

## Lab 7 Tutorial - In depth Exploration of Floorplanner and XPower

### Floorplanner

Start ISE and open a new project. Use the settings in the image below. Name your project anything you wish.

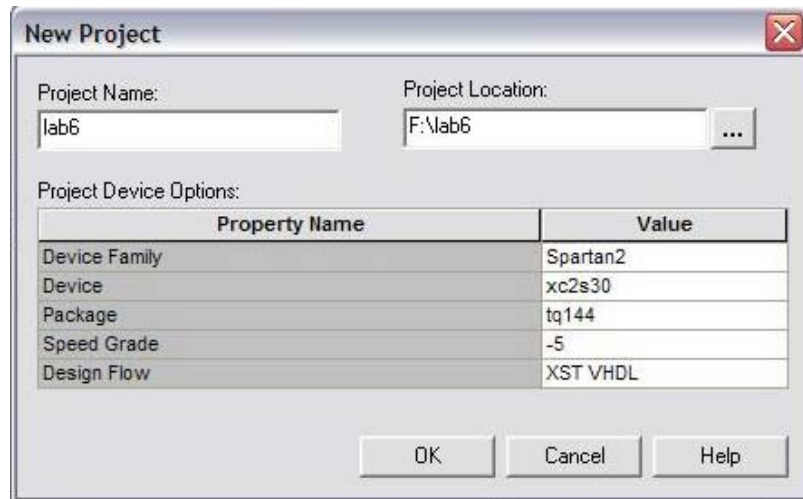


Figure 1

Create a new file and paste the following code into it. As you can see, the code is nothing more than a two input AND gate.

#### AND GATE CODE

```

--*****
--File:And Gate for Lab 7
--
--Purpose: To introduce students to Floorplanner
--and Xpower, this code develops
--a simple AND gate. It has two inputs:
--a and b and one output y.
--
--Created:08/01/02 CK
--
--*****

--Library declarations
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity FEWGATES is port (
a: in std_logic;
b: in std_logic;
y: out std_logic
);
end FEWGATES;

```

```

architecture behavioral of FEWGATES is
begin
fcn: process (a, b) begin
if (a = '1' and b = '1') then
y <= '1';
else
y <= '0';
end if;
end process;

end behavioral;

```

Go to **Project**, **Add Source** and add the file in as a **VHDL Module**.

You now need to assign some pins to your project. Go to **Project**, **New Source**. Choose **Implementation Constraints File**, name your file **const.ucf** and click **Next**, **Next**, **Finish**. In **Processes for Current Source** expand **User Constraints**. Double click **Edit Constraints (Text)**. Copy and paste the following lines into the file.

```

NET "y" LOC = "p11" ;
NET "b" LOC = "p132" ;
NET "a" LOC = "P131" ;

```

These lines assign pin 131 to a, pin 132 to b and pin 11 to y.

Make sure you have the vhd file selected then double click **Implement Design**. Expand **Implement Design and Place & Route**. Double click **View/Edit Placed Design (Floorplanner)**.

