YEDITEPE UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

INTRODUCTION TO DIGITAL SYSTEMS LABORATORY- LAB #5
Spring 2022

Objective:

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

General Information:

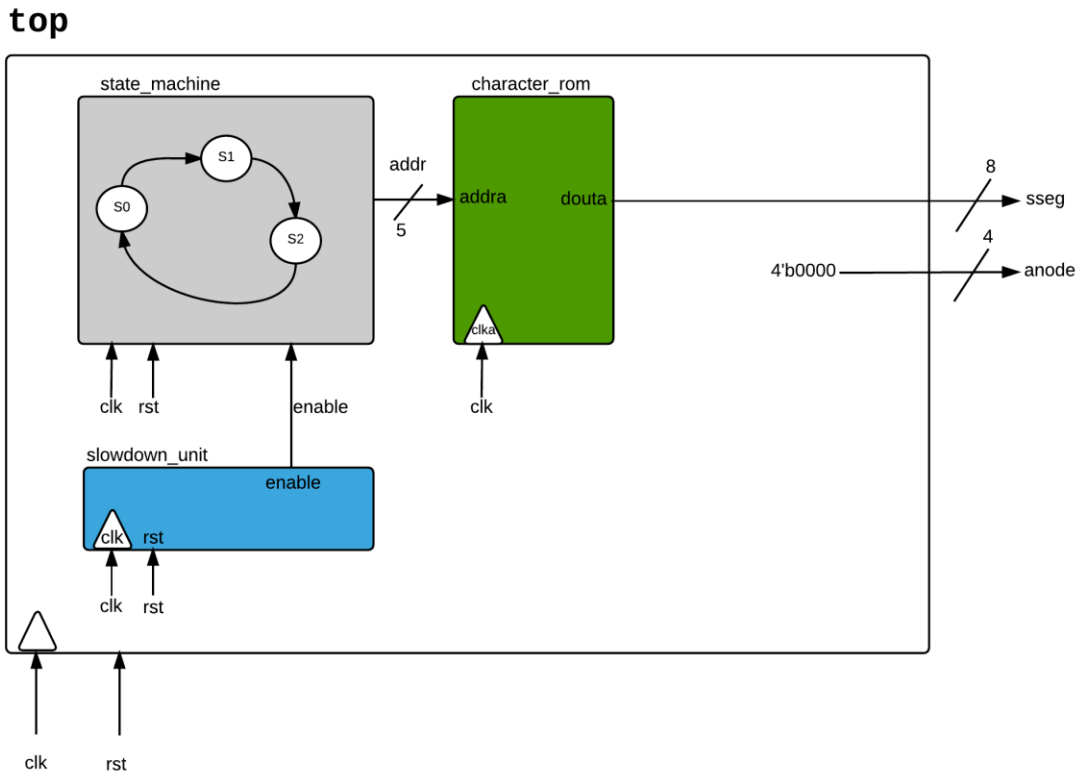
In this lab, you will design and implement sequential circuits which are described at register-transfer level. You are going to use Verilog hardware description language (HDL) both to model your design and implement it.

Lab Equipment:

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

Design #1: Seven-Segment Banner

Figure below shows the block diagram of seven-segment banner design. Notice that there are **clk** (**clock**) and **rst** (**reset**) signals in the design. There are also a read-only memory (ROM) and a state machine within the design.



Procedure:

1. Create a new folder named **lab_5** 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:

        rotating_banner

        ii. Location:

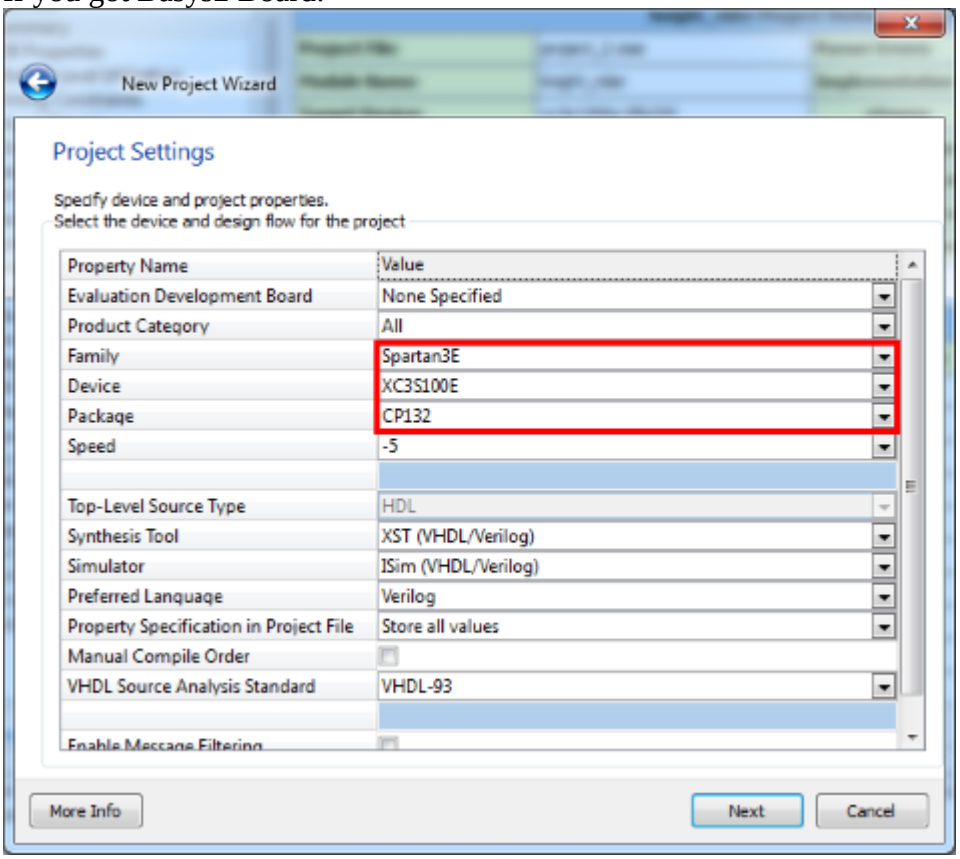
        C:\Users\<USER>\Desktop\lab_5\rotating_banner

