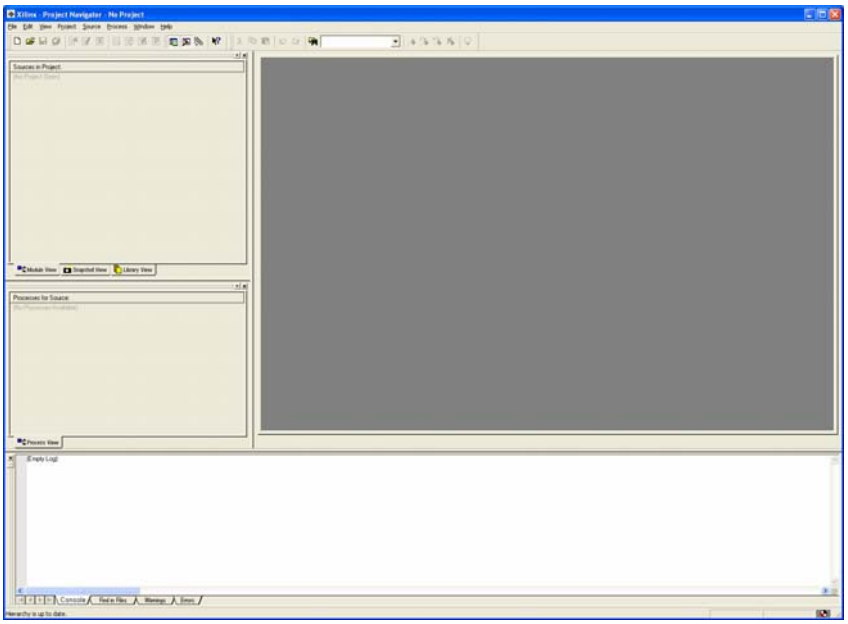
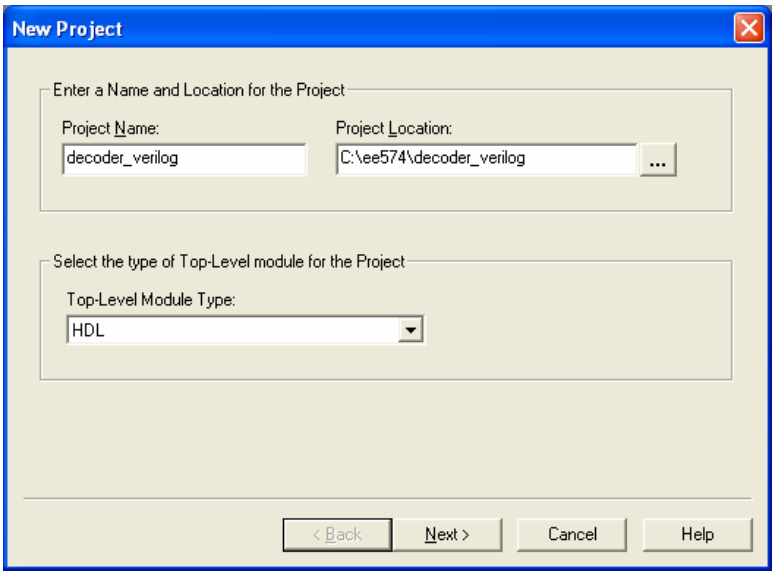
Spartan 3 Starter Board Tutorial - Verilog
(Simple decoder Design synthesized and loaded to board)
Jim Duckworth, March 2005, WPI.

Start Xilinx **Project Navigator**:



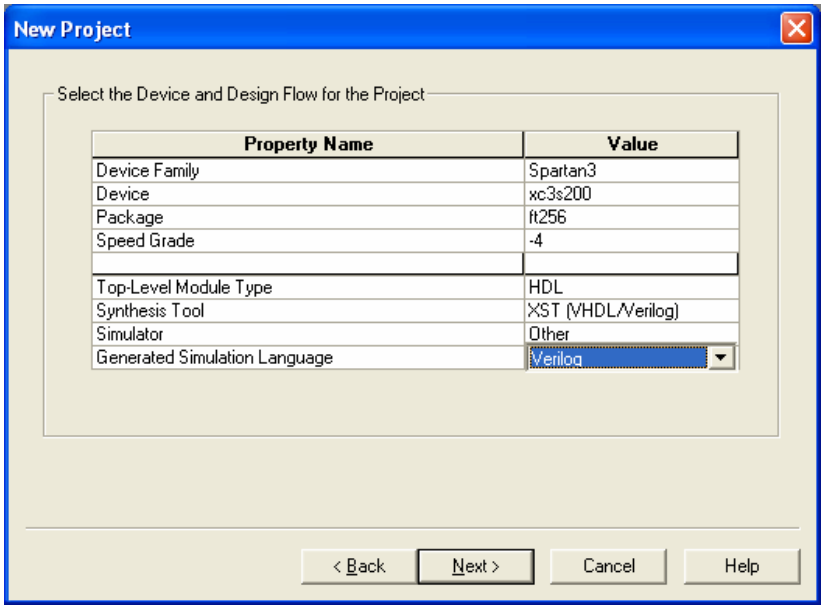
Select **File => New Project**

Select a project location and project name, for example:

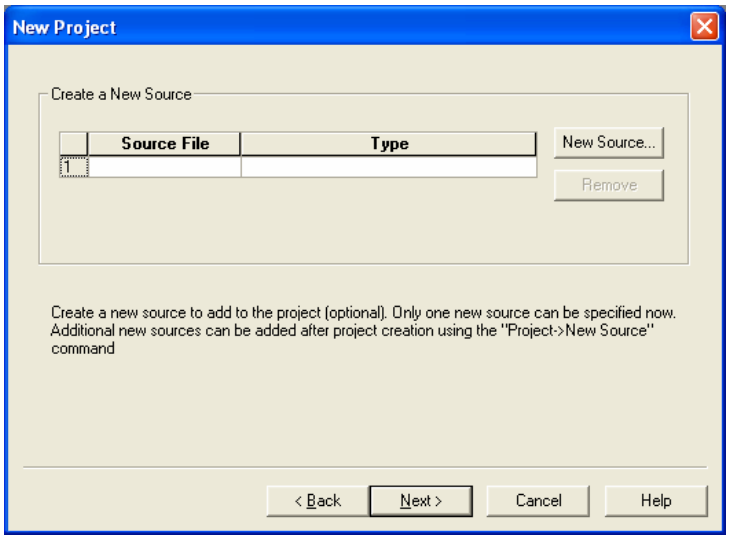


Click **Next**.