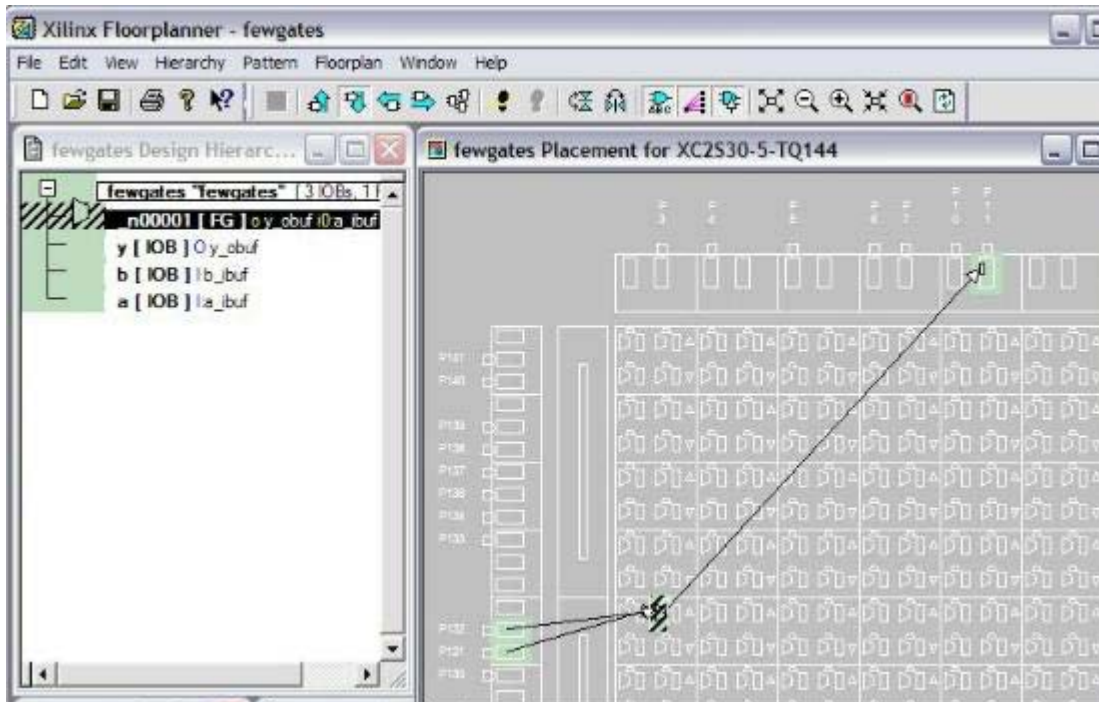


Figure 2

Zoom in by clicking on the + magnifying glass icon. By clicking once on the CLB's LUT you can see that the software has developed the AND gate by doing nothing more than using one of its look up tables. You can also see how it has mapped pins 131, 132 and 11 from the IOBs to the CLB. There is nothing new here. What you are looking at is how the software mapped, placed and routed the design onto the hardware. The choice of pins on the IOB was our choice and the choice of that specific CLB was the software's choice.

Close Floorplanner and return to Project Navigator (no need to save changes). Right click on *const.ucf* and click **Remove**. This will delete your .ucf file and its associated pin numbers.

You will now be exposed to some of the more advanced features of Floorplanner and FPGA editor. What Floorplanner is going to allow us to do is control placement of the design on the chip. This time I am going to place the logic in the upper left portion of the chip and use two inputs on the left side of the CLB and an output on the upper portion of the chip. In the past we have used constraints editor and .ucf files to control/constrain our pin assignments. This time we are going to use Floorplanner not only to control/constrain our pin numbers but also our placement of the logic.

In *Processes for Current Source* expand **Implement Design** and **Translate**. Double click **Floorplan Design**.

This GUI is a drag and drop type of vehicle. It will allow it to drop logic and input/output pins where you want them (based on the ability to support your desires). It is the same with the inputs and output pins.

