INTRODUCTION TO DIGITAL SYSTEMS

SYSC2310 (Fall 2020)
Dr. Mostafa Taha

Lab 4: Combinational Logic Circuits

Objectives

- Design a simple 4-bit ALU (Arithmetic Logic Unit)

Relation to course outcomes

The work in this lab is related to the two following course outcomes:

- Able to design, implement and analyze combinational logic with logic gates.

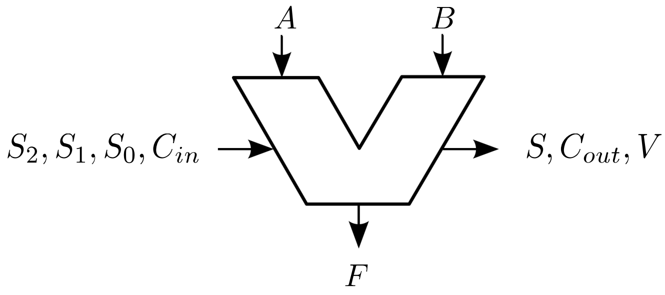
Preparation for the lab

1. Under your 'sysc2310lab/lab4' folder, create a folder 'Ex1' to save your work during this lab. Remember, by the end of the term, you are required to have the files for all the exercises in every lab. TAs and lab technicians will not be able to recover any deleted file, or any file that you have overwritten by mistake. Local Windows recycle bin doesn't work with the network M drive.

Exercise 1: Design a simple 4-bit ALU.

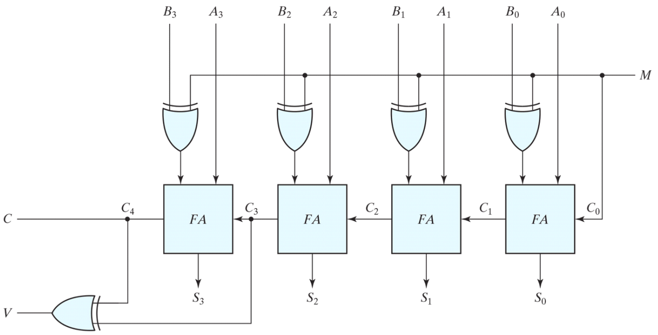
Design and implement the following 4-bit ALU. This ALU is capable of performing 4 logical operations and 8 arithmetic operations on two inputs (A and B), each of 4-bits.

The input operands are 4-bit each: $[A_3, A_2, A_1, A_0]$ and $[B_3, B_2, B_1, B_0]$.
The operations performed by the ALU are controlled by three selection inputs: S_2, S_1, S_0 (representing opcodes of the ALU) and an input carry C_{in} (representing part of the status register). The output result is also 4-bits $[F_3, F_2, F_1, F_0]$, in addition to a sign bit (S), an output carry C_{out}, and an overflow indicator (V) (representing other bits of the status register). The required function table is as follows:



S_2	S_1	S_0	C_{in}	The output F	
0	0	0	X	A AND B	Logic Functions
0	0	1	X	A OR B	
0	1	0	X	A XOR B	
0	1	1	X	A XNOR B	
1	0	0	1	$A+1$	Arithmetic Function
1	0	1	0	$A+B$	
1	1	0	1	$A-B$	
1	1	1	1	$B-A$	

- Here, S_2 works as a mode input: $S_2 = 0$ implements a logic function, $S_2 = 1$ implements an arithmetic function. The output result (F) will be displayed on 4 LEDs and 7-segment display. S_2 should be used to activate only the 4 LEDs on logical operations ($S_2 = 0$) or activate only the 7-segment display on arithmetic operations ($S_2 = 1$). In addition, there will be 3 LEDs to represent the sign (S), the output carry (C_{out}) and the overflow indicator (V).
- Note that you are working with a 4-bit representation for 2's complement numbers, the maximum positive number is +7 and the most negative number is –8. Hence, a single 7-segment display can be used to show the magnitude of the number. An independent LED, the sign (S), will be used for indicating the sign for arithmetic operations.
- You have multiple design choices. For the design of the logic control unit, you can use a combinational circuit or a MUX with gates. Write the truth table, derive the K-map and draw the minimum gate implementation. In case you use a MUX, implement the circuit for the MUX.
- For the design of the arithmetic unit you can make use of previously designed modules, such as your full adder from Lab#1, the overflow detection circuit from Fig 4.13 in the Mano and Ciletti Book (attached below), and the BCD to 7-Segment Decoder from Lab#3.



