FLORIDA INTERNATIONAL UNIVERSITY
DEPARTMENT OF ELECTRICAL ENGINEERING
VLSI & Intelligent Systems Design Lab

Precision RTL Tutorial

Prepared by:

Jaime Montenegro. PhD. Candidate

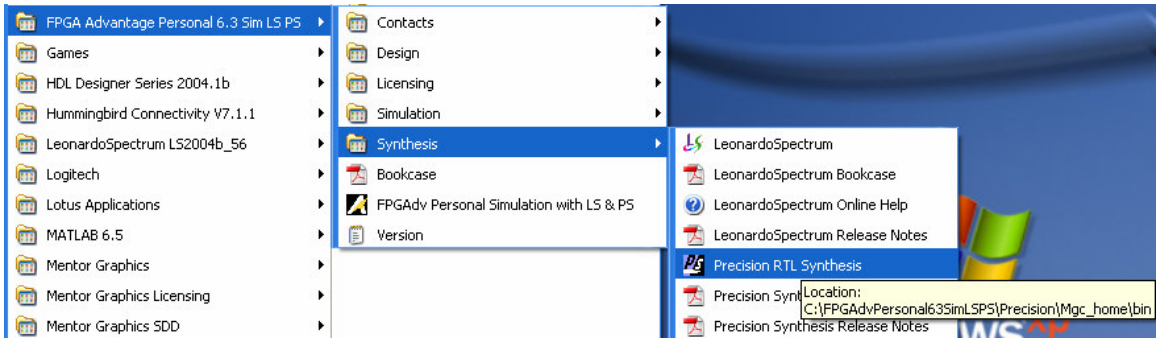
Precision RTL Synthesis is a comprehensive tool suite, which provides design capture in the form of VHDL and Verilog entry, advanced register- transfer-level logic synthesis, constraint-based optimization, state-of-the-art timing analysis, schematic viewing and encapsulated place-and-route. This tutorial describes the synthesis design process of a 4 bit shift register using the Precision RTL Synthesis Graphical User Interface (GUI) and provides information on how to perform synthesis tasks and analysis procedures.

The VHDL file to be synthesized is called *"regdesign.vhd"*; it contains the behavioral description of a 4 bit shift register as shown below.

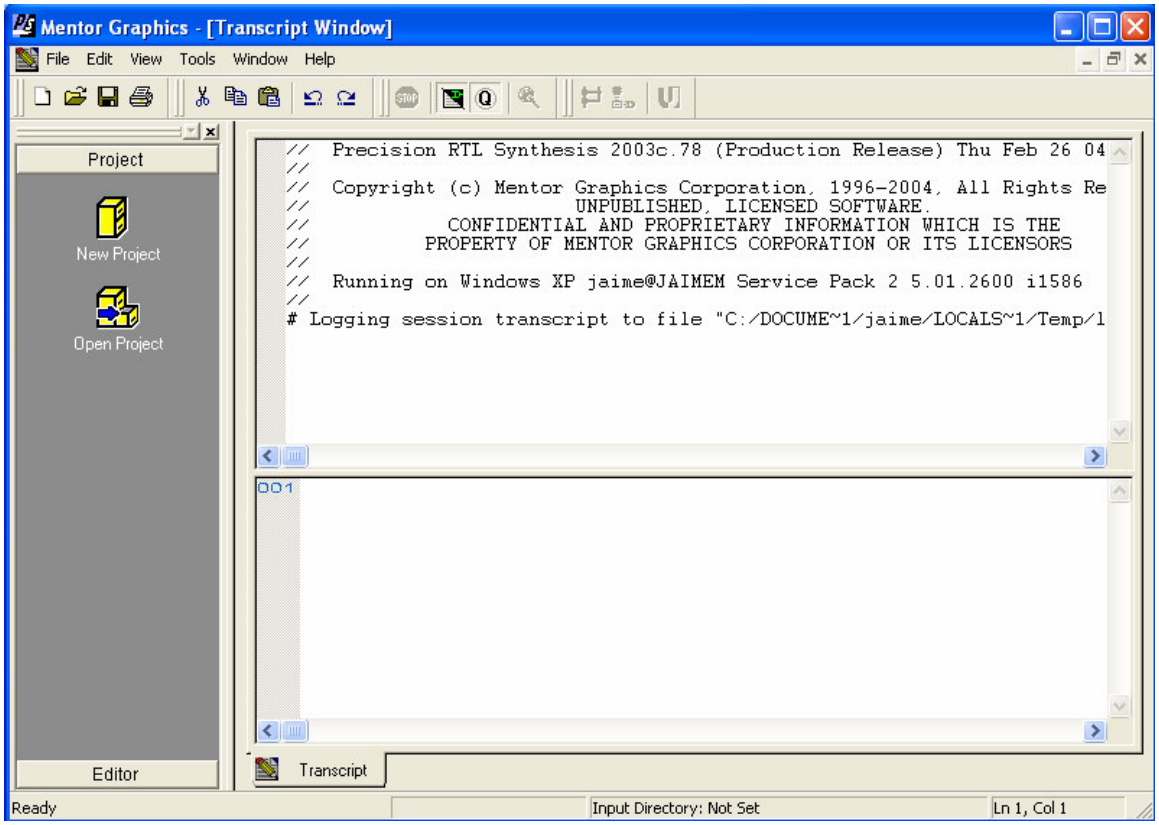
```
LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE IEEE.numeric_std.all;

entity shiftreg is

port(

shiftreg: out std_logic_vector (3 downto 0);-- output of shift register
parallel_data_in:        in  std_logic_vector (3 downto 0);  -- parallel load
serial_data_in:          in  std_logic;  -- serial data input
load:                    in  std_logic;  -- 1 = load, 0 = no load
shift:                   in  std_logic;  -- 1 = shift left, 0 = shift right
rotate:                  in  std_logic;  -- 1 = rotate left, 0 = rotate right
enable:                  in  std_logic;  -- 1 = Shift, 0 = no shift
rotenable:               in  std_logic;  -- 1 = rotation, 0 = no rotation
clock:                   in  std_logic;  -- System clock
reset:                   in  std_logic); -- System reset

end shiftreg;

architecture behavioral of shiftreg is

signal shiftregtemp :    std_logic_vector (3 downto 0);-- temporary shift register signal
BEGIN
    operations: PROCESS (clock, reset)
    BEGIN
        IF (reset = '1') THEN
            shiftregtemp <= (OTHERS => '0'); -- resetting the temporary signal

        ELSIF ((clock'EVENT) AND (clock='0')) THEN

            IF (load = '1') THEN
            shiftregtemp <= parallel_data_in; -- loading the parallel

            ELSIF  ((enable = '1') AND (shift = '1')) THEN
            shiftregtemp <= shiftregtemp(2 downto 0) & serial_data_in; -- Shifting left

            ELSIF  ((enable = '1') AND (shift = '0')) THEN
            shiftregtemp <= serial_data_in & shiftregtemp(3 downto 1); -- Shifting right

            ELSIF  ((rotenable = '1') AND (rotate = '1')) THEN
            shiftregtemp <= shiftregtemp(2 downto 0) & shiftregtemp(3); -- rotating left

            ELSIF  ((rotenable = '1') AND (rotate = '0')) THEN
            shiftregtemp <= shiftregtemp(0) & shiftregtemp(3 downto 1); -- rotating right

            ELSE
            shiftregtemp <= shiftregtemp; -- Hold and keep same output value

       END IF;
       END IF;
    END PROCESS operations;
shiftreg <= shiftregtemp; -- outputing value

end behavioral;
```

