Chapter 4: Combinational Logic Systems

Institute of Applied Sciences and Technology

Dr. ASSABAA Mohamed

1.0

09/09/2023

Table des matières

Objectifs

At the end of this course, the learner will be able to:

• Know combinational logic circuits.

• Understand the operating principles of combinational logic circuits.

• Study some iterative logic systems (Half-adder and full-adder, Half-subtractor and full-subtractor).

• Study the main combinational logic circuits used in digital systems (such as encoders, decoders, multiplexers, ...).

• Examine logical functions using combinational circuits.

Prerequisites:

• Boolean Algebra

• Numeration Systems and Number Coding

Introduction

A combinational circuit is a circuit in which the outputs depend only on the combination of input states at the moment of observation.

The problem of combinational logic is the study of automatisms in which the set of output variables depends only on the present state of the set of input variables, also we call this kind of system a system without memory.

The objective of combinational calculation is the realization of a system with the required characteristics, using a minimum number of elements (contacts, relays, switches, logic gates, etc.). Therefore, it can be stated that combinational synthesis is the translation of a logical function from a specification into a diagram. Various synthesis methods are possible; they differ in the form of the function used (canonical or simplified), on the type of operators or integrated circuits chosen, and the employed functional decomposition technique.

To design a combinational logic system, the focus is on using a minimal number of logic circuits, following these design steps:

1. Define the input variables and output variables.

2. Establish the operation table or truth table according to the problem.

3. Simplify the output functions.

4. Create the logic diagram using the minimum number of circuits.

The search for a solution can be approached in various ways, depending on whether the combinations of input variables appear in any order or in an order imposed by the system's operation.

I Problem in which the combinations of input variables appear in any order

From the statement of the problem, we address the truth table which gives us the value of the output for each combination of the input variables. The latter are classified according to the binary code. The expression of the output is directly deduced from the truth table, where the output function(s) is (are) expressed in the first canonical form. The next step involves simplifying the result before creating the logic diagram.

Example :

Design a combinational logic system that performs addition (or subtraction) of two bits, x and y, and the carry (or borrow) R, based on the state of a control signal M:

• M=0, the circuit performs addition.

• M=1, the circuit performs subtraction.

The outputs of the system are:

• The sum or difference defined by S

• The carry or borrow defined by R1

Solution :

Truth table:

M	x	y	R	S	R_1	
0	0	0	0	0	0	
0	0	0	1	1	0	
0	0	1	0	1	0	
0	0	1	1	0	1	Addition
0	1	0	0	1	0	
0	1	0	1	0	1	
0	1	1	0	0	1	
0	1	1	1	1	1	
1	0	0	0	0	0	
1	0	0	1	1	1	
1	0	1	0	1	1	
1	0	1	1	0	1	Subtraction
1	1	0	0	1	0	
1	1	0	1	0	0	
1	1	1	0	0	0	
1	1	1	1	1	1	

Simplification of the output function expressions:

YR \ MX	00	01	11	10
00	0	1	1	0
01	1	0	0	1
11	0	1	1	0
10	1	0	0	1

$$S = x\bar{y}\bar{R} + \bar{x}\bar{y}R + xyR + \bar{x}y\bar{R}$$
$$= x(\bar{y}\bar{R} + yR) + \bar{x}(\bar{y}R + y\bar{R})$$
$$= x(\overline{y \oplus R}) + \bar{x}(y \oplus R)$$
$$\Rightarrow S = x \oplus y \oplus R$$

YR \ MX	00	01	11	10
00	0	0	0	0
01	0	1	0	1
11	1	1	1	1
10	0	1	0	1

$$R_1 = yR + \overline{M}xR + \overline{M}xy + M\bar{x}y + M\bar{x}R$$
$$= yR + (y + R)(\overline{M}x + M\bar{x})$$
$$\Rightarrow S = yR + (y + R)(M \oplus x)$$

II Problem in which combinations of input variables appear in an order imposed by system operation

For this type of problems, the truth table is replaced by an operation table in which the input values are classified in the natural order of their occurrence during the system's operation. This table is then completed with the results of all possible maneuvers, including normal, abnormal, or impossible.

