

Tutorial Xilinx Foundation

This tutorial goes through the design of a simple combinational circuit : a **3-to-8 decoder**.
It details the steps in the design process using the Foundation CAD tools.

A. Create a project

Open Project Manager from the shortcut on the desktop

A dialog box pops up asking you whether you want to open an existing project or to create a new one. Select create a new Project, give it a name that helps you remember which design it is (Tutorial, Lab1, final_project...), select the directory where you want all the files to be created, and select the flow to be HDL.

A new directory is created in the path you specified, along with a .pdf file is created (it's the extension of foundation's project files)

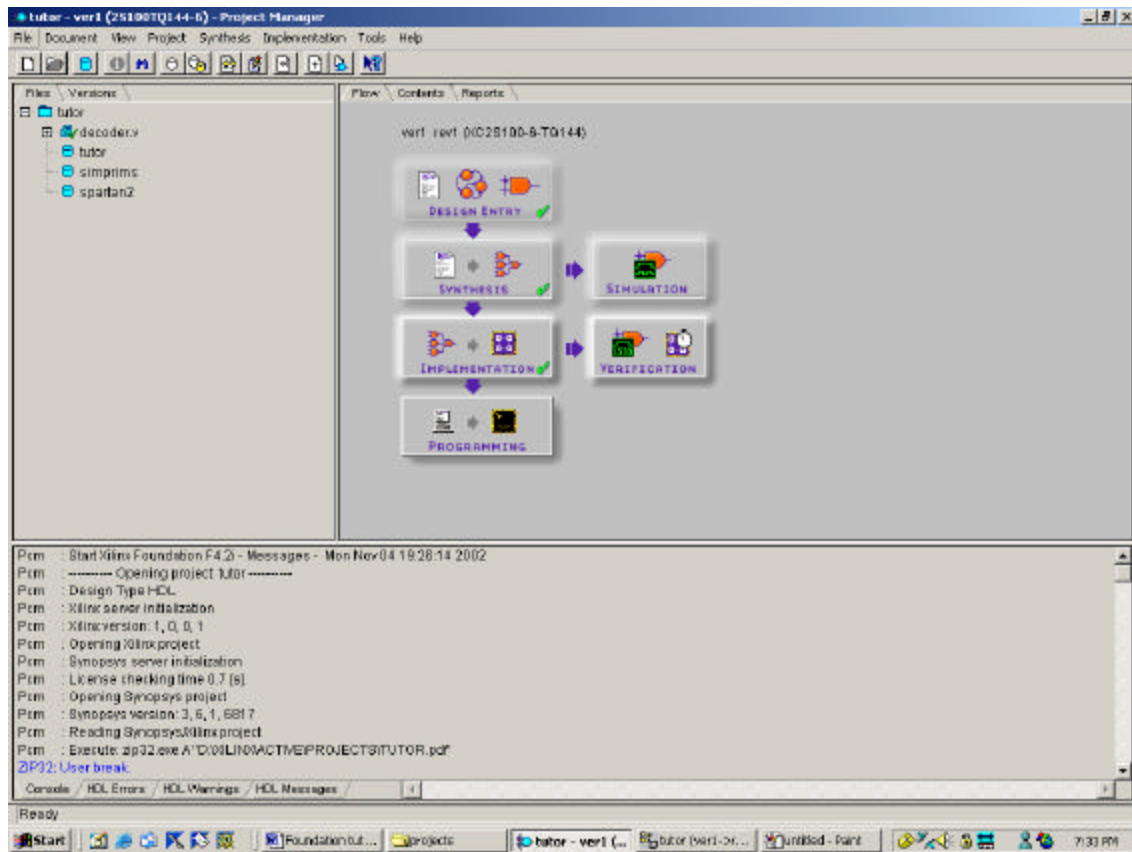


Fig. 1 : Project Manager

The window in fig.1 shows up. On the top left part are the files included in your project, source files and libraries.

On the top right part, the design flow ; these are the steps you can go through, and they are in an order that make sense.