In order to synthesize the file *"regdesign.vhd"* using Precision RTL, it is necessary to take the following steps:

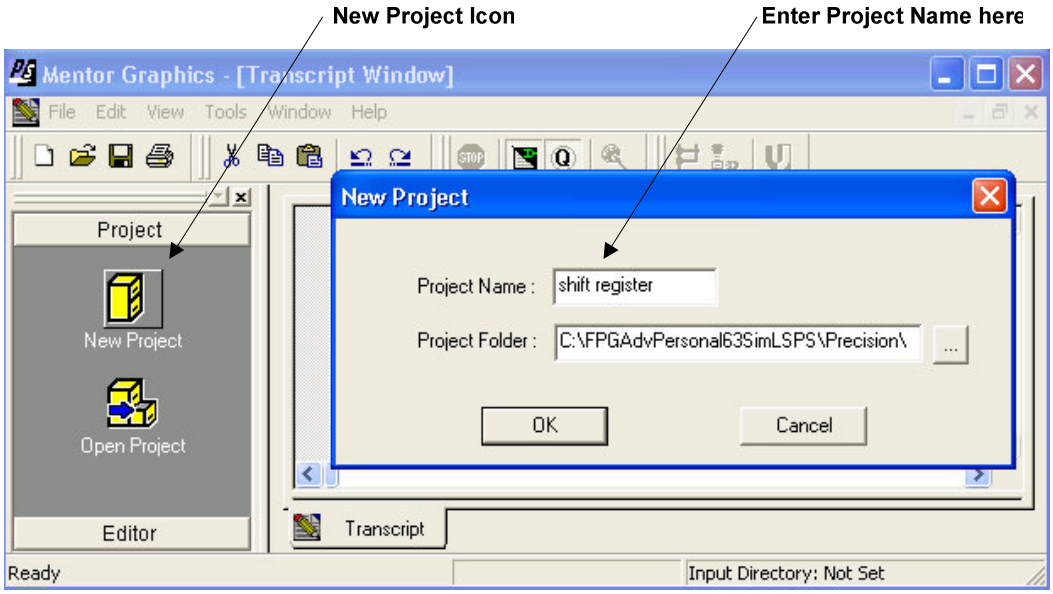
Step 1 (Invoking Precision RTL): find the location of the Precision RTL shortcut and click on it.



