INSTITUTO TECNOLÓGICO DE BUENOS AIRES

22.13 ELECTRÓNICA III

Trabajo Práctico de Laboratorio N° 1

Grupo 5:

Matías LARROQUE
Leg. 56597

Lucero Guadalupe FERNANDEZ
Leg. 57485

Manuel MOLLÓN
Leg. 58023

Ezequiel VIJANDE
Leg. 58057

Profesor:

Kevin DEWALD

Entregado: 6 de Septiembre de 2018

1. EJERCICIO 2

1.1. INTRODUCCIÓN

Se tiene una función dada por:

$$f(d,c,b,a) = \prod (M_0, M_1, M_5, M_7, M_8, M_{10}, M_{14}, M_{15})$$

A partir de esto se construyó la siguiente tabla de verdad, completando únicamente los máxterminos de aquellos estados en los que la función valía 0, ya que éstos son los que nos serán de utilidad:

Cuadro 1.1: Tabla de verdad de la función dada.

i	$d = X_1$	$c = X_2$	$b = X_3$	$a = X_4$	$f = ()$	M_i
0	0	0	0	0	0	$X_1 + X_2 + X_3 + X_4$
1	0	0	0	1	0	$X_1 + X_2 + X_3 + \overline{X_4}$
2	0	0	1	0	1	-
3	0	0	1	1	1	-
4	0	1	0	0	1	-
5	0	1	0	1	0	$X_1 + \overline{X_2} + X_3 + \overline{X_4}$
6	0	1	1	0	1	-
7	0	1	1	1	0	$X_1 + \overline{X_2} + \overline{X_3} + \overline{X_4}$
8	1	0	0	0	0	$\overline{X_1} + X_2 + X_3 + X_4$
9	1	0	0	1	1	-
10	1	0	1	0	0	$\overline{X_1} + X_2 + \overline{X_3} + X_4$
11	1	0	1	1	1	-
12	1	1	0	0	1	-
13	1	1	0	1	1	-
14	1	1	1	0	0	$\overline{X_1} + \overline{X_2} + \overline{X_3} + X_4$
15	1	1	1	1	0	$\overline{X_1} + \overline{X_2} + \overline{X_3} + \overline{X_4}$

De esta manera, la función está dada por productos de sumas de las variables (máxterminos).

1.2. SIMPLIFICACIÓN APLICANDO ÁLGEBRA BOOLEANA

Se tiene entonces:

$$f(d,c,b,a) = M_0 * M_1 * M_5 * M_7 * M_8 * M_{10} * M_{14} * M_{15} \tag{1.1}$$

Que es equivalente a:

$$\begin{aligned} f(d,c,b,a) = {} & (X_1 + X_2 + X_3 + X_4) * (X_1 + X_2 + X_3 + \overline{X_4}) * (X_1 + \overline{X_2} + X_3 + \overline{X_4}) * (X_1 + \overline{X_2} + \overline{X_3} + \overline{X_4}) \\ & * (\overline{X_1} + X_2 + X_3 + X_4) * (\overline{X_1} + X_2 + \overline{X_3} + X_4) * (\overline{X_1} + \overline{X_2} + \overline{X_3} + X_4) * (\overline{X_1} + \overline{X_2} + \overline{X_3} + \overline{X_4}) \end{aligned} \tag{1.2}$$

Utilizando la propiedad del álgebra booleana $(x + y)(x + \overline{y} = x)$ sobre los pares de términos $(M_0, M_1); (M_5, M_7); (M_8, M_{10}); (M_{14}, M_{15})$ se llegó a la siguiente expresión simplificada.

$$f(d,c,b,a) = (X_1 + X_2 + X_3) * (\overline{X_1} + \overline{X_2} + \overline{X_3}) * (X_1 + \overline{X_2} + \overline{X_4}) * (\overline{X_1} + X_2 + X_4) \tag{1.3}$$

Cuadro 1.2: Mapa de Karnaugh de la función dada.

