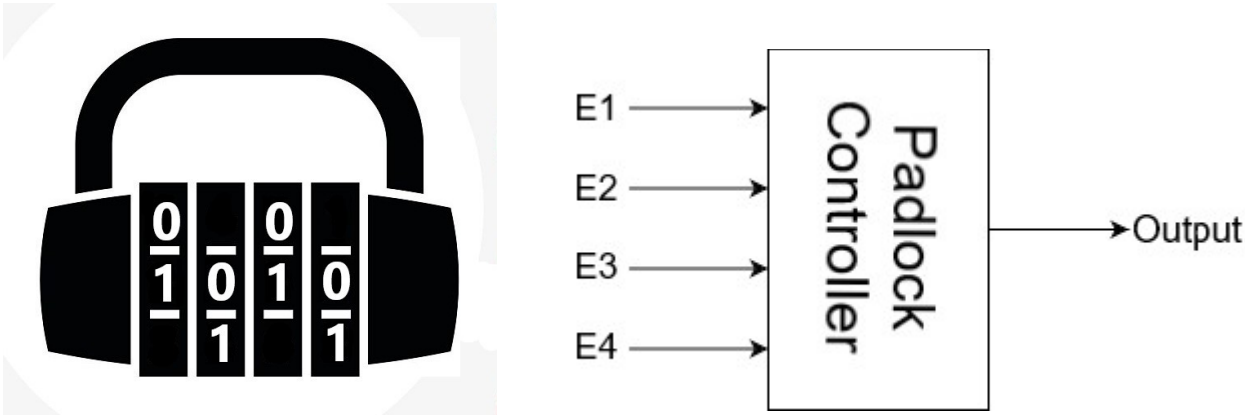
Sétif 1 University
Faculty of Sciences
Common Core Mathematics and Computer Science

2nd semester (year 2023/24)
Machine Structures 2 course
Kara Abdelaziz professor

Problem set 1
(Combinational Circuits)

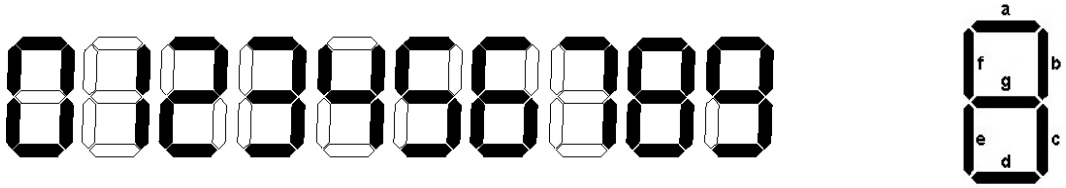
Exercise 01 :

Using the 5-step method, design the combinational circuit that control the opening and closing of an electronic lock (padlock). The lock keypad has 4 binary inputs, as shown below. The lock opens mechanically if the circuit outputs the value 1, otherwise it remains closed. The opening code is 1101.



Exercise 02 :

The combinational circuit of a 7-segment display is a circuit that controls a 7-segment display, it allows the display of a single digit in 7 segments as illustrated in the diagram. It receives as input a number encoded on 4 bits enclosed in the interval [0,9], and as output it produces the combination of segments which displays the digit like a decimal system upon the 7-segment display.



Using the 5-step method, construct the combinational circuit that controls the display on the 7-segment display, choosing for each segment the most reductive method possible, between undefined outputs and disjunctive canonical form or conjunctive canonical form.

Exercise 03 :

Implement the following digital circuits using the 5-step method :

