



CADSTAR

CADSTAR FPGA

Complete FPGA-PCB Design Flow Tutorial

created by:



**Table of Contents**

Introduction.....	2
Creating a New Design.....	2
Choosing Workspace and Design Names	2
Creating First Design Files	5
Creating Top-Level.....	12
Compiling and Simulating	14
Basic Debugging	17
Using Existing design for FPGA-PCB design flow	17
Synthesis with Xilinx ISE/WebPack 9.2 XST VHDL/Verilog	20
Post-Synthesis Simulation.....	24
Generating I/O Pin Constraints File for Implementation	25
Implementation with Xilinx ISE and Providing Implementation Constraints.....	28
Implementation with Xilinx ISE/WebPack in GUI Mode	31
Timing Simulation	33
Export of Implementation Reports Constraints file to CADSTAR PCB Tool.....	35
CADSTAR Schematics Block Creation.....	33
CADSTAR Parts Library Editor	37
CADSTAR Schematic Design	45
CADSTAR PCB Design.....	48
CADSTAR P.R. Editor XR (Place and Route Editor)	51
Import of SWAP Pin file from CADSTAR to Active-HDL.....	53
Conclusion.....	60



Introduction

This Tutorial was created to help you become familiar with the basic features of Active-HDL CADSTAR edition lite in the shortest possible time. No prior knowledge of HDL simulation tools is required, but elementary knowledge of VHDL will be helpful.



If you want to refresh your VHDL, you are welcome to use our Interactive VHDL Tutorial: just go to the **Help** menu, and then select the **Interactive VHDL Tutorial** option. The same tutorial is also accessible directly from the installation CD.

While reading this Guide, you will be able to create, compile, simulate and debug a simple, but fully functional design of a Counter. First section of the tutorial will use **Xilinx ISE/WebPack 9.2 XST VHDL/Verilog** and **Xilinx ISE/WebPack 9.2** as **Synthesis** and **Implementation** tools respectively.

During the installation of Active-HDL CADSTAR edition lite, user has the option to choose and install pre-compiled ready to use FPGA vendor libraries inside Active-HDL. It is highly recommended that the user installs the FPGA vendor libraries that he will need to use in his design. There is also an option to update to the latest version of FPGA vendor libraries from within Active-HDL CADSTAR Edition Lite by choosing Help-> Aldec on the Web ->Download Active-HDL Updates. If user does not install the vendor libraries during Active-HDL installation, then he has to bring the vendor libraries and recompile them inside Active-HDL.

The application note <http://support.aldec.com/KnowledgeBase/Article.aspx?aid=000033> describes the steps needed to recompile the libraries.

Note:[For the purpose of this tutorial **Xilinx ISE Webpack** Synthesis and Implementation tools will be used. They are free tools from Xilinx and can be downloaded from

http://www.xilinx.com/ise/logic_design_prod/webpack.htm Similarly, free vendor tools from Altera, Actel, Lattice and Quicklogic can also be downloaded from their respective websites and can be set up in Active-HDL the same way as Xilinx ISE Webpack.]

The first section of the Guide covers VHDL and Verilog entry, utilization of Language Assistant utility of Active-HDL while creating your design, interactive simulation and standard debugging. The second section emphasizes on using existing sample design for Synthesis, Implementation, I/O FPGA Pin assignment and export and import of I/O pin files to and from CADSTAR PCB tool.

Creating a New Design

Starting from the Opening Screen

If you double-click the Active-HDL icon on your desktop, you should see the splash-screen for a while, then the **Getting Started** window with the Active-HDL GUI in the background.

Right in the middle of that window you should notice a **Create new workspace** radio button. Select this button, and then click **OK**.

Starting from the GUI

If Active-HDL is already open (e.g. you were playing with sample designs) go to the **File** menu, select **New**, then **Workspace**.

Choosing Workspace and Design Names

1. No matter if you started from the GUI or the Getting Started window, you should see a **New Workspace** window. Type the name of the workspace (**MY_WORKSPACE**) in the first field; keep the **Add New Design to Workspace** checkbox selected and click **OK**.



- The **New Design Wizard** window appears. In this new window, select **create an empty design with design flow** and click **next>**.

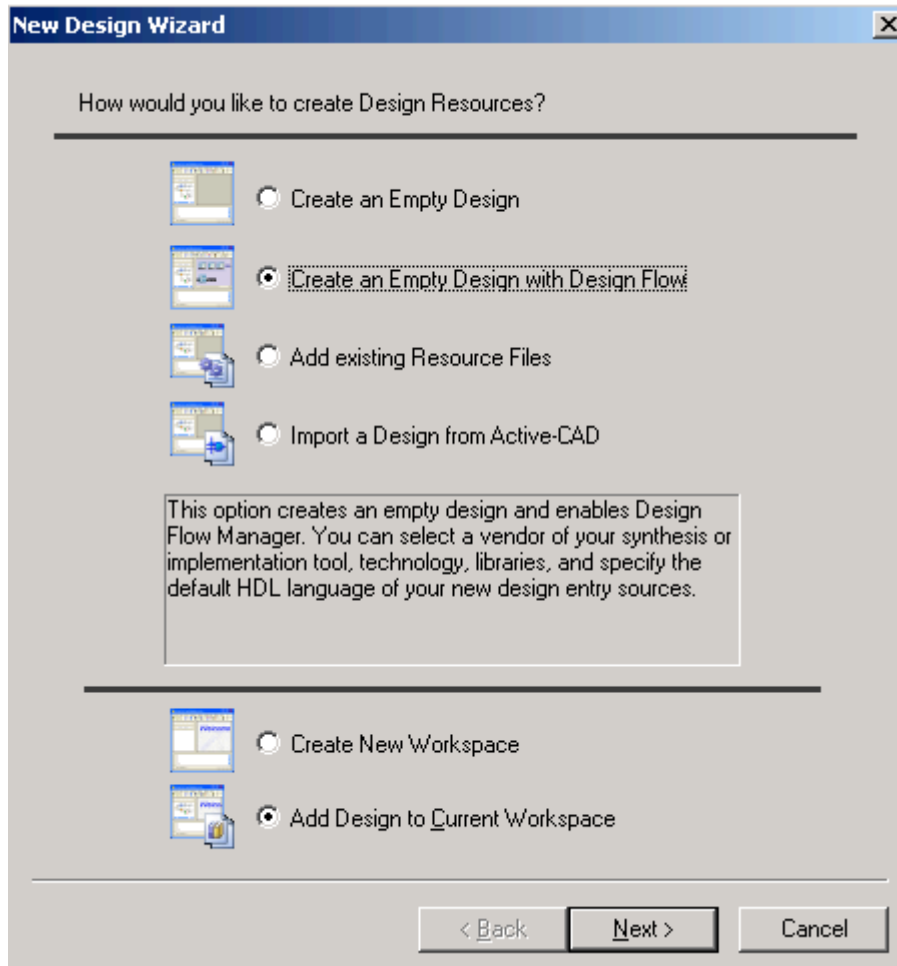


Figure 1: New Design Wizard.

- Click on the **Flow Setting** tab in the new window. Click on the **select** tab of the HDL synthesis tool and select the **Xilinx ISE/WebPack 9.2 XST VHDL/Verilog** from the option. Now click on the **select** tab of the implementation tool. Select **Xilinx ISE/WebPack 9.2** from the option. Now scroll down the window saying **family** and select **virtex**. Click OK.



You can also select your choice [for example Precision RTL] of synthesis tools and implementation tool [for example Actel Designer] in the flow setting window the same way as Xilinx ISE/ Webpack 9.2

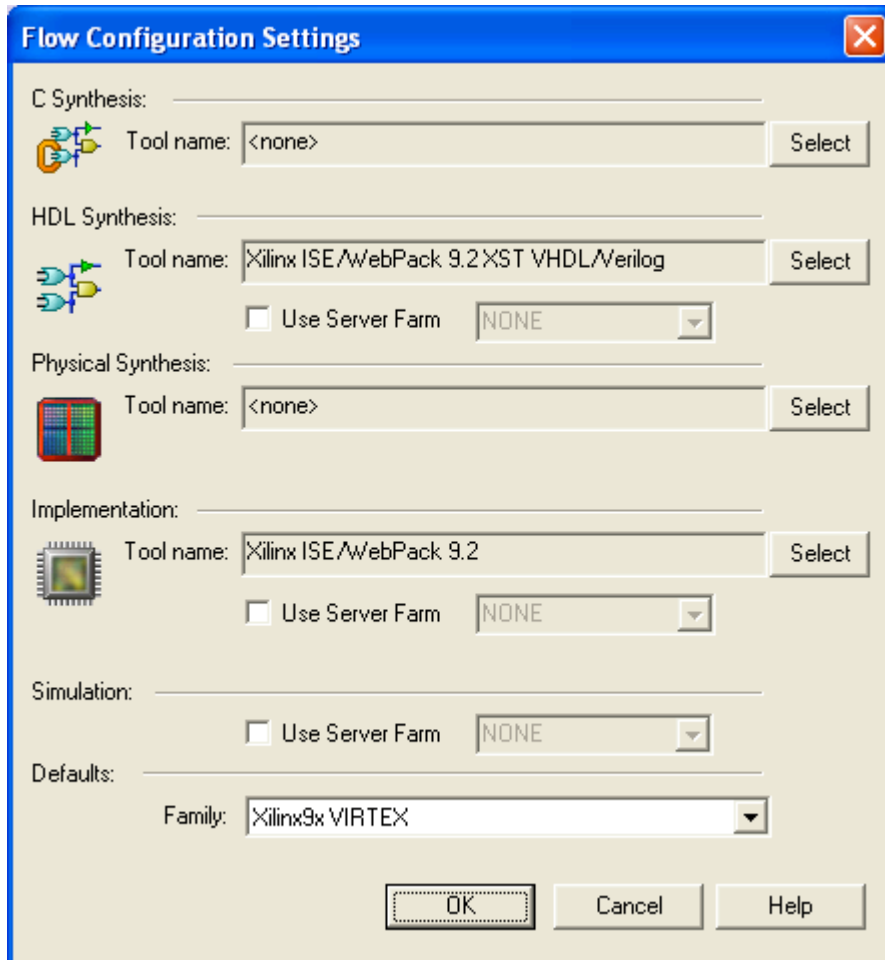


Figure 2: Flow Settings

4. Now you are back to the same window again but with enabled flow setting. Click next now.
5. Type the name of the design **Counter** in the new window and click next and click finish.
6. Now what you see in the design flow manager area is a complete FPGA flow starting from design entry to timing simulation. In addition you will see option for PCB Interface for Pin file Export/Import to and from CADSTAR PCB tool.

Your new workspace and design are ready to use!



To learn more about workspaces and designs, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Using Active-HDL / Workspaces and Designs**.



Creating First Design Files

Creating Cnt_4b.Vhd

The first source file we are going to create describes a Counter. The interface of the Counter can be quickly created using the **New Source File Wizard** and **Language Assistant**

1. From the main menu, Go to File->New ->VHDL Source. Click “Next” in the New Source File Wizard.
2. Type cnt_4b as the name of source file in the first box as shown in the Figure 3 . Click Next.

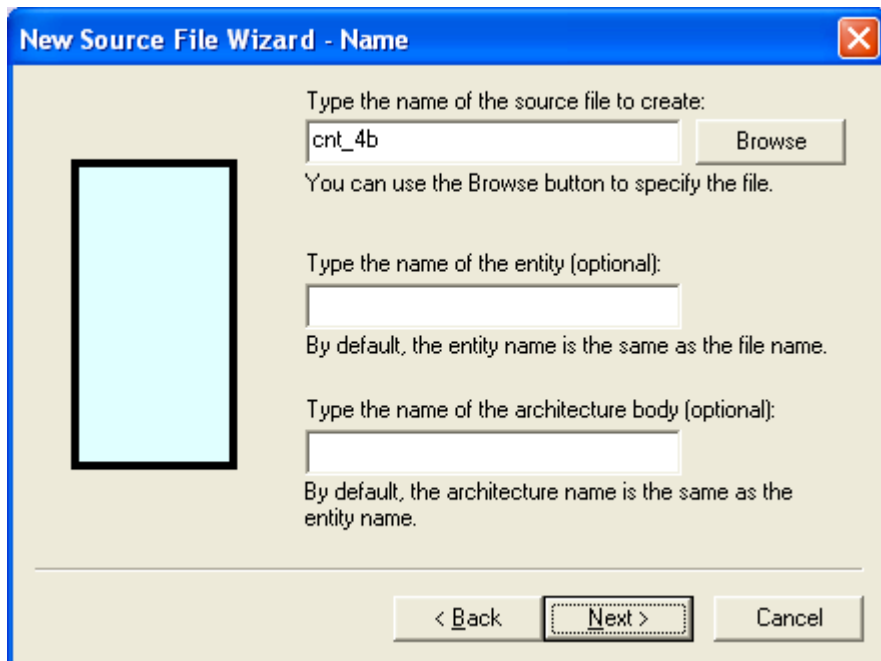


Figure 3: New Source File Wizard

3. In the next window, you will add ports to your source file. Click on “New” and type “CLK” under the “Name” tab. Make sure the Port direction is “in” as shown in Figure 4.

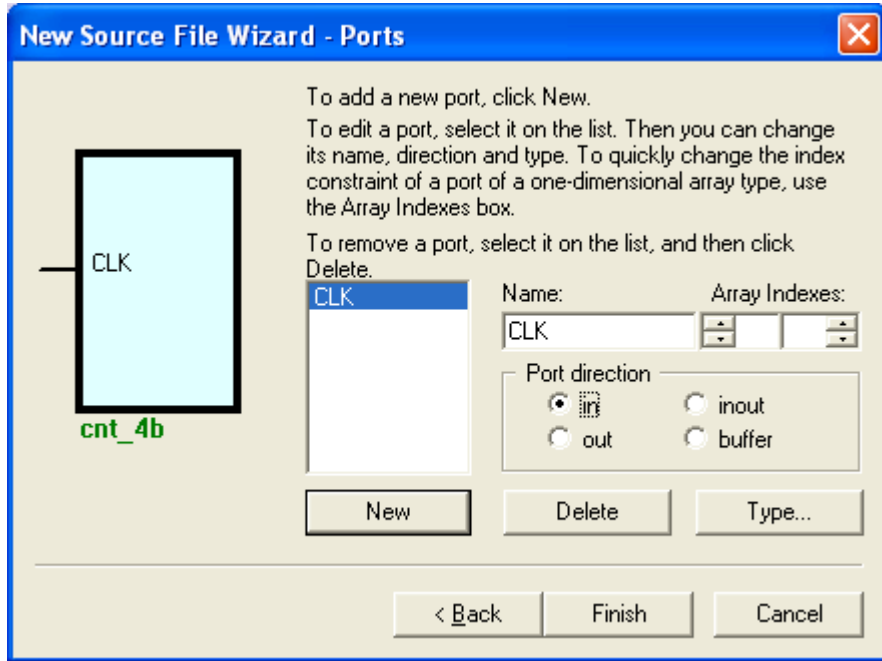


Figure 4: Ports

- Click on the "New" tab again and declare "RESET" with Port direction as "in". Similarly declare "ENABLE" as "in", FULL as "out". Finally declare "Q" as "out" and type "3" in first column of "Array Indexes" so that width of output port Q is 4-bit. Once you declare the ports the "New Source File Wizard" should look like in Figure 5. Click Finish.

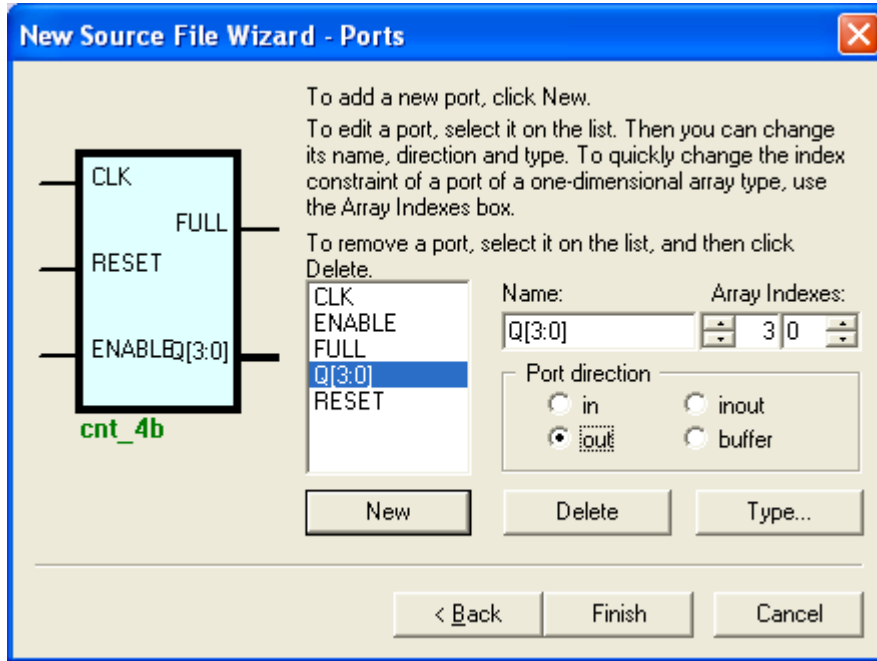


Figure 5: Ports

You will see cnt_4b.vhd in your design in the Design Browser. And it is almost ready except for the architecture body. You can make use of “Language Assistant” to fill the body of the architecture.

With the cursor placed in the body of the architecture, click on the icon for Language Assistant in the tool bar above. You can also select “Language Assistant” by going to Tools-> Language Assistant option from the main menu.

With the HDL Language selected as VHDL, Expand the “Tutorial” and click in “Counter” to find readily available process for our Counter, cnt_4b.vhd. Click on Use template icon and the process will be automatically inserted in the body of the architecture of cnt_4b.vhd.

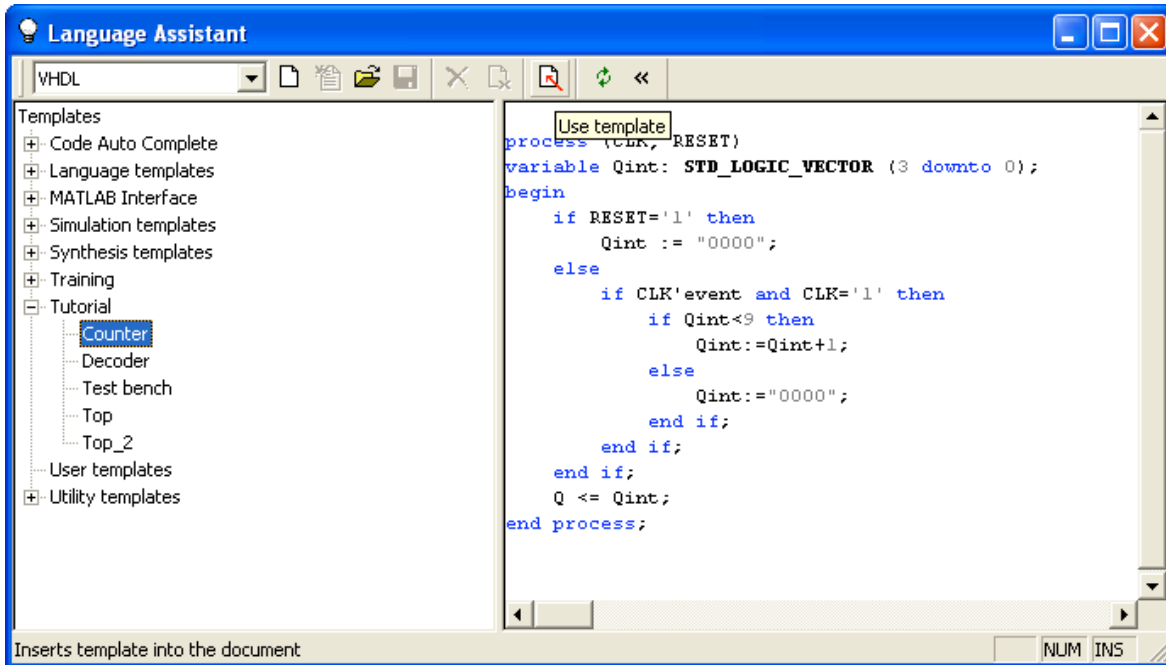


Figure 6: Language Assistant



To learn more about the Language Assistant, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Active-HDL Tools / Language Assistant**.

The cnt_4b.vhd is almost ready. Declare the ieee library clause for unsigned arithmetic, **use IEEE.STD_LOGIC_UNSIGNED.all;** in the Library declaration part at the top.



