I. Creating a New Project

A **project** in ISE is a collection of all files necessary to create and download a design to the selected device. The project will be adapted to use our FPGA. In this tutorial a frequency divider circuit will be created and used to make a counter counts every second. Here are the steps for creating a new project:

1. **Launch** Xilinx ISE Project Navigator.
2. Select **File** menu then choose **New Project**.
3. In the **New Project Wizard** dialog box, type the desired location in the **Project Location** field, or browse to the directory under which you want to create your new project directory using the browse button next to the **Project Location** field.
4. Enter "*FrequencyDivider*" in the **Project Name** field. When you enter "*FrequencyDivider*" in the **Project Name** field, a folder named *FrequencyDivider* is automatically added in the directory path in the **Project Location** field.
5. Use the pull-down arrow to select **HDL** from the **Top-Level Source Type** field. Click in the field to access the pull-down list.
6. Click **Next**.
7. In the **New Project Wizard Device and Design Flow** dialog box, use the pull-down arrow to select the **Value** for each **Property Name**. Click in the field to access the pull-down list. Make sure the values are as follows:
    - Family: **Spartan3A and 3AN**
    - Device: **XC3S700A**
    - Package: **FG484**
    - Speed Grade: **-4**
    - Top-Level Module Type: **HDL**
    - Synthesis Tool: **XST**
    - Simulator: **ISim**
    - Preferred language:**Verilog**
8. When the table is complete, your project properties should look like Figure 1.
9. Click **Next**.
10. Click **Finish** to create the project. A summary of the project will be shown as in Figure 2. A new project called *FrequencyDivider* is created and is shown in the left side panel under implementation view.

