Faculty of Engineering and Applied Science
Memorial University of Newfoundland
ENGI 3861 - Digital Logic
Fall 2016

Laboratory 3:
Two-bit Arithmetic Logic Unit

Pre-lab
This is a long, complex lab. Hence, it is vital that each group thoroughly complete the pre-lab sections. The pre-lab will be checked by a TA at the start of the lab. If a group's pre-lab work is not complete and consequently the group is not able to complete the lab during the lab period, a severe mark penalty will be applied.

Lab Report
Each group is expected to complete a lab report. At the end of the lab, hand in the lab report, which should include the pre-lab, along with the lab observations and comments requested. Where indicated, demonstrate the correct operation of your circuit to a TA and have them sign your lab report.

1. Introduction

The purpose of this lab is to design, build, and test a 2-bit ALU that is capable of performing simple arithmetic and logic operations. The ALU is obtained using two major sub-units: a 2-bit adder and multiplexers (MUXes).

2. 2-bit Ripple Adder

A multi-bit adder can be built using several methods, each differing in the hardware complexity and the resulting speed of operation. The simplest (but slow) way to build a 2-bit adder is a ripple adder. A ripple adder cascades two 1-bit adders, or full adders, each of which can be built using 2-input XOR, AND and OR gates. For the adder that you will build in this lab, you can use any necessary combination of the following devices:

7400 Quad NAND gates	7402 Quad NOR gates	7404 Hex inverters
7408 Quad AND gates	7432 Quad OR gates	7486 Quad XOR gates.

[Prelab] Create a logic diagram of the 2-bit ripple adder. This diagram should include the logic gates, part numbers and pin numbers, for ease of construction and debugging. **Before the lab period**, build the circuit on the digital board by first constructing two independent full adders and testing them separately before linking them to create a 2-bit adder. (Be sure to leave room on the board for the remaining part of the ALU circuit.) If your circuit does not work, try to debug your circuit systematically by trying to isolate the problem by identifying sub-modules that work. By now you should be finding that the logic probe is a very useful debugging tool. Record the results of your testing by listing various cases of inputs and the resulting outputs of the adder. At the start of the lab, demonstrate the working 2-bit adder to a TA and have them sign your lab report. DO

NOT DISMANTLE the adder, as it is required for the subsequent section. When transporting your board with an assembled circuit, be careful not to knock wires loose.

3. 2-bit ALU

In this section, using the 2-bit adder and multiplexers, design a 2-bit ALU that has the following specifications:

- The 2-bit operand inputs are A and B and the carry-in is CIN.
- Three control inputs, S1, S2, and S3, decide the operation to be performed.
- The output consists of a 2-bit output, RESULT, and the carry-out, COUT.

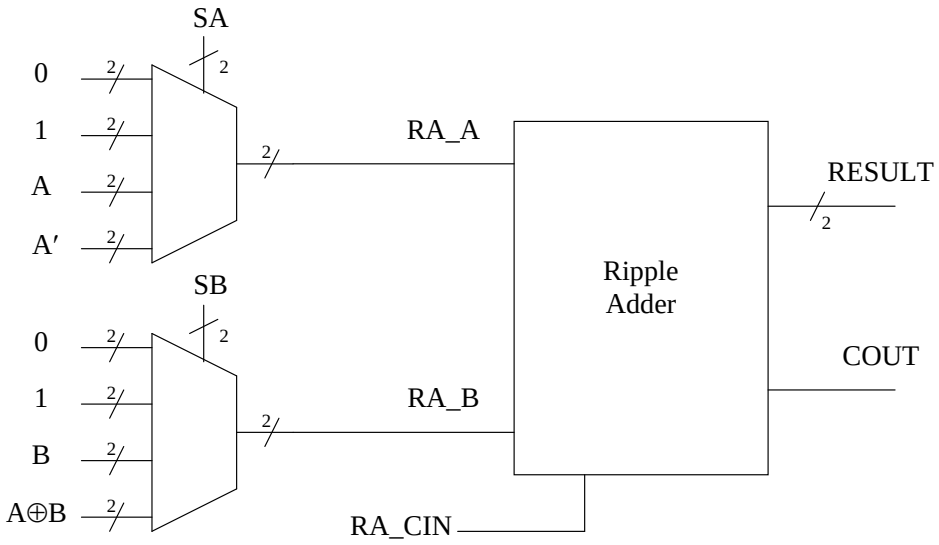
The ALU function is as follows:

S1	S2	S3	Operation	Result of operation
0	0	0	CLEAR	RESULT = 0, COUT = 0
0	0	1	LOADA (Load value of A)	RESULT = A, COUT = 0
0	1	0	LOADB (Load value of B)	RESULT = B, COUT = 0
0	1	1	CMPLA (One's complement of A)	RESULT = A′, COUT = 0
1	0	0	XORAB (Exclusive OR of A and B)	RESULT = A ⊕ B, COUT = 0
1	0	1	INCRA (Increment A)	RESULT = A + 1, COUT = carry from A+1
1	1	0	INCRB (Increment B)	RESULT = B + 1, COUT = carry from B+1
1	1	1	ADDAB (Add A with B including carry-in)	RESULT = A + B + CIN, COUT = carry from addition

[Prelab] Design the ALU and draw a logic diagram (including part and pin numbers) of the resulting circuit. If necessary, divide your diagram into a hierarchical structure. Your design documentation should contain block diagram(s), truth tables, Karnaugh maps, and a logic diagram which should include part and pin numbers. Your design should make use of multiplexers. You may use the suggested design below as a guideline for completing your design.

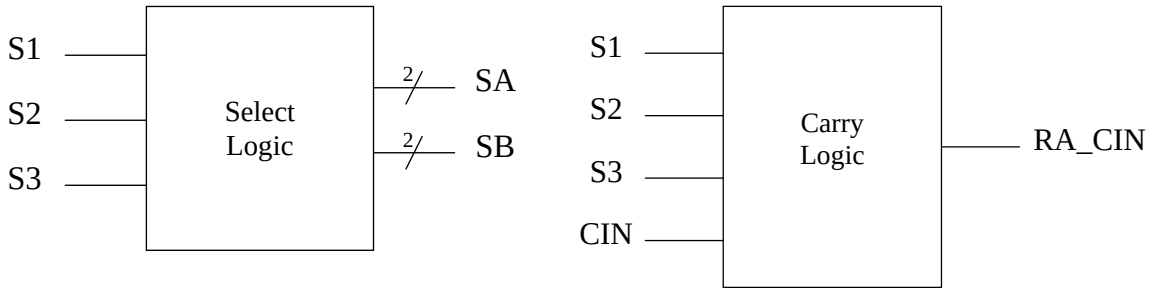
A suggested design:

One possible design is illustrated in the following block diagram:



If all five input signals to the adder are carefully selected for each operation (as specified by S1, S2, and S3), then the output of the adder (2-bit sum and carry-out) can be directly used as the final output RESULT and COUT. If this approach is taken, input A to the adder (RA_A) is 0, 1, A or A′ over all the 8 operations; similarly, input B to the adder (RA_B) is 0, 1, B or A ⊕ B, where "⊕" represents XOR. The CIN input to the ALU is a 1-bit input and, except for the ADDAB operation, is ignored. Part 74153 dual 4-to-1 muxes can be used to make the appropriate selection of inputs to the adder.

Signals SA, SB, and RA_CIN must be generated from the control inputs S1, S2, and S3 and CIN, as indicated by the following block diagrams:



Using this approach, the design can be completed by determining the select logic and the carry logic circuits.

Any design problem can be approached in different ways. The design approach described here is not the only one, nor is necessarily that which would produce the

simplest circuit. Outline another design approach using muxes that would give the correct functionality. For example, another design might also use 2-to-1 muxes.

How could your design be extended to a 32-bit ALU? Discuss how the speed of operation would change with an increase in the size of the ALU. Identify the slowest module in a 32-bit ALU and suggest ways of improving the speed.

[In the lab] Build and test your 2-bit ALU. Again, follow a systematic approach in building, testing and debugging of your circuit. You should consider testing a reasonably complete set of input scenarios to verify your circuit. In your lab report, be sure to identify which tests (i.e., values of inputs) you executed and the outcome of the tests. Demonstrate correct functioning of your circuit to a TA and have them sign your lab report.

4. Report Submission

Submit your lab report once your circuits have been verified by the TA. The lab report should include:

(a) a full description of the designs (including any block diagrams, truth tables, K-maps, logic diagrams, and appropriate comments)
(b) the details of the testing results
(c) a discussion of any problems encountered during the implementation in the lab and the methods used to overcome the problems