B38DB: Digital Design and Programming
ALU Examples

Mustafa Suphi Erden

Heriot-Watt University
School of Engineering & Physical Sciences
Electrical, Electronic and Computer Engineering
Room: EM 2.01
Phone: 0131-4514159
E-mail: m.s.erden@hw.ac.uk

Two ALU Examples from the Tutorial Document "detailed_solution_tutorial_04.pdf"

- Example 1:

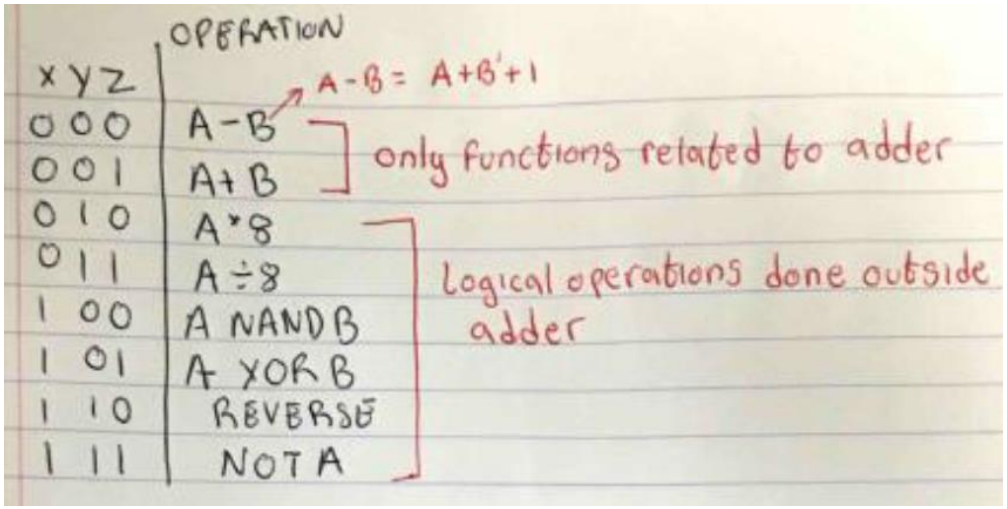
Design an ALU (Arithmetic and Logical Unit) with two 8-bit inputs A and B, and control signals x, y, and z. The ALU should support the operations described in the Table below. Use an 8-bit adder and an arithmetic/logic extender. Treat these as blocks with no internal structural details required.

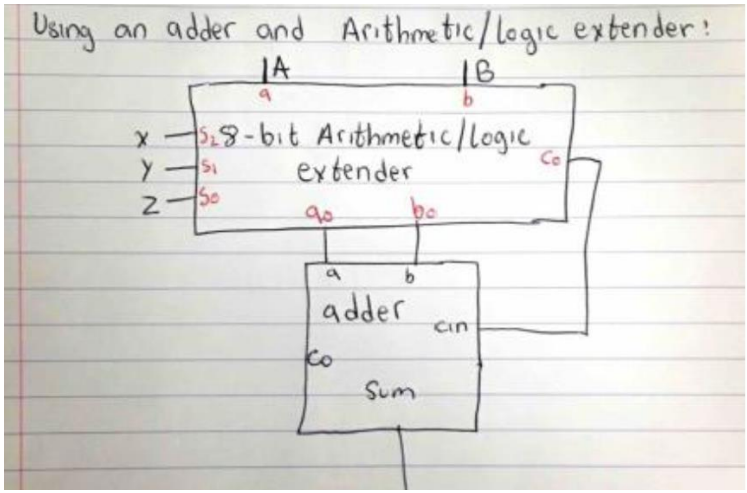
Inputs			Operation
x	y	z	
0	0	0	S = A - B
0	0	1	S = A + B
0	1	0	S = A * 8
0	1	1	S = A / 8
1	0	0	S = A NAND B (bitwise NAND)
1	0	1	S = A XOR B (bitwise XOR)
1	1	0	S = Reverse A (bit reversal)
1	1	1	S = NOT A (bitwise complement)

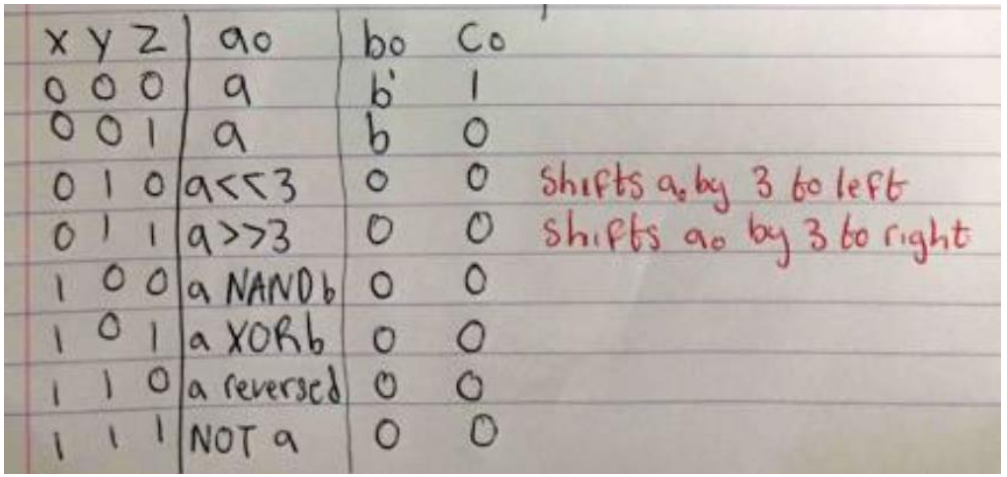
Two ALU Examples from the Tutorial Document "detailed_solution_tutorial_04.pdf"

- Solution 1:

Inputs			Operation
x	y	z	
0	0	0	S = A - B
0	0	1	S = A + B
0	1	0	S = A * 8
0	1	1	S = A / 8
1	0	0	S = A NAND B (bitwise NAND)
1	0	1	S = A XOR B (bitwise XOR)
1	1	0	S = Reverse A (bit reversal)
1	1	1	S = NOT A (bitwise complement)







Two ALU Examples from the Tutorial Document "detailed_solution_tutorial_04.pdf"

- Example 2:

Design a 3-bit Arithmetic Logic Unit (ALU) with two inputs (A, B), select signals (S1, S0), Output (O), and carry out (Cout). The ALU performs the following operations.

S1	S0	Operation
0	0	F = A – B
0	1	F = A + B
1	0	F = A – B – 1
1	1	F = – A + B – 1

Implement this ALU using a minimum number of 2-to-1 multiplexers, one single 3-bit adder (with a carry-in input), and a minimum number of aditional gates. Overflow bit is NOT necessary to be considered.

Two ALU Examples from the Tutorial Document "detailed_solution_tutorial_04.pdf"

- Solution 2:

S1	S0	Operation
0	0	F = A – B
0	1	F = A + B
1	0	F = A – B – 1
1	1	F = – A + B – 1