# YEDITEPE UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

**INTRODUCTION TO DIGITAL SYSTEMS LABORATORY- LAB #3**
**Fall 2021**

## Objective:

Sequential Circuit Design and Implementation at Register-Transfer Level (RTL).

## General Information:

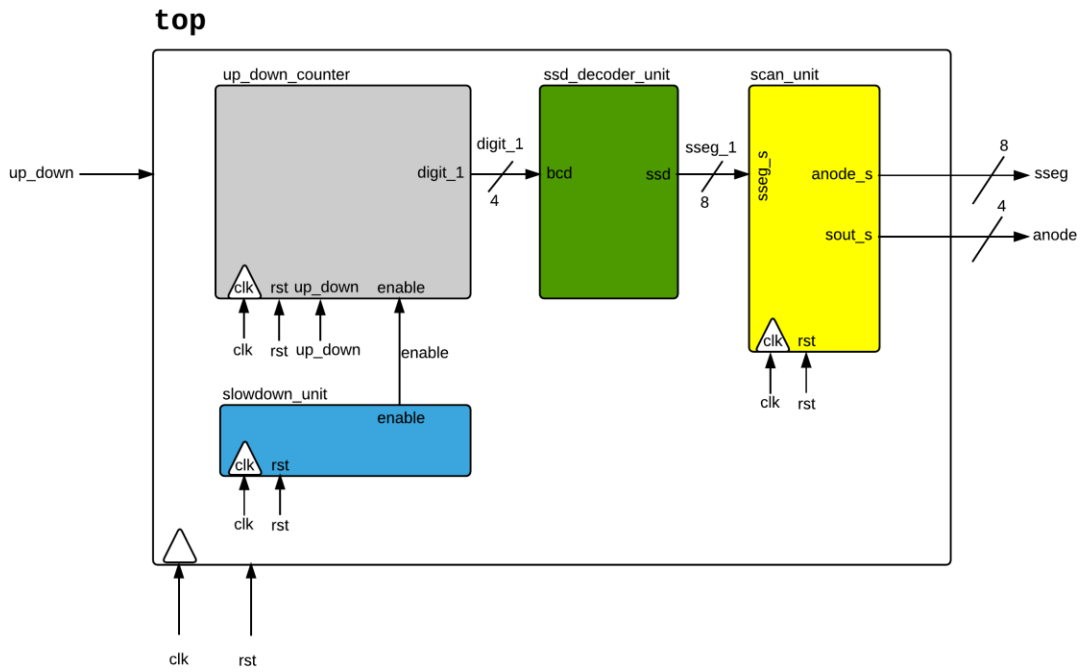
In this lab, you will design, simulate, and then implement sequential circuits which are described at register-transfer level. You are going to use Verilog hardware description language (HDL) both to model your design and simulate it.

## Lab Equipment:

Xilinx ISE Design Suite, Digilent Basys2 or Nexys2 or Nexys4 board.

## Design #1: 1-digit up-down counter

Figure below shows the block diagram of 1-digit up-down counter design. Notice that there are **clk** (**clock**) and **rst** (**reset**) signals in the design.




**Procedure:**

1. Create a new folder named **lab_4** under Desktop folder.
2. Invoke Xilinx ISE Design Suite.
3. Create a new project.
    a. Click on **New Project…** under File menu.
    b. Set the project name, location, and working directory as follows:
        i. Name:

        **up_down_counter**

        ii. Location:

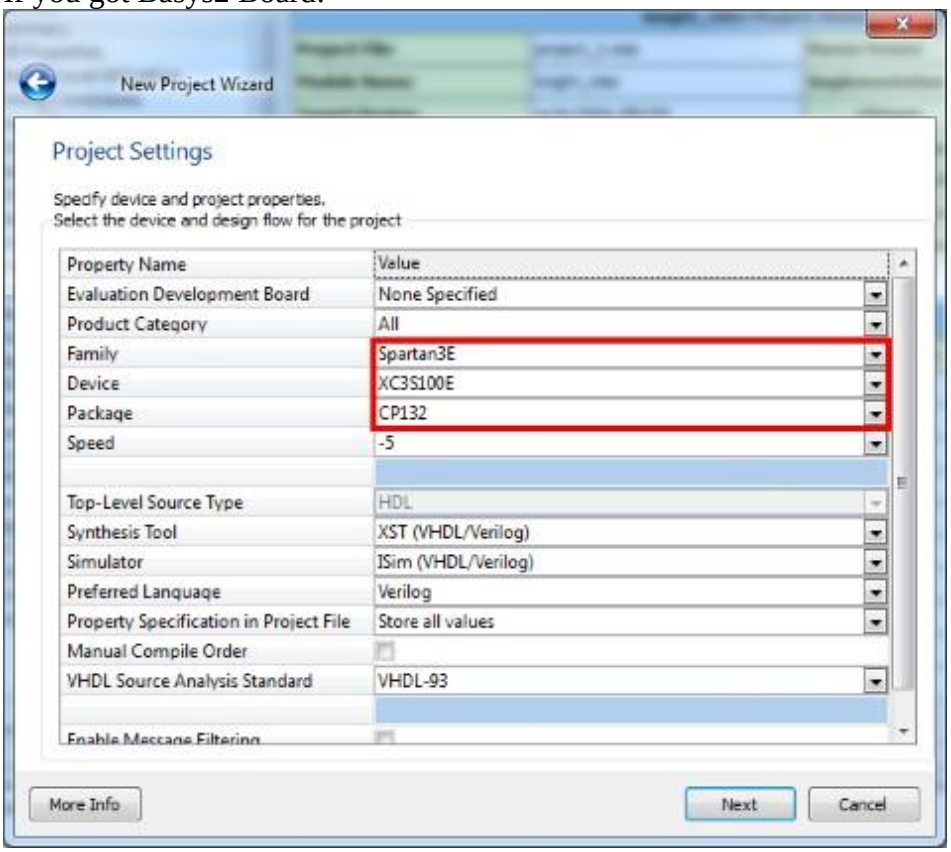
        **C:\Users\<USER>\Desktop\lab_4\up_down_counter**

        iii. Working Directory:

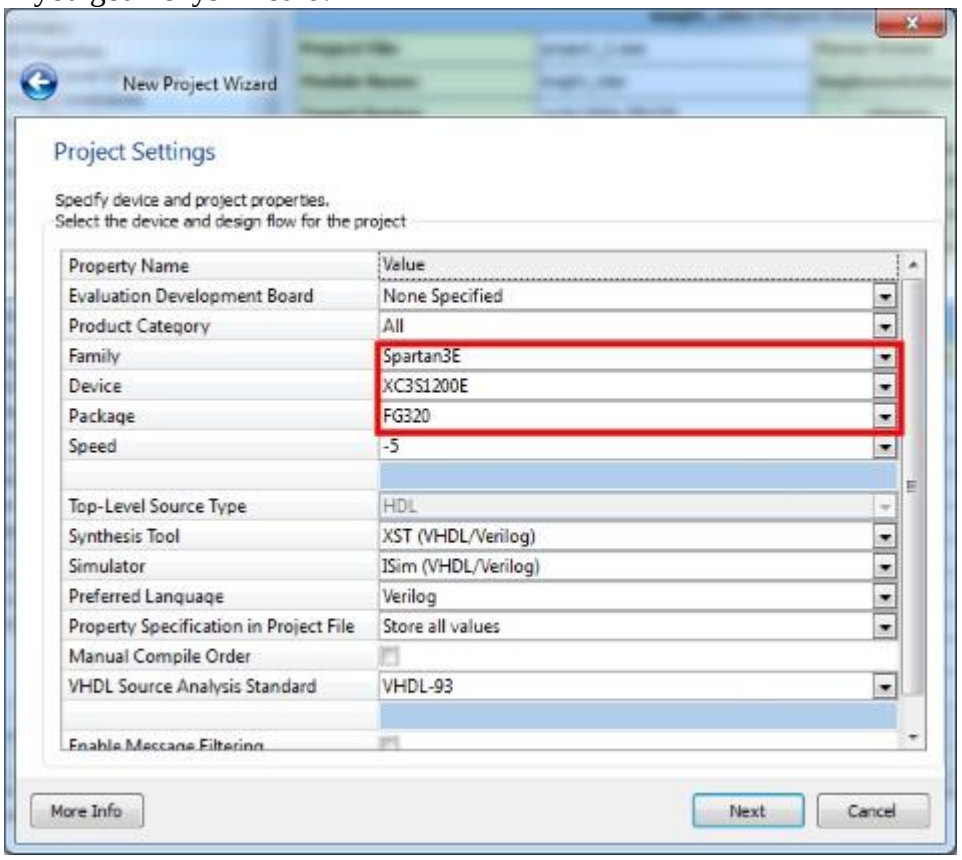
        **C:\Users\<USER>\Desktop\lab_3\up_down_counter**