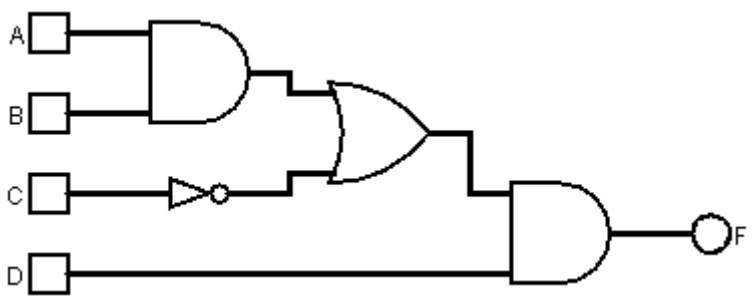
Circuits	Properties
Multiplexer*	2 inputs, 1 selector and 1 output (*n*=1)
Demultiplexer	1 input, 2 selectors and 4 outputs (*n*=2)
Encoder	4 inputs and 2 outputs (*n*=2)
Decoder*	3 inputs and 8 outputs (*n*=3)
Priority controller	4 inputs and 4 outputs (*n*=4)
Parity controller	4 bits input and 1 output
Adder	2 unsigned 2-bit integers input and 1 unsigned 2-bit integer output, plus carry output
1's-Compliment	1 signed 3-bit integer input and 1 signed 3-bit integer output
Shifter*	Left-Shifter, with a 2-bit unsigned integer input and a 2-bit shift *Amount*. And a 2-bit unsigned integer as output
Comparator	2 unsigned 2-bit integers as input and 3 outputs for *greater*, *less* and *equal*
Sign Extender	1 *2's complement* 4-bit integer as inputand 1 *2's complement* 8-bit integer as output

(*) These circuits are obligatory to do in TDs, however the rest is optional, depending on the progress.

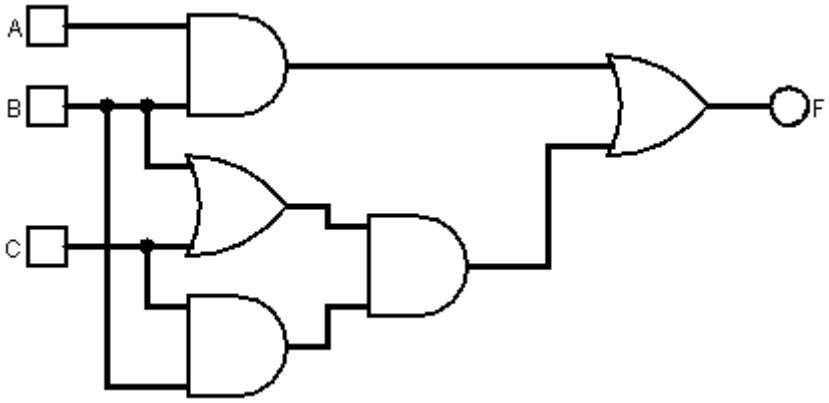
Exersice 04 :

1) Find the equivalent universal gate AND and OR of the following logic gates : AND, OR, NOT, NAND, NOR, XOR, XNOR.

2) Transform the following circuit into NOR gates by replacing each gate with its equivalent. Then with using the bubble pushing method.



3) Transform the following circuit into NAND gates by gate replacement method and by bubble pushing method.

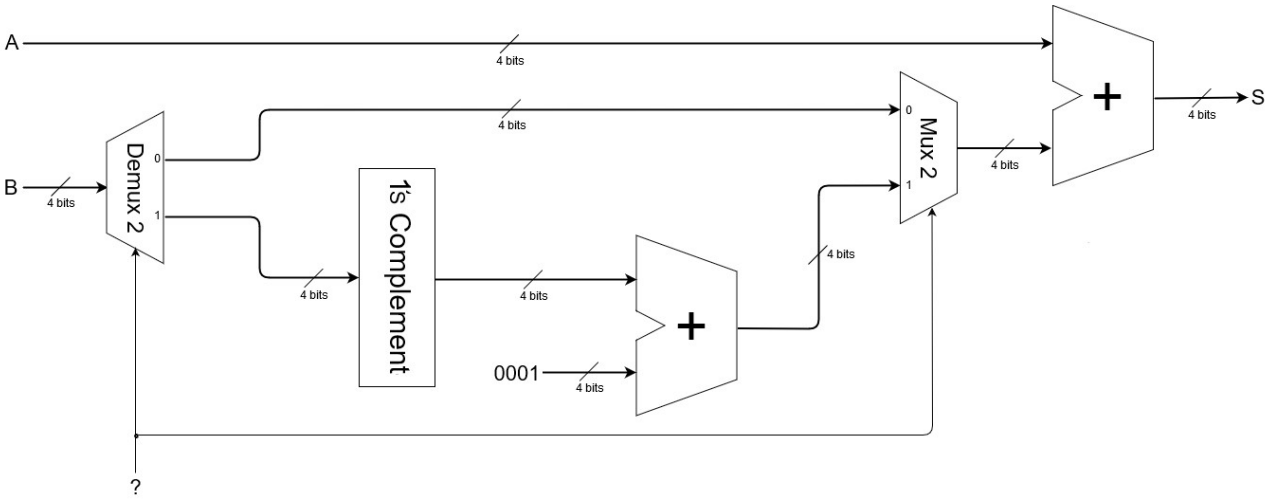


4) Give a conclusion after using both methods.