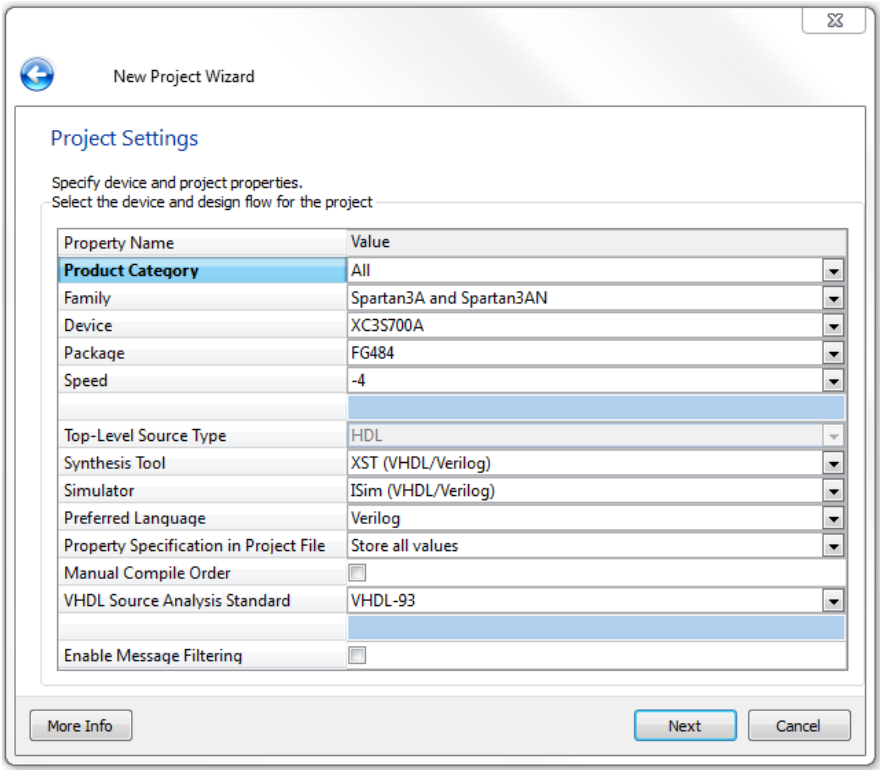

Figure 1

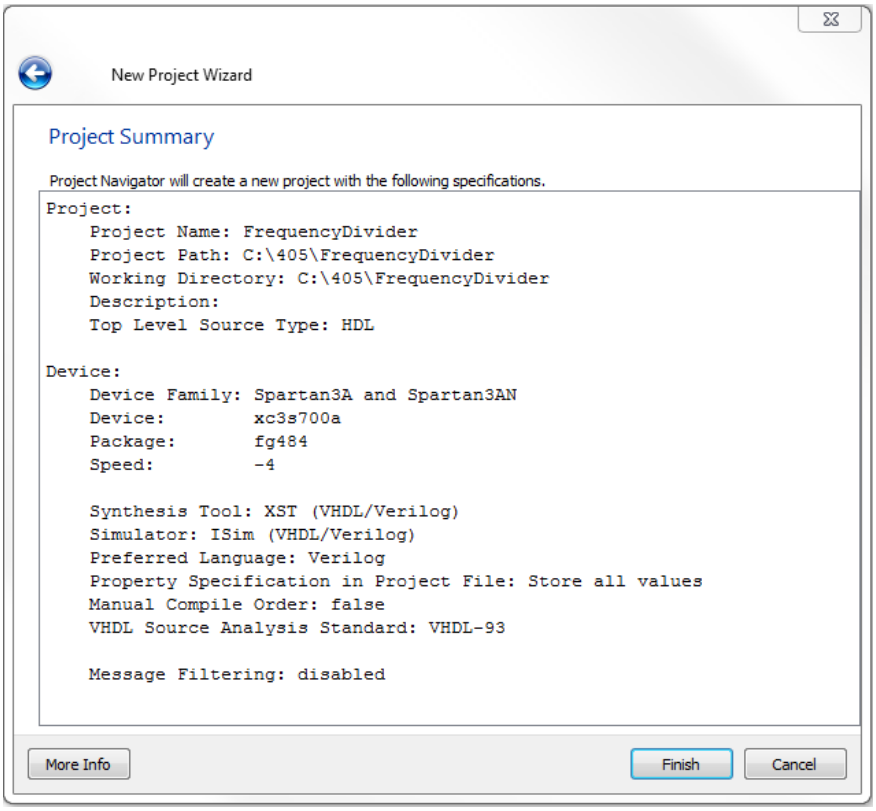

Figure 2

II. Creating a New Source

The project in ISE is composed of sources. A **source** can be schematic diagram, HDL module, implementation constraints file and others.

1. Right click on FrequencyDividerproject and choose New Source or select New Source from Project menu. Select Verilog Module from the box on the left and type in a file name for your project such as *"counter"* then click **Next**.
2. In this**optional** step, the circuit's input and output ports can be defined. A port can be a single bit or bus. For the counter add clk and rst as single bit inputs and count[3:0] as an output bus similar to Figure 3.

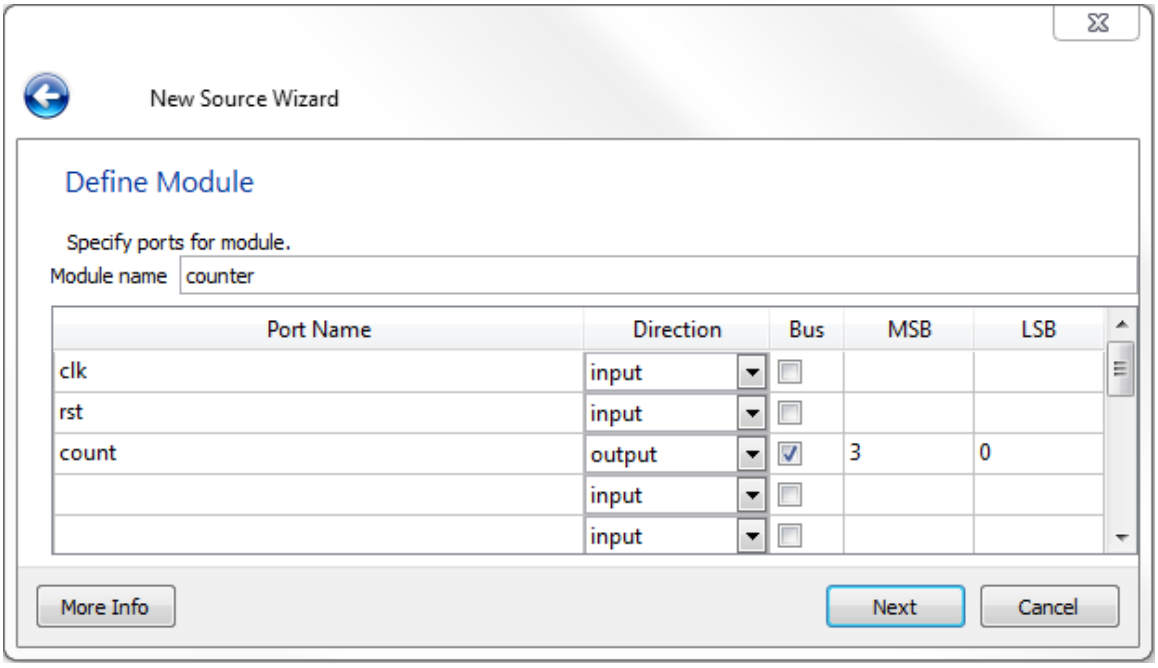

Figure 3

3. Click **Next**, a summary of the source is displayed. Click **Finish**.
4. ISE will create a file named *"counter.v"* in the project folder and add it to the project. It will also open that file for editing. The file will be listed under the FrequencyDivider project in the implementation view. Notice the symbol of the three squares to the left of the file name.

