Design of Digital Electronic Systems (DAT094) Lab 4: Downloading and evaluating on the FPGA

Lars Svensson

Version 2.0, October 11, 2021

1 Introduction

In this final lab in the series, you will use and extend the knowledge and skills gained in the previous labs in several ways: you will download syntheiszed code and test it on an FPGA board, you will learn how to adapt processing rates using clock-enable signals (a useful technique when dealing with slow peripherals).

This lab will again be carried out in pairs; each student pair uses one FPGA board. Hardware setups will be available in the lab halls starting on Tuesday morning. As a reminder, each student is still expected to produce and submit an individual report!

You may notice that these instructions are briefer than for the previous labs. This does not mean that this lab assignment is easier or will take less time, but rather that the skills you learned in the previous sessions should make detailed instructions less necessary; also, other documents will provide much of the information you need. Do not hesitate to ask the TAs if there are still questions!

2 Preparation

Read all of "Introduction to Xilinx Vivado", so also repeating what you read for lab 3. "Nexys4 FPGA board reference manual" contains a more complete description of the board. Depending on your previous exposure to digital development

boards, there may be some concepts that are new to you, but much of what you read here should be familiar from lectures and from previous experiences. (We will only use a small portion of the development-board feature set.)

The document "Hints on clocking", which is also available on the homepage, outlines design practices for clock definition and distribution in FPGA implementations, in particular when different parts of a design have different speed requirements. Read through this document as well.

Also, the lab will use an AD/DA board described in a document titled (you guessed it) "The AD/DA board".

3 Getting started

The lab starts much as Lab 3 did: by creating a lab directory and a Vivado project.

- Create a folder and a Vivado project called `lab4a`.
- Locate the VHDL file `MAC_gen_min_full.vhdl` and all files needed to compile and eventually simulate it. Copy these files into `lab4a` and include them in the project.
- Open and inspect `MAC_gen_min_full.vhdl`.

The code you will be working with is considerably more complex than what you have seen so far, but several parts should be recognizable. The core of the system is intended to apply a filter to a continuoulsy sampled signal. DA and AD converters are also included in the full system, in order to provide input signals and allow observation of the result; "glue logic" to interface with the converters is described in the VHDL files. (The system is designed to allow other operations as well, such as using different sample rates at in- and output; we will not be using those capabilities in this lab.)

1→ - Draw a block diagram of the system described by `MAC_gen_min_full.vhdl` and its associated files. Label components and signals, and indicate (for example with arrow heads) whether each port is an input or an output.

2→ - What are the `GENERIC` parameters used in `MAC_gen_min_full`? What aspects of the hardware do they control? How?

3→
- How are the signal wordlengths determined for the signals seen in your block diagram?
- The converters on the AD/DA board are the MCP4822 and MCP3202. Find the documentation for these parts online; it will be handy during the next task.
- Several clock signals are used in this system: the system clock, the sample
4→ clock, and the SPI clock. How are these signals generated? Describe the function of the hardware blocks involved. What are the relations of the different clock frequencies?

Your Lab-3 results gave an indication for a highest possible clock frequency for the `MAC_gen_min` block. You can use these values as a starting point for synthesis constraints to use for the full system as well. The provided constraint file specifies how design signals should be connected to the package pins.

- Edit `MAC_gen_min_full` to use the 12-bit, 17-tap version of `MAC_gen_min` which is downloadable from Canvas. (You will need to include the appropriate files in the project, etc.)
- Edit the constraint file such that the initial frequency is set to your Lab-3 value.
- In Vivado, perform synthesis and implementation for `MAC_gen_min_full`. If necessary, adjust the clock frequency / timing constraints to find the
5→ highest clock rate that results in a non-negative slack. Compare the clock frequency with your Lab-3 values.

6→
- Also, make note of the amounts of hardware spent on this design.

You will have noted that the achievable clock rate is nowhere close to the 100 MHz provided on the card. Next, you will return to the serial implementation of the same functionality, which should have a better chance to meet the constraints.

- Proceeding as above, create a directory and Vivado and QuestaSim projects named `lab4b`. Copy all files needed for the design `MAC_ser_min_full` and the 12-bit, 17-tap version of the corresponding filter into your directory and add them to the projects.

7→
- Compare and contrast the new design with the previous one. Consider both the internal differences and its interface with other blocks. Are all the interface requirements expressed in the `ENTITY` declarations?

8→ • Carry out synthesis and implementation in Vivado of `MAC_ser_min_full` in order to find the highest usable clock frequency, this time starting at 100 MHz. Make note of the amount of hardware used.

4 Execute on FPGA

You should now have a system which meets the timing constraints with the clock that is available on the FPGA board.

To check the functionality of the design, it is necessary to apply an input signal to the board and monitor the outputs. A daughter card with A/D and D/A converter is available for this purpose.

- Ensure that the AD/DA board is correctly connected to the FPGA board, and that the development board is connected to the lab computer via the USB cable.
- Following the procedure outlined in "Introduction to Xilinx Vivado", generate a bitstream to program the Nexys4 board and download the bitstream to the board.
- Connect a signal generator at the AD input to provide a 1-kHz sinewave signal of $1V_{pp}$. Use the oscilloscope to monitor both the input and output signals and verify that the signal makes it through the digital domain and back out.
- 9→ Increase the signal frequency until the amplitude of the output signal starts decreasing. At what frequency[1] has the amplitude decreased by a factor of 0.7?

5 A second parameter set

The parameter values in the package file determine the frequency response of the filter, and also the implementation cost. Next, you will replace these values with another parameter set and re-investigate the cutoff frequency. The new parameter sets are available in new file sets with the same names as you have seen so far, but with `12_17` replaced with `12_101`.

[1] An amplitude factor of $\frac{1}{\sqrt{2}} \approx 0.7071$ corresponds to an attenuation of 3dB, which conventionally defines the cutoff frequency of a filter.

- Create new projects and working directories for the new versions of the filter.
- Repeat the previous steps for the new versions in order to evaluate the maximum clock frequencies, the cutoff frequencies, and the hardware amounts. Compare with the results from the previous sections. Can you explain the differences?
- Summarize your impressions and the insights you gained from the several filter implementations you have investigated during this lab assignment.

6 Wrap-up

After completing this lab session, you are expected to be able to carry out the following tasks:

- Construct a simple test bench and *do* file to test the functionality of a small sequential VHDL block.
- Generate a configuration for a small FPGA board, download it onto the board, and test the functionality with external stimuli.
- Compare and contrast a serial and a parallel implementation of a filter implementation for hardware cost and performance.