Digital Design & Computer Arch.

Lab 5 Supplement: Implementing an ALU

(Presentation by Aaron Zeller)
Frank K. Gürkaynak
Seyyedmohammad Sadrosadati
ETH Zürich
Spring 2024
[16. April 2024]

What Will We Learn?

- In lab 5, you will Implement an Arithmetic Logic Unit (ALU) in Verilog and evaluate its **speed and resource utilization.**
- Draw a block level diagram of the MIPS 32-bit ALU, based on the description in the textbook.
- Implement the ALU using Verilog.
- Synthesize the ALU and evaluate speed and FPGA resource utilization.

Part 1: Designing an ALU

- We will design an ALU that can perform a subset of the ALU operations of a full MIPS ALU.
  - 2 32-bit inputs
  - 4-bit AluOp signal to select the operation
  - 32-bit output
  - Output flag zero that sets to logic-1 if all the bits of the result are 0.

AluOp (3:0)	Mnemonic	Result =	Description
0000	add	A + B	Addition
0010	sub	A - B	Subtraction
0100	and	A and B	Logical and
0101	or	A or B	Logical or
0110	xor	A xor B	Exclusive or
0111	nor	A nor B	Logical nor
1010	slt	(A - B)[31]	Set less than
Others	n.a.	Don't care	

Part 1: Designing an ALU

AluOp (3:0)	Mnemonic	Result =	Description
0000	add	A + B	Addition
0010	sub	A - B	Subtraction
0100	and	A and B	Logical and
0101	or	A or B	Logical or
0110	xor	A xor B	Exclusive or
0111	nor	A nor B	Logical nor
1010	slt	(A - B)[31]	Set less than
Others	n.a.	Don't care	

- Verilog is a hardware description language and thus you should not leave results unspecified.
  - If the AluOP belongs to "Others" you should set the result to some value, e.g. 0.

Part 1: Designing an ALU- Block Diagram

- First, you need to draw a block diagram of the ALU, like the one seen in Figure 5.15 of the H&H textbook.

- You are free to chose if you want to draw the complete block diagram or split it up into modules and elaborate the modules, similar to the next slide.

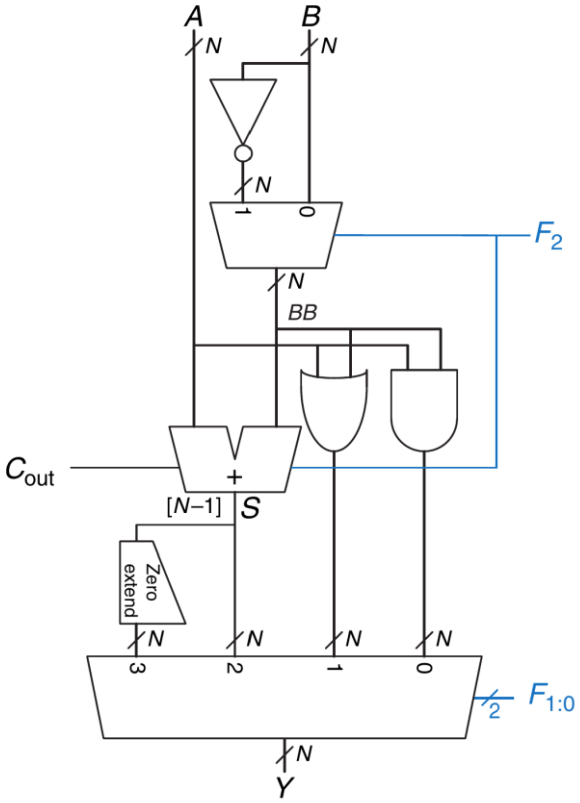
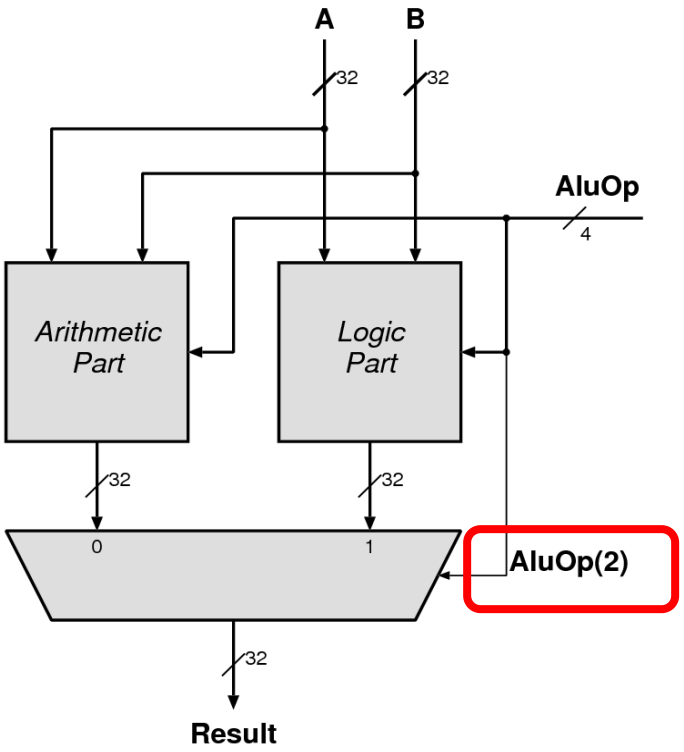


