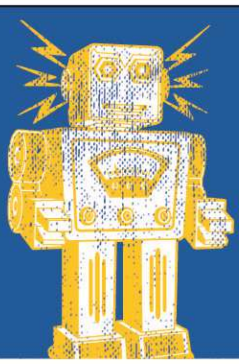
Elektronik



Digital System Design

Distance Learning Letter

Specification of Calculator Project

Part 6: Synthesis & Implementation

Gefördert von MA23 Wirtschaft, Arbeit Statistik StaDt Wien

iPLAT – Competence team for Innovative
Platforms for Mixed-Hardware/Software Systems
MA23 Project 18-06

Version: 0.1
Author: P. Rössler, R. Höller, C. Reisner

Copyright Notice

Introduction

At this point we have a VHDL design of the calculator project which was already successfully tested during various simulation runs, both on unit-level as well on the top-level. In the next step we want to implement the design on a Xilinx Artix-7 FPGA device using the already known FPGA implementation software “Xilinx Vivado”. As a final test of the calculator we will run the design on the Digilent Basys3 board.

Synthesis & Implementation

In order to implement the calculator project on the Xilinx Artix-7 FPGA simply follow the instructions in the distance learning “Getting Started with “Vivado HL WebPACK”. The main steps are as follows:

- Create a new Vivado project in the “impl” sub-folder of the project which is named, for example, “calculator”.
- Add all VHDL design files of the calculator to the project (all files from the sub-folder “vhdl”) but DO NOT add the testbenches (files from the sub-folder “tb”).
- Choose “xc7a35tcpg236-1” as the part since this is the FPGA device that is contained on the Basys3 board.
- Run synthesis and have a look at the status messages which are classified by Xilinx Vivado into three categories: “Info Messages” can be ignored. “Errors” have to be fixed by the user at all times since the tool cannot proceed when the cause of the error message is not found. “Warnings” may have two reasons:
  - For the calculator project you may see some warnings reporting that the signals led_o(14:0) are never used in the design. However, the reason is clear since LED0-14 are not used for the calculator. Therefore, you can ignore such kind of warnings.
  - Other warning may indicate serious problems, for example, when the reset state of some registers is not defined (since you have forgot to encode them in the corresponding VHDL processes). This kind of warnings must be fixed by the user since there is a chance that the design will not run properly on the FPGA. In general, have a look at all warnings and try to find out their reason! You should be able to explain all warnings in your design during the presentation of your project!
- Create a constraints file to define the clock frequency (100 MHz), the I/O configuration voltage as well as I/O pin assignments for all ports of the top-level design.
- Generate the FPGA bitstream. Again, have a look at all errors and warnings and try to find the reason for these messages!
- Download the generated bitstream to the FPGA and test your project on the board.
- If your design does not run on the board this may have two possible reasons: You (i) did not test your design thoroughly during simulation or (ii) you have ignored the Vivado warnings.

Finally, we want to have a closer look in the process of synthesis and implementation, done by Xilinx Vivado.

Synthesis

The logic synthesis tool transforms the RTL-level code (written in VHDL) into a gate-level netlist (i.e., a digital circuit). This transformation has to consider “constraints” which have to be met, like the desired clock frequency or the number of available logic resources in the FPGA (which are implicitly defined by the given FPGA device that was selected by the user during creation of the Vivado project). First, the VHDL code is checked for correct syntax and semantic. Status messages (info & error messages, warnings) are reported during the whole process. All messages are also saved to a log file.

Modern EDA (Electronic Design Automation) tools, like Xilinx Vivado, visualize the results of the synthesis in a graphical way. Figure 1 shows the schematic at the top level of the calculator project. This view can be activated in Xilinx Vivado if you select

RTL Analysis > Open Elaborated Design > Schematic

from the “Flow Navigator” window (do not confuse this with “Synthesis > Open Synthesized Design > Schematic” which shows an optimized post-synthesis view of the design, instead!).