The source code for cnt_4b will look like as shown in Figure 7

```

24
25  library IEEE;
26  use IEEE.STD_LOGIC_1164.all;
27  use IEEE.STD_LOGIC_UNSIGNED.all;
28
29  entity cnt_4b is
30      port(
31          CLK : in STD_LOGIC;
32          RESET : in STD_LOGIC;
33          ENABLE : in STD_LOGIC;
34          FULL : out STD_LOGIC;
35          Q : out STD_LOGIC_VECTOR(3 downto 0)
36      );
37  end cnt_4b;
38
39  --}) End of automatically maintained section
40
41  architecture cnt_4b of cnt_4b is
42  begin
43
44
45  process (CLK, RESET)
46  variable Qint: STD_LOGIC_VECTOR (3 downto 0);
47  begin
48      if RESET='1' then
49          Qint := "0000";
50      else
51          if CLK'event and CLK='1' then
52              if Qint<9 then
53                  Qint:=Qint+1;
54              else
55                  Qint:="0000";
56              end if;
57          end if;
58      end if;
59      Q <= Qint;
60  end process;-- enter your statements here --
61
62  end cnt_4b;

```

Figure 7: cnt_4b.vhd



5. Compile source file created at the end of wizard to check any possible DRC error as shown in Figure 8.

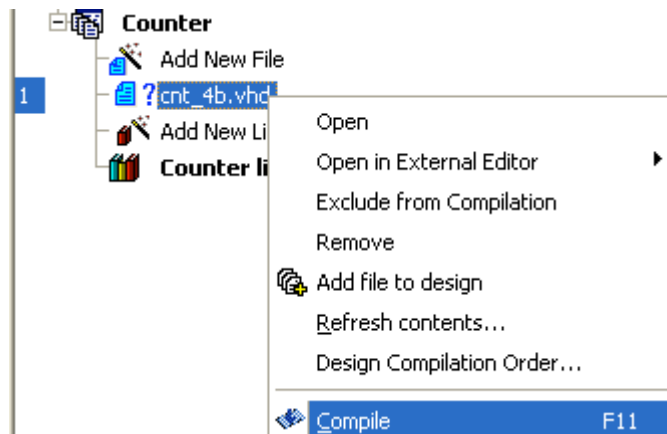


Figure 8: Compiling source.

Creating and2.v

1. From the main menu, Go to File->New ->Verilog Source. Click “Next” in the New Source File Wizard.
2. Type and2 as the name of source file and Click Next.
3. In the next window, you will add ports to your source file just like before. Have two input ports by name “a” and “b”. Add an output port by name “y”. For output “y” click on “Type” tab and select the Port Type as “reg” as shown in Figure 9. Click OK and Click Finish.

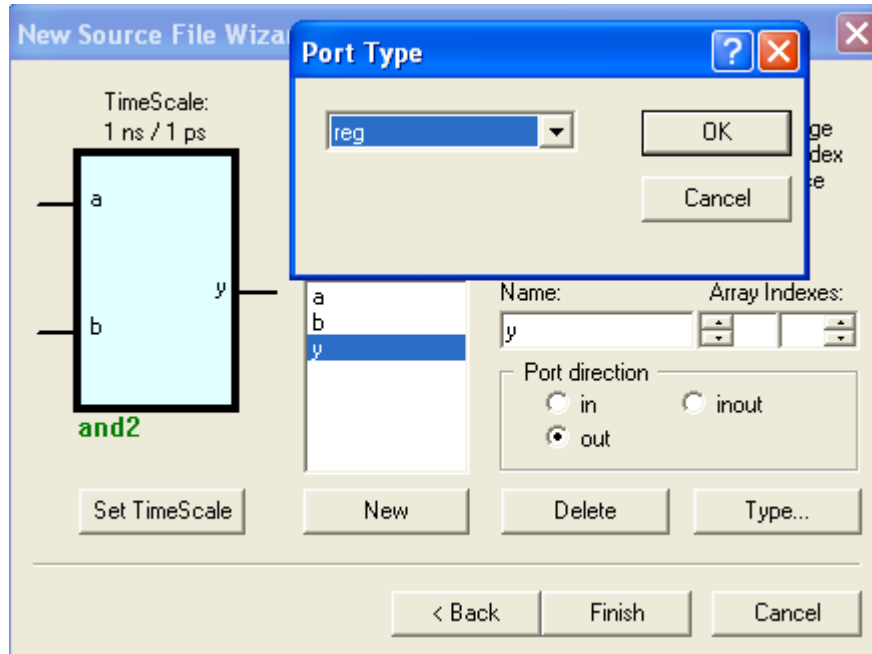


Figure 9: Ports and Port Type

- In the generated and2.v, please insert following statements just before the statement endmodule.

```
always @ (a or b)
```

```
y <= a & b;
```

The and2.v should look like as shown in Figure 10.



```
//-----
`timescale 1 ns / 1 ps

//{{ Section below this comment is automatically maintained
// and may be overwritten
//{module {and2}}
module and2 ( y ,a ,b );

output y ;
reg y ;

input a ;
wire a ;
input b ;
wire b ;

//}} End of automatically maintained section

// -- Enter your statements here -- //

always @ (a or b)
    y <= a & b;


endmodule
```

Figure 10: and2.v

5. Compile source file created just like before to check any possible DRC error.

Now we will create the top-level for our design using Block Diagram Editor.

Creating Top-Level

1. To get back to the design flow manager, press [**ALT+3**] on the keyboard.
2. Click the **BDE** tab to start the Block Diagram Editor.
3. Mark the option **Add the generated files to the design**. And Click on the **next**.
4. In the window select the language as **Verilog** and click on the **next**.
5. Type the name of the source file as **CNT_BCD** and click on the **next**.
6. In the next window click on the tab **NEW**, type **CLK** in the **NAME** window, select port direction as **IN** and hit enter on the keyboard. Add **RESET** and **GATE** following the same method.
7. Add **FULL** as an output. Also add **BCD_A,BCD_B,BCD_C,BCD_D** as outputs with array index defined as 3:0.
8. Click finish and end Block Diagram Wizard, it opens the Block Diagram Editor file on the screen with all your inputs and outputs defined in it.
9. Now hit [**s**] on the keyboard or click on  icon to open the symbol toolbox.



10. Drag and drop and2, cnt_4b symbols to your diagram editor so that the diagram contains three instances of and2 and four instances of cnt_4b symbols on the diagram.

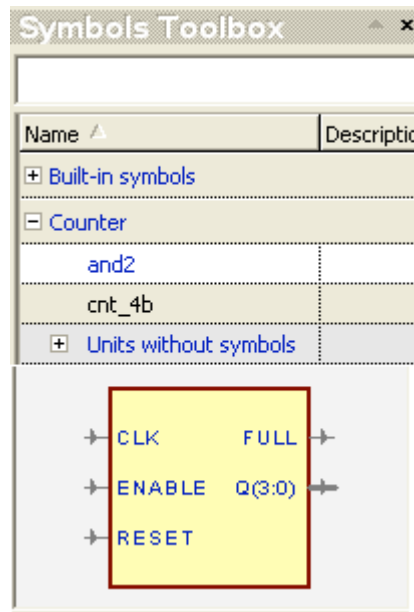




Figure 11: Symbol Toolbox.

11. Click on  icon and drag and drop the Global wire on to Diagram. Drop it two times as we need two Global wires one for CLK and one for RESET. Hit "Esc" On Keyboard.

12. Now double click on the  icons in Diagram and give the net name as CLK and RESET respectively.

12. Use shortcut [**w**] for wire and [**b**] for bus to connect different ports to respective location. **[See Figure 12]**

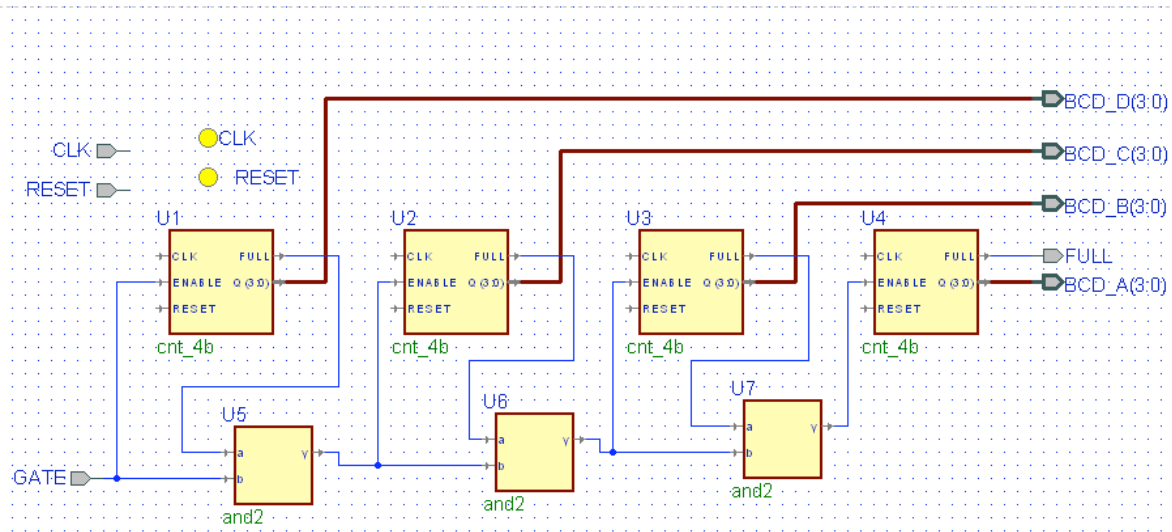



Figure 12: Block Diagram Editor Connections



To learn more about the Block Diagram Editor, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Active-HDL Tools / Block Diagram Editor**.

Compiling and Simulating

To compile the source file in Active-HDL you have several options:

- Right-click the icon of the source file in the Design Browser and select **Compile** from the pop-up menu.
- While the source file is open for editing/viewing, click the **Compile** button () on the main toolbar.
- While the source file icon is selected or the file is open, hit the **F11** key on your keyboard.



When compiling or simulating Verilog modules, Do not forget to add Verilog libraries to Design settings window. Go to **Design>>Settings** and add required libraries to subcategory **Verilog** of **compilation** and **simulation**.

Using the method you prefer, compile the CNT_BCD.bde file.

If the compilation was successful, a green confirmation message will be displayed in the **Console** window at the bottom of the workspace. If you see red messages in the console, it means that some errors were detected. In that case you should read the error message carefully, double-click the text of the message in the console to highlight the offending line in the source code (the document will be opened if it wasn't), correct the mistake and save/compile again.

A successfully compiled source file icon will have the list of the design units detected inside the file during compilation. Click the plus sign to the left of the CNT_BCD.bde icon to display the green [M] icon for the **module** detected while compiling the file.



Before you start the simulation, some module must be selected as the top level of your design. Since PulseGen is our design is being built as **top-level**, right click on the module icon of the **PulseGen** and click on the **Set as Top-Level**. When there are multiple pairs in your design, you can select one of them from the drop-down list at the very top of the Design Browser window.

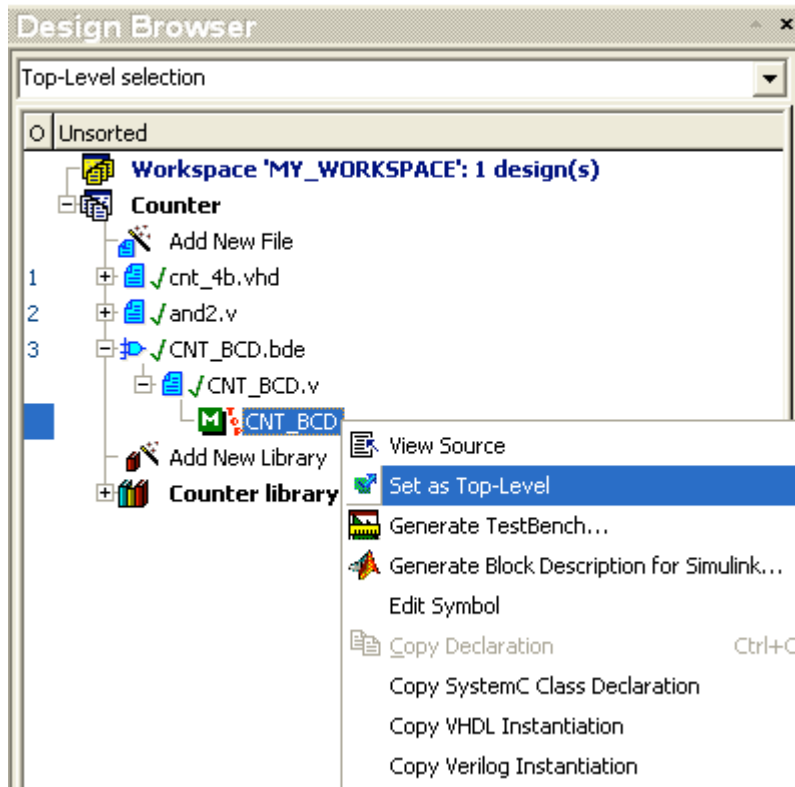


Figure 13: Setting Top-Level from Design Browser

When **CNT_BCD** is selected as the top level, you can start your simulation.

1. Select **Simulation | Initialize Simulation** from the menu; wait till the design browser switches to the **Structure** tab automatically.
2. Select **File | New | Waveform** from the menu (or click the matching icon in the main toolbar).
3. Highlight the green chip icon for **CNT_BCD** from the **Structure** tab of Design Browser window. The bottom window of Structure tab displays the signals. Using CTRL keystroke, select the signals **CLK, RESET, GATE, BCD_A** and drag and drop them onto the waveform editor.
4. We will Right-click the **CLK** port name in the waveform window and select **Stimulators** from the pop-up menu.
5. Drag the floating **Stimulators** window (Figure 14) to the bottom of the screen, so that you can see both the window and all signals in the waveform.

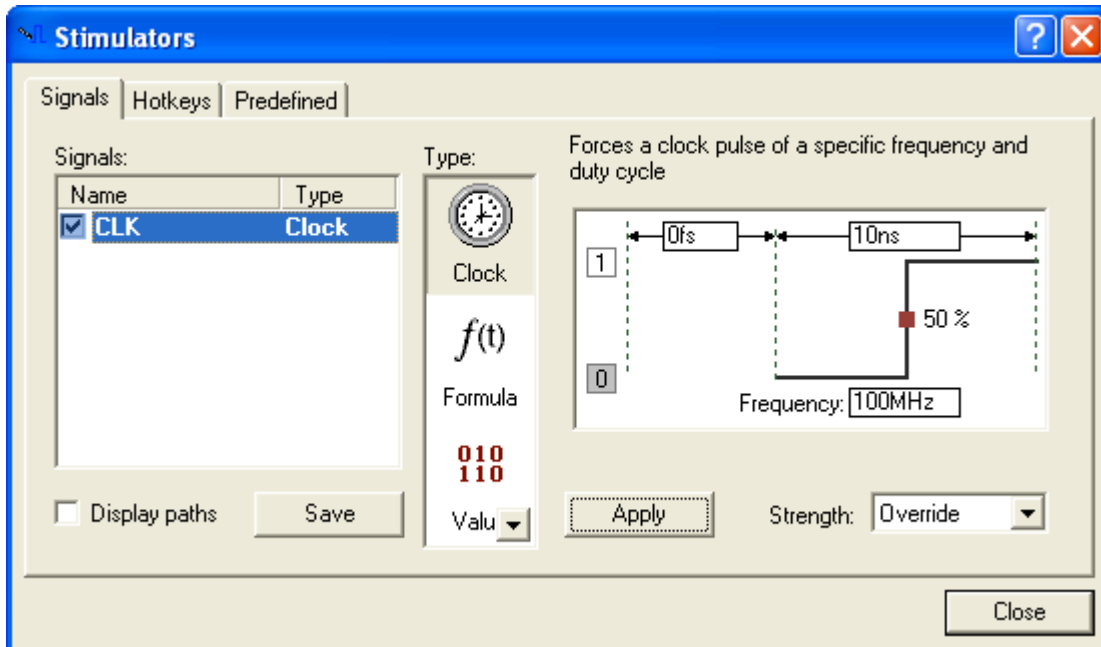


Figure 14: Stimulators window with Clock stimulator type selected.

6. While **CLK** is highlighted in the **Signals:** list, select **Clock** as the stimulator in the **Type:** list. Adjust clock frequency to 100MHz (clock period will adjust automatically to 10 ns) – the **Stimulators** window should look exactly as in Figure 10. Click **Apply**.
7. Without closing the **Stimulators** window, click the **RESET** port name in the waveform; the port name should be transferred to the **Signals:** list in the Stimulators window.
8. Select **Formula** as the stimulator type for **RESET** and type
1 0 ns, 0 120 ns
in the **Enter formula** field (don't skip any spaces!). Click **Apply**.
9. Without closing the **Stimulators** window, click the **GATE** port name in the waveform. Select **Formula** as the stimulator type for **GATE** and type
0 0ns, 1 220 ns
10. Click **Apply** and close the **Stimulators** window.
11. Select **Simulation | Run Until...** from the menu. Type 500 ns in the **Run Until** window. Click **OK**.

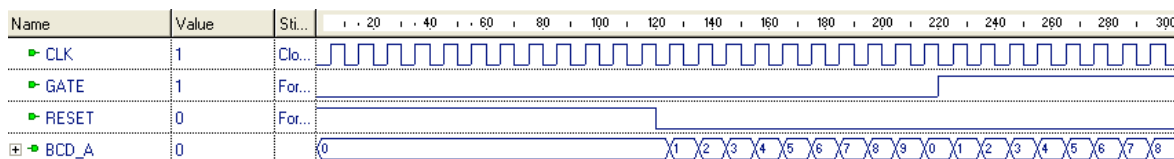


Figure 15: Functional Simulation Output



Analyze the results displayed in the waveform. If you are curious what is the signal value represented by a section of the waveform, click it and examine the **Value** field right by the signal name.

When you are done, select **Simulation | End simulation** from the menu, save the waveform as CNT_BCD.AWF and close the waveform window.



To learn more about simulation, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Using Active-HDL / Simulation**. We will be conducting more simulations in this guide.

Basic Debugging

Simulate the top level again. This time do not end simulation – use **Restart** toolbar button (or **Simulation | Restart Simulation** option from the main menu) to return to simulation time 0.

1. Run simulation for 10 us.
2. Use **Trace Into** button in the toolbar or **F7** key to step through the execution of your code. Observe yellow highlight marking currently executed line of your HDL code. Notice current values of signals displayed inside rectangular probes visible on block diagrams. (You can open sources manually or wait until they are executed and displayed automatically.)
3. Open any source file and find the line where you want the simulation to stop. Click that line and hit **F9** on your keyboard to set breakpoint.
4. Use **Run** button in the toolbar or **Alt+F5** keys on your keyboard to run simulation. Observe how simulation stops at the breakpoint. You can see the time when simulation stopped in the console or in the special field at the end of the toolbar.
5. End simulation and clear breakpoints (**Simulation | Clear All Breakpoints** option from the menu).



To learn more about debugging, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Using Active-HDL / Debugging HDL Code**.



Using Existing design for FPGA-PCB design flow

Now that we have learnt how to create a workspace with a mixed language design, simulation and debugging, let's use an existing sample design and go through Synthesis, Implementation, specifying FPGA Pin Constraints and export/import of pin information file to and from PCB tool CADSTAR.

We will use the sample design that comes with the installation of Active-HDL. From the main menu, go to File->Open Workspace/Design Explorer. In the Workspace/Design Explorer window, select Active-HDL->Samples->VHDL_designs->freq_meter. Double click on the freq_meter. freq_meter workspace with the design freq_meter attached appears inside Active-HDL as shown in Figure 16.