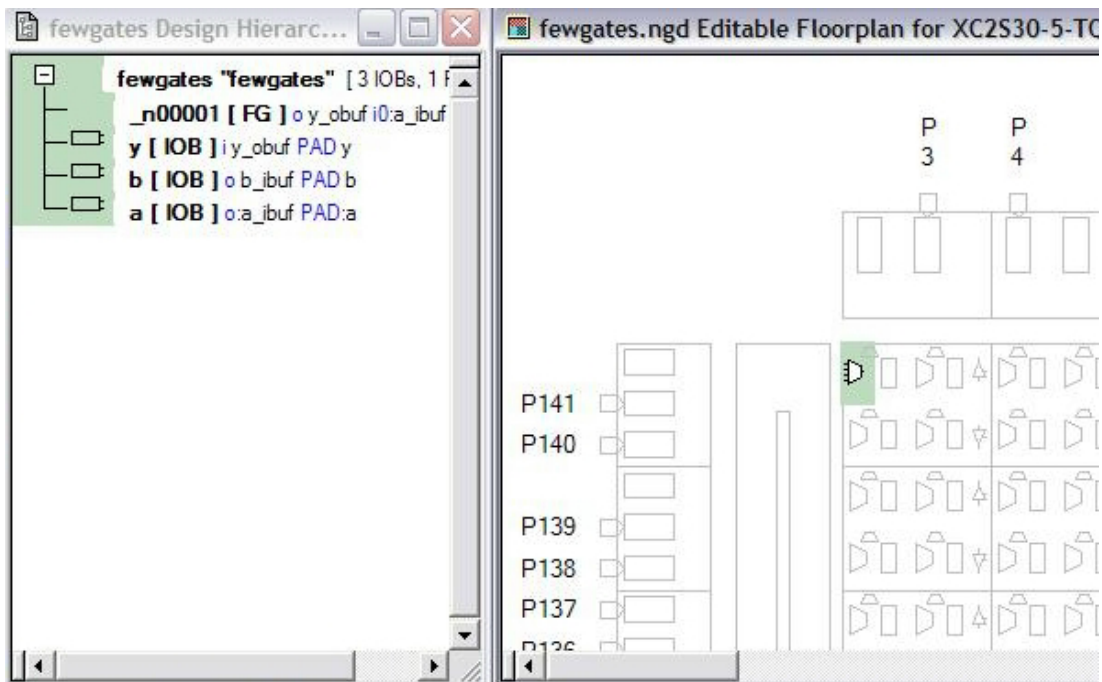


Figure 3

Zoom in on the upper left corner of the chip. Drag and drop the logic gate next to `_n00001` into the upper left corner of the CLB (see upside).

Now drag and drop `y` onto pin 3, `b` onto pin 141 and `a` onto pin 140.

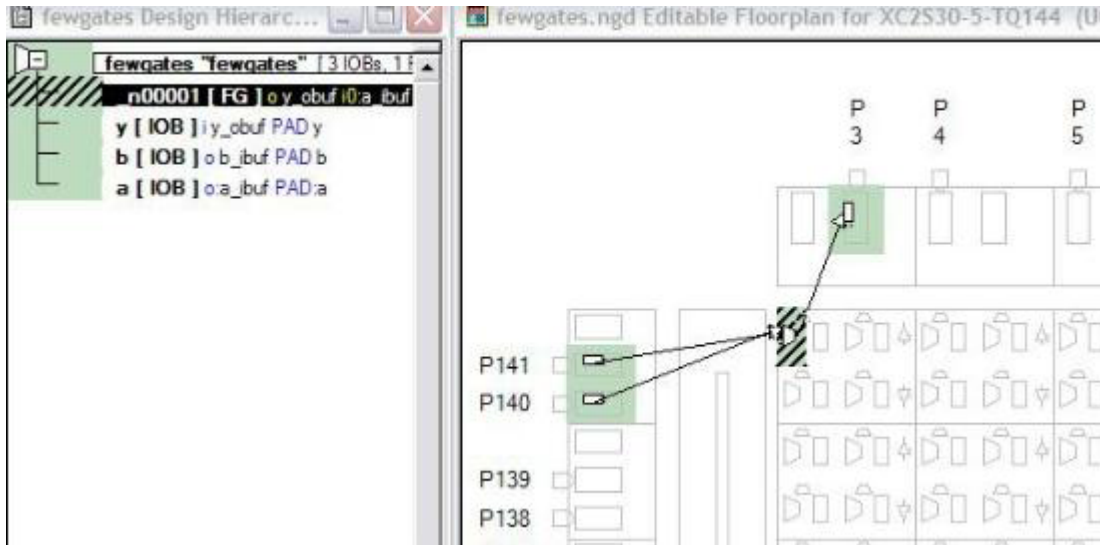


Figure 4

As you can see we have developed the entire project onto the upper portion of the chip. You might need this type of control to group various parts of your design to various sections of the chip. The general design strategy for FPGAs depends on how many gates are on the device. For devices with less than 100K system gates then the IO for the control signals should be on the top or the bottom and the IO for data should be on the left or right. The change comes for devices greater than 100K system gates. For these devices you would keep the same concepts as before but you would also now attempt to group control signals and data buses near related internal logic.

Close Floorplanner. It will prompt you to save changes to your .ucf file. Click **Yes**. Name your file and click **Save**. You have now constrained your pins and logic placement.

Next we will take a trip into FPGA editor to see a little better how the logic is developed within the CLB/slice. Expand **Implement Design** and **Map**. Double click **on Manually Place & Route (FPGA Editor)**.

Click once `y_o`.



Figure 5

Now click once on the *editblock* button on the right hand side. As you can see the potentials to control the design process are virtually limitless.