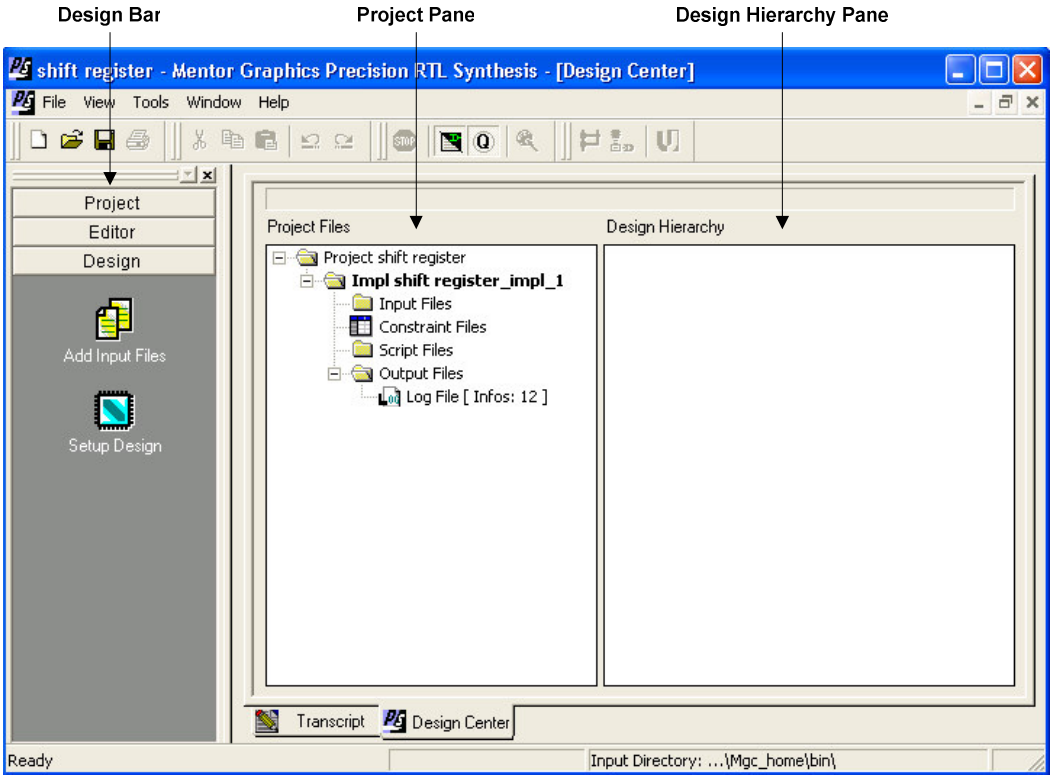
The main Precision RTL Transcript Window should appear:



Step 2 (Creating a Project): click on the New Project icon and enter a Project Name in the specified field. Change the Project folder if necessary. Click OK.



After clicking OK, the Design Center tab is added to the Main Window. The Main Window is divided into three primary control areas, *the Design Bar, the Project Pane, and the Design Hierarchy Pane.*