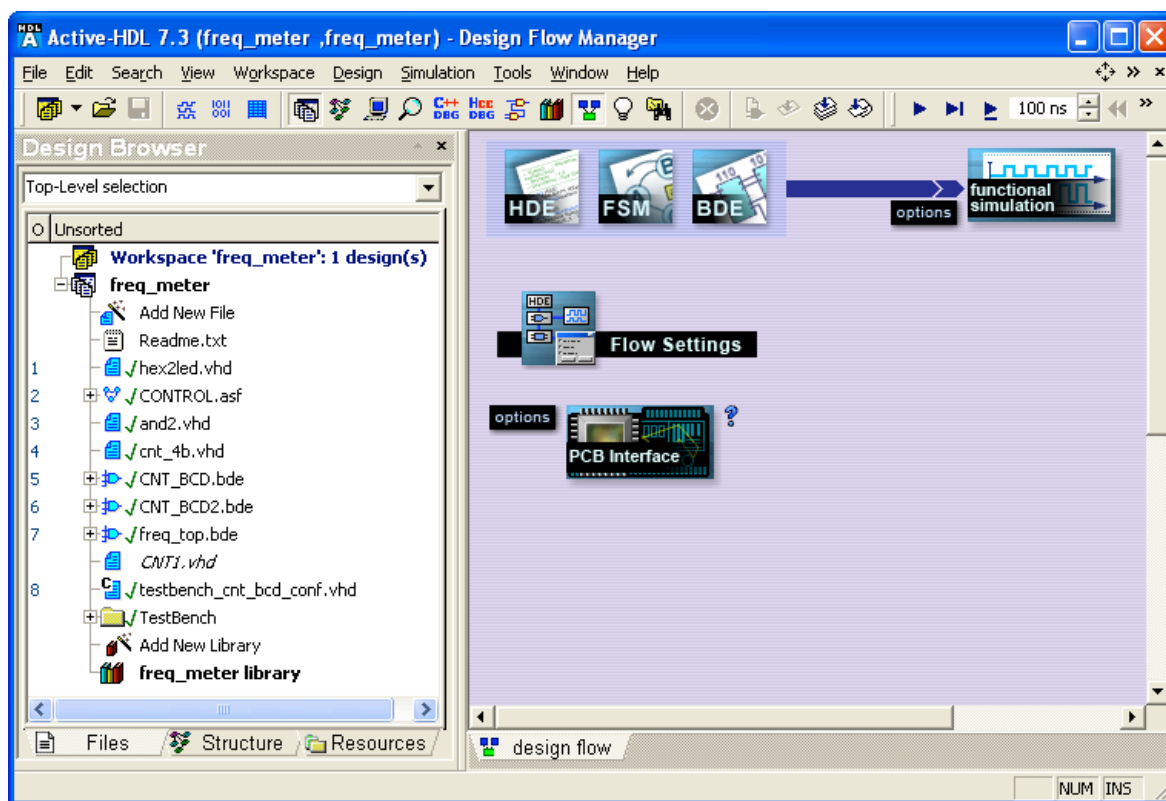


Figure 16: Sample Design freq_meter

If you examine the design, it contains mixture of HDL files, graphical Block Diagram Editor files and State Diagram Editor file. Expand the "TestBench" folder and execute the runme.do file as shown in Figure 17.

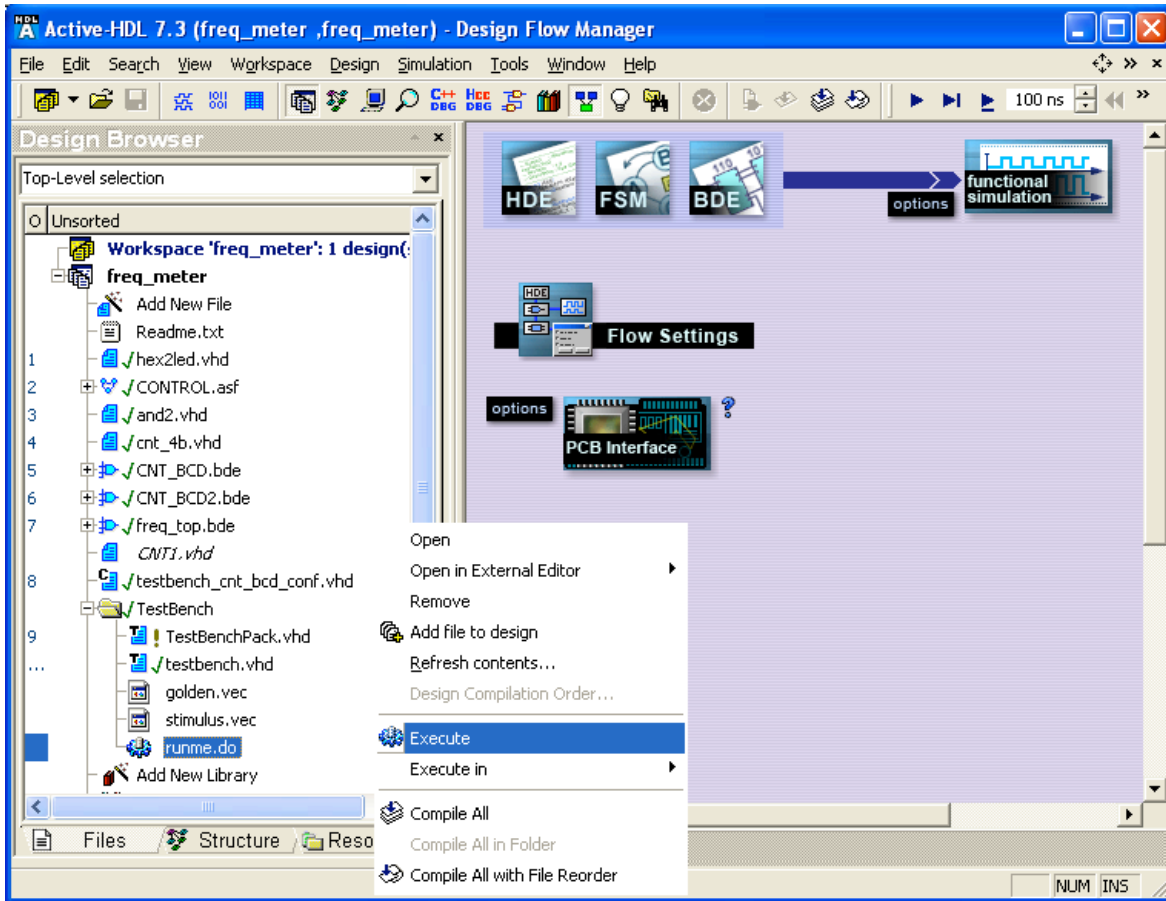


Figure 17: Execute runme.do

The runme.do contains commands that compile all the design files and simulates the testbench. The results of functional simulation are shown in Figure 18.

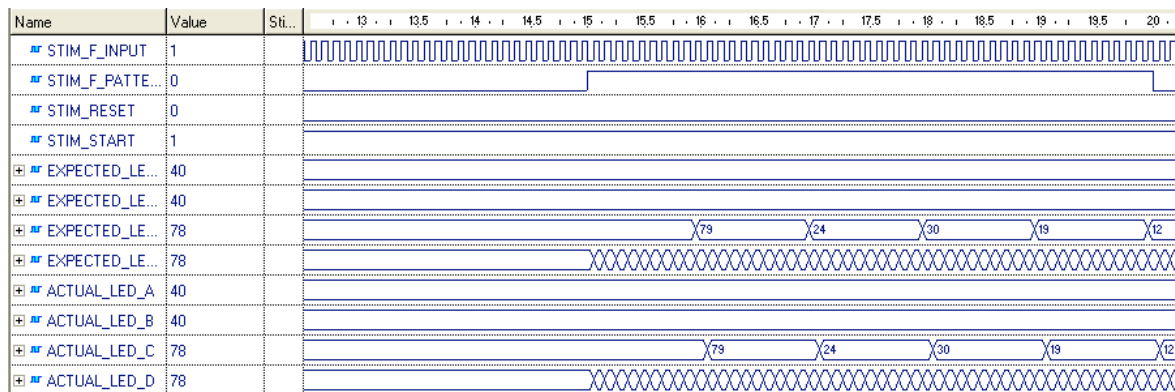


Figure 18: Functional Simulation

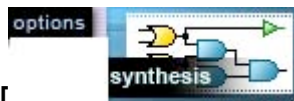


When the simulation finishes, click on “OK” in “Simulation” Dialog box and save the waveform by choosing File->Save as from the main menu. You can save the waveform as “functional.awf”.

Now click on the “Flow Settings” to set the HDL Synthesis tool and Implementation tool for this design. We will again use **Xilinx ISE/WebPack 9.2 XST VHDL/Verilog** as HDL Synthesis tool and **Xilinx ISE/WebPack 9.2** as implementation tool.

Synthesis with Xilinx ISE/WebPack 9.2 XST VHDL/Verilog

After RTL simulation has been completed in Active-HDL, the files can be passed to Synthesis. Synthesis settings can be configured with the Synthesis **options** button in the Design Flow Manager.



1. Clicking on the Synthesis **options** [ ] button invokes the options window for synthesis.
2. Make sure all design files are on the design tree of the synthesis option. You can select or deselect them by right clicking them. Let’s have six design files selected for Synthesis as shown in Figure 19.

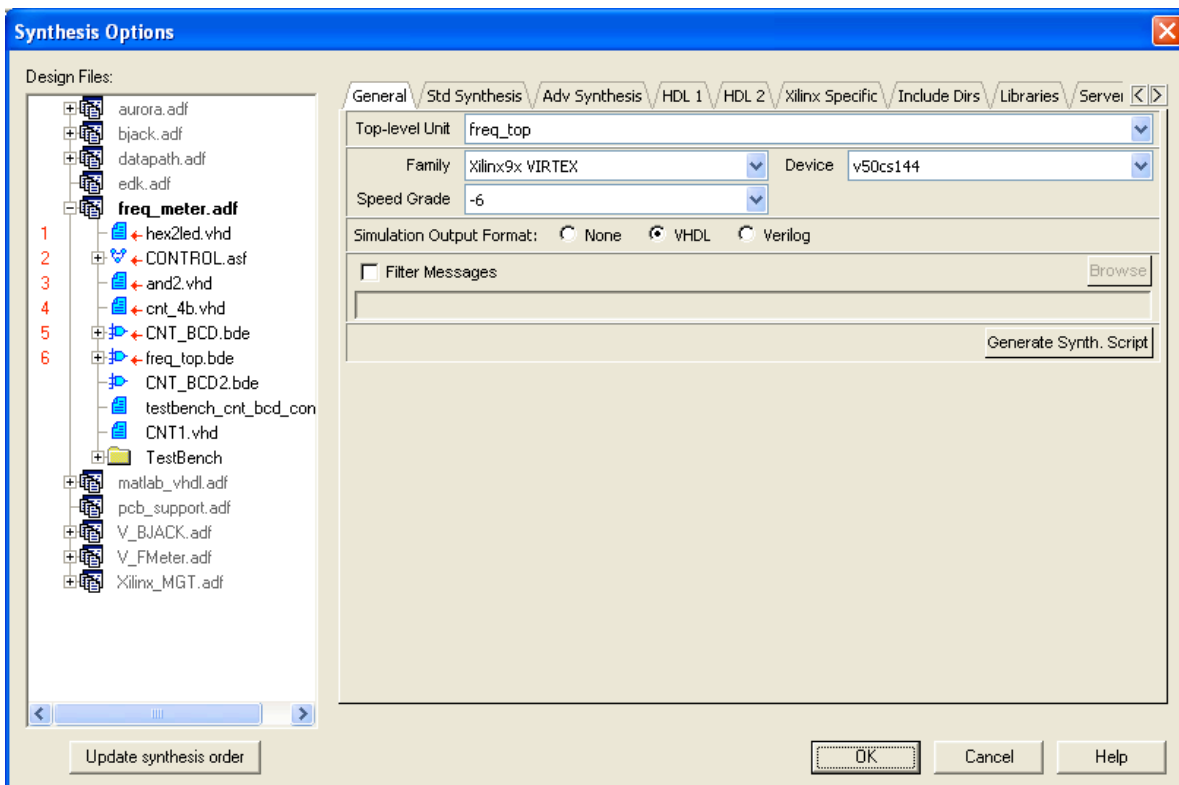


Figure 19: Xilinx ISE/WebPack XST Synthesis Options



3. Select Top-Level, Family, Device and speed grade as freq_top, virtex, v50cs144 and 6 respectively as shown in Figure 20.

Top-level Unit	freq_top		
Family	Xilinx9x VIRTEX	Device	v50cs144
Speed Grade	-6		

Figure 20: Selecting Top-Level, family and device.

4. Select simulation output format to **VHDL**
5. You can select and set other synthesis options also from the window.
6. When you are done with all the settings you can click on **OK**.
7. Now you can automatically perform push-button Synthesis by clicking the **synthesis**



] button in the Design Flow Manager.

- 8.



If synthesis is run in **batch mode**, Active-HDL takes full control of the process. All files will be automatically passed to the Synthesis tool and when Synthesis is complete; all results from the Synthesis Tool will be passed to Active-HDL. If the Synthesis tool is run in **GUI mode (This option is available for Select Synthesis tools in Active-HDL)**, then Active-HDL passes all the necessary design files to the Synthesis tool and the resulting files after synthesis can be back annotated into Active-HDL in a semi-automatic way.

9. While the Synthesis is running, all of the Synthesis results and information can be seen on-line in a special window.

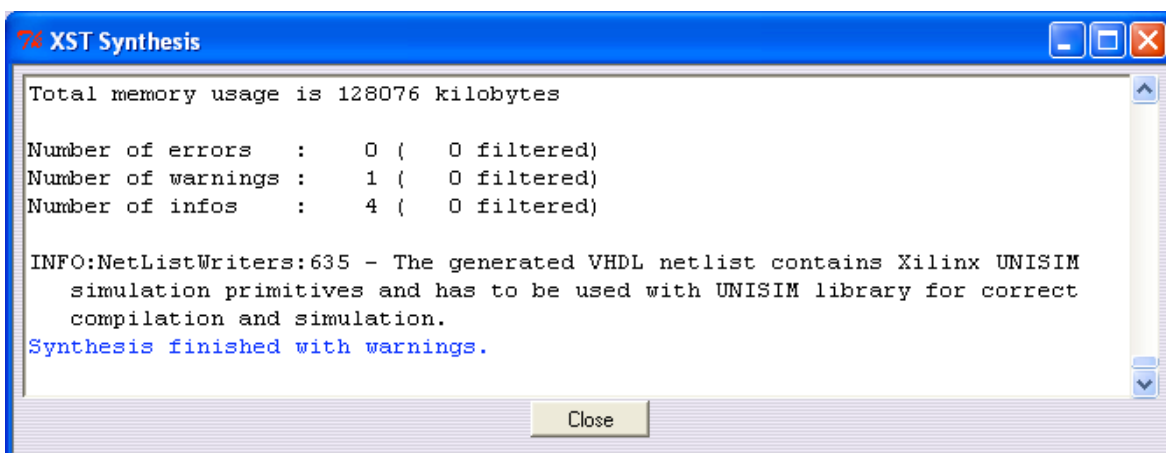


Figure 21: Xilinx XST Synthesis window.

10. Synthesis reports can then be viewed by clicking on the corresponding Synthesis **reports** button in the Design Flow Manager.

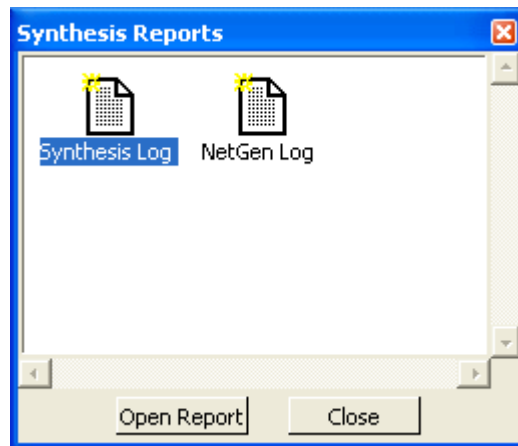


Figure 22: Synthesis Reports Window

11. You will see new folder and new libraries named post-synthesis created in the Design Browser.

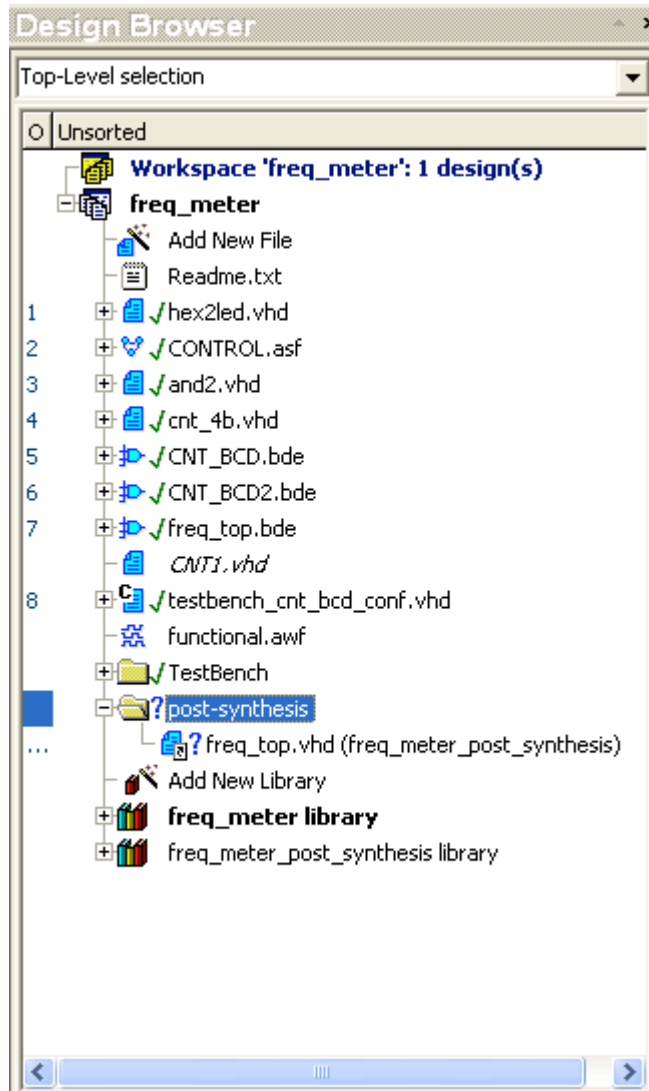


Figure 23: Design Browser after Post-Synthesis.



Post-Synthesis Simulation

1. Click on the **options** tab of the post synthesis simulation icon.

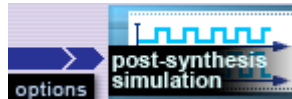



Figure 24: Post-Synthesis simulation tab

2. Click on the  icon to add synthesis netlist (synthesis/freq_top.vhd) and testbench file src/TestBench/TestBenchPack.vhd, src/TestBench/testbench.vhd in that order to opened window. Click **OK** when you are done adding files.

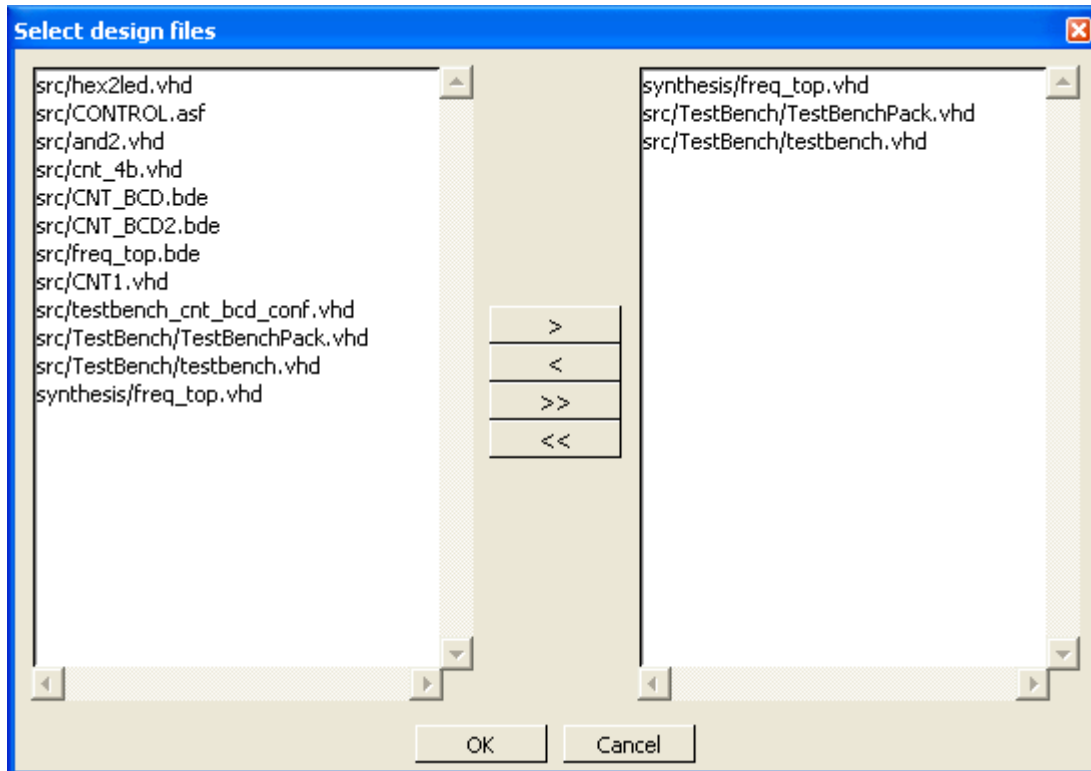


Figure 25: Selecting netlist file and testbench files for simulation.

3. Now click on the **choose** button to select the top-level of your design. Select "testbench", Click "Add" and click OK to go back to the options window again.