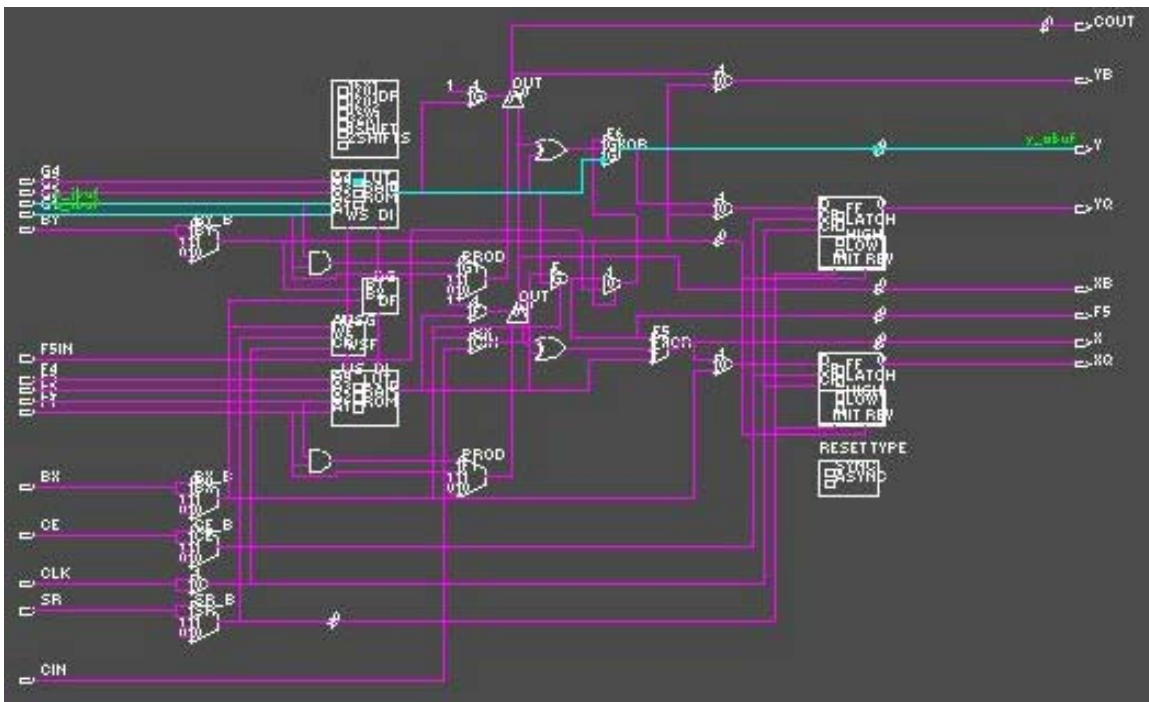


Figure 6

Now you are looking into the CLB. You can see the two inputs and the one output from the LUT. By clicking once on the blue square you can see the logic function ( $D = A1 * A2$ ).

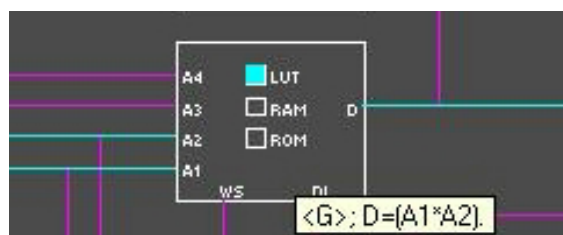


Figure 7

## 2. XPower

You have seen how you can control how the hardware is deployed onto the chip. Now you will see how much power your design will use. That is where XPower comes into play.

In the **Processes for Current Source** window, expand the **Implement Design** and **Place & Route**. Double click on **Analyze Power (XPower)**. When XPower loads an dialog box may pop up, just click **OK**.