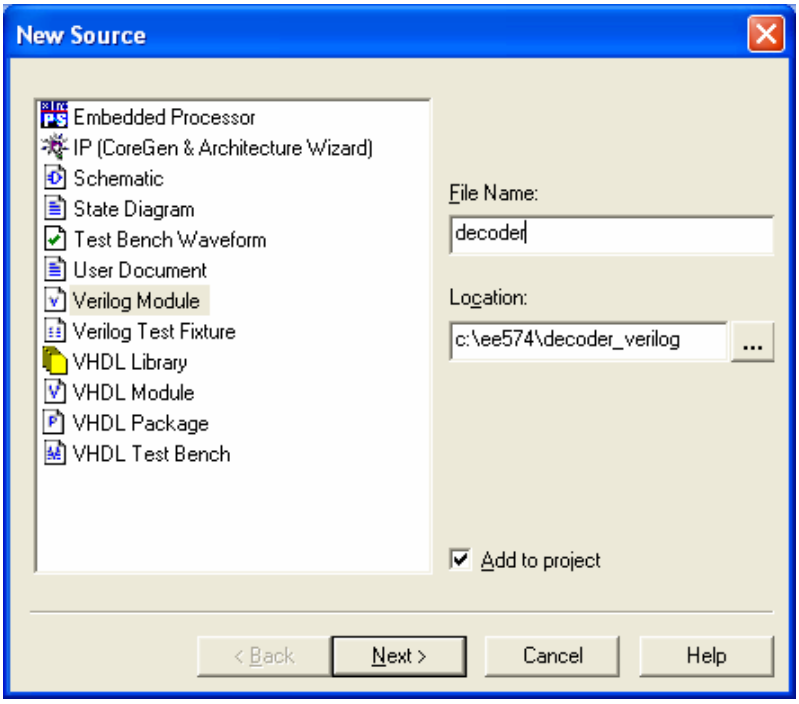
Select the device family, device, and package, and simulation language as shown below (this corresponds to the Spartan 3 device on the starter board:



Click **Next**:

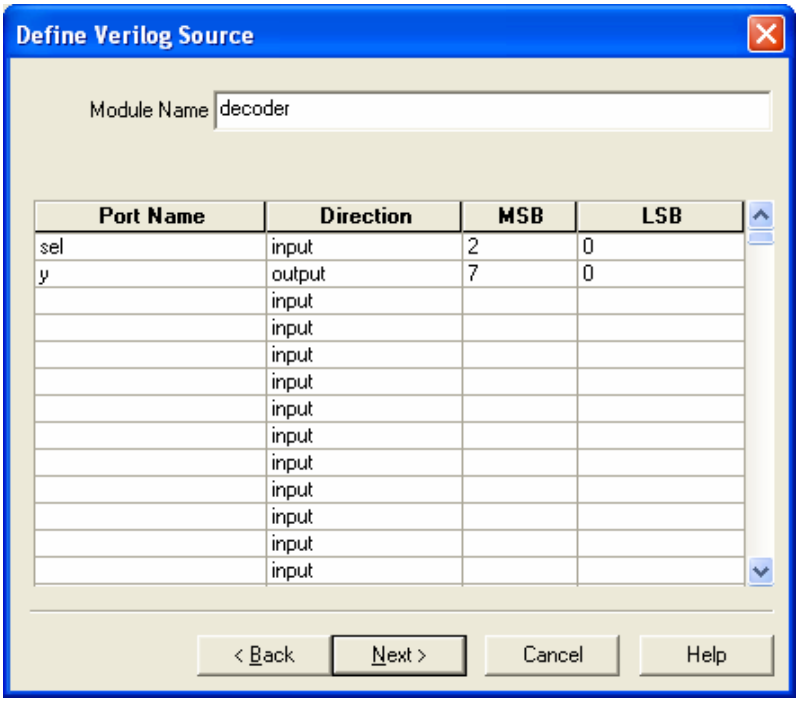


Click **New Source** and enter *decoder* for the file name and select the *Verilog module* for the type of source:



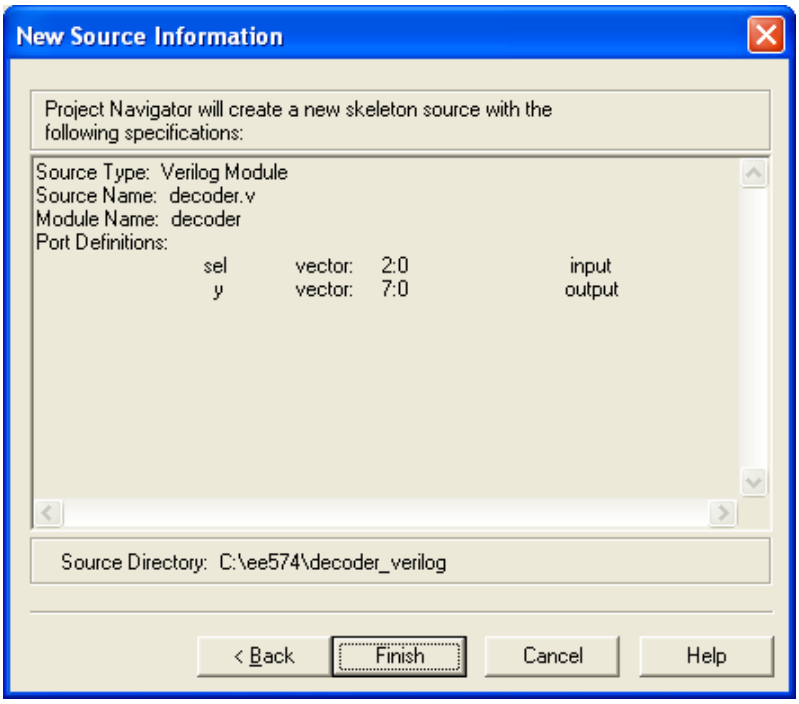
Click **Next**.