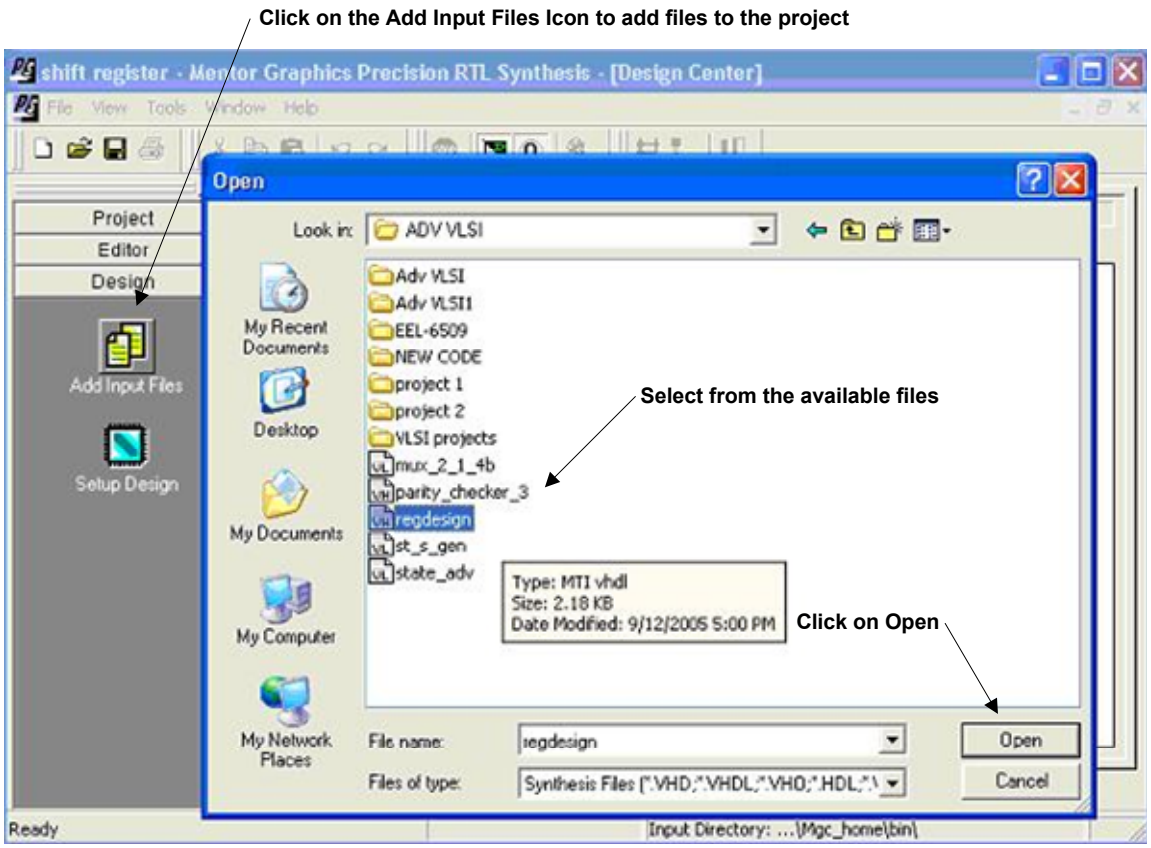


Design Bar: it controls the synthesis flow. The Design Bar provides a visual indicator of the sequential steps in the synthesis flow. Using progressive disclosure, the icons in the Design Bar change depending on where you are in the flow and reveal the possible next steps.

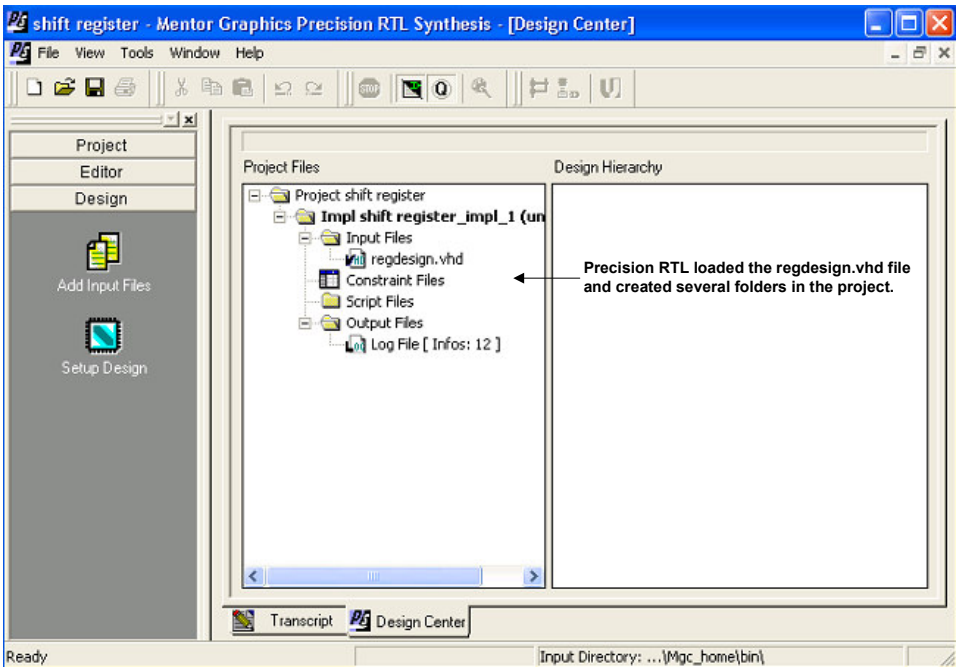
Project Pane: this area gives the user full visibility of the project files and also the ability to manage and access input files and output files.

Design Hierarchy Pane: in this area, several constraints and specifications can be applied to the design objects.

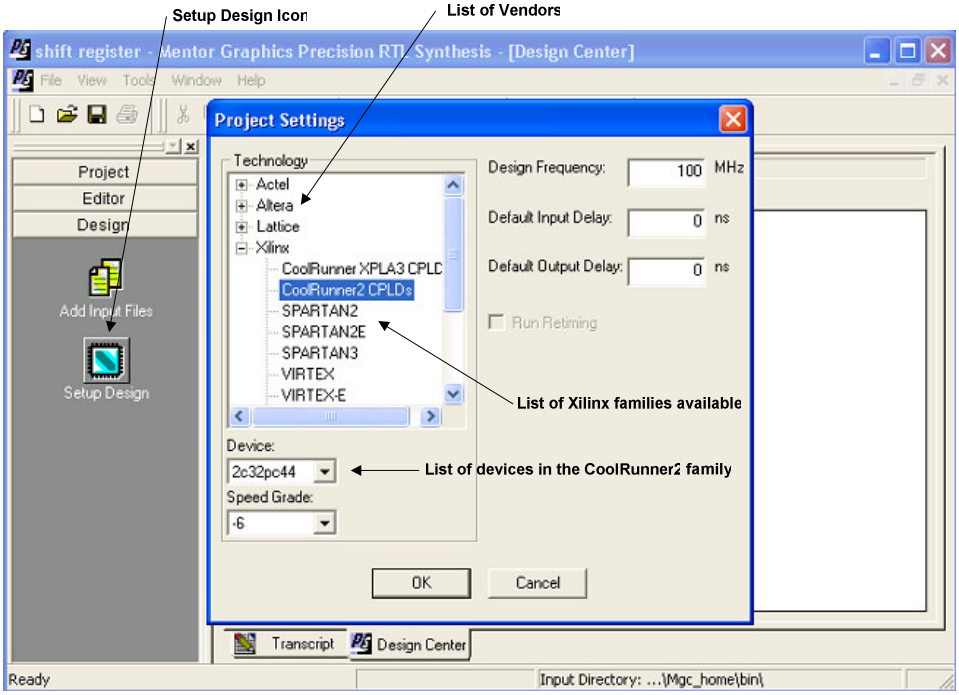
To add an input file to our recently created project (*shift register*), click on the Add Input Files icon, and select the file to be synthesized, in our case the file is *regdesign.vhd*, then click Open.



