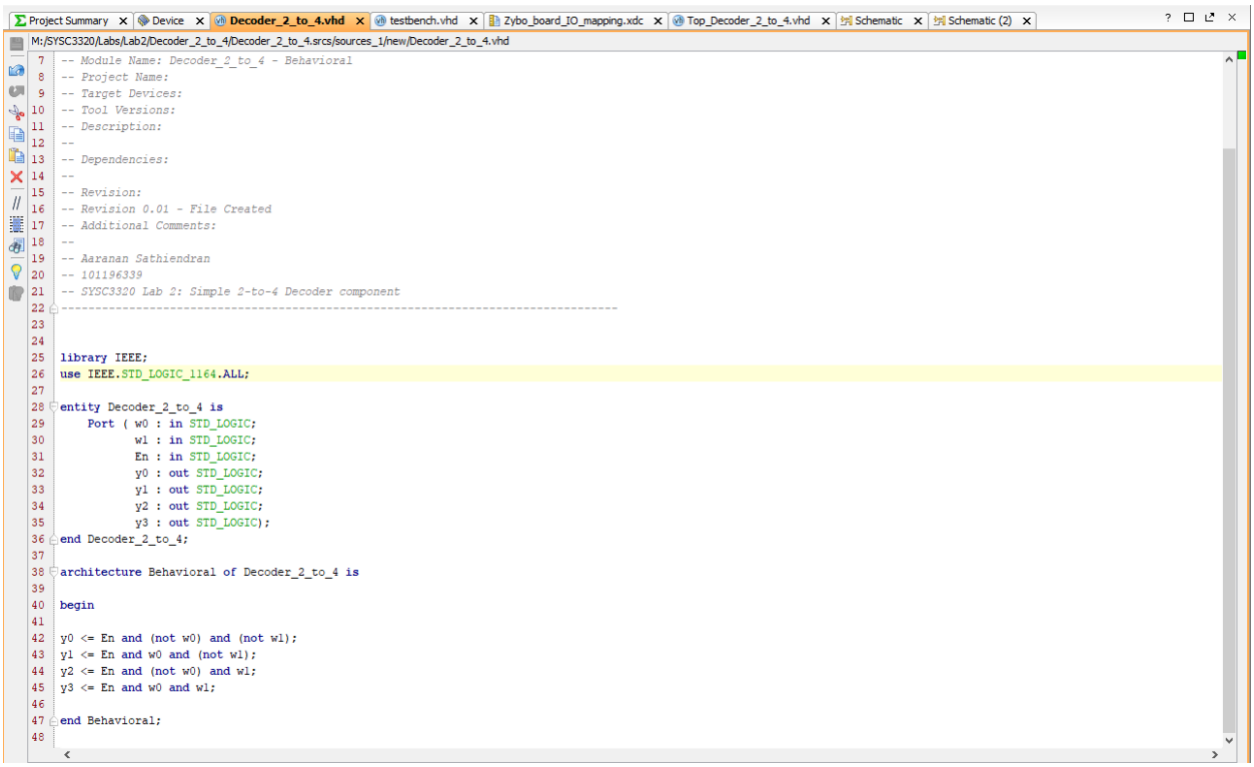
Department of
Systems and Computer Engineering

SYSC-3320: Laboratory 1
Decoder and Counter Design and Implementation using FPGA and VHDL on the Zynq-7000 SoC

Completed by: Aaranan Sathiendran (101196339)

Simple 2-to-4 Decoder Component:




```
-- Module Name: Decoder_2_to_4 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Aaranan Sathiendran
-- 101196339
-- SYSC3320 Lab 2: Simple 2-to-4 Decoder component
----------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder_2_to_4 is
    Port ( w0 : in STD_LOGIC;
           w1 : in STD_LOGIC;
           En : in STD_LOGIC;
           y0 : out STD_LOGIC;
           y1 : out STD_LOGIC;
           y2 : out STD_LOGIC;
           y3 : out STD_LOGIC);
end Decoder_2_to_4;

architecture Behavioral of Decoder_2_to_4 is

begin

y0 <= En and (not w0) and (not w1);
y1 <= En and w0 and (not w1);
y2 <= En and (not w0) and w1;
y3 <= En and w0 and w1;

end Behavioral;
```

Figure 1: VHDL code for 2-to-4 decoder

Project Summary | Device | Decoder_2_to_4.vhd | testbench.vhd | Zybo_board_IO_mapping.xdc | Top_Decoder_2_to_4.vhd | Schematic | Schematic (2)