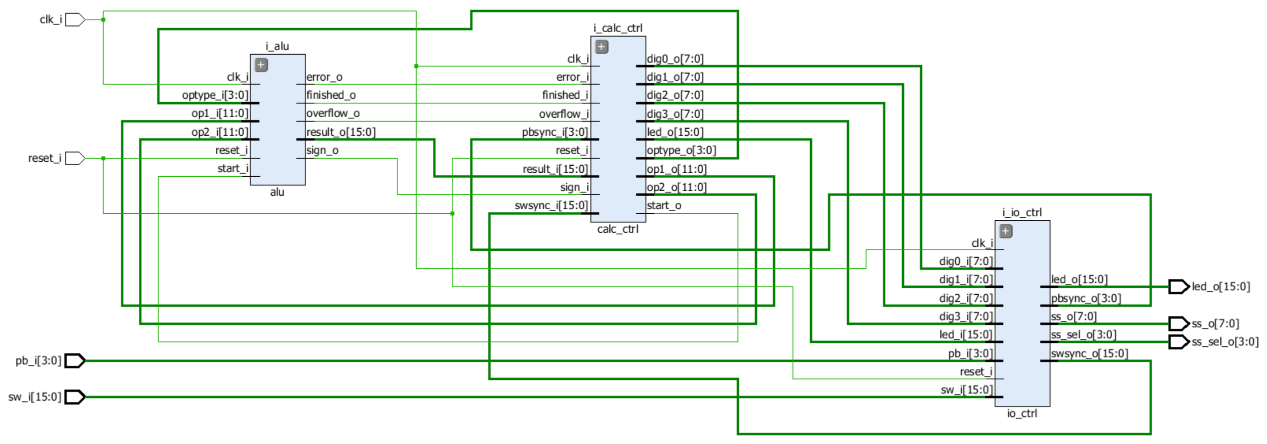


Figure 1: Top-level Design of the Calculator

Figure 1 shows that the “calculator” design consists of the three sub-units “io_ctrl”, “calc_ctrl” and “alu”. Starting from the top-level representation of the calculator, it is possible to step through the hierarchy of the design in order to see the internal circuitry of the individual sub-modules (in Xilinx Vivado, select the sub units and use the blue arrows).

Implementation (Place & Route)

As already mentioned, the synthesis tool generates an optimized gate-level netlist which consists of logic primitives (LUTs, flip-flops, I/O buffers, …) and their interconnections (so called “nets”). The logic primitives have to be placed and routed onto the FPGA which is done in the “implementation” step. The desired clock frequency as well as the connections of the top-level VHDL ports to the FPGA I/O pins have to be defined by the user previously. After the design was implemented the user can have a look at the results of the place & route designs steps. This is done by selecting

Implementation > Open Implemented Design

in the Vivado “Flow Navigator” window which opens another window called “Device”. The device window shows a physical representation of the design, including the FPGA logic cells and the IO pins. For example, Figure 2 shows the place & route results of the calculator design. By using the “Highlight Leaf Cells” command one can highlight the logic cells of the sub-units “io_ctrl”, “calc_ctrl” and “alu” in different colors (see Figure 2).