c. In Project Settings window, set Family, Device, and Package as follows:

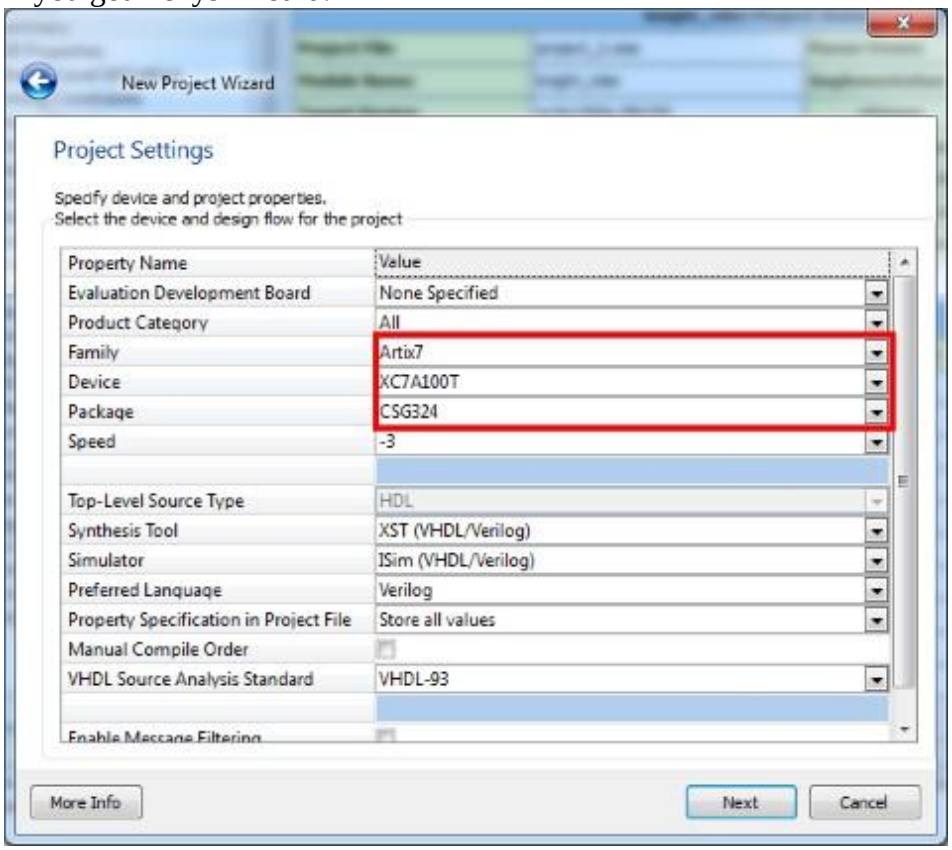
i. If you got Basys2 Board:



ii. If you got Nexys2 Board:




iii. If you got Nexys4 Board:



d. Don’t change anything in Project Settings window. Just click on Next.

e. Check Project Summary window to make sure that your project settings are correct. After that, click on Finish.

4. Create a new source file to describe the top design.

   a. Click on **New Source…** under Project menu.

   b. Select **Verilog Module** and set File name as **top** in Select Source Type window.

   c. Don’t change anything in Define Module window. Just click on Next.

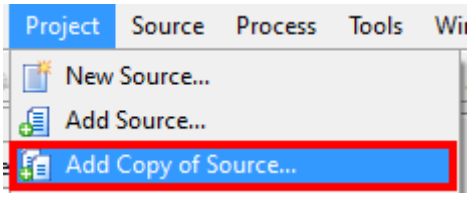
   d. Check Summary window to make sure that your module settings are correct. After that, click on Finish.

5. Complete the module as in the following figure and then click on  button to save your design.

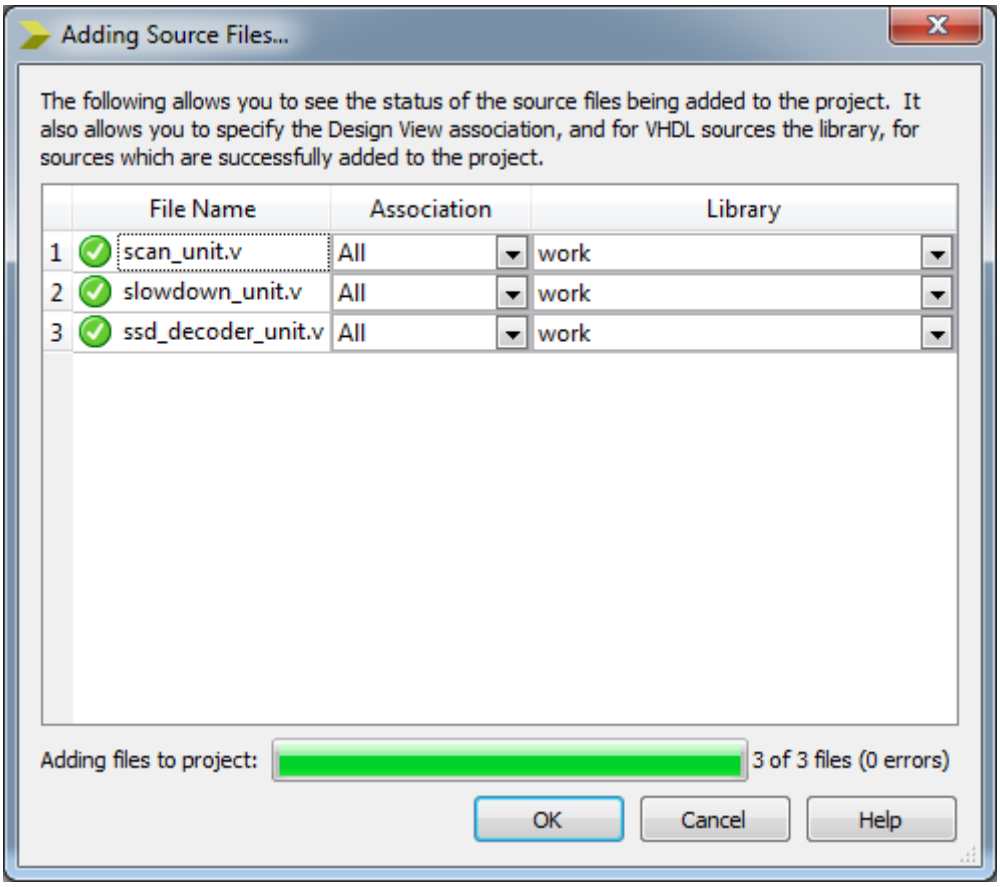
```
`timescale 1ns / 1ps

module top(clk, rst, up_down, sseg, anode);

input clk, rst, up_down;
output [7:0] sseg;
output [3:0] anode;

wire [3:0] digit_1;
wire [7:0] sseg_1;
wire enable;

up_down_counter inst_1(.clk(clk), .rst(rst), .enable(enable), .up_down(up_down), .digit_1(digit_1));

ssd_decoder_unit inst_2(.bcd(digit_1), .ssd(sseg_1));

slowdown_unit inst_3(.clk(clk), .rst(rst), .enable(enable));

scan_unit inst_4(.clk_s(clk), .rst_s(rst), .sseg_s({24'hFFFFFF, sseg_1}), .anode_s(anode), .sout_s(sseg));

endmodule
```

6. Download **ssd_decoder_unit, slowdown_unit,** and **scan_unit** modules from COADSYS as **.v** files and add them into the design.

   a. Click on **Add Copy of Source…** under Project menu.




   b. Select the downloaded Verilog files and click on Open and then click on OK.