M:/SYSC3320/Labs/Lab2/Decoder_2_to_4/Decoder_2_to_4.srcs/sim_1/new/testbench.vhd

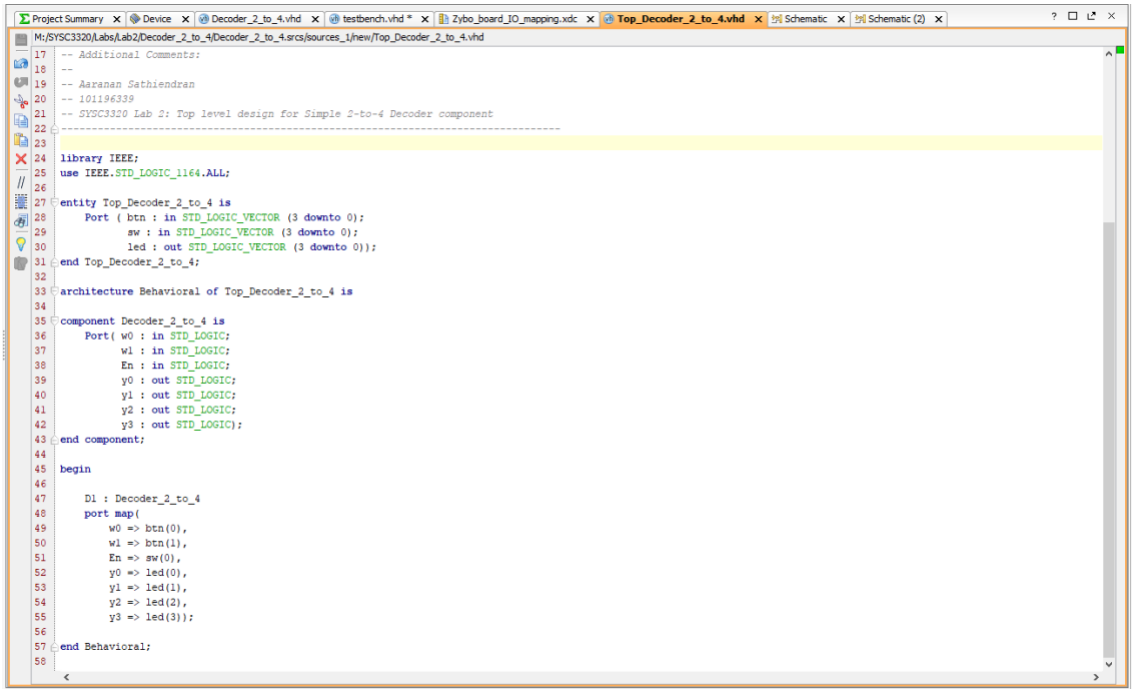
```vhdl
-- Aaranan Sathiendran
-- 101196339
-- SYSC3320 Lab 2: Testbench for Simple 2-to-4 Decoder component
--------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity testbench is
--  Port ( );
end testbench;

architecture Behavioral of testbench is
SIGNAL Inw0: STD_LOGIC:='0';
SIGNAL Inw1: STD_LOGIC:='0';
SIGNAL InEn: STD_LOGIC:='0';
SIGNAL Outy0: STD_LOGIC:='0';
SIGNAL Outy1: STD_LOGIC:='0';
SIGNAL Outy2: STD_LOGIC:='0';
SIGNAL Outy3: STD_LOGIC:='0';

component Decoder_2_to_4 is
    Port( w0 : in STD_LOGIC;
          w1 : in STD_LOGIC;
          En : in STD_LOGIC;
          y0 : out STD_LOGIC;
          y1 : out STD_LOGIC;
          y2 : out STD_LOGIC;
          y3 : out STD_LOGIC);
end component;

begin
    Dec_1 : Decoder_2_to_4
    port map(
        w0 => Inw0,
        w1 => Inw1,
        En => InEn,
        y0 => Outy0,
        y1 => Outy1,
        y2 => Outy2,
        y3 => Outy3);
```

Project Summary | Device | Decoder_2_to_4.vhd | testbench.vhd * | Zybo_board_IO_mapping.xdc | Top_Decoder_2_to_4.vhd | Schematic | Schematic (2)

M:/SYSC3320/Labs/Lab2/Decoder_2_to_4/Decoder_2_to_4.srcs/sim_1/new/testbench.vhd

```vhdl
        InEn <= '1';

        Inw0 <= '0';
        Inw1 <= '0';
        wait for 2 ns;

        Inw0 <= '0';
        Inw1 <= '1';
        wait for 2 ns;

        Inw0 <= '1';
        Inw1 <= '0';
        wait for 2 ns;

        Inw0 <= '1';
        Inw1 <= '1';
        wait for 2 ns;

        Inw0 <= '0';
        Inw1 <= '0';
        wait for 2 ns;

        InEn <= '0';

        Inw0 <= '0';
        Inw1 <= '0';
        wait for 2 ns;

        Inw0 <= '0';
        Inw1 <= '1';
        wait for 2 ns;

        Inw0 <= '1';
        Inw1 <= '0';
        wait for 2 ns;

        Inw0 <= '1';
        Inw1 <= '1';
        wait for 2 ns;

        Inw0 <= '0';
        Inw1 <= '0';
```

Figure 2: VHDL code for 2-to-4 decoder testbench



```vhdl
-- Additional Comments:
--
-- Aaranan Sathiendran
-- 101196339
-- SYSC3320 Lab 2: Top level design for Simple 2-to-4 Decoder component
----------------------------------------------------------------------------------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Top_Decoder_2_to_4 is
    Port ( btn : in STD_LOGIC_VECTOR (3 downto 0);
           sw : in STD_LOGIC_VECTOR (3 downto 0);
           led : out STD_LOGIC_VECTOR (3 downto 0));
end Top_Decoder_2_to_4;

architecture Behavioral of Top_Decoder_2_to_4 is

component Decoder_2_to_4 is
    Port( w0 : in STD_LOGIC;
          w1 : in STD_LOGIC;
          En : in STD_LOGIC;
          y0 : out STD_LOGIC;
          y1 : out STD_LOGIC;
          y2 : out STD_LOGIC;
          y3 : out STD_LOGIC);
end component;

begin

    D1 : Decoder_2_to_4
    port map(
        w0 => btn(0),
        w1 => btn(1),
        En => sw(0),
        y0 => led(0),
        y1 => led(1),
        y2 => led(2),
        y3 => led(3));

end Behavioral;
```

Figure 3: VHDL code for creating top level for 2-to-4 decoder

Figure 4: Simulation timing diagram for 2-to-4 decoder

Figure 4 shows that the implemented circuit for the decoder works as expected per the truth table. When the enable bit is 1 and the input is “00” (0–2 ns), y0 = 1, when the input is changed to “01” (2-4 ns), y1 = 1, when the input is changed to “10” (4-6 ns), y2 = 1, and when the input is changed to “11” (6-8 ns), y3 = 1. Once the enable bit is changed to 0 (10-20 ns), there is no longer an output for y.

