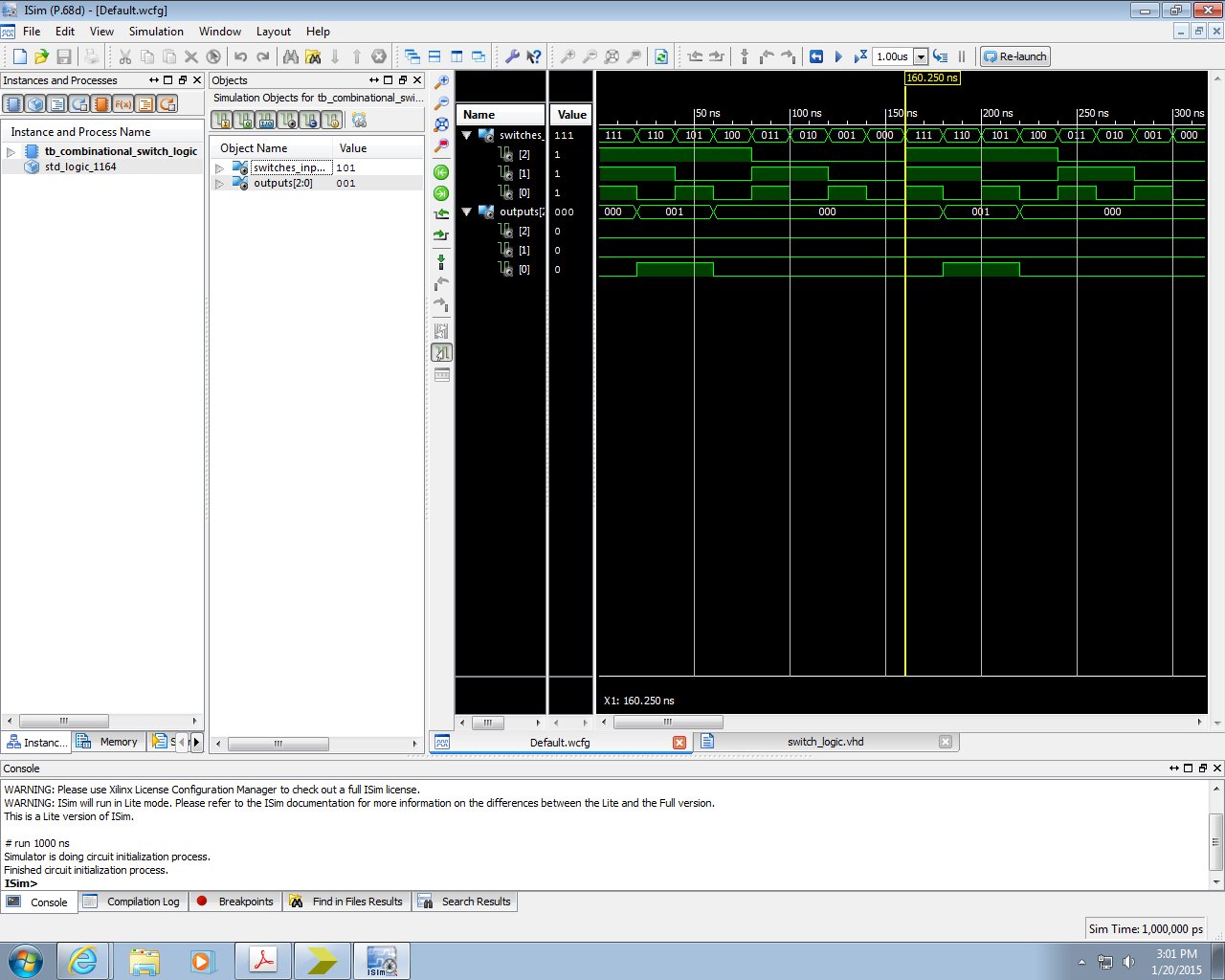
-- switches\_inputs(2) <= '0'; --C