7. Create a new source file to describe the up_down_counter design.

    a. Click on **New Source…** under Project menu.

    b. Select **Verilog Module** and set File name as **up_down_counter** in Select Source Type window.

    c. Don't change anything in Define Module window. Just click on Next.

    d. Check Summary window to make sure that your module settings are correct. After that, click on Finish.

8. Complete the module as in the following figure and then click on  button to save your design.

```
module up_down_counter(clk, rst, enable, up_down, digit_1);

   input clk, rst, enable, up_down;
   output reg [3:0] digit_1;

   reg [3:0] digit_1_next;

   always @(posedge clk) begin
      if(rst == 1'b1) begin //RESET
         digit_1 <= 4'b0000;
      end
      else begin
         digit_1 <= digit_1_next;
      end
   end

   always @(*) begin
      if(up_down == 1'b1 && enable == 1'b1) begin // COUNT UP
         if(digit_1 == 4'b1001) begin
            digit_1_next = 4'b0000;
         end
         else begin
            digit_1_next = digit_1 + 1'b1;
         end
      end
      else begin
         digit_1_next = digit_1;
      end
   end


endmodule
```

9. Create a new source file to describe the user constraints of top design.

    e. Click on **New Source…** under Project menu.

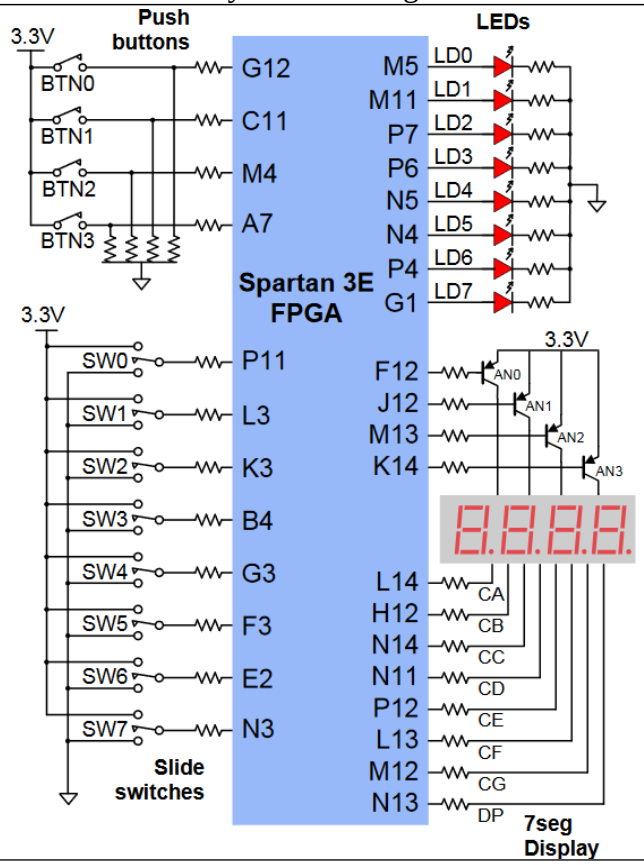
    f. Select **Implementation Constraints File** and set Filename as **top** in Select Source Type window. After that, click on Next.

    g. Click on Finish to add the file into your project.

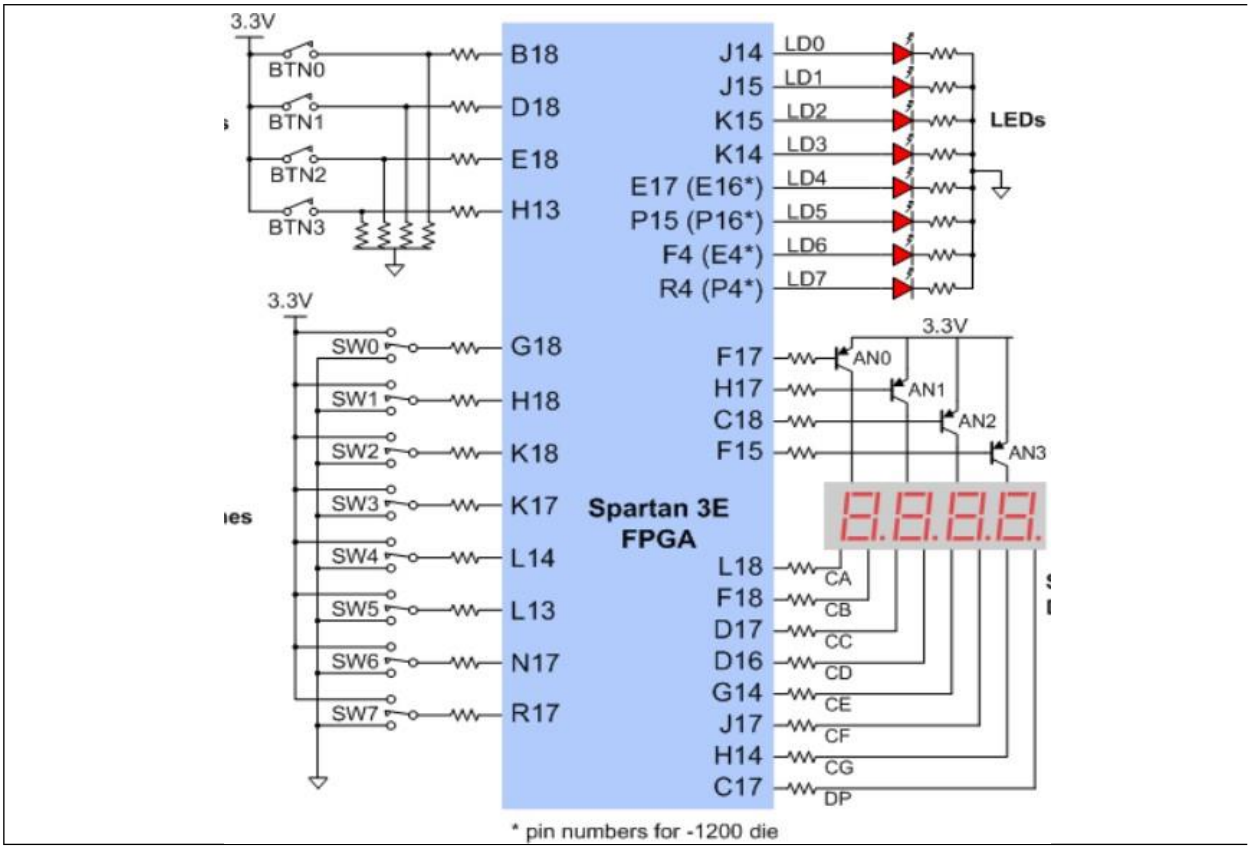
10. Add the following clock constraint to the **top.ucf** file and use the below figures to describe the user constraints of your top design.