Exercise 05 :

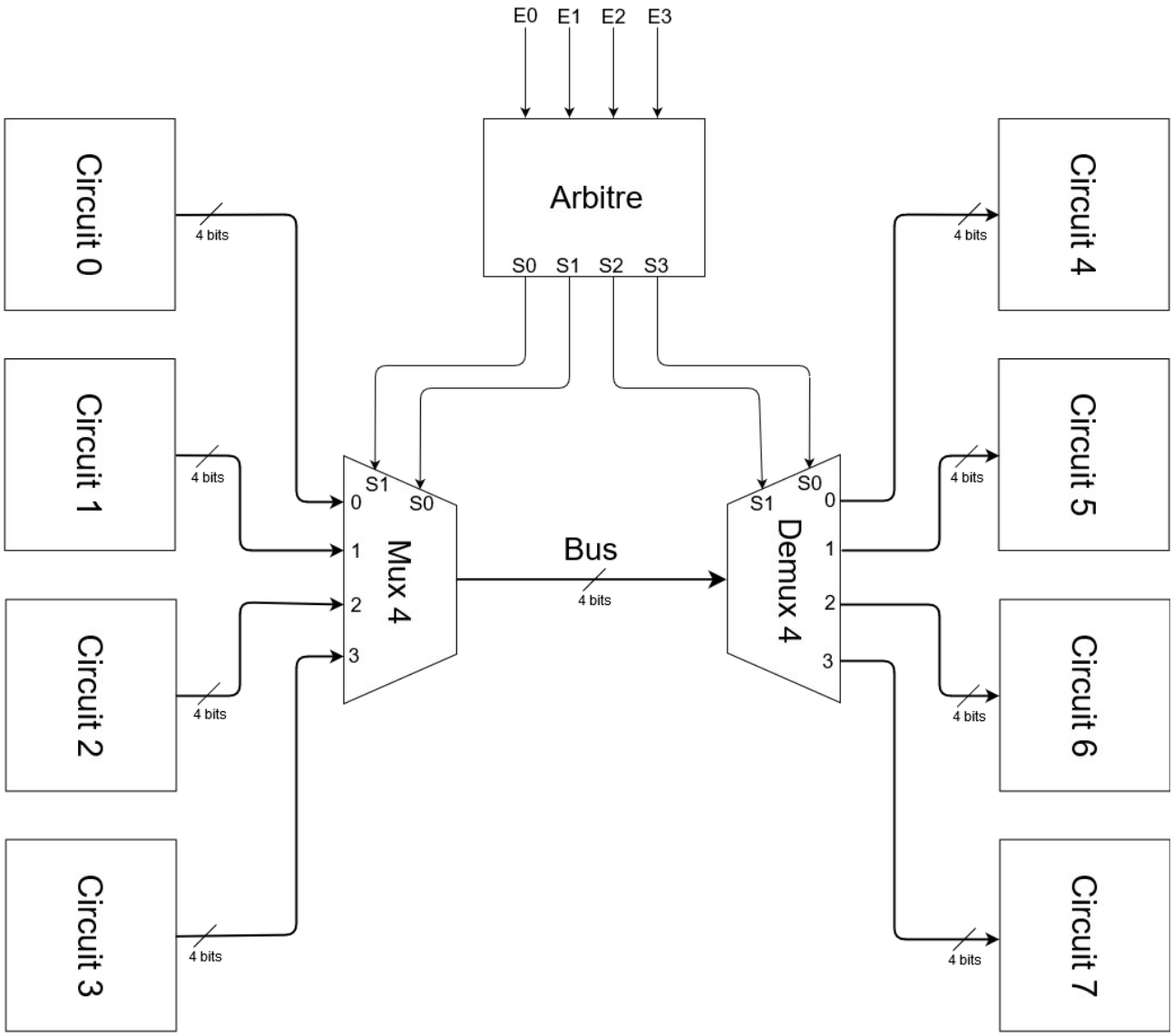
1) Use the following 3 methods for building a Multiplexer :
   - Building a 2-Input Multiplexer using the 5-step method.
   - Building a 2-input Multiplexer using Tristate Buffer gates.
   - Building an 8-input multiplexer from 2-input multiplexers using *slicing* technic.