You can now specify the inputs and outputs for the decoder.
We have one 3-bit input (sel) and one 8-bit output (y) as shown:

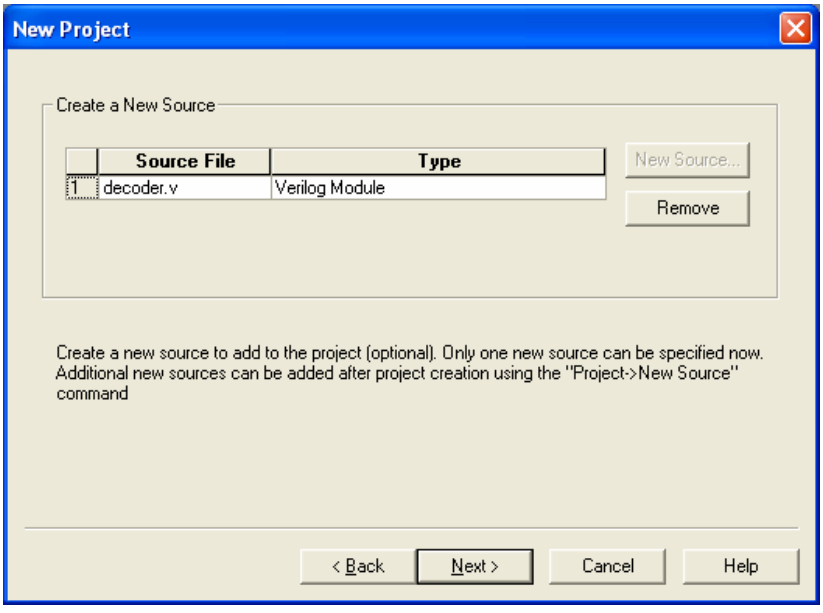


Click **Next**

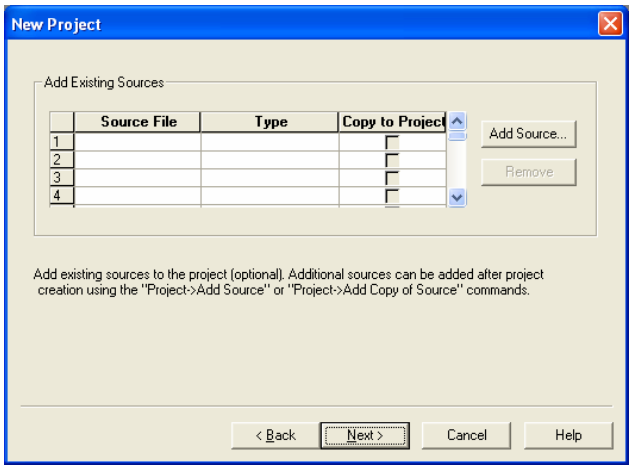
A summary window opens:



Click **Finish**

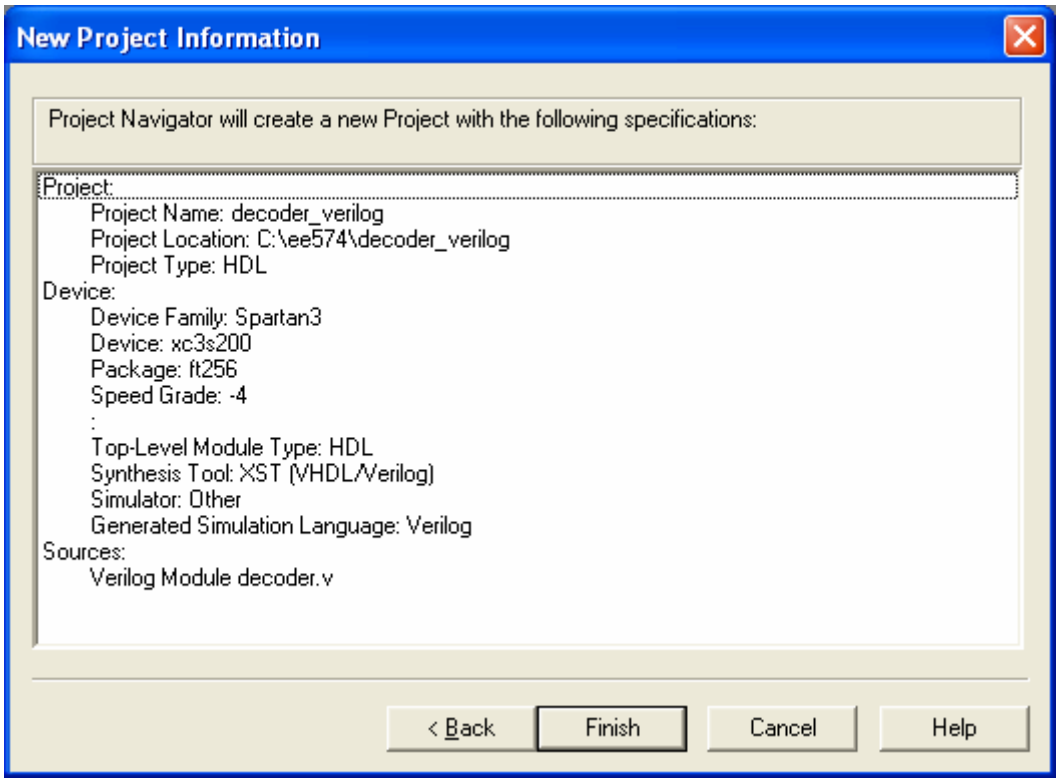


Click **Next**



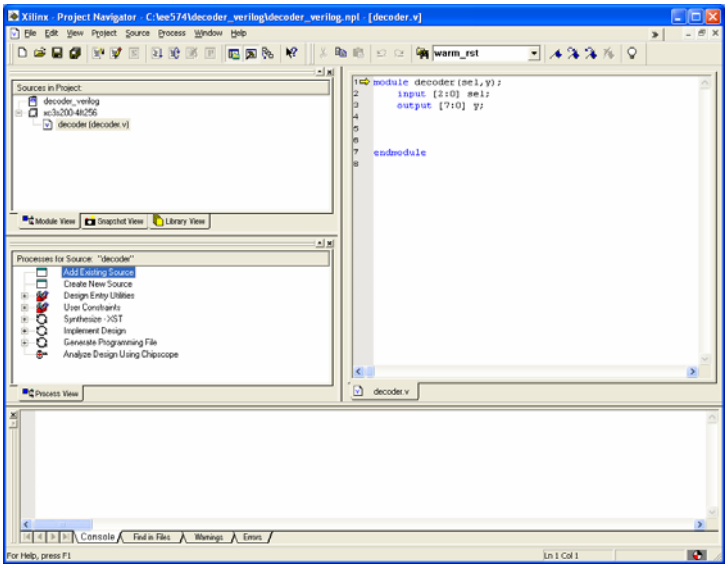
We do not need to add any existing sources so click **Next**

A final summary window is shown:



Click **Finish**

Project Navigator now shows your project including a top level VHDL file for the decoder.



