UNIVERSIDAD DE GRANADA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE TELECOMUNICACIÓN
GRADO EN INGENIERÍA INFORMÁTICA

# Tecnología y organización de computadores
# Prácticas

Autor : Alejandro Jerónimo Fuentes
Curso : 1ºC

# 1. Práctica 1: Análisis y diseño de circuitos combinacionales con Puertas lógicas

## 1.1. Análisis de un sistema combinacional

La primera parte de la práctica consiste en analizar los circuitos que aparecen en la figura mediante su implementación en el simulador lógico. Una vez implementado el circuito obtenemos experimentalmente las tablas de verdad de las funciones resultantes en la simulación.

| x | y | Valor |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Tabla 1: Tabla de verdad de la función f(x,y)

Obtenemos por tanto la expresión algebraica $f(x, y) = \bar{x}y + \bar{y}x = x \oplus y$
Realizamos lo mismo para la función g.

| x | y | Valor |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Tabla 2: Tabla de verdad de la función g(x,y)

Tenemos la misma expresión que la función f, por tanto tenemos
$g(x, y) = \bar{x}y + \bar{y}x = x \oplus y$

## 1.2. Diseño de un sistema combinacional

El objetivo es el diseño de un circuito lógico combinacional partiendo del enunciado de un problema, modelando el mismo mediante funciones de conmutación y realizando sus tablas de verdad, simplificación e implementación de diferentes formas equivalentes entre si y con la misma funcionalidad. El enunciado del problema es el siguiente:

*Un jurado consta de cuatro miembros que deben evaluar el examen de un candidato. El candidato aprobará el examen si y sólo si recibe dos o más votos favorables del jurado. Para votar, los miembros del jurado disponen cada uno de ellos de un interruptor (A, B, C ó D) de manera tal que pulsándolo (interruptor = 1) dan su voto favorable al candidato y no pulsándolo (interruptor = 0) dan su voto negativo al candidato.*

Se debe implementar un circuito lógico mínimo que genere la función que permita determinar si aprueba o suspende un candidato tomando como entradas los cuatro pulsadores A, B, C y D de que dispone el tribunal. Para ello, realizamos la tabla de verdad de la función y sacamos su expresión algebraica.

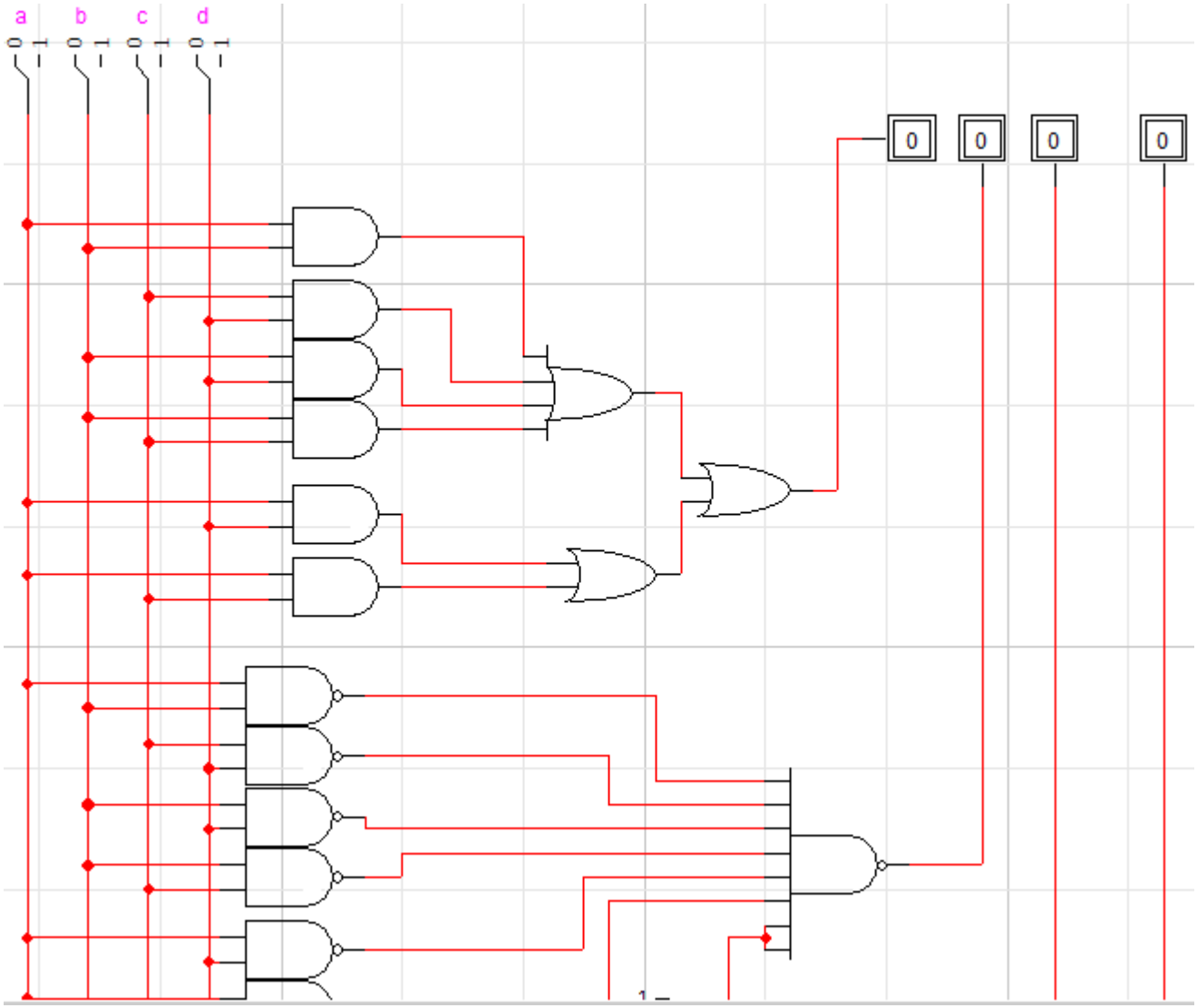
$f(A, B, C, D) = \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}BC\bar{D} + \bar{A}BCD + A\bar{B}\bar{C}D + A\bar{B}C\bar{D} + A\bar{B}CD + AB\bar{C}\bar{D} + AB\bar{C}D + ABC\bar{D} + ABCD$