x_3, x_4 \ x_1, x_2	00	01	11	10
00	M_0	M_4	M_{12}	M_8
01	M_1	M_5	M_{13}	M_9
11	M_3	M_7	M_{15}	M_{11}
10	M_2	M_6	M_{14}	M_{10}

$\longrightarrow$

x_3, x_4 \ x_1, x_2	00	01	11	10
00	0	1	1	0
01	0	0	1	1
11	1	0	0	1
10	1	1	0	0

1.3. Simplificación aplicando mapas de Karnaugh

Nos es de interés para simplificar aplicando mapas de Karnaugh aquellos máxterminos en los que la función vale 0, agrupándolos de a 8, 4, 2 ó 1, según se pueda e intentando que los grupos sean lo más grande posible. Si tomamos 4 grupos de a 2 verticales, como muestra la tabla

Figura 1.1: Mapa de Karnaugh de la función dada.

x_3, x_4 \ x_1, x_2	00	01	11	10
00	0	1	1	0
01	0	0	1	1
11	1	0	0	1
10	1	1	0	0

Aquellos seleccionados de manera unitaria en la última columna se tomaron como un grupo de 2. Con ésto en cuenta, se puede notar lo siguiente de cada pareja:

- En el primer grupo de la primera columna, las variables X_1, X_2 *y* X_3 valen 0 y no cambian; sí lo hace X_4.
- En el segundo grupo, en la segunda columna, las variables que no cambian son X_1, X_2 *y* X_4; X_1 vale 0, y X_1 *y* X_2 valen 1, de manera contraria, sí cambia X_3.
- En el tercer grupo de ceros, X_1, X_2 *y* X_3 no cambian y valen 1; notamos que X_4 sí modifica su valor.
- En el cuarto grupo en la última columna, X_1, X_2 *y* X_4 mantienen su valor; con X_1 valiendo 1, y $X_2,\ X_4$, 0; X_3 sí modifica su valor.

Para realizar entonces la simplificación, recordemos que queremos productos de sumas tal que la función sea nula. En el caso del primer grupo, la manera de lograrlo es $X_1 + X_2 + X_3$, ya que sólo cuando las tres variables valen cero simultáneamente, este término valdrá cero; como X_4 cambia su valor no nos es de interés para la expresión. De manera

análoga para el segundo grupo, podemos lograr que valga cero únicamente con $X_1 + \overline{X_2} + \overline{X_4}$. El hecho de negar las variables se debe a que estas valen 1, y se quiere lo contrario. Para el tercer grupo, el cero se consigue como $\overline{X_1} + \overline{X_2} + \overline{X_3}$, las tres variables valen 1, por eso se niegan. Con el cuarto y último grupo, la nulidad se consigue con $\overline{X_1} + X_2 + X_4$.

Si se combinan los términos, terminamos con la siguiente expresión:

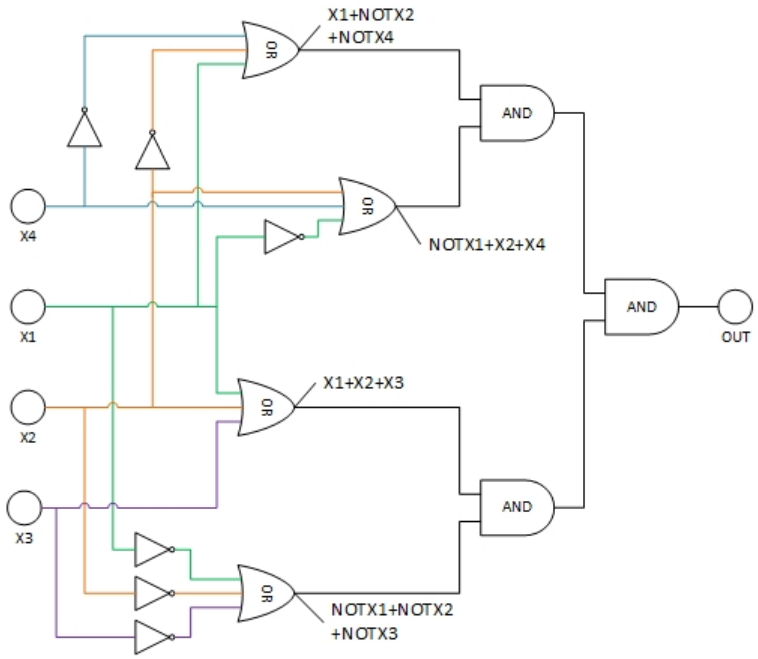
$$f(d,c,b,a) = (X_1 + X_2 + X_3) * (X_1 + \overline{X_2} + \overline{X_4}) * (\overline{X_1} + \overline{X_2} + \overline{X_3}) * (\overline{X_1} + X_2 + X_4) \tag{1.4}$$