The three squares symbol refers to the **Top Module** of the project. The Top Module is the module that will be programmed in the FPGA. It can be changed by right clicking another source and selecting **Set as Top Module**. As *counter.v* is the first source in *FrequencyDivider*, it is automatically selected as Top Module.

5. Complete the implementation of the source file (*counter.v*) then go to the next section. Do not forget to save every now and then by using the keyboard shortcut ctrl+s or selecting **Save** from **File** menu or pressing the **Save** icon on the toolbar.

III. Behavioral Simulation

ISE provides an integrated simulator called **ISim** that allows simulations to be run from ISE Project Navigator. In this section, we will introduce the concept of simulation and how to verify the function of our circuit through behavioral simulation.

1. In the project navigator to the left, click on **Design** tab. Then click on your Verilog file. Change to simulation mode by selecting **Simulation** radio button as shown in Figure 4.

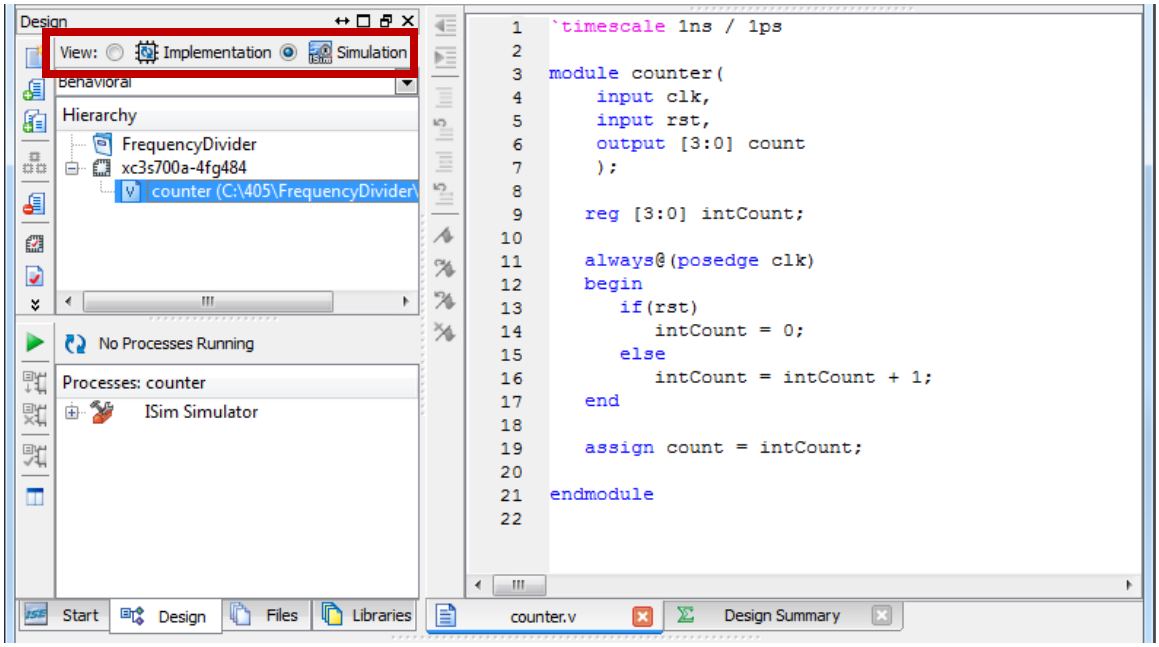

Figure 4

2. Press on the **+** mark in front of ISim Simulator to expand the list. Right click on the **Simulate Behavioral Model** and choose **Process Properties** to change simulation attributes. Uncheck the mark in front of **Run for Specified Time.** This will not limit the simulation for a specific run time. Press OK.
3. To run the simulation, double click on **Simulate Behavioral Model,** or right click and press **Run.**
4. ISE will launch ISim in a separate window. If it didn't, check the Errors log and correct the errors. Note that the simulation will fail to run if a current process of ISim is running, close any instance of ISim before running any new simulation.

5. **ISim** will launch automatically. The wave window displays the signals, buses and their waveforms. Note that there are four signal shown; clk, rst, count[3:0:] and intCount[3:0].
6. Right click on input clk in the objects window, and choose **Force Clock**. Add the following values:
    - Leading Edge Value: 0
    - Trailing Edge Value: 1
    - Period: 1 ns

Alternatively, you can write the isim command in the console window,

- **isim force add clk 0 -value 1 -time 500 ps -repeat 1 ns**

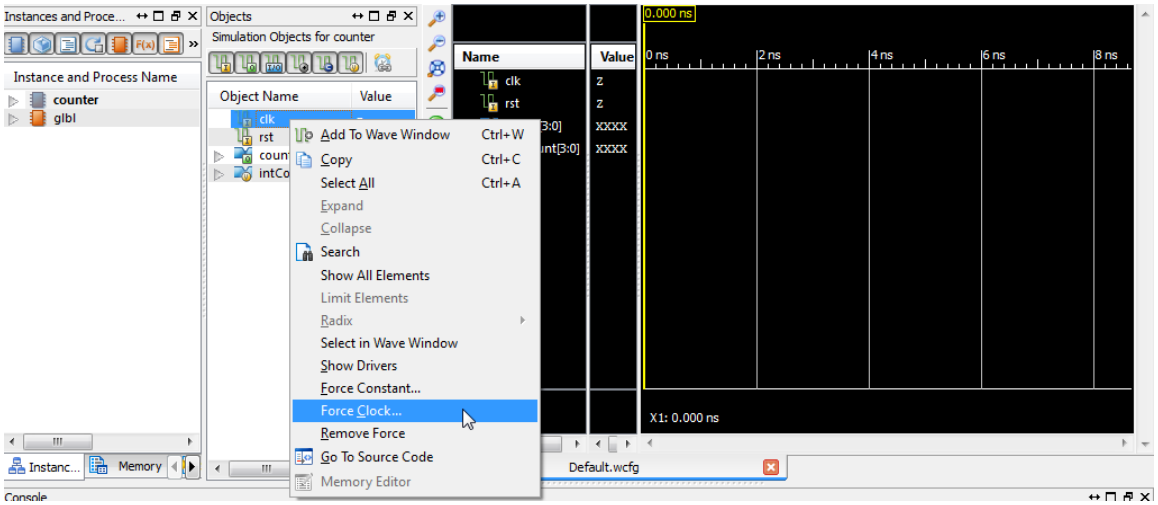

Figure 5

The force clock option (or its equivalentisim command) adds a rectangular wave with the given parameters to the selected signal. In this example, clk will start as 0 then after 0.5 ns will be 1 then after another 0.5 ns will be back to 0 and continues like that.

There is also **Force Constant** option that makes the signal fixed at a constant value. A series of force constant steps and simulation runs will lead to the required non-periodic signals. However using isim commands, the steps can be combined in a single command. The following three steps will make the rst input 1 for 3 clock periods (3ns) and then keep it 0 for the remaining time of the simulation.

7. Right click on input rst in the objects window and choose **Force Constant**. Add the following values:
    - Force to Value: 1
    - Starting at Time Offset: 0
8. Enter 3ns in the simulation time space in the toolbar, and then press **Run for the Time Specified in the Toolbar icon**.
9. Then right click again on input rst and choose **Force Constant**. Add the following values:

- Force to Value: 0
- Starting at Time Offset: 3ns

Alternatively, you can write the isim command in the console window,

- **isim force add rst1 -value 0 -time 3ns**

10. Enter 1us in the simulation time space in the toolbar, and then press **Run for the time specified in the toolbar** icon. Or type the isim command **run 1us** in the console window.
11. The simulator will show the behavior of the circuit according to the input signals, press **Zoom to Full View** in the toolbar to show the entire simulation period. You can Zoom in and Zoom out using the icons in the toolbar. The counter output should look like Figure 6.

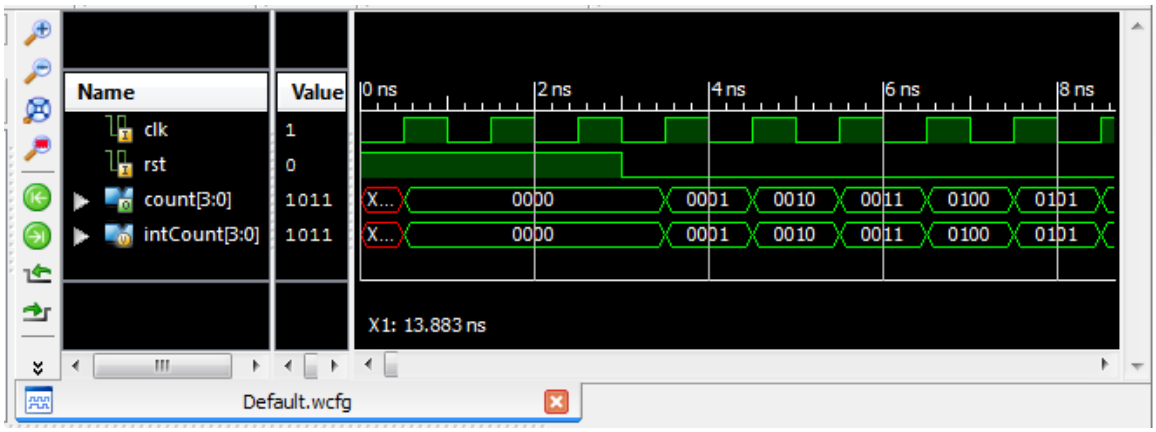

Figure 6

12. To restart the simulation, press on **Restart** icon on the toolbar. Note that restarting the simulation will also remove the force clock/constant values and isim commands. You have to apply force clock/constant or isim commands to the input objects before running the simulation again.
13. Close ISim and return to ISE Project Navigator.

IV. Hierarchical Design

To complete the frequency divider, a comparator is required. Also a module to combine the components will be developed. The FPGA board has a built in clock of frequency 50 MHz. The frequency divider should produce a frequency of 1 Hz meaning that it should be able to count to 50 million. That requires the counter and the comparator to be at least 26 bits in size.

1. Go back to **Design** view.
2. Add a new **Verilog Module** name it *comparator*. It has two inputs A[25:0] and B[25:0] and one output eq.
3. Complete the implementation of *comparator*.
4. Add a new **Verilog Module** name it *counterCE*. It has three inputs clk, rst and CE (count enable) and one output count[3:0]. Copy the code from *counter.v* then add the CE condition.

The CE functionality is added here to avoid using gated clock. For FPGA design it is advised that all clock pins should be connected to the main clock directly.

5. Select the *counter* source modify its size to be 26 bits.
6. Add a new **Verilog Module** name it *HzCounter*. It has two inputs clk and rst and one output count[3:0].
7. The *HzCounter* module will combine all previous modules: *counter*, *comparator* and *counterCE*. They should be connected according to Figure 7.

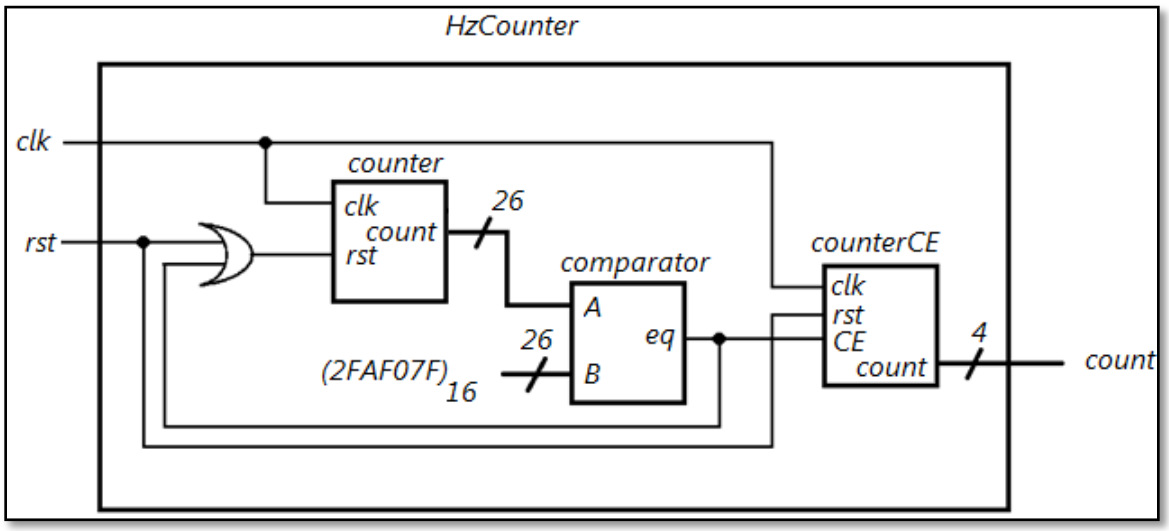

Figure 7

The *counter* starts counting from 0 until it reaches the constant ($[2FAF07F]_{16}$ = 50 x 10^6 – 1). The (-1) here is due to the synchronous reset of *counter*. When the value of the

constant is reached, *counterCE* (thereby *HzCounter*) will increment its count by 1 and *counter* will start counting from 0 again.
Note that ISE has detected that *counter* is contained in another module. Therefore, it changed the Top Module of the project to be *HzCounter*.

V. Package Pins Assignment

The objective here is to connect the inputs of the circuit (clk ad rst) to two input pins of the FPGA chip. Likewise, we want to connect the output of the circuit (count[3:0]) to four of the output pins of FPGA chip which are connected to the board's LEDs. This will allow us to test the circuit on the board. Ports on the FPGA board are connected as follows:

1. In the project navigator to the left, click on the **Design** tab. Then click on your Top Module (*HzCounter*). If needed, change to implementation view by selecting the **Implementation** radio button on top.
2. Click on **I/O Pin Planning (Plan Ahead) – Post-Synthesis** under **User Constraints**. This will launch PlanAhead. A popup message appears indicating that the project does not have an Implementation Constraints File (UCF). It will prompt to create a new one, choose Yes.
3. In PlanAhead, select the **I/O Ports** tab in the left panel, expand Scalar ports. You will find a list of the single-bit I/Os of your schematic design, i.e., **clk** and **rst**. Ports that are buses are listed separately and should be expanded to assign each member to a different site.
4. **Decide** the site number of the switch(es), pushbutton(s) and LED(s) for inputs and outputs on your board (represented by a small code between bracket on the board). In this tutorial, we are using **SOUTH**, **LD3**, **LD2**, **LD1** and **LD0** which have the sites **T15**, **U19**, **U20**, **T19**and **R20** respectively.
5. The **clk** input of the circuit will be connected to the main clock of the board which has site **E12**.
6. Click the column under **Site** on front of the I/O port and choose the corresponding site from the drop down list. Do it for **clk**, **rst**, **count[3]**, **count[2]**, **count[1] and count[0]**. Alternatively, you can write the site directly.

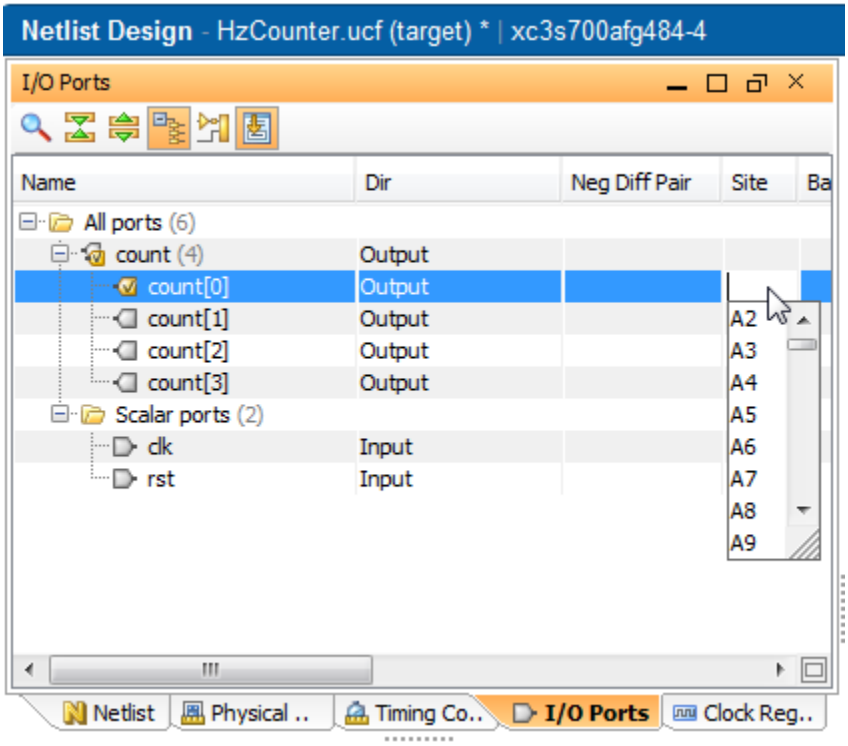

Figure 8

7. Once the pins are locked down, select **File** menu then choose **Save Design**. The changes made in PlanAhead are saved in the *HzCounter.ucf* file in your current working directory.
8. **Exit** PlanAhead.

Open the file *HzCounter.ucf* which is now listed under *HzCounter*. As you can see, the syntax of the file is simple. You can perform package pins assignment without using PlanAhead. If you do so, make sure that your file is listed among other sources in the project and have the same name as your Top Module with extension *.ucf*.

VI. Design Implementation

The objective here is to generate the configuration bits-file which, when downloaded to the FPGA chip, configures it to implement our specific design. Note: For more information about implementing a design, see ISE Help. Select **Help**, **ISE Help Contents**, expand either the FPGA or CPLD hierarchy in the left pane and expand the **Implementing Design** hierarchy.

1. **Click** on your Top Module source in the project navigator.
2. Double click on **Generate Programming File** in the **Processes** window. This runs all processes and creates the configuration bits file of this design. Be patient – this takes a while!
3. The BitGen program creates the *HzCounter.bit* bitstream file. The bitstream file contains the actual configuration data.
4. A check on the Processes for a source denotes a process that was run successfully. An exclamation sign indicates that the process was run and that there is a warning for the process. A cross indicated that the process failed with error. More information about warnings and errors can be obtained in the **Console** window.

VII. Programming the FPGA Board

The objective of this section is to download the design configuration bits file to the FPGA board. The detailed procedure is as follows:

1. Turn on your Xilnix Spartan-3A Board and make sure that the board cable is properly connected to the PC.
2. With your Top Module selected double click on **Configure Target Device**. Dismiss the popup message if it appears saying that no iMPACT project file exists.
3. ISE will automatically run **iMPACT.** Double click on **Boundary Scan** in the iMPACT Flows box on the left.
4. Right click in the middle of the white window to initialize new JTAG chain. Choose **Initialize Chain**. This will create and show a device chain. Click **Yes** if it asks to continue and assign configuration files.

5. Choose the *HzCounter.bit* file you generated in the proper directory. Click **Open.** Click No to attach SPI or BPI PROM.
6. A window will appear "**Add PROM File**", choose **Cancel.**
7. Another window will open, this time click **Bypass.** Then click Ok.
8. A window will appear "**Device Programming Properties**", click Ok.
9. Right-click on the first chip, named xc3s700a, and choose **Program.** When the program operation completes, a blue message with "**Program Succeeded**" appears.
10. Congratulation, your design is programmed in the Xilnix board. Go and have fun with it. You should be able to verify the design of the circuit using the pushbutton and LEDs.
11. When you close the program, it will ask you to "Save current project before exiting ISE IMPACT". Click No**.**
12. Remember to **turn off** your Xilnix board after you finish.

VIII. Saving the Design to the Board's PROM

By default, the configuration is cleared when the power goes off. This section will show how to save the design to the PROM so that the design stays in the board after the power is off. You might have seen that the board has an interesting demo installed on it. This demo will be replaced by the design saved to the PROM. The section will also show how to reset the board to its original state (i.e. download the demo on the borad).

1. Select the project's Top Module and then click the **+** to the left of **Configure Target Device**.
2. Double click on **Generate PROM/ACE File**. Dismiss the popup message if it appears saying that no iMPACT project file exists.
3. iMpact will open. Double click on **Create PROM File (PROM File Formatter)**.
4. A new window opens, it includes three steps to start generating the PROM file.
5. In Step 1. Select Storage Target, select **Configure Single FPGA** under **SPI Flash**. Then click the arrow pointing right to go to the next step.
6. In Step 2. Add Storage Device(s), check the checkbox at the bottom saying **Auto Select PROM**. You will be transferred to the next step automatically.
7. In Step 3. Enter Data, modify the **Output File Name** and choose where to store it. Keep the other options without modification (Checksum Fill Value FF and File Format MCS). The window now should look like Figure 9.