Figure 5.15 taken from the H&H textbook

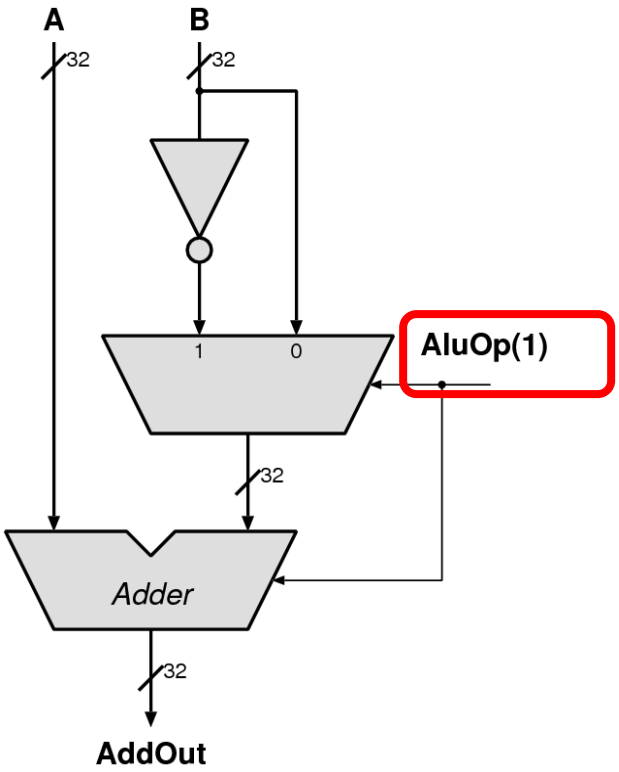
Part 1: Designing an ALU- Block Diagram

- A possible division in ALU Logic and Arithmetic operations:



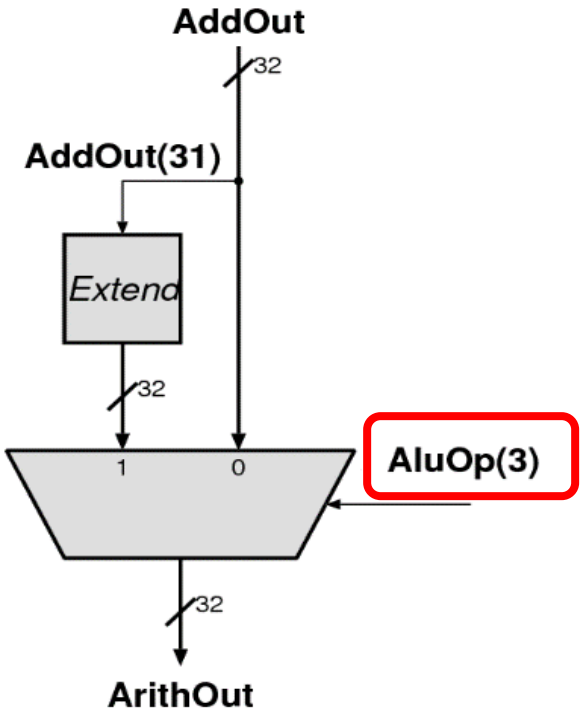
Part 1: Designing an ALU- Block Diagram

- A possible organization of ADD and SUB:



Part 1: Designing an ALU- Block Diagram

- A possible organization for SLT:



Part 2: Implementation

- Replace each block with a Verilog description.
- Synthesize and implement your design.
- We do not transfer the design to FPGA in this lab
  - No Constraint file - Bitstream generation will fail.
- At this point, we cannot verify the correctness of our circuit manually.
  - You will calculate how long the exhaustive search would take.
  - You learn how to use testbench to test the correctness of this circuit in lab 6!

Part 3: The performance of the circuit (I)

- In this lab, we will learn to check:
  - The speed (i.e., max frequency our circuit can run at)
  - The area (i.e., FPGA resource utilization).

- We will add a timing constraint to set the maximum delay that we would like our ALU to have.