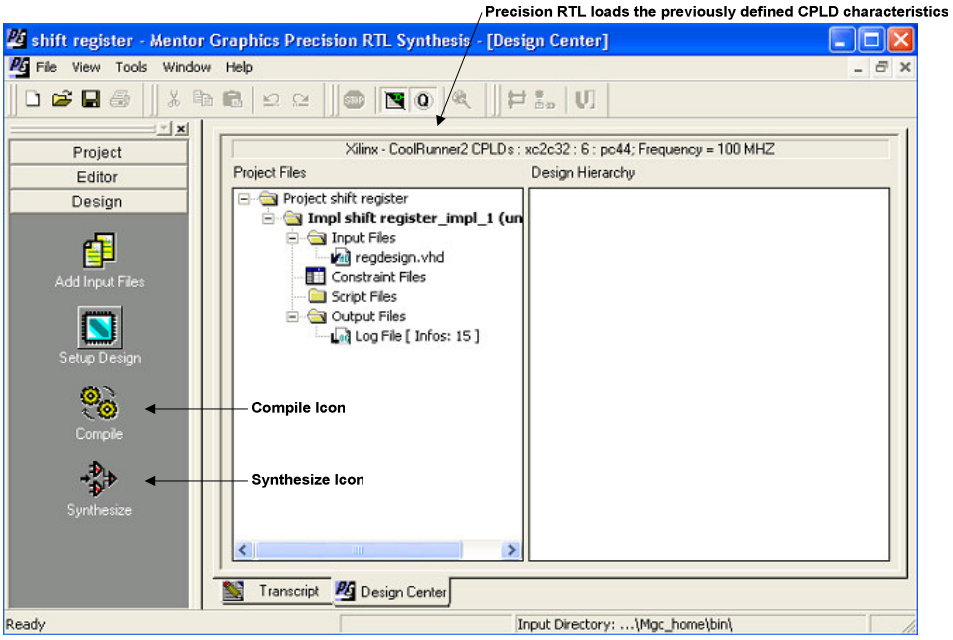
Once this is done, Precision RTL loads the file and shows it in the Project Pane as shown below. It creates several folders where the output and input files will be stored.



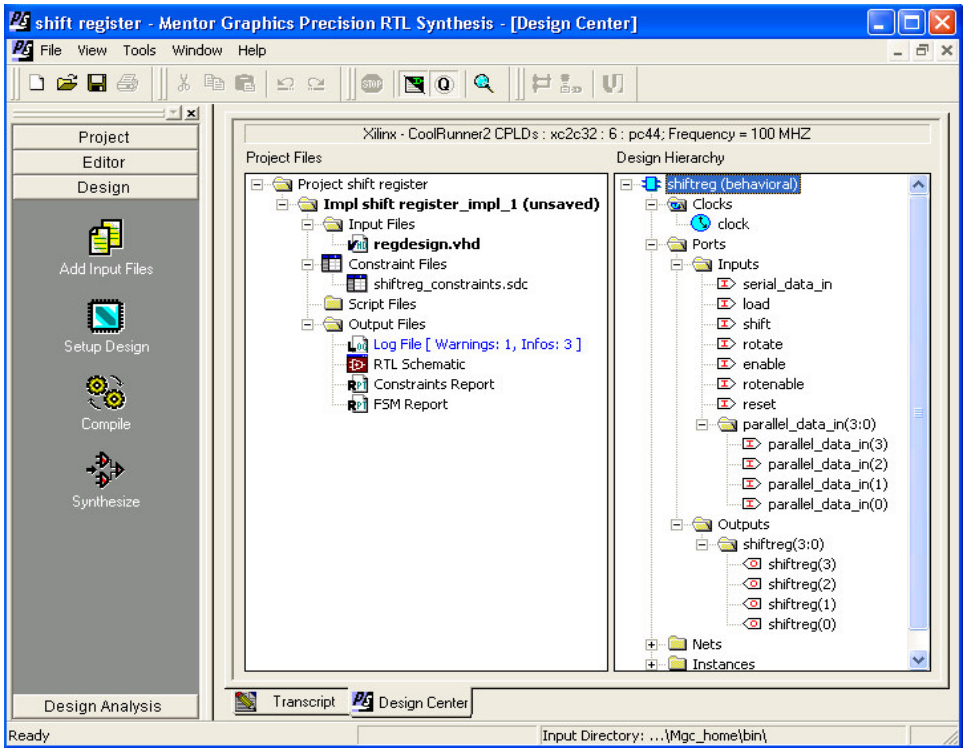
Step 3 (Setting up the Design): our next step is the setup of the design. At this stage, the user specifies the logic device where the *regdesign.vhd* file will be synthesized. For practical purposes we chose a 44 macrocells Xilinx CPLD working at 100 MHz. Click on the Setup Design Icon, Precision RTL gives you the option to select from several vendors, and within the vendors it provides libraries for the different families of the logic devices. Once a device is selected, you can specify the number of macrocells by selecting a device from that family. The speed grade, design frequency, as well as input and output delays can also be specified.