    NET "clk" LOC = "B8"; # **(FOR BASYS2 AND NEXYS2)**
    NET "clk" LOC = "E3" | IOSTANDARD = "LVCMOS33"; # **(FOR NEXYS4)**

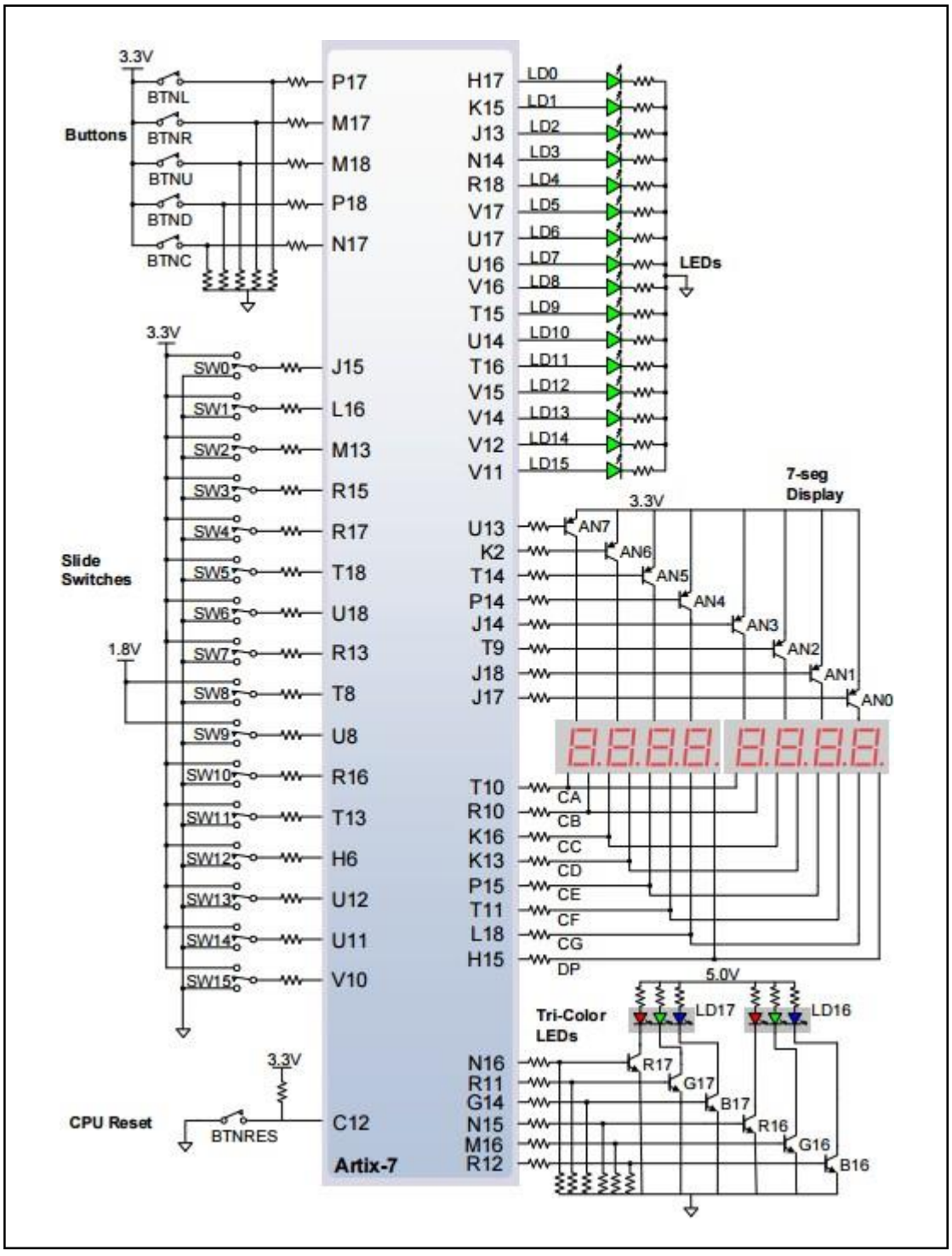
Basys2 Circuit Figure




Nexys2 Circuit Figure




Nexys4 Circuit Figure

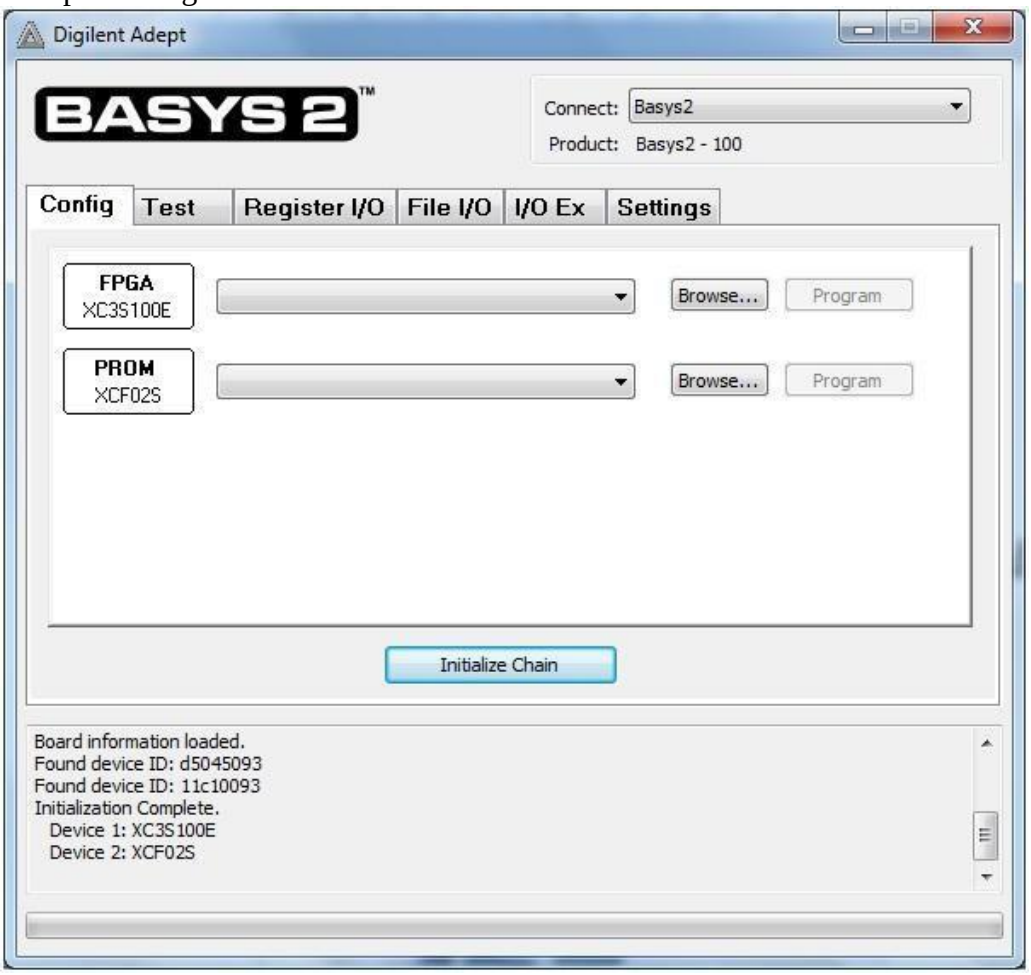



11. While on **Implementation** view, click on **Implementation** to start implementing the design.

12. After **Implementation** is completed click on **Generate Programming File** to generate "**.bit**" file.

13. After connecting the FPGA board to the computer and it is powered on, invoke Digilent Adept.

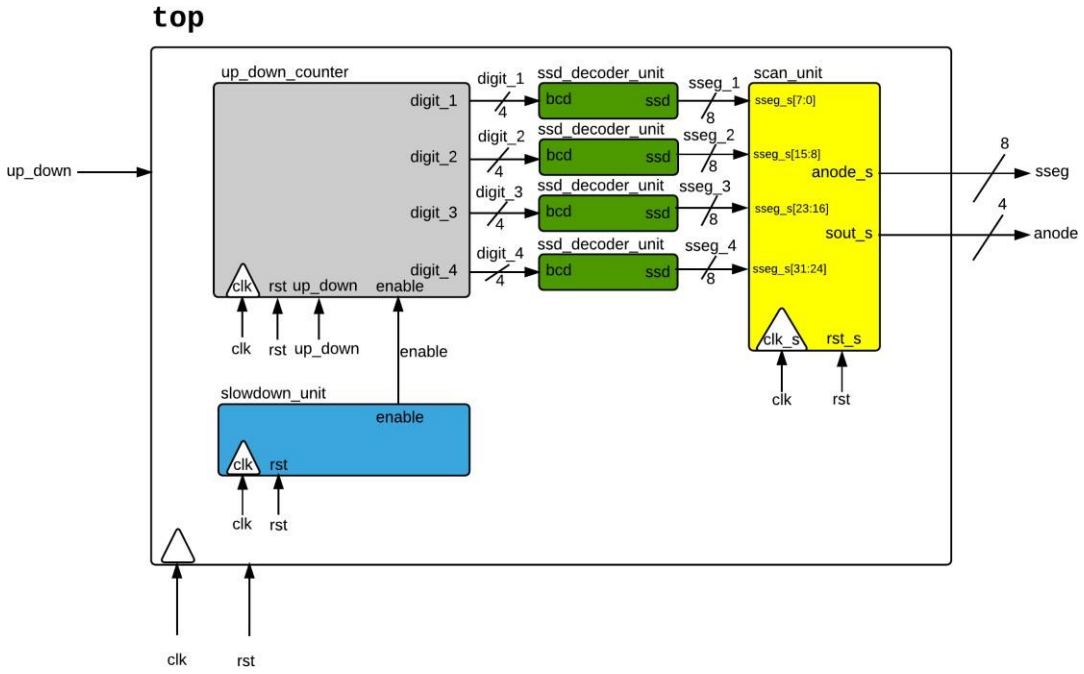

14. After invoking Adept Select "**top.bit**" from the project folder using browse and then press Program button.




15. After Program is successfully loaded to the FPGA board, check whether your design works properly.