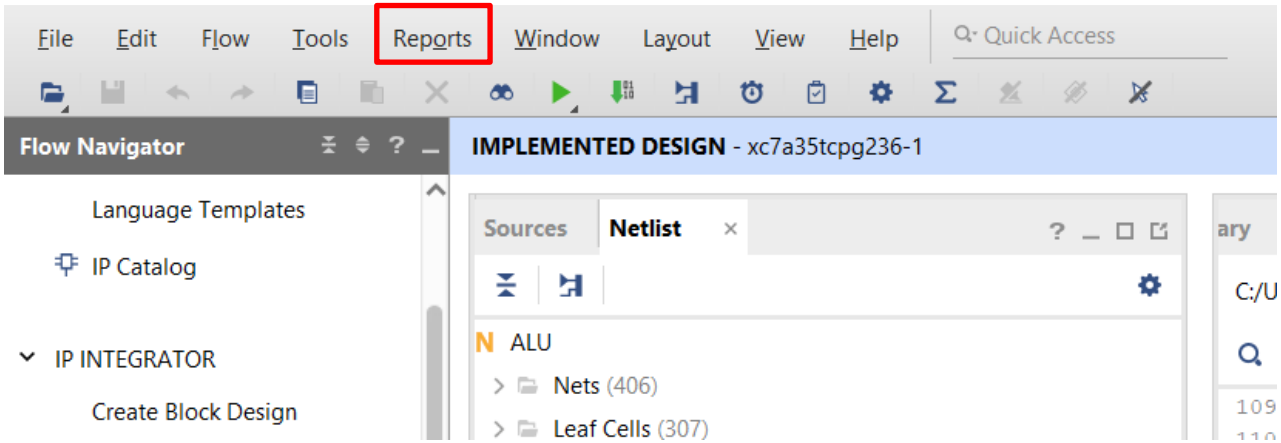
Part 3: The performance of the circuit (II)

- The information we will obtain:

Number of LUTs	
Number of bonded IOBs	
Which pin of the FPGA is the output 'zero' connected? (pin name)	
Where does the longest path start from	
Where does the longest path end	
How long is the longest path	
How much of the longest path is routing	
How many levels of logic is in the longest path	

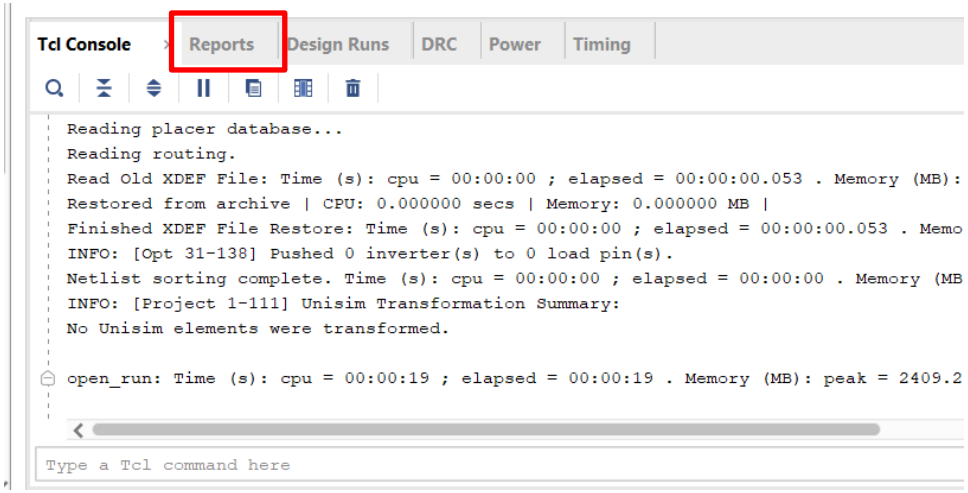
Part 3: Where to find the data

- You will find all data in the implementation reports.
- Take a look at the different reports to see all kinds of information on the implemented design.
- The most important information are stored in the route design and place design reports.



Part 3: Where to find the data

- You will find all data in the implementation reports.
- Take a look at the different reports to see all kinds of information on the implemented design.
- The most important information are stored in the route design and place design reports.



Last Words

- In lab 5, you will Implement an Arithmetic Logic Unit (ALU) in Verilog and evaluate its **speed and resource utilization.**
- Draw a block level diagram of the MIPS 32-bit ALU, based on the description in the textbook.
- Implement the ALU using Verilog.
- Synthesize the ALU and evaluate speed and FPGA resource utilization.
- In the report, you will use your adder from Lab 2 in the ALU and compare the resource utilization.

Report Deadline

[3. May 2024 23:59]

Digital Design & Computer Arch.

Lab 5 Supplement: Implementing an ALU

Frank K. Gürkaynak
Seyyedmohammad Sadrosadati
ETH Zürich

Spring 2024

[16. April 2024]