After setting up the design, and clicking Ok, the system loads the characteristics of the previously specified device, and it provides the options of compiling and synthesizing the design by loading the Compile Icon and the Synthesize Icon in the Design Bar area.

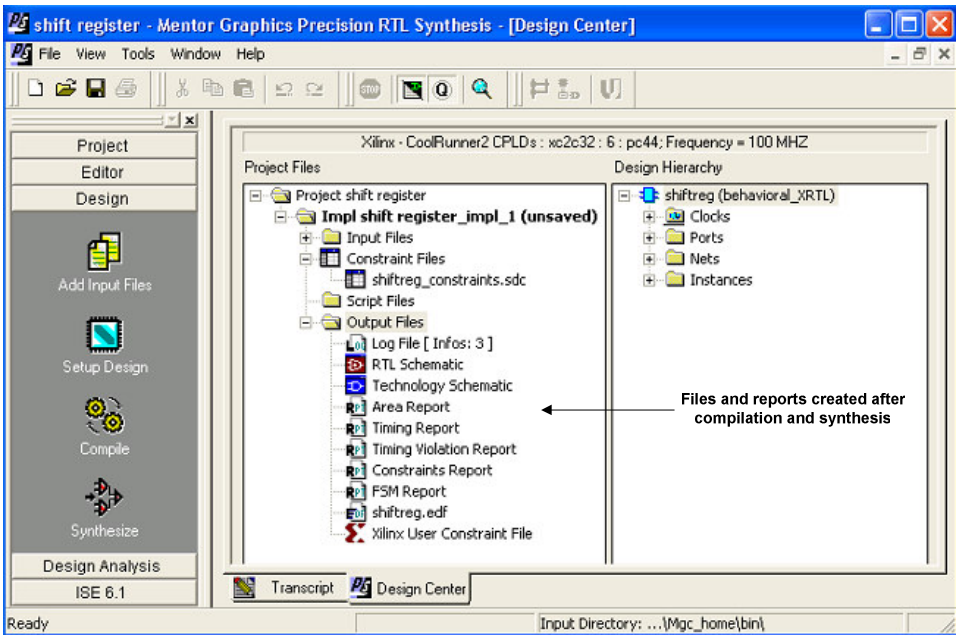


Step 4 (Compiling and Synthesizing the Design): our next step is to compile and synthesize the design. By clicking on the Compile Icon, the *regdesign.vhd* file is compiled and the Design Hierarchy Pane is loaded with the shift register objects.



The Design Hierarchy Pane is a graphical representation of the in-memory design database and allows the user to traverse through the design hierarchy to observe and set constraints such as Input Delay on ports and "Don't Touch" attributes on modules. It allows you to see inputs, outputs, clocks, instances, and nets in the design. You can also use this pane to flatten or preserve the design hierarchy. Objects selected and highlighted may also be highlighted in the schematic viewer. Furthermore, if the selected object initiates cross probing, then that line of code is highlighted in your HDL source code.

