



Politecnico di Torino
III Facoltà di Ingegneria

Lab 3 Report

Integrated Systems Architecture

Master degree in Computer Engineering

Authors: ISA36

Nicole Dai Prà s274501, Leonardo Izzi s278564

January 29, 2021

Many thanks to Prof. Mariagrazia Graziano for providing us with this template.

Contents

1	Introduction	1
2	Processor architecture	2
2.1	IF stage	2
2.2	ID stage	2
2.3	EXE stage	2
2.4	MEM stage	3
2.5	WB stage	3
2.6	Control unit	3
3	Synthesis	5
4	Place and Route	7
4.1	Timing report	7
4.2	Area report	8

CHAPTER 1

Introduction

In this lab we had to develop a scalar, in-order, RISC-V-lite processor. To the base unit, as a second task, we had to add a special functional unit to compute the absolute value of a number.

As required, there is a GitHub repository available at the following link: https://github.com/leoizzi/isa_labs/tree/main/lab3.

The processors are separated in two different folders, in `riscv` there is the base model while in `riscv_abs` there is the extended one. The folders share the same structure, hence it is reported only once:

- **innovus**: it contains all the files produced during the place and route by Innovus.
- **netlist**: it contains the output of the synthesis process.
- **sim**: it contains all the simulation-related files, such as the data to initialize the RAM and the ROM, as well as the simulation script.
- **src**: it contains the processor's source files to be synthesized and simulated.
- **syn**: it contains the synthesis scripts.
- **tb**: it contains all the testbenches developed to test the units and the processor itself.

CHAPTER 2

Processor architecture

In this chapter we will describe how we have implemented the processor. It is based on the structure studied during the lectures. At a glance, it has 5 stages, the control unit is hardwired and it does not support forwarding.

2.1 IF stage

In this stage new instructions to be executed are fetched from the ROM. Its structure is pretty simple: there is the program counter register (PC) that is updated at every clock cycle either with the target address, that is the address of a jump, or with $PC + 4$. In case of a stall, the enable signal of the register is de-asserted.

The current PC value is used to access a small ROM which is used only during simulations as it cannot be synthesized with the processor. The current PC value, $PC + 4$ and the ROM output are fed to the IF/ID registers.

2.2 ID stage

Here the decode of the instruction fetched in the previous cycle is performed. The register file, as well as the immediate generator, are in this stage. The register file is accessed regardless of the instruction type, and the control unit decides how the immediate should be extended. These values, along with the instruction PC, its subsequent value and the destination register address, are propagated to the ID/EXE registers.

The register file is capable of supporting two read and a single write per clock cycle. The write operation is done by the WB stage on the falling edge of the clock signal. This means that it is possible to reduce the number of stalls by one, because when an instruction writes in the RF from the WB stage its result can be immediately accessed by the instruction in the ID stage.

To allow the control unit to check whether there is the need for a stall, the source registers are sent to it.

2.3 EXE stage

The ALU supports two inputs, A and B. Based on the instruction, two muxes select which values have to be taken as operands. For the A port, the choice is between the PC , $PC + 4$, 0 or the value coming from the RF's port 1. For the B port, the choice is between the immediate, the RF's port 2

or 0. In the ALU, all the functional units perform the computation in parallel, then through a mux the result is selected. The functional units present in the ALU are:

- the adder;
- the shifter;
- the logicals (AND and XOR);
- the set-less-than;
- the absolute value (only for `risc_abs`).

Moreover, there is also a comparator that checks if the result is 0. In the affirmative case, it asserts the `zero` signal.

To support branches and jumps there is a second adder, that takes as inputs the `PC` and the immediate value shifted to the left by one. This value represents the target address of the IF stage. However, it is selected by the IF stage only if two conditions are true: in the control word the bit `JMP_ENABLE` is set to 1, and if

- the instruction is a `jal` (detected by looking at the mux control signals, that is $A = NPC$ and $B = 0$)
- the instruction is a `beq`, hence if the `zero` signal is asserted.

Finally, to support `store` operations, there is a direct connection between the second register port output and the EXE/MEM registers.

2.4 MEM stage

All instructions but `lw` and `sw` bypass the memory and their data is directly propagated to the MEM/WB registers. The two aforementioned instructions, instead, are able to respectively read or write the RAM, which as for the ROM is not synthesized but only simulated.

The RAM has a port dedicated to read operations and one dedicated to write operations, but only one of them can be executed at any time. This has been done only to simplify the testing, since in this way there is no risk to misuse the tristate transistors. To distinguish between read and write operations a dedicated `wr` signal has been added.

Since the given test file assumes that the memory has been preloaded with the `.data` segment, this RAM reads on reset its initialization content from a file.

2.5 WB stage

In this stage there is simply a mux that, based on the instruction, selects if in the RF should be written the data coming either from the ALU or the memory. The CU keeps track of whether the instruction is allowed to write content in the RF or not, since the RF has an additional signal that enables the write operation.

2.6 Control unit

The control unit is hardwired, since for each instruction read in the IF stage it loads all the control signals inside the registers and outputs them at the right time. This is achieved by pipelining the control word in the same way as the datapath.

In the ID stage it detects possible data hazards by looking at the data source addresses and the destination address in the ID/EXE and EXE/MEM pipelining registers. If an hazard is detected, the control unit stalls the processor by inserting NOPs in the proper stages.

The control unit is also notified whenever the datapath is about to jump, because it needs to flush the IF and ID stages, as their work is not needed anymore.

CHAPTER 3

Synthesis

For the synthesis we have followed the standard flow used in the previous laboratory sessions, hence we will not repeat it here.

We synthesized both the processors. The critical path lies in the EXE's adder, hence they have a very similar critical path and requires 2.7 *ns* to be traversed.

Even the area occupation is similar, as the functional unit added to `riscv_abs` is rather small. Indeed, the total area difference between the two versions is $\sim 2\%$. The area results are reported below.

riscv

Report : area

Design : core

Version: 0-2018.06-SP4

Date : Sun Jan 10 18:18:58 2021

Number of ports: 561

Number of nets: 7720

Number of cells: 6242

Number of combinational cells: 4783

Number of sequential cells: 1446

Number of macros/black boxes: 0

Number of buf/inv: 839

Number of references: 36

Combinational area: 5745.334081

Buf/Inv area: 597.435990

Noncombinational area: 6538.811763

Macro/Black Box area: 0.000000

Net Interconnect area: undefined (Wire load has zero net area)

Total cell area: 12284.145844

Total area: undefined

riscv_abs

Report : area

Design : core

Version: 0-2018.06-SP4

Date : Sun Jan 10 18:44:01 2021

Number of ports: 659

Number of nets: 7997

Number of cells: 6452

Number of combinational cells: 4989

Number of sequential cells: 1448

Number of macros/black boxes: 0

Number of buf/inv: 962

Number of references: 36

Combinational area: 5945.366083

Buf/Inv area: 667.659989

Noncombinational area: 6547.855762

Macro/Black Box area: 0.000000

Net Interconnect area: undefined (Wire load has zero net area)

Total cell area: 12493.221846

Total area: undefined

If interested, the reader can find the other results in the respective `syn` folders.

CHAPTER 4

Place and Route

For what concerns the place & route, we have followed the guidelines provided for the first laboratory activity, hence there is nothing particular to be explained.

4.1 Timing report

Here it is reported the reg-to-reg analysis provided by Innovus after the RISC-V basic design has been placed and routed.

```
Path 1: MET Setup Check with Pin d/ids/reg_file/reg_i_24/q_reg[1]/CK
Endpoint:  d/ids/reg_file/reg_i_24/q_reg[1]/D (^) checked with trailing edge
of 'my_clk'
Beginpoint: c/curr_wb_reg[1]/Q                (^) triggered by leading edge
of 'my_clk'
Path Groups: {reg2reg}
Analysis View: MyAnView
Other End Arrival Time          1.350
- Setup                        0.042
+ Phase Shift                   0.000
- Uncertainty                   0.070
= Required Time                 1.238
- Arrival Time                 1.049
= Slack Time                    0.189
Clock Rise Edge                 0.000
= Beginpoint Arrival Time      0.000
Timing Path:
```

Pin	Edge	Net	Cell	Delay	Arrival Time	Required Time
clk	~	clk			0.000	0.189
c/curr_wb_reg[1]/CK	~	clk	DFF_X1	0.000	0.000	0.189
c/curr_wb_reg[1]/Q	~	wb_en	DFF_X1	0.098	0.098	0.287
U1945/A3	~	wb_en	NAND3_X1	0.000	0.098	0.287
U1945/ZN	v	n1509	NAND3_X1	0.060	0.157	0.346
U2625/B1	v	n1509	OAI21_X1	0.001	0.158	0.347
U2625/ZN	~	n1529	OAI21_X1	0.493	0.651	0.840
U2349/A2	~	n1529	NAND2_X1	0.006	0.658	0.847

U2349/ZN	v	n1530	NAND2_X1	0.277	0.935	1.124	
U3821/B2	v	n1530	OAI22_X1	0.027	0.962	1.151	
U3821/ZN	^	n2053	OAI22_X1	0.087	1.049	1.238	
d/ids/reg_file/reg_i_24/q_reg[1]/D	^	n2053	DFF_X1	0.000	1.049	1.238	
+-----+							
Clock Fall Edge	1.350						
= Beginpoint Arrival Time	1.350						
Other End Path:							
+-----+							
Pin	Edge	Net	Cell	Delay	Arrival	Required	
					Time	Time	
+-----+							
clk	v	clk			1.350	1.161	
U1796/A	v	clk	INV_X2	0.000	1.350	1.161	
U1796/ZN	^	n59	INV_X2	0.000	1.350	1.161	
d/ids/reg_file/reg_i_24/q_reg[1]/CK	^	n59	DFF_X1	0.000	1.350	1.161	
+-----+							

As it can be seen, the critical path is different from the one reported by the synthesis tool. In fact, the critical path obtained with the synthesis was in the EXE adder, while here it starts from the WB stage and arrives to the register file. We think that this difference is due to the fact that Innovus takes into consideration the delay of the nets, since it knows exactly their length and where their position. Moreover, this critical path makes sense because the logic is constrained to work in half clock cycle, because the register file samples the data on the clock's falling edge.

The critical path for the enhanced version of the processor is similar, as it arrives to the same register but to a different flip-flop, hence it will not be reported here.

4.2 Area report

The area reported by Innovus is a bit smaller than the one reported by Synopsys. For the basic RISC-V processor we obtained an area occupation of $11817.8 \mu m^2$, while for the processor with the ABS unit the area is $12172.7 \mu m^2$. This reduction is expected, since after the place and route flow the size is no more an estimation and the tool is able to better optimise the placing of the cells.