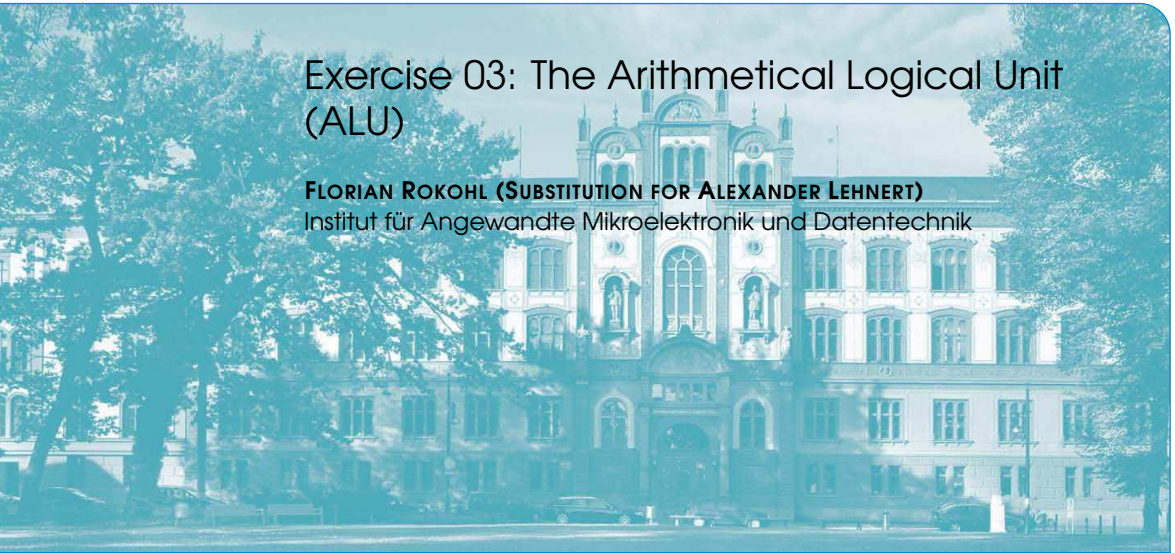


Exercise 03: The Arithmetical Logical Unit (ALU)

Florian Rokohl (Substitution for Alexander Lehnert)
Institut für Angewandte Mikroelektronik und Datentechnik

Table of Contents

Goal of this Exercise

- Build an **A**rithmetical **L**ogical **U**nit (ALU)
- The *brain* of the processor
- Arithmetical operations: `add`, `sub`
- Logical operations: `and`, `or`, `xor`
- Selection between the operations: Multiplexer

Recap: Last Exercise - Program Counter

Three componentes

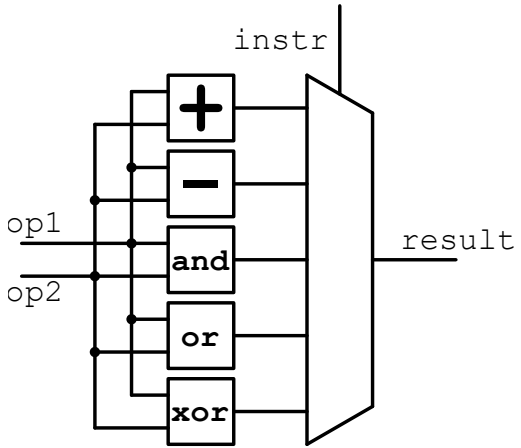
- Adder
- Flip Flop / Register
- Program Counter

Which parts **can be reused**?

Is the ALU **combinational** or **sequential** logic?

Schematic

Schematic



Packages

Definition of

- Constants
- Data types
- Functions
  - Header
  - Body
- and much more...

Package Structure

```
[<library and package declarations>]
package <PACKAGE_NAME> is
    <DECLARATIONS>
end <PACKAGE_NAME>;

-- package body
package body <PACKAGE_NAME> is
    <DEFINITIONS>
end <PACKAGE_NAME>;
```

Resource Sharing

- Consider ALU for addition and subtraction
- What needs to be implemented?

→ Addition and Subtraction

→ Resources can be shared

→ Or not ..

→ This is determined during synthesis

Resource Sharing

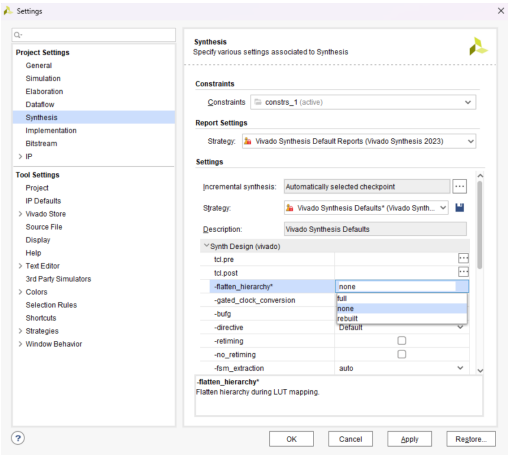
- Consider ALU for addition and subtraction
- What needs to be implemented?

→ Addition and Subtraction

→ Resources can be shared

→ Or not ..

→ This is determined during synthesis



What are Instruction Sets?

- Encoding of instruction
- Which bits are operands?
  - Direct (immediate)
  - Indirect (register)
- Which function is to be executed?
  - ALU-instruction
  - Control signals
  - Where is the result stored?

What are Instruction Sets?

- Encoding of instruction
- Which bits are operands?
  - Direct (immediate)
  - Indirect (register)
- Which function is to be executed?
  - ALU-instruction
  - Control signals
  - Where is the result stored?

This is non-trivial!

- Instruction Set defines control logic overhead

  How do we get the best Instruction Set?

- Infer it on our own → Digitale Systeme
- Let Vivado infer it?
- This lecture: `RISC-V` Instruction set

Task 1: Arithmetical Unit

- Implement units for
  - Addition
  - Subtraction
- Implement Arithmetical Unit
- Three inputs
  - Two operands
  - Instruction Select
- One output (result)

Analysis

- How to use packages
- Resource Sharing
- Mapping to FPGA resources

Task 2: Logic Extension

- Extend design from Task 1
- Implement units for
  - AND
  - OR
  - XOR
- Three inputs
  - Two operands
  - Instruction Select
- One output (result)

Analysis

- Simulation: Test your design
- Synthesis result: Mapping
- Demonstration: Execution on the FPGA

Optional Task: Full RISC-V

- Does the ALU now cover the RISC-V base integer instruction set?
- No, there are more instructions
- Complete your ALU
- Verify the design functionally