ES fácil ver que la expresión es idéntica a la simplificación con el álgebra booleana. Vale agregar que tomando los grupos más grandes posibles al hacer la simplificación con mapas de Karnaugh se llega a la expresión más simplificada posible. Esto verifica también que con el álgebra booleana se pudo reducir la función a su mínima expresión.

1.4. Circuito lógico resultante

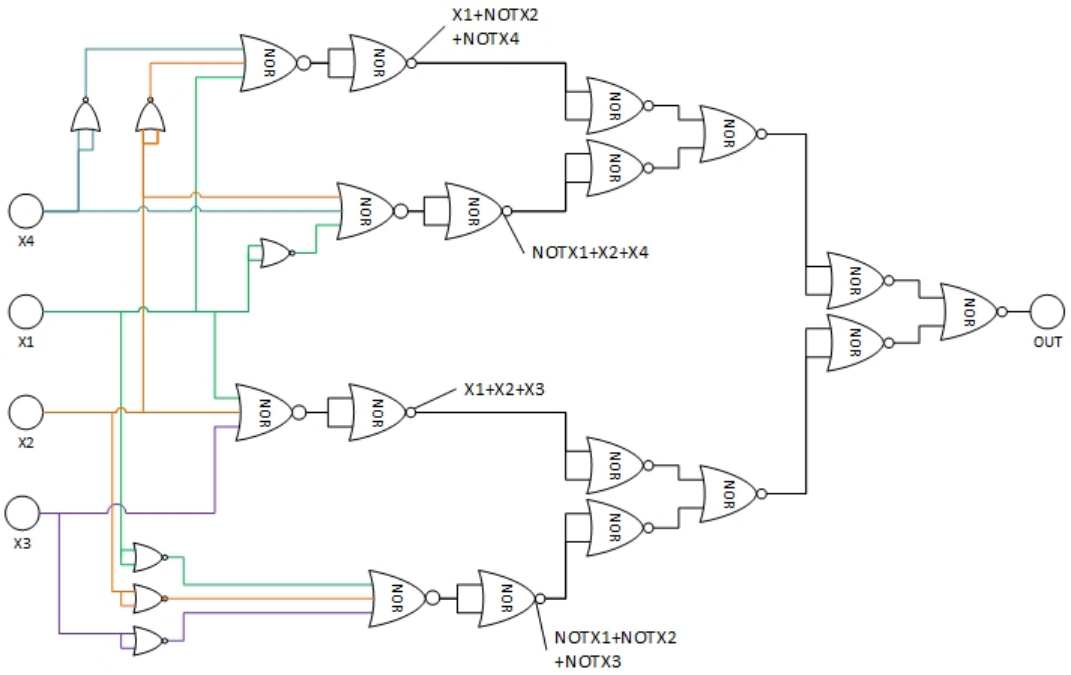
1.4.1. Utilizando compuertas AND, OR, NOT.

Para implementar el circuito lógico, simplemente se tuvo en cuenta la expresión simplificada y se utilizaron compuertas AND, OR, y NOT cuando se requerían.