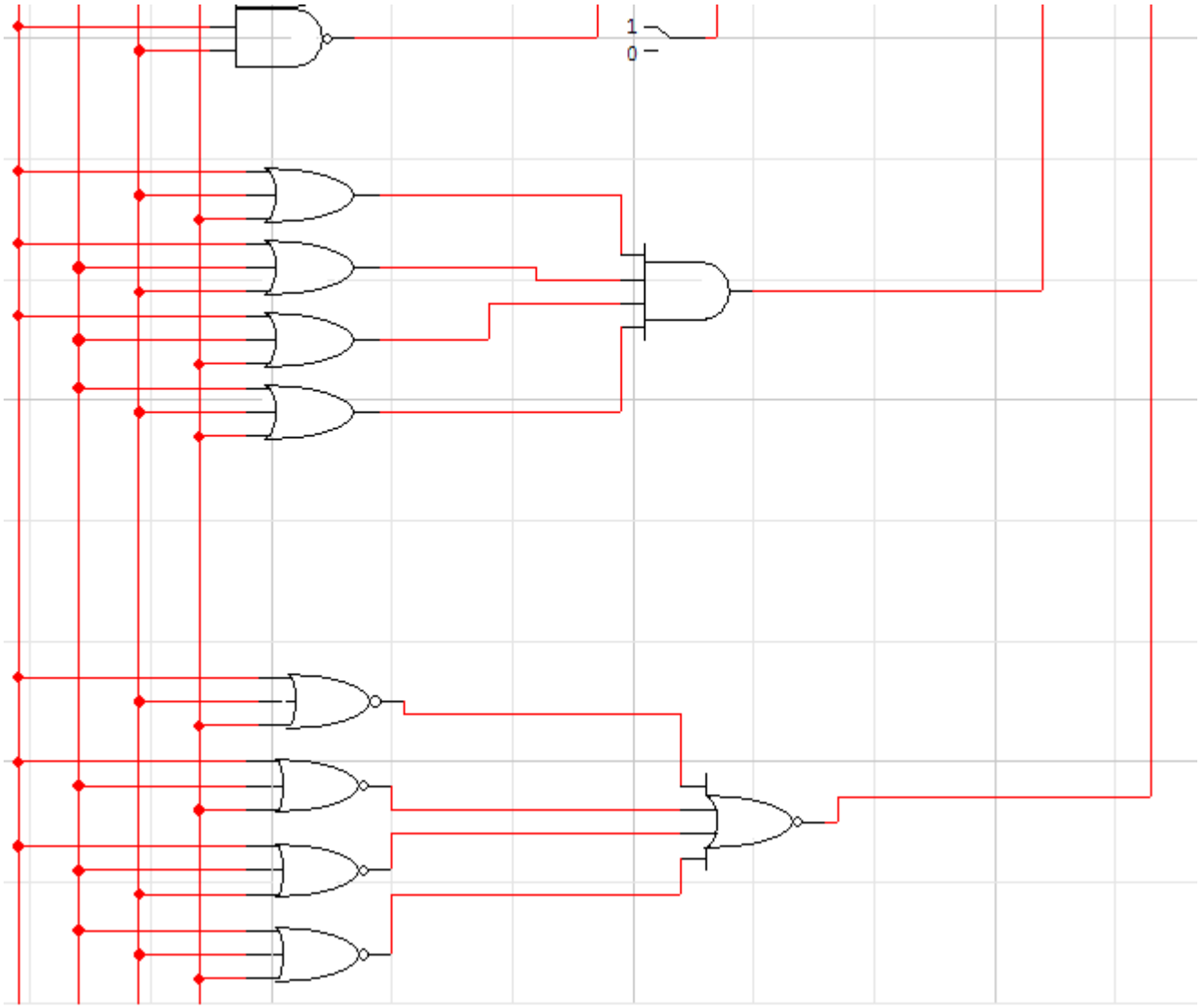
Simplificamos la función usando el Mapa de Karnaugh.

| AB/CD | ¬A¬B | ¬AB | AB | A¬B |
|---|---|---|---|---|
| ¬C¬D | | | 1 | |
| ¬CD | | 1 | 1 | 1 |
| CD | 1 | 1 | 1 | 1 |
| C¬D | | 1 | 1 | 1 |

Una vez realizado el mapa de Karnauhg nos queda la siguiente función $f(A, B, C, D) = CD + AB + BD + AD + BC + AC$

Por último implementamos un circuito lógico que genera la función que permita determinar si aprueba o suspende un candidato.






# 2. Práctica 2: Diseño de circuitos aritméticos. Sumadores y restadores.

## 2.1. Circuito semi-sumador

Utilizando el simulador lógico, realizamos y comprobamos el funcionamiento de un semi-sumador binario cuya tabla de verdad se representa en la Tabla 2.1. Creamos un símbolo para el semi-sumador como el que se representa en la Figura 2.1.

| $A_0$ $B_0$ | $S_0$ (suma) | $C_1$ (acarreo) |
|---|---|---|
| 0 0 | 0 | 0 |
| 0 1 | 1 | 0 |
| 1 0 | 1 | 0 |
| 1 1 | 0 | 1 |

*Tabla 2.1*

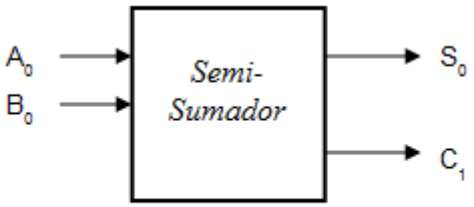
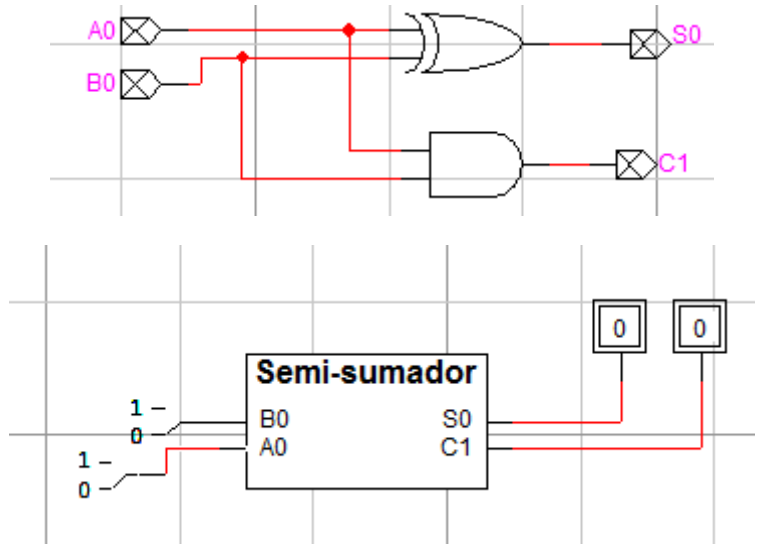



*Figura 2.1*

Una vez creado el semi-sumador binario comprobamos que corresponde a la tabla de verdad 2.1




## 2.2. Circuito sumador completo de 1 bit

A partir de dos semi-sumadores (como el realizado en el apartado 2.1) y las puertas lógicas que consideramos oportunas del simulador, construimos un sumador completo de 1 bit. El símbolo del sumador completo se muestra en la Figura 2.2. Guardamos dicho circuito con el nombre SC1.cct y añada su símbolo en una librería denominada milib.clf.