2) Describe the functionning of the following circuit (the Adder in the diagram does the *2's Complement/Unsigned* addition) :



   - What does the circuit do ?
   - Give a name for the 3rd input (in ?) at the bottom.
   - Replace the 1's Complement circuit with a simpler equivalent circuit.
   - Draw a Global scheme for the circuit.

3) Construct the combinational circuit of the Bus Arbitrator in such a way that the 2 inputs E0 and E1 represent an encoding of the starting circuits (the left circuits 0-3). And the 2 inputs E2 and E3 represent the encoding of the outcoming circuits (right circuits 4-7)

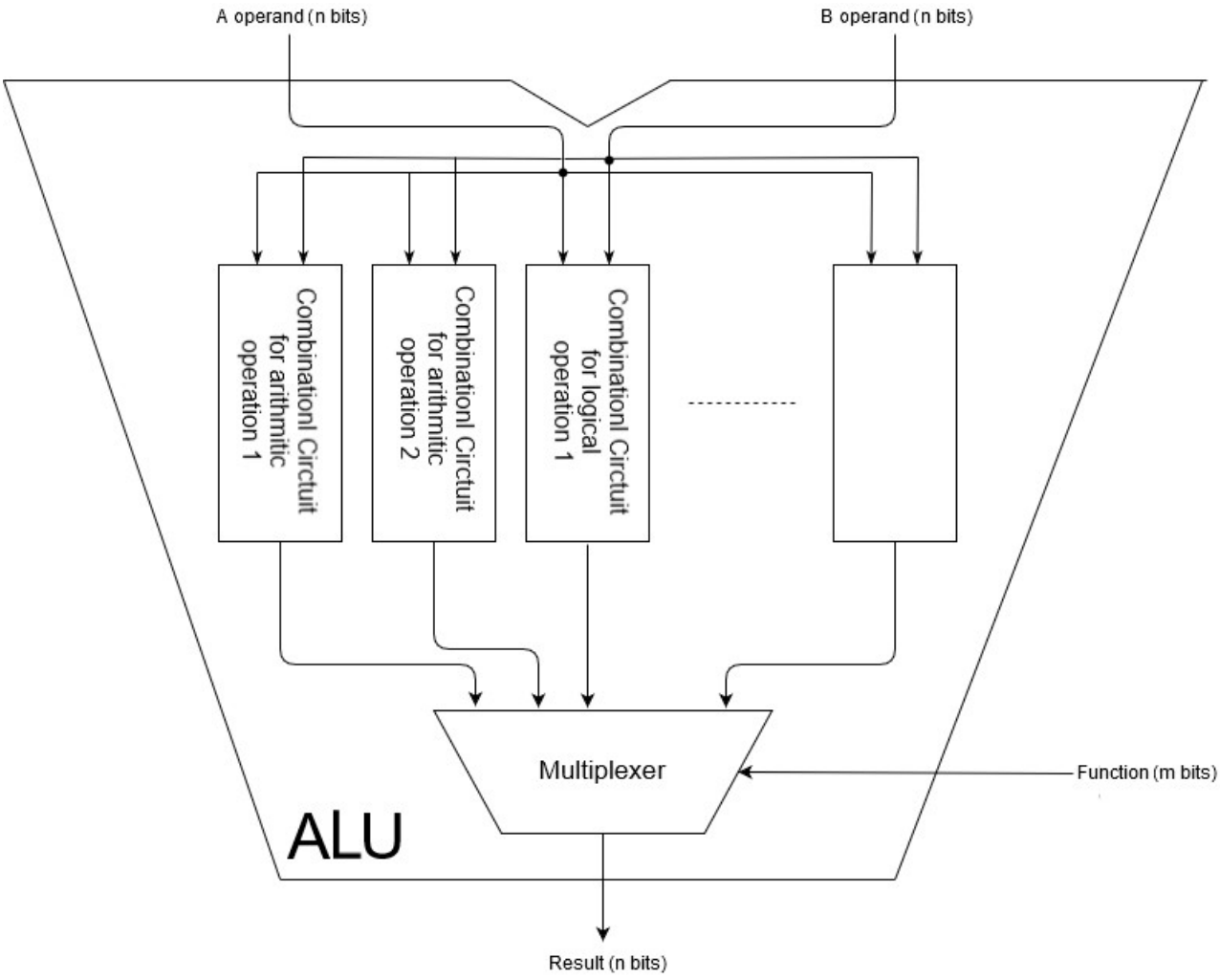


4) Build a microarchitecture in which data transmissions are done using an 8-bit Z-state bus. The microarchitecture must contain 4 circuits that can all communicate in pairs in both directions. The Bus is regulated by a Bus Arbiter controling the circuit outputs using Tristate Buffers, and the inputs using the *enable* pins. Many circuits do not support the Z value as inputs when the Bus is inactive, transform the Bus in a way the Z value is replaced with a 0-*weak*.

Secondly, implement the Arbitrator combinational circuit with the 4 inputs. For the first 3 inputs, each combination represents a circuit pair, for example 000 represents the Circuit1-Circuit2 pair, the combination 001 is the Circuit1-Circuit3 pair, and so on until covering all possible pairs. The last entry corresponds to the direction of transmission. For example 0 for a transmission from Circuit1 to Circuit2, and 1 inversely from Circuit2 to Circuit1.

Exercice 06 :

The ALU is one of the well-known combinational circuits used in microarchitectures. Theoretically, the construction of an ALU is based on the model referenced in the figure below, in which the UAL is presented in the form of several combinatorial circuits each operating out a unique arithmetic or logical