Organización del Computador 1

Práctica 2

1er cuatrimestre 2022

Índice

1. Ejercicio 1

Calculando las tablas de verdad podemos ver la equivalencia de las fórmulas booleanas.

1.a.

$p = (p.q) + (p.\overline{q})$

p	q	$(p.q)$	$+$	$(p.\overline{q})$
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	0

1.b.

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

x	y	z	$x.z$	$(x+y)$	$.$	$(x+\overline{y})$	$.$	$(\overline{x}+z)$
0	0	0	0	0	0	1	0	1
0	0	1	0	0	0	1	0	1
0	1	0	0	1	0	0	0	1
0	1	1	0	1	0	0	0	1
1	0	0	0	1	1	1	0	0
1	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	0	0
1	1	1	1	1	1	1	1	1

2. Ejercicio 2

Resolviendo mediante propiedades llegamos a 2 fórmulas que a priori no parecen ser equivalentes. Notar en la última línea que a la izquierda tenemos $\overline{y.z}$ mientras que a la derecha tenemos $\overline{y}.\overline{z}$.

$$
\begin{aligned}
x \oplus (y.z) &= (x \oplus y).(x \oplus z) \\
\overline{x}.y.z + x.\overline{y.z} &= (\overline{x}.y + x.\overline{y}).(\overline{x}.z + x.\overline{z}) \\
\overline{x}.y.z + x.\overline{y.z} &= (\overline{x}.y + x.\overline{y}).\overline{x}.z + (\overline{x}.y + x.\overline{y}).x.\overline{z} \\
\overline{x}.y.z + x.\overline{y.z} &= \overline{x}.y.\overline{x}.z + x.\overline{y}.\overline{x}.z + \overline{x}.y.x.\overline{z} + x.\overline{y}.x.\overline{z} \\
\overline{x}.y.z + x.\overline{y.z} &= \overline{x}.y.z + x.\overline{y}.\overline{z}
\end{aligned}
$$

Calculamos la tabla de verdad para verificar.

x	y	z	x	$\oplus$	$(y.z)$	$(x \oplus y)$	$.$	$(x \oplus z)$	
1	1	1	1	0	1	0	0	0	
1	1	0	1	1	0	0	0	1	✖
1	0	1	1	1	0	1	0	0	✖
1	0	0	1	1	0	1	1	1	
0	1	1	0	1	1	1	1	1	
0	1	0	0	0	0	1	0	0	
0	0	1	0	0	0	0	0	1	
0	0	0	0	0	0	0	0	0	

Conclusión: la propiedad planteada es falsa.

3. Ejercicio 3

3.a.

Verdadero, con el operador NAND ($p|q = \overline{p.q}$) podemos representar todas las funciones booleanas: AND, OR, NOT.

Recordemos la tabla de verdad del NAND.

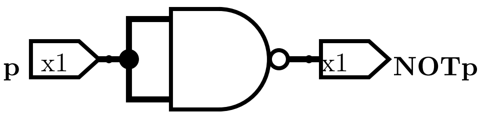
p	q	$p.q$	$p\|q = \overline{p.q}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

3.a.1. NOT

Utilizando la misma entrada 2 veces en un NAND podemos obtener un NOT.

$p|p = \overline{p}$

p	$p\|p$	$\overline{p}$
0	1	1
1	0	0

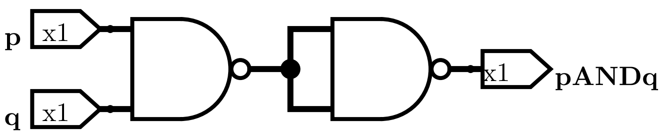


3.a.2. AND

Utilizando el NOT ya construido, podemos encadenarlo a la salida de un NAND para cancelar su negación y así obtener el resultado original del AND.