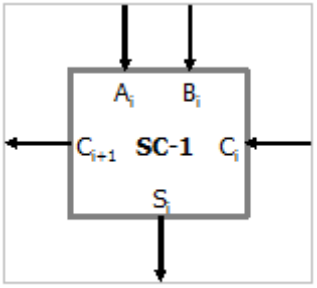
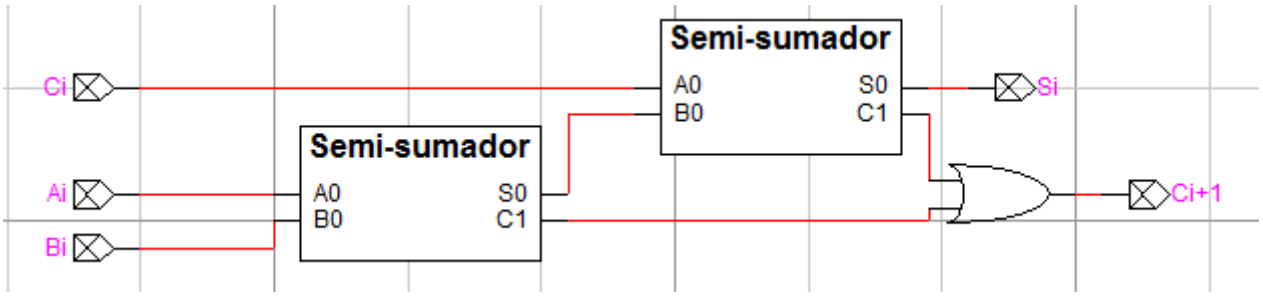
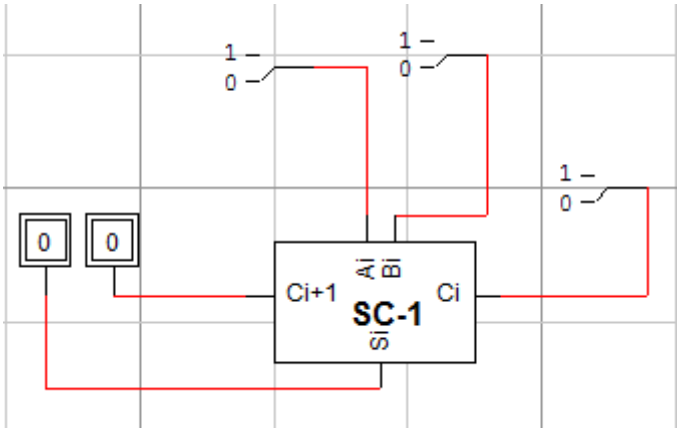



*Figura 2.2*

La resolución del ejercicio se puede ver en las imágenes.

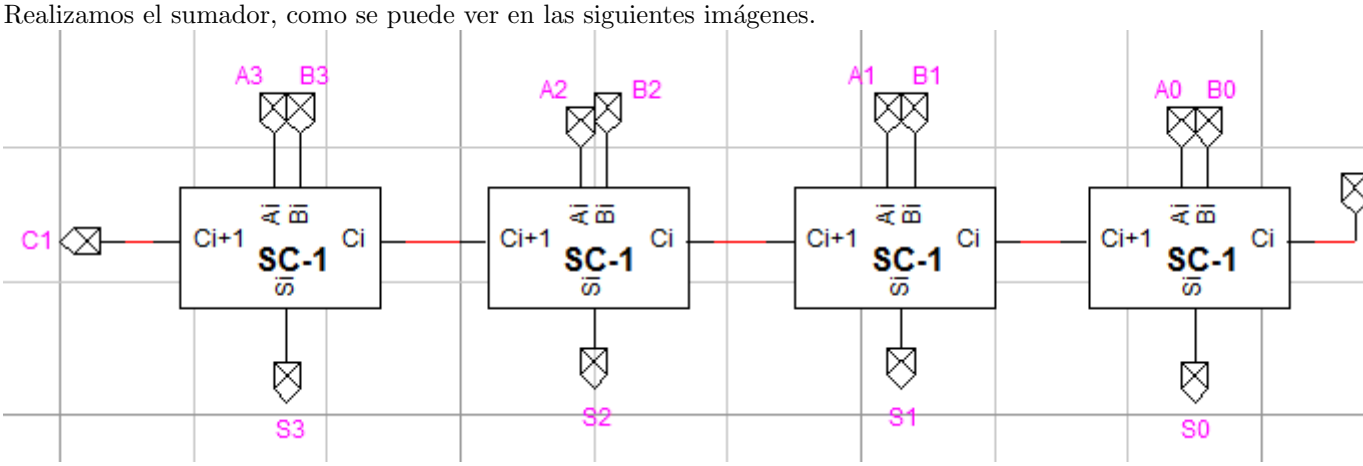


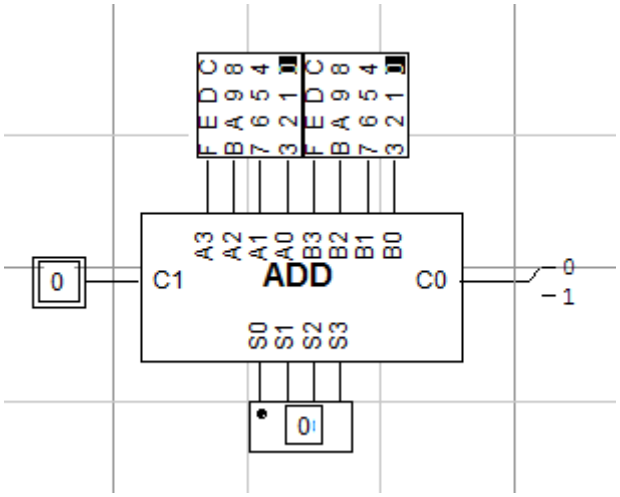



## 2.3. Circuito sumador completo de 4 bits

Utilizando cuatro sumadores completos de 1 bit como el diseñado en el apartado 2.2, realizamos un sumador completo para datos de 4 bits. Utilizamos los componentes HEX DISPLAY y HEX KEYBOARD wo/STB de la biblioteca Simulation IO.clf para realizar el test del circuito. Guarde dicho circuito con el nombre ADD_4.cct y añada su símbolo a la librería milib.clf.

Realizamos el sumador, como se puede ver en las siguientes imágenes.




## 2.4. Circuito sumador-restador de 4 bits

Realizamos un sumador/restador de 4 bits, añadiendo al sumador binario de 4 bits realizado en el apartado 2.3 las puertas lógicas que considere necesarias. En el caso de la resta, ésta se realizará

sumando al minuendo el complemento a dos del sustraendo. Guardamos dicho circuito con el nombre ADD_SUB_4.cct y añadimos su símbolo a la librería milib.clf.