## Design #2: 4-digit counter

Figure below shows the block diagram of 4-digit counter design. To see how the design works, watch the video on COADSYS.




**Since we are dealing with a sequential circuit, we need to input a clock signal to our design. In our case, we will use a**

- **50 MHz (20 ns) clock signal (for Basys2 and Nexys2)**
- **100 MHz (10 ns) clock signal (for Nexys4)**

**while implementing our design.**

**Procedure:**

1. Create a new project.

    f. Click on **New Project…** under File menu.

    g. Set the project name, location, and working directory as follows:

        i. Name:

        **counter_4_digit**

        ii. Location:

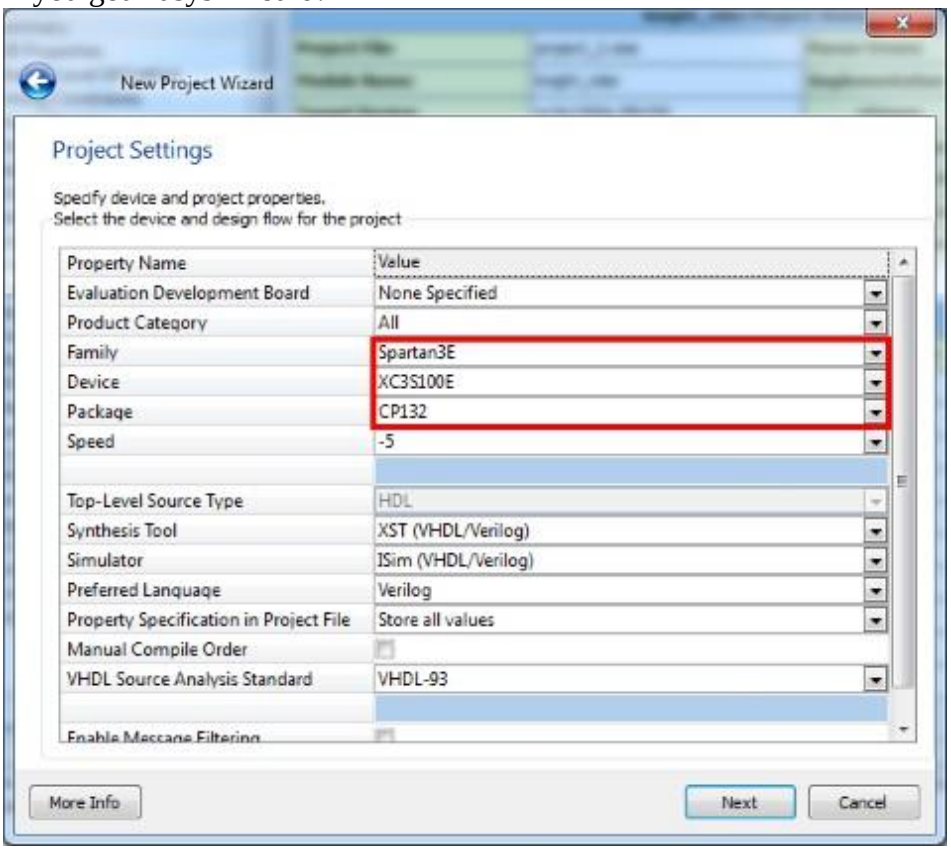
        **C:\Users\<USER>\Desktop\lab_4\counter_4_digit**

        iii. Working Directory:

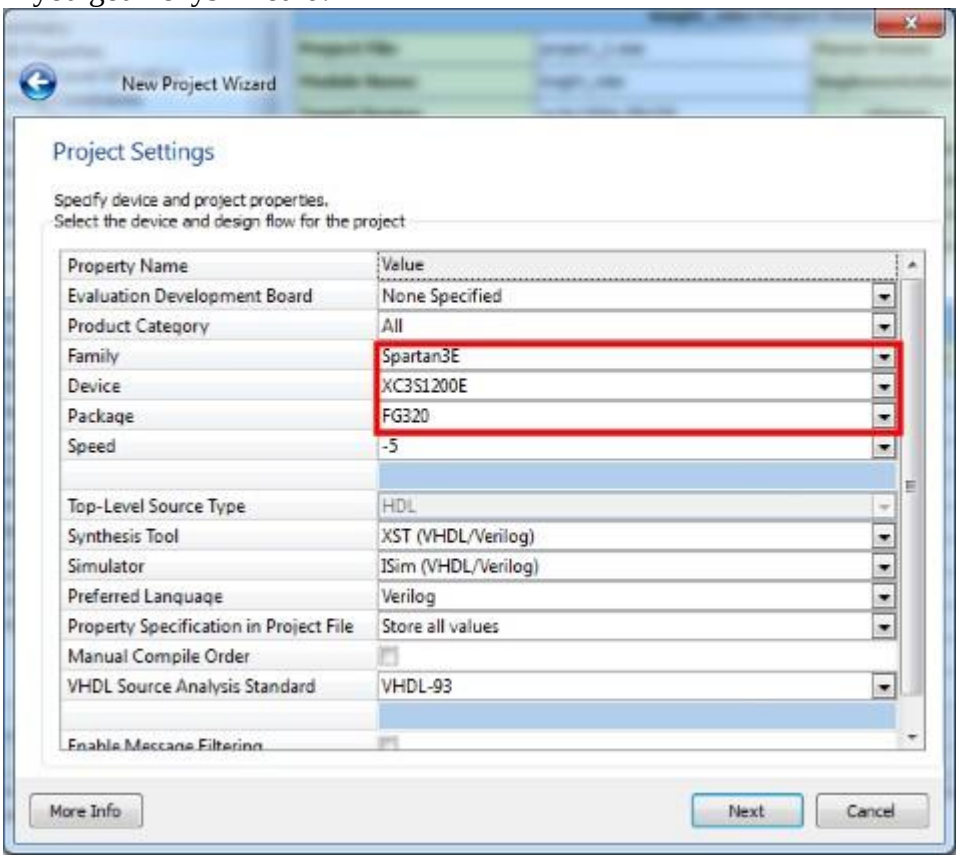
        **C:\Users\<USER>\Desktop\lab_3\counter_4_digit**

    h. In Project Settings window, set Family, Device, and Package as follows:

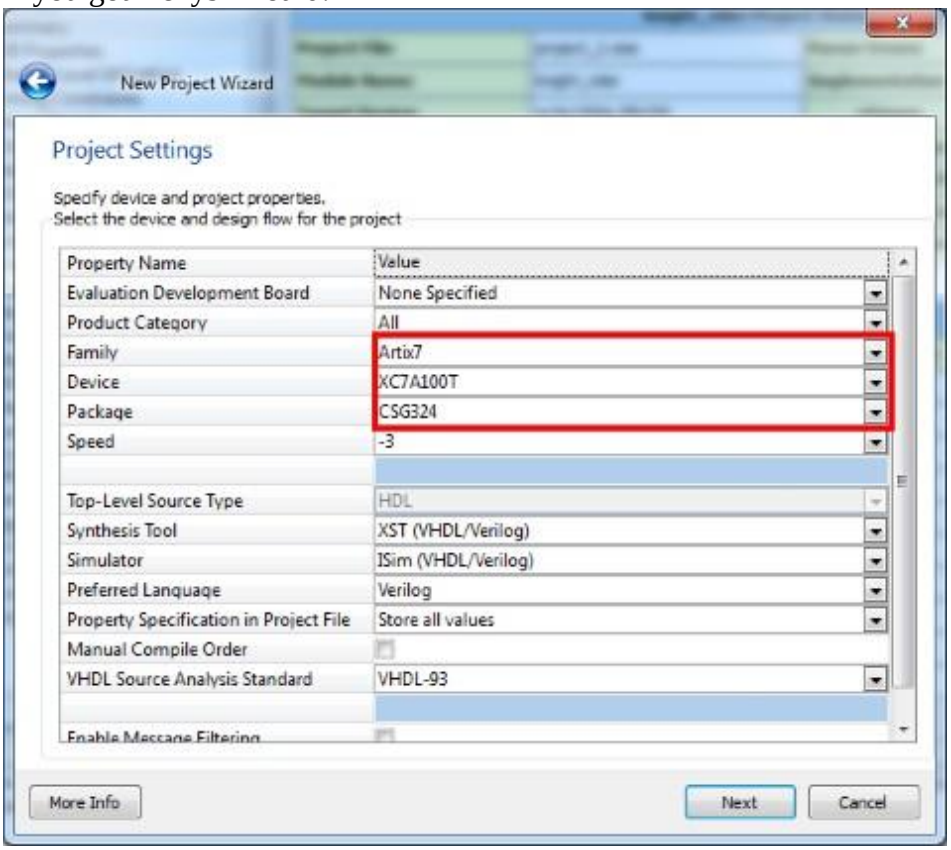
        i. If you got Basys2 Board:



ii. If you got Nexys2 Board:




iii. If you got Nexys4 Board:



i. Don’t change anything in Project Settings window. Just click on Next.

j. Check Project Summary window to make sure that your project settings are correct. After that, click on Finish.

2. Create a new source file to describe the top design.

e. Click on **New Source…** under Project menu.

f. Select **Verilog Module** and set File name as **top** in Select Source Type window.

g. Don’t change anything in Define Module window. Just click on Next.

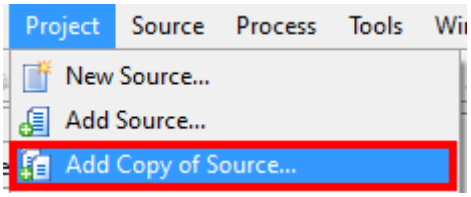
h. Check Summary window to make sure that your module settings are correct. After that, click on Finish.

3. Complete the module as in the following figure and then click on  button to save your design.

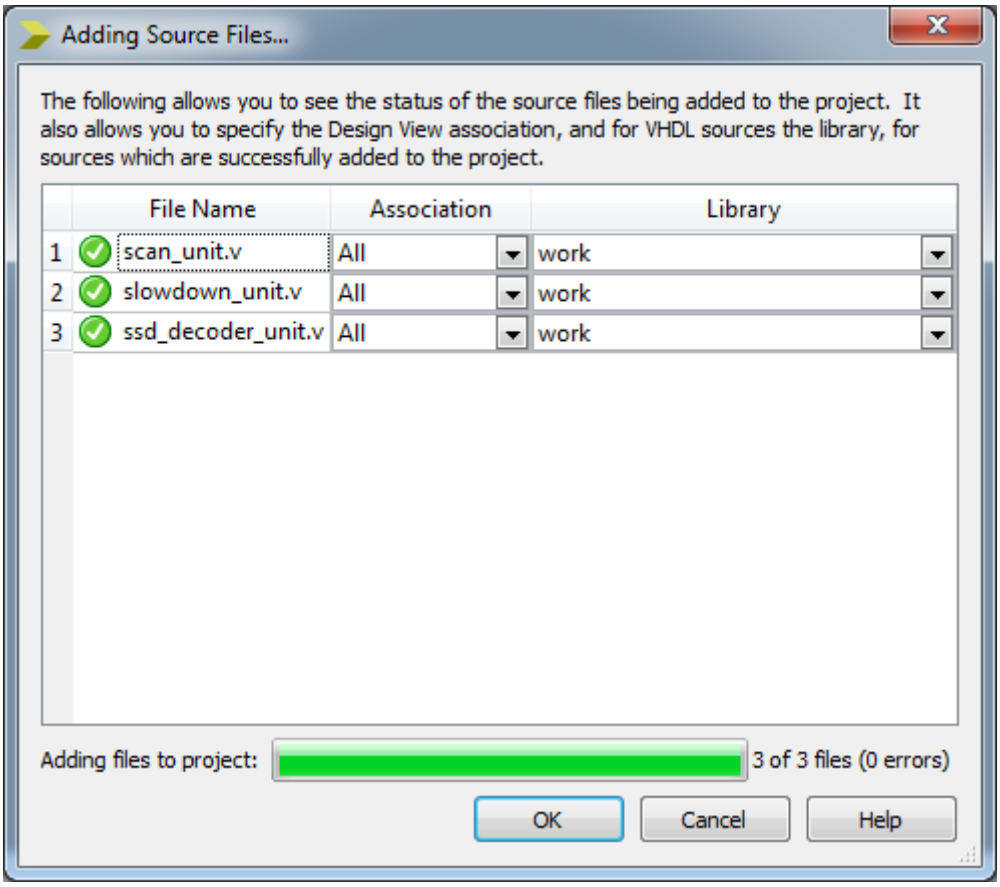
```
1  `timescale 1ns / 1ps
2  module top(clk, rst, up_down, sseg, anode);
3
4     input clk, rst, up_down;
5     output [7:0] sseg;
6     output [3:0] anode;
7
8     wire [3:0] digit_1,digit_2,digit_3,digit_4;
9     wire [7:0] sseg_1,sseg_2,sseg_3,sseg_4;
10    wire enable;
11
12    up_down_counter instance_up_down_counter(.clk(clk),
13                                             .rst(rst),
14                                             .enable(enable),
15                                             .up_down(up_down),
16                                             .digit_1(digit_1),
17                                             .digit_2(digit_2),
18                                             .digit_3(digit_3),
19                                             .digit_4(digit_4));
20
21    ssd_decoder_unit instance_ssd_decoder_1(.bcd(digit_1), .ssd(sseg_1));
22    ssd_decoder_unit instance_ssd_decoder_2(.bcd(digit_2), .ssd(sseg_2));
23    ssd_decoder_unit instance_ssd_decoder_3(.bcd(digit_3), .ssd(sseg_3));
24    ssd_decoder_unit instance_ssd_decoder_4(.bcd(digit_4), .ssd(sseg_4));
25
26    slowdown_unit instance_slowdown(.clk(clk), .rst(rst), .enable(enable));
27
28    scan_unit instance_scan(.clk_s(clk),
29                            .rst_s(rst),
30                            .sseg_s({sseg_4,sseg_3,sseg_2,sseg_1}),
31                            .anode_s(anode),
32                            .sout_s(sseg));
33
34 endmodule
```