$(p|q)|(p|q) = p.q$

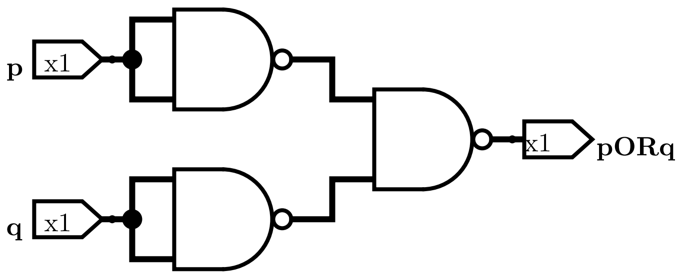
p	q	$p\|q$	$(p\|q)\|(p\|q)$	$p.q$
0	0	1	0	0
0	1	1	0	0
1	0	1	0	0
1	1	0	1	1



3.a.3. OR

$(p|p)|(q|q) = \overline{p}|\overline{q} = \overline{\overline{p}.\overline{q}} = \overline{\overline{p}} + \overline{\overline{q}} = p + q$

p	q	$(p\|p)$	$\|$	$(q\|q)$	$p+q$
0	0	1	0	1	0
0	1	1	1	0	1
1	0	0	1	1	1
1	1	0	1	0	1



3.b.

Verdadero, con el operador NOR ($p \downarrow q = \overline{p+q}$) podemos representar todas las funciones booleanas: AND, OR, NOT.

Recordemos la tabla de verdad del NOR.

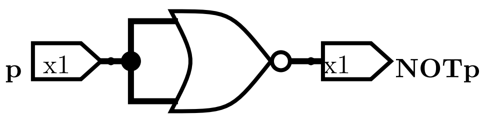
p	q	$p+q$	$p \downarrow q = \overline{p+q}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Observemos que los circuitos para armar las funciones booleanas utilizando solo la compuerta NOR son análogos a los utilizandos con la compuerta NAND.

3.b.1. NOT

$p \downarrow p = \overline{p}$

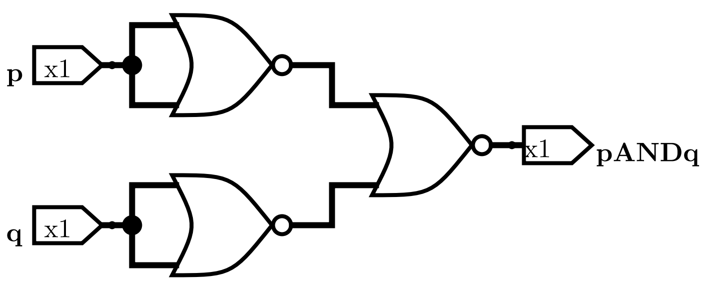
p	$p \downarrow p$	$\overline{p}$
0	1	1
1	0	0



3.b.2. AND

$(p \downarrow p) \downarrow (q \downarrow q) = \overline{p} \downarrow \overline{q} = \overline{\overline{p}+\overline{q}} = \overline{\overline{p}}.\overline{\overline{q}} = p.q$

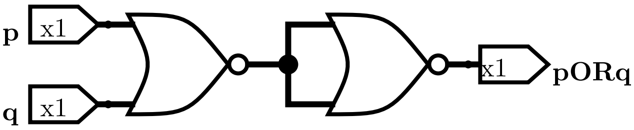
p	q	$(p \downarrow p)$	$\downarrow$	$(q \downarrow q)$	$p.q$
0	0	1	0	1	0
0	1	1	0	0	0
1	0	0	0	1	0
1	1	0	1	0	1



3.b.3. OR

$(p \downarrow q) \downarrow (p \downarrow q) = (\overline{p+q}) \downarrow (\overline{p+q}) = \overline{\overline{p+q}} = p+q$

p	q	$p \downarrow q$	$(p \downarrow q) \downarrow (p \downarrow q)$	$p + q$
0	0	1	0	0
0	1	0	1	1
1	0	0	1	1
1	1	0	1	1

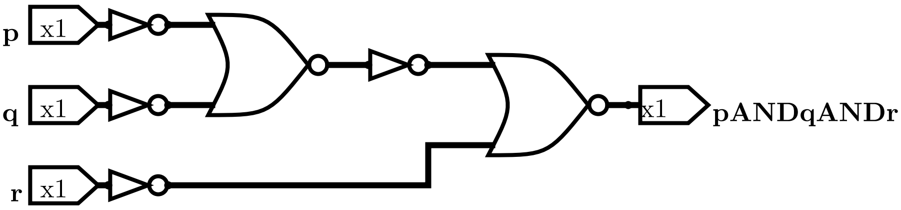


4. Ejercicio 4

Resuelto en el ejercicio 3.