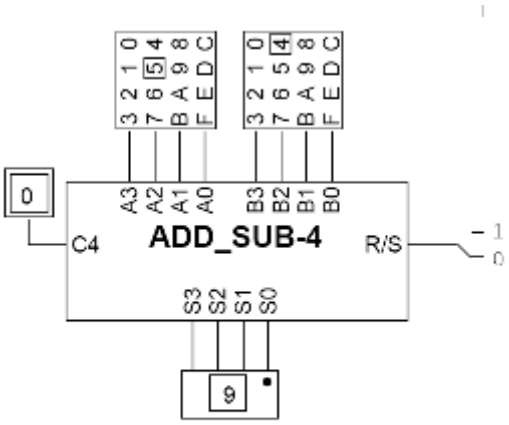
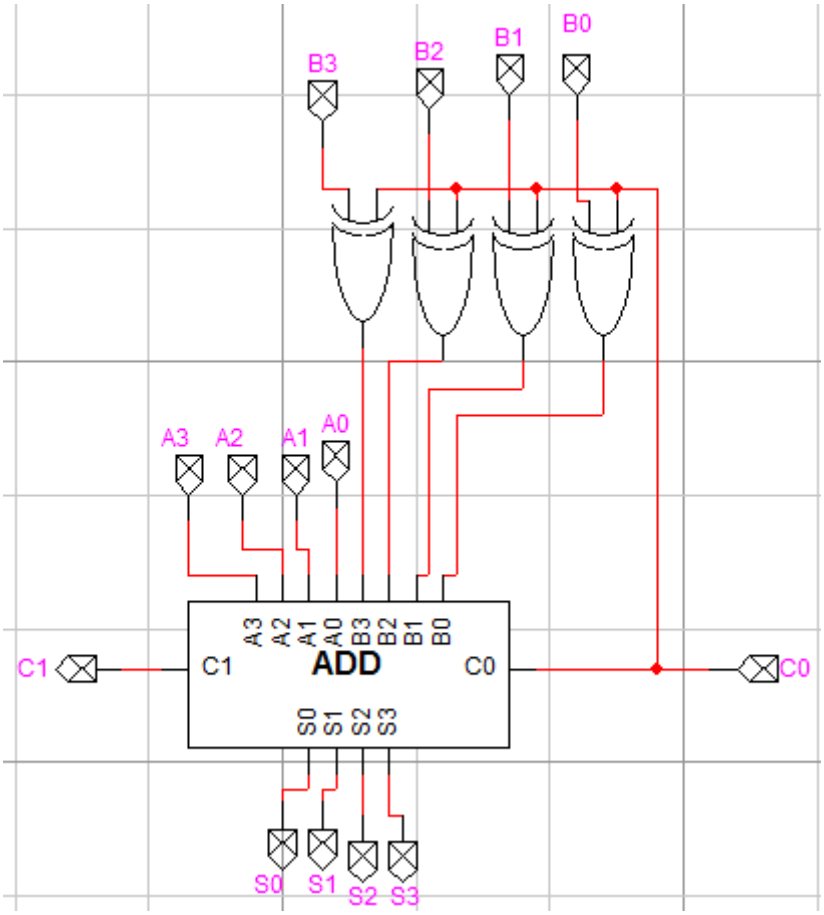
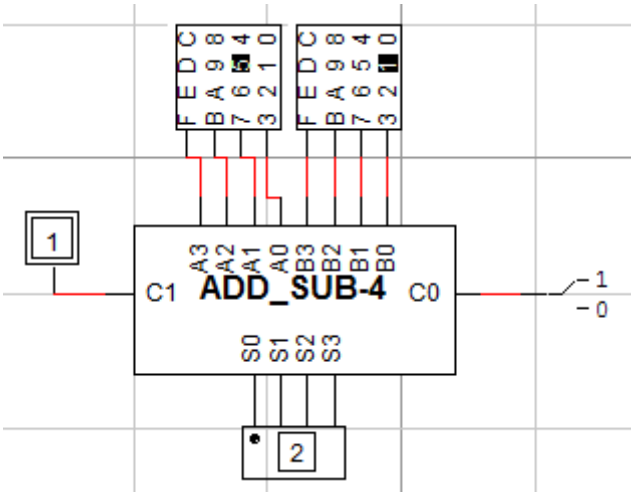



*Figura 2.3*

Creamos el circuito y el símbolo






# 3. Práctica 3: Análisis y realización de una Unidad Aritmético-lógica (ALU) de 4 bits

## 3.1. Estudio preliminar

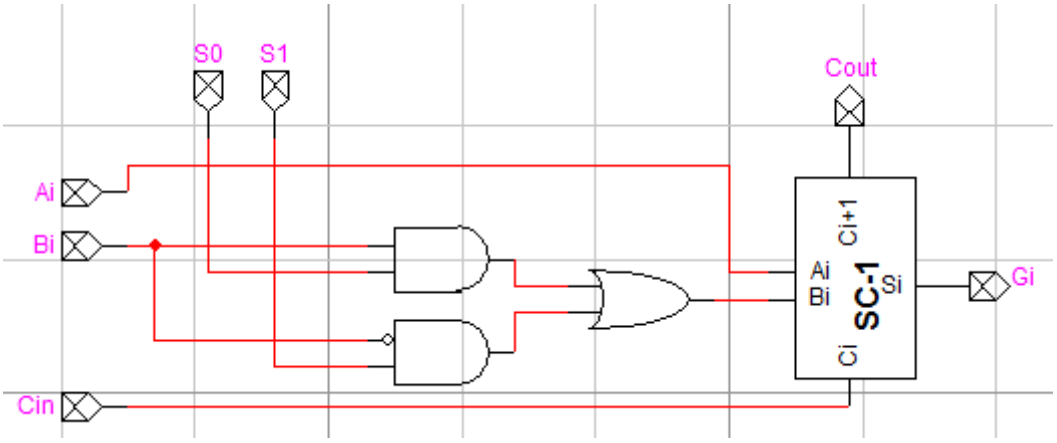
Se pretende analizar, realizar y verificar, utilizando un simulador lógico, una Unidad Aritmético-Lógica (ALU) que sea capaz de operar con 2 datos A y B de 4 bits. Esta ALU proporcionará las operaciones aritméticas y lógicas indicadas en la Figura 3.1. Cada operación se seleccionará según el valor que tomen (1 ó 0) unas señales de control S2, S1, S0 y un acarreo de entrada Cin.