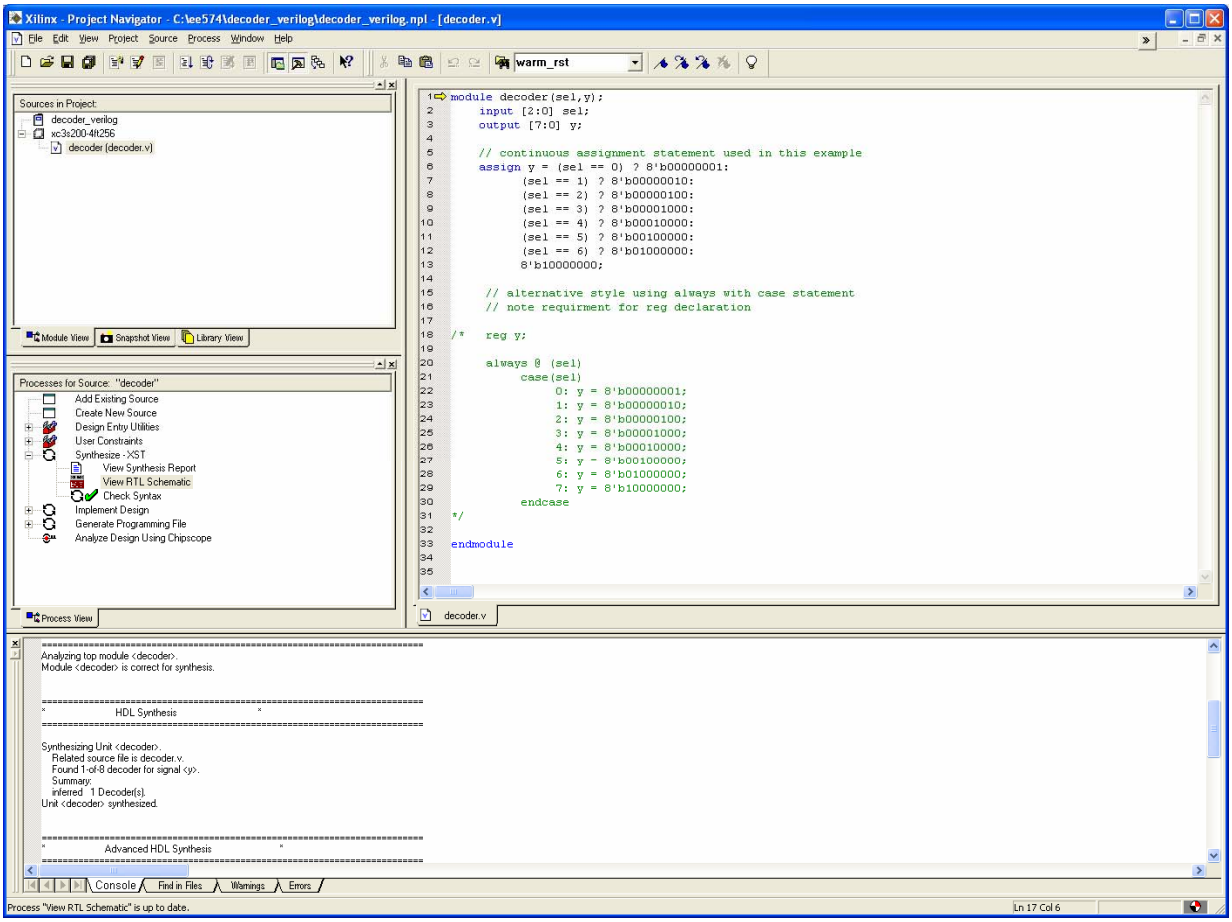
We now need to describe the behavior of the decoder using statements in the architecture body. In this example we will use a continuous assignment *assign* statement but we have also included an alternative ways of describing the decoder using an *always* statement:

decoder.v * - ISE Text Editor

File Edit

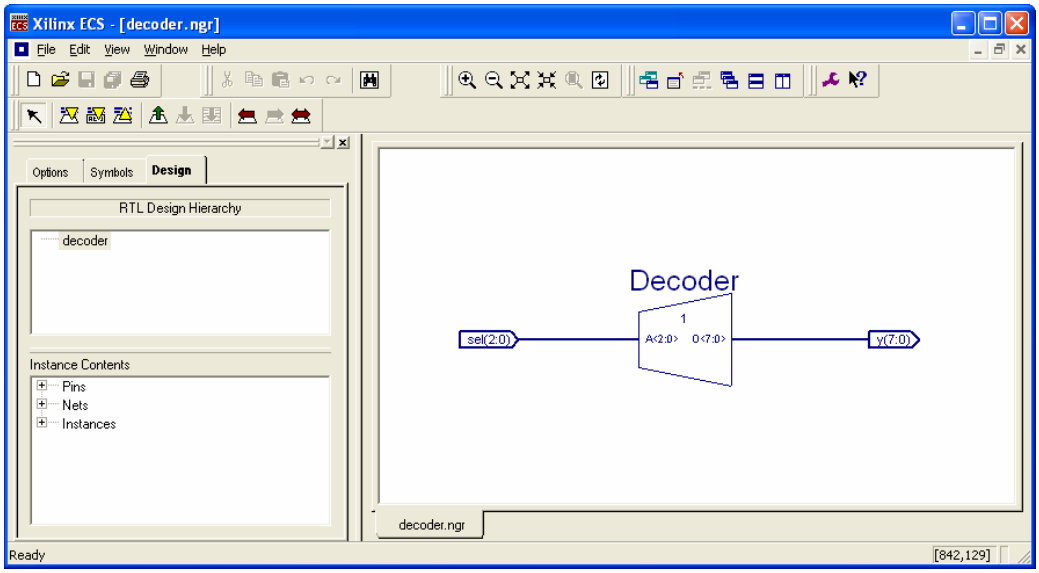
```
1  module decoder(sel,y);
2      input [2:0] sel;
3      output [7:0] y;
4
5      // continuous assignment statement used in this example
6      assign y = (sel == 0) ? 8'b00000001:
7             (sel == 1) ? 8'b00000010:
8             (sel == 2) ? 8'b00000100:
9             (sel == 3) ? 8'b00001000:
10            (sel == 4) ? 8'b00010000:
11            (sel == 5) ? 8'b00100000:
12            (sel == 6) ? 8'b01000000:
13           8'b10000000;
14
15     // alternative style using always with case statement
16     // note requirment for reg declaration
17
18 /*   reg y;
19
20     always @ (sel)
21          case(sel)
22               0: y = 8'b00000001;
23               1: y = 8'b00000010;
24               2: y = 8'b00000100;
25               3: y = 8'b00001000;
26               4: y = 8'b00010000;
27               5: y = 8'b00100000;
28               6: y = 8'b01000000;
29               7: y = 8'b10000000;
30          endcase
31 */
32
33 endmodule
34
```