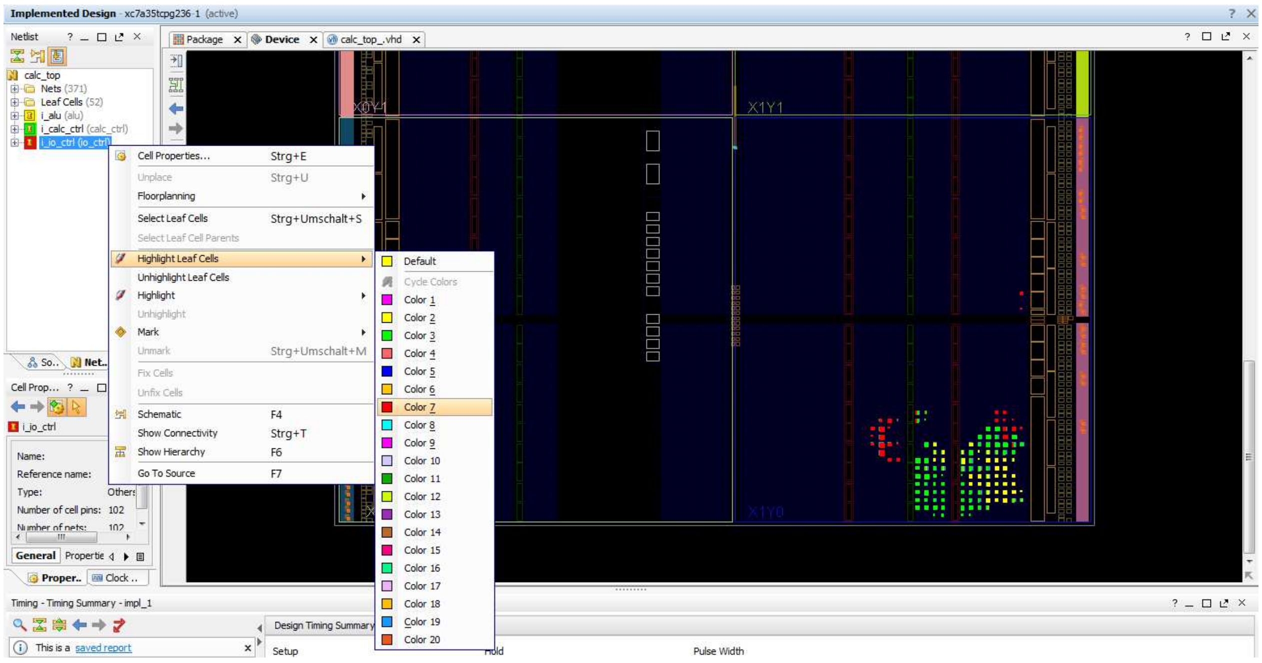


Figure 2: Calculator Design after Place & Route

A left-click on the “Routing Resources” icon turns on the visibility of the routing resources (see Figure 3).

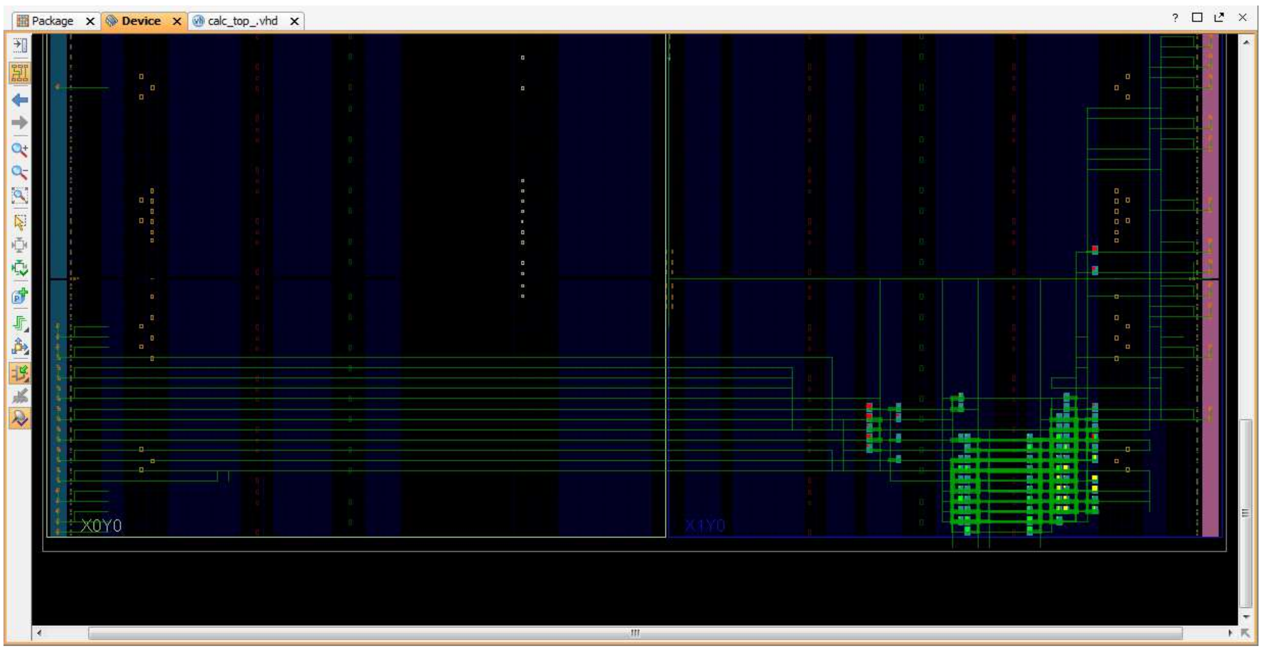


Figure 3: Calculator Design after Place & Route (with Routing Resources visible)

If you hover over an IO pin with the mouse pointer the name of the IO pin becomes visible (see Figure 4).