1.4.2. Utilizando compuertas NOR.

En este caso se utilizó la compuerta universal NOR para crear las compuertas necesarias (AND, OR y NOT). De esta manera, se utiliza un número mayor de compuertas, pero universales. El circuito resultante se muestra a continuación:



2. EJERCICIO 4

2.1. INTRODUCCIÓN

Para implementar un circuito que convierta un número binario de 4 bits en su complemento a dos se empezó pensando este circuito como una caja negra con 4 entradas y 4 salidas:

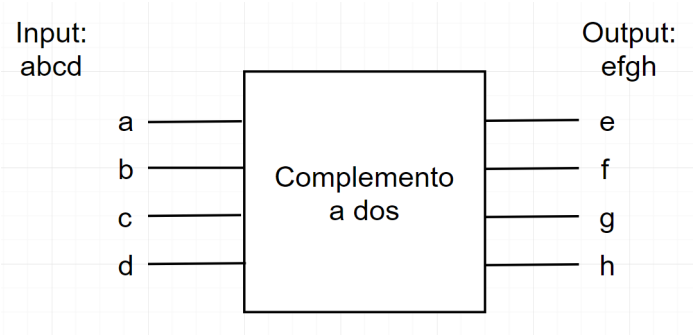


Figura 2.1: Caja Negra Complemento a 2

A su vez, sabemos que el complemento a dos se realiza aplicando el complemento a uno y luego sumando uno al resultado. Por lo que se puede representar mediante el siguiente conjunto de çajas negras":

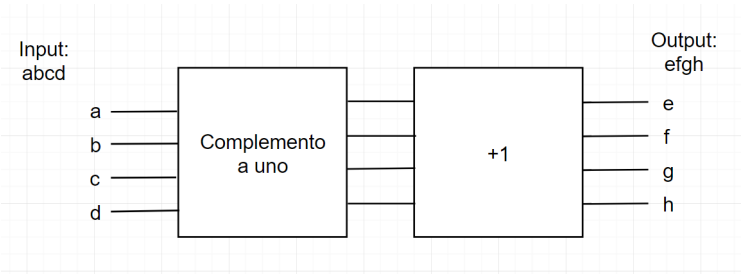


Figura 2.2: Caja Negra Complemento a 2

La salida tras el complemento a uno sabemos que es cada bit negado, es decir que atraviesan una compuerta NOT. De esta manera a la salida del complemento a uno que se obtiene es:

$$\overline{a}\,\overline{b}\,\overline{c}\,\overline{d}$$

Ahora lo único que falta es sumarle uno.

2.2. Expresión de la salida en mintérminos

La expresión de la salida queda determinada por las entradas a través de la siguiente fórmula deducida en la introducción:

$$\begin{array}{c} \overline{a}\,\overline{b}\,\overline{c}\,\overline{d} \\ +0001 \\ \hline efgh \end{array}$$

Para la creación de las tablas de valor se debe observar que el output de cada bit depende del resultado de el input de los bits menos significativos que con el que se está trabajando. Para el BMS la tabla de verdad es muy simple y no vale la pena el uso de mintérminos por lo que se verá a continuación:

Cuadro 2.1: Tabla para bit h

$\overline{d}$	**1**	**h**
1	1	0
0	1	1

Fácilmente de la tabla se puede observar que h resulta ser la entrada negada. Por lo que resulta la primera relación de entrada-salida:

$$h(d) = \overline{\overline{d}} \tag{2.1}$$

Para la salida correspondiente al bit g la tabla de verdad es de la siguiente manera:

Cuadro 2.2: Tabla para bit g

$\overline{c}$	$\overline{d}$	**g**
0	0	0
0	1	1 (m_1)
1	0	1 (m_2)
1	1	0

Al escribir la salida en función de los mintérminos se llega a la siguiente expresión:

$$g(c,d) = m_1 + m_2$$

$$g(c,d) = \overline{\overline{c}}\,\overline{d} + \overline{c}\,\overline{\overline{d}} \tag{2.2}$$