5. Ejercicio 5

p	q	r	$p.q.r$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



6. Ejercicio 6

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

Nota: La tabla de verdad fue ordenada para una lectura más fácil.

6.a.

$F(A, B, C) = \overline{A}.B.C + A.\overline{B}.\overline{C} + A.\overline{B}.C + A.B.C$

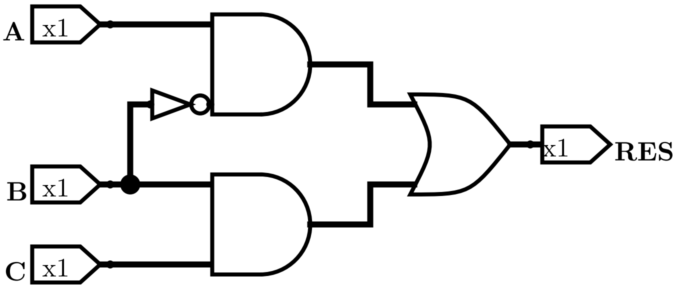
La implementación literal requiere un total de 15 compuertas: 3 OR, 8 AND y 4 NOT.

6.b.

$F(A,B,C) = \overline{A}.B.C + A.\overline{B}.\overline{C} + A.\overline{B}.C + A.B.C = B.C(A+\overline{A}) + A.\overline{B}(\overline{C}+C) = B.C + A.\overline{B}$

La implementación optimizada requiere un total de 4 compuertas: 1 OR, 2 AND y 1 NOT.

A	B	C	$B.C$	$+$	$A.\overline{B}$
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	0	0	0
1	1	1	1	1	0



7. Ejercicio 7: demultiplexor

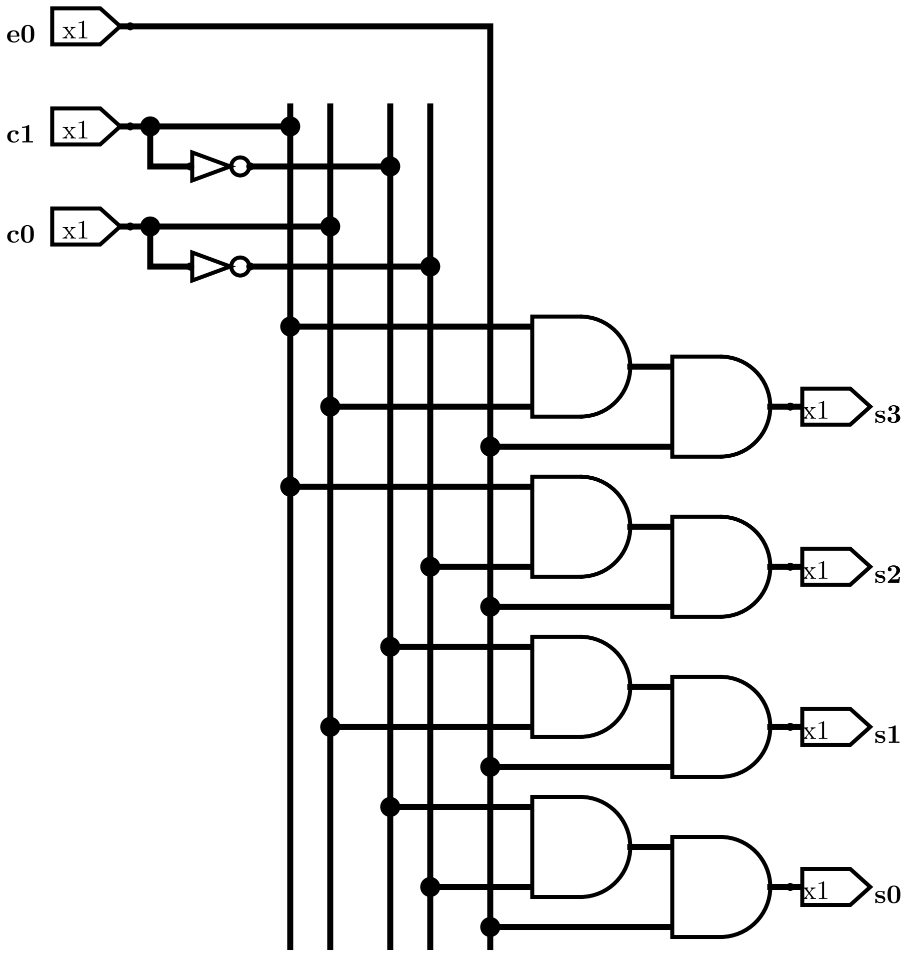
En la primer tabla de verdad planteamos una columna para los valores explícitos de e_0. En la segunda tabla eliminamos esta columna y en cambio planteamos e_0 como el valor de salida cuando las líneas de control computan a 1 para esa fila. Es decir, dada la fórmula booleana para cada salida, primero vemos si el AND entre las 2 líneas de control daría 1, y solo en ese caso el resultado final va a ser determinado por e_0. En todos los otros casos, como el AND entre las líneas de control da 0, es indistinto el valor de e_0 y la salida va a ser siempre 0.