4. Download **ssd_decoder_unit, slowdown_unit,** and **scan_unit** modules from COADSYS as **.v** files and add them into the design.

   c. Click on **Add Copy of Source…** under Project menu.




   d. Select the downloaded Verilog files and click on Open and then click on OK.




5. Create a new source file to describe the up_down_counter design.

    h. Click on **New Source…** under Project menu.

    i. Select **Verilog Module** and set File name as **up_down_counter** in Select Source Type window.

    j. Don't change anything in Define Module window. Just click on Next.

    k. Check Summary window to make sure that your module settings are correct. After that, click on Finish.

6. Complete the module as in the following figure and then click on  button to save your design.

```
module up_down_counter(clk, rst, enable, up_down, digit_1, digit_2, digit_3, digit_4);

   input clk, rst, enable, up_down;
   output reg [3:0] digit_1, digit_2, digit_3, digit_4;

   reg [3:0] digit_1_next, digit_2_next, digit_3_next, digit_4_next;

   always @(posedge clk) begin
      if(rst == 1'b1) begin //RESET
         //TODO
      end
      else begin
         //TODO
      end
   end

   always @(*) begin
      if(up_down == 1'b1 && enable == 1'b1) begin // COUNT UP
         //TODO
      end
      else if(up_down == 1'b0 && enable == 1'b1) begin // COUNT UP
         //TODO
      end
      else begin
         //TODO
      end
   end


endmodule
```

7. Edit the **slowdown_unit** module that you get from COADSYS so that enable will be set every **100ms**.

8. Create a new source file to describe the user constraints of top design.

   l. Click on **New Source…** under Project menu.

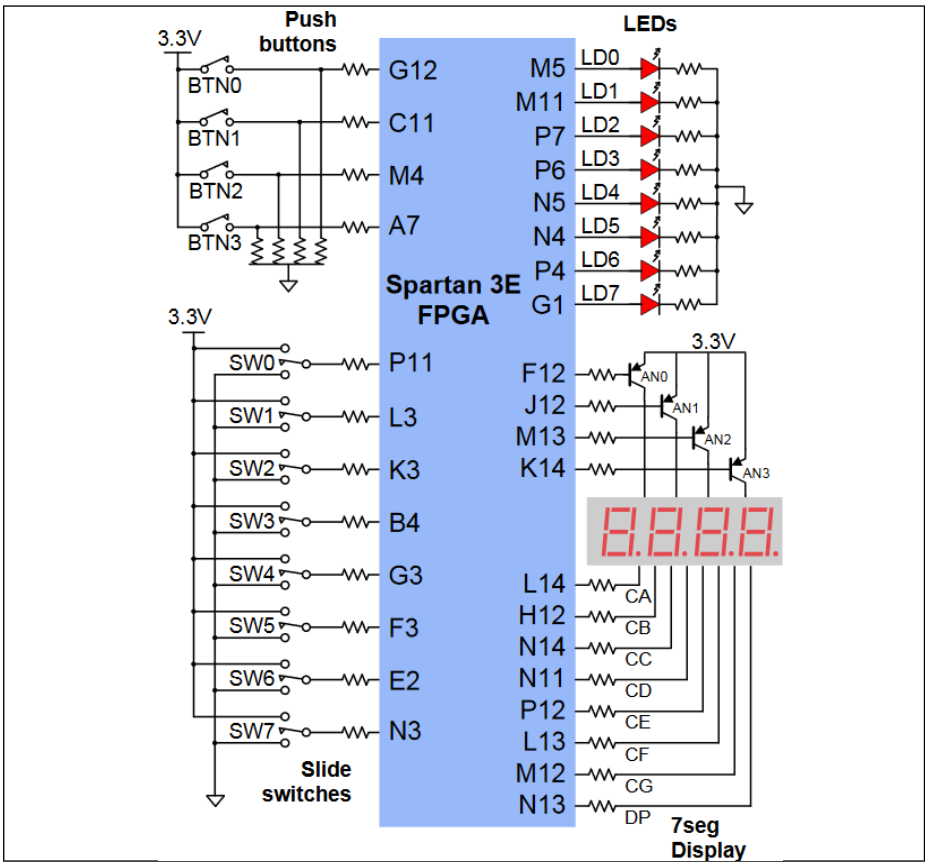
   m. Select **Implementation Constraints File** and set Filename as **top** in Select Source Type window. After that, click on Next.

   n. Click on Finish to add the file into your project.

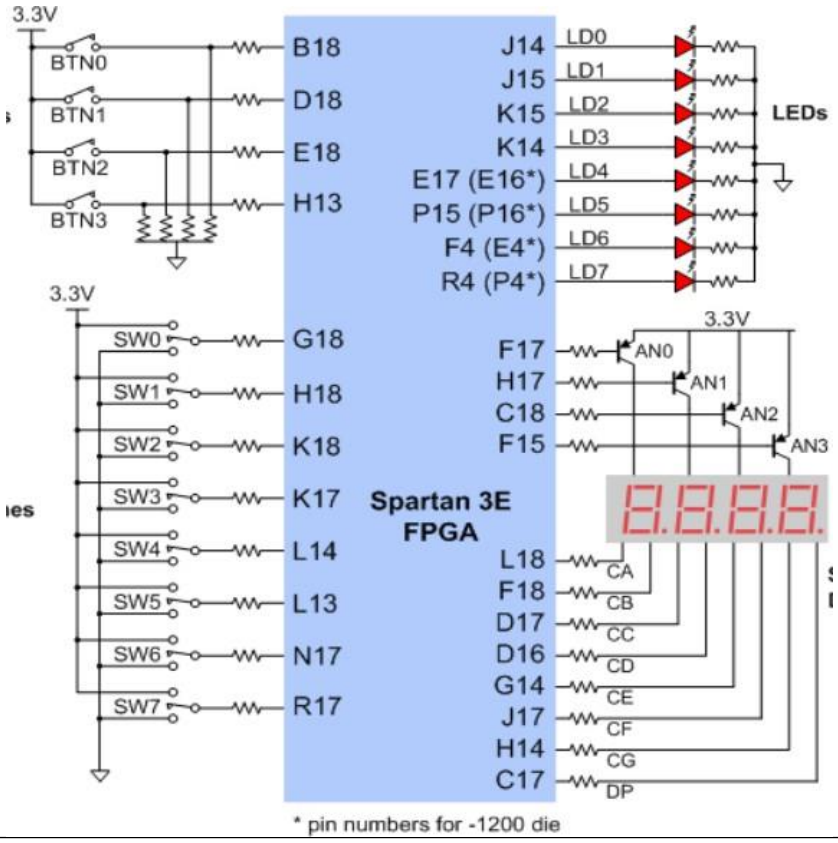
9. Add the following clock constraint to the **top.ucf** file and use the below figures to describe the user constraints of your top design.