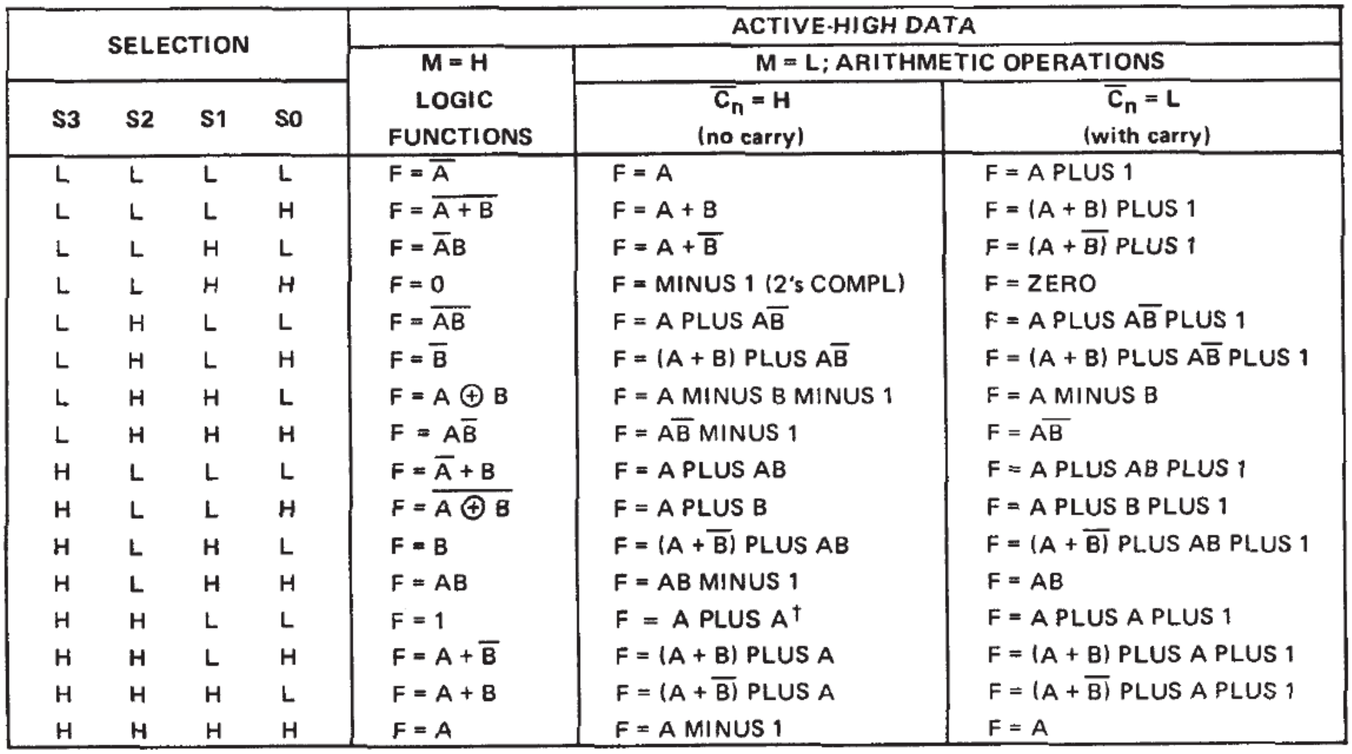
Four-bit adder–subtractor with overflow detection.

- Compute the total number of gates, assuming that all the gates have the same implementation area.
- Compute the maximum delay of the overall ALU, assuming that all the gates have the same delay.

- Implement the circuit in Logisim.
- Enable logging to 'Lab4_Ex1_log.txt', and use the poke tool (👆) to change the values of the input bits. Notice output LEDs and 7-segment.
- Save the circuit as 'Lab4_Ex1.circ'.

Exercise 2:

The circuit implemented in Exercise 1 represents a simple ALU with limited capabilities. A circuit similar to this one was introduced in late 1960s and made a huge impact on how we design computers of the 1970s era. Explore the datasheet of the 74181 ALU in cuLearn. This simple ALU is capable of performing the following operations:



SELECTION				ACTIVE-HIGH DATA		
				M = H	M = L; ARITHMETIC OPERATIONS	
S3	S2	S1	S0	LOGIC FUNCTIONS	$\overline{C_n}$ = H (no carry)	$\overline{C_n}$ = L (with carry)
L	L	L	L	$F = \overline{A}$	F = A	F = A PLUS 1
L	L	L	H	$F = \overline{A + B}$	F = A + B	F = (A + B) PLUS 1
L	L	H	L	$F = \overline{A}B$	$F = A + \overline{B}$	$F = (A + \overline{B})$ PLUS 1
L	L	H	H	F = 0	F = MINUS 1 (2's COMPL)	F = ZERO
L	H	L	L	$F = \overline{AB}$	F = A PLUS $A\overline{B}$	F = A PLUS $A\overline{B}$ PLUS 1
L	H	L	H	$F = \overline{B}$	F = (A + B) PLUS $A\overline{B}$	F = (A + B) PLUS $A\overline{B}$ PLUS 1
L	H	H	L	$F = A \oplus B$	F = A MINUS B MINUS 1	F = A MINUS B
L	H	H	H	$F = A\overline{B}$	$F = A\overline{B}$ MINUS 1	$F = A\overline{B}$
H	L	L	L	$F = \overline{A} + B$	F = A PLUS AB	F = A PLUS AB PLUS 1
H	L	L	H	$F = \overline{A \oplus B}$	F = A PLUS B	F = A PLUS B PLUS 1
H	L	H	L	F = B	$F = (A + \overline{B})$ PLUS AB	$F = (A + \overline{B})$ PLUS AB PLUS 1
H	L	H	H	F = AB	F = AB MINUS 1	F = AB
H	H	L	L	F = 1	F = A PLUS A†	F = A PLUS A PLUS 1
H	H	L	H	$F = A + \overline{B}$	F = (A + B) PLUS A	F = (A + B) PLUS A PLUS 1
H	H	H	L	F = A + B	$F = (A + \overline{B})$ PLUS A	$F = (A + \overline{B})$ PLUS A PLUS 1
H	H	H	H	F = A	F = A MINUS 1	F = A

Note that: + is an OR function, AB is A AND B, A APLUS B is a mathematical addition.

- On the attached logic diagram of the 74181 ALU, find the logic output of each gate in the design in the following cases:

1. $[S_3S_2S_1S_0] = [LLLL]$, $M = L$, thus arithmetic operation and $\overline{C_n} = L$, therefor, performing $F = A$ plus 1, for A=[1001] and A=[1100]. Verify correctness of the output.
2. $[S_3S_2S_1S_0] = [HLLH]$, $M = L$, thus arithmetic operation and $\overline{C_n} = H$, therefor, performing $F = A$ plus B, for A=[1001] and B=[1100]. Verify correctness of the output.
3. $[S_3S_2S_1S_0] = [HLLL]$, $M = H$, thus logical operation and $\overline{C_n} = H$, therefor, performing $F = \overline{A} + B$, for A=[1001] and B=[1100]. Verify correctness of the output.
4. Find the total number of gates in the design of 74181 ALU, assuming that all the gates have an equivalent area.
5. Find the maximum delay of the 74181 ALU, assuming that all the gates have an equivalent delay.

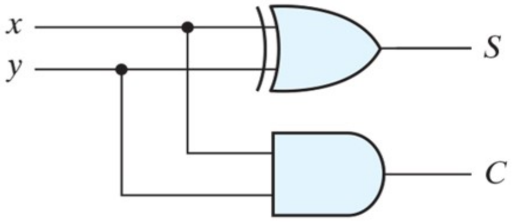
Exercise 3 [Bonus]:

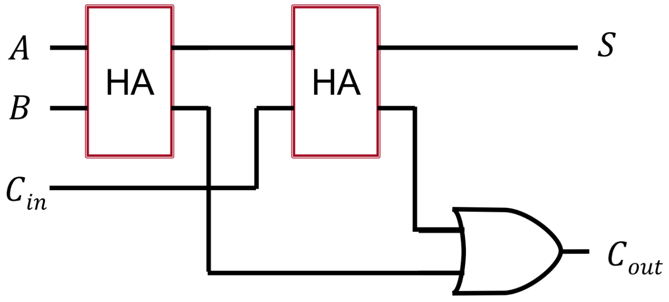
Describe the circuit proposed in Exercise 1 using only text (letters, numbers and symbols, but not drawings). The description should be detailed enough to enable any other student, the TA or even a computer software to replicate your design without errors. All answers are good as long as they provide a full description.

For instance, the following Full Adder can be written as:

Circuit Name: Half-Adder
Input: 1-bit x, 1-bit y
Output: 1-bit S, 1-bit C
Function:
S = x XOR y
C = x AND y

Circuit Name: Full-Adder
Input: 1-bit A, 1-bit B, 1-bit Cin
Output: 1-bit S, 1-bit Cout
Uses: the Half-Adder circuit defined above.
Function:
[M1, M2] = Half-Adder (A, B),
[S, M3] = Half-Adder (M1, Cin),
Cout = M2 XOR M3





Demonstration:

- In order to get full credit, complete all exercises and be prepared to answer questions asked by the TA during the one-on-one meeting.
- Before you leave, even if you could not complete the lab, make sure you sign-out with the TA.
- If you could not complete the lab during the scheduled lab time, finish up the lab on your own time. In this case, the demonstration points will be reduced in accordance to the percentage of work completed.
- If the student is absent or does not show the TA the work completed, 0 points are attributed for this part.