c_1	c_0	e_0	s_3	s_2	s_1	s_0
0	0	0	0	0	0	0
0	1	0	0	0	0	0
1	0	0	0	0	0	0
1	1	0	0	0	0	0
0	0	1	0	0	0	1
0	1	1	0	0	1	0
1	0	1	0	1	0	0
1	1	1	1	0	0	0

c_1	c_0	s_3	s_2	s_1	s_0
0	0	0	0	0	e_0
0	1	0	0	e_0	0
1	0	0	e_0	0	0
1	1	e_0	0	0	0

Podemos plantear las siguientes fórmulas booleanas para cada una de las salidas.

$s_3 = c_1.c_0.e_0$ $\qquad$ $s_2 = c_1.\overline{c_0}.e_0$ $\qquad$ $s_1 = \overline{c_1}.c_0.e_0$ $\qquad$ $s_0 = \overline{c_1}.\overline{c_0}.e_0$



8. Ejercicio 8: codificador

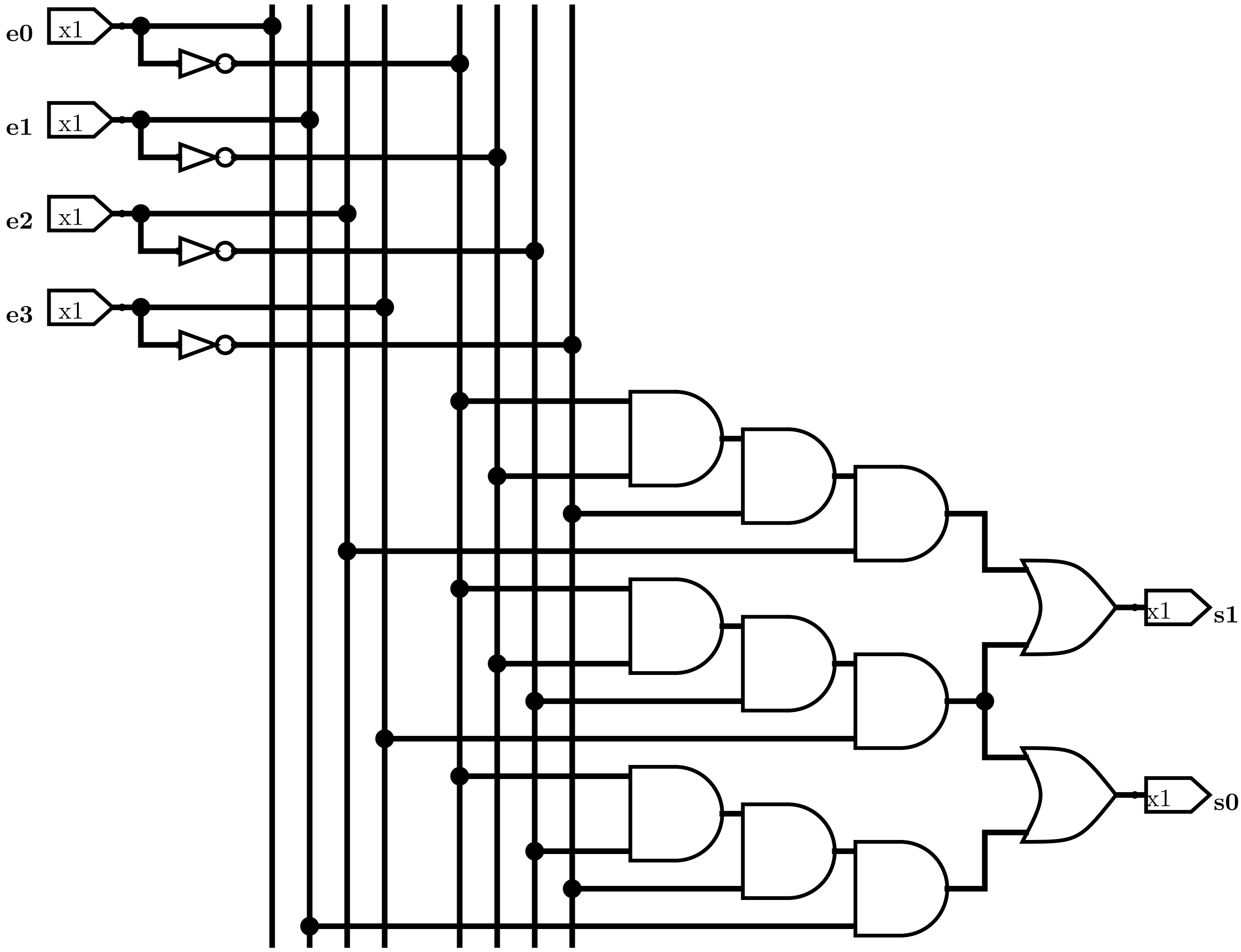
8.a.