Example :

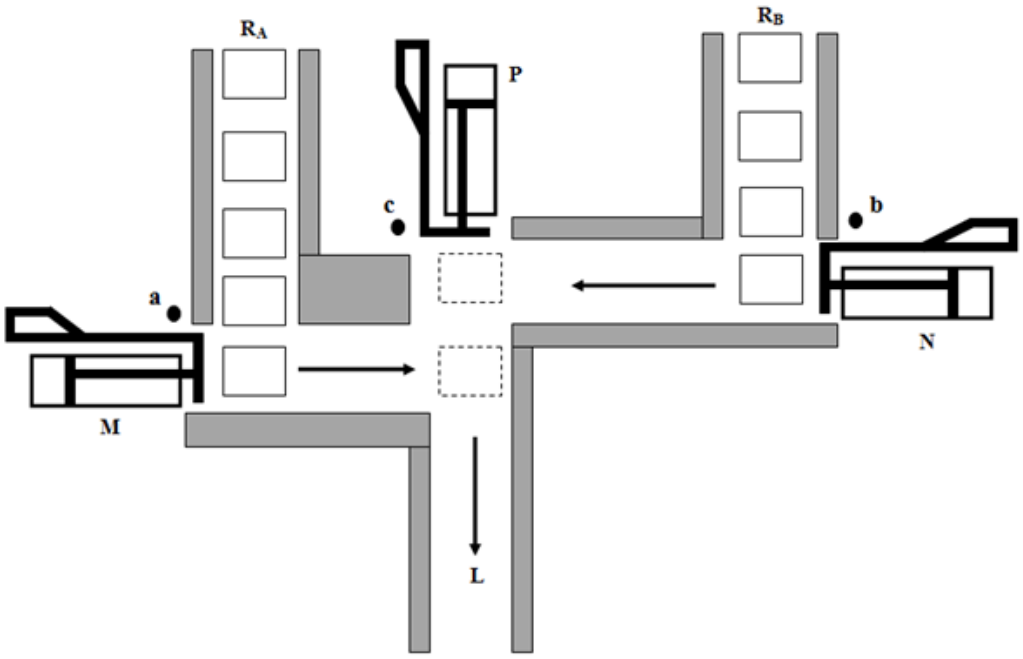
Parts of model A and B are stored in reservoirs RA and RB, respectively; must arrive grouped on a line L. To achieve this, three relays, M, N, and P, are used. Relays M and N respectively bring parts A and B in front of the cylinder P, which then pushes the assembly on the line L. Contacts a, b, and c are actuated at the end of the stroke of each cylinder (as shown in the device below).

The operating cycle is as follows:

Closing of a switch I.

1. Advance of M
2. Advance of N
3. Retraction of M
4. Advance of P
5. Retraction of N
6. Retraction of P; and the cycle begins again.

Design such a system.



Solution :

Operation table:

I	a	b	c	M	N	P
0	0	0	0	0	0	0
1	0	0	0	1	0	0
1	1	0	0	1	1	0
1	1	1	0	0	1	0
1	0	1	0	0	1	1
1	0	1	1	0	0	1
1	0	0	1	0	0	0
1	0	0	0	1	0	0

The other maneuvers of the input variables are abnormal or impossible maneuvers. Simplification of the output function expressions:

Ia / bc	00	01	11	10
00	0	Ø	1	1
01	Ø	Ø	Ø	0
11	Ø	Ø	Ø	0
10	Ø	Ø	0	0

$$M = I\bar{b}.\bar{c}$$

bc \ la	00	01	11	10
00	0	Ø	1	0
01	Ø	Ø	Ø	0
11	Ø	Ø	Ø	0
10	Ø	Ø	1	1

$$N = a + b\bar{c}$$

bc \ la	00	01	11	10
00	0	Ø	0	0
01	Ø	Ø	Ø	0
11	Ø	Ø	Ø	1
10	Ø	Ø	0	1

$$N = \bar{a}b$$

III Iterative logical systems

In some cases, a logical system can be decomposed into a set of identical subsystems arranged in a cascade. The inputs to each subsystem, also called "cell," are formed by the outputs of the previous subsystem (except for the first subsystem) and external inputs. The output of the last cell represents the output of the complete system. The problem therefore amounts to designing a cell, then generalizing to the complete system.