   NET "clk"  LOC = "B8"; # **(FOR BASYS2 AND NEXYS2)**
   NET "clk"  LOC = "E3"; # **(FOR NEXYS4)**

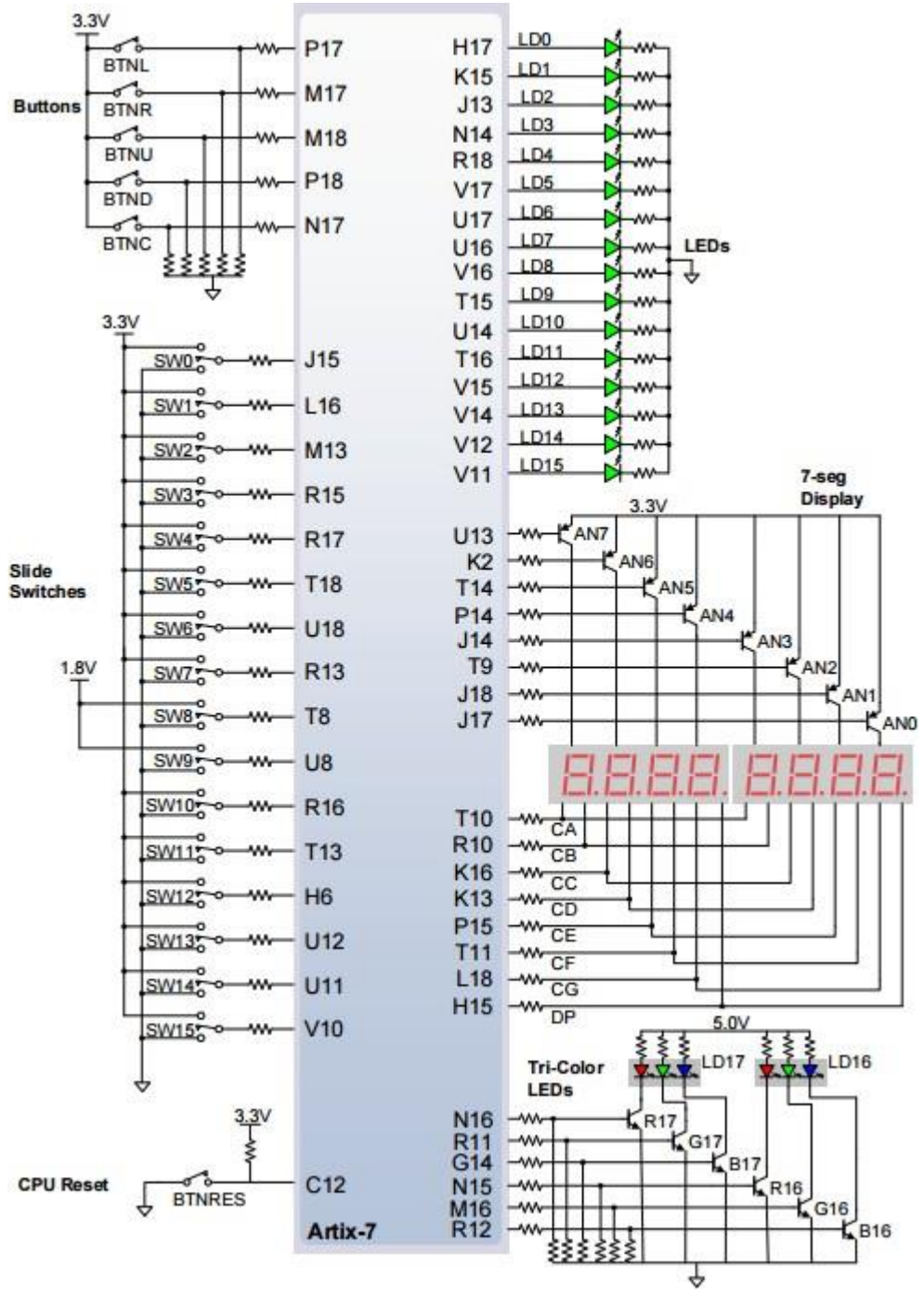
Basys2 Circuit Figure




Nexys2 Circuit Figure



Nexys4 Circuit Figure

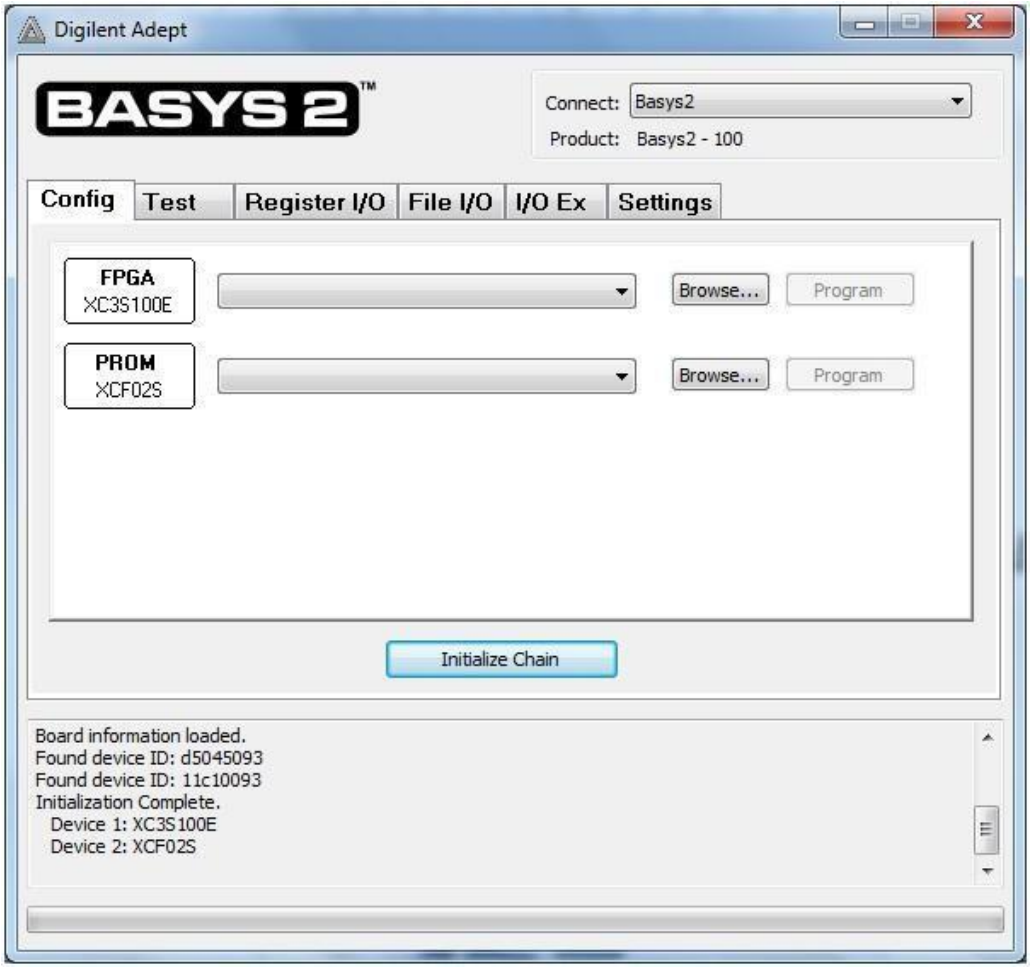



10. While on **Implementation** view, click on **Implementation** to start implementing the design.

11. After **Implementation** is completed click on **Generate Programming File** to generate "**.bit**" file.

12. After connecting the FPGA board to the computer and it is powered on, invoke Digilent Adept.


13. After invoking Adept Select "**top.bit**" from the project folder using browse and then press Program button.




14. After Program is successfully loaded to the FPGA board, check whether your design works properly.