Xilinx ISE 10.x Tutorial for a Counter

Original Author: Jerry Wu
Revisions: William Gibb

Start up ISE

- If your on windows, launch the gui from the start menu.
- If your on linux, launch the gui from the command line.
    - user@user-desktop:~$ cd /opt/Xilinx/10.1/ISE/
    - user@user-desktop:/opt/Xilinx/10.1/ISE$ source settings32.sh
    - user@user-desktop:/opt/Xilinx/10.1/ISE$ ise

Create a New Project:

- Select **File → New Project**.
- Type **Exercise** in the *Project Name* field.
    - You may want to keep all of your ISE projects in a single folder, such as ~/Projects.
    - In this case, make sure that you type **home/$user/Projects** (use the path appropriate to your system!) in the *Project location* field.
    - Your project files will be hosted in ~/Projects/Exercise. This is convenient if you want to copy or move your project.
    - A word of warning for Windows users: Do not include spaces in the path to the project location.
- Verify that **HDL** is selected from the *Top-Level Source Type* list.
- Click Next to edit the device properties.
- Edit the properties in the table as shown below:
    - *Family:* **Spartan3E**
    - *Device:* **XC3S500E**
    - *Package:* **FG320**
    - *Preferred Language:* **Verilog**
- Select the checkbox of **Enable Enhanced Design Summary**
- Click **Next** to process to the *Create New Source* window in the *New Project Wizard*.

Create a RTL Code with Verilog HDL:

- Click **New Source** in the *New Project* dialog box.
- Select **Verilog Module** as the source type in the *New Source* dialog box.
- Type in the file name **counter.v**.
- Verify that the **Add to Project** checkbox is selected.
- Click **Next** through the screens and then **Finish** to create the project.
- Open up the file, **counter.v**
- Copy and paste the code for module counter from the counter.txt file.
- Select **File → Save**.

Set up Test Bench and Simulate the RTL Code:

- Select the **counter.v** HDL file in the *Sources* window.

- Select **Project → New Source** to create a new test bench source.
- In the *New Source Wizard*, select **Test Bench WaveForm** as the source type, and type **counter_tb** in the *File Name* field and then click **Next**.
- The *Associated Source* page shows that you are associating the test bench waveform with the source file counter. Click **Next**.
- The *Summary page* shows that the source will be added to the project, and it displays the source directory, type and name. Click **Finish**.
- To set the clock frequency, setup time and output delay times in the *Initialize Timing* dialog box before the test bench waveform editing window opens:
    - *Clock High Time:* **10ns**
    - *Clock Low Time:* **10ns**
    - *Input Setup Time:* **5ns**
    - *Output Valid Delay:* **5ns**
    - *Offset:* **50ns**
    - *Global Signals:* **GSR (FPGA)**
    - *Initial Length of Test* Bench: **1000ns**
- Click **Finish** to complete the timing initialization
- Select **Behavioral Simulation** and **counter_tb** in the *Source* window.
- In the *Process* tab, click the + to expand the *ISE Simulator* process and double-click the **Simulate Behavioral Model** process.
- To verify the simulation result, select the **Simulation** tab and zoom in on the transitions.

Modify the Design:

- After verify the simulation result, we want to be able to check the implemented behavior of the counter.
- To do that, we need to change the count frequency in the counter.v HDL file.
    - In the counter.v file, we set t==4 so the counter add 1 every 5 clock cycle, which means the counter frequency is 10MHz since the internal clock frequency in the Spartan 3E-Starter board is 50MHz.
    - Now, we should lower the frequency in order for us to be able to detect the LEDs lighting change after we implement the RTL code on the board.
- Double-click the file **counter.v** in the *Sources* window.
- Change **if (t==4)** to **if (t==99999999)** in the counter.v file. Now the counter frequency is changed from 10MHz to 0.5Hz. This low count frequency (0.5Hz frequency, 2 second period) will allow us to notice that LED status changes (counting up) every 2 second.
- Select **File → Save**

Assign PinConstraints and Implement the Design:

- Select the **counter.v** HDL file in the *Sources* window.
- Now we will implement package-pin constraints. There are two ways to do this.
- Via gui
    - Double-click the **Assign Package Pins** process in the *User Constraints* process group. The *Pinout and Area Constraints Editor* opens.
    - Select the **Package View** tab.
    - In the *Design Object List* window, enter a pin location for each pin in the **Loc** column using the following information:

        - *clock*: **C9** (On-Board 50MHz Oscillator signal on board, the internal clock with 50MHz)
        - *count_out<0>:* **F12** (LD0 signal on boad)
        - *count_out<1>:* **E12** (LD1 signal on boad)
        - *count_out<2>:* **E11** (LD2 signal on boad)
        - *count_out<3>:* **F11** (LD3 signal on boad)
    - Select **File → Save**.
- Via text file
    - Click **New Source** in the *New Project* dialog box.
    - Select **Implementation Constraints File** the source type in the *New Source* dialog box.
    - Type in the file name **counter.ucf**.
    - Verify that the **Add to Project** checkbox is selected.
    - Click **Next** through the screens and then **Finish** to create the project.
    - Under the **Sources** window, expand the tree to show **counter.ucf.**
    - Under the **Processes** window, expand the tree **User Constraints** and click **Edit Constraints (Text)**.
    - Enter the following lines into the file.
        - NET "clock" LOC = "C9" | IOSTANDARD = LVCMOS33 | PERIOD = 20 ns HIGH 50%;
        - NET "count_out<0>" LOC = "F12" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = SLOW;
        - NET "count_out<1>" LOC = "E12" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = SLOW;
        - NET "count_out<2>" LOC = "E11" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = SLOW;
        - NET "count_out<3>" LOC = "F11" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = SLOW;
    - Select **File → Save**.
- Double-clicking the **View Design Summary** process in the *Processes* tab to open the *Design Summary*.
- Double-click the **Implement Design** process in the *Processes* tab.
- As implementation steps complete, green check marks will show up in the check boxes.
- Select the **Pinout Report** and select the **Signal Name** column header to sort the signal names. Verify that signals are routed to the correct package pins.

Download Design to the Spartan 3E-Starter Board:

- Connect the 5V DC power cable to the board (J20) and connect download USB cable between PC and the board (J18). Slide the power switch (next to the power input) to on.
- Select **Synthesis/Implementation** from the drop-down list in the *Sources* window.
- Select **counter.v** in the *Sources* window.
- Click the + to expand the **Generate Programming File** processes in the *Processes* window.
- Double-click the **Configure Device (iMPACT)** process.

- In the *Welcome to iMPACT* dialog box, select **Configure devices using Boundary-Scan (JTAG)** and **Automatically connect to a cable and identify Boundary-Scan chain**.
- Click **Finish**. The devices connected to the JTAG chain on the board will be detected and displayed in the impact window.
- When the *Assign New Configuration File* dialog box appears, select the **counter.bit** file and click **Open** so that the configuration file can be assigned to the xc3s500e device in the JTAG chain.
- Select **Bypass** to skip any remaining devices.
- Right-click on the **xc3s500e** device image, and select **Program**. The *Programming Properties* dialog box opens.
- Click **OK** to program the device. When programming is complete, the *Program Succeeded* message is displayed.
    - On the board, LEDs 0, 1, 2, and 3 are lit, indicating that the counter is running.