Ahora para el bit f se requiere una tabla de verdad de tres variables ya que ahora depende del carry causado por las variables de entrada c y d:

Cuadro 2.3: Tabla para bit f

$\overline{b}$	$\overline{c}$	$\overline{d}$	**f**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1(m_3)
1	0	0	1(m_4)
1	0	1	1(m_5)
1	1	0	1(m_6)
1	1	1	0

Al escribir la salida en función de los mintérminos se llega a la siguiente expresión:

$$f(b,c,d) = m_3 + m_4 + m_5 + m_6$$

$$f(b,c,d) = \overline{\overline{b}}\,\overline{c}\,\overline{d} + \overline{\overline{b}}\,\overline{\overline{c}}\,\overline{\overline{d}} + \overline{b}\,\overline{\overline{c}}\,\overline{d} + \overline{b}\,\overline{c}\,\overline{\overline{d}} \tag{2.3}$$

Para el bit e la tabla de verdad es la siguiente:

Cuadro 2.4: Tabla para bit e

$\overline{a}$	$\overline{b}$	$\overline{c}$	$\overline{d}$	**e**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	$1(m_6)$
0	1	1	1	$1(m_7)$
1	0	0	0	$1(m_8)$
1	0	0	1	$1(m_9)$
1	0	1	0	$1(m_{10})$
1	0	1	1	$1(m_{11})$
1	1	0	0	$1(m_{12})$
1	1	0	1	$1(m_{13})$
1	1	1	0	$1(m_{14})$
1	1	1	1	0

Al escribir la salida en función de los mintérminos se llega a la siguiente expresión:

$$e(a,b,c,d) = m_6 + m_7 + m_8 + m_9 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14}$$

$$e(a,b,c,d) = \overline{\overline{a}}b\overline{\overline{c}}\overline{d} + \overline{\overline{a}}b\overline{\overline{c}}\overline{\overline{d}} + \overline{\overline{a}}b\overline{\overline{c}}d + \overline{\overline{a}}b\overline{c}\overline{\overline{d}} + \overline{\overline{a}}b\overline{c}\overline{d} + \overline{\overline{a}}bc\overline{\overline{d}} + \overline{\overline{a}}bc\overline{d} + \overline{\overline{a}}bcd \tag{2.4}$$

2.3. SIMPLIFICACIÓN DE LAS ECUACIONES RESULTANTES

Para reducir las 4 expresiones resultantes procedemos a utilizar la propiedad $\overline{\overline{a}} = a$.
Lo cual da como resultado las 4 siguientes ecuaciones:

$$h = d$$
$$g = c\overline{d} + \overline{c}d$$
$$f = b\overline{c}\overline{d} + \overline{b}cd + \overline{b}c\overline{d} + \overline{b}\overline{c}d$$
$$e = a\overline{b}\overline{c}\overline{d} + \overline{a}bcd + \overline{a}bc\overline{d} + \overline{a}b\overline{c}d + \overline{a}b\overline{c}\overline{d} + \overline{a}\overline{b}cd + \overline{a}\overline{b}c\overline{d} + \overline{a}\overline{b}\overline{c}d$$

De estas cuatro expresiones las primeras son irreducibles. Pero es posible reducir la tercera y la cuarta con el uso de las siguientes propiedades:

$$a + a = a$$
$$a + \overline{a} = 1$$
$$ab + cb = b(a + c)$$

La primera propiedad significa que puedo sumar uno de los mintérminos a la expresión que esta no se verá afectada si ese mintérmino ya pertenecía a la expresión. Con la segunda expresión puedo sacar 'factor común' los términos que difieran en una variable y su complemento de manera que al hacer el factor común quede de la forma $f(a,b,...)(z+\overline{z})$ y así utilizar la tercera propiedad que simplifica esa expresión a $f(a,b,...)$. De esta manera las ecuaciones se simplifican a su forma final que es de la siguiente manera:

$$h = d$$
$$g = c\overline{d} + \overline{c}d$$
$$f = b\overline{c}\overline{d} + \overline{b}c + \overline{b}d$$
$$e = a\overline{b}\overline{c}\overline{d} + \overline{a}cd + \overline{a}\overline{b}c + \overline{a}b$$