On the bottom is the message board. Look at it when something is going wrong, it'll show you the errors that have occurred or will direct you to the adequate report.

B. Design entry

To start writing your code, click on the button on the right part of the window, under the “flow” tab, on the left part of the design entry box. It is the first step, before you can implement or synthesize anything, so it is the only thing you can click on at that point.

Once you've opened the HDL editor, either create a new document, or open a new one.

We will not go into the details of the verilog code into this tutorial, but you can refer to the verilog tutorial available at www.stanford.edu/class/ee108/tutorials/

```
module three_to_eight_decoder (dipsw, LED);

    input [2:0] dipsw;    //three bit dipswitches
    output [7:0] LED;    //Bar-LED outputs

    reg [7:0] LED;

    always @(dipsw)
    begin
        case (dipsw[2:0])
            3'd0 : LED = 8'b00000001;
            3'd1 : LED = 8'b00000010;
            3'd2 : LED = 8'b00000100;
            3'd3 : LED = 8'b00001000;
            3'd4 : LED = 8'b00010000;
            3'd5 : LED = 8'b00100000;
            3'd6 : LED = 8'b01000000;
            3'd7 : LED = 8'b10000000;
        endcase
    end

endmodule
```

Table 1 : decoder.v

Write the module given in table 1, and save it as decoder.v

.v is the extension for verilog code files. You can then click on the icon with a down arrow and a question mark, in the toolbar of the hdl editor. This checks your code for syntax errors ; you shouldn't have any.

Then close the window, and go back to the Foundation Project Manager.

On the left part of the window are the files included in your project, and the libraries. Right click on the root of the file tree, and choose add HDL source file, then select the source file you just created.

Now, on the right part of the Project Manager window, new buttons have appeared for you to click on.

We have finished the design entry part. At that point, you can simulate your design to see if it implements the right function. Refer to the modelsim tutorial for how to perform simulations.

C. Synthesis

Click on the synthesis button. A settings window will pop up.
Choose the top level module. You don't need to change the version name nor the synthesis settings.
Set the target family to SPARTAN2 and the target device to 2S100TQ144 and click on Run

Now you can look at the work that has been done by editing the EDIF file in your project directory.
You can see the different elements that have been synthesized from your hdl file : inputs and output buffers, logic equations that the software has generated and nets. Basically, the synthesis generate the different blocks in your design ; it creates a schematic structure from the hdl files.
Now we are done with the synthesis part.

D. Implementation

Now you can click on the implementation button.
The same window as for the synthesis pops up. Now you can fill the bottom part of it. The revision name doesn't matter. You shouldn't need to change anything in the options. However, you need to specify a user constraints file (.ucf) that contains pin assignments for your design. Click on the SET button, and specify the user constraint file that you want to use.
A default ucf file is always created when you create the project in the project directory, and it contains only comments. Edit it and add the following lines :

```
NET "dipsw<0>" LOC = "p76";  
NET "dipsw<1>" LOC = "p77";  
NET "dipsw<2>" LOC = "p80";  
NET "LED<0>" LOC = "p68";  
NET "LED<1>" LOC = "p44";  
NET "LED<2>" LOC = "p46";  
NET "LED<3>" LOC = "p49";  
NET "LED<4>" LOC = "p57";  
NET "LED<5>" LOC = "p60";  
NET "LED<6>" LOC = "p62";  
NET "LED<7>" LOC = "p67";
```

Table 2 : User-constraint file

Now save it with the name of your project and the .ucf extension and place it in your project root folder.
Then, go back to the implementation settings window, and click on Run.

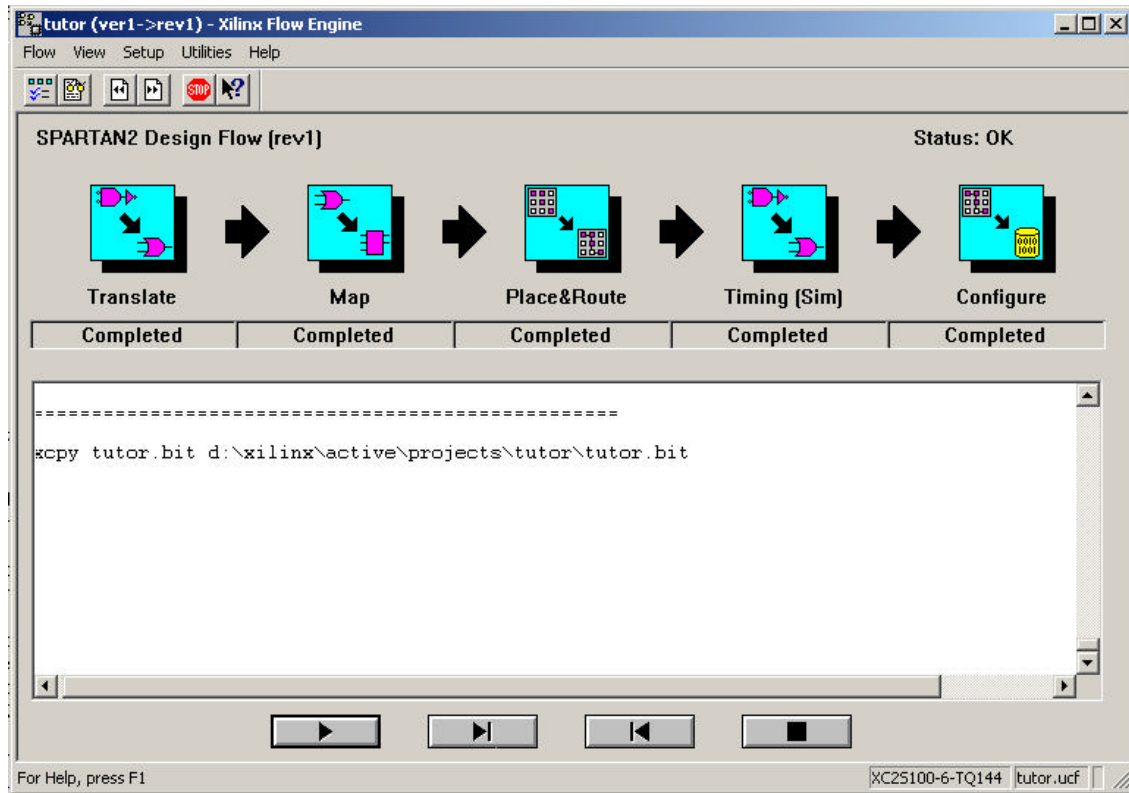


Fig. 2 : Implementation flow engine

The flow engine window opens, and shows the implementation steps :

- Translate optimises your code
- Map assigns different parts of your code to different hardware elements on the FPGA
- Place and Route positions the different blocks and wires them together. It optimises the routing delay
- Timing calculates the minpath and maxpath of your design, to determine at which frequency you can run it.
- Configure creates the .bit file that you will download onto the board.

E. Reports

Once you're done, there are a few things you can look at ; under the report tab in the Project Manager, you can find the implementation report files and the implementation log file. A few interesting numbers are in the implementation log file :

- The total equivalent gate count gives you an idea of the size of your design
- The maximum combinational path delay is the I/O delay of your circuit.

In addition, you can look at the layout of your design on the board by opening the FPGA editor : click on Tools/Implementation/FPGA Editor, and choose the .ncd file that corresponds to your project's name.

Now zoom in the area where your design is concentrated, and put the mouse on the different parts of your design. Comments will show up with the description of the elements you are looking at. Double click on the blue blocks, either the CLBs or the I/O blocks, and a diagram showing the CLB structure and the part that are used will appear.

F. Modify and archive

If you modify your design, you will need to go through all those steps again. To avoid bugs, we advise you to clear the implementation data prior to re-synthesize and re-implement it.

To store your project, click on File/Archive Project. Set the compression factor to maximum, and add files to the archive if you want to. It will create a zip file that you can then use to restore your project on another machine, without having to move all the files and the libraries.