Importing Structural VHDL Descriptions into Composer Schematics.

This tutorial will take you through the steps required to import structural VHDL descriptions into the cadence design framework. The schematics produced using this will be used to run Layout versus Schematic (LVS) tool to verify that your layouts match the VHDL descriptions. The main tool used for this is called vhdlin and it can be accessed through the CIW.

There are few points to be noted before going to the exact procedure. The vhdlin tool is to be run using icfb command and you will run it from ~/cadence/cell_design. The vhdlin tool can synthesis schematics from vhdl files that are written in structural descriptions only. Also you will have to create the schematics of all the standard cells that you are using in you design manually. When the instances of these standard cells are called in a VHDL file the vhdlin tool will synthesis the schematic for you. The example used in this tutorial is that of an inverter where we will create a standard cell called inv and then call an instance of the inv in the inv1 entity.

There are a couple of modification that you will have to make to your cds.lib file in the cadence/cell_design directory. Add the following lines to the file if they do not exist already. This file is being generated under the assumption that you have been writing all you vhdl code under the ~/cadence/vhdl library.

Lines to be added to the cds.lib file in ~/cadence/cell_design directory.

DEFINE IEEE ${CDS_INST_DIR}/tools/leapfrog/files/IEEE
DEFINE vhdl $HOME/cadence/vhdl
DEFINE std ${CDS_INST_DIR}/tools/leapfrog/files/std
DEFINE basic /cds/ic-qsr/tools.sun4v/dfII/etc/cdslib/basic.

where $HOME is to be replaced by your home directory.

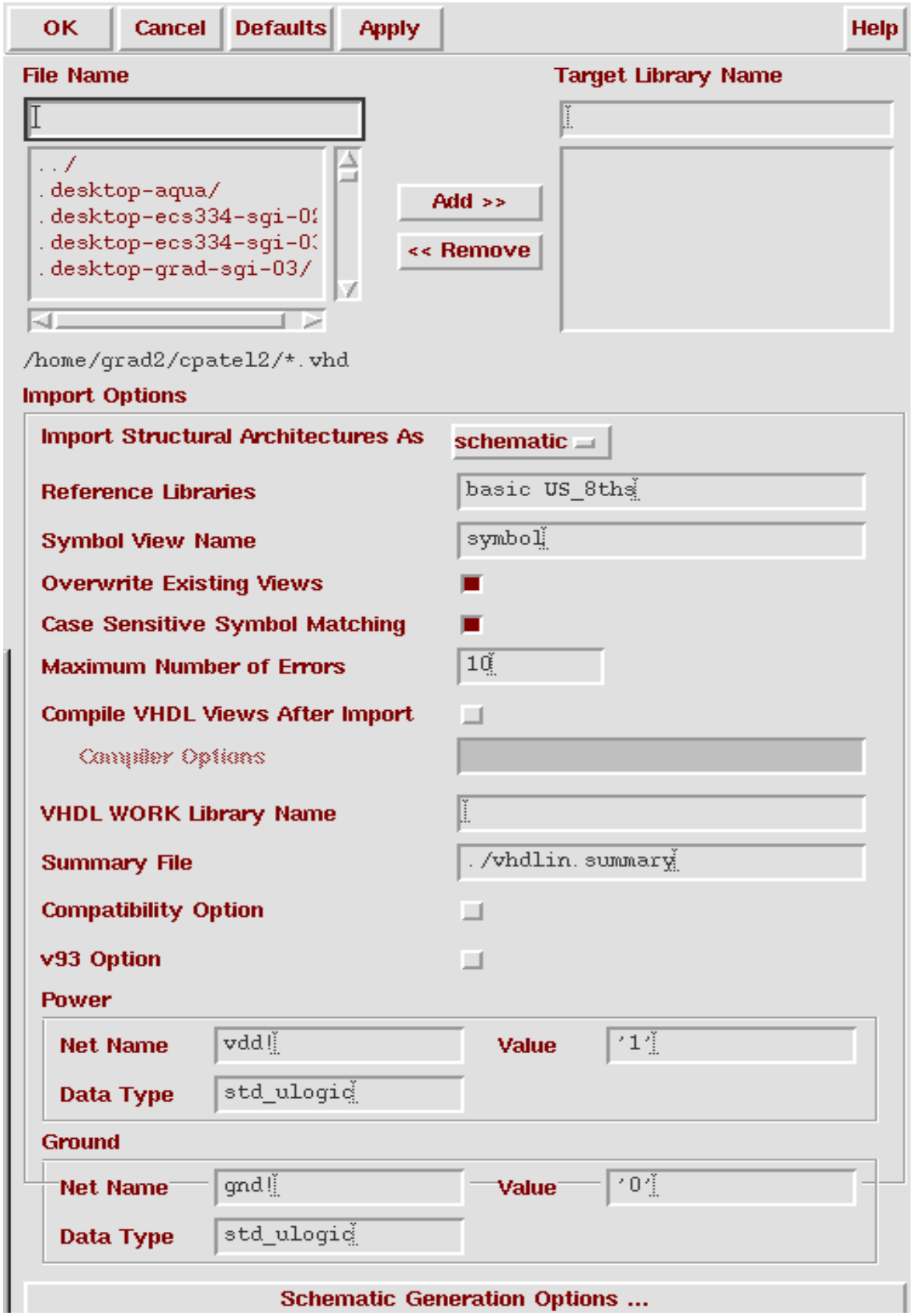
The IEEE and std are the design libraries that you include in your vhdl files. The basic library is sometimes required to import vhdl files into the design framework. The vhdl directory is your vhdl work directory as defined in you hdl.var file. As we were concerned only with simulating vhdl when we created the vhdl library there was no technology file attached to this library. As we are going to use the same library to import you vhdl descriptions we will have to attach a technology file to this library. To do this select the vhdl library in your library manager window and right click to select the attach tech file option. Select the proper technology file (0.6 um AMI in your project) from the browser that pops up and hit OK.