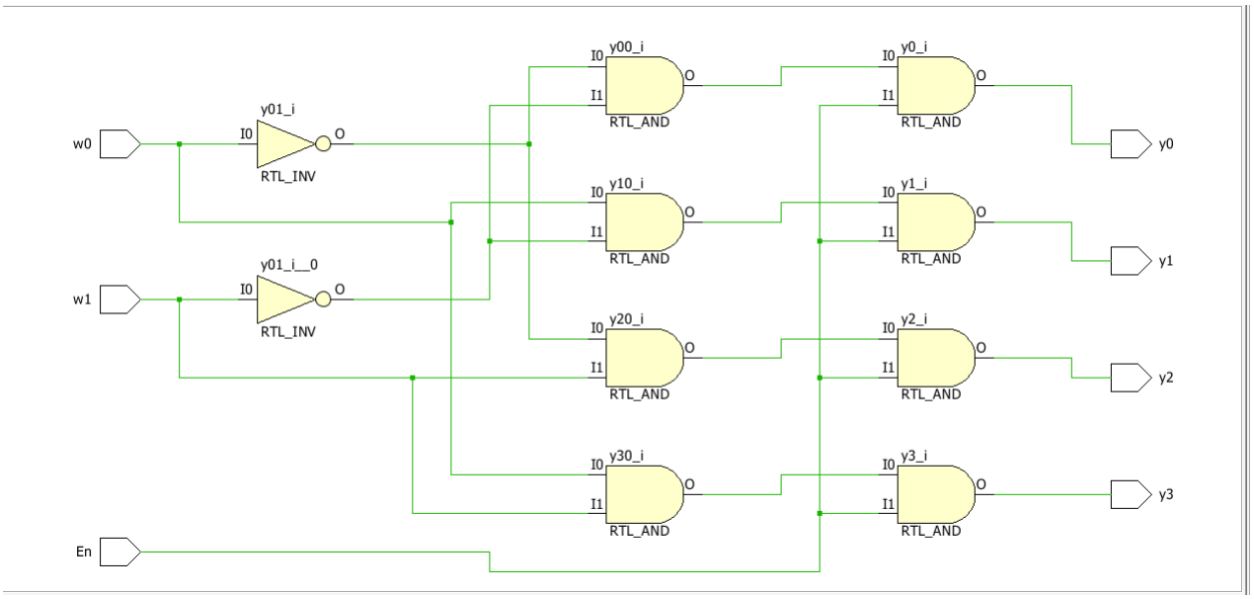


Figure 5: RTL analysis schematic for 2-to-4 decoder

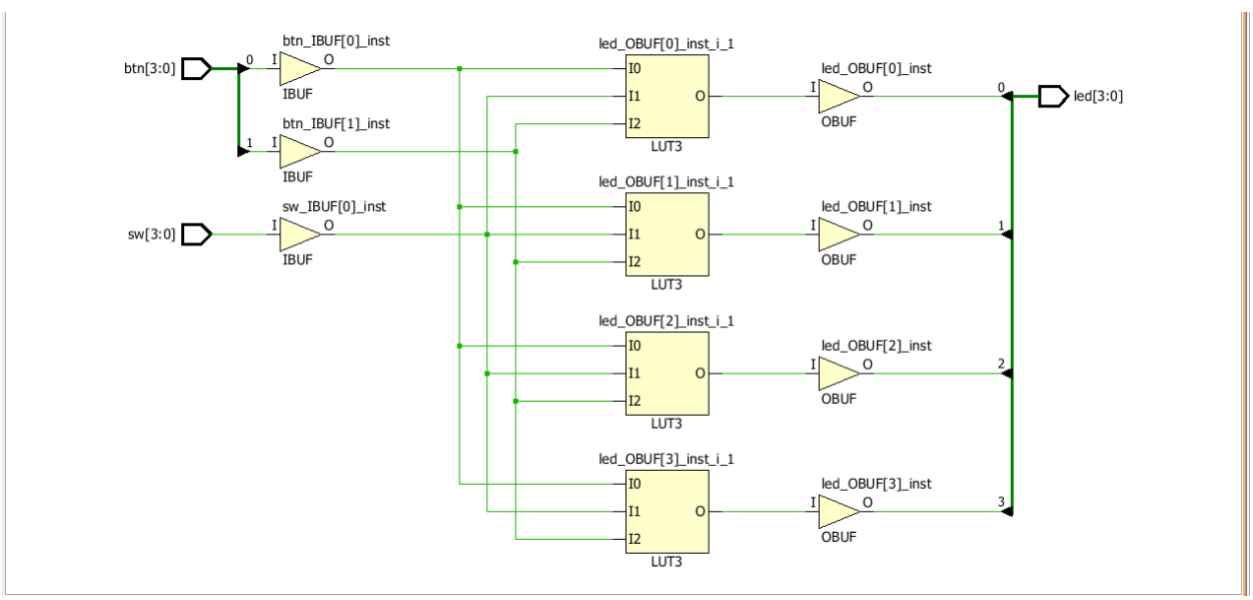


Figure 6: Synthesis schematic for top level of 2-to-4 decoder

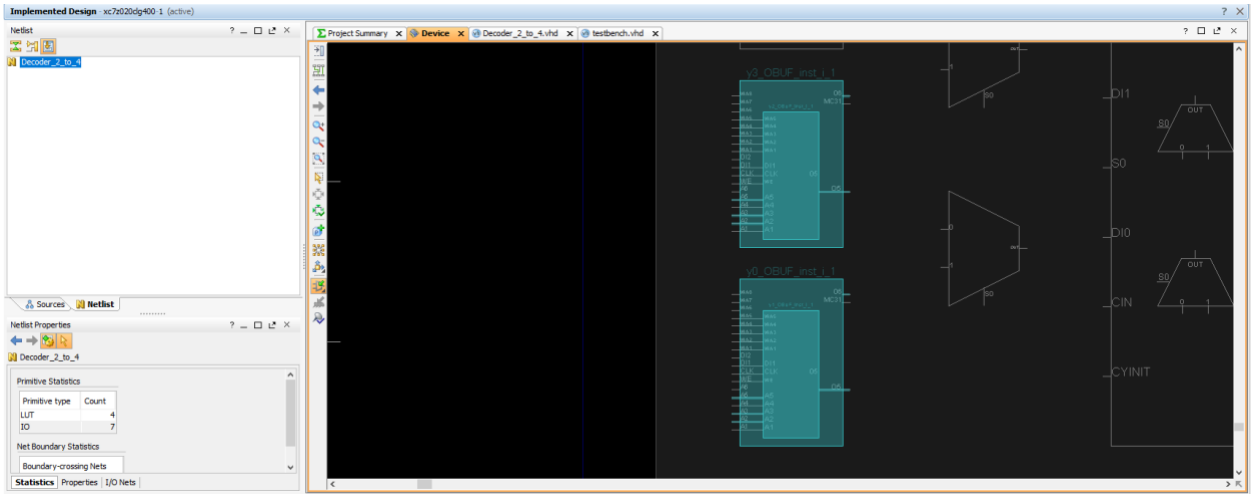


Figure 7: Physical implementation of 2-to-4 decoder on FPGA board

Figure 7 shows that there are 4 lookup tables and 7 I/O blocks in the design. Each LUT is responsible for each y output (y0-y3):

W0	W1	Y0
0	0	1
0	1	0
1	0	0
1	1	0

W0	W1	Y1
0	0	0
0	1	1
1	0	0
1	1	0

W0	W1	Y2
0	0	0
0	1	0
1	0	1
1	1	0