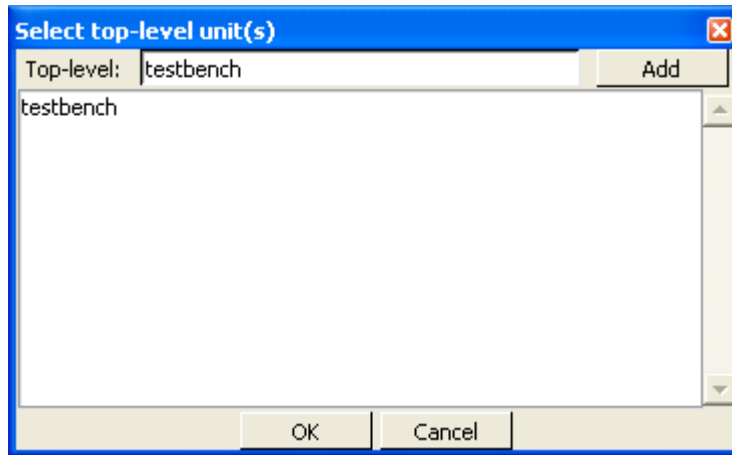


Figure 26: Selecting top-level of the design.

- Click on the **post-synthesis tab** and wait till waveform editor window opens. Once waveform editor window opens, click on **run for 25 us** tab. Check for any mismatches in the waveform.

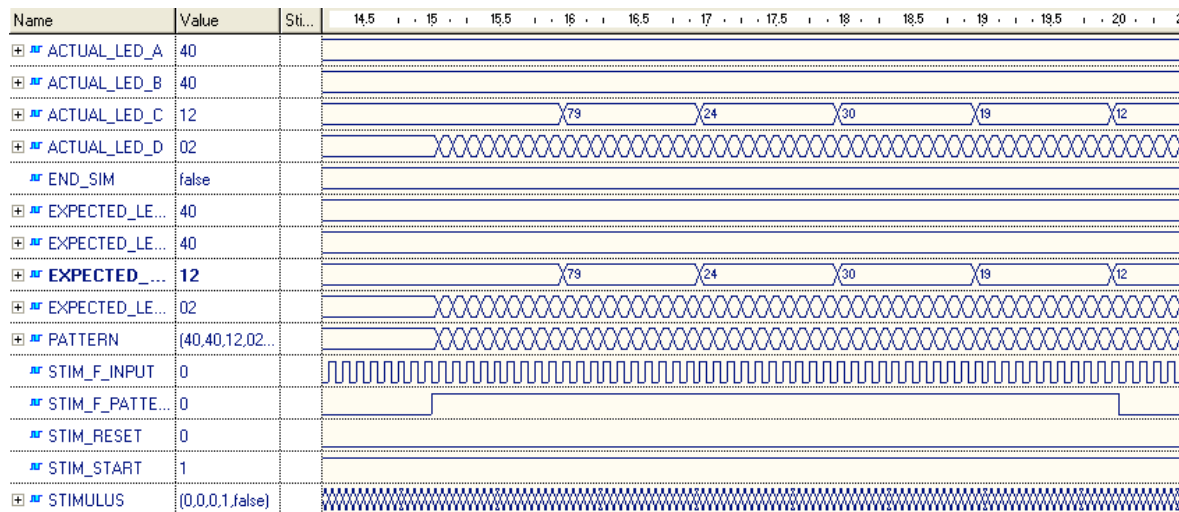
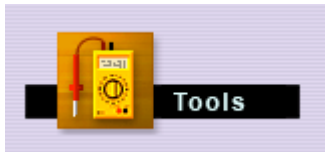


Figure 27: Post-synthesis Simulation Output.



Generating I/O Pin Constraints File for Implementation

Now we are ready to implement our design to Xilinx Virtex FPGA. Before we start the implementation process, a very important step is to define the I/O Pin Constraints for the design. To write the Constraints file we can use the Xilinx PACE and Constraints Editors. Click on the Tools button



in the design flow manager. "Tools" button opens up Design Entry tools of Xilinx and it includes PACE and Constraints Editor tools.

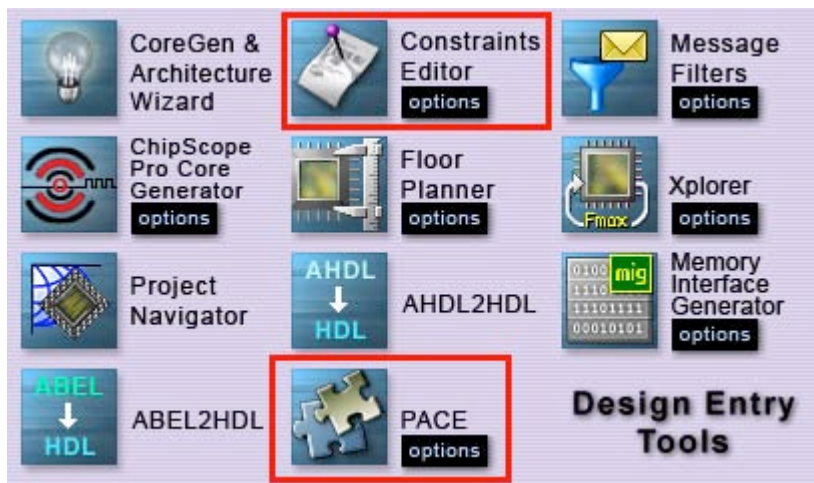


Figure 28: Xilinx Design Entry Tools

Pinout and Area Constraints Editor (PACE) is an interactive graphical tool that you can use to:

- 1) View and edit location of constraints for I/Os and global logic
- 2) View and create area constraints for logic in the design
- 3) Determine resource requirements of the design
- 4) Determine resource layout of the target device

PACE is used during initial design entry or after consolidation of a design into a netlist file. For initial design entry, PACE reads and writes VHDL and Verilog files limited to I/O definitions. For consolidated netlists, PACE reads the NGD file. In both cases, PACE reads and writes user constraint files (UCFs).



To learn more about PACE and how to use it in Active-HDL, go to *Active-HDL Online Documentation*, then in the Contents tab select **Active-HDL Help / Active-HDL Tools / Design Flow Manager / Multivendor Flowchart / Launching Additional Flow Tools / Xilinx / PACE Options Dialog Box**.



Constraints Editor is a program that you can use to create and modify the most commonly used constraints



To learn more about Constraints Editor and how to use it in Active-HDL, go to *Active-HDL On-line Documentation*, then in the Contents tab select **Active-HDL Help / Active-HDL Tools /Design Flow Manager /Multivendor Flowchart/ Launching Additional Flow Tools/Xilinx/Constraints Editor Options Dialog Box**.

Using PACE and Constraints Editor, you can prepare a customized Xilinx constraints file (ucf) file.

A simple example of Xilinx constraints file for design freq_meter targeted to virtex family v50cs144 device looks like in Figure 29. Please note that this is a very basic Xilinx constraints file with Pin names and locations on the device. Please refer to Xilinx Documentation and Xilinx PACE and Constraints Editors (which can be launched from Active-HDL) on how to write constraints files for FPGA design targeted to Xilinx FPGAs.



```

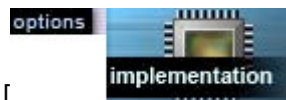
NET F_INPUT LOC=A6;
NET F_PATTERN LOC=A7;
NET RESET LOC=G1;
NET START LOC=H1;
NET LED_A<2> LOC=A8;
NET LED_C<4> LOC=B7;
NET LED_C<1> LOC=B10;
NET LED_C<3> LOC=C8;
NET LED_C<5> LOC=C9;
NET LED_C<0> LOC=C13;
NET LED_A<3> LOC=D8;
NET LED_C<2> LOC=D12;
NET LED_C<6> LOC=E13;
NET LED_A<5> LOC=F12;
NET LED_A<1> LOC=F13;
NET LED_A<6> LOC=G13;
NET LED_A<0> LOC=H13;
NET LED_A<4> LOC=K11;
NET LED_B<0> LOC=K13;
NET LED_B<5> LOC=L11;
NET LED_B<3> LOC=M11;
NET LED_B<4> LOC=N11;
NET LED_B<6> LOC=N8;
NET LED_B<1> LOC=N9;
NET LED_B<2> LOC=M8;
NET LED_D<0> LOC=M10;
NET LED_D<4> LOC=K9;
NET LED_D<3> LOC=K8;
NET LED_D<6> LOC=K4;
NET LED_D<5> LOC=K5;
NET LED_D<1> LOC=K6;
NET LED_D<2> LOC=N5;
    
```



Figure 29: Simple Example of Xilinx I/O Constraints file (Xilinx.ucf)

[Please click here to download the Xilinx.ucf file](#)

Implementation with Xilinx ISE and Providing Implementation Constraints

To run the Xilinx Implementation, similar steps are taken as for synthesis.



1. Click on the implementation **options** button [ ] in the Design Flow Manager.
2. Select the simulation output format as **VHDL** and running mode as **Batch**.
3. You can set other implementation options also in this area.

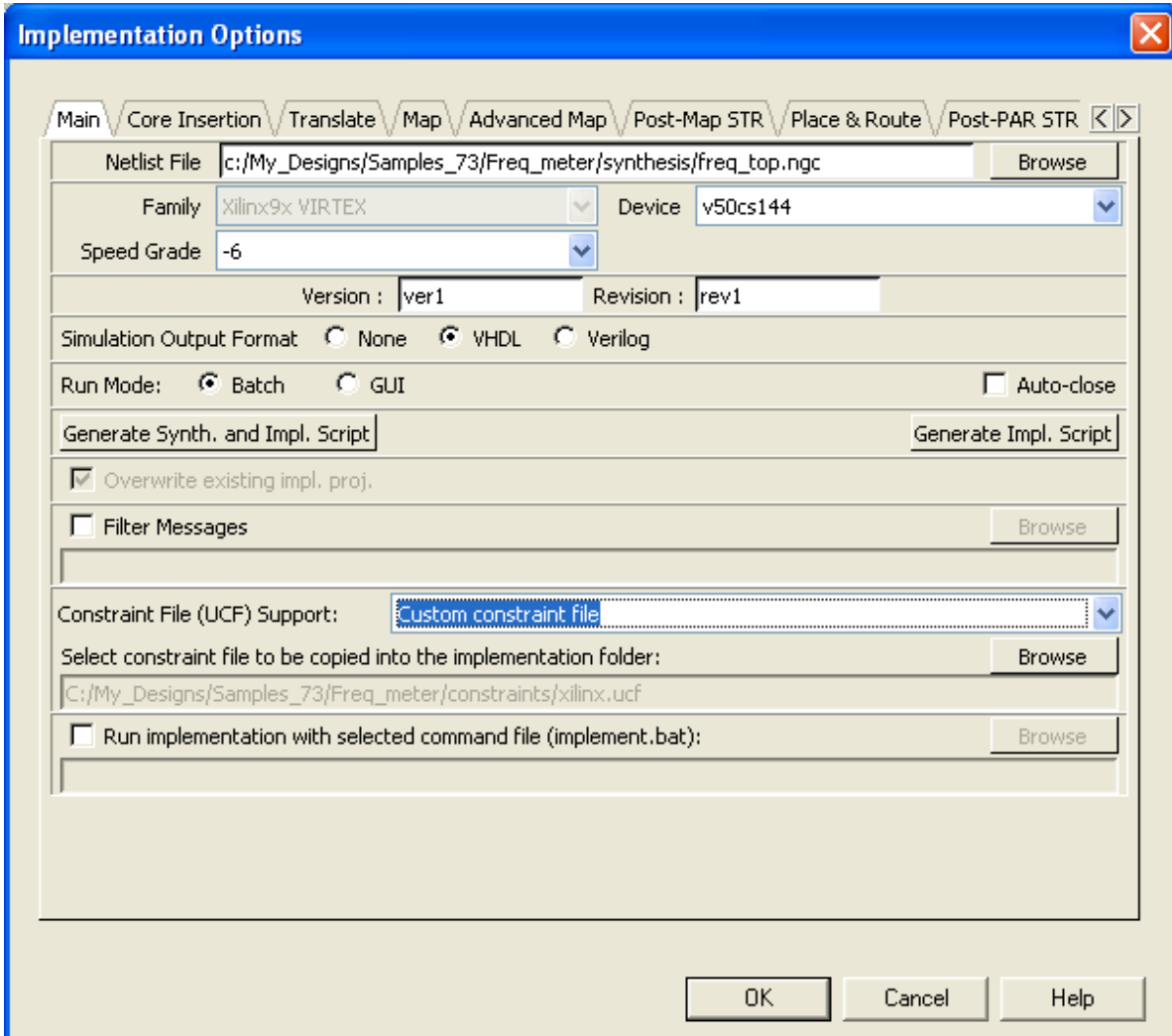


Figure 30: Xilinx ISE\WebPack Implementation Options with Xilinx.ucf selected as Custom constraints File

In the Constraint File (UCF) Support field, there are three options to specify implementation constraints.

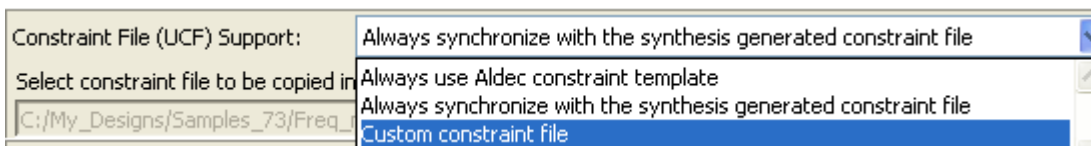


Figure 31: Select Constraints File Options



Always use Aldec constraint template

Specifies to fetch during implementation the constraint template delivered with Active-HDL (\$ALDEC\Dat\Template.ucf). The path to the template is displayed in the Select constraint file to be copied into the implementation folder edit box. When this option is selected, the Browse button is disabled. Note that \$ALDEC\Dat\Template.ucf file is just a template. You have to modify this to include the constraints for the design.


Always synchronize with the synthesis generated constraint file

Specifies to fetch during implementation the constraint file generated by the selected synthesis tool. This option is selected automatically when Synplify/Synplify Pro and Xilinx tools are in use. The path to the file is displayed in the Select constraint file to be copied into the implementation folder edit box. When this option is selected, the Browse button is disabled.

Custom constraint file

Specifies to fetch during implementation a user-defined constraint file.(example is Xilinx.ucf as discussed above) When this option is selected, the Browse button is enabled and you can point to the constraint file that you have downloaded in the dialog box. The path to the file is displayed in the Select constraint file to be copied into the implementation folder edit box.



4. When you are done with the settings, click **OK**. Click **Implementation** tab [] on the design flow manager.
5. While the Implementation is running, all of the Implementation results and information can be seen on-line in a special window.
- 6.

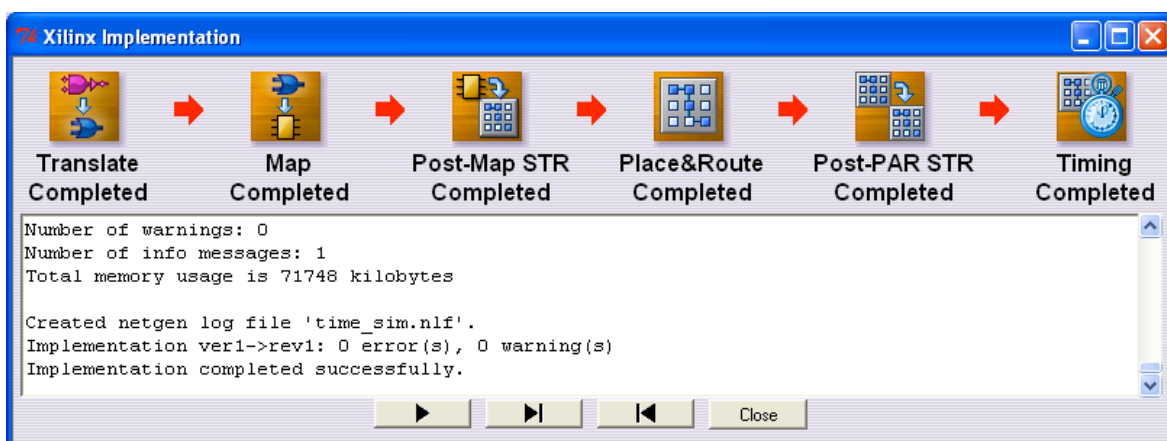


Figure 32: Xilinx ISE/WebPack implementation window in batch mode.

7. Implementation reports can then be viewed by clicking on the corresponding implementation **reports** button in the Design Flow Manager.

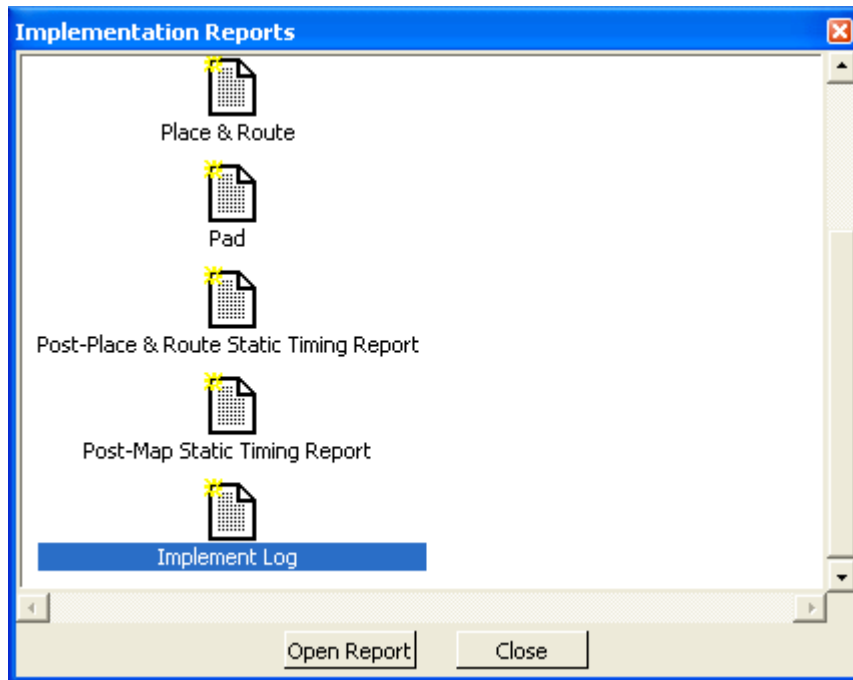


Figure 33: Implementation Reports Window.



If Implementation is run in **batch mode**. All files will be automatically passed to the Implementation tool and when task is complete; all results will be passed to Active-HDL. If you run in **GUI mode**, then Active-HDL passes all the necessary design files to the Implementation tool.

After Successful Implementation of the design, Implementation tool generates implementation report constraints file, *freq_top_pad.csv* in the *C:\My_Designs\Samples_73\Freq_meter\implement\ver1\rev1* folder.

Implementation with Xilinx ISE/WebPack in GUI Mode

1. To run the implementation in GUI mode select **run mode** to GUI in implementation options.

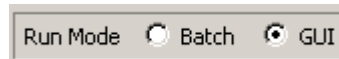


Figure 34: Selecting Run mode for the implementation

2. You can select and set other Implementation options also from the window.
3. When you are done with all the settings you can click on **OK**.
4. Now you can automatically perform push-button Implementation by clicking the **Implementation**



[**implementation**] button in the Design Flow Manager.

5. You will see the Xilinx ISE Project Navigator running on the screen. When you see the process is over you can go back to Active-HDL and click on the **refresh file list** tab.



Figure 35: Refreshing implementation file lists

6. You will be prompted with choose source files window. Make sure you select the correct file if they have not been selected automatically by tool. Hit **OK** now.