iii. Working Directory:

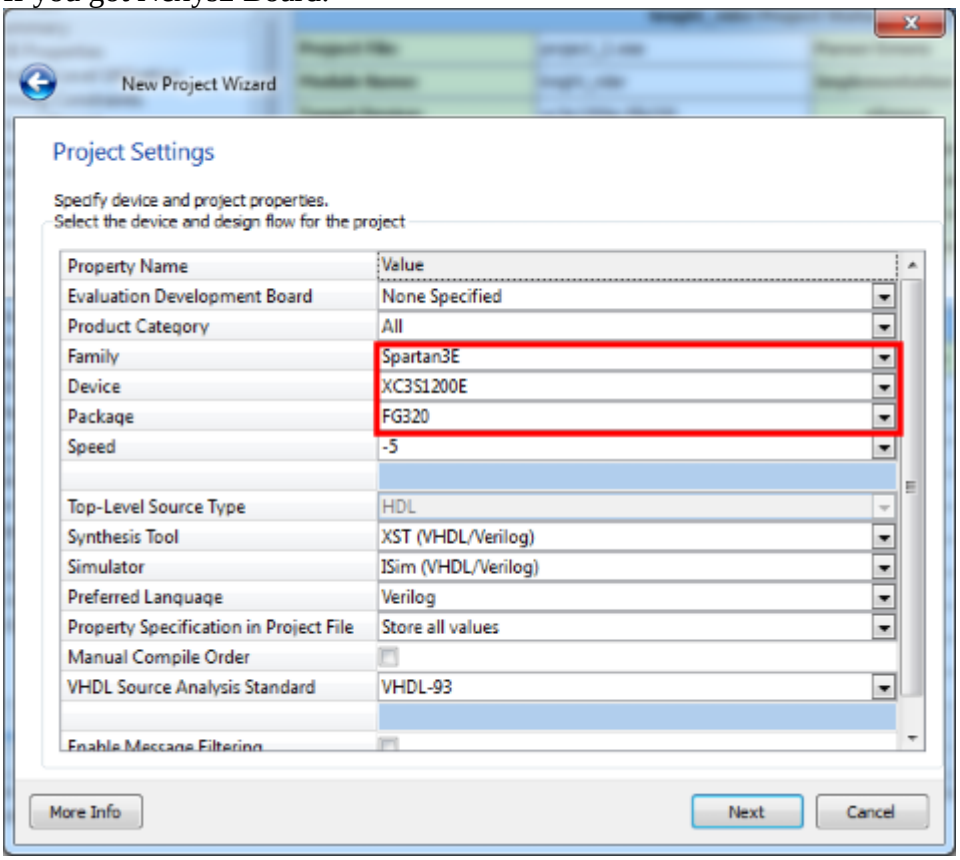
C:\Users\<USER>\Desktop\lab_5\rotating_banner

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

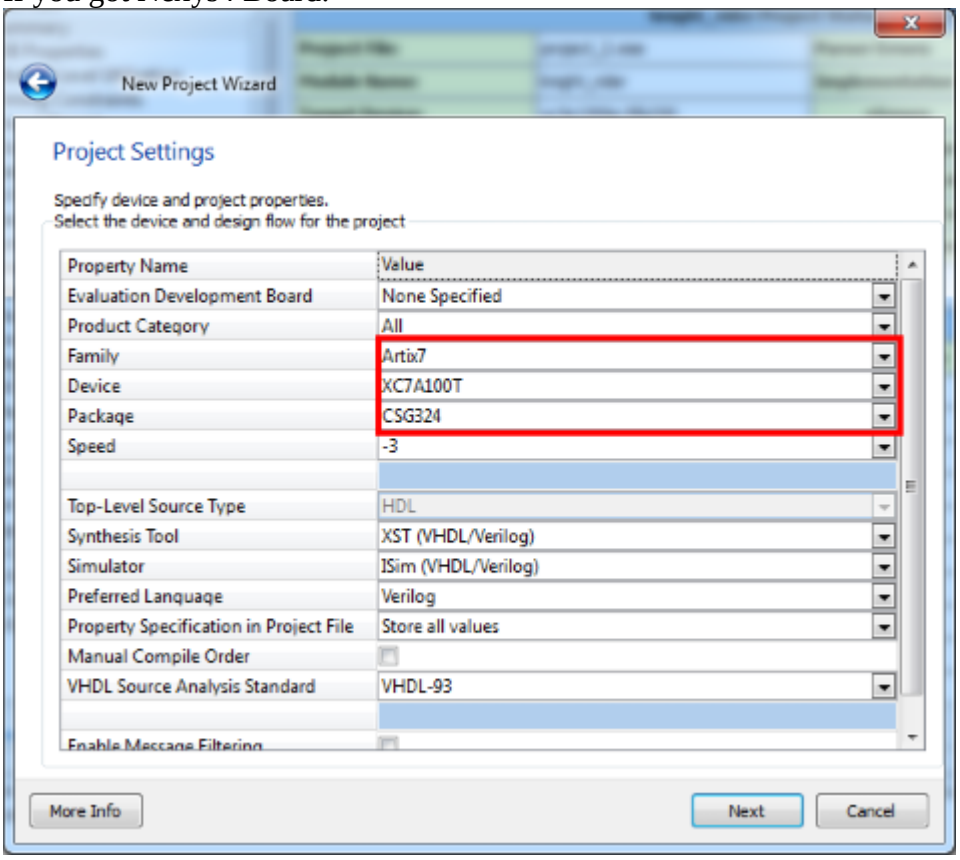
i. If you got Basys2 Board:



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.

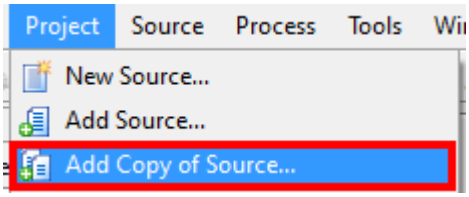
   d. Check Summary window to make sure that your module settings are correct. After that, click on Finish.

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, sseg, anode);

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

wire enable;
reg [4:0] addr;
wire [7:0] dout;

reg [3:0] state_current, state_next;

always@(posedge clk) begin
   if(rst) begin
      state_current <= 0;
   end
   else begin
      state_current <= state_next;
   end
end

always@(*) begin
   state_next = state_current;
   case(state_current)
      0: begin
         addr = 30;
         if(enable == 1'b1) begin
            state_next = 1;
         end
      end
      default: begin
         addr = 0;
      end
   endcase
end

assign sseg = dout;
assign anode = 4'b0000;

slowdown_unit inst_1(.clk(clk), .rst(rst), .enable(enable));

character_rom inst_2(
  .clka(clk), // input clka
  .addra(addr), // input [4 : 0] addra
  .douta(dout) // output [7 : 0] douta
);

endmodule
```

6. Download **slowdown_unit** module from COADSYS as **.v** files and add them into the design.

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

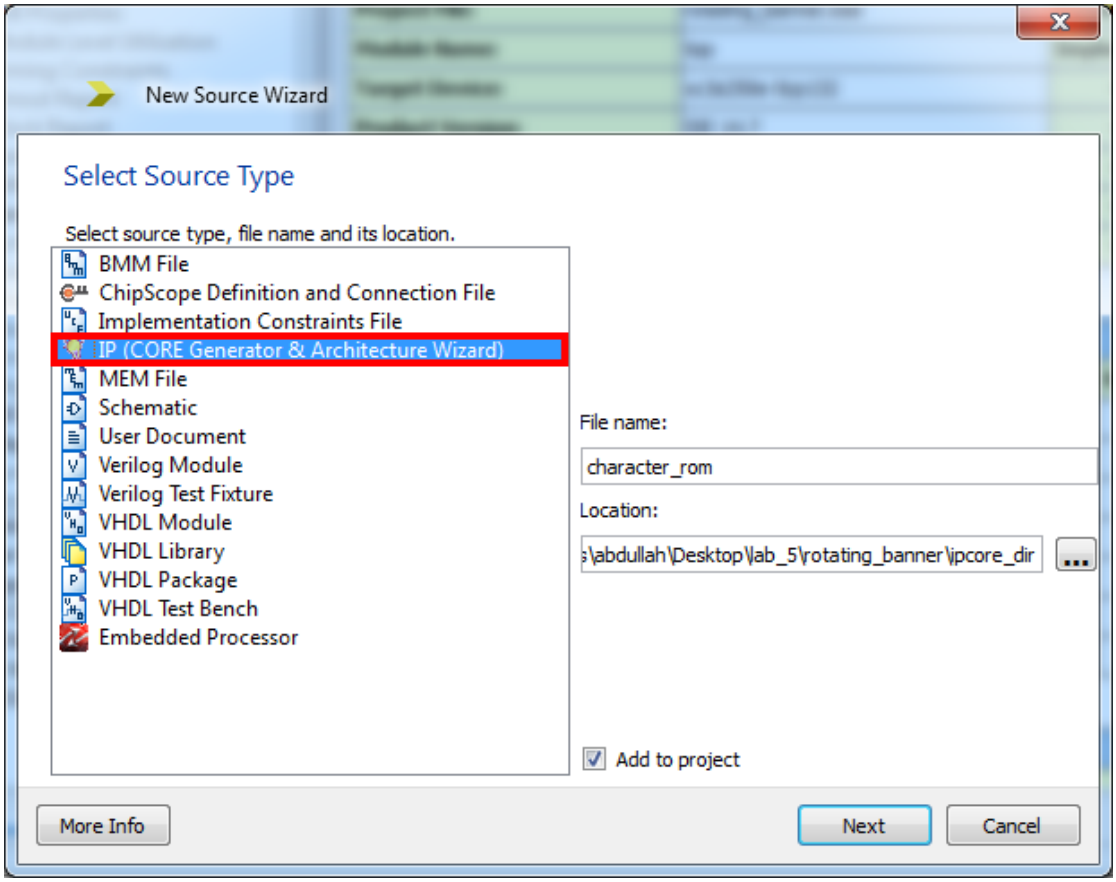


    b. Select the downloaded Verilog files and click on Open and then click on OK.

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

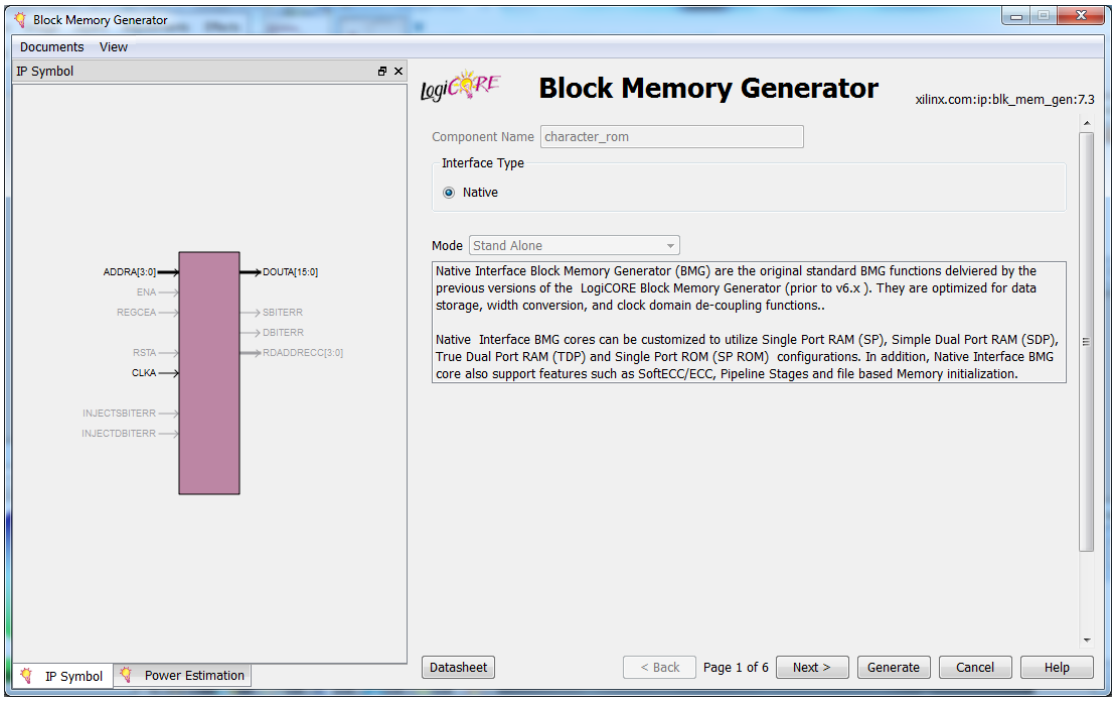