Decrementar A
Sumar: A más B
Transferir A
Transferir complemento de A
Operación A AND B
Sumar A más B más 1
Operación: A EXOR B
Sumar A con el complemento a 1 de B
Incrementar A
Operación: A OR B
Restar: A –B (en complemento a 2)

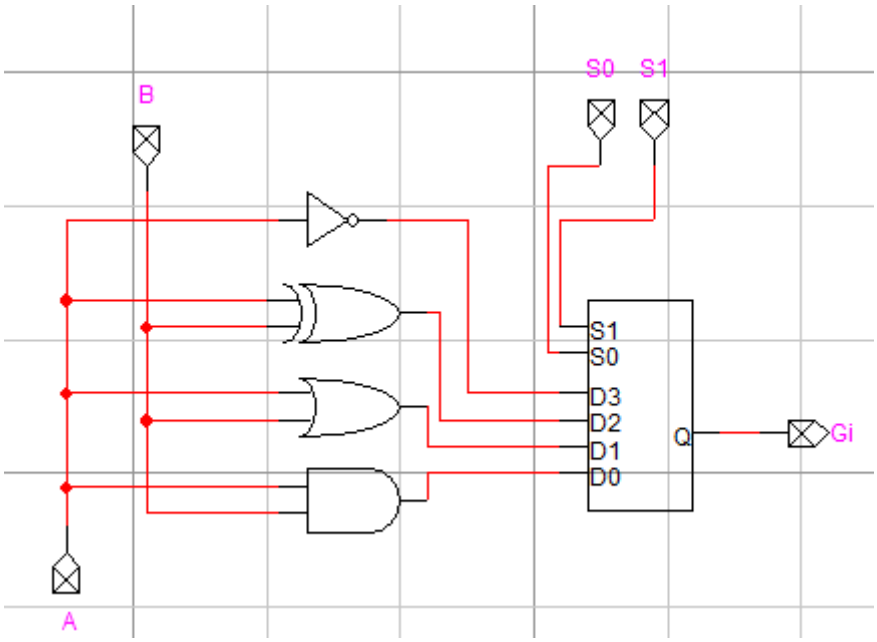
## 3.2. Realización práctica

Analizamos previamente de forma teórica los esquemáticos de los circuitos y etapas de la ALU mostradas en las Figuras 3.2, 3.3 y 3.4. Deducimos, razonadamente, para qué combinaciones de las señales de control S2, S1, S0 y valor de Cin se realiza cada una de las operaciones indicadas en la Figura 3.1. El resultado lo indicamos en la columna de la siguiente Tabla correspondiente a “Operaciones de la ALU deducidas Teóricamente”.

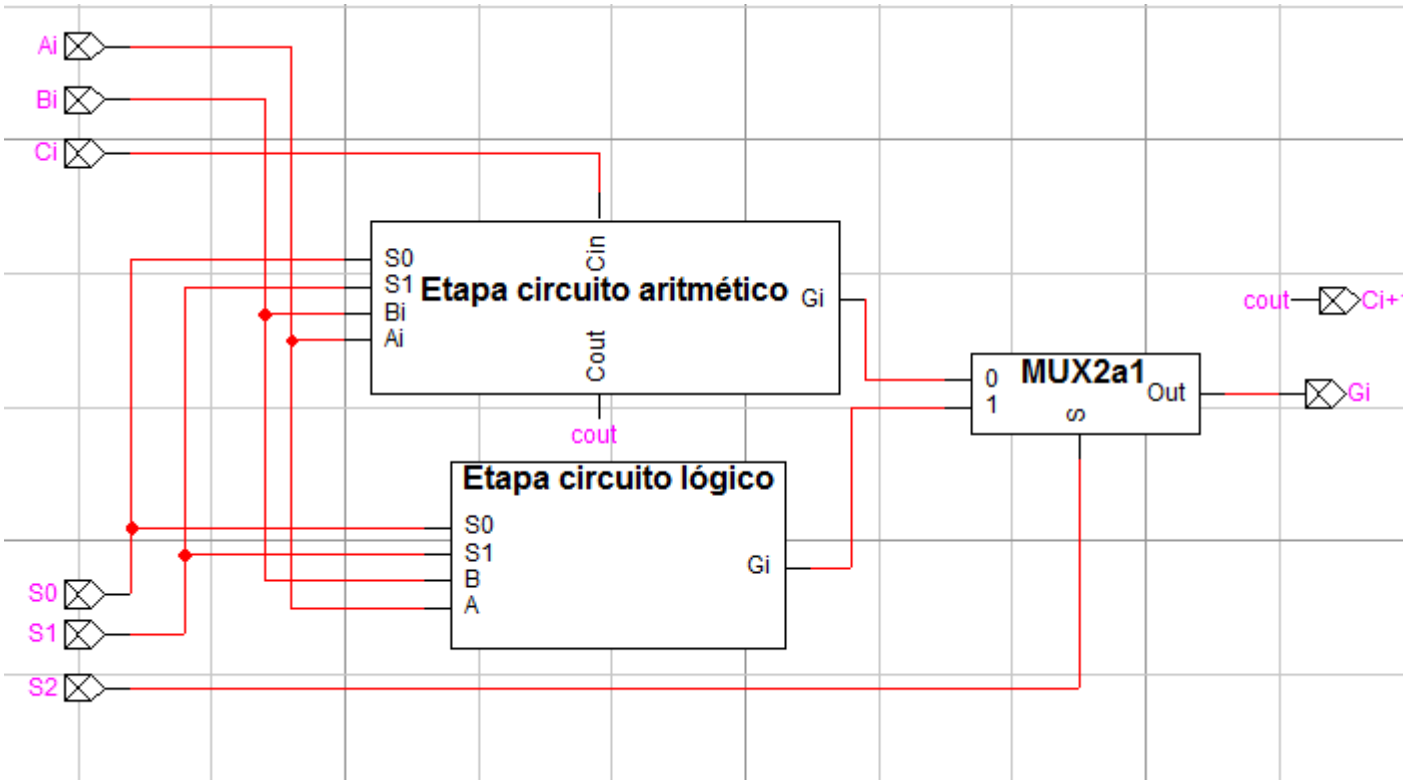
A continuación, en las siguientes imágenes se muestran los pasos para la realización de la ALU de 4 bits.