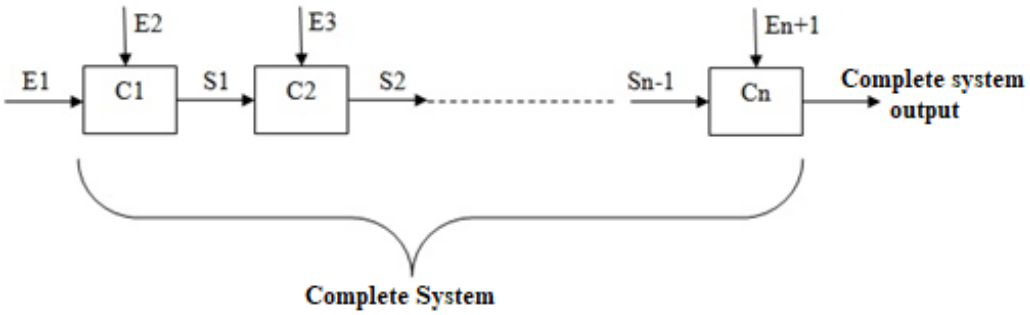


Figure 1. General scheme of iterative logical systems

Iterative design offers an advantage, especially for problems with a large number of variables, but it does not necessarily lead to a simpler model.

1. Some iterative systems

Définition : Half adder and full adder

Half adder :

It is a circuit which makes the sum of two bits b_0 and b_1 of the same weight without taking into account the previous carry. The truth table for this circuit is:

b_0	b_1	S_0	R'_0
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S_0 = \bar{b}_0 b_1 + b_0 \bar{b}_1 = b_0 \oplus b_1$$
$$R'_0 = b_0 b_1$$

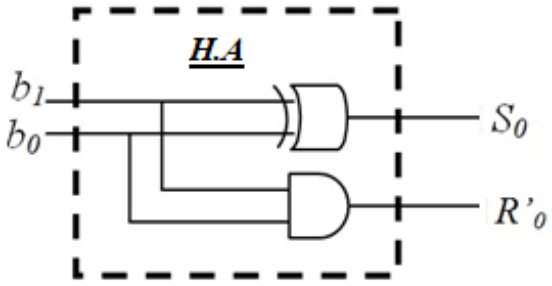


Figure 2. Half Adder Logic Diagram

Full adder :

A full adder has 3 inputs: the two bits to be added, b_0 and b_1, and the carry of the previous stage, R'_0.

It has 2 outputs: the sum S_1 and the outgoing carry R_1.

Its truth table is as follows:

b_0	b_1	R_0	S_1	R_1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S_1 = \bar{b}_0\bar{b}_1R_0 + \bar{b}_0b_1\bar{R}_0 + b_0\bar{b}_1\bar{R}_0 + b_0b_1R_0$$
$$S_1 = R_0(\bar{b}_0\bar{b}_1 + b_0b_1) + \bar{R}_0(\bar{b}_0b_1 + b_0\bar{b}_1)$$
$$S_1 = R_0\overline{(b_0 \oplus b_1)} + \bar{R}_0(b_0 \oplus b_1)$$
$$S_1 = R_0 \oplus (b_0 \oplus b_1)$$
$$S_1 = R_0 \oplus S_0$$

$$R_1 = \bar{b}_0b_1R_0 + b_0\bar{b}_1R_0 + b_0b_1\bar{R}_0 + b_0b_1R_0$$
$$R_1 = R_0(\bar{b}_0b_1 + b_0\bar{b}_1) + b_0b_1(\bar{R}_0 + R_0)$$
$$R_1 = R_0(b_0 \oplus b_1) + b_0b_1$$
$$R_1 = R'_0 + R_0S_0$$

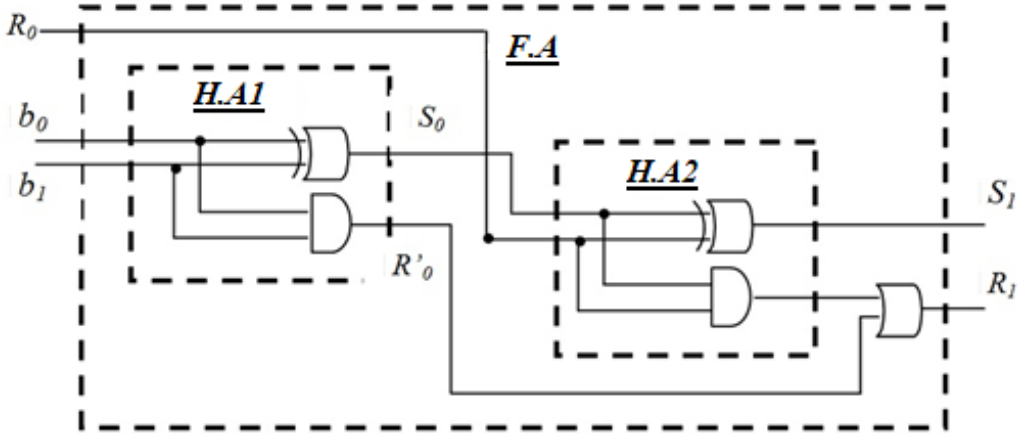


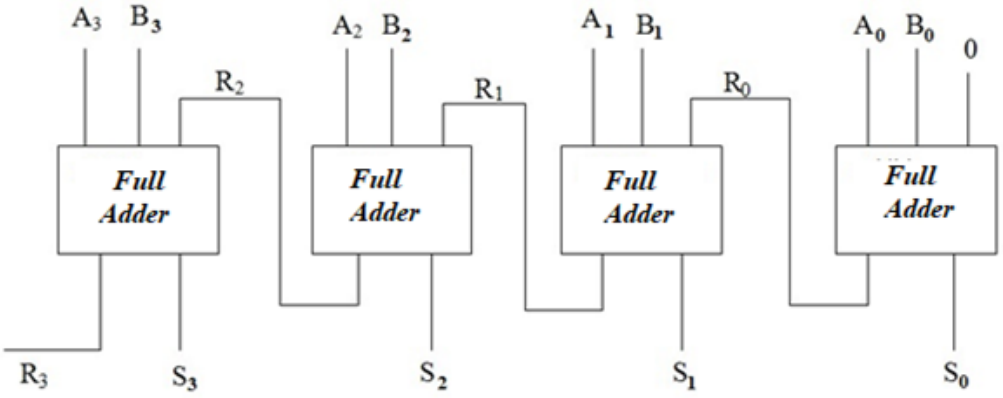
Figure 3. Full Adder Logic Diagram

The full adder thus represents of two half adders wired in cascade.

The addition of two n-bit words requires n full adders. The carry propagates from the least significant binary elements to the most significant binary elements.

Example :

Make a logic diagram that performs the binary addition of two four-bit numbers, A and B.



This architecture is interesting from a hardware point of view because it is repetitive. However, the obtained result depends on the number of full adders and thus the size of the words to be added. The carry R, is delivered after the first addition, and so on.

Définition : Half subtractor and full subtractor

Half Subtractor :

It is a circuit that performs the subtraction of two bits, b0 and b1, of the same weight, without considering a potential borrow from the bits of lower weight. The truth table of this circuit is as follows:

b_0	b_1	D_0	R'_0
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$D_0 = \bar{b}_0 b_1 + b_0 \bar{b}_1 = b_0 \oplus b_1$$

$$R'_0 = \bar{b}_0 b_1$$

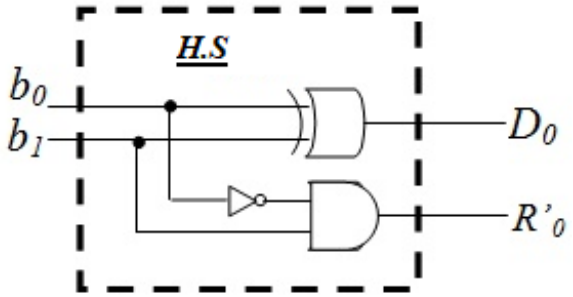


Figure 4. Logic diagram of a half-subtractor

Full Subtractor :

It is a circuit that performs the subtraction of two bits, b_0 and b_1, of the same weight, plus the carry from the previous stage R'_0.

b_0	b_1	R_0	D_1	R_1
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$D_1 = \bar{b}_0 \bar{b}_1 R_0 + \bar{b}_0 b_1 \bar{R}_0 + b_0 \bar{b}_1 \bar{R}_0 + b_0 b_1 R_0$$

$$D_1 = R_0 (\bar{b}_0 \bar{b}_1 + b_0 b_1) + \bar{R}_0 (\bar{b}_0 b_1 + b_0 \bar{b}_1)$$

$$D_1 = R_0 \overline{(b_0 \oplus b_1)} + \bar{R}_0 (b_0 \oplus b_1)$$

$$D_1 = R_0 \oplus (b_0 \oplus b_1)$$

$$D_1 = R_0 \oplus S_0$$

$$R_1 = \bar{b}_0 \bar{b}_1 R_0 + \bar{b}_0 b_1 \bar{R}_0 + \bar{b}_0 b_1 R_0 + b_0 b_1 R_0$$

$$R_1 = R_0 (\bar{b}_0 \bar{b}_1 + b_0 \bar{b}_1) + \bar{b}_0 b_1 (\bar{R}_0 + R_0)$$

$$R_1 = R_0 \overline{(b_0 \oplus b_1)} + \bar{b}_0 b_1$$

$$R_1 = R'_0 + R_0 \bar{D}_0$$

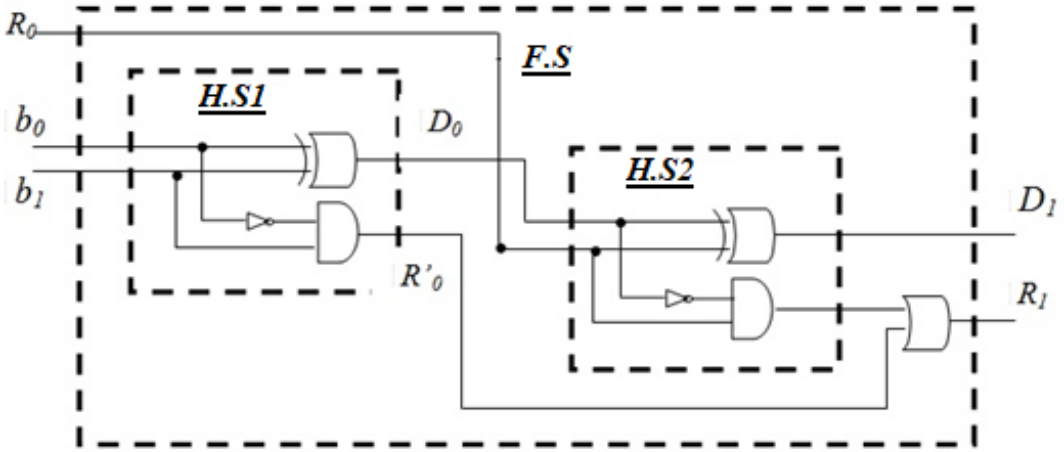


Figure 5. Logic Diagram of a Full Subtractor

IV Control circuits

1. Encoder

It is a circuit with 2^n inputs and n outputs, used for encoding a decimal number into binary or BCD. The decimal number arrives at the input of the encoder (only one input is active at a time). When one of the input lines is activated, the encoder produces an n-bit output word corresponding to the encoding of the information identified by the activated line. For instance, an 8-to-3 encoder will have 8 inputs and 3 address lines as outputs.

E	S_0	S_1	S_2
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

The following figure presents the general form of an encoder.

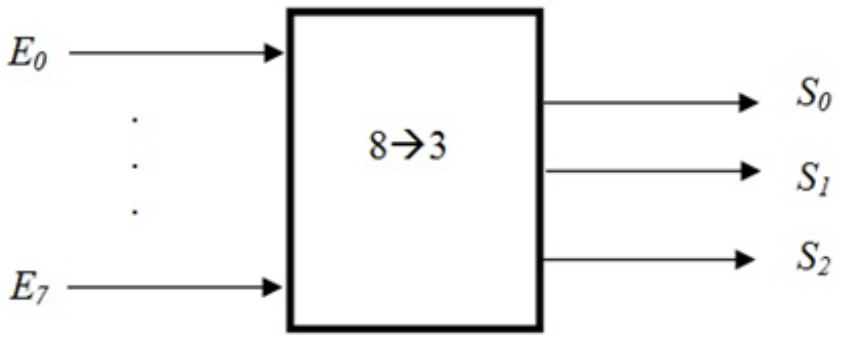


Figure 6. General form of an encoder

2. Decoder

The decoder performs the inverse function of the encoder. It is a logic circuit with n inputs and 2^n outputs. When the enable signal is active, only the output corresponding to the binary value displayed at the input becomes active. All other outputs remain inactive. For example, a 2→4 decoder has 2 inputs designated as A and B, and 4 outputs S0, S1, S2, and S3, one for each of the 4 possible combinations of inputs.

A	B	S_0	S_1	S_2	S_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

For a decoder operating in positive logic, the expressions of the outputs as a function of the inputs are:

$$S_0 = \overline{A}\,\overline{B},\ S_1 = \overline{A}B,\ S_2 = A\overline{B},\ S_3 = AB$$

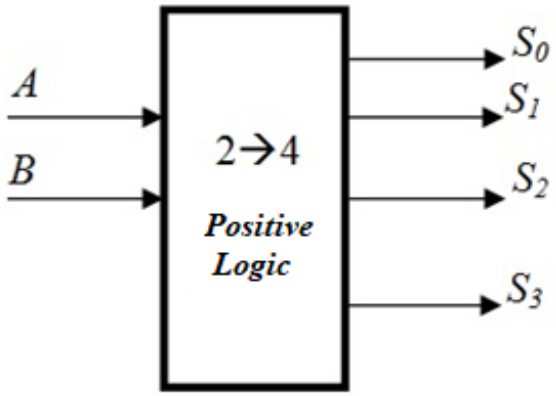


Figure 7. General form of a decoder

3. Multiplexer

The multiplexer (MUX) is a combinational system designed to select one out of 2^n inputs and transmit it to a single output. The selection is performed using n control lines. For example, a 4-input multiplexer (E_i, i=0,..,3) has 2 control lines (A, B) and one output S. The operation of such a circuit is defined by the following truth table:

A	B	S
0	0	E_0
0	1	E_1
1	0	E_2
1	1	E_3

$$S = \overline{A}\,\overline{B}E_0 + \overline{A}BE_1 + A\overline{B}E_2 + ABE_3$$

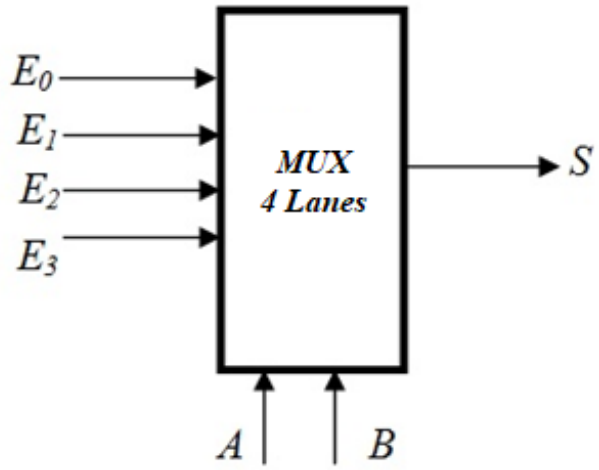


Figure 8. General form of a multiplexer

4. Demultiplexer

The demultiplexer (DEMUX) performs the inverse function of a multiplexer: it routes a single input E to one out of 2^n output paths (S0, S1, ...). The transfer of the input to one of the outputs is done using n control signals (A, B, ...), and the outputs are mutually exclusive – it operates like a switch. The following truth table shows more clearly the operation of a 4-way demultiplexer:

A	B	S_0	S_1	S_2	S_3
0	0	E	0	0	0
0	1	0	E	0	0
1	0	0	0	E	0
1	1	0	0	0	E

The expressions of the outputs as a function of the inputs are then defined as follows:

$$S_0 = \overline{A}\,\overline{B}E,\ S_1 = \overline{A}BE,\ S_2 = A\overline{B}E,\ S_3 = ABE$$

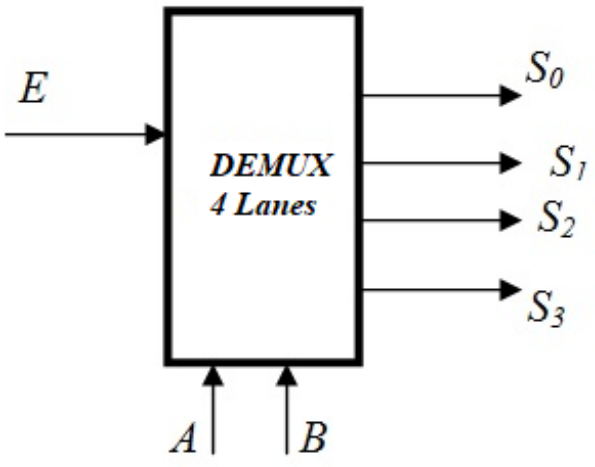


Figure 9. General form of a demultiplexe

V Exercises :

1. Exercise 01 :

Two binary numbers A and B are each represented using two bits: A1 A0 for A and B1 B0 for B.

Provide a synthesis of a combinational system designed to determine whether: A=B; A<B, or A>B.

a) Solve this problem considering the system as a 4-input combinational system with 3 outputs S1, S2, and S3.

b) Solve the same problem by determining S1, S2, and S3 from the result of the subtraction: A-B.

2. Exercise 02:

A reservoir is supplied by two valves, V1 and V2.