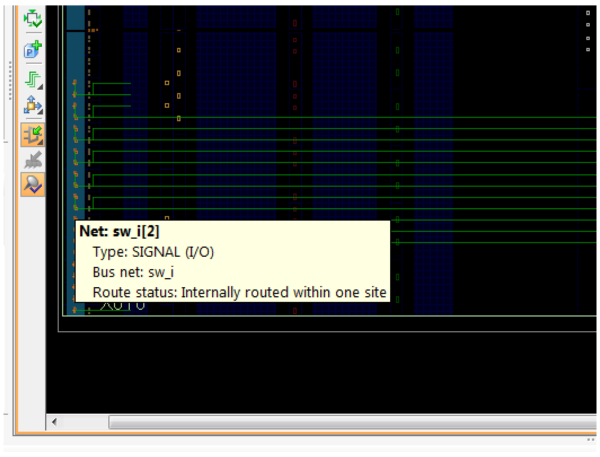

Net: sw_i[2]
Type: SIGNAL (I/O)
Bus net: sw_i
Route status: Internally routed within one site

Figure 4: I/O resource of the Calculator Design

By using the “Zoom In” icon you can have a look at each single logic cell in the FPGA device (see Figure 5).

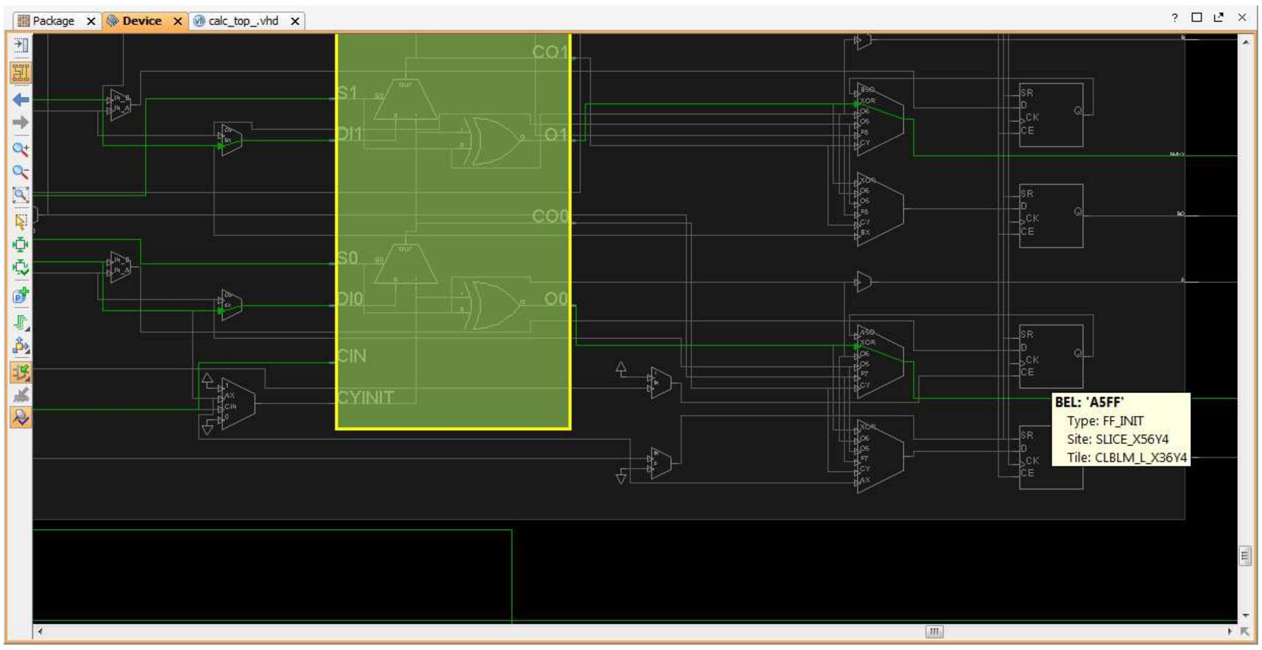


Figure 5: Logic Cells of the Calculator Design

Project Presentation

Congratulation! You have mastered the final project of the “Digital Design” course! Please have a final look into the distance learning letter “Overview or Calculator Project” in order to prepare yourself for the presentation of your project!

Version

Version 0.1, 2017-02-25	Initial release of this document

If you find mistakes or inconsistencies, please report them to the course supervisors via email. Thank you!