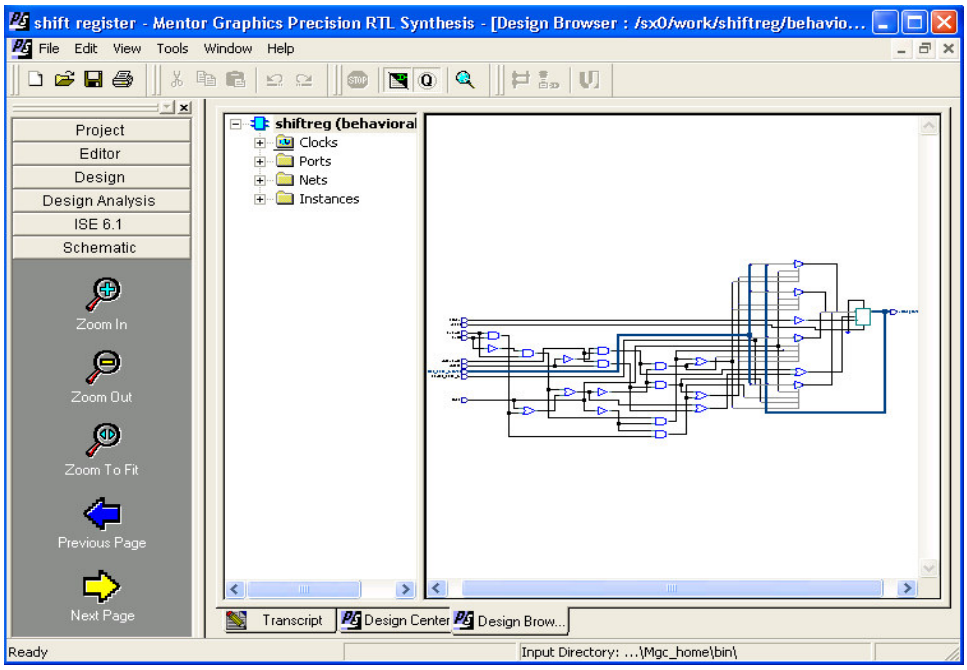
Once the code is compiled, click on the Synthesize Icon to run the synthesis of the code on the specified CPLD. After the synthesis is performed, several files are created and shown in the Project Pane area. Each of these files is a report that provides information about the design.



Step 5 (understanding the information): each of the generated files or reports provides useful information about the design. Feel free to browse and click through each one of those to obtain information about the shift register design.

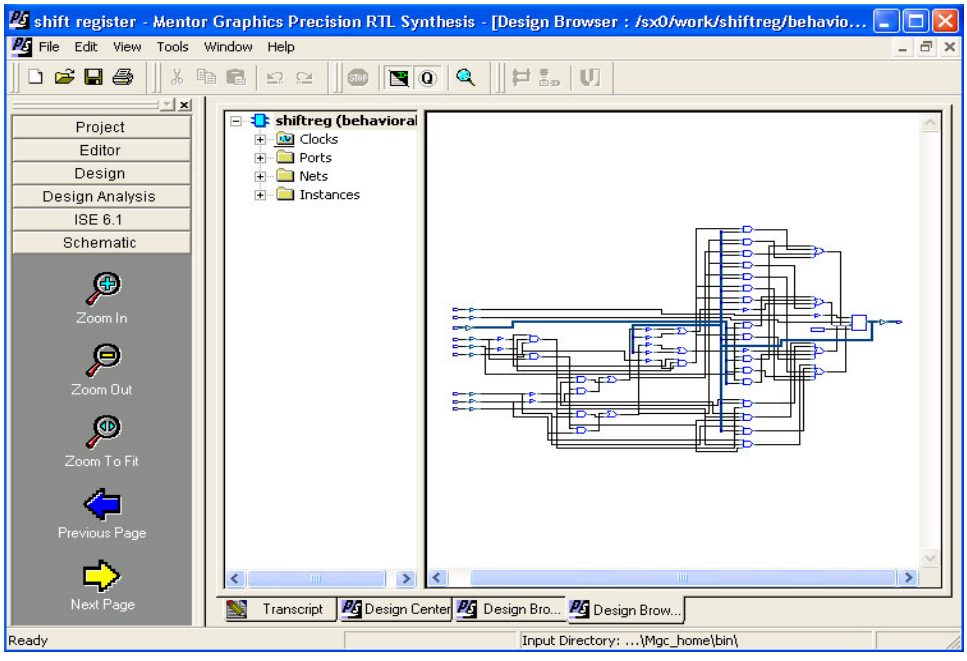
RTL schematic:

By examining the transcript and viewing the RTL Schematic, you can determine whether Precision has created the generic gate-level implementation that you expect. You can analyze both the warnings in the transcript and the quality of the initial results. Even though the output of the compile command is technology independent, the quality of the compiled design is a precursor to the quality of the final technology design. You can zoom in and out of the schematic by pressing the icons located in the Design Bar area. If the design schematic does not fit on a single page, you can browse through the pages with Previous and Next Page icons.