    a. Click on **New Source…** under Project menu.

    b. Select **IP (CORE Generator & Architecture Wizard)** and set File name as **character_rom** in Select Source Type window and then click on Next.

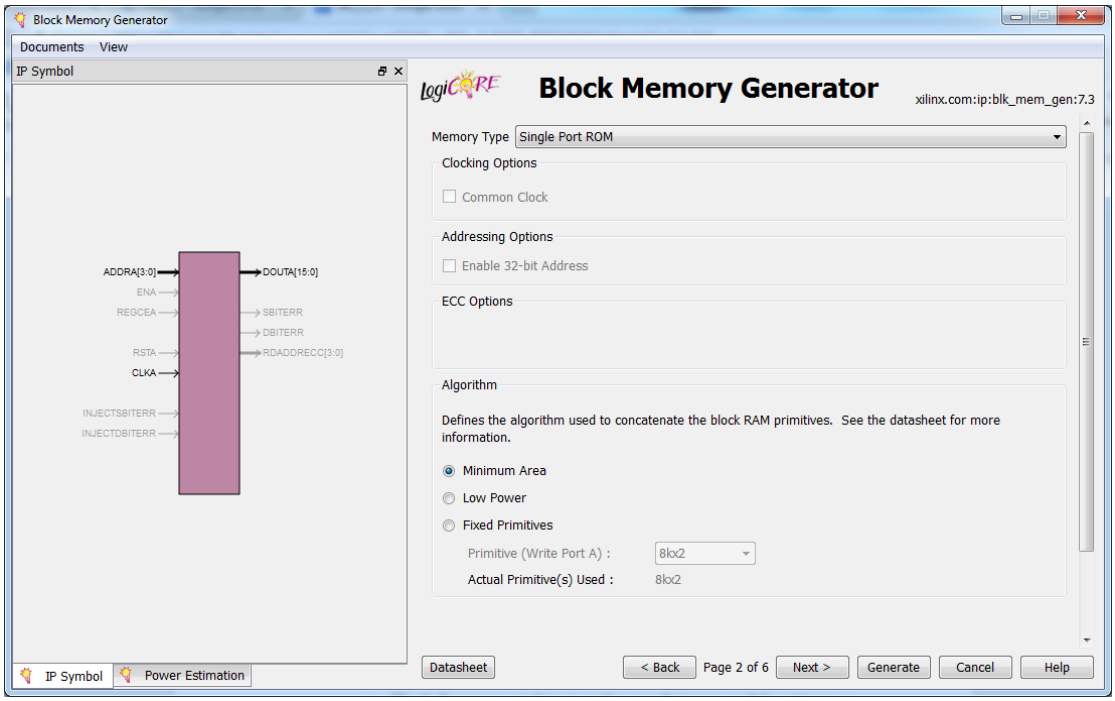


    c. Select **Block Memory Generator** under **RAMs & ROMs** of **Memories & Storage Elements**. Click on Next and then Finish.

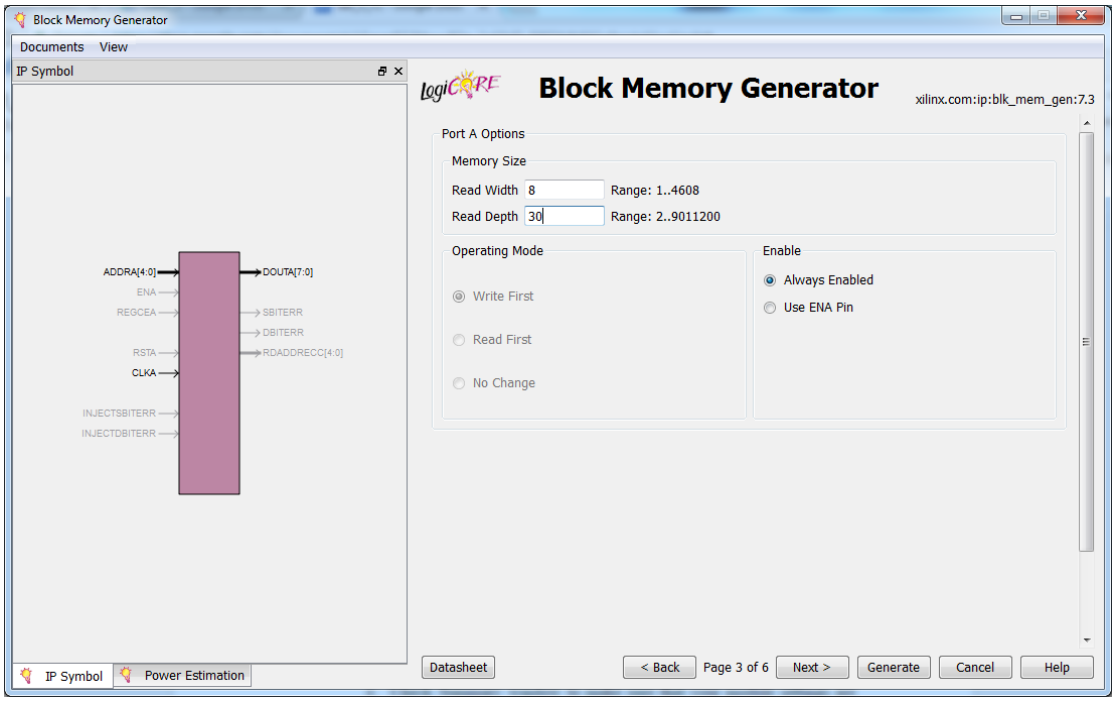
d. Skip the first page by clicking on Next.



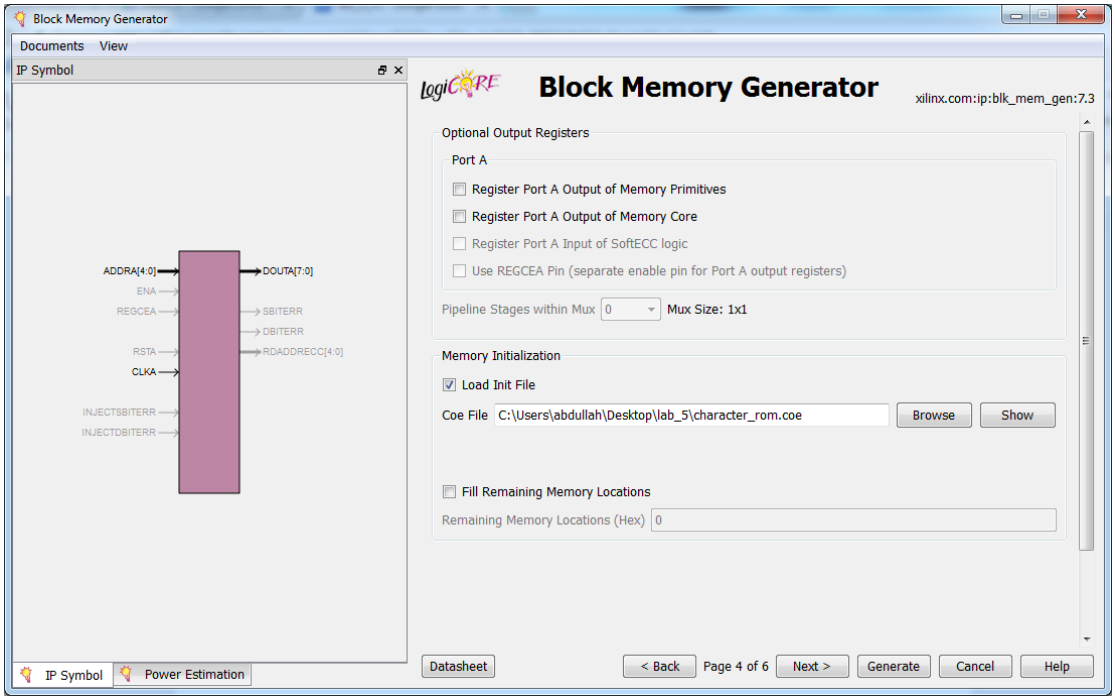
e. Select **Single Port ROM** as Memory Type on the second page and click on Next.

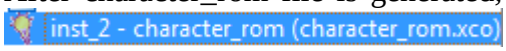


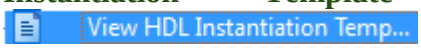