operation. The number of these circuits is 2^m. All circuits receive the 2 operands A and B on *n* bits, and do all their calculations in parallel and return their results to the Multiplexer. A single result is chosen by the Multiplexer among the 2^m proposed. The choice of the Multiplexer to let the result of a specific circuit pass, is taken in relation to the binary number encoded in *m* bits provided on the *Functions* input of the Multiplexer. A table endoding all the operation is often supplied with the ALU to specify the *m*-bit code for each operation.



1) Create the ALU that performs the 2's complement operations described in the table below, such that *n* = 8 bits and *m* = 3, therefore 8 functions.

Note 1: This ALU is real, it was used to build the Mic-2 academic processor.

Function	F2	F1	F0	ALU result
0	0	0	0	A
1	0	0	1	B
2	0	1	0	B+1
3	0	1	1	A+B
4	1	0	0	A-B
5	1	0	1	not A
6	1	1	0	A and B
7	1	1	1	A or B

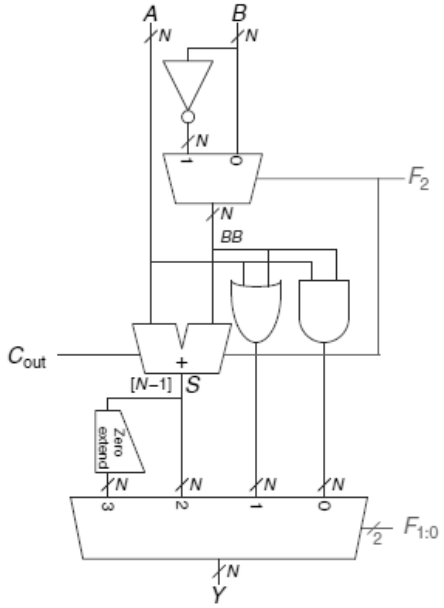
Note 2: The logical operations of the ALU *not*, *and* and *or* are bitwise operations, and not word by word (a word is 8 bits long).

2) To make the implementation of the ALU more efficient, perform optimization on operations B+1, A+B and A-B in order to minimize the number of gates in the circuit.

3) The UAL shown below was taken directly from the book *Digital Design and Computer Architecture* (page 249). It will be used in chapter 7 of the book to create a MIPS-like processor. Explain how the ALU works.

Table 5.1 **ALU operations**

$F_{2:0}$	Function
000	A AND B
001	A OR B
010	A + B
011	not used
100	A AND $\overline{B}$
101	A OR $\overline{B}$
110	A – B
111	SLT



4)** Build the combinational circuit that does Multiplication and Division on 4 bits in Unsigned encoding. You can refer to the book *Digital Design and Computer Architecture* on page 252.

** Difficult question, to do if you have enough time or to do at home.