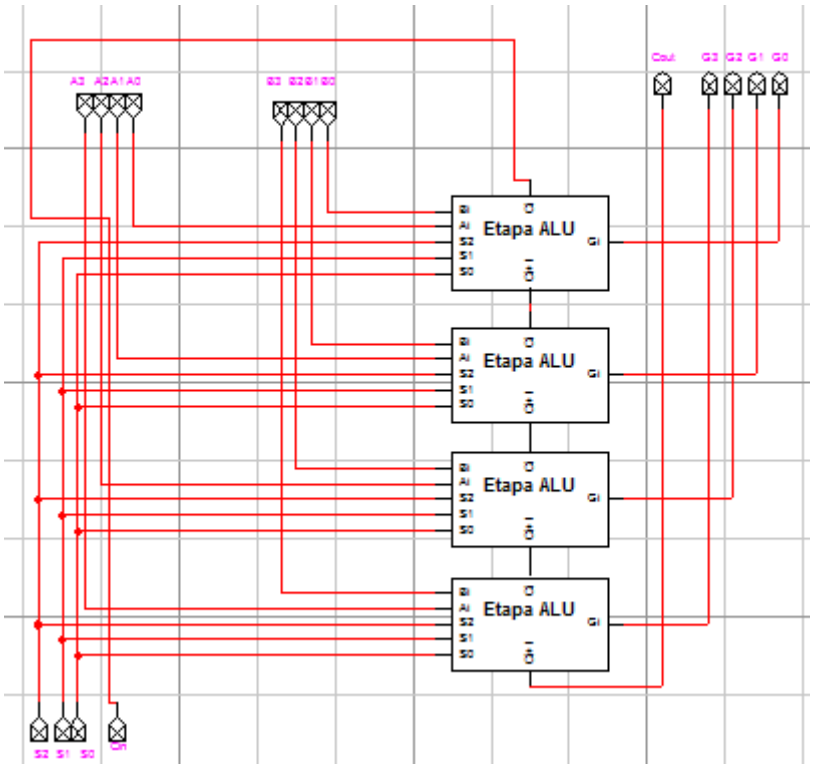

Etapa circuito aritmético



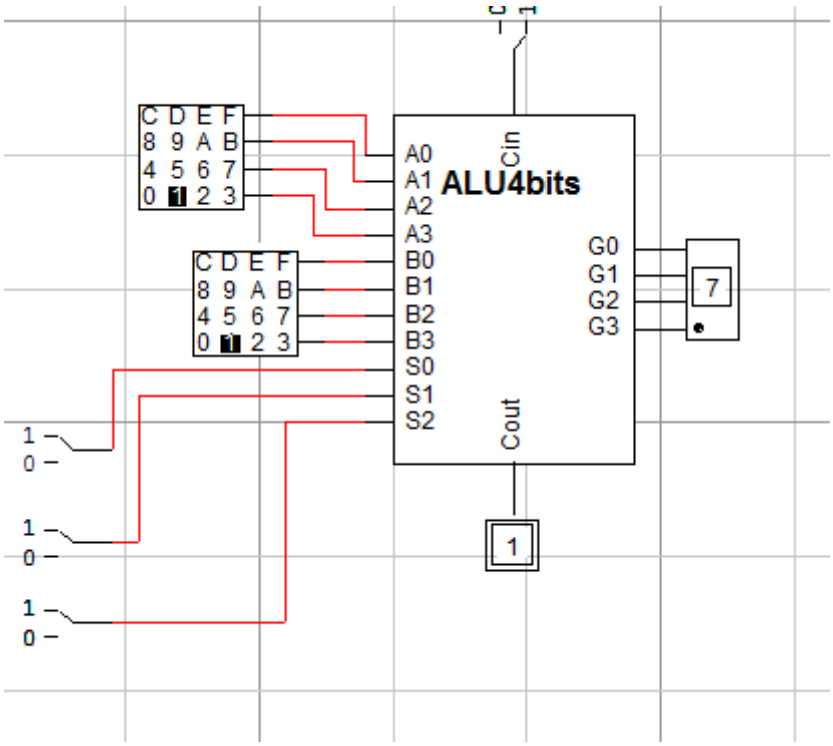

Etapa circuito lógico




Etapa ALU

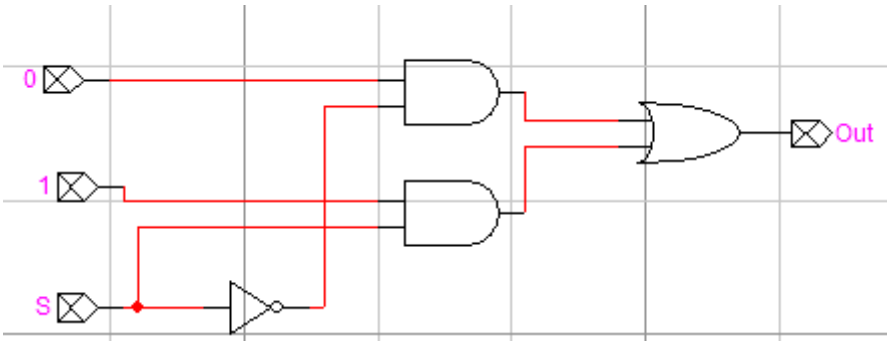



ALU de 4 bits




Circuito de prueba

También se ha creado un multiplexor de 2 a 1

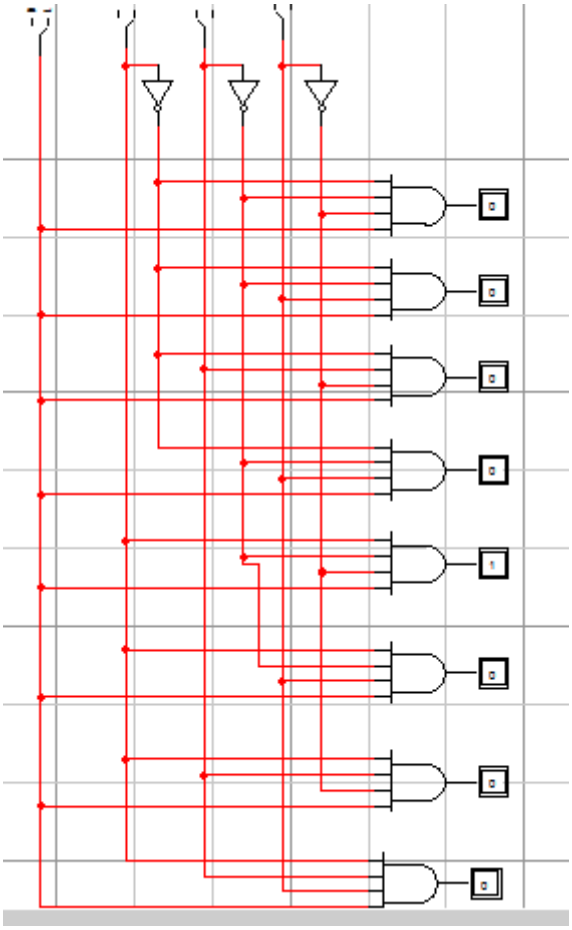



Multiplexor 2 a 1

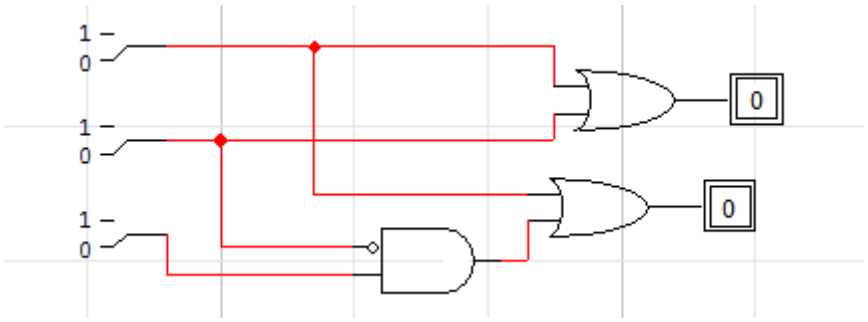
# 4. Práctica 4: Funcionamiento de codificadores/descodificadores y multiplexores/demultiplexores

## 4.1. Descodificadores y codificadores sencillos

Realizamos y simulamos en LogicWorks los siguientes circuitos.
a)Un decodificador binario de 3 entradas y 8 salidas con entrada de habilitación CE.



b) Un codificador binario con prioridad de 4 entradas y 2 salidas.




