Carleton University

Department of Systems and Computer Engineering

SYSC 3006 (Computer Organization) Fall 2020

Lab / Assignment 2

Prerequisites

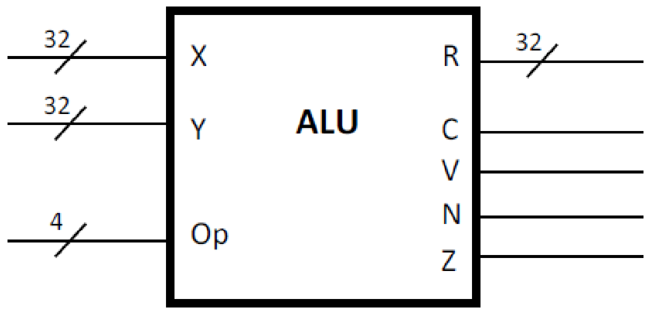
Lab / Assignment 1 and all materials on cuLearn related to Lecture 1 and 2; and especially materials related to Lecture 3.

Goal

- Design an ALU circuit similar to one that might be included in a processor.;
- Implement and test all designs using Logisim.

Introduction

The component-level (black box) abstraction of the ALU is shown here:



The ALU must satisfy the following block-level requirements. Additional design and implementation requirements are given later.

The ALU Interface

2 x 32-bit Inputs: X and Y	inputs
1 x 32-bit Output: R	output: R represents the result of applying an operation to X and Y
1 x 4-bit Operation Selector: Op	input: represents the operation to be applied to X and Y to produce R
4 x 1-bit Status Flags: C, V, N, Z	outputs: represent additional information about the result.

ALU Operations

The table below encodes the Operations performed by the ALU. Not all of the Op encodings have been used (just to keep the size of the lab down).   Note: *iff = if and only if*

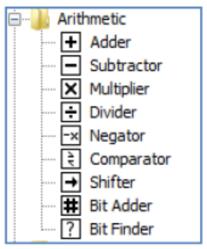
Operation	4-bit Op	Result	C	V	N	Z
NOP	0000	R= don't care	don't care	don't care	don't care	don't care
ADD	0001	R = X + Y	Carry from msb	1 iff signed overflow	1 iff R < 0 as signed value	1 iff R = 0
SUB	0010	R = X – Y	Borrow into msb	1 iff signed overflow	1 iff R < 0 as signed value	1 iff R = 0
RY	0011	R = Y	0	0	1 iff R < 0 as signed value	1 iff R = 0
NEG	1110	R = 2sComplement(Y)	1 iff the addition of 1 (in the 2's complement operation) generated a carry	1 iff signed overflow	1 iff R < 0 as signed value	1 iff R = 0
UMIN	1001	R=minimum(X,Y) unsigned values	0	0	1 iff R < 0 as signed value	1 iff R = 0
SMIN	1010	R=minimum(X,Y) signed values	0	0	1 iff R < 0 as signed value	1 iff R = 0

Background

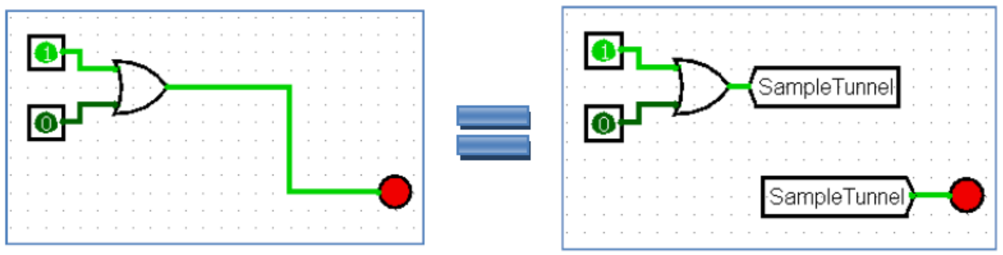
Review ELEC 2607 Lab 3 (Adder/Subtractor) to remind yourself about the conversion of decimal values to/from binary values, 2's complement representation of signed integers, sign extension, taking the 2's complement of a value, addition, carry and overflow. Some background information covered in SYSC 3006 class has been included at the end of this Lab description along with some self-study questions.

Logisim files setup and components information

Think about how each of Logisim's Arithmetic components might be helpful in the implementation of the ALU operations. You need to come up with design ideas for the implementation of each operation. Also think about how to design the *control circuitry* to integrate the individual operations into a single ALU circuit (HINT: the class lecture slides have a hint on this! ☺).



The ALU implementation may result in many wires crisscrossing the circuit. Use Logisim Tunnels (in the Wiring Library) to simplify the circuit diagram. From the Logisim documentation: "A tunnel acts like a wire that binds points together, but unlike a wire the connection is not explicitly drawn. This is helpful when you need to connect points far apart in the circuit and a network of wires would make the circuit ugly.” Always use appropriate names for the tunnels.



a) Direct wiring b) Tunnels

Your Assignment

All questions below are included in an editable .docx file (MS-Word), use it for your answers, then save it as a PDF file before submitting it.

Part I – Answer those questions [1-mark/3]

1-1

a) [0.25-mark] Why can a single binary value represents both a signed integer value and an unsigned integer value?

b) [0.25-mark] For each component supplied in Logisim’s Arithmetic Library, give a simple one-sentence summary (in your own words) of the function performed by the component.

c) [0.25-mark] Why are separate UMAX and SMAX operations included? Give an example in which the same input values (X and Y) would result in different output values (R) from the functions.

d) [0.25-mark] For each operation, propose test cases (i.e. X and Y input values) that can be used to test the ALU to demonstrate that it complies with the requirements of the operation given above. Each test case should expose different important aspects of the required functionality. There should be enough test cases to expose all important aspects of the required functionality. Describe each test in terms of any initial conditions, the inputs, the expected outputs, the particular aspects of the requirements that are tested, and why the test validates the aspects of the requirements being tested. Chose reasonable tests that give reasonable requirements coverage of both output values and flags, and reasonable descriptions.

Part II – Implementation of the ALU [2-mark/3]

Implement the required ALU. Use 32-bit registers for the inputs (X and Y). Use a 4-bit register as the operation selector (Op). Use a 32-bit probe to monitor the R output. Use individual 1-bit probes to monitor the Status Flag outputs.

2.1 - Circuit wiring [0.50-mark]

Insert a screenshot of your final ALU circuit here below and save a copy of your Logisim circuit (Save as) and name it “Lab2.circ” *(your “Lab2.circ” Logisim circuit must be included with your submission. We may need to be able to verify your circuit and rum few test cases in order to give you the marks for this part).* Briefly explain your circuit or some parts of your circuits as required

2.2 [1.5-mark]

Simulate and Log the test cases that you proposed in Part I above (unless you think that other test cases might be more appropriate ☺). Log the inputs and outputs of the circuit, including the flags (check Logisim documentation and the posted video about logging, they will give you an idea). Briefly comment your log to explain your results. If your solution is not complete before the deadline, try to log at least the working parts and give some explanations.

Note: If you have any question please ask them during your TA online live session. Do not wait till the deadline to ask questions as there in no warranty you will get answered before the deadline.

Good Luck