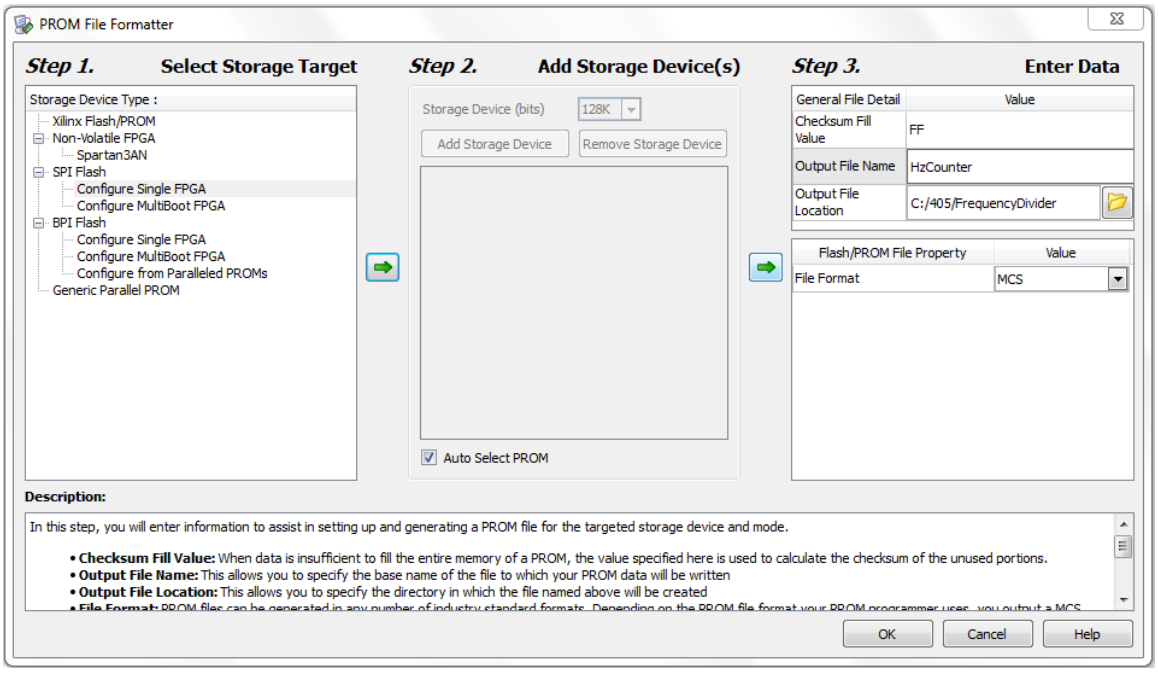

Figure 9

8. Click Ok. The PROM File Formatter window will be dismissed. A popup message titled Add Device will appear. Click Ok.
9. Select the bitstream file that have the same name as your Top Module and extention .bit (*HzCounter.bit*).
10. A popup message titled also Add Device will appear, choose No. Another popup will appear saying You have completed the device entry. Click Ok.
11. From iMPACT Processes box to the left, double click **Generate File…**. A blue message saying Generate Succeded will appear. If not check for errors.
12. Make sure the board is ON and properly connected to the PC. Double click on **Boundary Scan** in the iMPACT Flows box on the left.
13. Right click in the middle of the white window and choose **Initialize Chain**. Click **No** if it asks to continue and assign configuration files.
14. Right click inside the dashed blue rectangle above the figure of the first chip. Select the only option **Add SPI/BPI Flash**. If there is a chip-like image instead, choose **Assign Ne Configuration File…**.

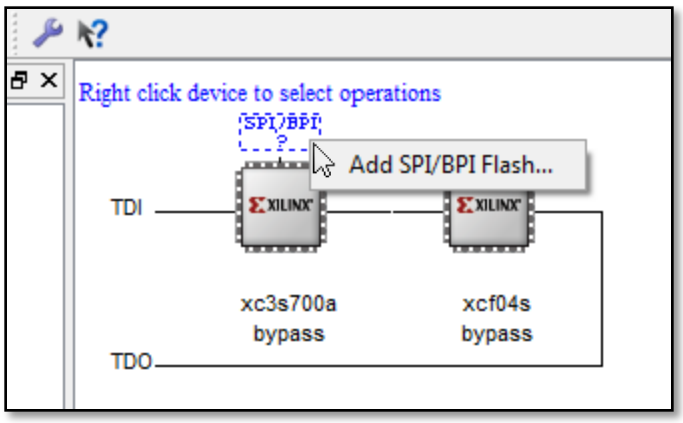

Figure 10

15. Select the .mcs file that was created at setp 11. Keep **SPI Flash** in the first dropdown menu and change the second to **AT45DB161D**. This is the PROM model for the board. Click Ok.
16. The dashed blue rectangle will change into a chip-like image. Right click on it and choose **Program**. Click Ok for the window that appears next.
17. PROM programming take much more time than normal programming, so please be patient.
18. After programming succeeds, the design is saved to the board's PROM. If you turn the power off an then on again, the design will run automatically.

The demo will be erased by now. To restore the demo, go to the website http://www.xilinx.com/products/boards/s3astarter/reference_designs.htm. This website has some other sample design, feel free to try them. Download **Demo Design for the Starter Kit Board**. Extract the compressed folder (*s3ask_demo.zip*) which contains several folders. The PROM files are stored in *~\s3ask_demo\mcs_files\mcs*. Start iMPACT and repeate steps 12 to 16. You should use the mcs file that corresponds to the board's PROM moodel i.e. *at45db161.mcs*. After successful programming, the board should be in its original state.