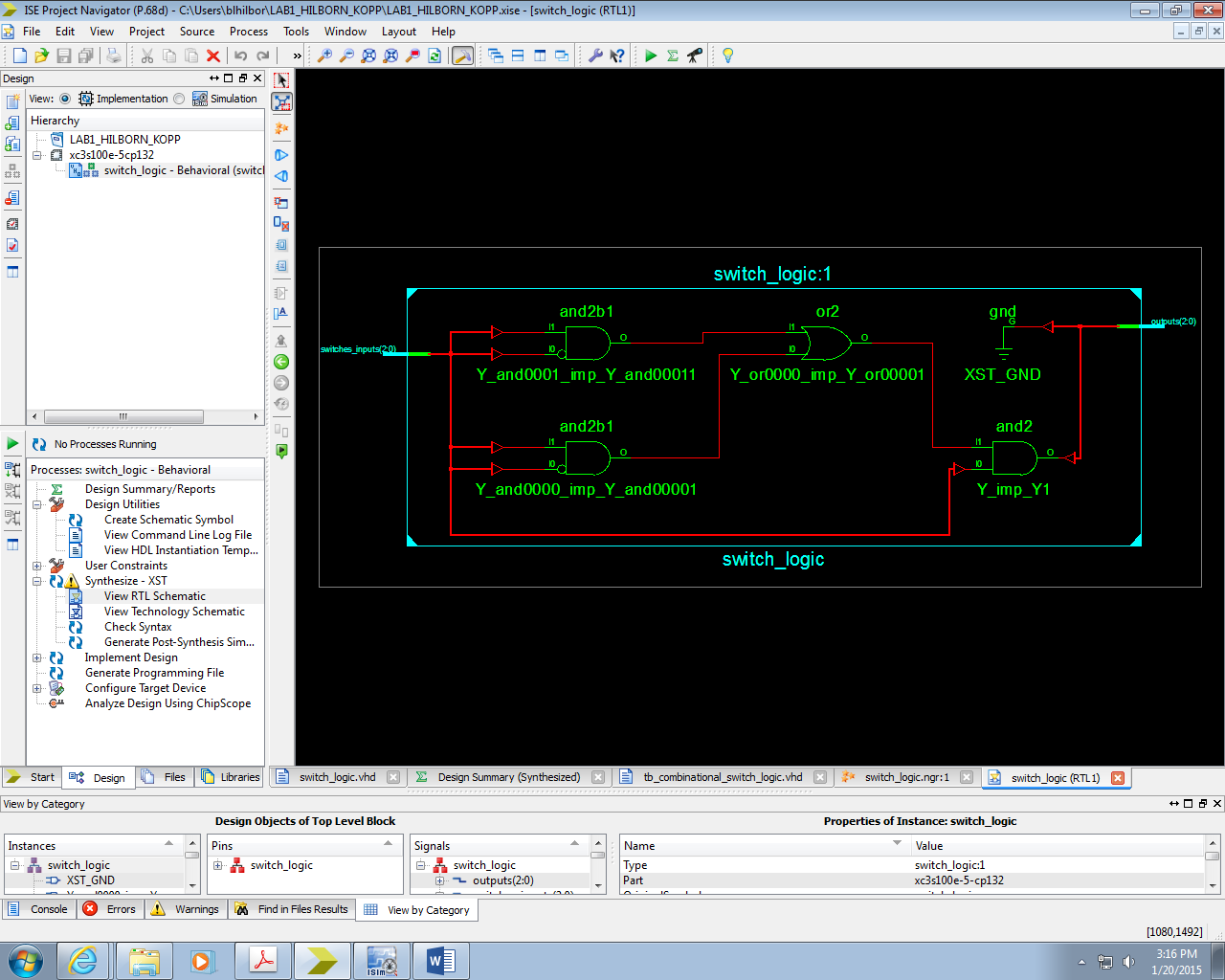
-- wait for 100 ns;

wait; -- Keeps it from restarting

end process;

