

Práctica 2 - Lógica Digital - Parte A

Organización del Computador 1

Primer Cuatrimestre 2018 - Turno Mañana

Todas las compuertas mencionadas en esta práctica son de 1 ó 2 entradas, a menos que se indique lo contrario. Usaremos los símbolos detallados a continuación para representar las distintas funciones lógicas: XOR $\rightarrow \oplus$, NAND $\rightarrow \downarrow$, NOR $\rightarrow \downarrow$.

Durante la presente práctica se recomienda fuertemente la utilización de un simulador para experimentar con los componentes y circuitos propuestos y verificar las soluciones. Una recomendación es el Logisim (<http://www.cburch.com/logisim/>).

Circuitos Combinatorios

Ejercicio 1 Demostrar la equivalencia de las siguientes fórmulas booleanas:

a) $p = (p.q) + (p.\bar{q})$

b) $x.z = (x + y).(x + \bar{y}).(\bar{x} + z)$

Ejercicio 2 ¹ Sea $p \oplus q = (\bar{p}.q) + (p.\bar{q})$. Demostrar si la siguiente propiedad de distributividad es verdadera o es falsa:

$$x \oplus (y.z) = (x \oplus y).(x \oplus z)$$

Ejercicio 3 Determinar la veracidad o falsedad de las siguientes afirmaciones:

a) Sea $p|q = \bar{p}.\bar{q}$ ¿Alcanza con este operador para representar todas las funciones booleanas?

b) Sea $p \downarrow q = \overline{p+q}$ ¿Alcanza con este operador para representar todas las funciones booleanas?

Ejercicio 4 Mostrar cómo se puede construir la función booleana $f(A, B) = A.B$ a partir de 2 compuertas NAND. Mostrar además cómo construir la función $f(A) = \bar{A}$ utilizando únicamente compuertas NAND.

Ejercicio 5 Dibujar un circuito que implemente la función booleana $f(A, B, C) = A.B.C$ usando 2 compuertas NOR y varias compuertas NOT.

Ejercicio 6 Dada la función booleana F definida a partir de la siguiente tabla de verdad,

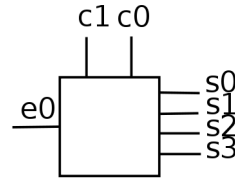
A	B	C	$F(A, B, C)$
1	1	0	0
1	0	0	1
1	0	1	1
0	1	0	0
0	0	1	0
0	1	1	1
1	1	1	1
0	0	0	0

¹Ej 7. Capítulo 3 del L. Null & J. Lobur - Essentials Of Computer Organization And Architecture

- a) Escribir la *suma de productos* para la función F . Calcular la cantidad de compuertas que la implementación literal de la función requeriría.
- b) ¿Se puede simplificar la expresión usando propiedades del álgebra booleana? Dibujar el circuito correspondiente utilizando la menor cantidad de compuertas que pueda.

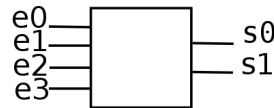
Ejercicio 7 Dibujar el diagrama lógico de un *demultiplexor* de 2 líneas de control, 1 línea de entrada y 4 líneas de salida. Este circuito dirige la única línea de entrada a una de cuatro líneas de salida, dependiendo del estado de las dos líneas de control.

c_1	c_0	s_i
0	0	$s_0 = e_0, s_i = 0$ si $i \neq 0$
0	1	$s_1 = e_0, s_i = 0$ si $i \neq 1$
1	0	$s_2 = e_0, s_i = 0$ si $i \neq 2$
1	1	$s_3 = e_0, s_i = 0$ si $i \neq 3$



Ejercicio 8

- a) Dibujar el diagrama lógico de un *codificador* de 4 líneas de entrada (e_i) y 2 líneas de salida (s_i). Si únicamente e_i está alta, las salidas deben representar el número i en notación sin signo. No está definido cuál es el resultado si no se cumple que sólo una de las líneas de entrada tiene valor 1.



- b) Dotar al circuito anterior de una salida adicional que indique si el estado de la entrada es válido o inválido.

Ejercicio 9

- a) Dibujar con compuertas lógicas el circuito de un *decodificador* de 2 líneas de entrada (e_i) y 4 líneas de salida (s_i), cuya tabla de verdad es la siguiente:

e_1	e_0	s_3	s_2	s_1	s_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

- b) Usando el circuito anterior, reescribir el *demultiplexor* de 1 línea de entrada, 2 líneas de control y 4 líneas de salida.

Ejercicio 10 Un *carry left shifter 3-4* es un componente de 3 líneas de entrada (e_2, e_1, e_0), 4 líneas de salida (s_3, s_2, s_1, s_0) y un línea de control (c_0) que se comporta de la siguiente manera:

- si $c_0 = 0$, $s_i = e_i$ para todo $0 \leq i < 3$, $s_3 = 0$
- si $c_0 = 1$, $s_{i+1} = e_i$ para todo $0 \leq i < 3$, $s_0 = 0$

- a) Dibujar el diagrama lógico de un carry left shifter 3-4.
- b) Si las líneas de entrada y de salida codifican un número entero en notación sin signo, ¿qué significa matemáticamente el shift a la izquierda? ¿Y a la derecha?

Ejercicio 11

- a) Diseñar un *full adder* de 1 *bit*. La tabla de verdad del *full adder* es la siguiente:

A	B	$carry_{in}$	$suma$	$carry_{out}$
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

- b) Suponiendo que todas las compuertas elementales tienen el mismo retardo (*delay*) t , calcule el retardo total del circuito para producir todas sus señales de salida.

Ejercicio 12

- a) Diseñar un *full adder* de 4 *bits* combinando 4 *full adders* de 1 *bit*.
b) Suponiendo que los enteros se codifican con notación complemento a 2. Diseñar circuitos anexos que observen los siguientes *flags*:

Negative: $N = 1 \iff$ la salida representa un número negativo (complemento a 2)

Overflow: $V = 1 \iff$ el resultado no es representable (complemento a 2)

Carry: $C = 1 \iff$ la suma de la codificación binaria produjo acarreo

Zero: $Z = 1 \iff$ el resultado representa el número 0

- c) ¿Se puede usar el *mismo* circuito para sumar números enteros codificados en notación sin signo?
d) ¿Cómo se podría aprovechar este sumador para realizar restas tanto de números codificados en notación complemento a 2 como en notación sin signo?
e) Modificar la señal de *Carry* de tal modo que represente lo siguiente:
- Si la operación es suma: $C = 1 \iff$ la suma *bit a bit* produjo acarreo
 - Si la operación es resta: $C = 1 \iff$ la resta *bit a bit* produjo borrow (dame uno)
- f) Para comparar dos números A y B se realiza la operación $A - B$. Indicar los valores de los flags que caracterizan las siguientes condiciones.

condición	complemento a 2	notación sin signo
$A < B$		
$A \leq B$		
$A = B$		
$A \geq B$		
$A > B$		

Ejercicio 13

- a) Diseñar un componente con 4 entradas e_0, \dots, e_3 y 4 salidas s_0, \dots, s_3 tal que cada salida s_i valga $\overline{e_i}$.
b) Diseñar un componente con 4 entradas e_0, \dots, e_3 y 4 salidas s_0, \dots, s_3 que calcule el inverso aditivo del número codificado en complemento a 2 por la entrada.
c) Modificar el circuito anterior para que en una nueva salida indique si el número de la entrada no tiene un inverso aditivo representable con 4 *bits* en complemento a 2.