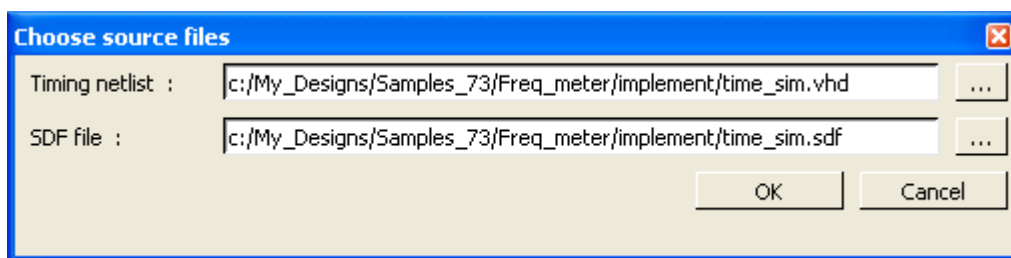


Figure 36: Choosing source files

7. You will see new folder *timing* and new library *freq_meter_timing* library created in design browser.

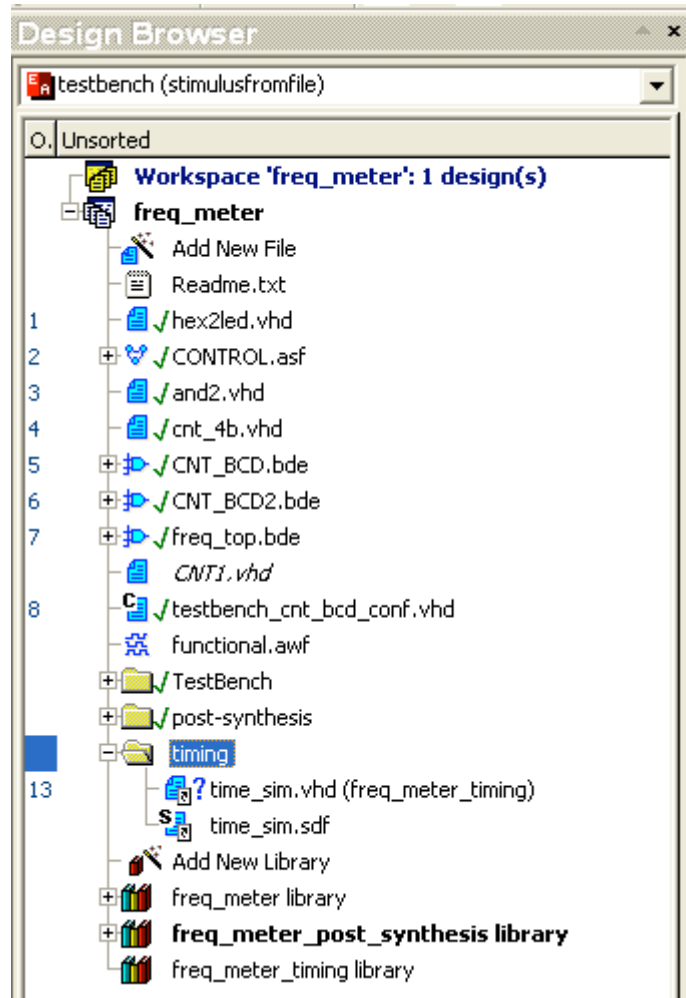


Figure 37: Design browser after Implementation

Timing Simulation

Timing simulation uses the output files created by the implementation tool. Depending on the current implementation options, the implementation tool generates VHDL code, Verilog code, or EDIF netlist.



1. Click on the **options** [options] of the timing simulation tab.
2. Here you should be able to see .sdf in the **SDF File** box and **Input files** box contains output vhdl netlist file of the implementation.

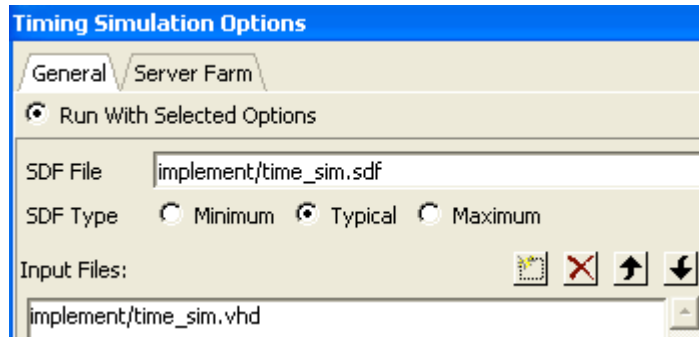



Figure 38: Timing simulation files in Options tab

- Click on  icon to add your timing netlist. Select the proper files from the list appeared in the new window and click on the **arrow tab** to add them to the list and then click **Ok** to back to options window. Add the TestbenchPack.vhd and testbench.vhd in that order.

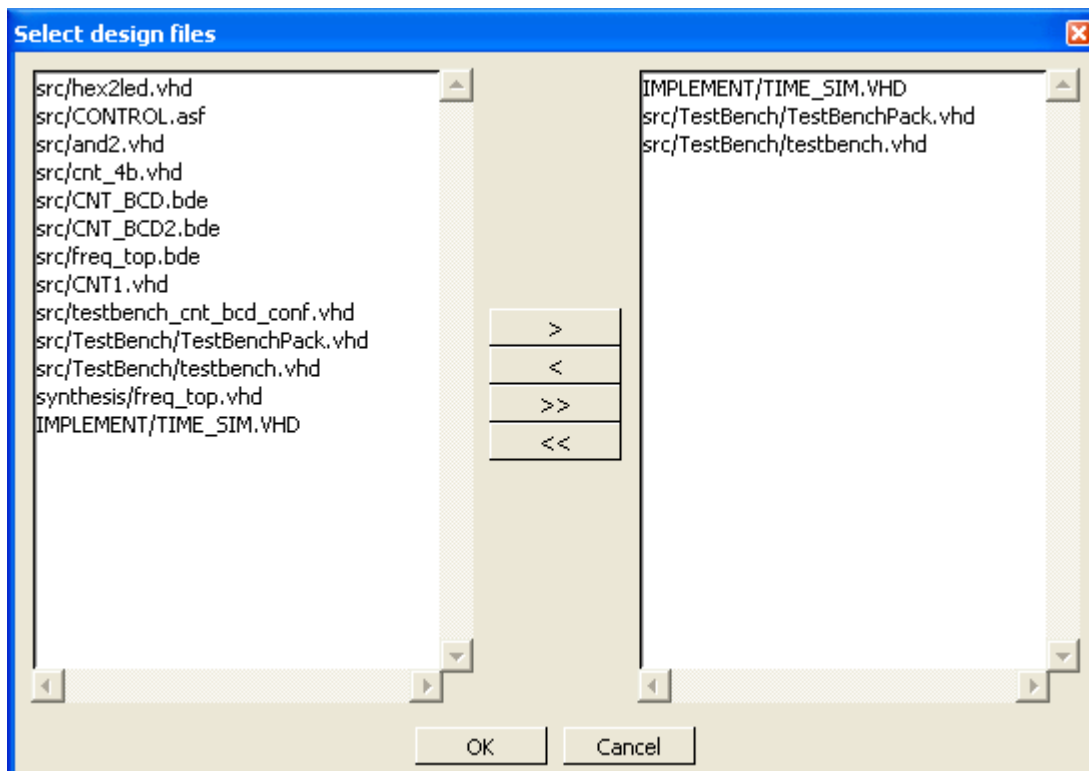


Figure 39: Adding timing netlist and testbench files for simulation

- Click on the **Choose** button. Select the top-level from the available list and then click Add and click OK.

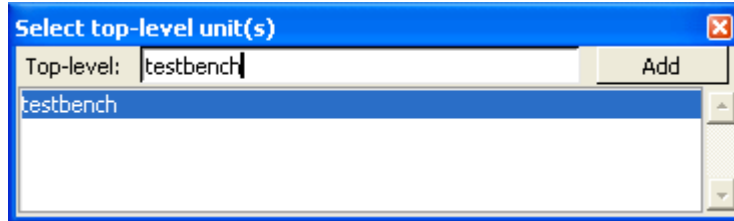



Figure 40: Selecting top-level unit(s)

- When you are done with the options, click on the **Timing simulation** tab  on the design flow manager.
- Now you can run the simulation and debugging the same way we did in the functional simulation and post-synthesis simulation.

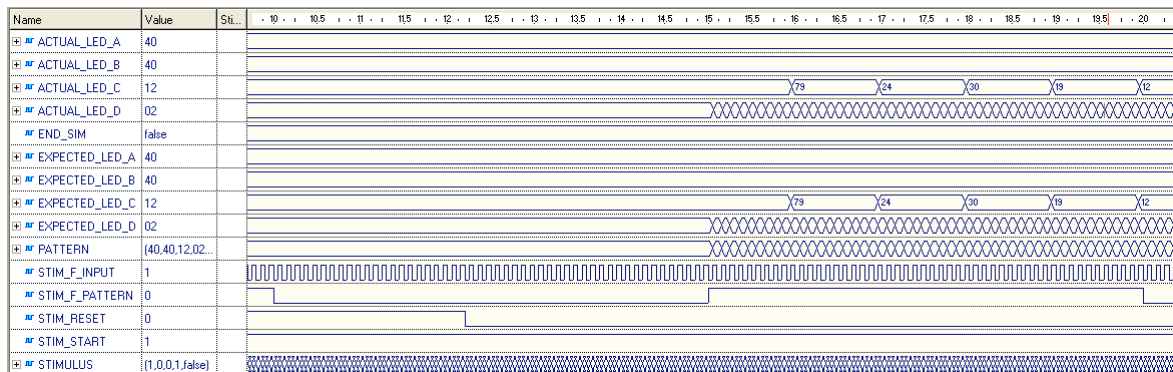


Figure 41: Timing Simulation output.

Export of Implementation Reports Constraints file to CADSTAR PCB Tool.

After Successful Implementation of the design, Implementation tool generates implementation report constraints file, *freq_top_pad.csv* in the *C:\My_Designs\Samples_73\Freq_meter\implement\ver1\rev1* folder. Now we can easily export the implementation report constraints file *freq_top_pad.csv* to CADSTAR PCB tool so that PCB designer can apply constraints to the PCB design which often runs parallel to the FPGA design. Click on the **options** to the left of **PCB Interface** Button in the Design Flow Manager.





In the **PCB Interface Options** window, choose the option “Export to PCB”. Make sure that PCB Tool is set to **CADSTAR**.

Note: Export to PCB reads information about pins from implementation reports, implementation constraints and synthesis constraint files and writes it to the CSV file. Implementation reports from Actel Designer, Altera Quartus, Lattice ispLEVER and Xilinx ISE could be read. Also implementation constraints used by these tools are supported. Synthesis constraints from Synplicity Synplify, PrecisionRTL and Xilinx XST are also supported.

For the **PCB File** option, Click on **Browse** and select a location where you would like to save the exported PCB constraints file in csv file format.

For the Synthesis/Implementation Tool Options select Xilinx ISE and Input File type be “Implementation Reports”. Click on the Browse button for Input File and locate the implementation reports file, freq_top_pad.csv that was created after implementation of the design. The Family should be automatically set to VIRTEX. If not, go through the drop-down box and select VIRTEX.

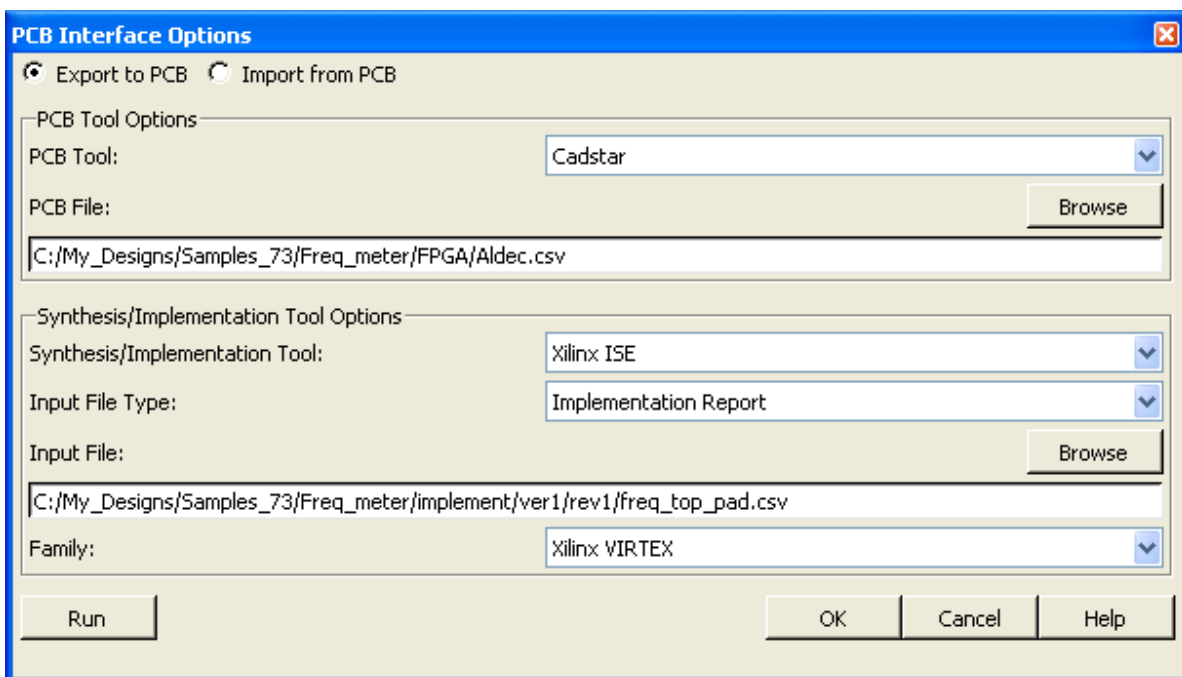


Figure 42: PCB Interface Options- Export to PCB



To learn more about the PCB Interface Options, click on the **Help** tab in the PCB Interface Options Window.



When you are done with Options, click the “Run” button OR click OK and then hit the icon the design flow manager. You should be able to see successful export of the file to PCB messages in the console window and right check mark next to the PCB Interface icon in the design flow.



PCB designer then would use the **Aldec.csv** file to apply constraints to the PCB Layout design. Often times Swapping of Pins and Pin name change is required for Optimal PCB routing.

CADSTAR Schematics Block Creation Wizard

The first time you create a FPGA for CADSTAR you have to create a schematic symbol, use an existing PCB Component (or create a new one) and create a Part. If you double-click the CADSTAR icon on your desktop you should see the CADSTAR Design Editor Window.

Select in the menu bar **File – New – Schematic Symbol – Block Wizard** and follow the steps through the wizard in order to create first a schematic symbol to be used later in your schematics design. Select **OK** as shown in Figure 43.

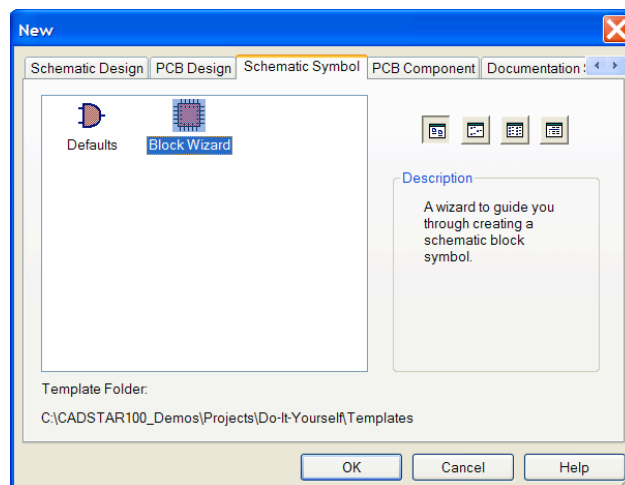


Figure 43: Schematic Symbol

Next as shown in Figure 44, you can fill in any name as Reference Name for the symbol you want to create (for this exercise you don’t need to fill in the Alternate Name and Version). You might want to change the units from thousand of an inch to millimeters. Select **Next**

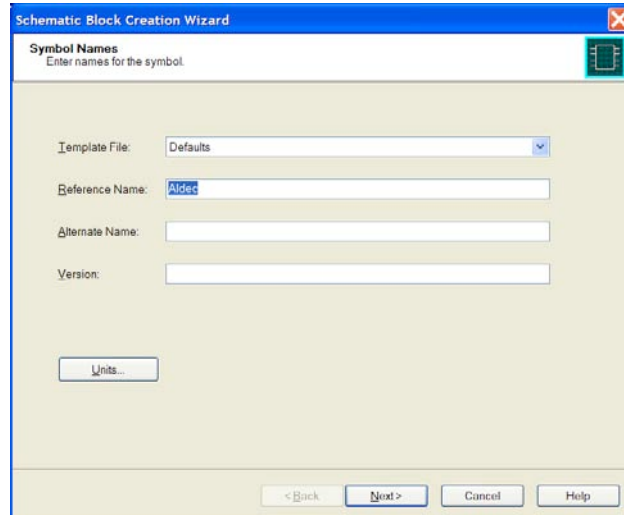


Figure 44: Block Creation Wizard

You can enter the symbol dimensions as shown in Figure 45, by using the values as in the example. Select **Next**

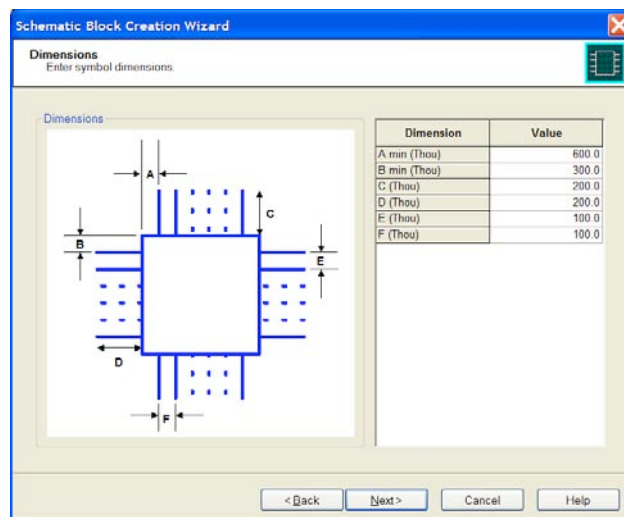


Figure 45: Symbol Dimensions

You can enter the pin locations as shown in Figure 46. Enable **Import Legacy Symbol Text**

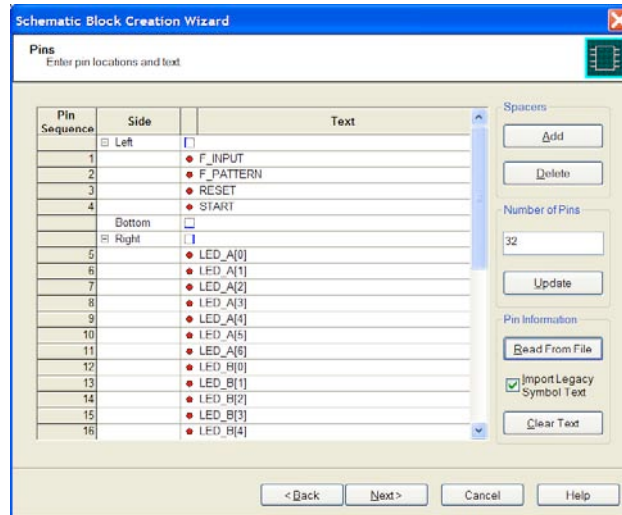


Figure 46: Pin Locations

CADSTAR will automatically sort the pin locations first based on the positioning (LEFT, BOTTOM, RIGHT and TOP) as in the pin list file from Aldec (if this information is available) and the second automatic sorting is done alphanumeric on the label names. If the result is not satisfactorily you can manually sort the labels as you like. To do so you can for example select the *Pin Sequence* numbers 1 until 4 by using the CTRL/SHIFT key and drag and drop the *Pin Sequence* column to *Bottom*. If any pins in the current FPGA design are not connected (not used) it's preferable to keep them and divide along the four sides (LEFT, RIGHT, TOP and BOTTOM) in the same way as described above.

Select **Read From File**. Choose **Files of type**: CSV (Aldec FPGA Data) and select '**Aldec.csv**' as shown in Figure 47.

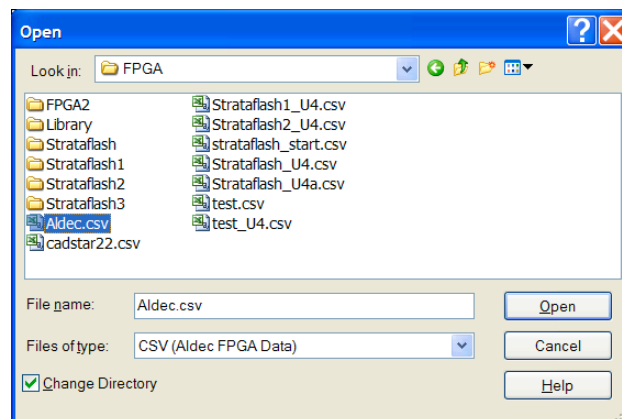


Figure 47: Selecting Aldec.csv file

After you have imported the file select **Next**. You can now setup the pin name and label origins (you can use the default values)

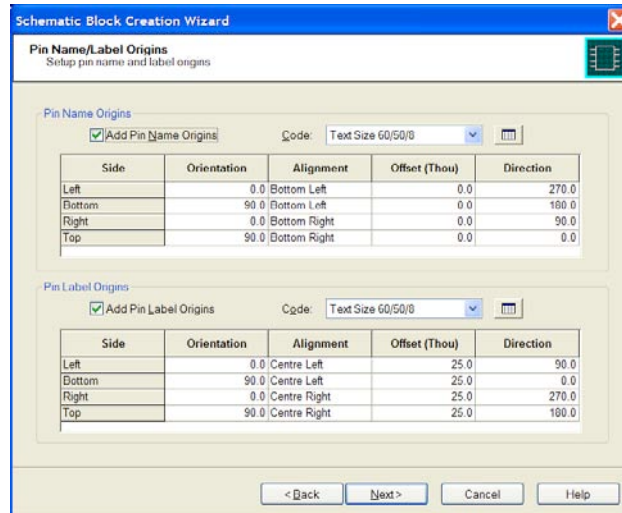


Figure 48: Pin name and Label Origins

Select **Next** to enter the assignments to be used for terminals and outlines (you can use the default values)

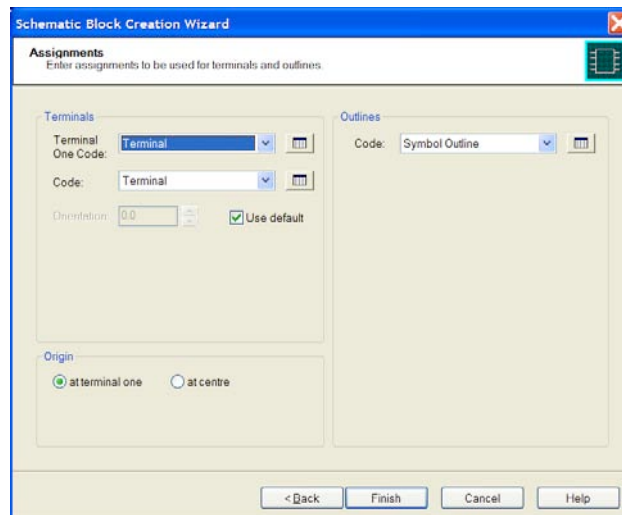


Figure 49: Assignments for Terminals & Outlines

Select **Finish** and the new symbol will appear in the Design Editor window as shown in **Figure 50**.

