Architectural Wizard and IP Catalog

Introduction

Today's Xilinx FPGAs contain many more resources than basic, LUT, CLB, IOB, and routing. The FPGAs are now being used to implement much more complex digital circuits compared to glue logic when they were invented. Some complex architectural resources, such as clocking, must be configured and instantiated instead of inferred. There are also commonly used complex circuits, such as the Reed-Solomon decoder and tools so that a designer does not have to "reinvent the wheel" and develop the basic features on their own. This lab introduces architectural wizard and IP Catalog tools available through the IP Catalog. *Please refer to the Vivado tutorial on how to use the Vivado tool for creating projects and verifying digital circuits*.

Objectives

After completing this lab, you will be able to:

- Use the Architectural Wizard to configure clocking resource
- Use the IP Catalog tool to configure and use counters and memories

Architectural Wizard Part 1

Some specialized and advanced architectural resources can efficiently be utilized when configured and instantiated properly instead of inferring them. Depending on the FPGA family being used, the number and types of such resources vary. In the Artix-7 family, clocking, SelectIO, and soft error mitigation (SEM) resources are supported by the architectural wizard. In the Artix-7 family, an additional architectural resource, GTP Transceiver, is available. These resources are accessed under the IP Catalog capability of the Vivado tool.

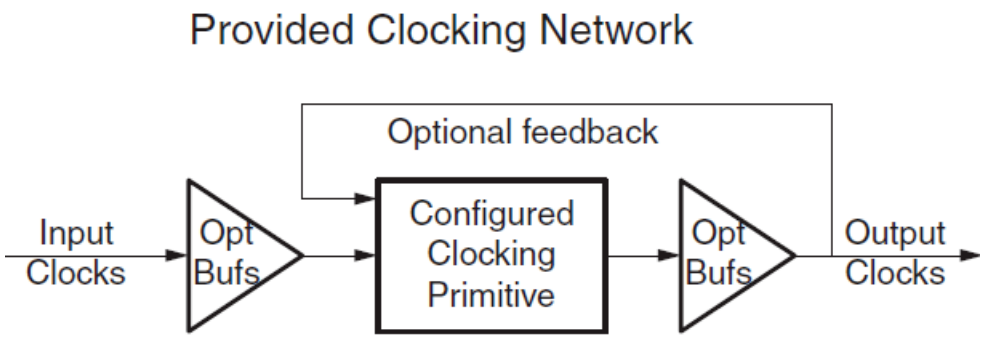
On the Basys3 and the Nexys4 DDR board, a 100 MHz clock source is available which is connected to the W5 (Basys3) and E3 (Nexys4 DDR) of the FPGA. This clock source can be used to generate a number of clocks of different frequencies and phase shifts. This is done by using architectural resources called Mixed-Mode Clock Manager (MMCM) and Phase Locked Loop (PLL) of the Artix-7 family FPGA. The clocking source generator can be invoked by double-clicking the Clocking Wizard entry under the Clocking sub-folder of the FPGA Feature and Design folder of the IP Catalog.

Name	AXI4	Status	License
Automotive & Industrial			
AXI Infrastructure			
BaseIP			
Basic Elements			
Communication & Networking			
Debug & Verification			
Digital Signal Processing			
Embedded Processing			
FPGA Features and Design			
Clocking			
Clocking Wizard	AXI4	Production	Included
IO Interfaces			
Soft Error Mitigation			
XADC			
Math Functions			
Memories & Storage Elements			
Standard Bus Interfaces			
Video & Image Processing			

The wizard makes it easy to create HDL source code wrappers for clock circuits customized to your clocking requirements. The wizard guides you in setting the appropriate attributes for your clocking primitive, and also allows you to override any wizard-calculated parameter. In addition to providing an HDL wrapper for implementing the desired clocking circuit, the Clocking Wizard also delivers a timing parameter summary generated by the Xilinx timing tools for the circuit. The main features of the wizard include:

- Accepts up to two input clocks and up to seven output clocks per clock network
- Automatically chooses correct clocking primitive for a selected device
- Automatically configures clocking primitive based on user-selected clocking features
- Automatically implements overall configuration that supports phase shift and duty cycle requirements
- Optionally buffers clock signals

The functionality of the generated core can be viewed as:



Suppose we want to generate a 5 MHz clock which is in phase with 100 MHz input clock. Follow the steps below to achieve that:

Double-click on the Clocking Wizard. When the wizard opens, you will notice that there are five configurable pages (steps):

The first page (Clocking Options) has parameters related to input clock and clocking features. The input frequency value and range for the Basys3 or the Nexys4 DDR are given as defaults and we will keep it at that value. Make sure the name of the IP is changed to **clk_5MHz**.