f. Set **Read Width** as **8** and **Read Depth** as **30** on the third page and then click on Next.



g. Check the **Loan Init File** option on the fourth page and set the name of **Coe File** as **character_rom.coe** (Download **character_rom.coe** file from COADSYS).Then click on Generate.



h. After character_rom file is generated, click on character_rom module 

and then **View HDL Instantiation Template** under **Core Generator** 

window.

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

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

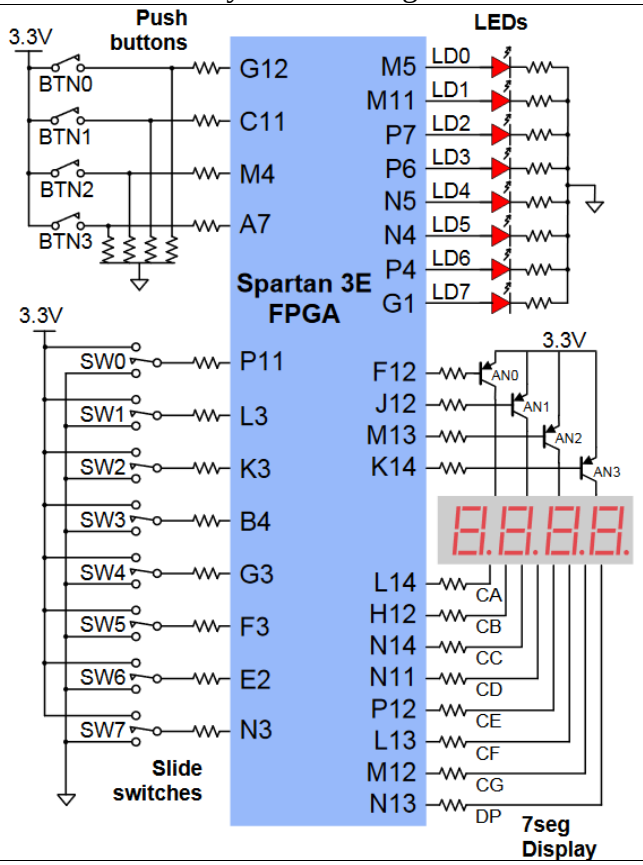
   j. Select **Implementation Constraints File** and set Filename as **top** in Select Source Type window. After that, click on Next.

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

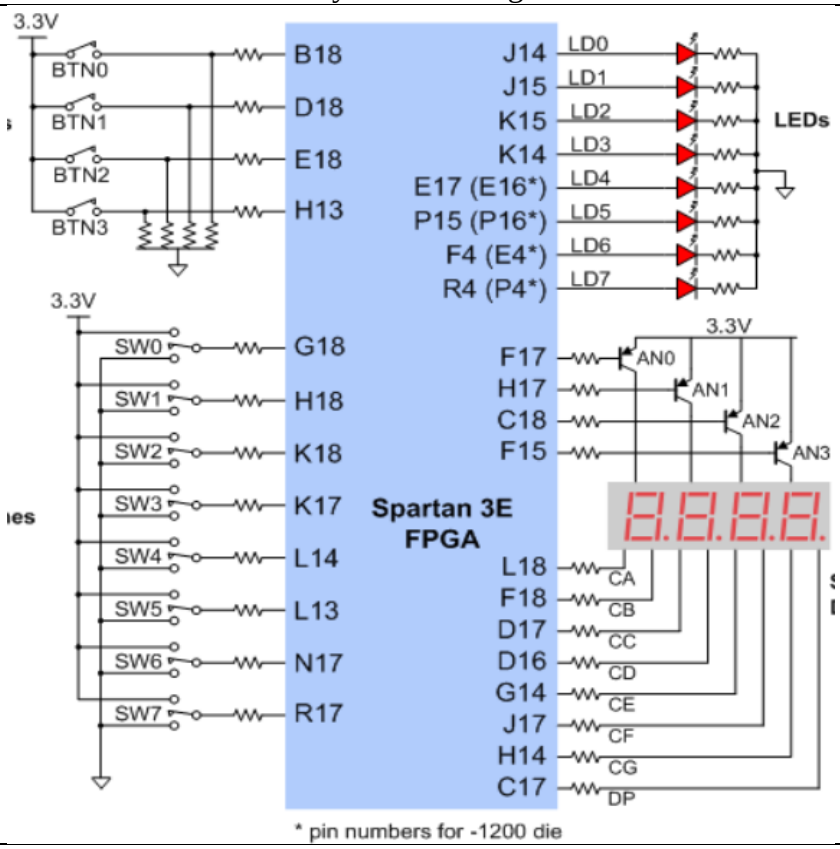
9. Add the following clock constraint to the **top.ucf** file and use the below figures to describe the user constraints of your top design.

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

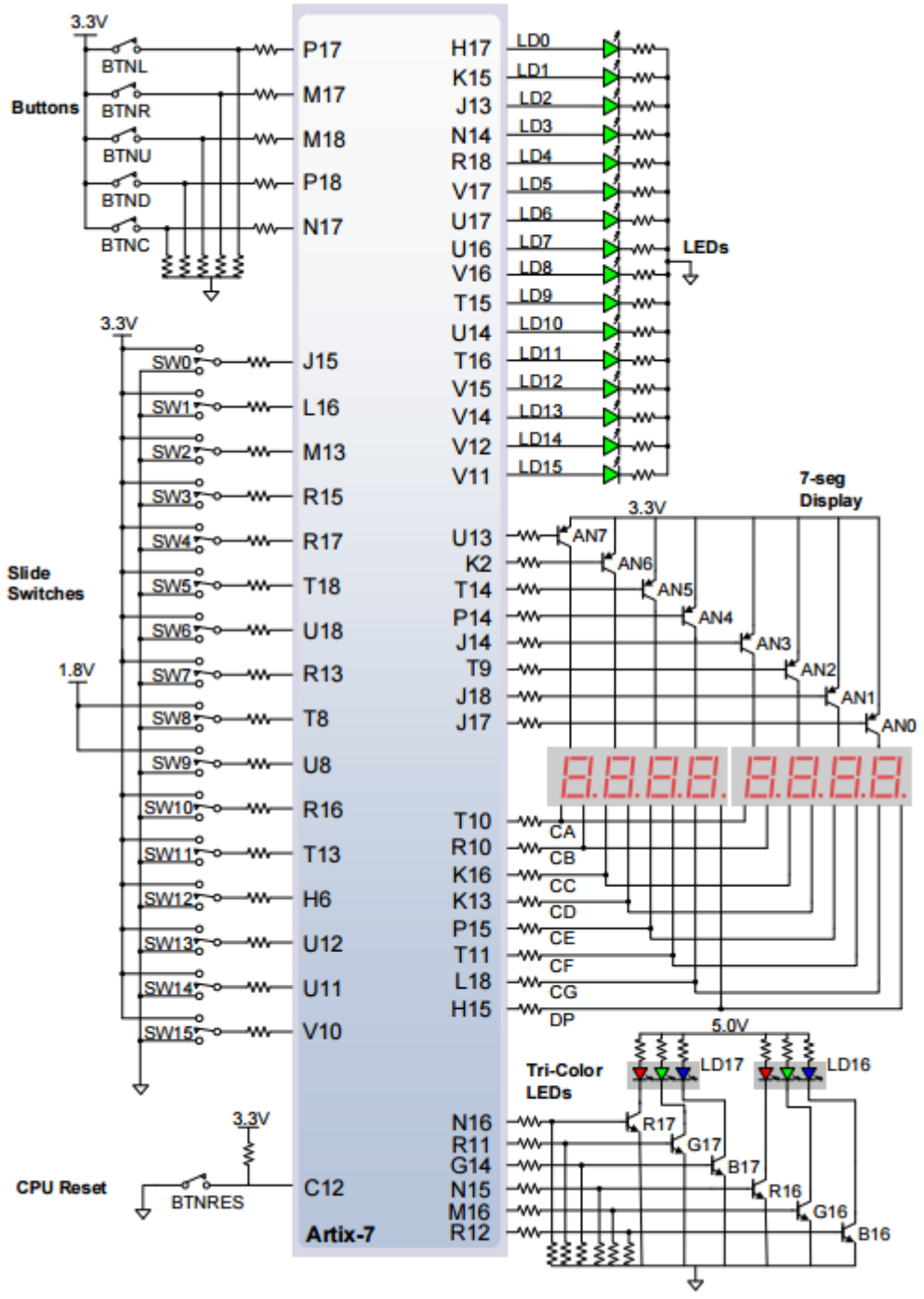
Basys2 Circuit Figure



Nexys2 Circuit Figure



Nexys4 Circuit Figure

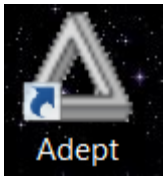


10. While on **Implementation** view, click on **Implementation** to start implementing the design.

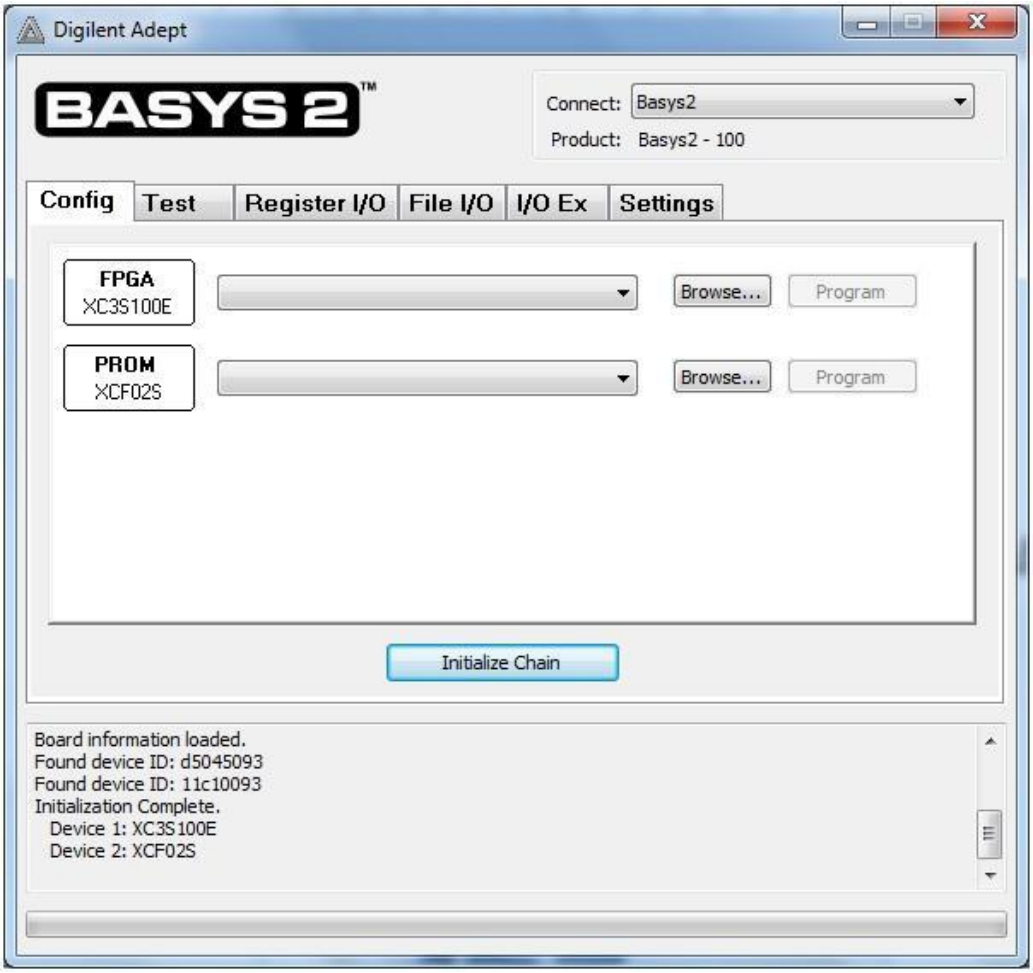
11. After **Implementation** is completed click on **Generate Programming File** to

generate "**.bit**" file.

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



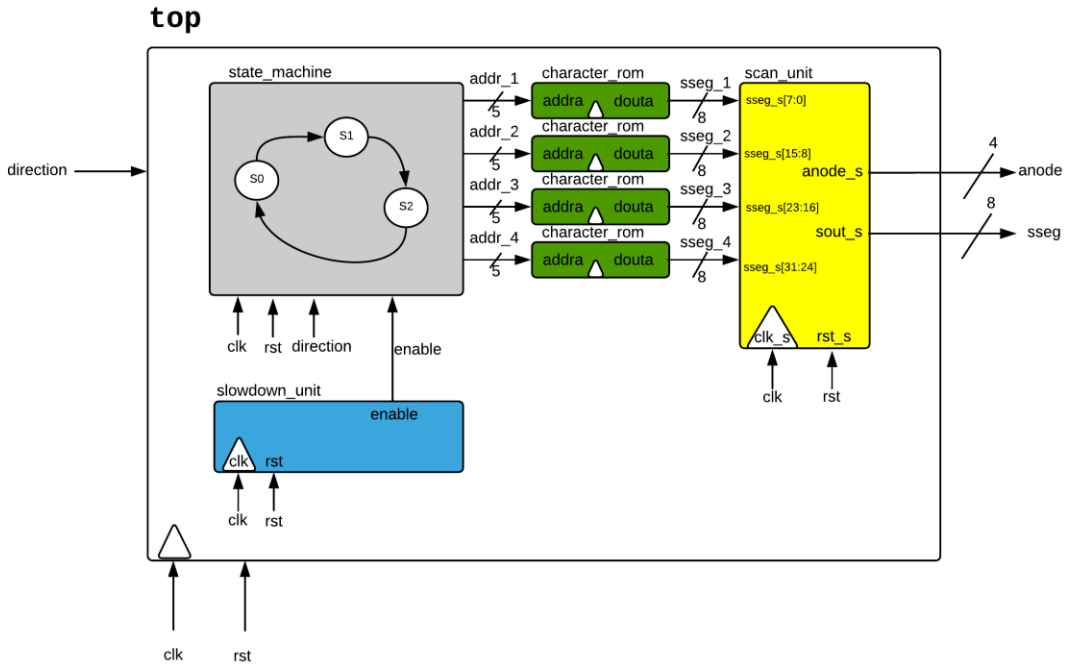
13. After invoking Adept Select "**top.bit**" from the project folder using browse and then press Program button.



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

Design #2: Rotating Seven-Segment Banner

Modify the existing design to implement a rotating seven-segment banner. Display **7tEPE** on seven-segment displays and add a switch to control the direction of state transitions.



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.

CHARACTER MAP of character_rom module

ADDRESS	DATA
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	b
12	C
13	d
14	E

ADDRESS	DATA
15	F
16	g
17	h
18	ı
19	j
20	L
21	n
22	o
23	P
24	r
25	S
26	t
27	u
28	y
29	Z