You can now decide to manual optimize the symbol or **Save** the symbol directly into the library, by selecting from the menu bar **Libraries – Schematic Symbols...** and select **Save Symbol** and click on **OK**

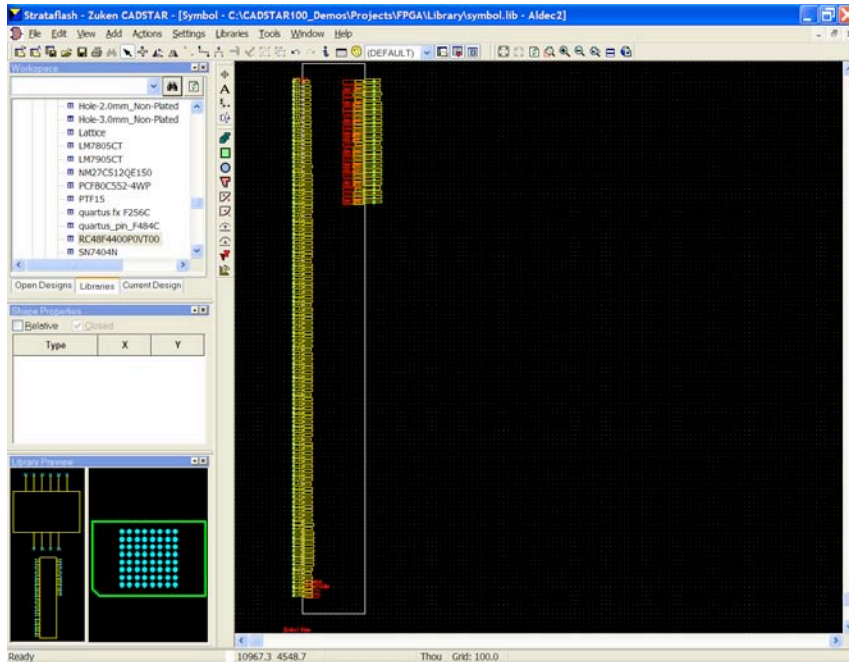


Figure 50: Generated Symbol

CADSTAR Parts Library Editor

Once the symbol has been created we must now create the Part, which will link the created schematic symbol and PCB component (footprint) together.

To do so, select from the menu bar **Libraries – Parts** and Click on **Files...**

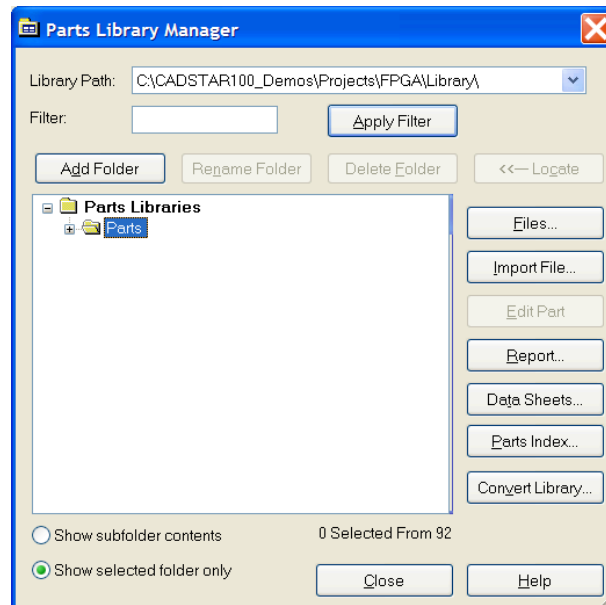


Figure 51: Part Library Manager



Select the library **Parts** from in window and Click on the button **Edit File...**

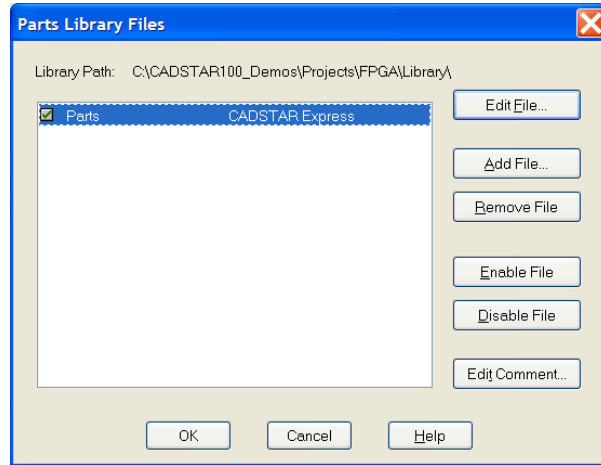


Figure 52: Part Library Files

It might look as shown in Figure 53.

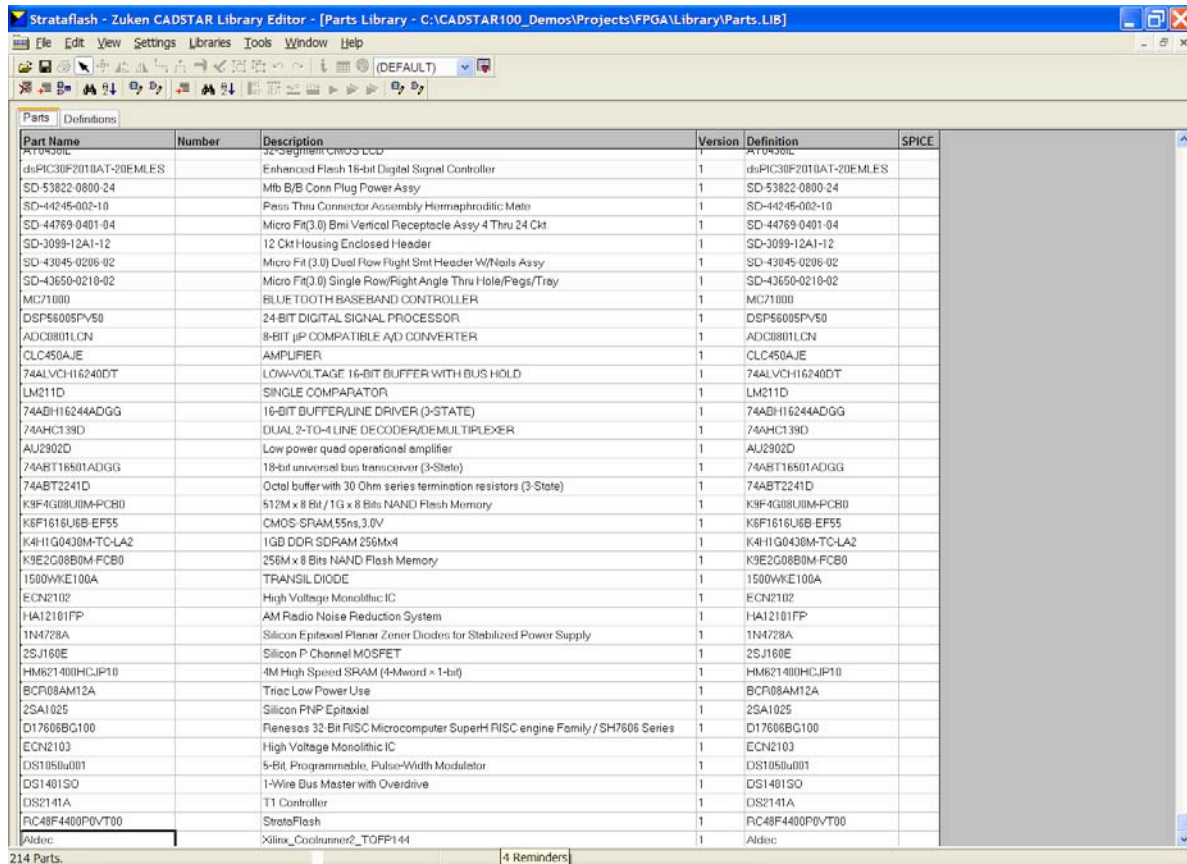


Figure 53: Library Editor



Select from the menu bar **Edit – Add New Row** and fill in the column 'Part Name' for the new created row (for example the Part Name: Aldec). You can fill in any description. And for the Definition you should fill in for example as well Aldec. Once done you can use the right mouse menu and select **Show Part Definition**.

Definition	Component	Max Pin	Stem	FPGA	VALUE	SPICE	Value	Wattage	Tolerance	Manufacturer	Automatic insertion
SD-53822-0800-24	SD-53822-0800-24	24	PL				SD-53822-0800-24			Molex	O
SD-44245-002-10	SD-44245-002-10	10	PL				SD-44245-002-10			Molex	O
SD-44769-0401-04	SD-44769-0401-04	4	SK				SD-44769-0401-04			Molex	O
SD-3099-12A1-12	SD-3099-12A1-12	12	PL				SD-3099-12A1-12			Molex	O
SD-43045-0206-02	SD-43045-0206-02	4	SK				SD-43045-0206-02			Molex	O
SD-43650-0210-02	SD-43650-0210-02	2	SK				SD-43650-0210-02			Molex	O
MC21000	MAPBGA-100	100	U				MC21000			MOTOROLA	IC
DSP56005PV50	TOFP-144	144	U				DSP56005PV50			MOTOROLA	IC
ADC0801LCN	MDIP-20 (N20A)	20	U				ADC0801LCN			NATIONAL_SEMICONDUCTOR	IC
CLC450AJE	SOP-8 (M08A)	8	U				CLC450AJE			NATIONAL_SEMICONDUCTOR	IC
74ALVCH16240DT	TSSOP-48 (DT)	48	U				74ALVCH16240DT			ON SEMICONDUCTOR	IC
LM211D	SO-8 (D)	8	U				LM211D			ON SEMICONDUCTOR	IC
74ABH16244ADGG	TSSOP-48 (SOT362-1)	48	U				74ABH16244ADGG			PHILIPS	IC
74AHC139D	SO-16 (SOT109-1)	16	U				74AHC139D			PHILIPS	IC
AU2902D	SO-14 (SOT108-1)	14	U				AU2902D			PHILIPS	IC
74ABT16501ADGG	TSSOP-56 (SOT364-1)	56	U				74ABT16501ADGG			PHILIPS	IC
74ABT2241D	SO-20 (SOT163-1)	20	U				74ABT2241D			PHILIPS	IC
K9F4G00U0M-PCD0	TSOP160P2000-48 (N)	48	U				K9F4G00U0M-PCD0			SAMSUNG	IC
K6F1616U6B-EF55	TBGA75P7x7-48 (N)	48	U				K6F1616U6B-EF55			SAMSUNG	IC
K4H11G0430M-TC-LA2	TSOP165P1176-66 (N)	66	U				K4H11G0430M-TC-LA2			SAMSUNG	IC
K9E2G08B0M-FCB0	WSOP160P1700-48 (N)	48	U				K9E2G08B0M-V1B0			SAMSUNG	IC
1500WKE100A	CB429-2	2	D				1500WKE100A			SGS_THOMSON	D
ECN2102	ECN2102 (FP-48)	48	U				ECN2102			Renesas	IC
HA12181FP	SOP-16 (FP-16DA)	16	U				HA12181FP			Renesas	IC
1N4728A	DO-41-2	2	Z				1N4728A			Renesas	D
2SJ160E	SC85-3	3	O				2SJ160E			Renesas	M
H1M621400HCJP10	PSOJ-32 (400mils)	32	U				H1M621400HCJP10			Renesas	IC
BCR08AM12A	TO92-3	3	O				BCR08AM12A			Renesas	TI
2SA1025	TO92-3	3	O				2SA1025			Renesas	TI
D17608RG100	LFRGA-176	176	U				D17608RG100			Renesas	IC
ECN2103	TOFP-80 (1.4mmx20mm)	80	U				ECN2103			Renesas	IC
DS1050a001	uSOP-8 (118mils)	8	U				DS1050a001			MAXIM	IC
DS1481SO	SO-14	14	U				DS1481SO			MAXIM	IC
DS2141A	DIP-40 (600mils)	40	U				DS2141A			MAXIM	IC
RC48F4400P0VT00	bgs64 (reflow)	64	U	Y			RC48F4400P0VT00			INTEL	IC
Aldec	cs144 (reflow)	144	D	Y							

Figure 54 : Adding new Parts

Just make sure to fill in the column FPGA a **Y** as value for the row Aldec. Once done you can use the right mouse menu and select **Edit Part Definition**.

You will end up with an empty window that you have to complete, starting with the **Component Tab**.

Click on the **Select** button to choose from the library the component **cs144(reflow)**. For the field **Name Stem**, which is the preferred character the component name to start with, type in for example **D**. When loading the component later on in the schematic design the naming of the component will automatically start with D1, D2, D3 etc.

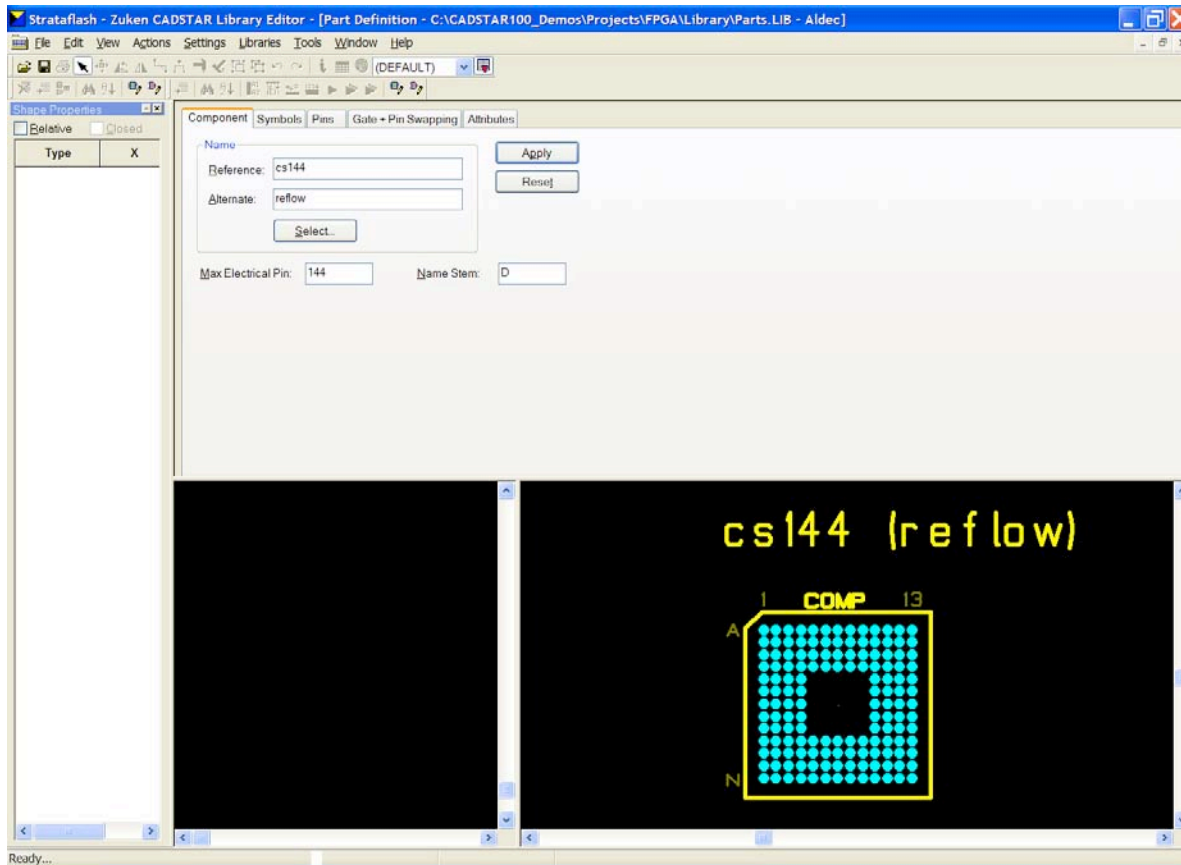


Figure 55: Component CS144

The next step is to select the **Symbols Tab** and select from the menu bar **Edit – Add New Row**. Use the right mouse menu **Select Symbol** to choose from the library the symbol **Aldec**. It will now look as in Figure 56, showing the Symbol and PCB Component:

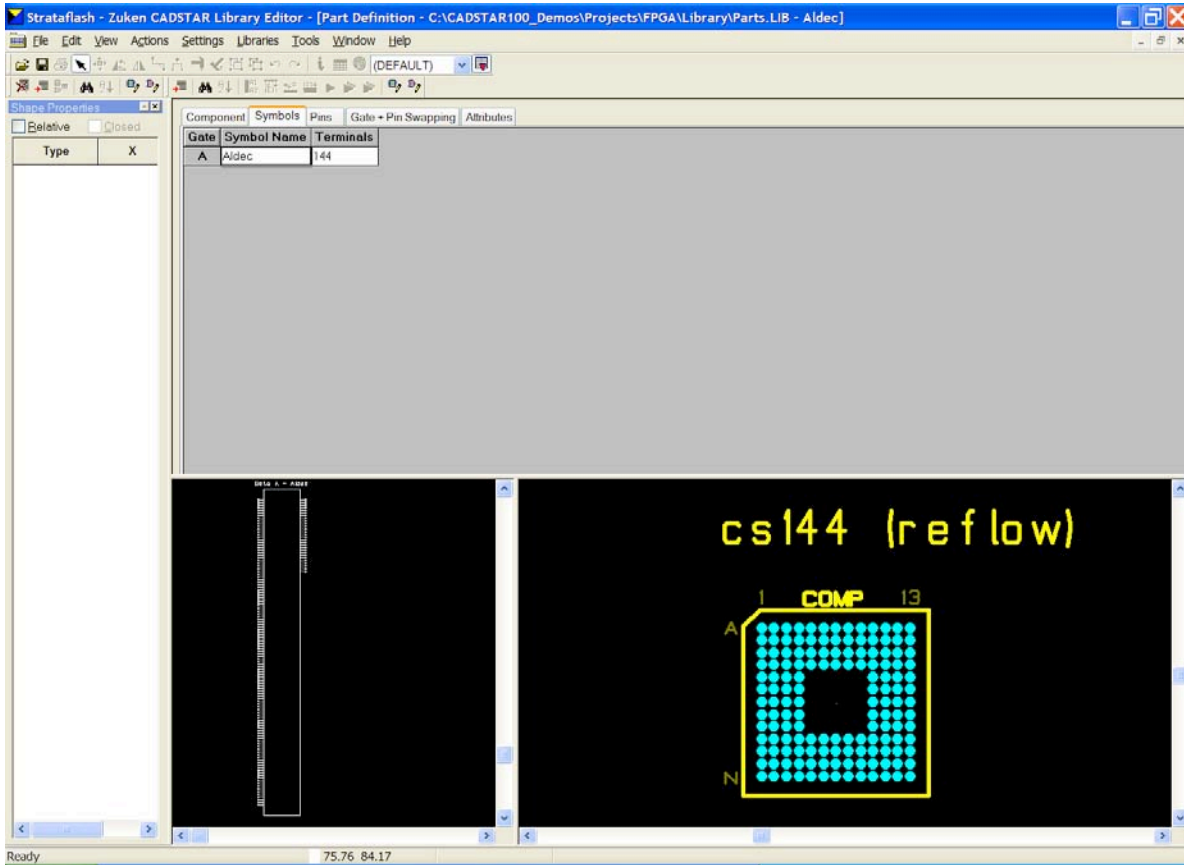


Figure 56: Symbol and PCB Component

The next step is to select the **Pins Tab**. Now we have to start first to Name the physical pins of the PCB Component.