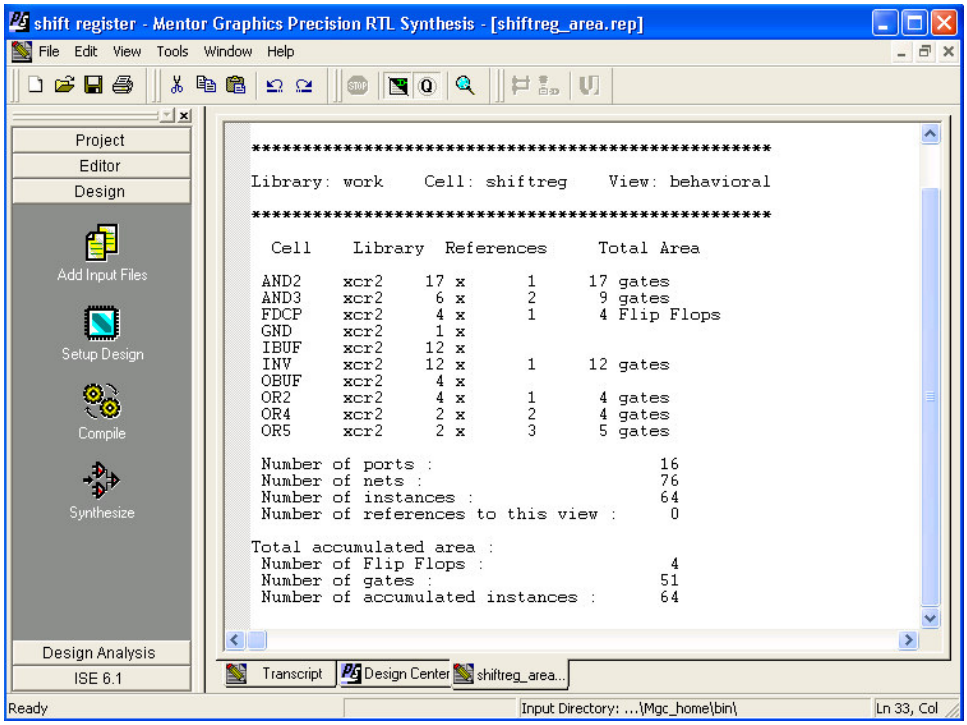
Technology schematic:

To view a gate-level design after synthesizing, double-click on the Technology Schematic file in the Project Files pane of the Design Center window. After Synthesis, a technology-mapped database is created in addition to the generic database. When you double click on the Technology Schematic file, a schematic of the selected CPLD (Xilinx 44 macrocells CoolRunner2) design comes up. Technology schematics show where and how device specific resources are utilized.



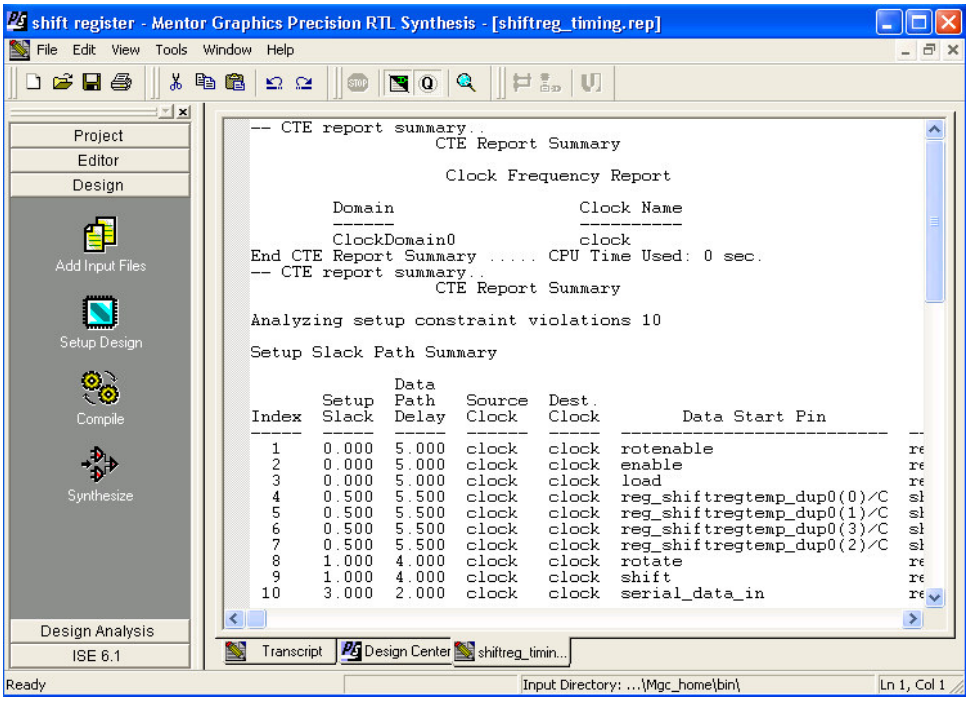
Area report:

This report is used to examine the quality of the results. You can check for unintended latches, and inferred flip-flops. In FPGA designs, synthesis tools infer latches due to incomplete signal assignments. Unintended latches have two negative affects on the design. One, they increase the size of the design because these sequential cells prevent the optimizer from combining boolean logic on either side of the latch. Two, unintended latches block timing analysis. Since synthesis tools only optimize boolean logic, it is very important that you verify that each block is inferring the approximate number of flip-flops that you expected. Once a flip-flop or latch is part of the design, the synthesis tool will not optimize it away as long as it affects the design. In our design 4 flip-flops are being used, along with other logic gates, such as inverters, OR, and AND gates.



Timing Report:

Timing reports display timing paths through hierarchical boundaries.



Timing Violations:

This report provides information that verifies if all timing constraints were met.