END;





NET "outputs(0)" LOC="M5"; #LED0

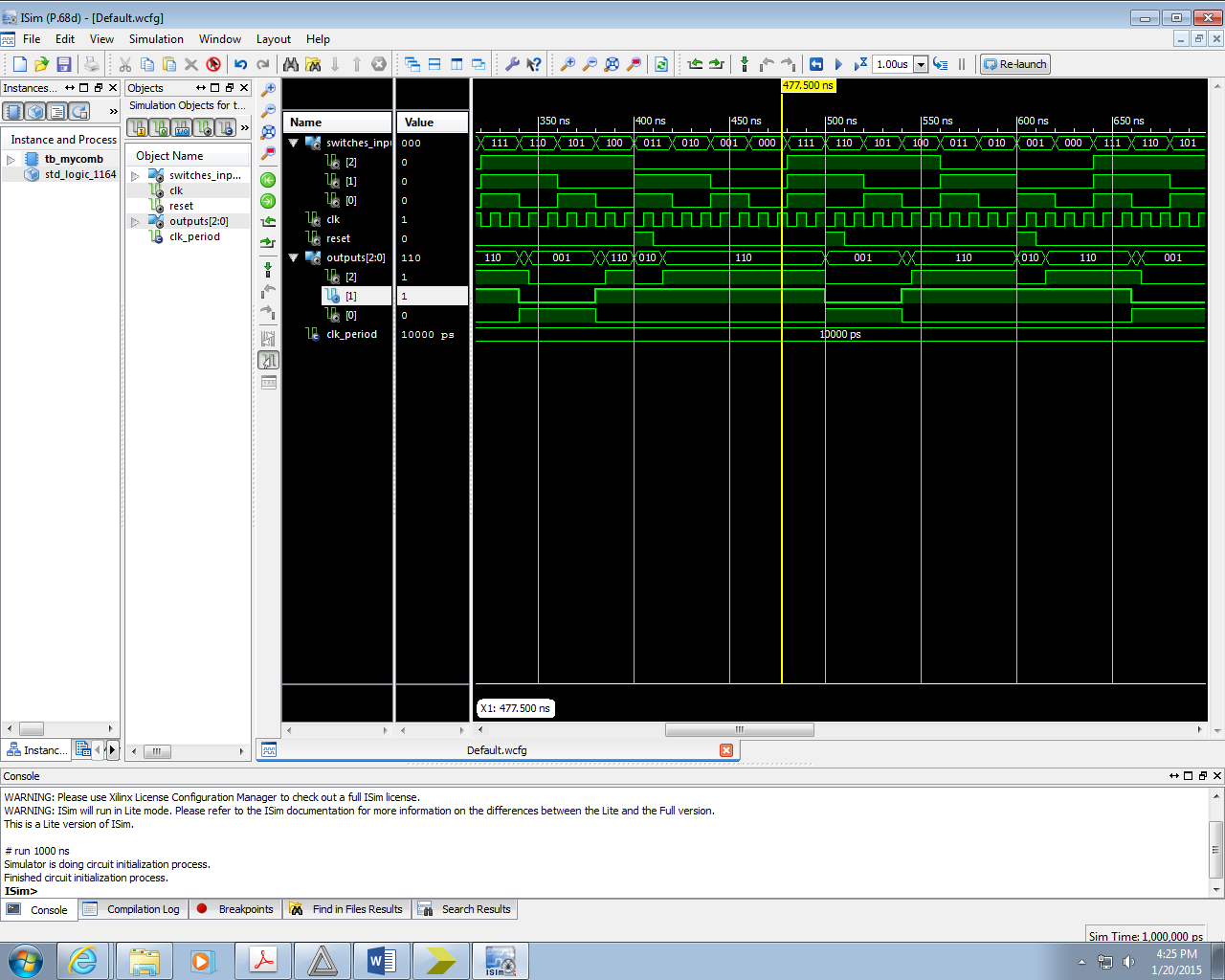
NET "outputs(1)" LOC="M11"; #LED1

NET "outputs(2)" LOC="P7"; #LED2

NET "switches\_inputs(0)" LOC="N3"; #switch0

NET "switches\_inputs(1)" LOC="E2"; #switch1

NET "switches\_inputs(2)" LOC="F3"; #switch2



architecture Behavioral of switch\_logic is

-- Internal signals:

signal Y: std\_logic;

signal X\_th, X\_mine: std\_logic;

signal A, B, C: std\_logic;

begin

logic\_of\_switches: process(clk, reset) begin

if (reset = '1') then

-- Reset X\_mine to a known state

X\_mine <= '0';

elsif (rising\_edge(clk)) then

-- Assign logic to X\_mine here

X\_mine <= (not C) or (B and A) or ((not A) and (not B));

end if;

end process;

-- Combinational logic!

Y <= C and (((not B) and A) or ((not A) and B));

X\_th <= not (C and (((not B) and A) or ((not A) and B)));

-- Assign the outputs. We only have one signal for now

outputs(0) <= Y;

outputs(1) <= X\_th; -- We will connect these later

outputs(2) <= X\_mine;

-- Grab the inputs from the slide switches on the FPGA board

A <= switches\_inputs(0);

B <= switches\_inputs(1);

C <= switches\_inputs(2);

end architecture Behavioral;

--------------------------------------------------------------------------------

-- Company:

-- Engineer:

--

-- Create Date: 16:06:58 01/20/2015

-- Design Name:

-- Module Name: C:/Users/blhilbor/LAB1\_HILBORN\_KOPP/tb\_mycomb.vhd

-- Project Name: LAB1\_HILBORN\_KOPP

-- Target Device:

-- Tool versions:

-- Description:

--

-- VHDL Test Bench Created by ISE for module: switch\_logic

--

-- 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 tb\_mycomb IS

END tb\_mycomb;

ARCHITECTURE behavior OF tb\_mycomb IS

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

COMPONENT switch\_logic

PORT(

switches\_inputs : IN std\_logic\_vector(2 downto 0);

outputs : OUT std\_logic\_vector(2 downto 0);

clk : IN std\_logic;

reset : IN std\_logic

);

END COMPONENT;

--Inputs

signal switches\_inputs : std\_logic\_vector(2 downto 0) := (others => '0');

signal clk : std\_logic := '0';

signal reset : std\_logic := '0';

--Outputs

signal outputs : std\_logic\_vector(2 downto 0);

-- Clock period definitions

constant clk\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: switch\_logic PORT MAP (

switches\_inputs => switches\_inputs,

outputs => outputs,

clk => clk,

reset => reset

);

-- Clock process definitions

clk\_process :process

begin

clk <= '0';

wait for clk\_period/2;

clk <= '1';

wait for clk\_period/2;

end process;

A\_process: process

begin

switches\_inputs(0) <= '1';

--YOU DO THIS

wait for 20ns; --(pick some value, for example 20 ns)

switches\_inputs(0) <= '0';

wait for 20ns;

end process;

B\_process: process

begin

switches\_inputs(1) <= '1';

--YOU DO THIS

wait for 40ns;

switches\_inputs(1) <= '0';

wait for 40ns;

end process;

C\_process: process

begin

switches\_inputs(2) <= '1';

--YOU DO THIS

wait for 80ns;

switches\_inputs(2) <= '0';

wait for 80ns;

end process;

R\_process: process

begin

reset <= '1';

--YOU DO THIS

wait for 10ns;

reset <= '0';

wait for 90ns;

end process;

-- Stimulus process

stim\_proc: process

begin

-- hold reset state for 100 ns.

wait for 100 ns;

wait for clk\_period\*10;

-- insert stimulus here

wait;

end process;

END;