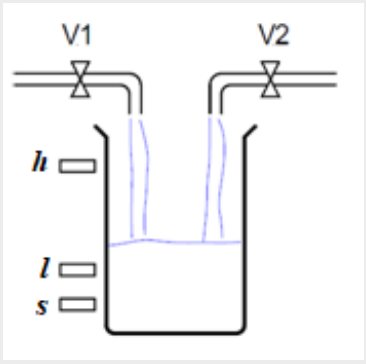
There are three levels: Safety (s), Low (l), and High (h).

When the level is below (s), both valves V1 and V2 are opened.

When the level is between (s) and (l), only valve V1 is opened.

When the level is between (l) and (h), only valve V2 is opened.

When the level is detected by (h), both valves are closed.



QUESTION :

Determine the logical equations for opening the two valves, V1 and V2, according to the level detected by the 3 sensors (s), (b), and (h).

Note: When the level is detected by a sensor, its state is = 1.

3. Exercise 03 :

A beverage dispenser allows delivering to the consumer:

- Water

- Grenadine mixed with water

- Mint mixed with water

But it must not allow obtaining:

- Pure grenadine

- Pure mint

- Both grenadine and mint.

The front panel of the dispenser has 3 buttons:

- W (water)

- G (grenadine)

- M (mint)

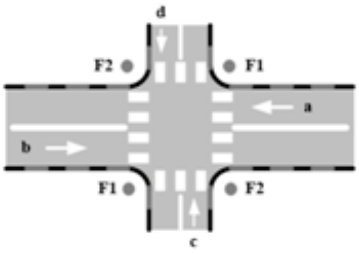
1) Determine the equation for controlling the opening of the distributor valve.

2) Deduce the corresponding logic diagram.

4. Exercise 04 :

We are interested in an intersection between a main road and a secondary road, for which we gives the operational principle as well as an extract from the functional specifications.

Presence sensors for cars are placed along lanes ***a*** and ***b*** for the main road, and lanes ***c*** and ***d*** for the secondary road. The outputs of these sensors are set to ***1*** when cars are detected.



• Traffic light ***F1*** is green when there are cars in ***a*** and ***b*** at the same time.

• Traffic light ***F1*** is green when there are cars on either lane ***a*** or ***b*** and there are no cars on lanes ***c*** or ***d***.

• Traffic light ***F2*** is green when there are cars on both lanes ***c*** and ***d***, and there are no cars on lanes ***a*** or ***b***.

• Traffic light ***F2*** is green when there are cars on either lane ***c*** or ***d***, and there are neither in ***a*** nor in ***b***.

• Traffic light ***F1*** is green when there are no cars at all.

The variable corresponding to a traffic light is set to ***1*** when the light is green.

1. Create the truth table that allows translating this problem, with inputs being ***a***, ***b***, ***c***, and ***d***, and outputs being ***F1*** and ***F2***.

2. Determine the logical equation expressions for ***F1*** and ***F2*** in the form of sum of products.

3. Graphically simplify the expressions of ***F1*** and ***F2***.

5. Exercise 05:

Using an appropriate decoder and logic gates, perform the following logic function:

$$F = \overline{A}\,B\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + AB\overline{C} + ABC$$

6. Exercise 06:

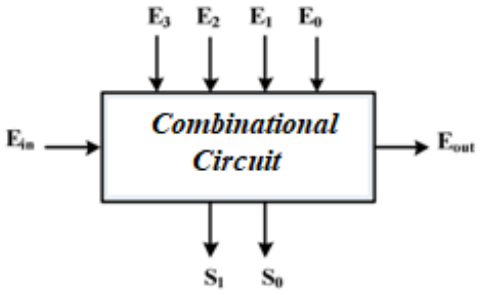
4 to 2 Encoder

It is a circuit that has 4 inputs and 2 outputs. For each activated input, its binary code is displayed on the outputs.

Create the truth table for this circuit and deduce the simplified logical equations for the different outputs.

7. Exercise 07 :

Suppose a combinational circuit with 5 input lines and 3 output lines, as shown in the figure below:



The operation is as follows:

• When only one input line among ***E0***, ***E1***, ***E2***, ***E3*** is at a high level, its number is encoded in binary on the outputs (***S1S0***).

• If multiple lines are simultaneously at a high level, the highest number is encoded.

• If all input lines are at a low level, the code (***S1S0***) = (***00***), but it is indicated by ***Eout=1*** that this code is not valid. In all other cases, ***Eout=0***.

• The operation described so far is observed when ***Ein=1***. If ***Ein=0***, then ***S1=S0=Eout=0***.

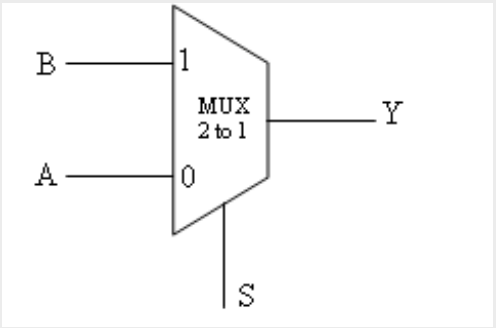
1. Create the truth table for this encoder.

2. Establish the logical equations of the outputs ***S0***, ***S1***, and ***Eout*** in terms of the inputs ***E0***, ***E1***, ***E2***, ***E3***, and ***Ein***.

3. Illustrate the logic diagram of the encoder.

8. Exercise 08 :

The 2 to 1 multiplexer function, whose logic symbol is shown in the figure opposite, corresponds to the following definition:



Definition : If the control signal ***S*** is worth ***0*** then the multiplexer transmits the input signal ***A*** to output ***Y***, otherwise the multiplexer transmits the input signal ***B*** to output ***Y***.

1. After establishing the Truth Table and then the Karnaugh map of this Multiplexer function, provide the expression of the output ***Y*** in terms of the inputs ***A*** and ***B***, and the control signal ***S***:

a) By performing a synthesis on the ***1*** (expression in the sum of products form),

b) By performing a synthesis on the ***0*** (expression in the product of sums form),

c) Verify the equality of the 2 expressions.

2. Establish the logic diagram of the 2 to 1 Multiplexer.

9. Exercise 09 :

Let the function F(A,B,C,D)=∑(0,2,5,7,11,13,14,15)

Perform this function using a mutiplexer

10. Exercise 10 :

We give the following function:

$$F(a,b,c) = \bar{a}bc + a\bar{c} + abc$$

Perform this function using a single mutiplexer with 3 selection inputs.

11. Exercise 11:

We want to create a parity detector (P) of 4 bits (a, b, c, and d). This detector should indicate the odd number of 1 (i. e., P = 1 if the number of 1 in the inputs is odd).

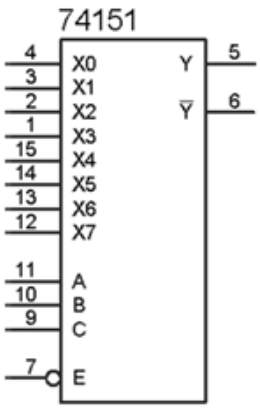
1. Give the truth table for this detector.

2. Give the equation for P.

3. Perform the equation for P using the 74151 multiplexer (see appendix).

Operation table of 74151

INPUTS				OUTPUTS	
SELECT			ENABLE	Y	$\overline{Y}$
C	B	A	$\overline{E}$		
x	x	x	H	L	H
L	L	L	L	X_0	$\overline{X}_0$
L	L	H	L	X_1	$\overline{X}_1$
L	H	L	L	X_2	$\overline{X}_2$
L	H	H	L	X_3	$\overline{X}_3$
H	L	L	L	X_4	$\overline{X}_4$
H	L	H	L	X_5	$\overline{X}_5$
H	H	L	L	X_6	$\overline{X}_6$
H	H	H	L	X_7	$\overline{X}_7$



H : high level L : low level x : Not definied

$$Y = \bar{E}(\bar{C}\bar{B}\bar{A}X_0 + \bar{C}\bar{B}AX_1 + \bar{C}B\bar{A}X_2 + \bar{C}BAX_3 + C\bar{B}\bar{A}X_4 + C\bar{B}AX_5 + CB\bar{A}X_6 + CBAX_7)$$

Bibliographie

J.C., Buisson Concevoir son microprocesseur, structure des systèmes logiques ; Collection Technosup 2006

S. Tisserant., Architecture et Technologie des Ordinateurs ; ESIL, 2003

M. Sbaï., Electronique Numérique, Logique combinatoire et composants numériques ; Technosup, 2013.

N. Mansouri., Les Systèmes Logiques", Tome 1 ; EUMC

R. Strandh et I. Durand., Architecture de l'ordinateur ; Dunod, 2005

E. Messerli, Y. Meyer., Electronique Numérique, Systèmes combinatoires, Tome 1; Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud, Septembre 2010