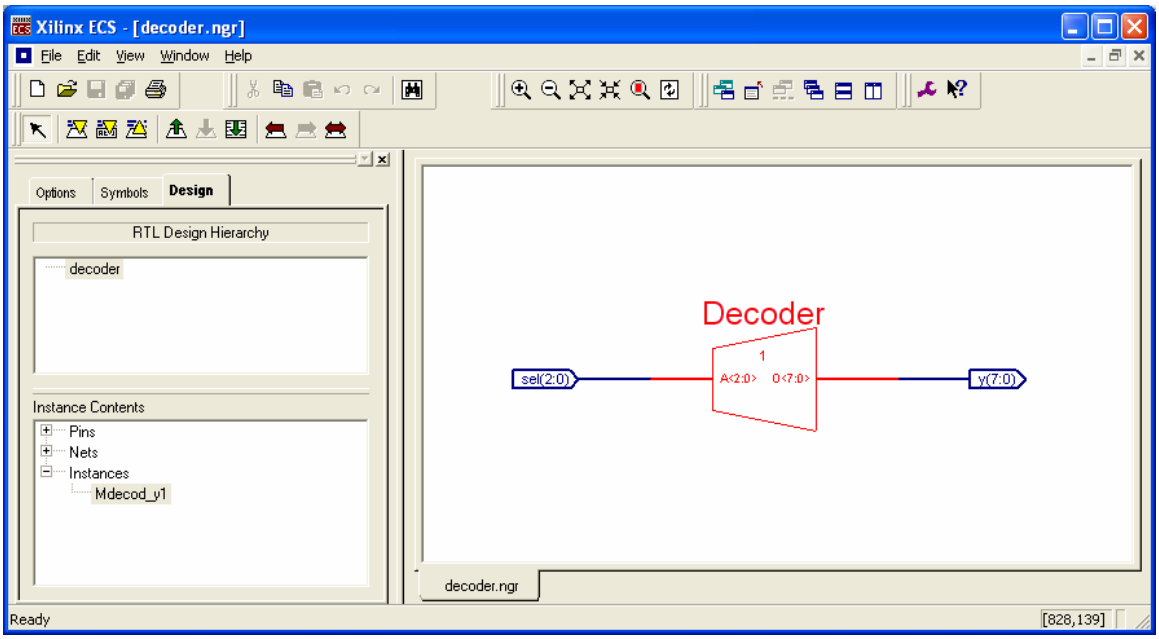
For Help, press F1



Notice the inferred decoder message in the console window.

Whichever style is used (continuous assignment *assign* or *always* statement) the same hardware will be produced as shown by the following RTL schematics:



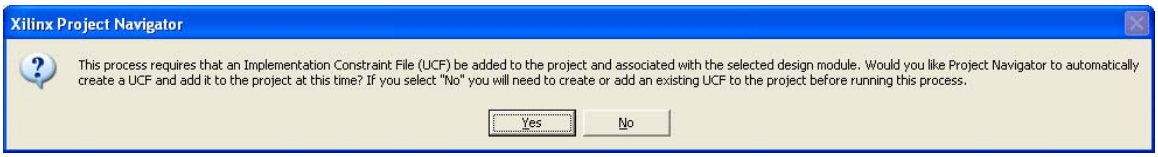


Before we can synthesize this design we need to specify what pins on the FPGA the inputs and outputs are connected to. There are a number of ways to do this.

1) Click on the *Create Timing Constraints* process in the left middle window.

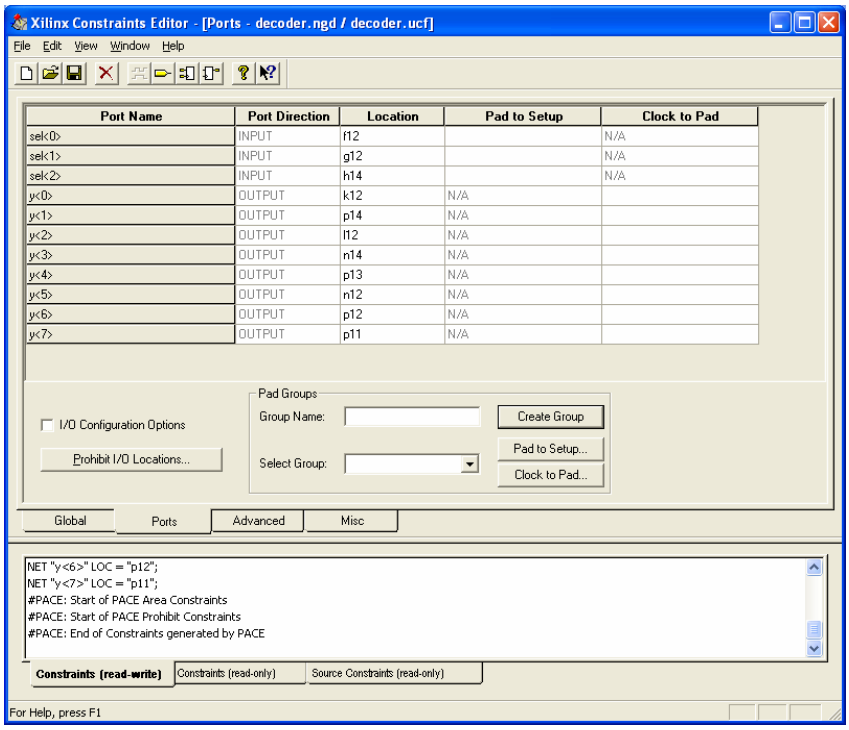
Note: You will be asked to save the file and your design will be checked for syntax errors (these will need to be fixed before you can proceed).

The tools will prompt you to create a UCF file:

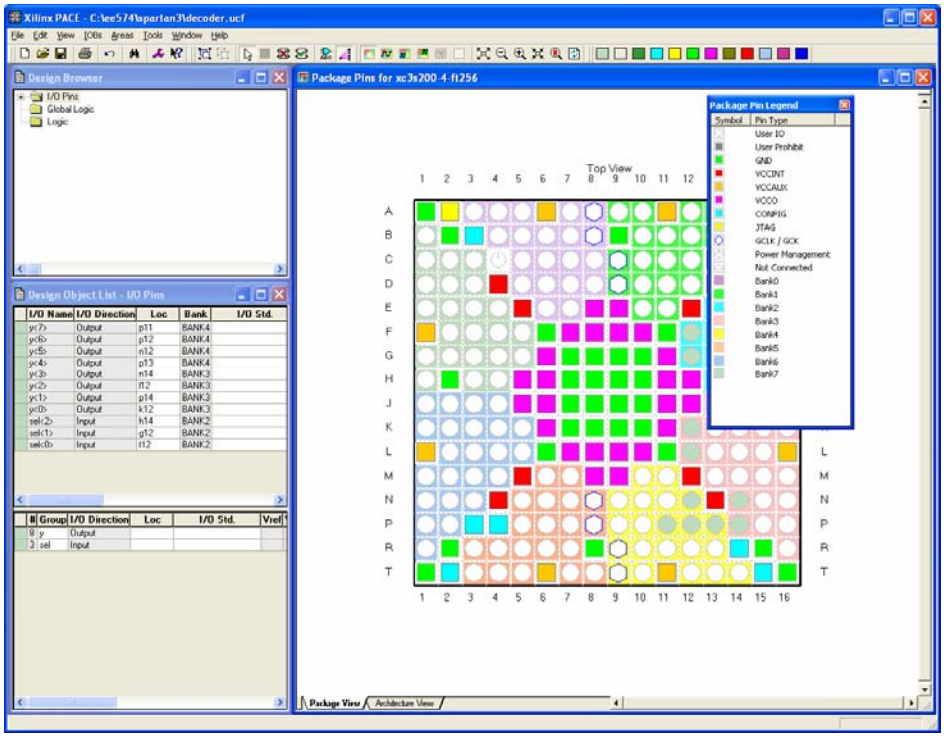


Click **Yes**

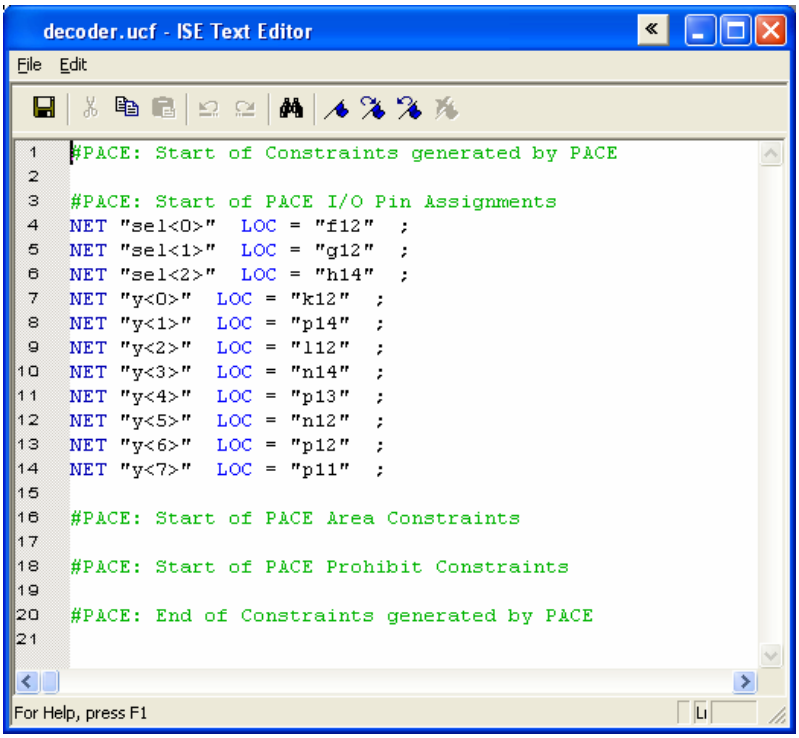
The following window opens, select the *Ports* tab at the bottom and enter the I/O Locations.



2) Click on the *Assign Package Pins* process or *Create Area Constraints* process and the following window opens allowing you to enter the I/O locations:



3) Or create a text file called decoder.ucf

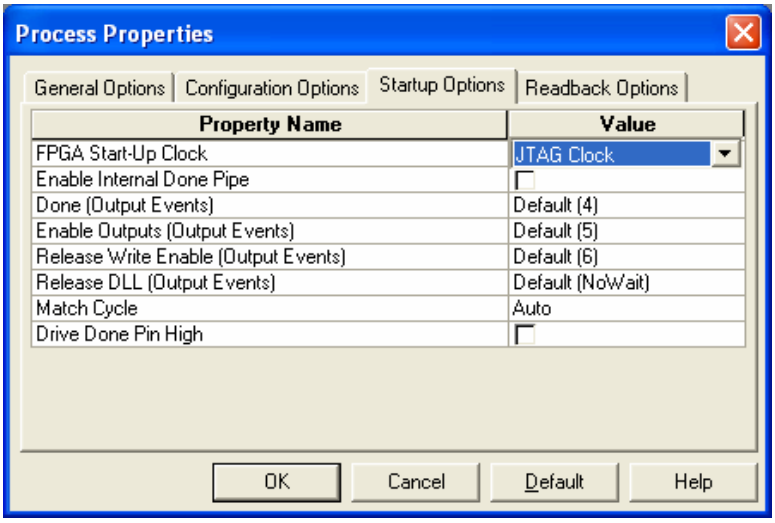


Go back and select the decoder source in the top left pane.

Before we can load the design into the board we need to configure the JTAG connection.

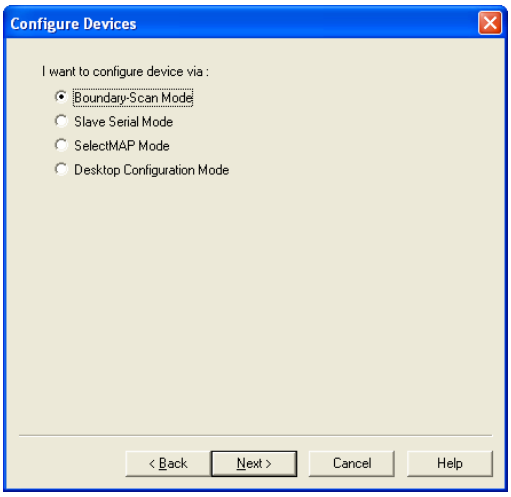
Right-click on the *Generate Programming File* process in the process window.

Select the *Startup Options* tab and change the *FPGA Start-Up Clock* to JTAG clock as shown:



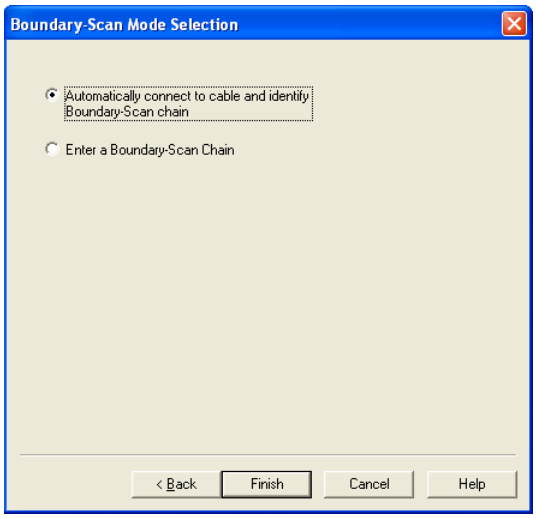
Click **OK**.

Select the *Configure Device (iMPACT)* under the *Generate Programming File* process in the process window.