e_0	e_1	e_2	e_3	s_1	s_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

$s_1 = \overline{e_0}.\overline{e_1}.e_2.\overline{e_3} + \overline{e_0}.\overline{e_1}.\overline{e_2}.e_3$

$s_0 = \overline{e_0}.e_1.\overline{e_2}.\overline{e_3} + \overline{e_0}.\overline{e_1}.\overline{e_2}.e_3$

Nota: Consideramos s_1 como el bit más significativo de la representación binaria del número i.



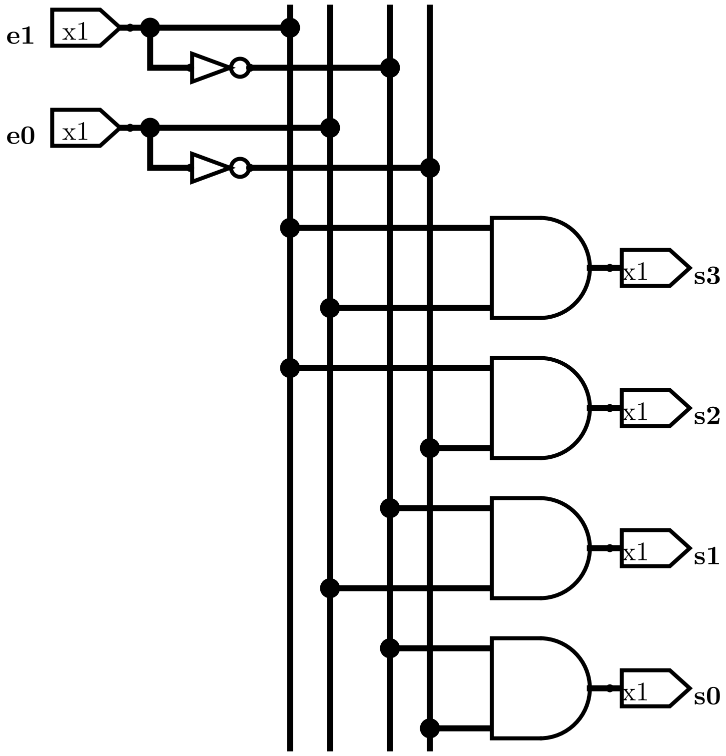
8.b.

9. Ejercicio 9: decodificador

9.a.

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

$s_3 = e_1.e_0$ $\quad$ $s_2 = e_1.\overline{e_0}$ $\quad$ $s_1 = \overline{e_1}.e_0$ $\quad$ $s_0 = \overline{e_1}.\overline{e_0}$



9.b.