W0	W1	Y3
0	0	0
0	1	0
1	0	0
1	1	1

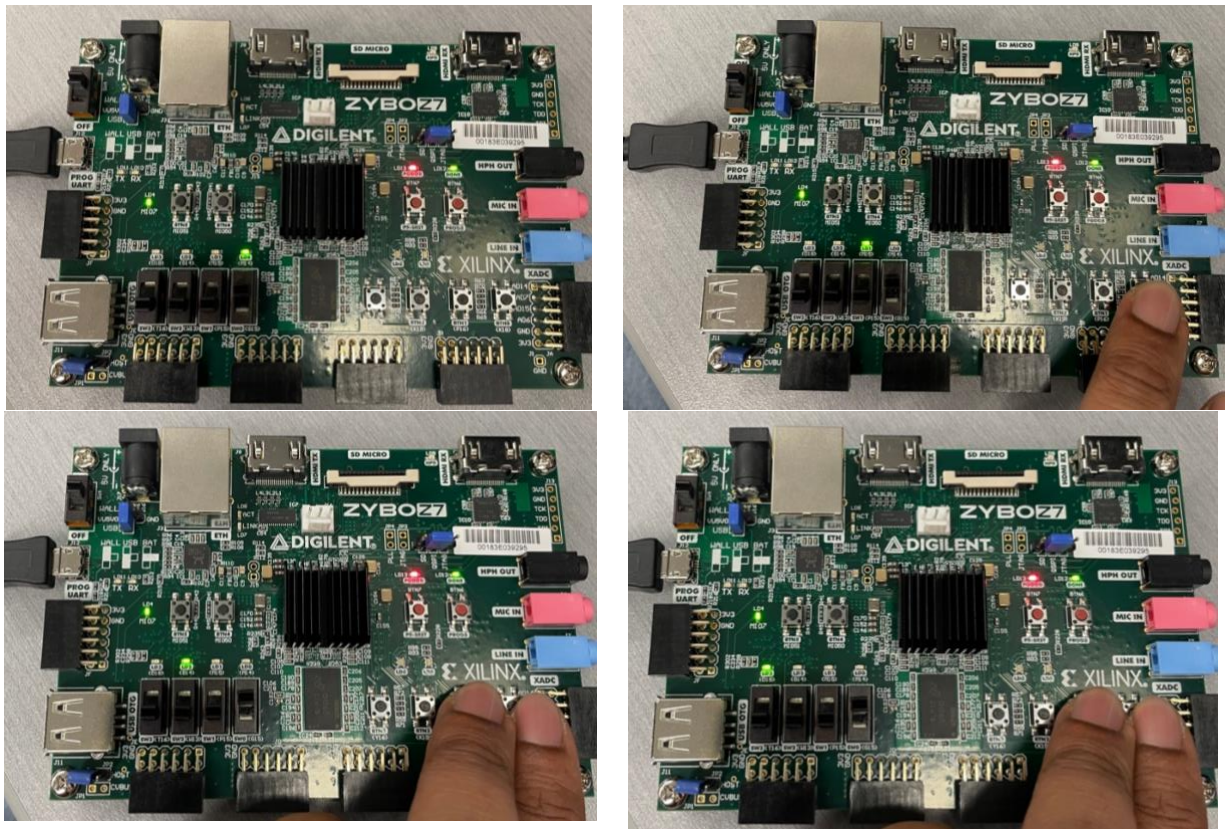


Figure 8: Hardware showing LED output of each input combination for 2-to-4 decoder implemented on FPGA

Like Figure 4, Figure 8 shows that the circuit behaves as expected per the truth table. When the enable switch is on and the input is “00” LED 0 is on, when the input is changed to “01” LED 1 is on, when the input is changed to “10” LED 2 is on, and when the input is changed to “11” LED 3 is on. Once the enable switch is off, none of the LEDs will turn on.

4-bit up-counter:

```vhdl
-- 101196339
-- SYSC3320 Lab 2: 4 bit up counter design using adder/subtractor
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.all;

entity Counter_design1 is
    Port ( clock : in STD_LOGIC;
           reset : in STD_LOGIC;
           count : out STD_LOGIC_VECTOR (3 downto 0));
end Counter_design1;

architecture Behavioral of Counter_design1 is

signal cnt : std_logic_vector (3 downto 0);

begin

    process(clock, reset)
    begin
        if reset = '1' then
            cnt <= "0000";
           elsif rising_edge(clock) then
            cnt <= cnt + '1';
        end if;
    end process;

    count <= cnt;

end Behavioral;
```

Figure 9: VHDL code for counter design 1 (using adder/subtractor)

```vhdl
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Counter_design2 is
    Port ( clock : in STD_LOGIC;
           reset : in STD_LOGIC;
           count : out STD_LOGIC_VECTOR (3 downto 0));
end Counter_design2;

architecture Behavioral of Counter_design2 is

signal cnt : std_logic_vector (3 downto 0);

begin

    process(clock, reset)
    begin
        if reset = '1' then
            cnt <= "0000";
       elsif rising_edge(clock) then
        cnt(3) <= cnt(3) xor (cnt(2) and cnt(1) and cnt(0));
        cnt(2) <= cnt(2) xor (cnt(1) and cnt(0));
        cnt(1) <= cnt(1) xor cnt(0);
        cnt(0) <= not cnt(0);
        end if;
    end process;

    count <= cnt;

end Behavioral;
```

Figure 10: VHDL code for counter design 2 (using flip flops)

Counter_design1.vhd | Counter_design2.vhd | testbench.vhd | Schematic | Schematic (2) | Zybo_board_IO_mapping.xdc | Top_counter.vhd | Schematic (3) | Schemat

M:/SYSC3320/Labs/Lab2/Counter/Counter.srcs/sim_1/new/testbench.vhd

```vhdl
-- SYSC3320 Lab 2: testbench for 4 bit up counter design
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity testbench is
--  Port ( );
end testbench;

architecture Behavioral of testbench is
    signal Inclock: std_logic :='0';
    signal Inreset: std_logic :='0';
    signal Outcount1: std_logic_vector (3 downto 0);
    signal Outcount2: std_logic_vector (3 downto 0);

component Counter_design1 is
    Port ( clock : in STD_LOGIC;
           reset : in STD_LOGIC;
           count : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Counter_design2 is
    Port ( clock : in STD_LOGIC;
           reset : in STD_LOGIC;
           count : out STD_LOGIC_VECTOR (3 downto 0));
end component;

begin
    C1: Counter_design1 port map(clock => Inclock, reset => Inreset, count => Outcount1);
    C2: Counter_design2 port map(clock => Inclock, reset => Inreset, count => Outcount2);

    process begin
```

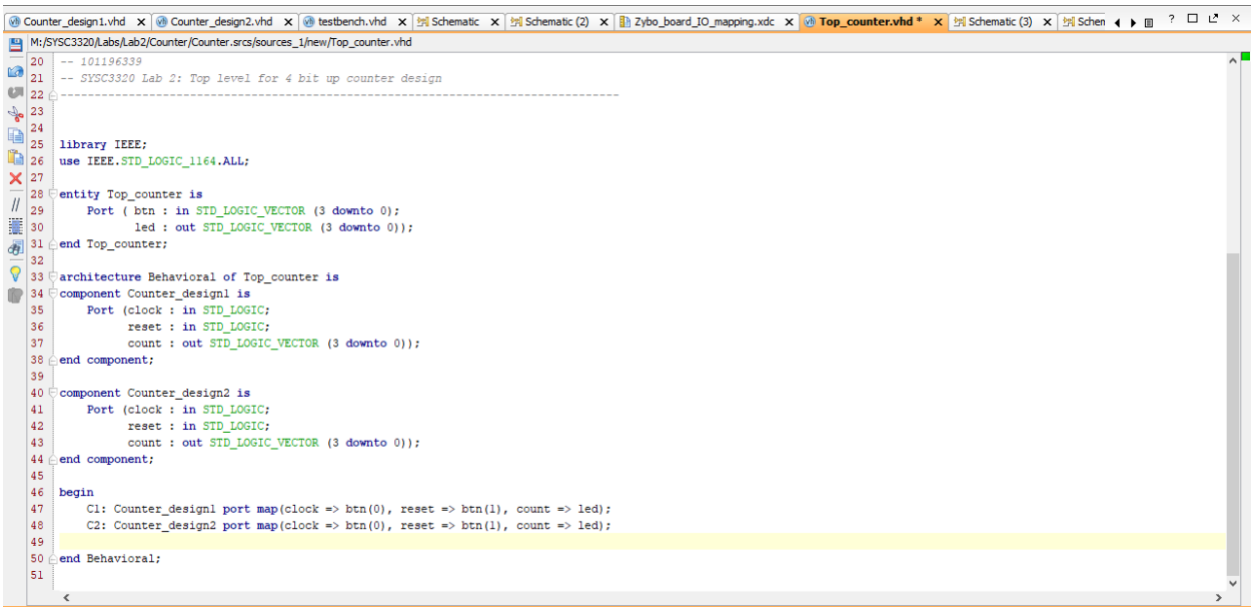
Counter_design1.vhd | Counter_design2.vhd | testbench.vhd | Schematic | Schematic (2) | Zybo_board_IO_mapping.xdc | Top_counter.vhd | Schematic (3) | Schemat

M:/SYSC3320/Labs/Lab2/Counter/Counter.srcs/sim_1/new/testbench.vhd

```vhdl
end component;

component Counter_design2 is
    Port ( clock : in STD_LOGIC;
           reset : in STD_LOGIC;
           count : out STD_LOGIC_VECTOR (3 downto 0));
end component;

begin
    C1: Counter_design1 port map(clock => Inclock, reset => Inreset, count => Outcount1);
    C2: Counter_design2 port map(clock => Inclock, reset => Inreset, count => Outcount2);
```


```vhdl
    process begin
        Inreset <= '1';
        Inclock <= '0';
        wait for 2 ns;
        Inclock <= '1';
        wait for 2ns;
        Inreset <= '0';
        Inclock <= '0';
        wait for 2 ns;

        for i in 0 to 15 loop
            Inclock <= '1';
            wait for 2ns;
            Inclock <= '0';
            wait for 2 ns;
        end loop;
    end process;

end Behavioral;
```

Figure 11: VHDL code for counter testbench



```vhdl
-- 101196339
-- SYSC3320 Lab 2: Top level for 4 bit up counter design
--------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Top_counter is
    Port ( btn : in STD_LOGIC_VECTOR (3 downto 0);
           led : out STD_LOGIC_VECTOR (3 downto 0));
end Top_counter;

architecture Behavioral of Top_counter is
component Counter_design1 is
    Port (clock : in STD_LOGIC;
          reset : in STD_LOGIC;
          count : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Counter_design2 is
    Port (clock : in STD_LOGIC;
          reset : in STD_LOGIC;
          count : out STD_LOGIC_VECTOR (3 downto 0));
end component;

begin
    C1: Counter_design1 port map(clock => btn(0), reset => btn(1), count => led);
    C2: Counter_design2 port map(clock => btn(0), reset => btn(1), count => led);

end Behavioral;
```

Figure 12: VHDL code for top level of counter

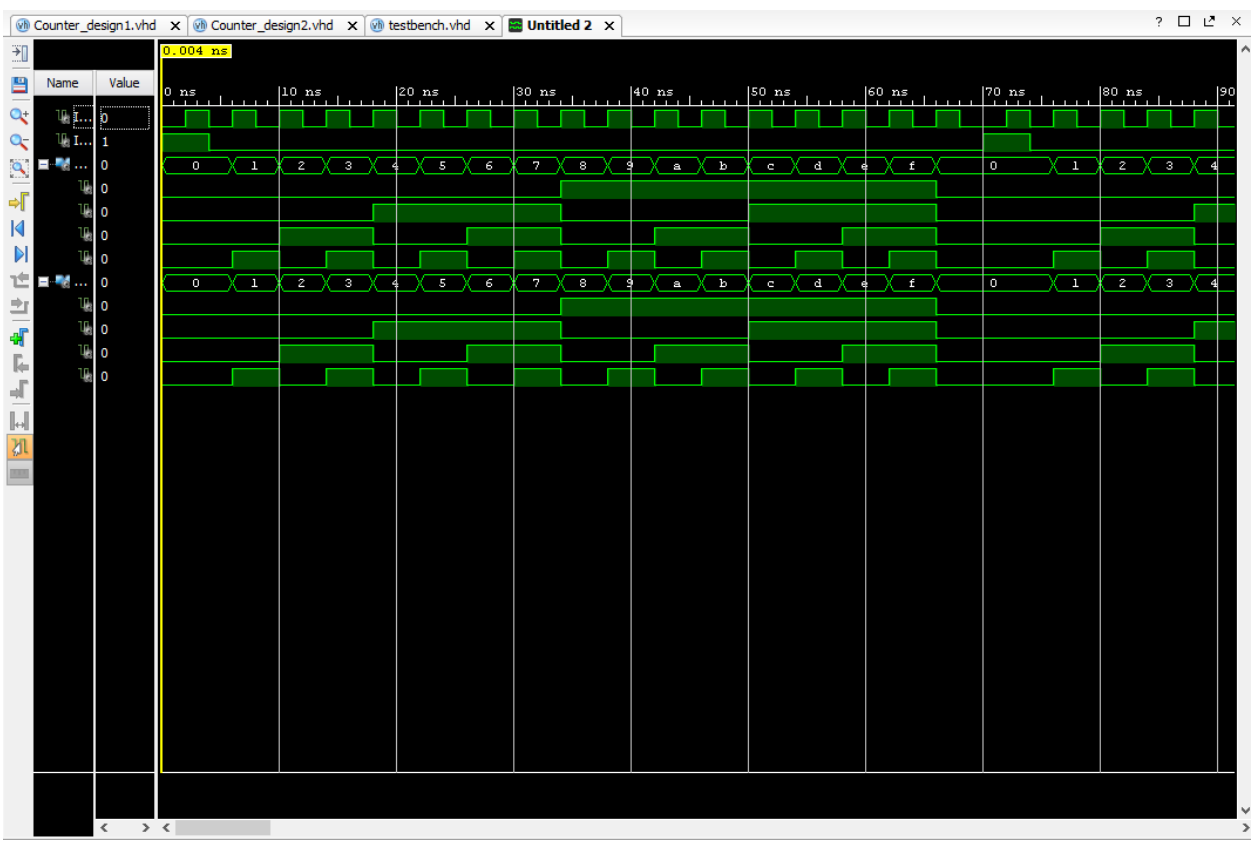


Figure 13: Simulation timing diagram for counter

As shown in Figure 13, both designs for the counter implement the correct design. Every time the clock signal is on the rising edge, the count increments until it reaches 15 and resets.

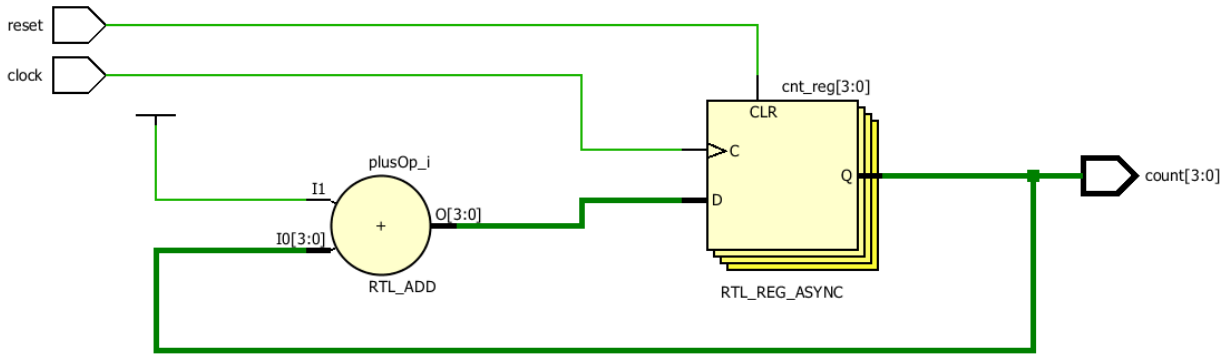


Figure 14: RTL analysis schematic for counter design 1 (using adder/subtractor)

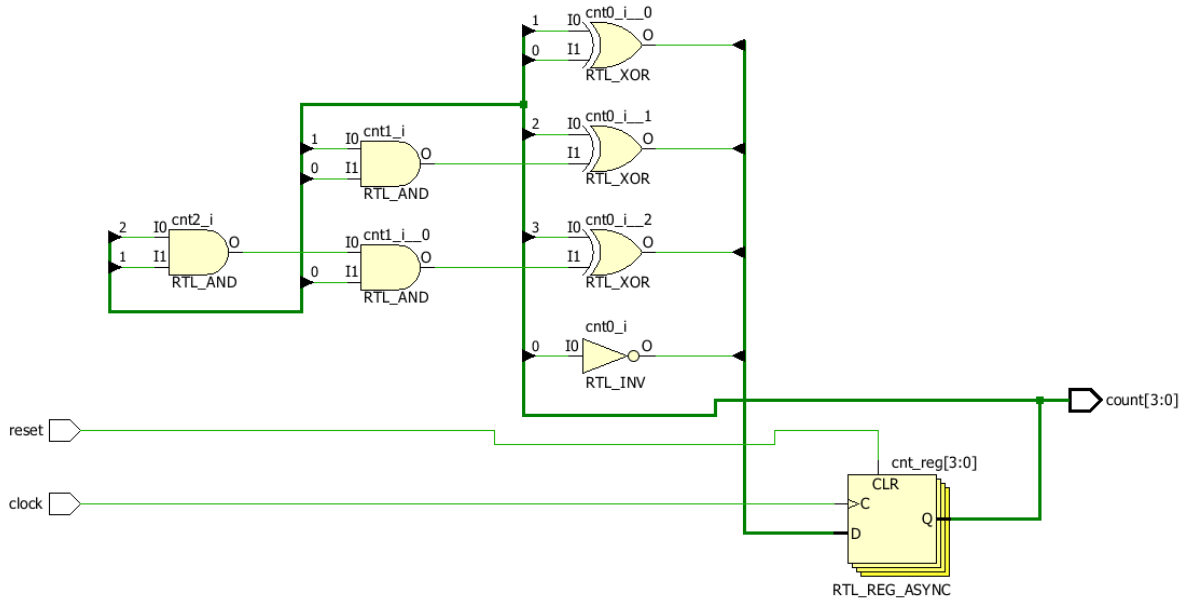


Figure 15: RTL analysis schematic for counter design 2 (using flip flops)

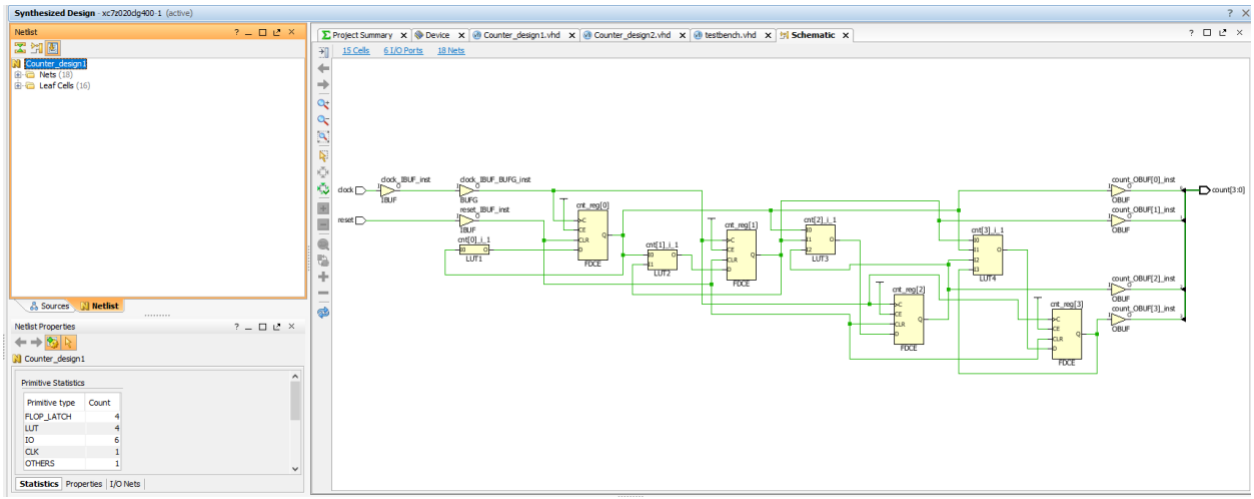


Figure 16: Synthesized design schematic for counter design 1 (using adder/subtractor)

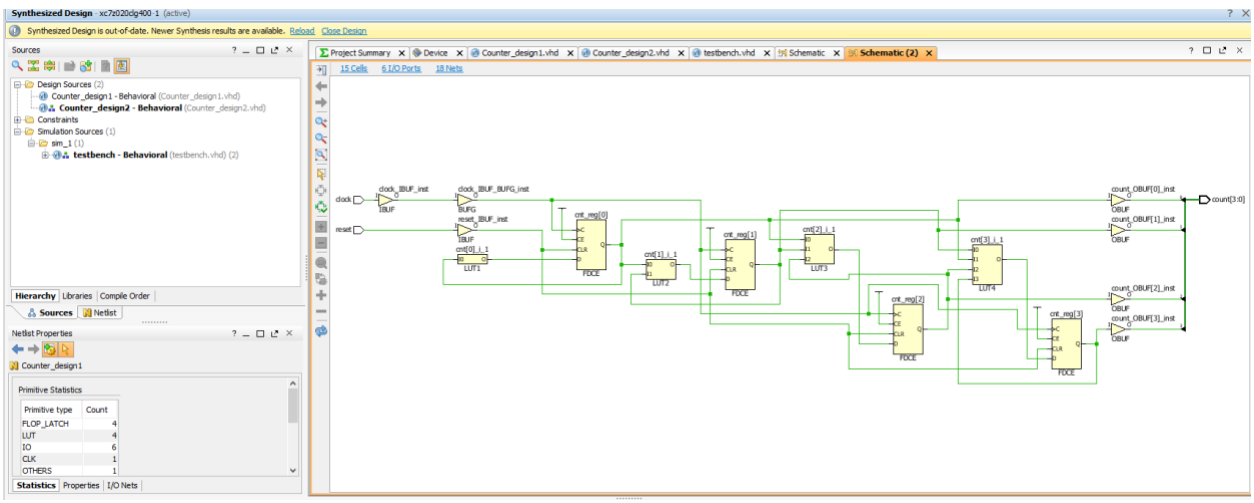


Figure 17: Synthesized design schematic for counter design 2 (using flip flops)

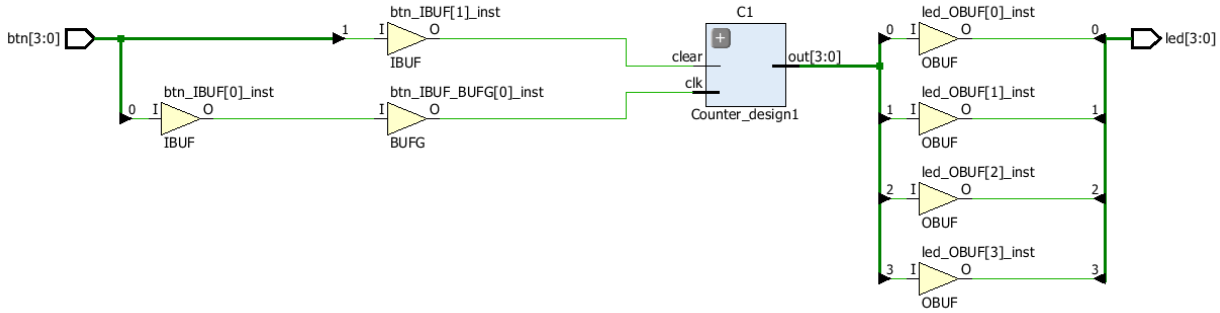


Figure 18: Example of synthesized design schematic for top level of counter design (design 1)

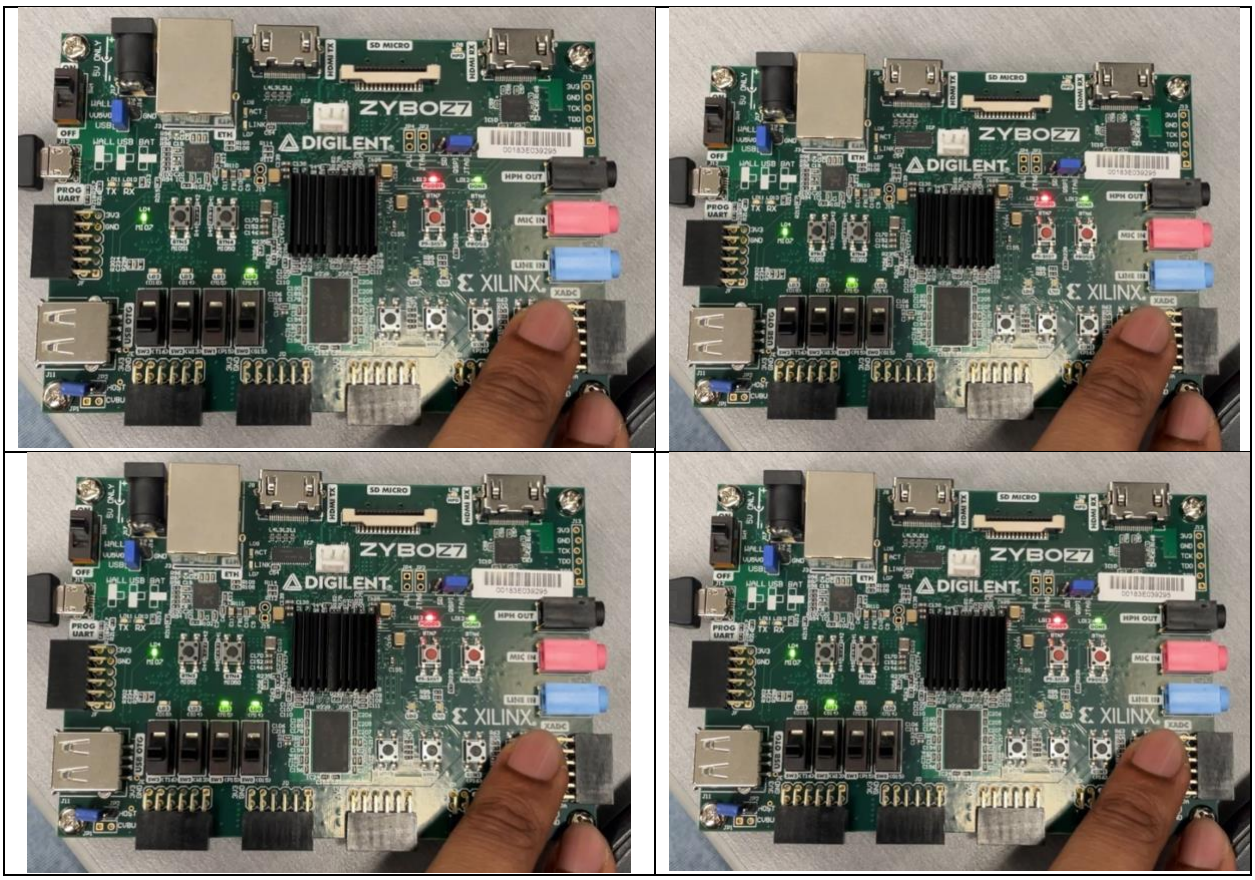


Figure 19: Hardware showing LED output for binary inputs 1 to 4 for physical implementation of counter on FPGA board

Figure 19 shows that the circuit implementation on the board correctly displays the correct binary value when incrementing the counter from 1-4. When the counter is incremented using the button, the LEDs display “0001”, “0010”, “0011” and “0100”.