Lab #2

Microprocessor design

Professor:
Yvon Savaria
yvon.savaria@polymtl.ca

Inspired from the work of:
Mickaël Fiorentino
mickael.fiorentino@polymtl.ca

Lab instructor:
Justin Pabot
justin.pabot@polymtl.ca

Lab instructor:
Timothée TREMBLY
timothee-mateo.trembly@polymtl.ca

Automne 2023

CONTENTS

1 INTRODUCTION

The complexity of a very large scale integrated circuit, such as a microprocessor, is impossible to manage with only layout edition Electronic Design Automation (EDA) tools that we have used in the first laboratory. To better handle the complexity of integrated circuits, while keeping developement cost within boundaries, *synthesis* tools, associated with Hardware Description Languages (HDL) such as VHDL or Verilog, as well as automated *Place and Route* (PaR) tools, allow to generate the layout of a circuit from its high-level behavioral description.

1.1 OBJECTIVES

This second laboratory consists in designing a simple microprocessor. The goal is to familiarize the students with the design, the simulation, and synthesis, and the physical implementation of digital integrated circuits using industry-standard EDA tools. This laboratory passes through most of the standard design flow, as represented on FIGURE 1. In particular, this laboratory enables you to:

- Design a microprocessor in VHDL.
- Perform behavioral simulations (VHDL models), and timing simulations (post-synthesis and post-PaR netlists), of the processor.
- Perform logical synthesis of the processor using the GPDK045 ASIC design kit.
- Perform the placement and the routing of the processor from its post-synthesis netlist.

Moreover, the following skills are not evaluated, but remain interesting to take a look at if time permits:

- Add Design For Test (DFT) capabilities to the processor, and generate test vectors.
- Evaluate the power consumption of the processor from its post-place-and-route information, and the activity it has generated in timing simulations.

The processor documentation contains all the information that are necessary to complete its design. With the help of the tutorials and the documentation, you will first design each of the modules of the processor, which you will then use to design the pipelined *core*.The VHDL language is presented in class. When you are finished with the behavioral modelling of the processor, you will follow the digital tutorial, which will guide you through the simulation, the synthesis, the testability, the placement and routing, and the power consumption evaluation steps using a BCD counter. The VHDL model of the latter is also provided if you wish to practice on a simple case before tackling the processor.

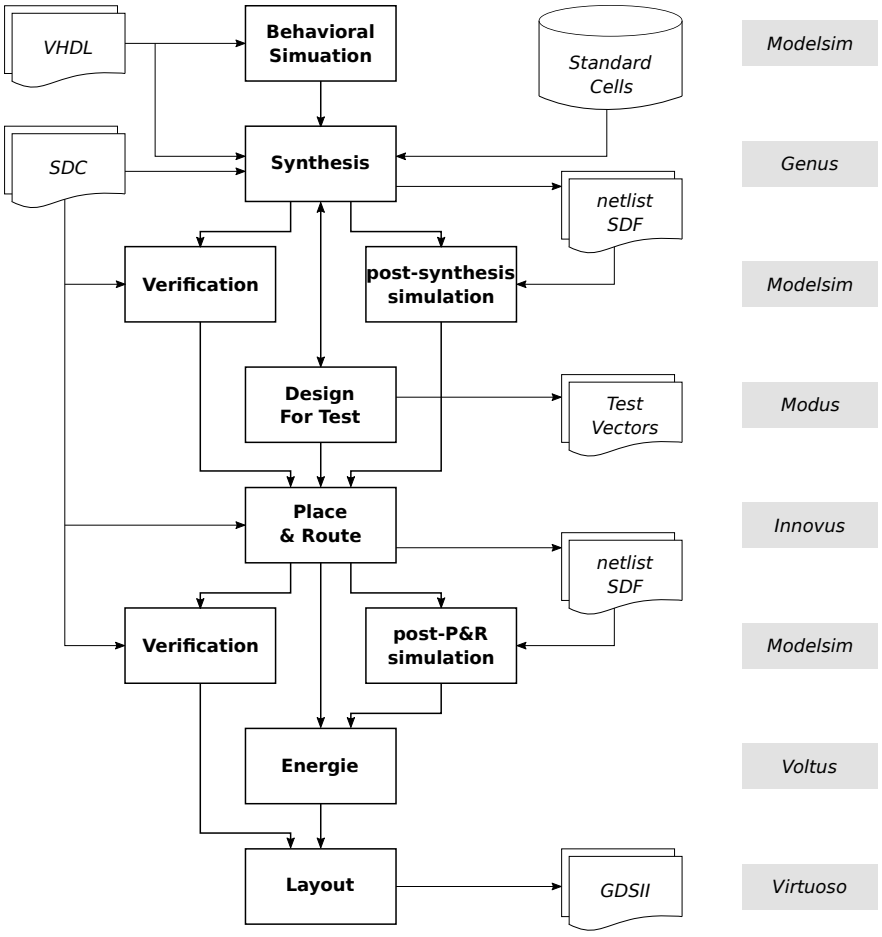

FIGURE 1: Design flow

1.2 WORK ENVIRONMENT

The work environment associated with this second lab is available as a git repository. Download this repository on your workstation before you begin:

```
% cd ~/Labs
% git clone https://git.step.polymtl.ca/ele8304/lab2.git
```