First of all let us import a standard cell called inv into the vhdl library. Create a directory called inv in ~/cadence/vhdl/ directory. Make a directory called vhdl in ~/cadence/vhdl/inv. Change directory to the inv/vhdl directory and create a vhdl.vhd file that has the description of the inv. The example inv VHDL file is given here. Note that you will now have to have the Vdd and Gnd terminals (ports) in all your designs to verify the substrate contacts in the layout. In the vhdl files they can be declared as inputs to each module and can be left unused. If the vhdl compiler gives errors because of this just assign these ports to dummy signals in you architecture so that they do not affect your circuit description.

Example INV VHDL file:

```
--
-- Entity: inv
-- Architecture: structural
-- Author: cpatel2
-- Created On: 10/20/00 at 13:32
--
library IEEE;
use IEEE.std_logic_1164.all;

entity inv is

  port (
    A    : in  std_logic;
    Y    : out std_logic;
    Vdd : in std_logic;
    Gnd : in std_logic);

end inv;

architecture structural of inv is
begin

  Y <= not A;

end structural;
```

After completing the vhdl.vhd file in ~/cadence/vhdl/inv/vhdl directory open icfb from ~/ cadence/cell_design. Go to file and choose import and then VHDL. The import vhdl form will show up. The figure shows the import vhdl form. Under the file name browser in the vhdlin form go and select the file ~/cadence/vhdl/inv/vhdl/vhdl.vhd and hit the Add button on the right hand side of the browser window. The vhdl.vhd file should show up in the list under the Target Library Name. In the Target Library Name field enter the library to import to as vhdl. Select schematic in the Import Structural Architecture as selection menu. Do not change the reference libraries field form the default one. Set the symbol view name to symbol. Turn off the overwrite option. Note that if you want to import an already imported design than you will have to turn this option on. Set the vhdl work library option to vhdl. Turn on the V93 option. You will not have to change any other fields in the form. Hit the OK button after rechecking all the fields in the form. You will get a message saying Started vhdlin in background in your CIW window. On completing the import process the system will come back wither with a successful or error run message. View the log file to find errors. The vhdlin may not work successfully if you have any errors in you vhdl code. Thus to be absolutely sure compile the vhdl code before starting the import process using the commands shown in the VHDL tutorial.


OK
Cancel
Defaults
Apply
Help
File Name
Target Library Name
../
.desktop-aqua/
.desktop-ecs334-sgi-0
.desktop-ecs334-sgi-0
.desktop-grad-sgi-03/
Add >>
<< Remove
/home/grad2/cpatel2/*.vhd
Import Options
Import Structural Architectures As
schematic
Reference Libraries
basic US_8ths
Symbol View Name
symbol
Overwrite Existing Views
Case Sensitive Symbol Matching
Maximum Number of Errors
10
Compile VHDL Views After Import
Compiler Options
VHDL WORK Library Name
Summary File
./vhdlin.summary
Compatibility Option
v93 Option
Power
Net Name
vdd!
Value
'1'
Data Type
std_ulogic
Ground
Net Name
gnd!
Value
'0'
Data Type
std_ulogic
Schematic Generation Options ...

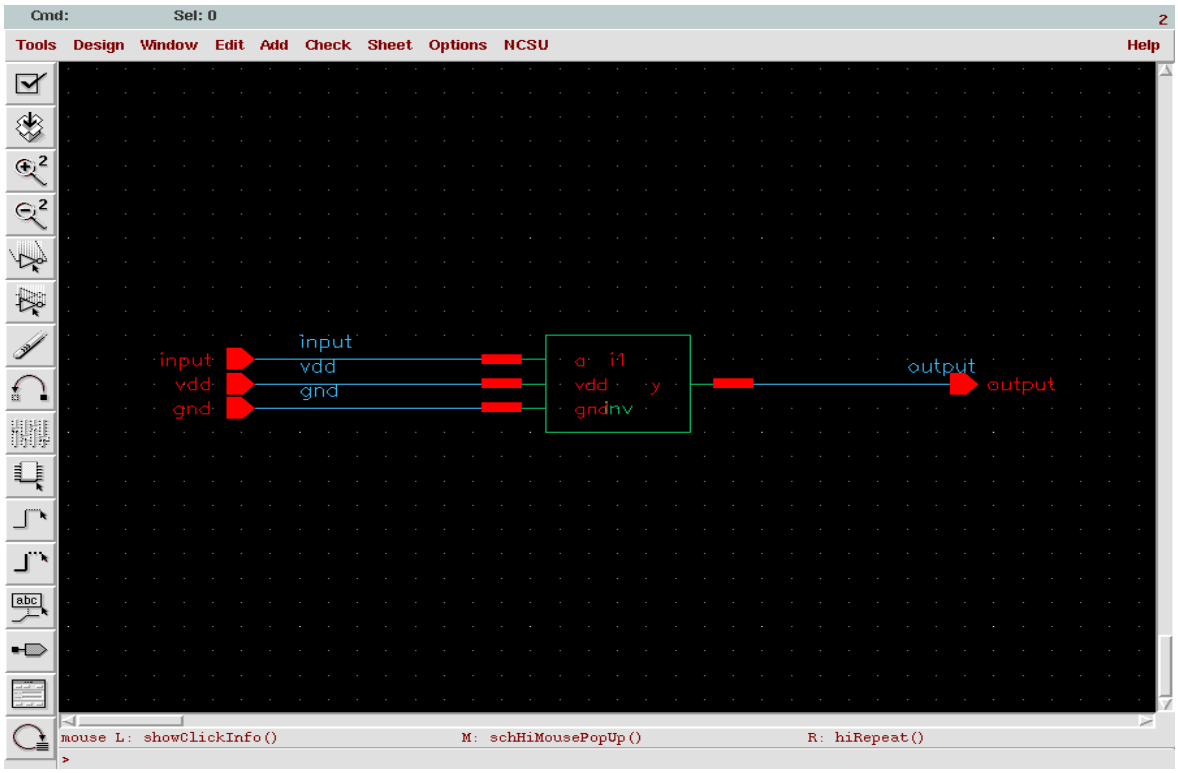
On successful completion of the import process the library vhdl will have a cell called inv with entity, structural, vhdl and symbol views. Check the symbol view to see that all the inputs and output ports are present. Now the next step is to manually generate a schematic for the inv standard cell. Create a new cellview called schematic for the inv and make a CMOS schematic for the inverter using composer schematic tool. Ensure that you can do a check and save on your design with not warnings. All the pins that you place in your schematic have to match with the ports in you symbol view. Thus now you have a standard cell called inv that you can use anywhere you want to place an inverter in your design. In a similar fashion create the same cell views for all the standard cells that you will require in your design.

The next step is to use the inv standard cell and place an instance of it in the inv1 entity and then verify the inv1 layout with the schematic. The example inv1 vhdl file is shown below.

Example inv1 VHDL file:

```
--
-- Entity: inverter
-- Architecture : structural
-- Author: cpatel2
-- Created On: 10/20/00 at 13:32
--
library IEEE;
use IEEE.std_logic_1164.all;

entity inv1 is

  port (
    Input    : in  std_logic;
    Output   : out std_logic;
    Vdd      : in  std_logic;
    Gnd      : in  std_logic);

end inv1;

architecture structural of inv1 is

component inv
    port ( A   : in  std_logic;
           Y   : out std_logic;
           Vdd : in  std_logic;
           Gnd : in  std_logic);
end component inv;

for I1: inv use entity work.inv(structural);
begin

I1: inv port map (Input,Output,Vdd,Gnd);
end structural;
```

Import the inv1 module into a schematic using the same procedure as explained for the inv. Click on the structural view and you should have a schematic as shown in the figure below.



The inv1 schematic will have an instance of the inv standard cell and will have the inputs and outputs as defined in the inv1 file. Click on the inv instance and choose design hierarchy descend read to look at the schematic of the inv standard cell. Click design hierarchy return to return to the inv1 schematic. Now create a layout for the inv1 cell and get the extracted view. Run LVS between the extracted view and the structural view and verify that the netlists match.