Page two (Output Clocks) parameters which are related to the output clocks and desired frequencies. Change the Requested Output Frequency to 1.000 MHz and notice that it is highlighted red, indicating an illegal option. Change it to **5.000 MHz** for now. Also keep the default settings with the Reset and Locked boxes checked. We will unclick the **RESET** option and create an asynchronous reset. We like to see when the clock is stable.

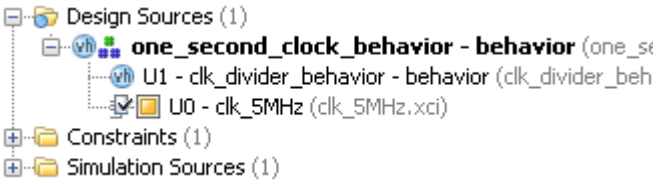
Page three (MMCM Settings) contain the settings needed to fine-tune the operation of the MMCM. In this page you can find ways to edit the jitter settings, clock divisor values and phase shift values. Leave everything as defaults on this page.

Page four (Port Renaming) allows you to rename the MMCM clock input and output ports, leave everything as is on this page.

Page five (Summary) summarizes the various settings you have selected in the previous pages. This page is not editable.

Clicking **Okay** at the bottom of the wizard confirms all the settings and generates the IP to be integrated into the design. Vivado will ask you whether you want to generate output products, this is a necessary step to properly use the IP. Architectural sources need to explicitly generate output products.

Assuming the core is named properly and instantiated correctly in the VHDL code, it will be integrated directly into Design Sources tree of the Sources window.



1-1. Design a one-second pulse generator. Use the clocking wizard to generate 5 MHz clock, dividing it further by a clock divider (written in behavioral modeling) to generate one second period signal. The steps on using the Clocking Wizard described above (and the resultant instantiation template) can be used for this exercise. Use the on-board clock source of 100 MHz, the BTNU button to reset the circuit, SW0 as enable, LED0 to output the generated one second signal, and LED15 to output the MMCM lock signal. Go through the design flow, generate the bitstream, and download it into the Basys3 or the Nexys4 DDR board. Verify the functionality. *Make sure you specify the language of the project to VHDL to generate the appropriate output files.*

Since the 7-segment displays on the Basys3 or the Nexys4 DDR board use common cathodes and a particular display is illuminated by asserting the corresponding anode pin, a scanning circuit is required to display information (digits) on more than one display. This circuit should drive the anode signals and corresponding cathode patterns of each digit in a repeating, continuous succession, at an update rate that is faster than the human eye can detect. In order for each of the digits to appear bright and continuously illuminated, all desired digits should be driven once every 1 to 16ms, for a refresh frequency of 1 KHz to 60 Hz. If the update or "refresh" rate is slowed to around 45 Hz, most people will begin to see the display flicker.