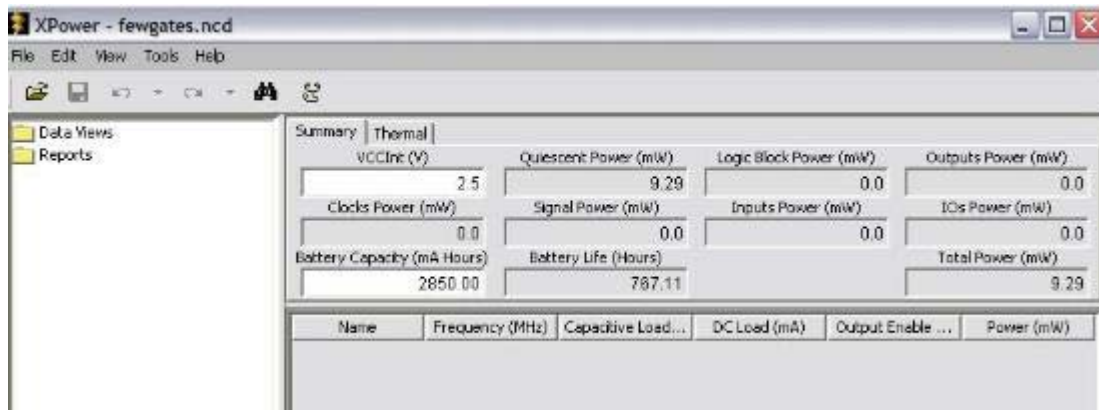


Figure 8

It should be pointed out before we go any further that it is only possible to obtain an accurate power consumption indication by fully specifying all input and signal frequencies involved with the device.

As you can see from the picture above, the default voltage (VCCInt) is 2.5V. By increasing or decreasing this voltage we can increase or decrease the required power for the device. Unfortunately, this is not a limitless capability. For example, if we decrease the voltage to 2.374V (click into the block, change the value and hit enter) we get the following warning in the history bar at the bottom of the screen.

*WARNING:Power:510 - VccInt <2.374> not in recommended range [2.375..2.625]V.*

This tells us our recommended range of our voltage. Set it for 2.4V and then for 2.5V and you will see that as voltage increases, **Total Power** also increases from approximately 7.99 milliwatts to around 9.29 milliwatts.

Next click on the **Thermal** tab.



Figure 9

Now, change the **Ambient Temp** value. XPower's default temperature setting is always 25 degrees Celsius but its acceptable temperature range is –40 degrees C to 125 degrees C. Again try entering several temperature values within that range to see how the **Junction Temp** is affected by changing the Ambient Temp. It should be noted that Junction Temperature is affected by changes in power values and package choices as well as by changes in the Ambient Temperature.

Now to get a final accurate assessment of my power consumption I will go back to my default **VCCInt (V)** and **Ambient Temp** values of 2.5 and 25. The final step is to assign frequencies to my input and internal signals. By expanding **Data Views, Types** and **Signals** you can see the two inputs **a\_ibuf** and **b\_ibuf** and the output **y\_obuf**. In case you cannot remember what these signals are or what they do, highlight one of them and right click. Click the **Show in FPGA Editor** option. Once FPGA editor launches, on the right hand side, highlight the signal and then click on **editblock**. It will bring up the schematic and show you the IOB and the routing of the signal. Now let's constrain the signals to a specific frequency for power analysis purposes.

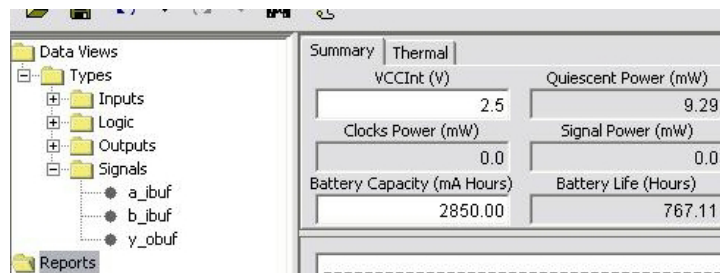


Figure 10

To do so I click into the Frequency block and enter 20.

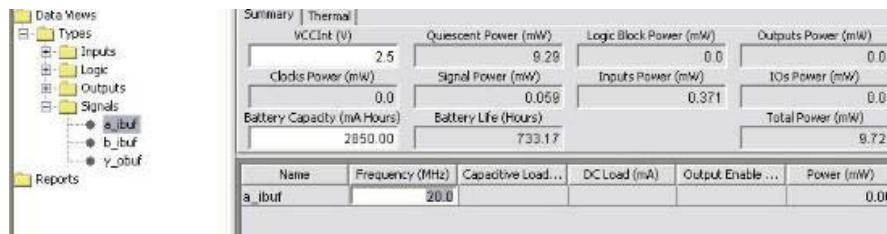


Figure 11

This means that I intend to clock this input at 20 MHz. Once you type in 20 hit Enter to see the change take effect. What you will notice is that you have increased the Total Power and the Junction Temperature. The faster you clock the project the greater your power consumption and the hotter the device. To get a detailed report of how your changes effect your designs consumption you can click into the Reports section and then click on Power Report.

It should be pointed out that XPower for CPLDs is slightly different than it is for FPGAs. Due to the nature of CPLDs and their preference in smaller battery operated devices, there is a greater emphasis on battery life.