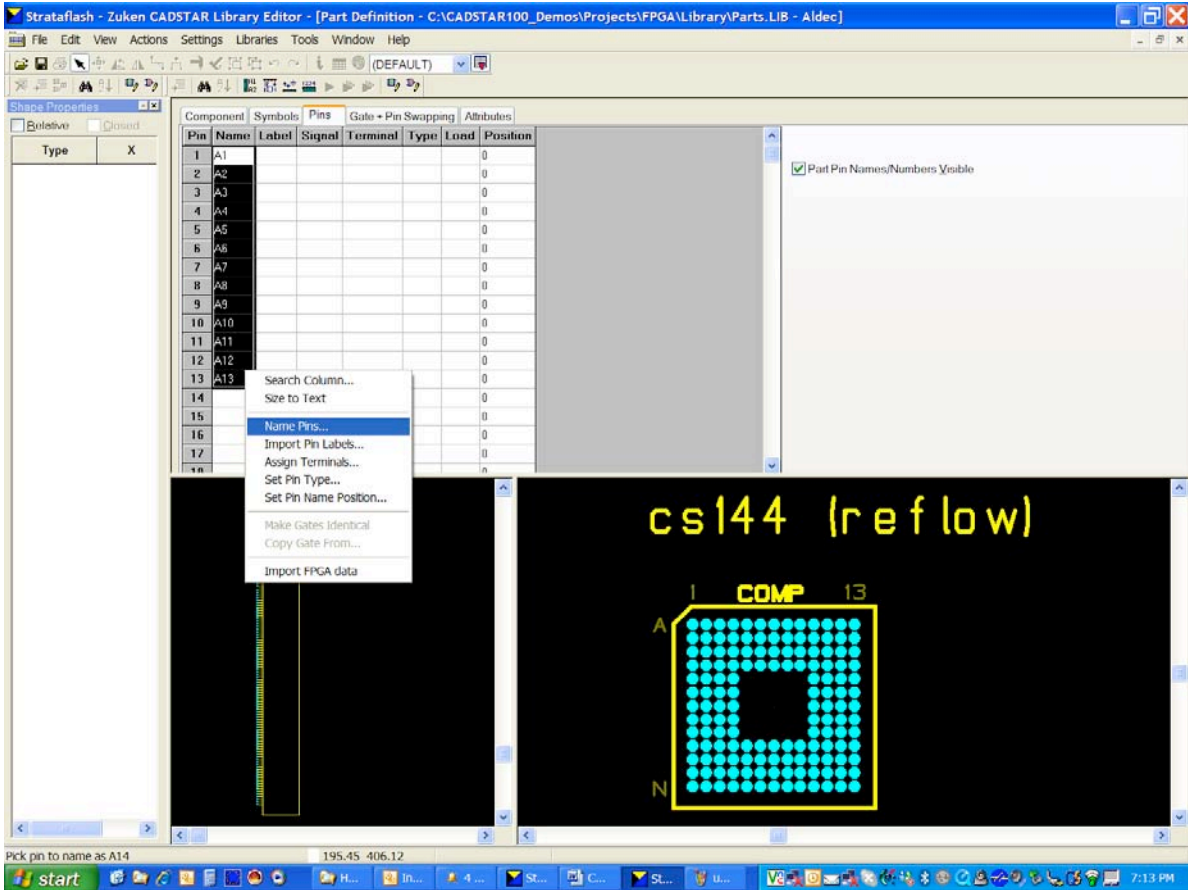


Figure 57 : Naming Physical Pins of PCB Component

To do so you have first to select the fields 1 till 13 of the column **Name** and use the right mouse menu **Name Pins...** and type in **A1** (Increment) as shown in Figure 58

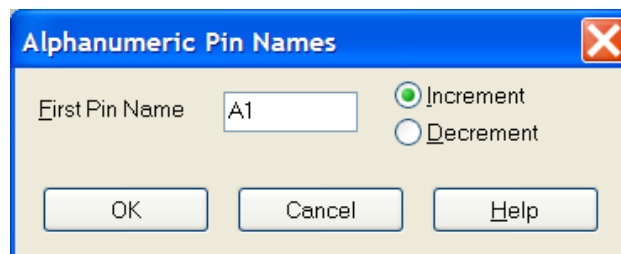


Figure 58: Pin Names



Scroll down and repeat this action for fields 14 till 26 of the column **Name** and use the right mouse menu **Name Pins...** and type in **B1** (Increment)

Scroll down and repeat this action for fields 27 till 39 of the column **Name** and use the right mouse menu **Name Pins...** and type in **C1** (Increment)

Scroll down and repeat this action for fields 40 till 52 of the column **Name** and use the right mouse menu **Name Pins...** and type in **D1** (Increment)

Scroll down and repeat this action for fields 53 till 56 of the column **Name** and use the right mouse menu **Name Pins...** and type in **E1** (Increment)

Scroll down and repeat this action for fields 57 till 60 of the column **Name** and use the right mouse menu **Name Pins...** and type in **E10** (Increment)

Scroll down and repeat this action for fields 61 till 64 of the column **Name** and use the right mouse menu **Name Pins...** and type in **F1** (Increment)

Scroll down and repeat this action for fields 65 till 68 of the column **Name** and use the right mouse menu **Name Pins...** and type in **F10** (Increment)

Scroll down and repeat this action for fields 69 till 72 of the column **Name** and use the right mouse menu **Name Pins...** and type in **G1** (Increment)

Scroll down and repeat this action for fields 73 till 76 of the column **Name** and use the right mouse menu **Name Pins...** and type in **G10** (Increment)

Scroll down and repeat this action for fields 77 till 80 of the column **Name** and use the right mouse menu **Name Pins...** and type in **H1** (Increment)

Scroll down and repeat this action for fields 81 till 84 of the column **Name** and use the right mouse menu **Name Pins...** and type in **H10** (Increment)

Scroll down and repeat this action for fields 85 till 88 of the column **Name** and use the right mouse menu **Name Pins...** and type in **J1** (Increment)

Scroll down and repeat this action for fields 89 till 92 of the column **Name** and use the right mouse menu **Name Pins...** and type in **J1** (Increment)

Scroll down and repeat this action for fields 93 till 105 of the column **Name** and use the right mouse menu **Name Pins...** and type in **K1** (Increment)

Scroll down and repeat this action for fields 106 till 118 of the column **Name** and use the right mouse menu **Name Pins...** and type in **L1** (Increment)

Scroll down and repeat this action for fields 119 till 131 of the column **Name** and use the right mouse menu **Name Pins...** and type in **M1** (Increment)

Scroll down and repeat this action for fields 132 till 144 of the column **Name** and use the right mouse menu **Name Pins...** and type in **N1** (Increment)

If you finished this action then you can use the right mouse menu **Import FPGA Data** as shown in Figure 59.

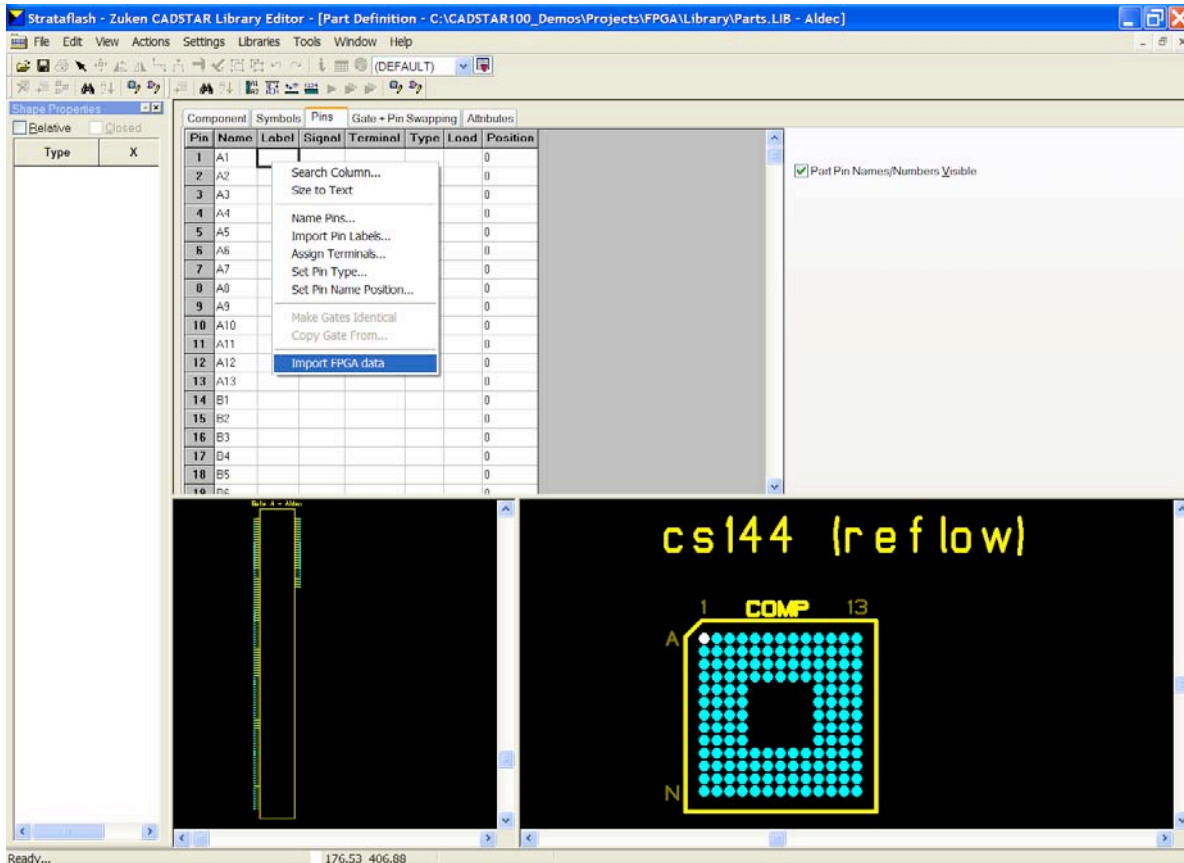


Figure 59: Import FPGA Data

Select Browse... and choose **Aldec.csv** (make sure to enable all options) as shown in Figure 60.

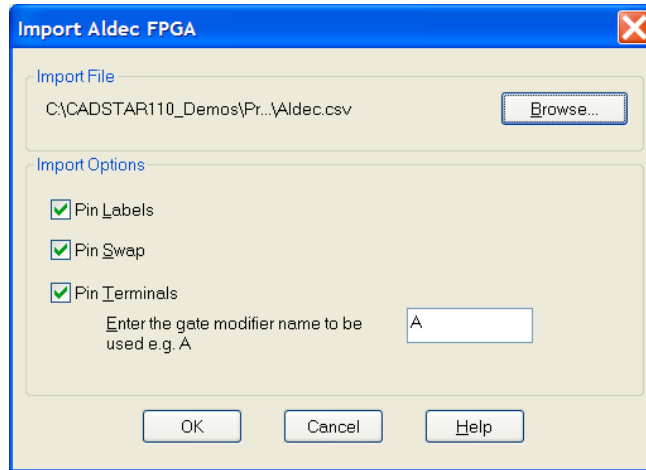


Figure 60: Import Aldec FPGA

By selection **OK** the Pin Labels will be added, Pin Swap Groups will be added and automatically the terminals will be assigned to the appropriate Pin Names.

You can check the Pin Swap Groups by selecting the **Gate + Pin Swapping Tab** as shown in Figure 61.

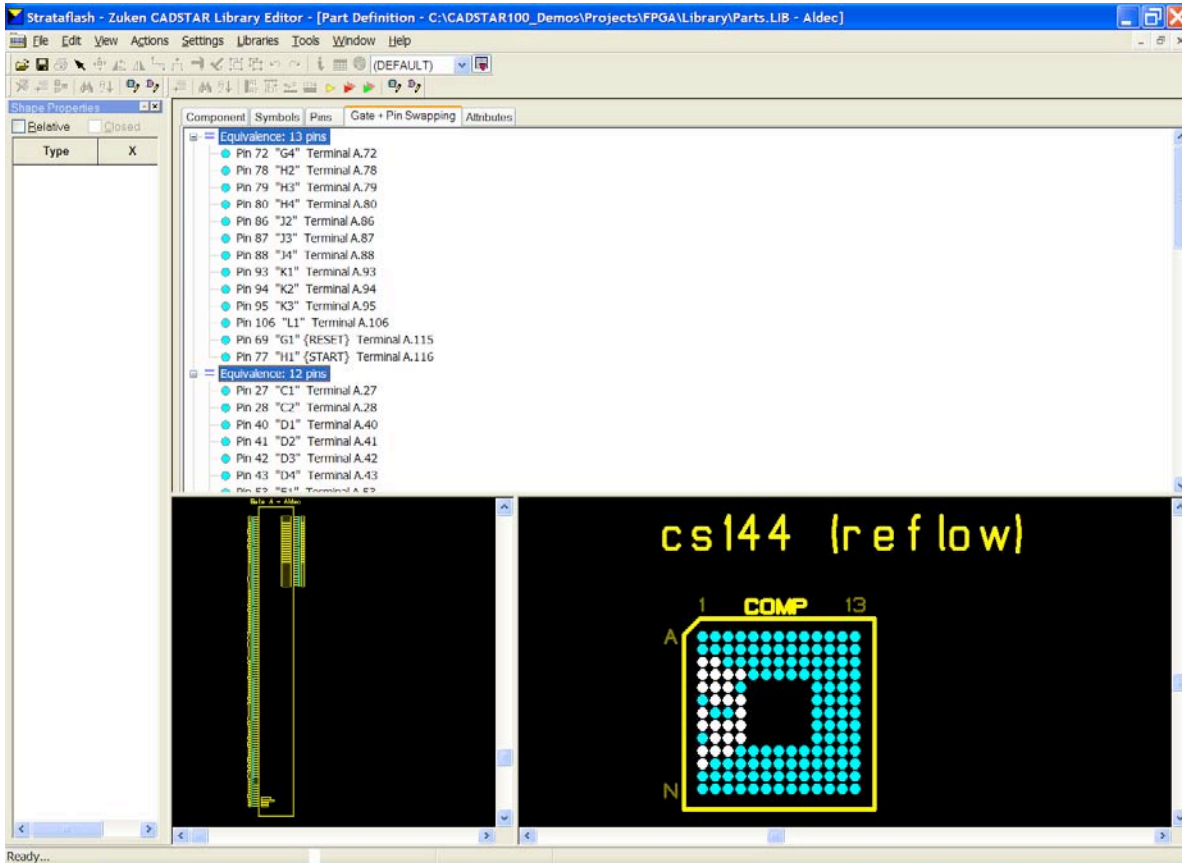


Figure 61: Symbol and PCB Component

Well done! You succeeded in defining a new Part for a FPGA device. You can Save and Exit the Library Editor.

CADSTAR Schematic Design

You can now start your schematics by selecting in the menu bar **File – New – Schematic Design – Form C1**

Call out components from the library



in the workspace window and search for '**Aldec**'

Drag the device from the Workspace window, i.e. highlight '**Aldec**', click on it by using the left-hand mouse button, without releasing the button and drag it out onto the design template. While dragging, you can use the right-hand mouse button for mirror and/or rotation for the placement of a symbol or use a programmable function key like F3 to rotate. You can setup the function keys by selecting **Tools – Customise – Keyboard**



Do the same for another component, like '**Con-Euro96-MA**' and place one connector at the left side of the Aldec FPGA device and the second connector on the right side from the device. When adding the connector into the design you can click on the right-hand mouse button to mirror the connector before placing it.

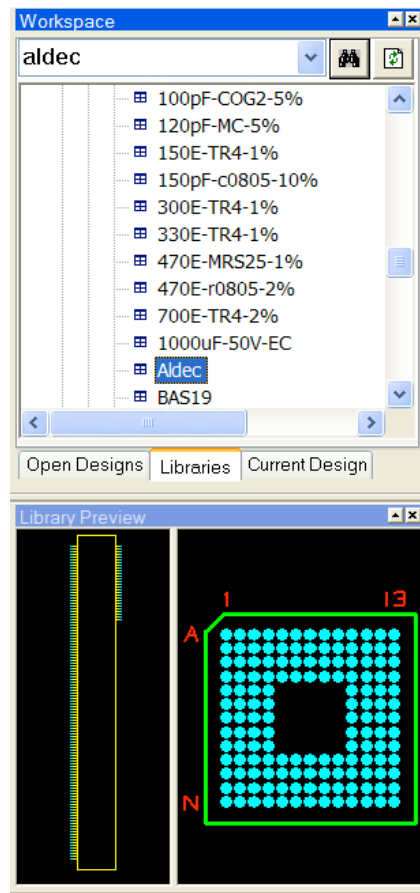


Figure 62: Workspace Window

You can connect two components simply by placing the connecting terminals onto each other. Select the first connector and drag and drop it on top of the terminals of the FPGA device. Then drag it away from the device and notice the connections are created.

Repeat this action for the other connector. It should look like as shown in Figure 63.

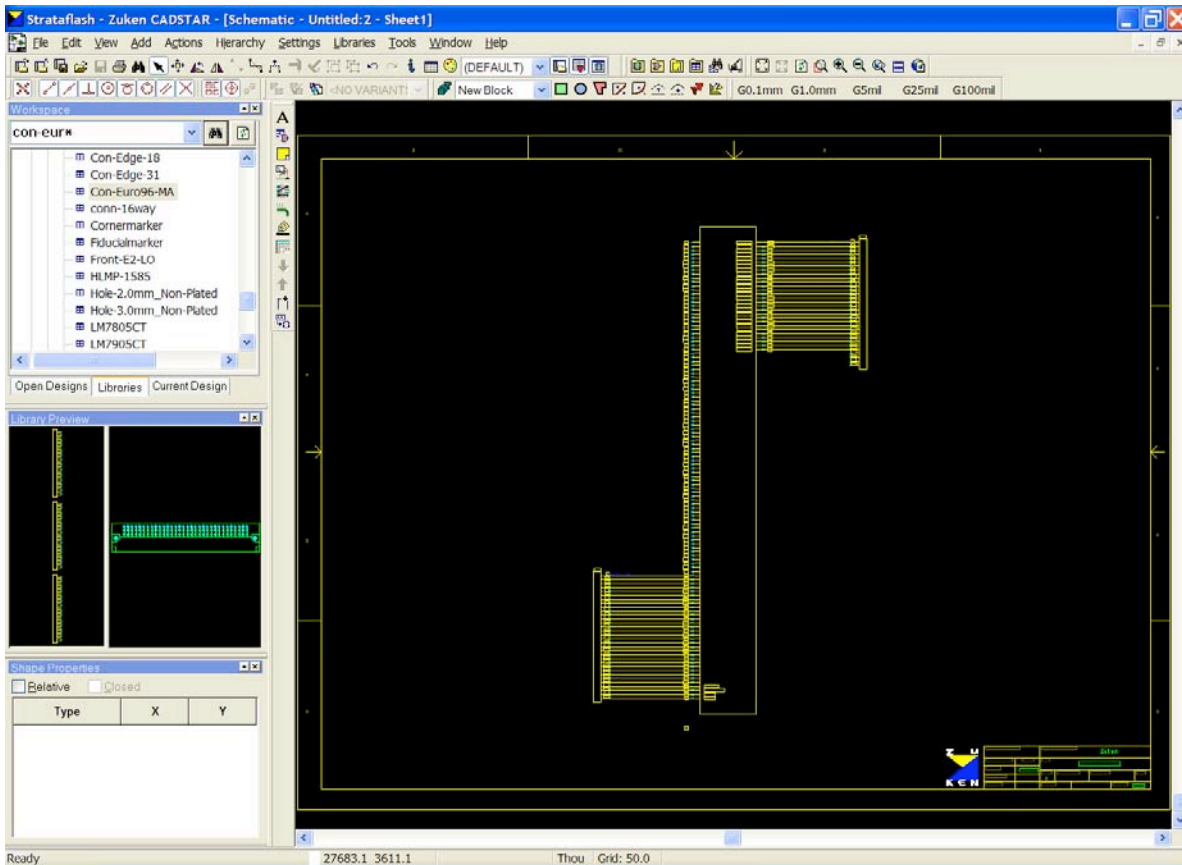



Figure 63: Schematic Design

When completed, save this schematic design 

If you receive in the meantime an updated pin list file from the FPGA designer, you can select the FPGA device in CADSTAR Schematics and use the right-hand mouse button and select '**Import FPGA Data**' to update the FPGA device.

Now you can transfer the schematic to PCB through File → **Transfer to PCB...**, choose '**Defaults.pcb**' as PCB Technology as shown in Figure 64.