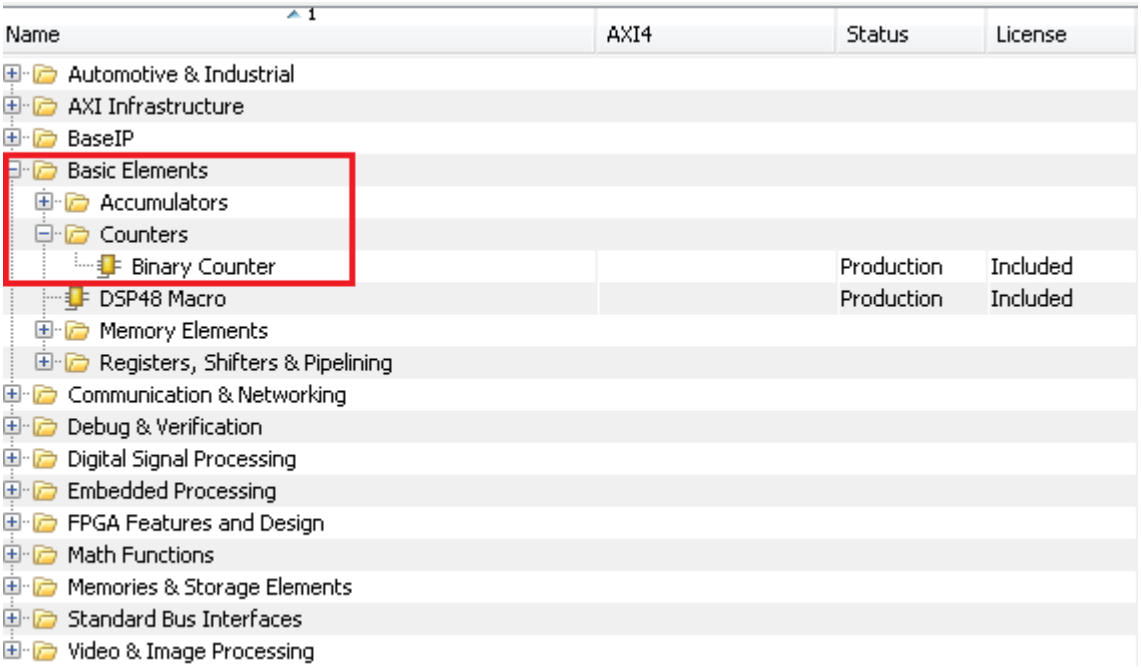
1-2. Modify the design of Lab2_2_1 (binary to BCD converter) to display the 4-bit binary inputs converted to BCD values on two 7-segment displays (instead of one 7-segment and one LED). Use the 100 MHz clock source to generate a 5 MHz clock and the appropriate clock divider circuit to drive the two 7-segment displays with a refresh rate of about 500 Hz. Generate the bitstream and download it into the Basys3 or the Nexys4 DDR board to verify the functionality.

IP Catalog System Part 2

The IP Catalog system, available in the IP Catalog of the Vivado tool allows you to configure and generate various functional cores. In IP Catalog, the cores are grouped according to functionality which varies from simple basic cores such as an adder to quite complex cores such as the MicroBlaze processor. It also covers cores of various application areas ranging from Automotive to Video and Image Processing.

The process of configuring and generating the cores is similar to the Architectural Wizard. The cores will use various resources including LUT, CLB, DSP48, BRAM etc. as needed. Let us look at how to configure and generate a counter core.

The Binary Counter core generation can be started by double-clicking the Binary Counter entry under the Counters sub-folder located under the Basic Elements branch of the IP catalog.



When invoked, you will see only two pages of the configuration. The configuration parameters of the core include:

Implement Using: Fabric or DSP48
Output Width
Increment Value
Loadable, Restrict Count, Count Mode (Up, Down, UPDPWN), Synchronous Clear, Clock Enable and various other settings.

You can select the desired functionality and click on the Okay button. The synthesized output and simulation files still need to be explicitly generated.

2-1. Use IP Catalog system to generate a simple 4-bit counter core which counts up from 0 to 9 (Hint: Use Threshold output when configuring the counter core). Instantiate it two times to create a two digit BCD counter which counts up every one second. Use Architectural Wizard to generate a 5 MHz clock and then use behavioral modeling to generate 1 Hz precise signal to drive the counters. Display the result on the two 7-segment displays. The design input will be a 100 MHz clock source, a reset signal using the BTNU button, and an enable signal using SW0. Verify the design functionality in hardware using the Basys3 or the Nexys4 DDR board.

Conclusion

In this lab, you learned about the Architectural Wizard and the IP Catalog system available in the IP Catalog of the Vivado tool. You used the Architectural Wizard to generate a 5 MHz clock and the IP Catalog to generate a counter. The IP Catalog system is a powerful tool providing various functional blocks enabling higher productivity.