This folder is organized as follows:

```
asm/........................... # Assembly benchmarks
scripts/....................... # Design flow automation
sources/....................... # VHDL sources
constraints/................... # Constraints files
simulations/................... # Behavioral and timing simulations
implementation/................ # Synthesis, place-and-route
doc/........................... # Documentation, reports
```

1.3 GRADING SYSTEM

TABLE 1: Grading system

Modules	**/5**
Core	**/9**
Implementation	**/4**
Report quality	**/2**
Performances (Bonus)	**/2**

TABLE 1 details the grading scheme of the lab. 14 points are allocated to the design and the simulation of the modules and core in VHDL. 4 points are allocated to the design flow (synthesis, design for test, place-and-route, verifications). 2 points are allocated to the quality of the report. Finally, **2 bonus points** are 3 points are allocated to the performance and the power consumption evaluations.

1.4 DELAY PENALTIES

If you submit your laboratory after the deadline, your grade will be reduced by a delay penalty P_{delay}, which is calculated from the number of hours h of delay by the following equation:

$$P_{delay} = 0.5 \times \left[1 + \left(\frac{h}{24}\right)^2\right]$$

1.5 REPORT

Your report must include all the materials that you think are relevant to justify your analysis. This may include your simulation, synthesis, and implementation results, as well as any other information that you judge necessary to demonstrate your good understanding of the laboratory. 2 points are allocated to the quality of the report, including the quality of your sentences, the grammar, the figures, and the overall form of the report. Only *one copy per group* is required. We ask you to upload a compressed (`.zip`) file on Moodle before the deadline (see the Moodle calendar) containing:

- Your report in PDF format.
- Your VHDL sources (modules, core, test-benches).
- Your *tcl* scripts (simulation, synthesis, implementation).
- Your post-synthesis and post-PaR results (netlists, STA reports, DRC, LVS, power and area reports, simulation waveforms, *etc.*).

***Note**—We are particularly interested in having your feedback regarding the difficulties that you may have encountered during the lab.*

2 GUIDELINES

2.1 MODULES

In this first section, we ask you to design the *modules* of the processor. Detailed descriptions of the expected behavior of these modules, including diagrams and VHDL interfaces, are presented in the processor documentation. In particular, you are asked to:

- Understand and design the VHDL model of all the modules presented in the processor documentation, and validate their behavior using a test bench and behavioral simulations. Use the *exact entity* presented in the documentation. (**5 pts**)

To speed things up, a number of modules are already given to you, in particular:

- The Register File
- The *Program Counter*
- The Arithmetic and Logic Unit (ALU)

Hence, you only need to create the VHDL module of the 32-bit adder, as well as the test benches for all 4 modules.

2.2 CORE

In this section, we ask you to design the *core* of the processor, by reusing the modules designed previously. The behavior of the core pipeline is detailed in the processor documentation. In particular, you are asked to:

- Design the VHDL model of the core by following the specifications of the documentation. Use the *exact entity* presented in the documentation. (**6 pts**)
- Validate its behavior using a test-bench and a behavioral simulation showing the correct execution of the *benchmark* **riscv_basic.S** provided as part of the work environment. (**3 pts**)

To be able to execute a program with your processor, we also provide a testbench. Do note that this testbench instantiates the dual-port memory (`dpm.vhd`) provided as part of the work environment. Note that you can also perform additional tests using your own programs written in assembly by using the Makefile in the *asm* folder. Make sure to include the instruction memory initialization file (`*.hex`), and verify that it has been properly updated by the compilation process.

2.3 IMPLEMENTATION

In this section, we ask you to perform the physical implementation of the processor using the 45 nm GPDK045 Cadence design kit. In particular, you are asked to:

- Perform the synthesis of the processor, **without including the DFT**. DFT can be implemented if time permits, and would be grounds for a bonus. Verify that the timing constraints are met using STA, and validate the processor behavior using a post-synthesis timing simulation. Use the same test-bench as for the behavioral simulation, and highlight delays in the circuit. (**2 pt**)
- Perform the placement and the routing of the processor. Verify that the timing constraints are met using STA, check the integrity of the layout with DRC and LVS verifications, and validate the processor behavior using a post-PaR timing simulation (same test-bench). (**2 pt**)

2.4 PERFORMANCES - BONUS

In this section, we ask you to evaluate the performances and the power consumption of your processor post-implementation. In other words, compare the running time of the *benchmark* **riscv_basic.S** with the energy consumed during its execution. In particular, you are asked to:

- Perform a post-implementation timing simulation of the processor using the *benchmark* **riscv_basic.S**, and record the activity generated by the simulation in a `vcd` file. Use the testbench previously used for behavioral simulation.
- Determine the performance of your processor in Millions of Instructions Per Seconds (MIPS), using information provided by your performance counters.
- Evaluate the average power consumption P (in mW) of your processor during the execution of the *benchmark* **riscv_basic.S**, using the activity recorded in post-implementation simulation. The $MIPS/P$ ratio that you obtain is the figure of merit chosen to evaluate performance.

N.B.: Do note that it is possible to make sizing modifications (including circuit and design flow modifications) in order to maximize performance. If you choose to do this, verify that the timing constraints are still respected using STA.

The number of points awarded for some/all of the tasks presented above will be left to the discretion of the lab instructors, taking into account the care taken in their realization and the average progress of the group.