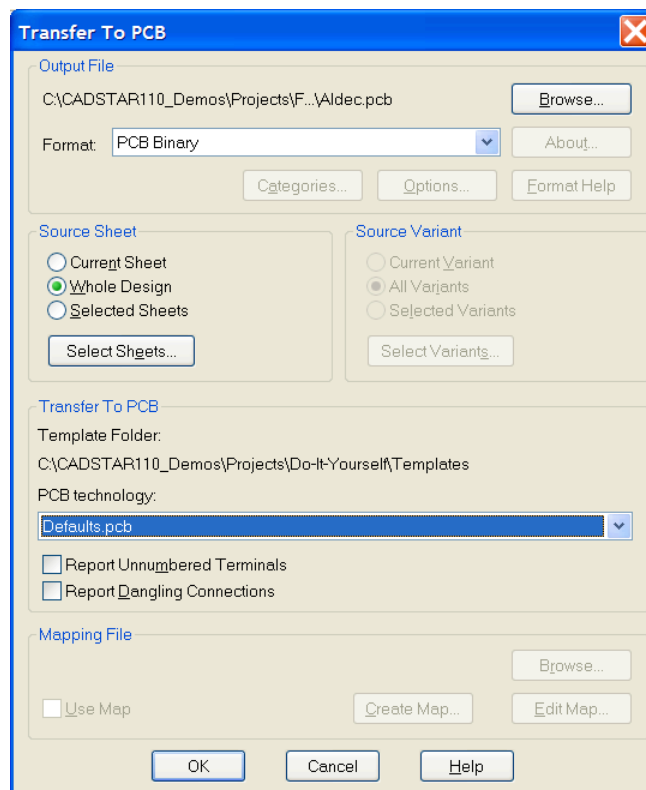


Figure 64: Transfer Schematic to PCB

If you get the following question as in Figure 65, just choose **Add New Code...**

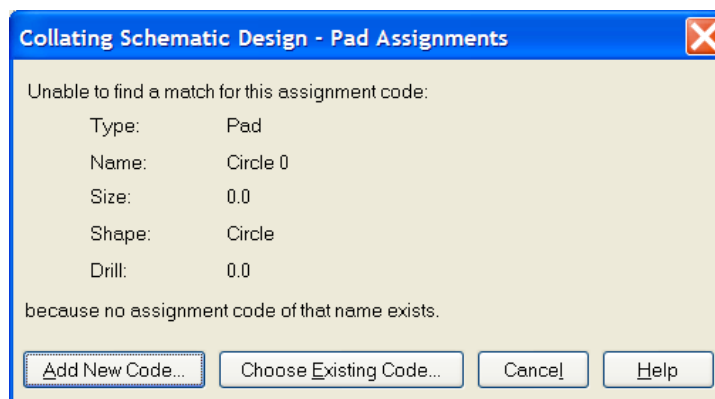


Figure 65: Pad Assignments



CADSTAR PCB Design

You are now in the PCB Layout area with the 2 components stacked onto each other

First, you will have now to draw a PCB outline. Change in the shape toolbar the default **Shape** type to **Board** then click any of the *drawing tool* icons  and begin drawing a rectangular outline big enough to contain the connector and FPGA device.

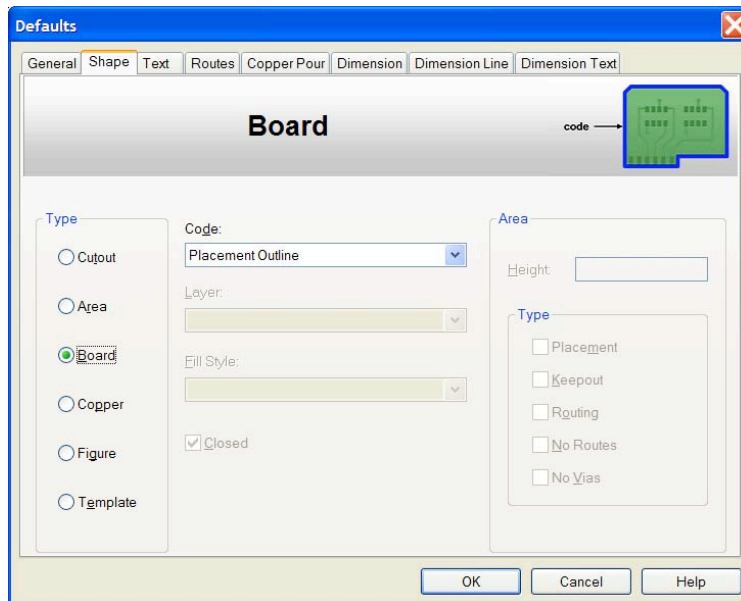



Figure 66: Defaults type for Board

Select  the connector and FPGA device and place them within the board outline.

If you receive in the meantime again an updated pin list file from the FPGA designer, you can select the FPGA device in CADSTAR PCB Design and use the right-hand mouse button and select '**Import FPGA Data**' to update the FPGA device.

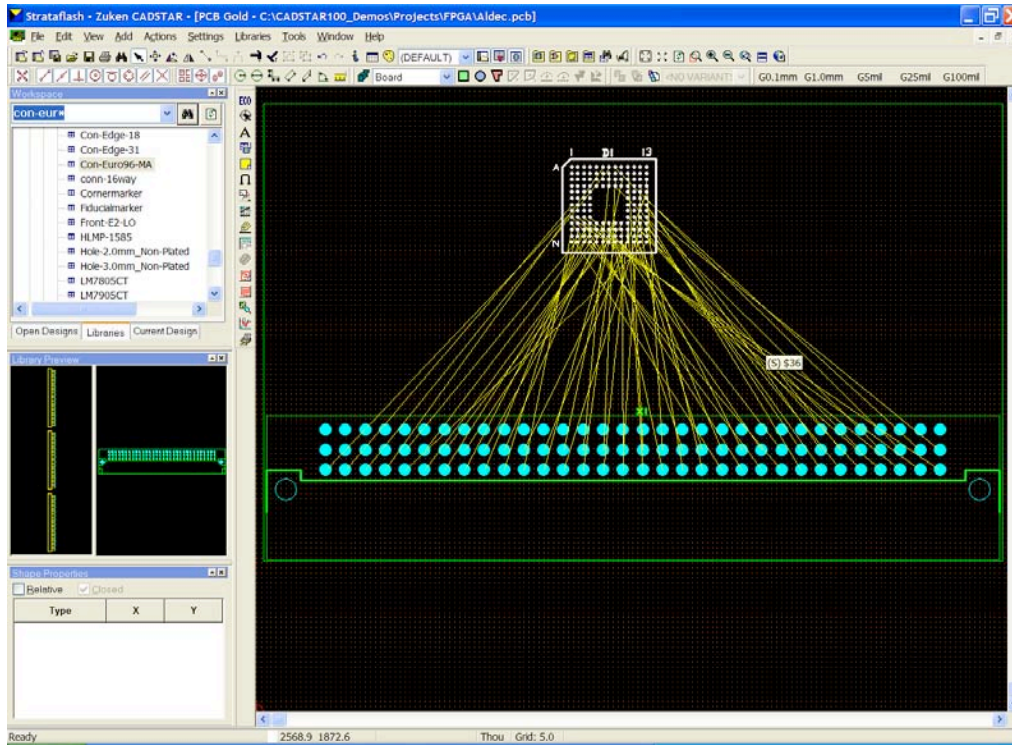


Figure 67: Imported FPGA data

The first optimization of the layout pattern in the PCB design you can do by selecting the FPGA device (make sure the component is highlighted) and select in the menu bar **Actions – Gate and Pin Swap – Automatic Gate and Pin Swap...**, disable Gate Swap and select **Start** as shown in Figure 68.

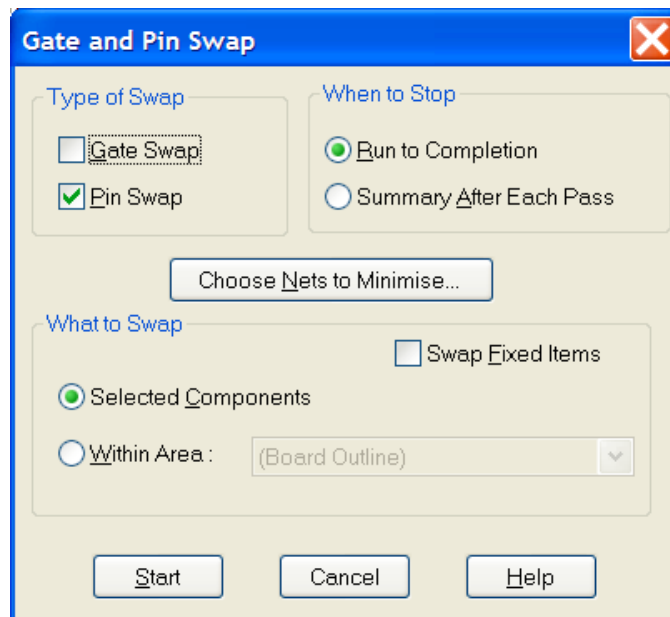


Figure 68: Gate and Pin Swap



Figure 69: Report of Gate and Pin Swap

CADSTAR will try to swap as many as possible pins (as you remember depending on the swapping groups as defined in Edit Part Definition during the creation of the FPGA device) to end up with the shortest possible connection length. This might be of course not the best result as you will find out during routing of the FPGA device.

But before starting to route you have to check and change the default via code by selecting in the menu bar **Settings – Defaults – Routes** and change the Via code to Circle 20/10 as shown in Figure 70.

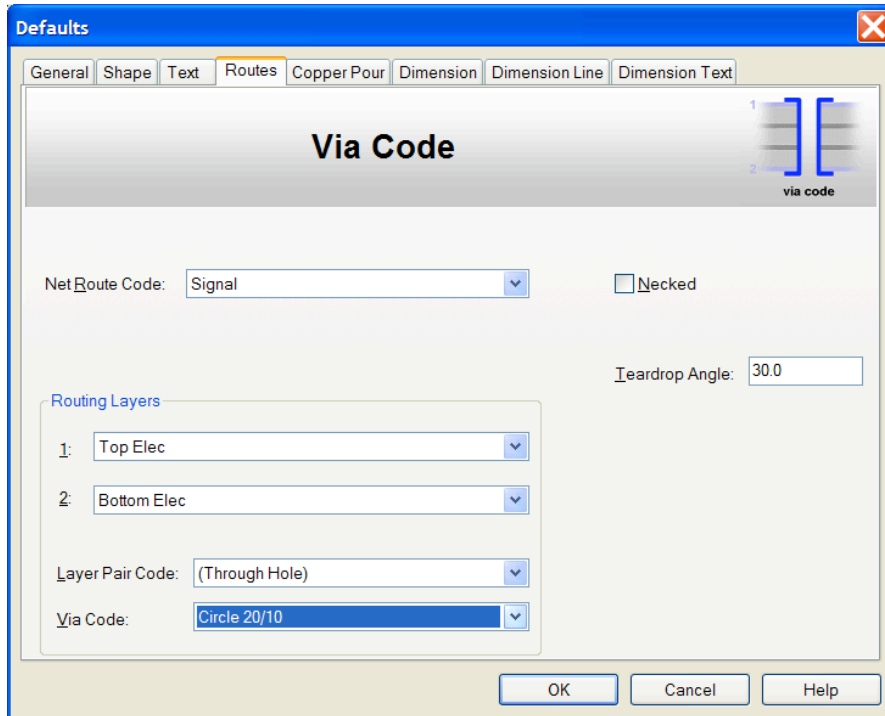



Figure 70: Via Code

CADSTAR P.R.Editor XR (Place and Route Editor)

You can go to the P.R.Editor XR by selecting **Tools** → **P.R.Editor XR** 

When transferring to the P.R.Editor a *RIF Export Option* window will be showed automatically. Ensure to use the settings as shown in Figure 71:

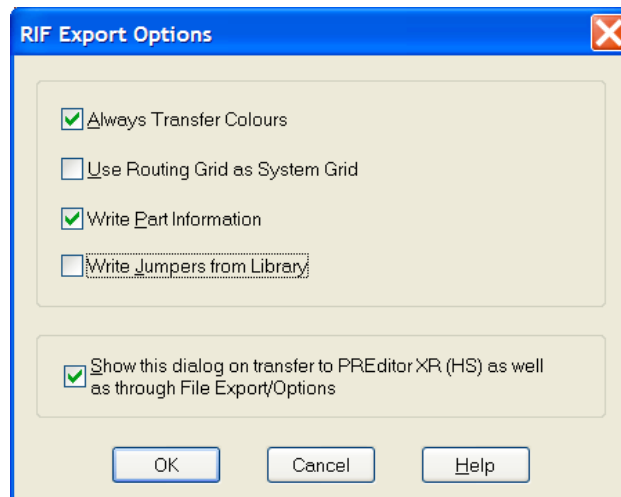


Figure 71: RIF Export Option



You are likely to be at the **P.R.Editor XR** window by now, but before starting any routing or further placement I suggest you to check the Routing Tool Options (CTRL-T). Setting the routing options is very important before any routing! Select **Configure → Routing → Routing Tool** in the menu bar. Ensure the settings are equal to the example and Pin Swap is allowed the routing default routing width is set to Type with the value 6 thousandth of an inch as shown in Figure 72.

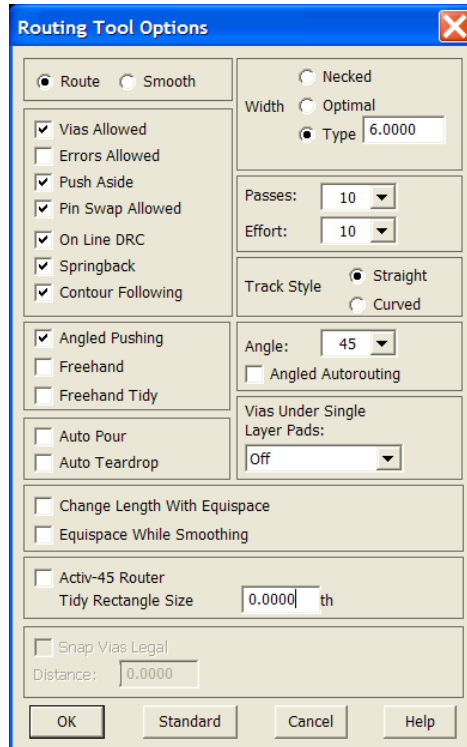



Figure 72: Routing Tool Options

You can now start manual routing by selecting the manual routing icon  and start routing from the connector into the direction of the FPGA device. You will notice a number of flags on top of certain balls of the FPGA device indicating which balls are swappable. If you get closer the connection will automatically swap. You can use multiple layers to optimize and finalize the routing pattern around the FPGA device.

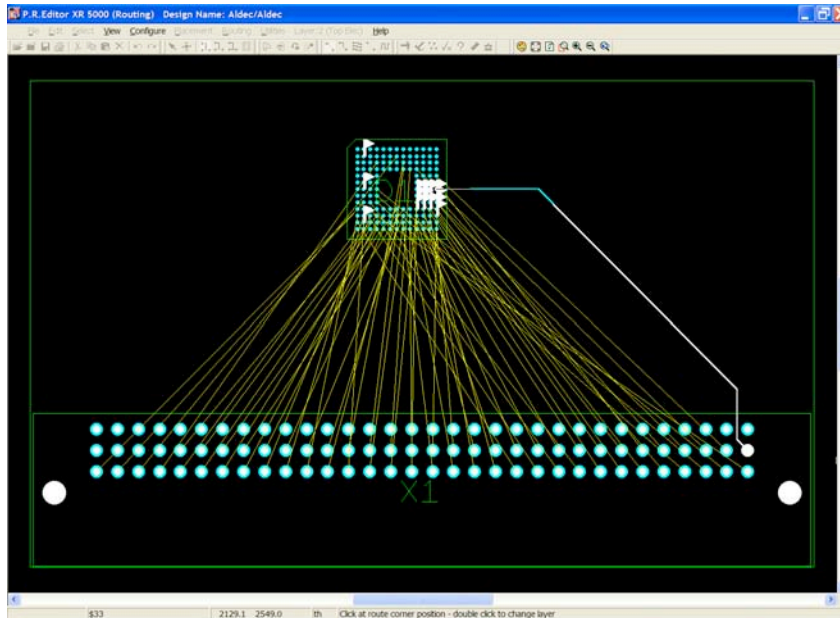


Figure 73: Flags indicating swappable balls

Once you are finished you can Exit the P.R.Editor XR and rebuild all changes into the layout.

You can either back annotate the changes first to the CADSTAR Schematics design or select the FPGA device in CADSTAR PCB Design and use the right-hand mouse button and select '**Export FPGA Data**' to update the FPGA device to write out a new pin file that should be imported to Active-HDL from CADSTAR to update the FPGA in accordance with the latest PCB layout.

Import of SWAP Pin file from CADSTAR to Active-HDL

To import the SWAP Pin file from CADSTAR to Active-HDL in order to update the FPGA design, Click on the **options** to the left of **PCB Interface** Button in the Design Flow Manager.

In the **PCB Interface Options** window, choose the option "Import from PCB". Make sure that PCB Tool is set to **Cadstar**.

Note: Import from PCB reads information about pins from CADSTAR CSV file and write it to the synthesis or implementation constraints. Currently following tools are supported: Actel Designer, Altera Quartus, Lattice ispLEVER, Xilinx ISE (implementation constraints), Synplicity Synplify, PrecisionRTL and Xilinx XST (synthesis constraints)

For the **PCB File** option, Click on **Browse** and select a location where the SWAP pin file is located.

For the Synthesis/Implementation Tool Options select Xilinx ISE and Output File type be "Implementation Constraints". Click on the Browse button for Output File and specify a location and name of the Implementation Constraints file to be generated.

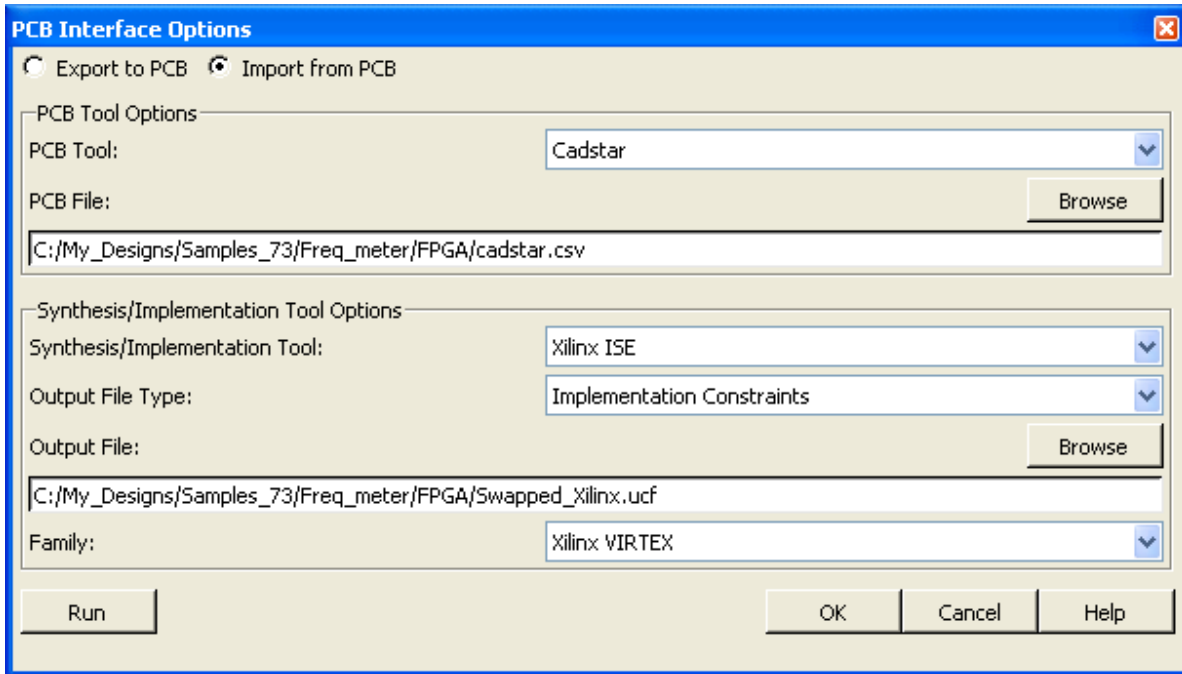


Figure 74: PCB Interface Options- Import from PCB



When you are done with Options, click the “Run” button OR click OK and then hit the icon the design flow manager. You should be able to see successful import from PCB to Active-HDL messages in the console window and right check mark next to the PCB Interface icon in the design flow.

After successful import of the SWAP pin file, you have to run the Xilinx Implementation (Same as explained before in the Tutorial) again in the Design Flow Manager by applying the imported Swap pin Constraints file to update the FPGA in accordance with the PCB design.

Conclusion

After the completion of this Tutorial you should be familiar with all basic procedures in Active-HDL CADSTAR Edition Lite. Feel free to investigate on-line documentation at <http://www.aldec.com/products/active-hdl/> and play with more advanced options! To learn more about the CADSTAR PCB tool, visit www.cadstarworld.com