2.4. Circuito Resultante

Al implementar las expresiones halladas en la sección anterior con compuertas AND, OR y NOT se llega al siguiente circuito:

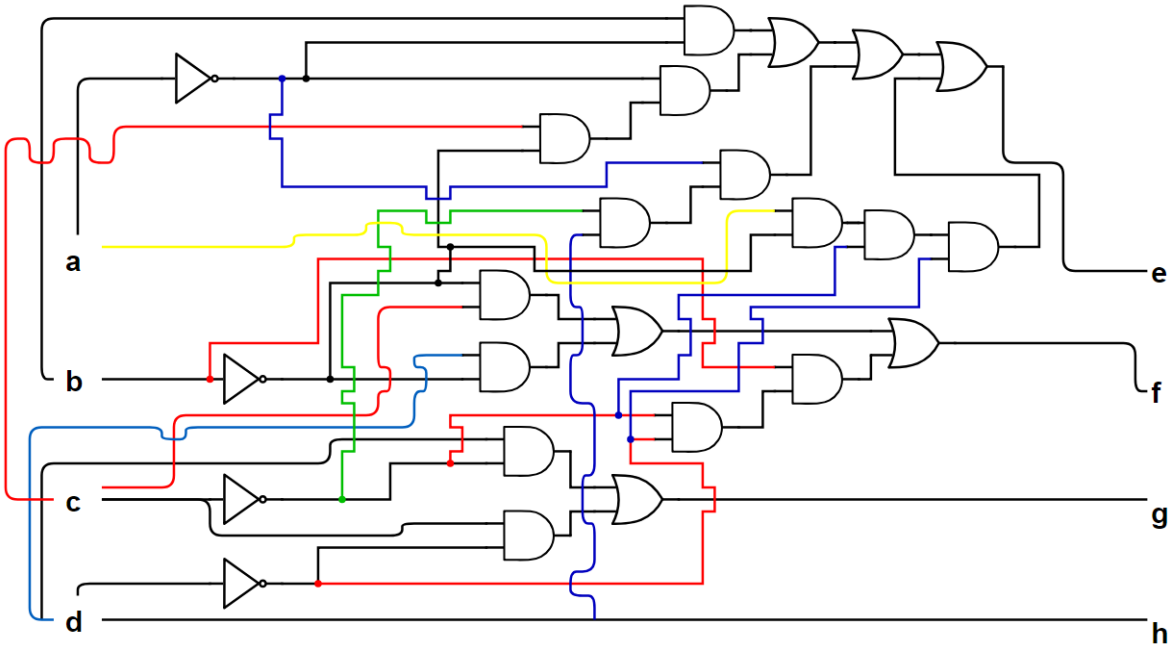


Figura 2.3: Circuito de conversión a complemento a dos

3. Ejercicio 5

3.1. Introducción

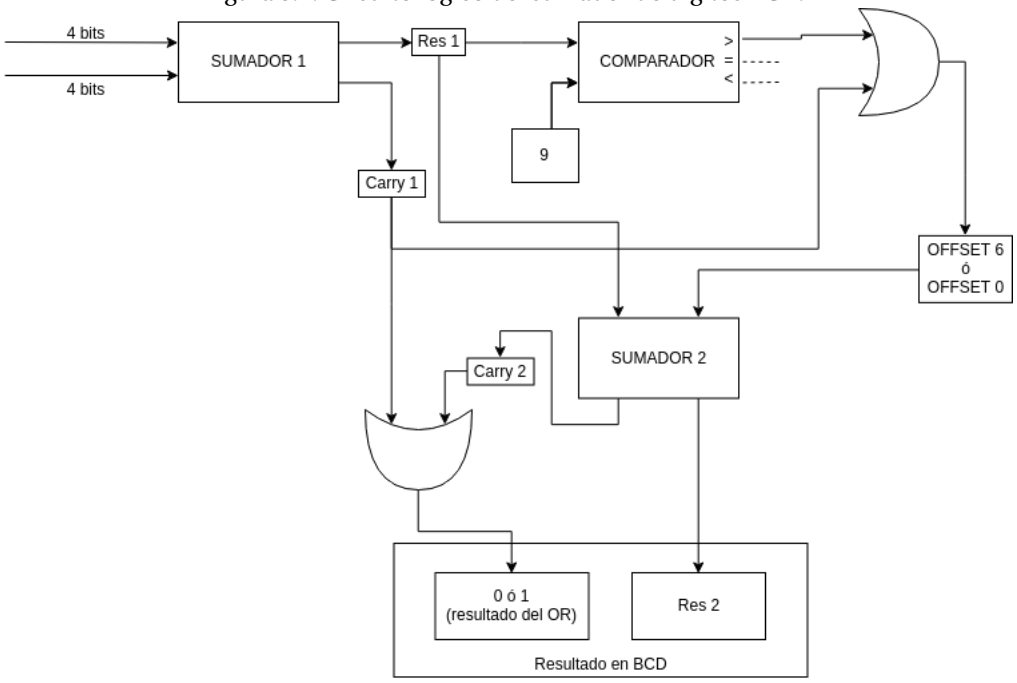
Se implementó un sumador de dos números de un dígito en BCD, expresando la salida como un número de dos dígitos, también en BCD. Para ésto, primero se realizó un análisis sobre la suma en este formato, y luego se implementó en un circuito.
Se tienen entonces dos números en BCD, que son dos dígitos menores a 9 (pues están en dicho formato). Se realiza

una suma convencional en binario. Seguidamente, se compara con 9, ya que si la suma es menor o igual a éste, está representado automáticamente en BCD. Si la suma es mayor a 9, se le debe sumar un offset de 6 para obtener un 1 en el dígito más significativo, y así, el resultado final. El offset de 6 se debe a que en binario, se puede representar como máximo el número 15; en BCD, por otro lado, hasta el 9. La diferencia entre ambos es de 6, por lo que si queremos que 9+1=10 ($10_{10} = 1010_2 = 0001\ \ 0000_{BCD}$), tenemos que sumar 6 para que 1010_2 se convierta en $1\ \ 0000_{BCD}$.

3.2. Circuito lógico

Con este razonamiento, se realizó la siguiente implementación:

Figura 3.1: Circuito lógico del sumador de dígitos BCD.



Se puede ver de manera sencilla la lógica que se siguió para el sumador pedido. El resultado final serán dos nibbles. El nibble más significativo es el OR entre los carry de ambos sumadores, que se relaciona con la comparación de si la suma convencional en binario da mayor a 9 o no. Por último el nibble menos significativo es el resultado del sumador 2, es decir, la suma en binario con el offset, que es +6 ó 0; siendo 0 si el Sumador 1 no da mayor a 9.

4. Ejercicio 6

4.1. Introducción

Se implementó una ALU con las siguientes operaciones: suma, resta, complemento a dos, shift left, y como operaciones bit a bit: AND, OR, NOT, XOR. Se tomó como entrada dos nibbles como los operandos, y un selector de 3 bits para seleccionar la operación. Por el otro lado, la salida consta de un resultado, además de un bit de carry u overflow. Se reutilizaron la mayor cantidad posible de módulos previamente programados y testeados en los incisos anteriores. La razón de esto es que si se testearon los módulos de manera individual, al combinarlos, no habrá que volver a probarlos, pues ya han sido testeados. Es decir, asegurándonos que cada módulo funciona individualmente, la combinación lineal de éstos también funciona, por linealidad.

4.2. ESQUEMA LÓGICO

El circuito lógico resultante se presenta a continuación:

Figura 4.1: Esquema lógico de la ALU.