## 4.2. Conversor a siete segmentos

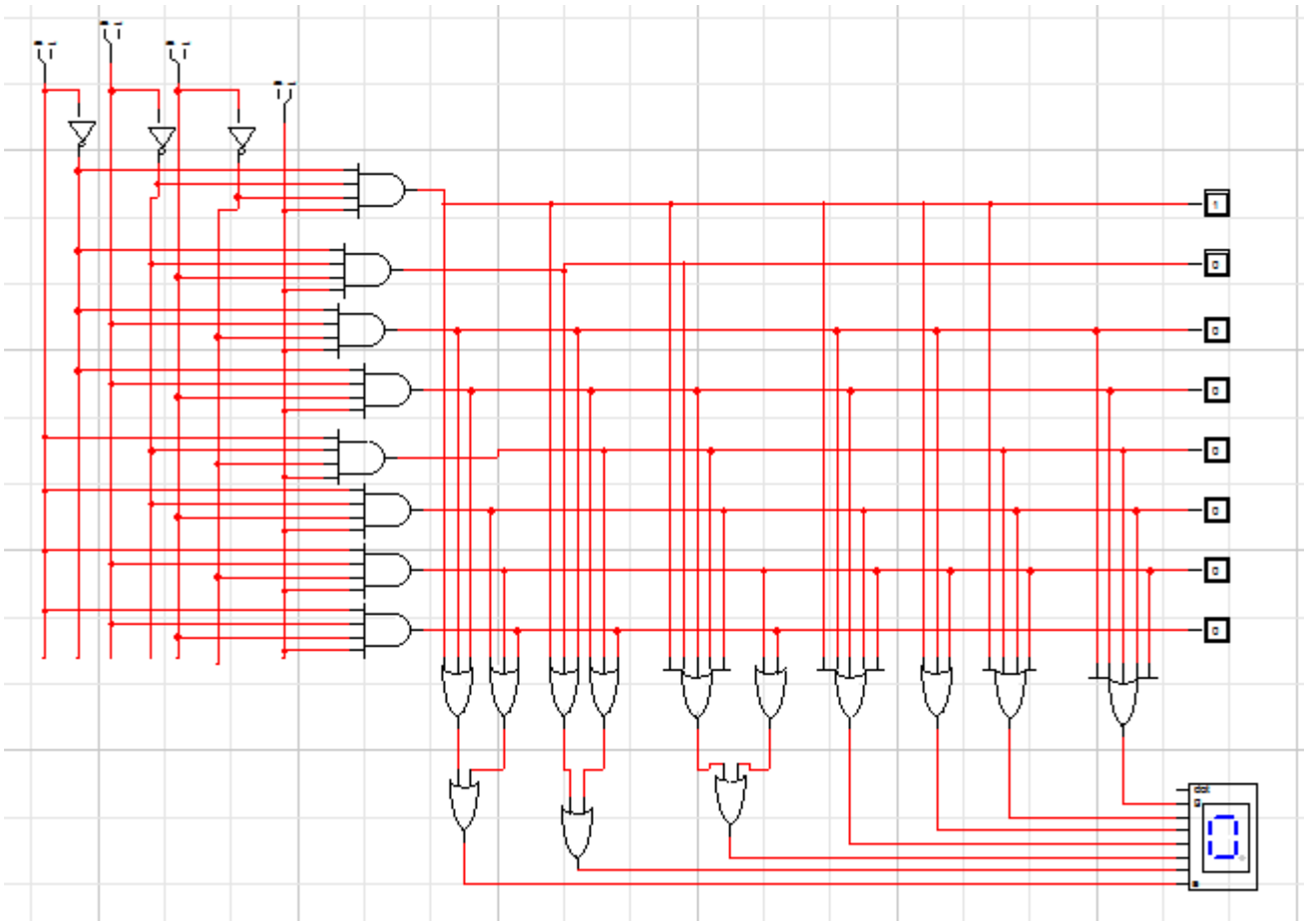
Un conversor de códigos es un circuito combinacional con n entradas y m salidas tales que para cada combinación de entradas se genera una y sólo una combinación de salida. En esta práctica se va a realizar un conversor de código para asignar a dígitos decimales del 0 al 7 un código que permita encender o apagar los led's de un visualizador de 7 segmentos. Realice, utilizando Logic Works un conversor de código para un visualizador de 7 segmentos. Para su realización hay que saber:

a) Los ocho dígitos se codifican en binario con los valores 000 para el 0 hasta el 111 para el 7 utilizando los conmutadores binarios de Logic Works (Binary Switch).

b) Un visualizador de 7 segmentos tiene 7 led's que se encienden o se apagan dependiendo de si hay un 1 (encendido) o un 0 (apagado) en su entrada. Para simularlo, se utiliza el visualizador de 7 segmentos disponible en la biblioteca "SIMULATION IO" de Logic Works.

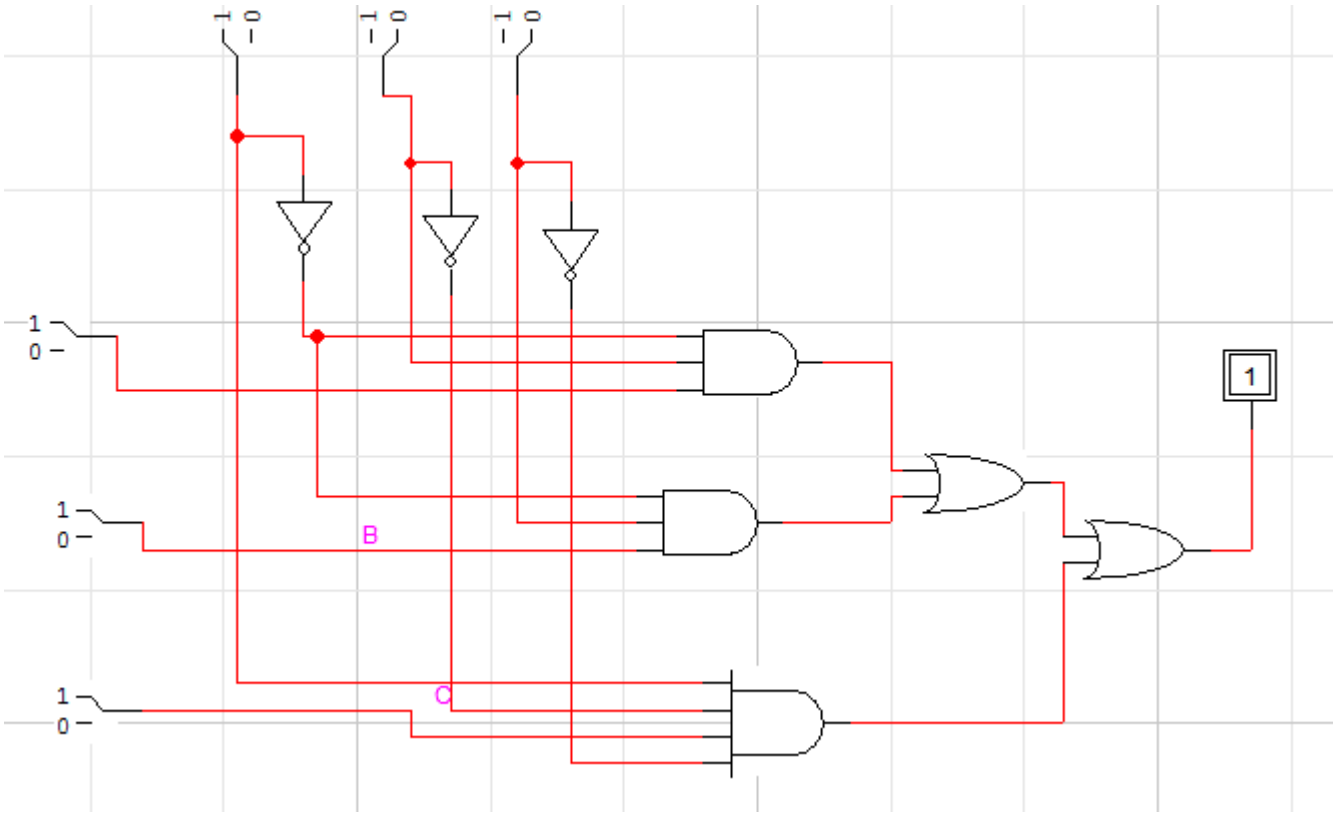
c) Teniendo en cuenta que dichos led's en el display de 7 segmentos reciben un nombre (a, b, c, d, e, f y g, ver figura 4.1), se tendrán que realizar 7 funciones de 3 variables.

El conversor a siete segmentos se puede ver en la imagen



## 4.3. Síntesis de funciones lógicas con multiplexores

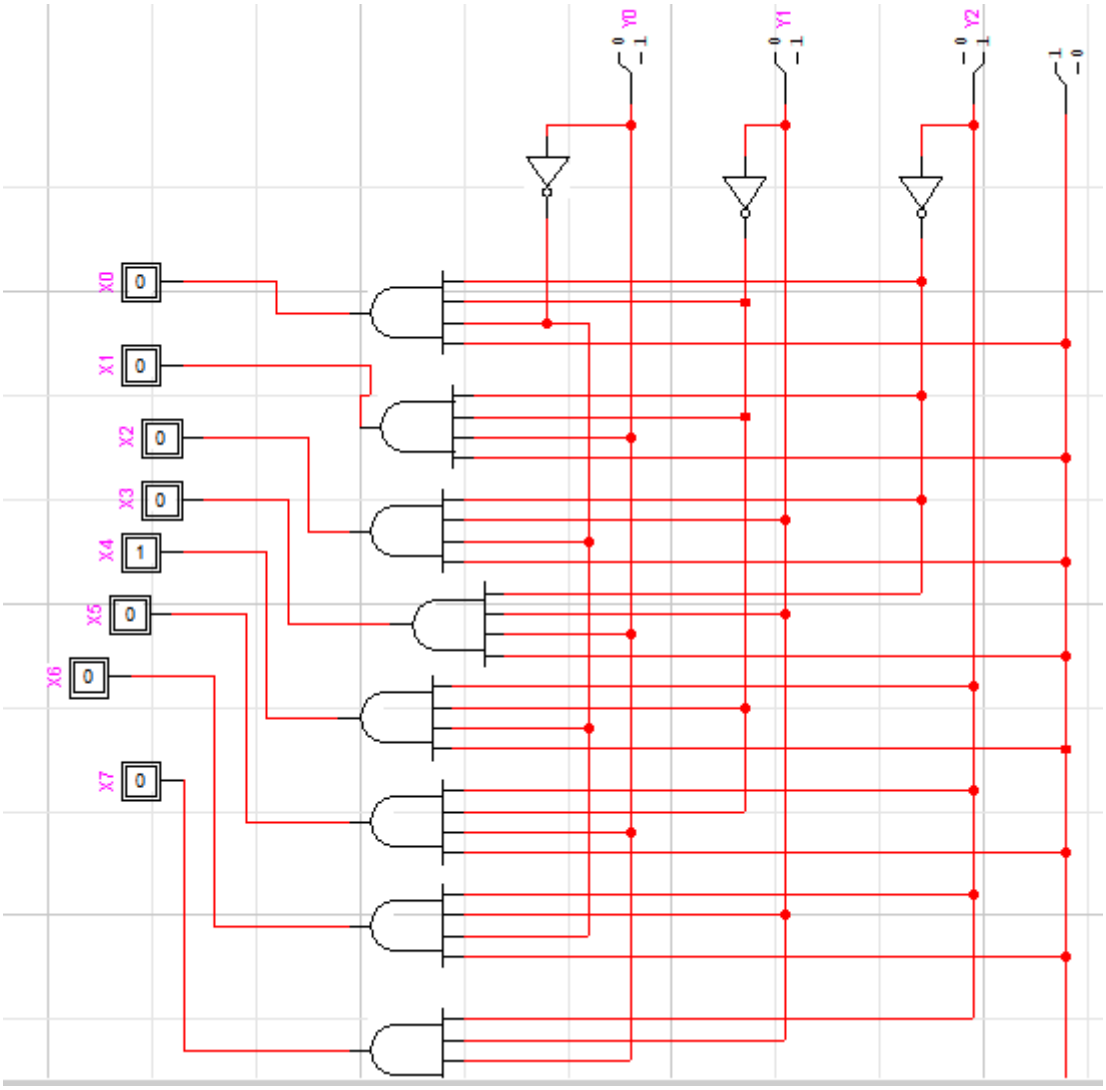
Implementamos la función de tres variables f(A, B, C) cuya tabla de verdad se presenta en la Tabla 4.3, utilizando multiplexores de 2 a 1. Realizamos cada multiplexor a partir de las puertas lógicas de que disponemos en el simulador lógico




## 4.4. Realización de demultiplexores

Realizamos un demultiplexor de 1 a 8. Comparamos este circuito con el decodificador binario de 3 entradas y 8 salidas con entrada de habilitación de chip (CE) implementado en el apartado 4.1.a de esta práctica.

El demultiplexor se puede ver en la siguiente imagen


Y0
Y1
Y2
X0
X1
X2
X3
X4
X5
X6
X7
0
1