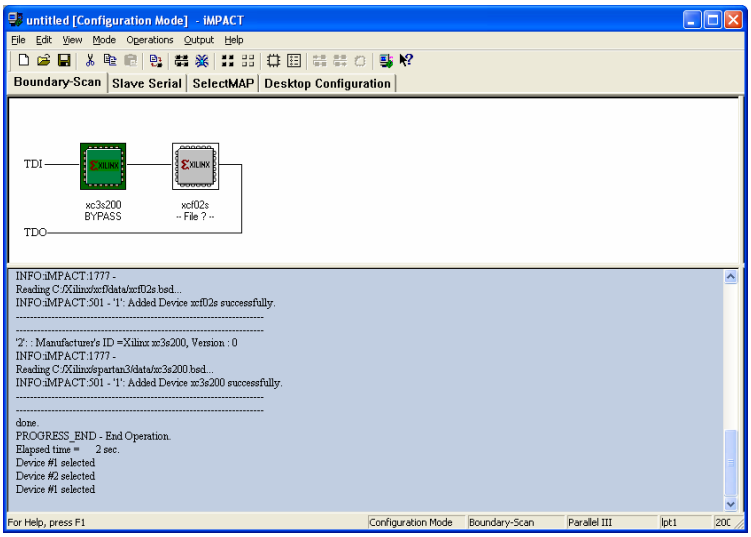
Make sure your board is powered up and the JTAG cable is connected.

Click **Next**

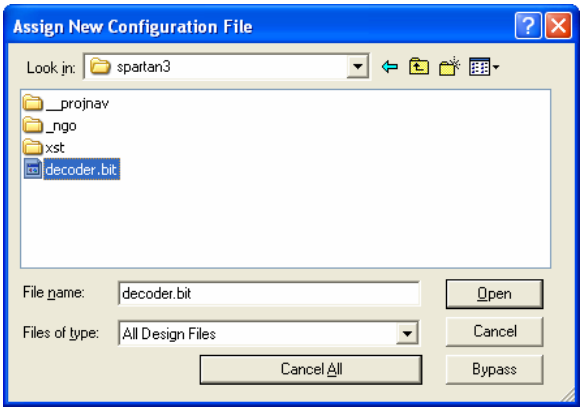


Click **Finish**

Click **OK**

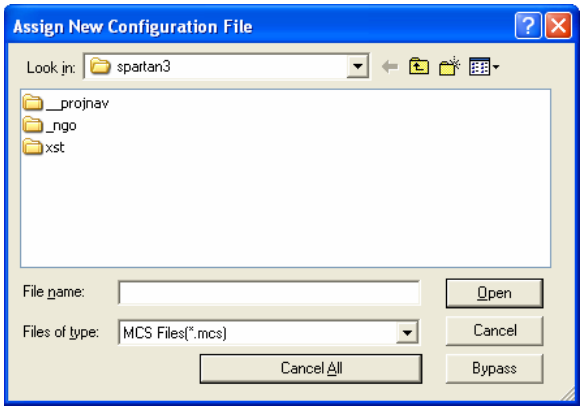


Select the *decoder.bit* file for the first device (the FPGA) in the chain:



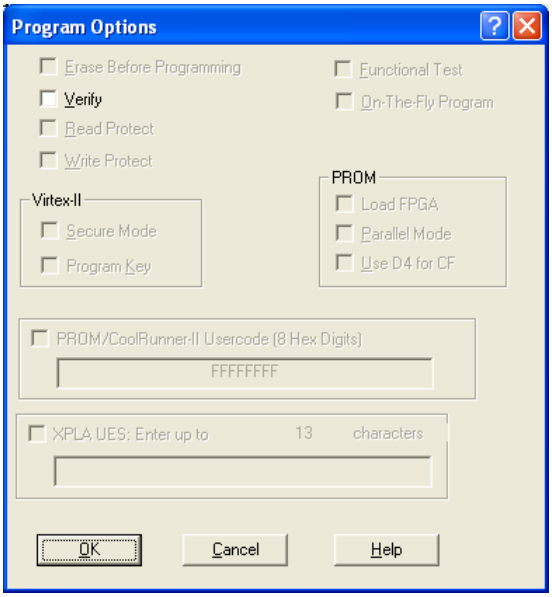
Click **Open**

A new dialog box opens for the second device (the PROM).



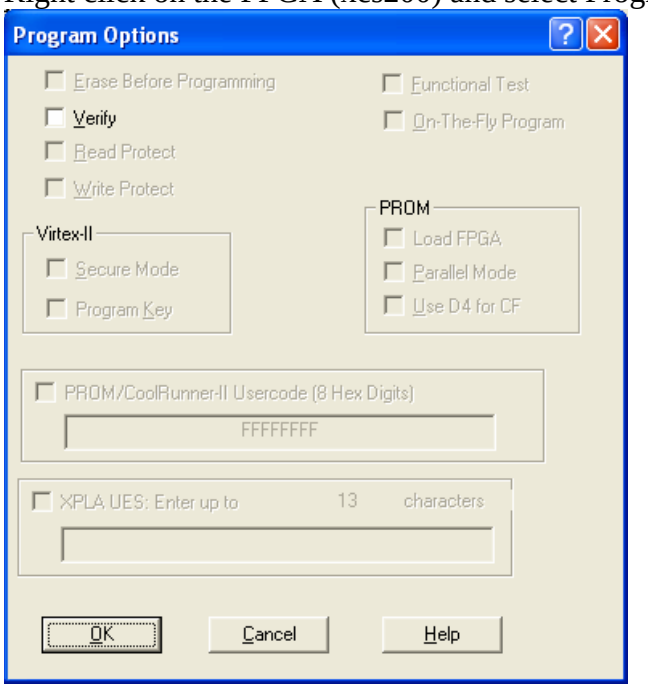
Click **Bypass:**

Right-click on the first device (the FPGA) and select **Program**.
The *Program Options* dialog box opens:



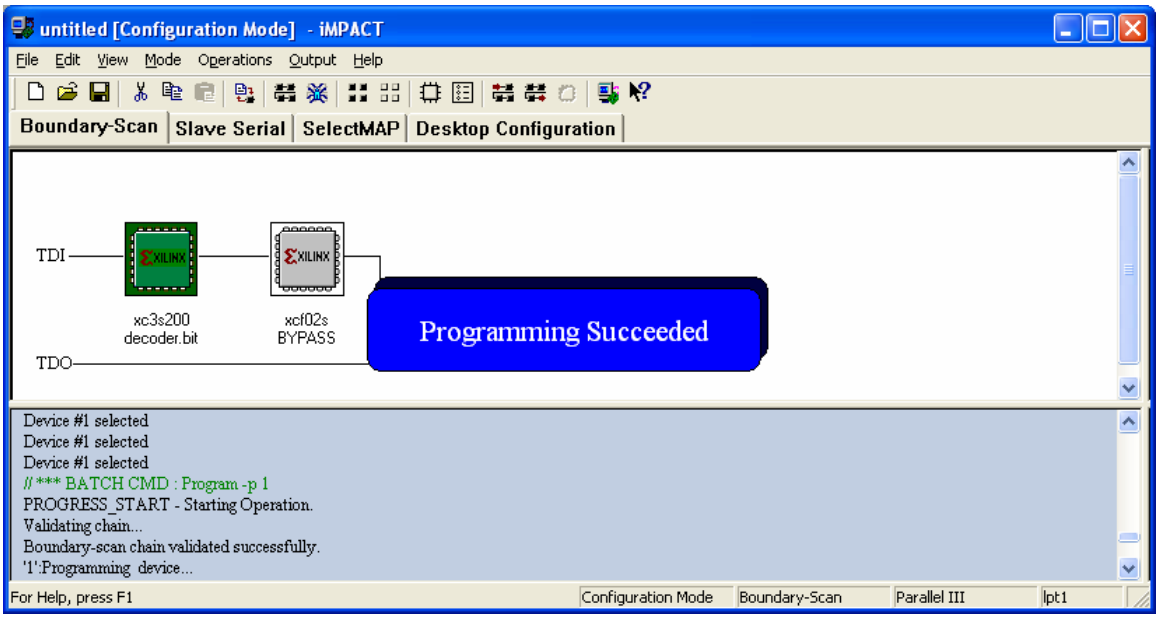
Click **OK**

Right click on the FPGA (xcs200) and select Program



Click **OK**

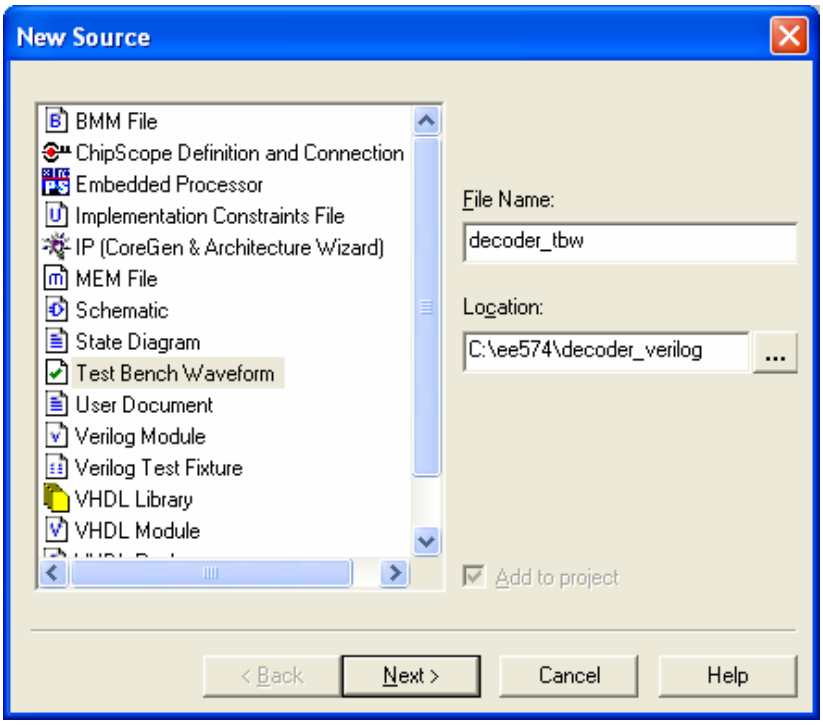
The configuration file is sent to the FPGA, after a few seconds the programming succeeded message should be displayed:



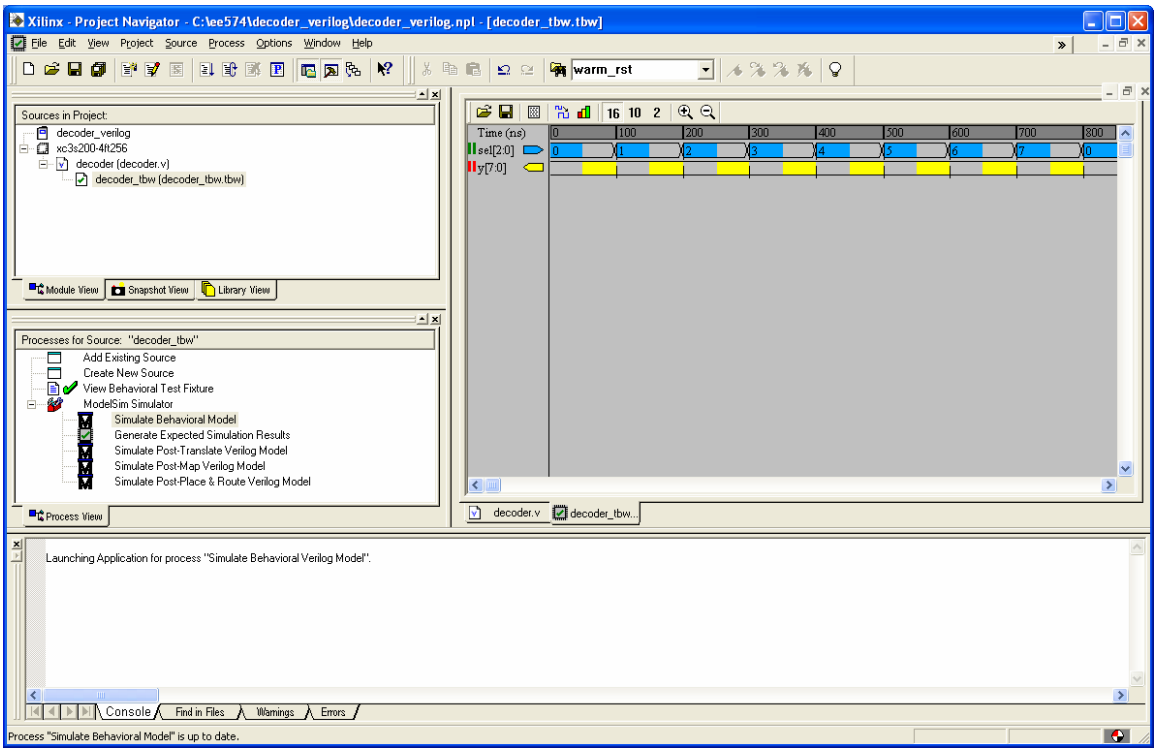
The starter board should now have your decoder design loaded – if you change SW0, Sw1, and SW2 you should see the LEDs change.

Simulating the Decoder

It is possible to simulate the decoder by creating a test bench waveform file
Select **Project = > New Source** and select **Test Bench Waveform**:



Modify the input conditions by changing the values of the sel input – for example as shown below:



Click **on Simulate Behavioral Model** to simulate the design in ModelSim and display the waveform: