

Introducción

Características – Sistemas Digitales vs. Analógicos.

Digitales	Analógicos
Utiliza sólo dos estados diferentes.	Utiliza una cantidad infinita de valores dentro de un rango de valores continuo.
Dados los mismos valores, en la misma secuencia, produce los mismos resultados.	El resultado puede variar con la temperatura, tensión de alimentación, etc.
Diseño más sencillo. Utiliza lógica. "Oculta" la parte analógica.	Diseño más complejo. Más matemática involucrada.
Necesidad de adaptar y convertir entre niveles digitales y analógicos. (Eventual pérdida de información)	No necesitan conversiones.

Características – Sistemas Digitales.

Utiliza dos estados diferentes.

Verdadero - Falso

} Lógica tradicional.

Cerrado - Abierto

Encendido - Apagado

} Lógica de relés.

Blanco - Negro

Iluminado - Oscuro

} Estados en una fibra óptica.

Alto - Bajo

Uno - Cero

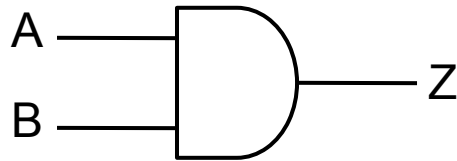
} Sistemas digitales modernos
Alto (High, H) y
Bajo (Low, L).

etc.

Circuitos Lógicos

Elementos lógicos básicos

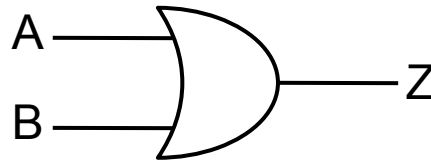
AND - “Y”



A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

$$Z = A \cdot B$$

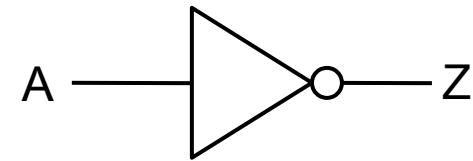
OR - “O”



A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

$$Z = A + B$$

NOT - “NO”
 (“Inversor”)

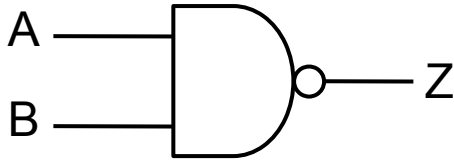


A	Z
0	1
1	0

$$Z = \bar{A}$$

Elementos lógicos derivados de los anteriores

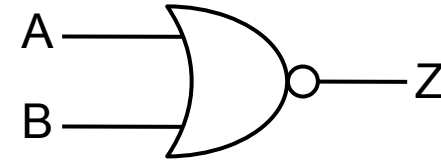
NAND - "NO-Y"



A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

$$Z = \overline{A \cdot B}$$

NOR - "NO-O"

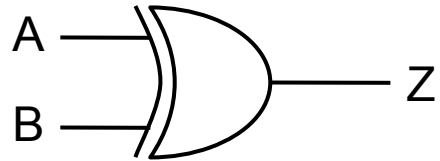


A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

$$Z = \overline{A + B}$$

Elementos lógicos derivados de los anteriores

XOR – “O-Exclusivo”

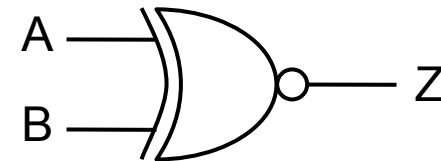


A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

$$Z = A \oplus B$$

$$Z = \bar{A} \cdot B + A \cdot \bar{B}$$

XNOR – “No O-Exclusivo”

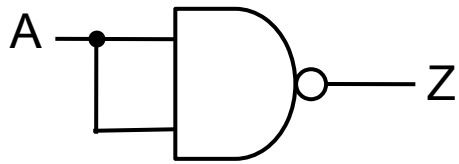


A	B	Z
0	0	1
0	1	0
1	0	0
1	1	1

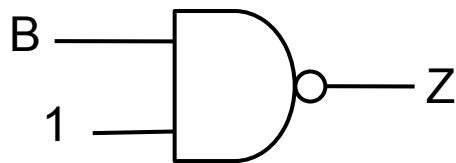
$$Z = \overline{A \oplus B}$$

Elementos lógicos derivados de los anteriores

Inversor con NAND

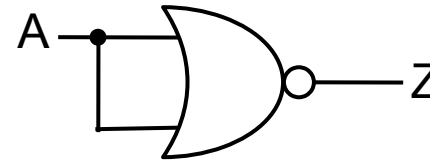


A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

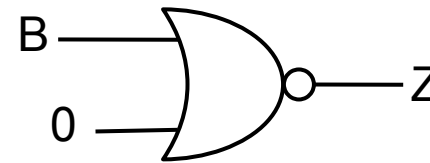


A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

Inversor con NOR



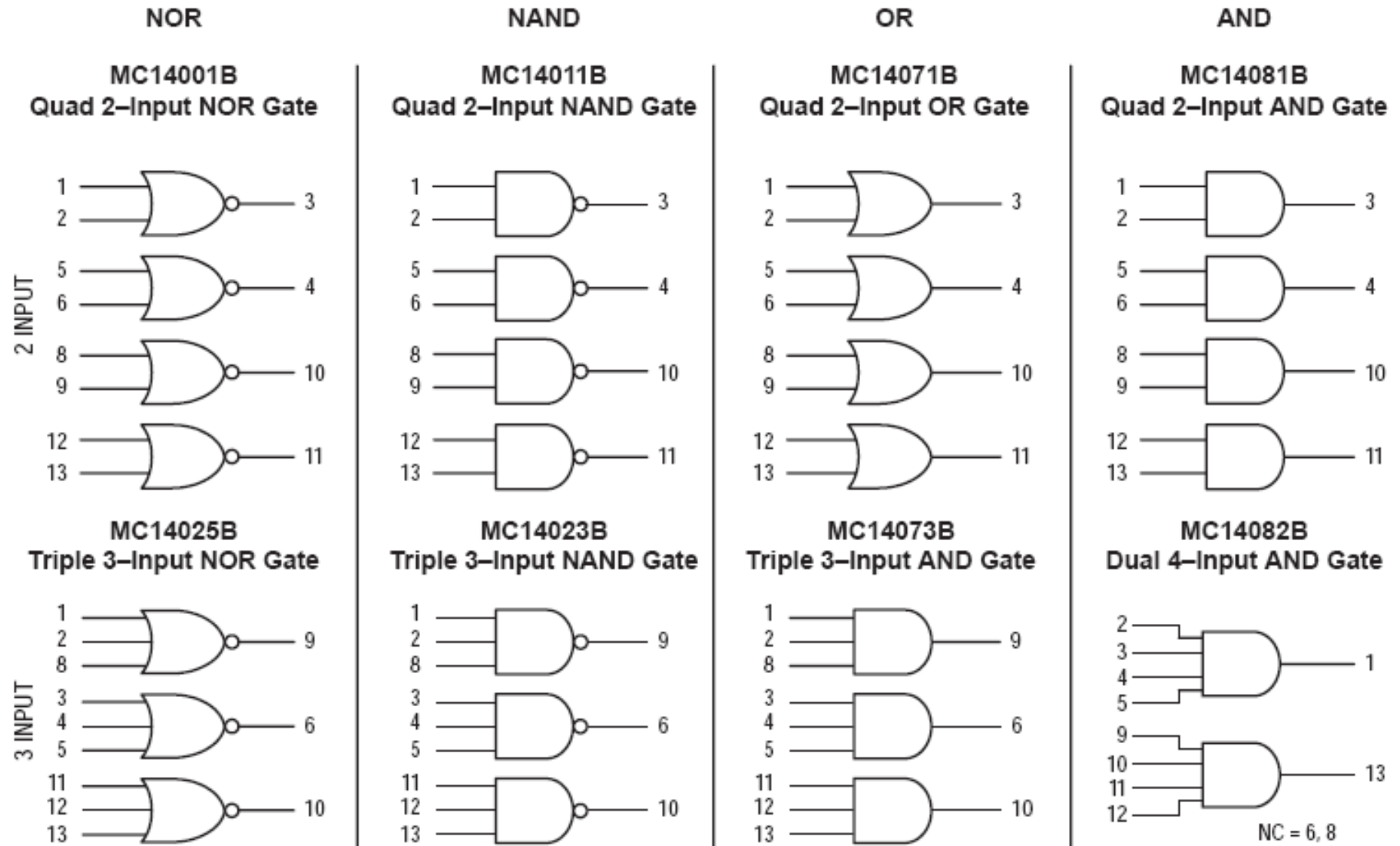
A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0



A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

Circuitos Integrados con compuertas.

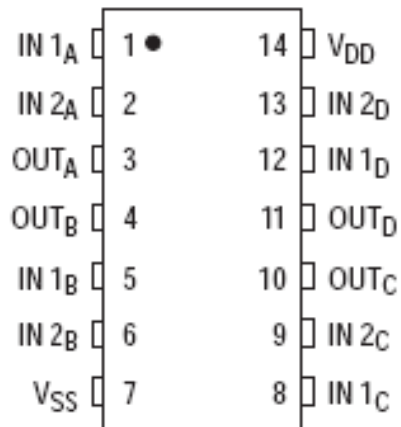
Códigos y distribución de pines de integrados con compuertas. Familia 4000B



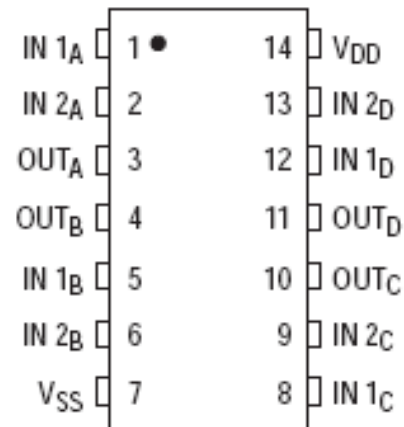
Circuitos Integrados con compuertas.

Códigos y distribución de pines de integrados con compuertas. Familia 4000B

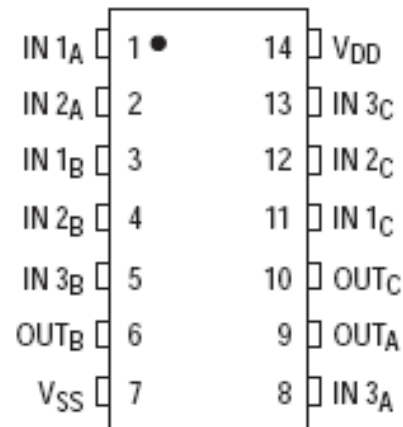
MC14001B
Quad 2-Input NOR Gate



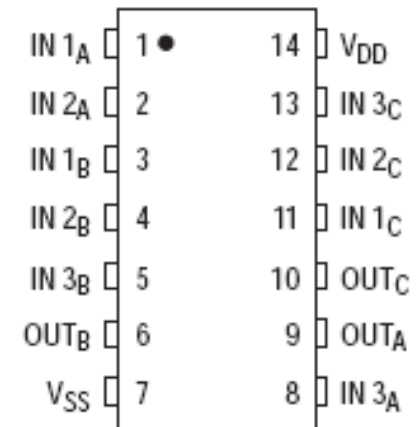
MC14011B
Quad 2-Input NAND Gate



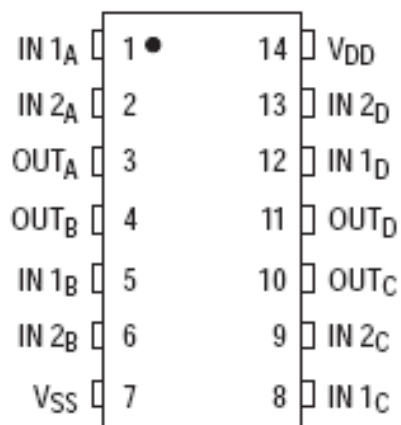
MC14023B
Triple 3-Input NAND Gate



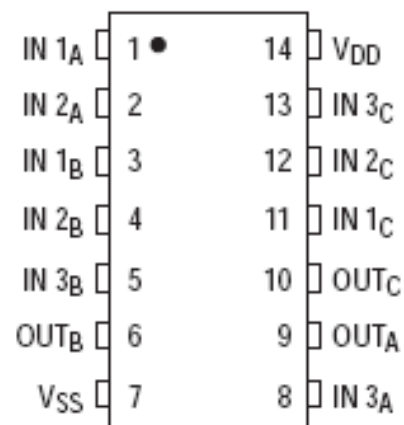
MC14025B
Triple 3-Input NOR Gate



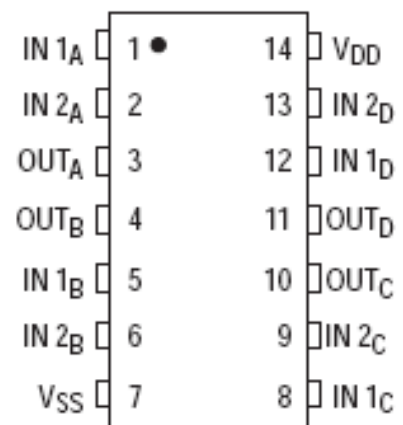
MC14071B
Quad 2-Input OR Gate



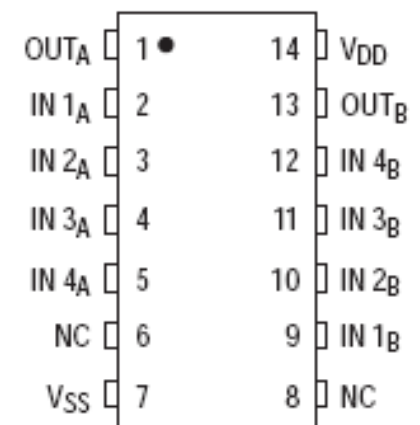
MC14073B
Triple 3-Input AND Gate



MC14081B
Quad 2-Input AND Gate



MC14082B
Dual 4-Input AND Gate



NC = NO CONNECTION

Características importantes de los circuitos integrados digitales.

Familia 4000B

Characteristic	Symbol	V _{DD} Vdc	- 55°C		25°C			125°C		Unit
			Min	Max	Min	Typ ^(3.)	Max	Min	Max	
Output Voltage V _{in} = V _{DD} or 0 V _{in} = 0 or V _{DD}	V _{OL}	5.0	—	0.05	—	0	0.05	—	0.05	Vdc
		10	—	0.05	—	0	0.05	—	0.05	
		15	—	0.05	—	0	0.05	—	0.05	
	V _{OH}	5.0	4.95	—	4.95	5.0	—	4.95	—	Vdc
		10	9.95	—	9.95	10	—	9.95	—	
		15	14.95	—	14.95	15	—	14.95	—	
Input Voltage (V _O = 4.5 or 0.5 Vdc) (V _O = 9.0 or 1.0 Vdc) (V _O = 13.5 or 1.5 Vdc) (V _O = 0.5 or 4.5 Vdc) (V _O = 1.0 or 9.0 Vdc) (V _O = 1.5 or 13.5 Vdc)	V _{IL}	5.0	—	1.5	—	2.25	1.5	—	1.5	Vdc
		10	—	3.0	—	4.50	3.0	—	3.0	
		15	—	4.0	—	6.75	4.0	—	4.0	
	V _{IH}	5.0	3.5	—	3.5	2.75	—	3.5	—	Vdc
		10	7.0	—	7.0	5.50	—	7.0	—	
		15	11	—	11	8.25	—	11	—	
Output Drive Current (V _{OH} = 2.5 Vdc) (V _{OH} = 4.6 Vdc) (V _{OH} = 9.5 Vdc) (V _{OH} = 13.5 Vdc) (V _{OL} = 0.4 Vdc) (V _{OL} = 0.5 Vdc) (V _{OL} = 1.5 Vdc)	I _{OH}	5.0	- 3.0	—	- 2.4	- 4.2	—	- 1.7	—	mAdc
		5.0	- 0.64	—	- 0.51	- 0.88	—	- 0.36	—	
		10	- 1.6	—	- 1.3	- 2.25	—	- 0.9	—	
		15	- 4.2	—	- 3.4	- 8.8	—	- 2.4	—	
	I _{OL}	5.0	0.64	—	0.51	0.88	—	0.36	—	mAdc
		10	1.6	—	1.3	2.25	—	0.9	—	
		15	4.2	—	3.4	8.8	—	2.4	—	
Input Current	I _{in}	15	—	± 0.1	—	± 0.00001	± 0.1	—	± 1.0	µAdc
Input Capacitance (V _{in} = 0)	C _{in}	—	—	—	—	5.0	7.5	—	—	pF

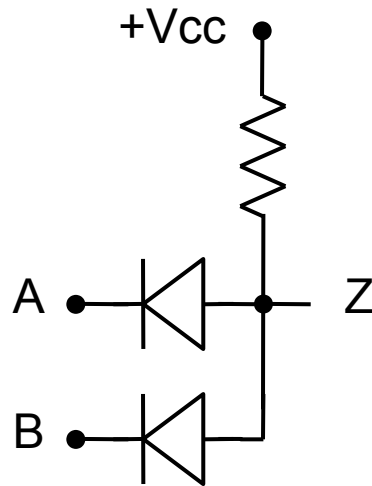
Características importantes de los circuitos integrados digitales.

Familia 74HC

Symbol	Parameter	Condition	V _{CC} V	Guaranteed Limit			Unit
				-55 to 25°C	≤85°C	≤125°C	
V _{IH}	Minimum High-Level Input Voltage	V _{out} = 0.1V or V _{CC} - 0.1V I _{out} ≤ 20μA	2.0	1.50	1.50	1.50	V
			3.0	2.10	2.10	2.10	
			4.5	3.15	3.15	3.15	
			6.0	4.20	4.20	4.20	
V _{IL}	Maximum Low-Level Input Voltage	V _{out} = 0.1V or V _{CC} - 0.1V I _{out} ≤ 20μA	2.0	0.50	0.50	0.50	V
			3.0	0.90	0.90	0.90	
			4.5	1.35	1.35	1.35	
			6.0	1.80	1.80	1.80	
V _{OH}	Minimum High-Level Output Voltage	V _{in} = V _{IH} or V _{IL} I _{out} ≤ 20μA	2.0	1.9	1.9	1.9	V
			4.5	4.4	4.4	4.4	
			6.0	5.9	5.9	5.9	
		V _{in} = V _{IH} or V _{IL} I _{out} ≤ 2.4mA I _{out} ≤ 4.0mA I _{out} ≤ 5.2mA	3.0	2.48	2.34	2.20	
			4.5	3.98	3.84	3.70	
			6.0	5.48	5.34	5.20	
V _{OL}	Maximum Low-Level Output Voltage	V _{in} = V _{IH} or V _{IL} I _{out} ≤ 20μA	2.0	0.1	0.1	0.1	V
			4.5	0.1	0.1	0.1	
			6.0	0.1	0.1	0.1	
		V _{in} = V _{IH} or V _{IL} I _{out} ≤ 2.4mA I _{out} ≤ 4.0mA I _{out} ≤ 5.2mA	3.0	0.26	0.33	0.40	
			4.5	0.26	0.33	0.40	
			6.0	0.26	0.33	0.40	
I _{in}	Maximum Input Leakage Current	V _{in} = V _{CC} or GND	6.0	±0.1	±1.0	±1.0	μA
I _{CC}	Maximum Quiescent Supply Current (per Package)	V _{in} = V _{CC} or GND I _{out} = 0μA	6.0	1.0	10	40	μA

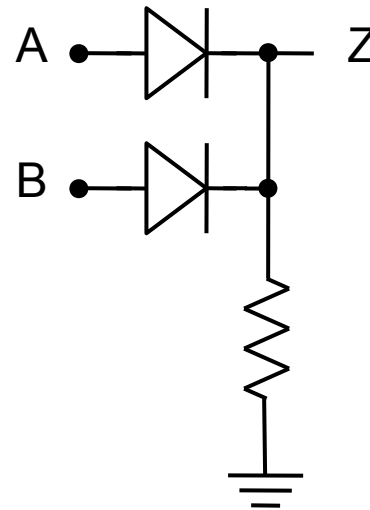
Elementos lógicos básicos con componentes discretos.

AND - "Y"



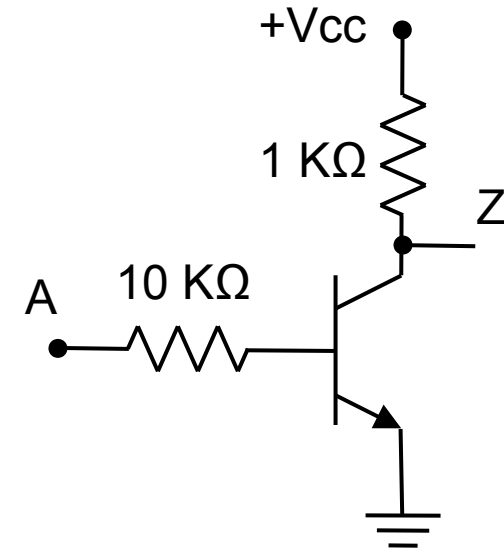
Si A o B = V_{cc} , ningún diodo conduce. Por lo tanto, no circula I por el resistor* y $Z = V_{cc}$.
- Si A y/o B es 0V, uno o ambos diodos conducen haciendo que V en Z $\approx 0,7V$

OR - "O"



Una tensión V_e en A y/o B producen una tensión en Z $\approx V_e - 0,7V$.

NOT - "NO"
("Inversor")



Si $V_A = V_{cc}$, el transistor satura, por lo que $V_Z = V_{CEsat} \approx 0,2V$.
Si $V_A \approx 0V$, el transistor está al corte, por lo que $V_Z \approx V_{cc}$, ya que no circula I por el resistor* de $1K \Omega$

* La I en el resistor depende del circuito conectado después de Z

Construcción de circuitos lógicos a partir de una tabla de verdad.

A	B	F
0	0	0
0	1	1
1	0	0
1	1	1

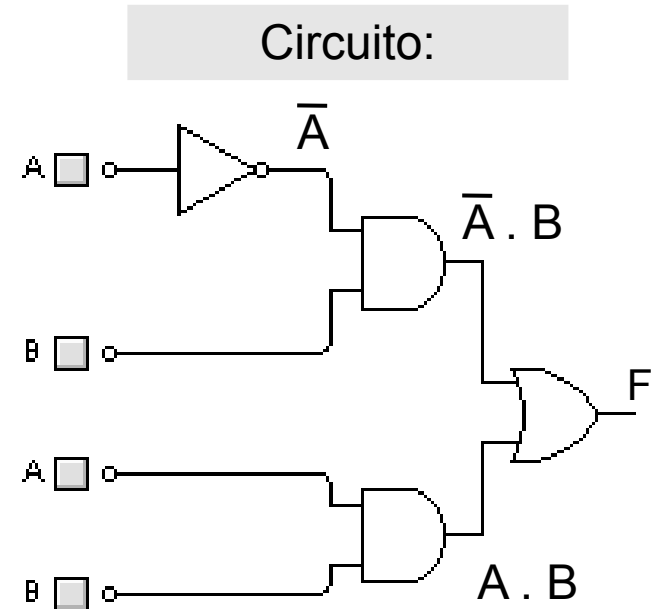
Se observa que
 $F=1$ cuando....

$A=0$ Y $B=1$

No $A = \bar{A}$

O

$A=1$ Y $B=1$



Expresión lógica:

$$F = \bar{A} . B + A . B$$



Indica que, para que $F = 1$,
A tiene que ser 0 (**NO** A) Y B tiene que ser 1

O

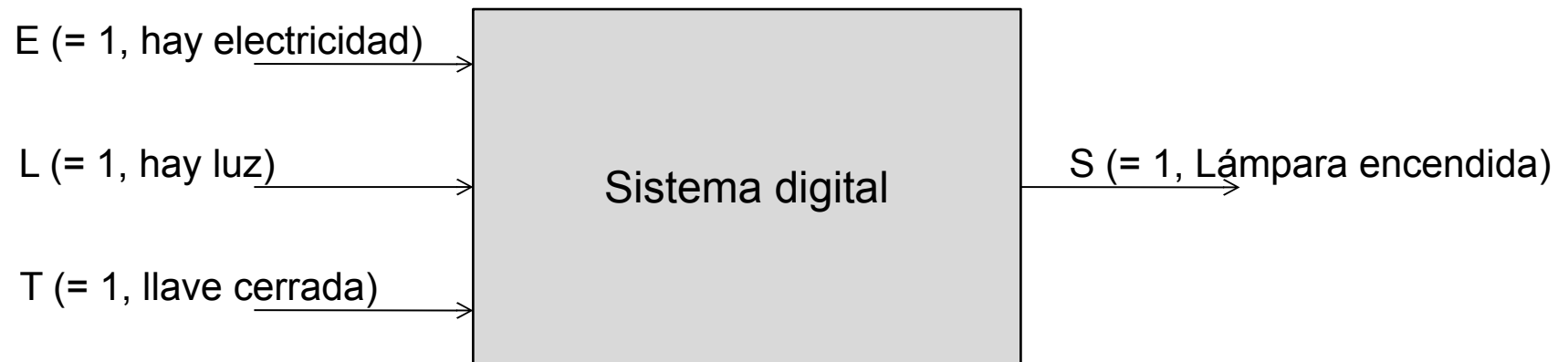
A Y B tienen que ser 1

Análisis lógico de problemas - Tablas de verdad.

Realizar un circuito que encienda una lámpara si:

- No hay electricidad y no hay luz y una llave está cerrada.
 - Hay electricidad, hay luz, y la llave está abierta.
 - Hay electricidad, hay luz, y la llave está cerrada.

1º Identificar entradas y salidas, asignar estados lógicos a las entradas:



Análisis lógico de problemas - Tablas de verdad.

2º Armar una tabla con todos las combinaciones de entrada posibles y las salidas correspondientes.

Con 3 entradas que pueden adoptar 2 valores, la cantidad de combinaciones es $2^3 = 8$.

En general a n entradas de 2 estados, le corresponden 2^n combinaciones.

- No hay electricidad y no hay luz y unallave está cerrada.

- Hay electricidad, hay luz, y la llave está abierta.

- Hay electricidad, hay luz, y la llave está cerrada.

E	L	T	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Encender lámpara

Construcción de circuitos lógicos a partir de una tabla de verdad.

3º Escribir las expresiones lógicas cuando la salida es 1*; plantear el circuito.

E	L	T	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Se observa que
S=1 cuando....

E=0 Y L=0 Y T=1

No A = $\bar{A}$

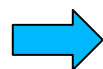
0

E=1 Y L=1 Y T=0

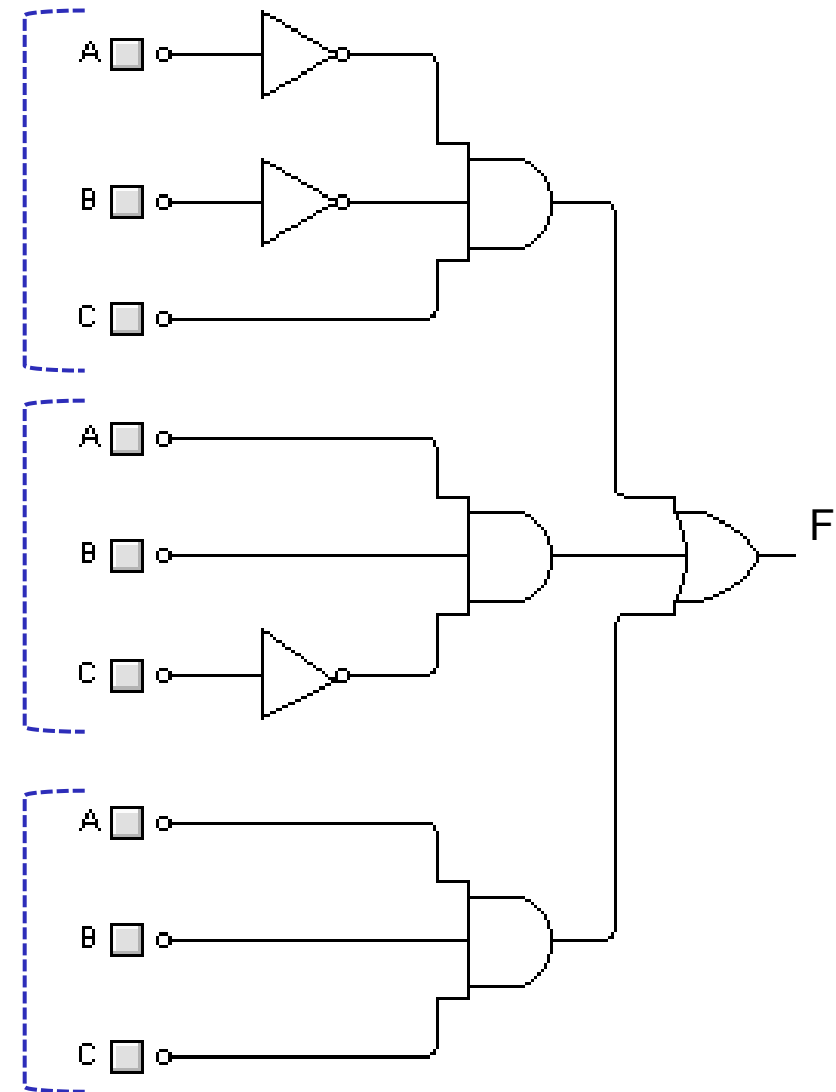
0

E=1 Y L=1 Y T=1

Expresión
lógica:



$$S = \bar{E} \cdot \bar{L} \cdot T + E \cdot L \cdot \bar{T} + E \cdot L \cdot T$$



(*En ocasiones conviene usar las expresiones correspondientes para cuando la salida es "0")

Switching Algebra - Álgebra de conmutación (Álgebra de Boole)

Algunos teoremas útiles para simplificar y trabajar con circuitos lógicos.

Función AND

$$(1) \quad 0 \cdot 0 = 0$$

$$(2) \quad 0 \cdot 1 = 0$$

$$(3) \quad 1 \cdot 1 = 1$$

$$(4) \quad X \cdot 0 = 0$$

$$(5) \quad X \cdot 1 = X$$

$$(6) \quad X \cdot X = X$$

$$(7) \quad X \cdot \bar{X} = 0$$

Función OR

$$(8) \quad 0 + 0 = 0$$

$$(9) \quad 0 + 1 = 1$$

$$(10) \quad 1 + 1 = 1$$

$$(11) \quad X + 0 = X$$

$$(12) \quad X + 1 = 1$$

$$(13) \quad X + X = X$$

$$(14) \quad X + \bar{X} = 1$$

Nota: La operación NOT puede escribirse de varias formas: $\bar{A}$, $\sim A$, A' , $/A$

Switching Algebra - Álgebra de conmutación (Álgebra de Boole)

Función NOT

$$(15) \quad \overline{0} = 1$$

$$(16) \quad \underline{\underline{1}} = 0$$

$$(17) \quad \overline{\overline{X}} = X$$

Ley Conmutativa

$$(18) \quad X \cdot Y = Y \cdot X$$

$$(19) \quad X + Y = Y + X$$

Ley Distributiva

$$(20) \quad X \cdot (Y + Z) = X \cdot Y + X \cdot Z$$

$$(21) \quad X + Y \cdot Z = (X + Y) \cdot (X + Z)$$

Ley Asociativa

$$(22) \quad X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z$$

$$(23) \quad X + (Y + Z) = (X + Y) + Z$$

Leyes de Absorción

$$(24) \quad X + X \cdot Y = X$$

$$(25) \quad X \cdot (X + Y) = X$$

Leyes de De Morgan

$$(26) \quad \overline{X + Y} = \overline{X} \cdot \overline{Y}$$

$$(27) \quad \overline{X \cdot Y} = \overline{X} + \overline{Y}$$

Además...

$$(28) \quad X + \overline{X} \cdot Y = X + Y$$

$$(29) \quad X \cdot (\overline{X} + Y) = X \cdot Y$$

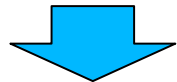
Construcción de circuitos lógicos a partir de una tabla de verdad.

A veces se puede simplificar la tabla o las expresiones, antes de armar el circuito...

Inspeccionando
la tabla...

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

En estos dos
renglones, $F = 1$,
independientemente
del valor de C

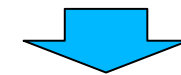


Queda...

$$F = \bar{C} \cdot \bar{B} \cdot A + C \cdot B$$

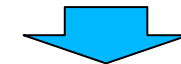
Con álgebra...

$$F = \bar{C} \cdot \bar{B} \cdot A + C \cdot B \cdot \bar{A} + C \cdot B \cdot A$$



Prop. Distributiva.

$$F = \bar{C} \cdot \bar{B} \cdot A + C \cdot B \cdot (\bar{A} + A)$$



$$A + \bar{A} = 1 + 0 = 1$$

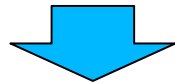
$$F = \bar{C} \cdot \bar{B} \cdot A + C \cdot B$$

El estado "X"

Cuando "da igual si es uno o cero", una condición "no importa", se utiliza una "X"

En la tabla del ejemplo...

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



En estos dos renglones, $F = 1$, independientemente del valor de C

Se podría escribir abreviada:

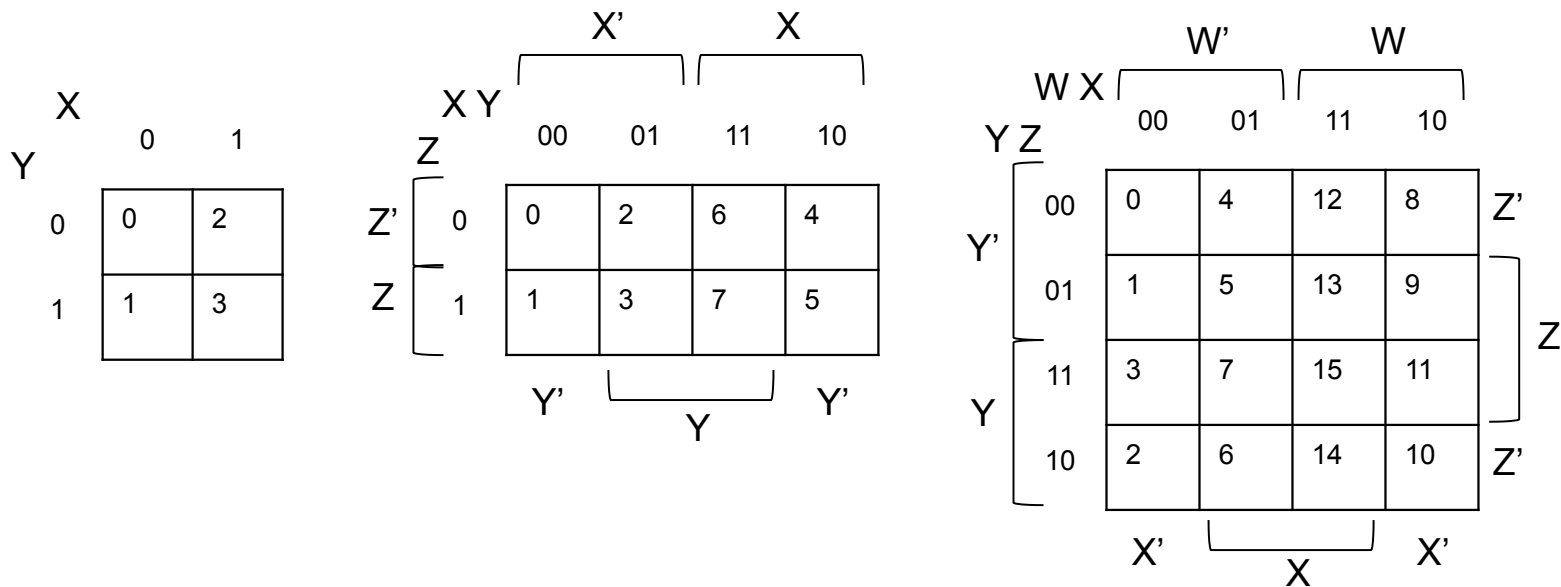
C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	X	1

Si $C=1$ y $B=1$, no importa el valor de A, $F=1$

Mapas de Karnaugh

Otro método para simplificar expresiones lógicas.

Es una forma gráfica de representar tablas de verdad.



Tablas de 2, 3 y 4 variables (pueden ser de más...)

- A cada renglón de la tabla de verdad, le corresponde un casillero.
- En el casillero se coloca el resultado (columna “F” en el ejemplo) de la función.

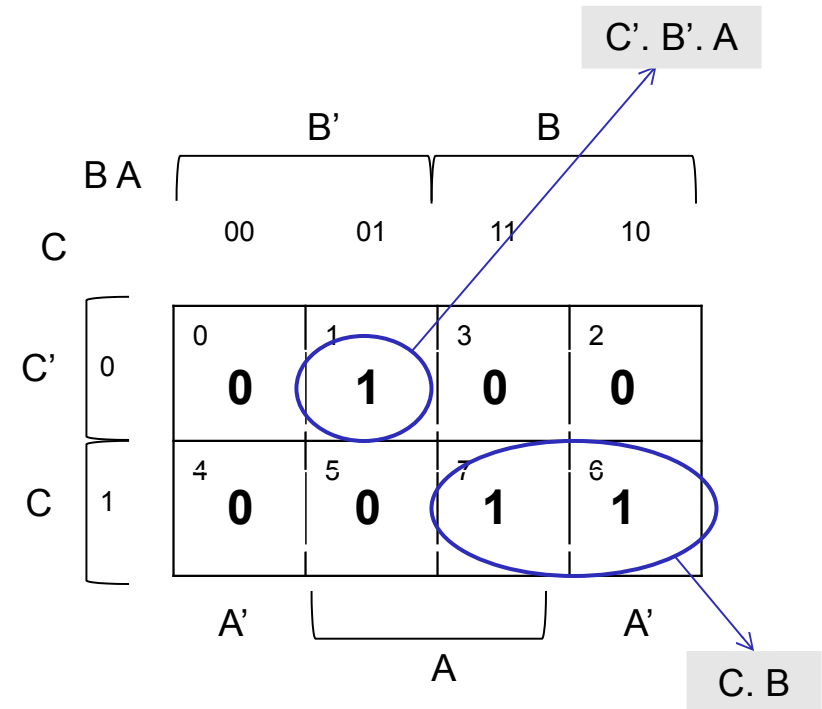
Mapas de Karnaugh

Ejemplo de simplificación con mapa de Karnaugh

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



1. Completar el mapa según la tabla de verdad....



2. Agrupar los "1" adyacentes, en grupos lo mayor posible, con una cantidad par de unos.
(Si hay "X", se pueden usar como unos o ceros)

3. Escribir la expresión de cada grupo...

4. La expresión final surge de la suma de las expresiones de los grupos:



$$F = C' . B' . A + C . B$$

Sistemas de numeración

Sistemas de numeración. Notación posicional.

En un sistema de numeración que utiliza **notación posicional**, cada dígito posee un valor diferente que **depende de su** posición relativa, de su **peso**.

El valor de cada dígito está dado por: $d \cdot b^{p-1}$ donde **d** es el dígito, **b** es la base del sistema de numeración (que determina la cantidad de símbolos disponibles) y **p** es la posición del dígito.

El valor del número resulta la suma del valor de sus dígitos.

3486, en sistema decimal (base = 10), es el resultado de hacer:

$$3 \cdot 10^3 + 4 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0 = 3 \cdot 1000 + 4 \cdot 100 + 8 \cdot 10 + 6 \cdot 1$$

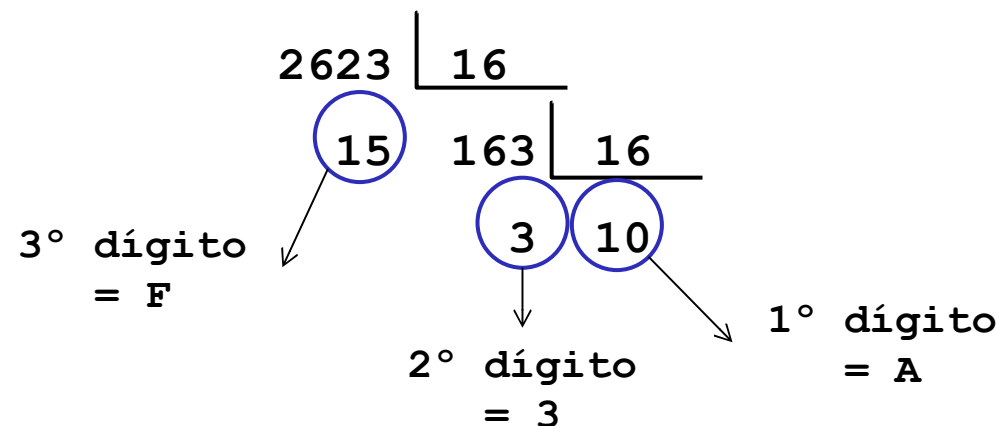
Este método permite obtener el valor en base 10 de un número expresado en cualquier base.

El valor máximo que se puede representar con n dígitos está dado por: $b^n - 1$

Sistemas de numeración. Conversión de base 10 a base n.

Para convertir de decimal a una base n, se divide el número por n, hasta que no pueda obtenerse un entero, y se toma el resultado de la última división y los restos en orden invertido.

Ej.: Convertir **2623** en decimal a base 16:



El número decimal **2623** en hexadecimal es **A3F**

Este sistema se puede utilizar para cualquier base. Para convertir a binario, por ejemplo, se divide por 2.

Sistema hexadecimal.

El sistema **hexadecimal** tiene **16 símbolos** para representar números:
de 0 a 9 y de A a F (la A = 10, la B = 11, etc.)

A3F en sistema hexadecimal será:

$$A \cdot 16^2 + 3 \cdot 16^1 + F \cdot 16^0 = 10 \cdot 256 + 3 \cdot 16 + 15 = \mathbf{2623} \text{ en decimal}$$

Cada dígito hexadecimal se puede escribir individualmente en binario y viceversa,
posibilitando una conversión rápida entre estas bases.

Ej.: C 9 en hexadecimal es...

Ej.: 1010 1000 en binario es...

1100 1001 en binario.

A 8 en hexadecimal.

4 bits se pueden representar con un dígito “hexa”.

Cantidad de bits	Cantidad de dígitos hexa	Máximo valor
8	2	256
16	4	65.536
32	8	4.294.967.296

Sistema binario.

El sistema **binario** tiene base 2, o sea, **dos símbolos**
para representar números: **0 y 1**

1011 en sistema binario será:

$$1.2^3 + 0.2^2 + 1.2^1 + 1.2^0 = 1.8 + 0.4 + 1.2 + 1.1 = \mathbf{11} \text{ en decimal}$$

Cada dígito en sistema binario se llama “**bit**”

Ej. 10110 es un número de 5 bits.

Cantidad de bits	Máximo valor (decimal)
4	16
7	128
8	256
10	1024 (1 K)
16	65.536 (64 K = 64 * 1024)
20	1.048.576 (1 M = 1024 * 1024)
30	1.073.741.824 (1 G = 1024 * 1024 * 1024)
32	4.294.967.296 (4 G)

Algunos números en decimal, hexadecimal y binario.

Decimal	Hexa (1 dígito)	Binario (4 bits)	Binario (3 bits)
0	0	0000	000
1	1	0001	001
2	2	0010	010
3	3	0011	011
4	4	0100	100
5	5	0101	101
6	6	0110	110
7	7	0111	111
8	8	1000	–
9	9	1001	–
10	A	1010	–
11	B	1011	–
12	C	1100	–
13	D	1101	–
14	E	1110	–
15	F	1111	–

Otros códigos binarios

Otros códigos binarios

Hay más códigos binarios, además del “**natural**” que es el que se trató en la sección anterior.

- Códigos BCD (Binary-Coded Decimal, Decimal codificado en binario)

Representan los dígitos decimales en binario.

En BCD natural será: 0111 1000 0101

Diagram showing the decimal digits 7, 8, and 5 with arrows pointing to their respective 4-bit BCD natural representations: 0111 for 7, 1000 for 8, and 0101 for 5.

- Código Gray: Sólo cambia **un bit** entre combinaciones sucesivas.

En natural

¿ 1 1 1 ? - ¿ 1 1 0 ?
¿ 1 0 1 ? - ¿ 1 0 0 ?

1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0
7	6	5	4	3	2	1	0

En Gray

¿ 1 1 1 ? - ¿ 1 0 1 ?

1	1	1	1	0	0	0	0
0	0	1	1	1	1	0	0
0	1	1	0	0	1	1	0
7	6	5	4	3	2	1	0

Otros códigos binarios

Decimal	Binario natural (4b)	Gray (4b)	BCD Natural	BCD Exceso 3	BCD Aiken
0	0000	0000	0000	0011	0000
1	0001	0001	0001	0100	0001
2	0010	0011	0010	0101	0010
3	0011	0010	0011	0110	0011
4	0100	0110	0100	0111	0100
5	0101	0111	0101	1000	1011
6	0110	0101	0110	1001	1100
7	0111	0100	0111	1010	1101
8	1000	1100	1000	1011	1110
9	1001	1101	1001	1100	1111
10	1010	1111	-	-	-
11	1011	1110	-	-	-
12	1100	1010	-	-	-
13	1101	1011	-	-	-
14	1110	1001	-	-	-
15	1111	1000	-	-	-

Aritmética binaria

Suma de números binarios.

Se utiliza el método de suma y resta “a mano” usada con los números decimales.

Donde:

$0_2 + 0_2 = 0_2$
$0_2 + 1_2 = 1_2$
$1_2 + 0_2 = 1_2$
$1_2 + 1_2 = 10_2$ (2 en decimal)
$1_2 + 1_2 + 1_2 = 11_2$ (3 en decimal)

Entonces:

$$\begin{array}{r} + \quad 0 \ 0 \ 1 \ 0 \quad 2 \\ \quad 1 \ 0 \ 0 \ 1 \quad 9 \\ \hline 1 \ 0 \ 1 \ 1 \quad 11 \end{array}$$

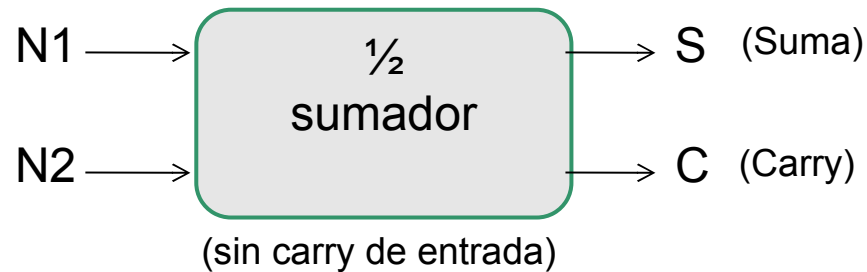
$$\begin{array}{r} \quad \quad 1 \\ + \quad 1 \ 0 \ 1 \ 1 \quad 11 \\ \quad 0 \ 0 \ 1 \ 0 \quad 2 \\ \hline 1 \ 1 \ 0 \ 1 \quad 13 \end{array}$$

$$\begin{array}{r} \quad \quad 1 \quad 1 \ 1 \\ + \quad 1 \ 0 \ 1 \ 1 \quad 11 \\ \quad 1 \ 0 \ 1 \ 1 \quad 11 \\ \hline 1 \ 0 \ 1 \ 1 \ 0 \quad 22 \end{array}$$

Acarreo / Carry (C)

En el último ejemplo, 22 no se puede representar en 4 bits, se produce “carry”.
Si este bit no se toma en cuenta, el resultado es incorrecto: 0110 (6)

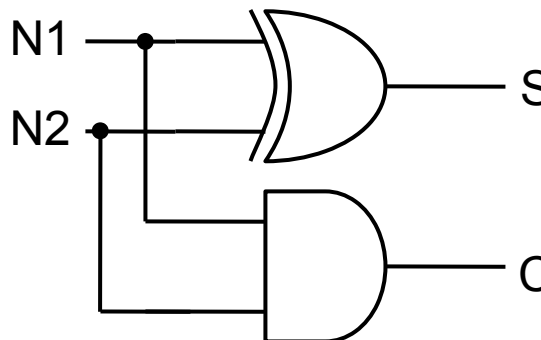
Suma de números binarios. Circuito sumador de 1 bit.



N1	N2	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = N1 \oplus N2$$

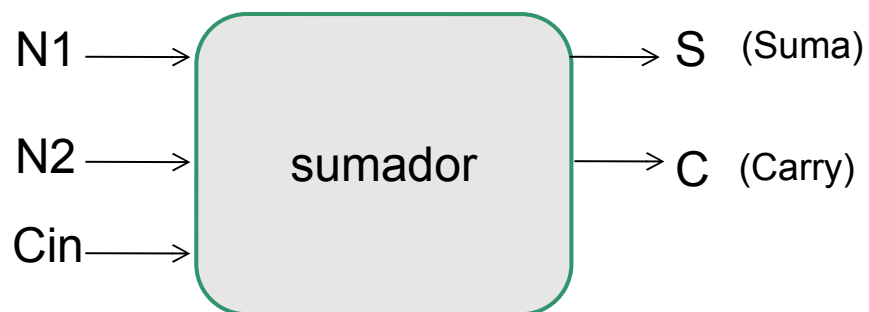
$$C = N1 \cdot N2$$



Este circuito no tiene en cuenta un eventual Carry en la entrada.

Por eso se le llama **medio sumador**.

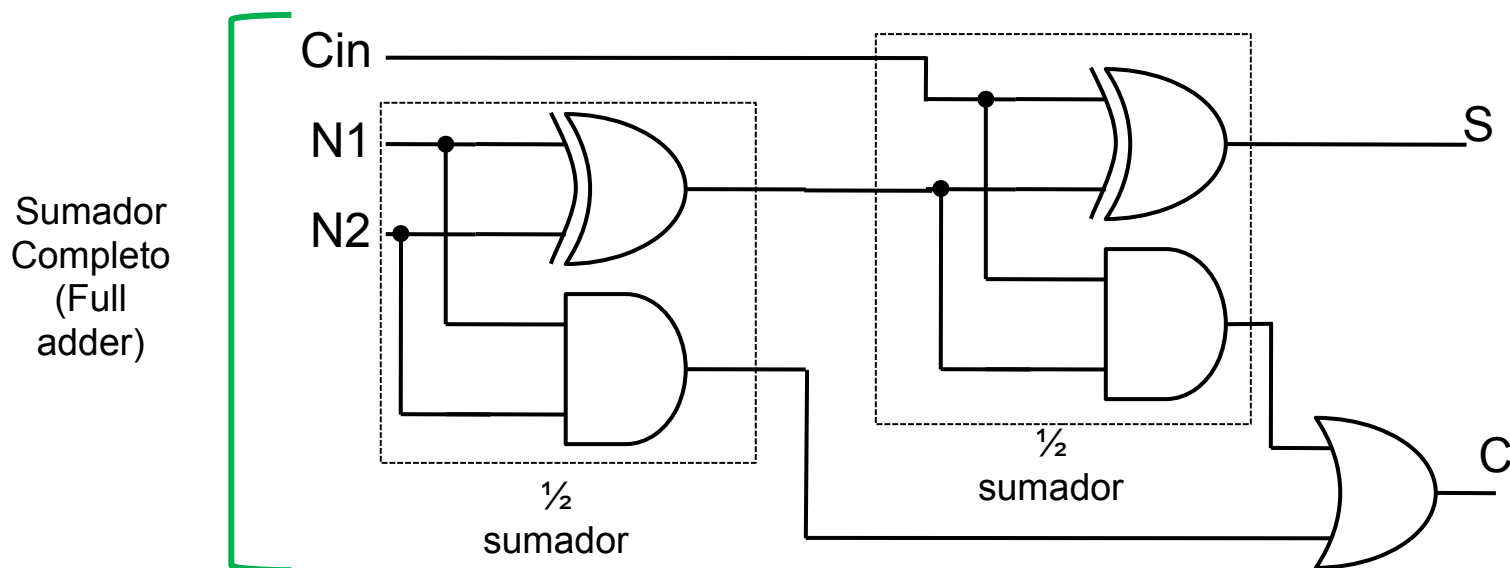
Suma de números binarios. Circuito sumador completo de 1 bit.



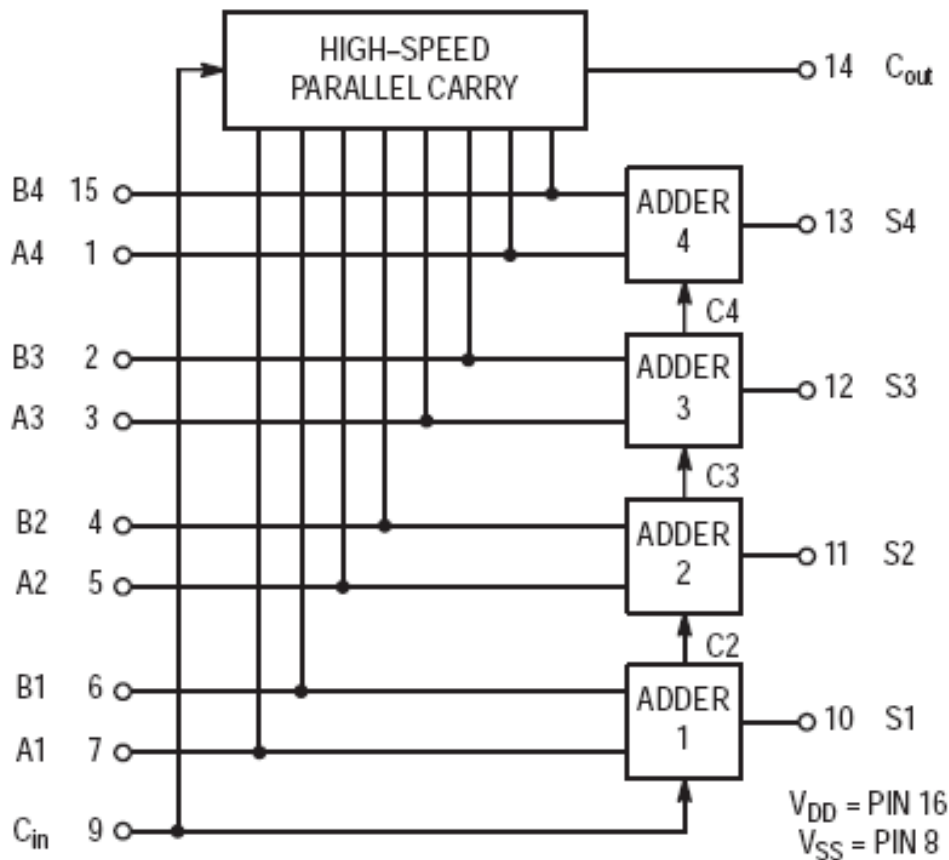
$$S = N1 \oplus N2 \oplus Cin$$

$$Cout = N1 \cdot N2 + Cin \cdot N1 \cdot N2$$

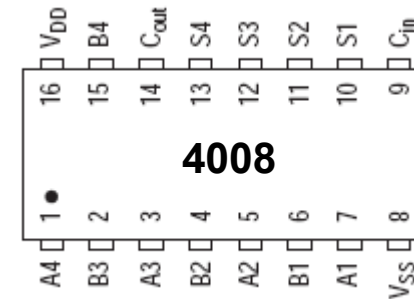
Cin	N1	N2	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



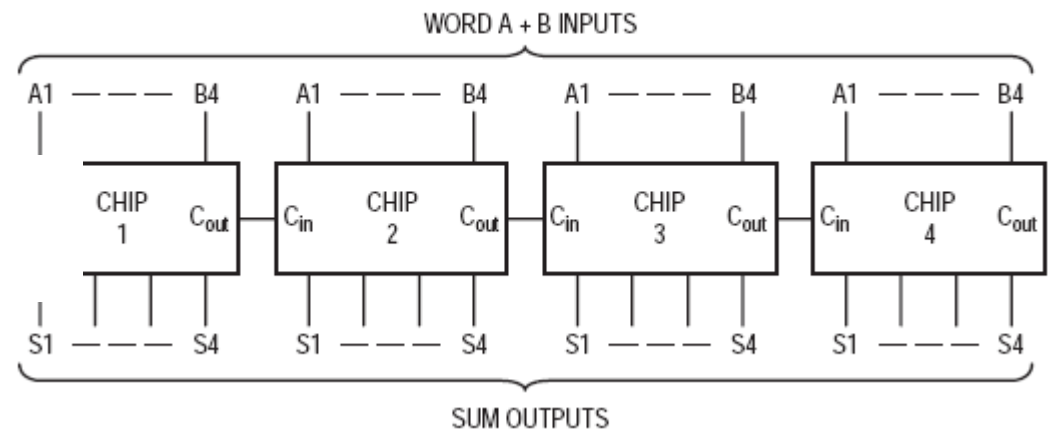
Sumadores integrados. 4008



C _{in}	B	A	C _{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



TYPICAL APPLICATION



Calculation of 16-bit adder speed:

$$t_{p \text{ total}} = t_p (\text{Sum to Carry}) + t_p (\text{Carry to Sum}) + 2 t_p (\text{Carry to Carry})$$

The guaranteed 16-bit adder speed at 10 V, 25°C, C_L = 50 pF is:

$$t_{p \text{ total}} = 290 + 310 + 300 = 900 \text{ ns}$$

Representación de números negativos con sistema binario.

Para representar números con signo (positivo o negativo) y realizar operaciones con ellos, el sistema más conveniente es **Complemento a 2**.

Se utiliza en sistemas con un **número fijo de dígitos**, no sirve si el número de dígitos varía*

Para obtener un número con signo negativo, se debe:

1. Invertir / complementar cada bit.
2. Sumar uno.

Ej.: Si usamos 8 bits de "ancho" para nuestro sistema:

$$\begin{array}{r} -119_{10} \quad 01110111 \text{ (+119)} \\ + 10001000 \text{ se complementan todos los bits.} \\ \hline \quad \quad \quad 1 \\ \hline 10001001 \end{array} \rightarrow \text{Que es el número -119 representado en CA2}$$

Si quiero obtener el valor absoluto de un número en CA2
repito el mismo procedimiento...

*Si se requiere se puede adaptar un número con signo de una cantidad de bits dada a una mayor solamente, repitiendo el bit de signo hacia la izquierda.

Representación de números negativos con sistema binario.

Para obtener el valor absoluto (el valor “positivo”) de un número representado en CA2 se usa el mismo procedimiento que para “hacer negativo” el número.

Entonces, para obtener el valor absoluto de un número en CA2:

▢ Invertir / complementar cada bit.

▢ Sumar uno.

0 en el Bit
más
significativo
indica +

-119_{10}

Ej.: Usando 8 bits de “ancho”:

01110111 (+119)
+ 10001000 se complementan todos los bits.

1 indica -

1
10001001

→ Que es el número -119 representado en CA2

Y entonces... 10001001 (-119)
+ 01110110 se complementan todos los bits.

1

01110111 → Que es el $+119_{10}$

En CA2 se utiliza un bit para el signo. Con n bits ya no se puede representar 2^n números.

El rango de números es entre $-2^{(n-1)}$ hasta $-2^{(n-1)}-1$. Con 8 bits, entre -128 y 127.

Operaciones de números con signo.

Con los números en CA2 se suma o resta con el mismo método visto antes, pero...

Si el **rango** de números posibles **se excede**, el **resultado es erróneo**.

Ejemplos usando CA2 de 4 bits:

$$\begin{array}{r} + \quad +3 \quad 0011 \\ +4 \quad 0100 \\ \hline +7 \quad 0111 \end{array}$$

$$\begin{array}{r} + \quad +6 \quad 0110 \\ -3 \quad 1101 \\ \hline +3 \quad 1 \quad 0011 \end{array}$$

$$\begin{array}{r} + \quad -2 \quad 1110 \\ -6 \quad 1010 \\ \hline -8 \quad 1 \quad 1000 \end{array}$$

$$\begin{array}{r} + \quad +4 \quad 0100 \\ -7 \quad 1001 \\ \hline -3 \quad 1101 \end{array}$$

En CA2 el carry se descarta,
no se toma en cuenta.

$$\begin{array}{r} + \quad -3 \quad 1101 \\ -6 \quad 1010 \\ \hline -9 \quad 1 \quad 0111 \rightarrow +7 \end{array}$$

Suma de 2 números
negativos y el resultado es +

$$\begin{array}{r} + \quad -8 \quad 1000 \\ -8 \quad 1000 \\ \hline -16 \quad 1 \quad 0000 \rightarrow 0 \end{array}$$

Suma de 2 números
negativos y el resultado es +

$$\begin{array}{r} + \quad +5 \quad 0101 \\ +6 \quad 0110 \\ \hline +11 \quad 1011 \rightarrow -5 \end{array}$$

Suma de 2 números positivos
y el resultado es -

Overflow: Cuando el resultado no se puede representar con una cantidad de bits dada.

Habrà Overflow si:
Los signos de los sumandos son iguales,
pero distintos del signo del resultado.

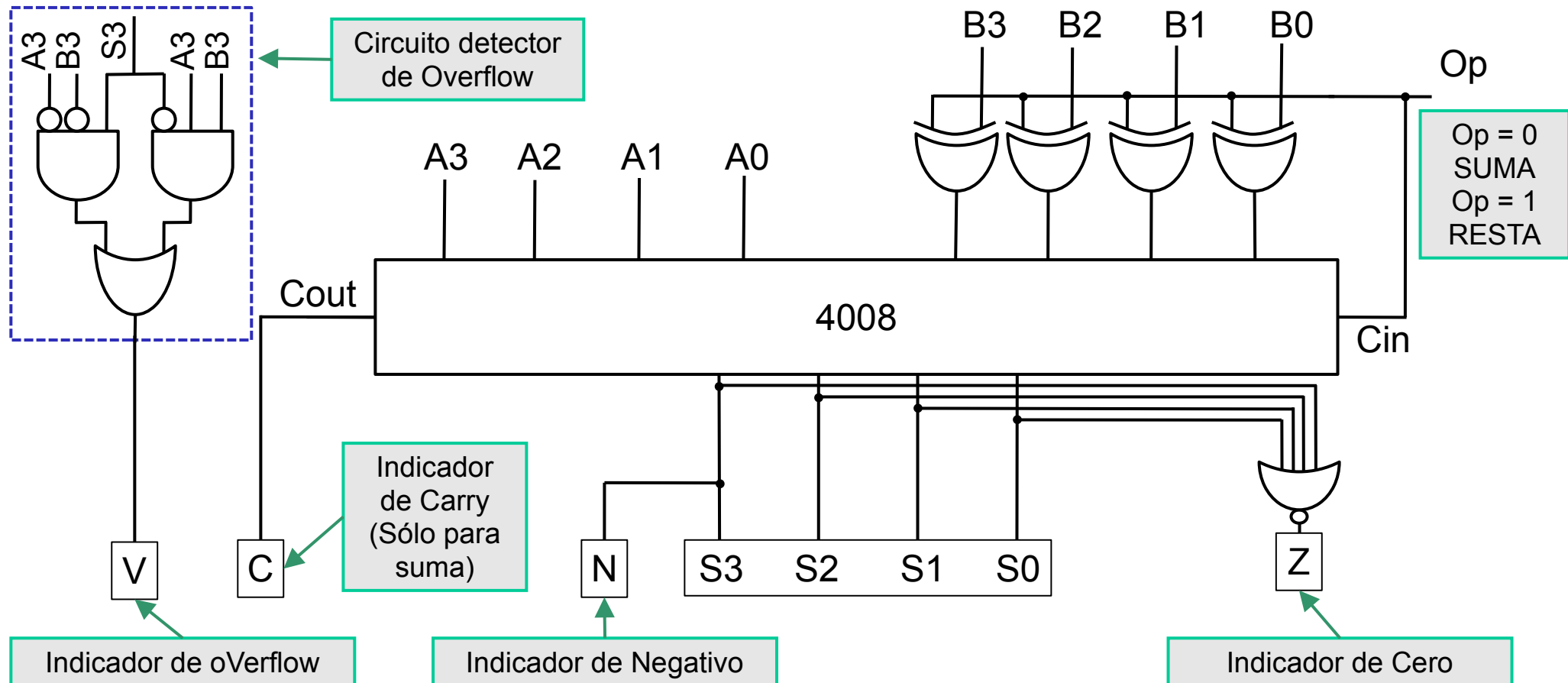
Operaciones de números con signo.

Para restar en CA2 se suma, pero cambiando el signo del segundo sumando.

Ej.:
$$\begin{array}{r} +4 \\ -3 \\ \hline +1 \end{array} \quad \begin{array}{r} 0100 \\ 0011 \\ \hline 0111 \end{array} \quad \begin{array}{r} 0100 \\ +1100 \\ \hline 1\ 0001 \end{array}$$

-3 en CA2

Circuito sumador / restador de 4 bits utilizando un sumador 4008



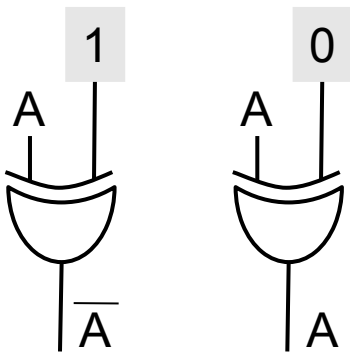
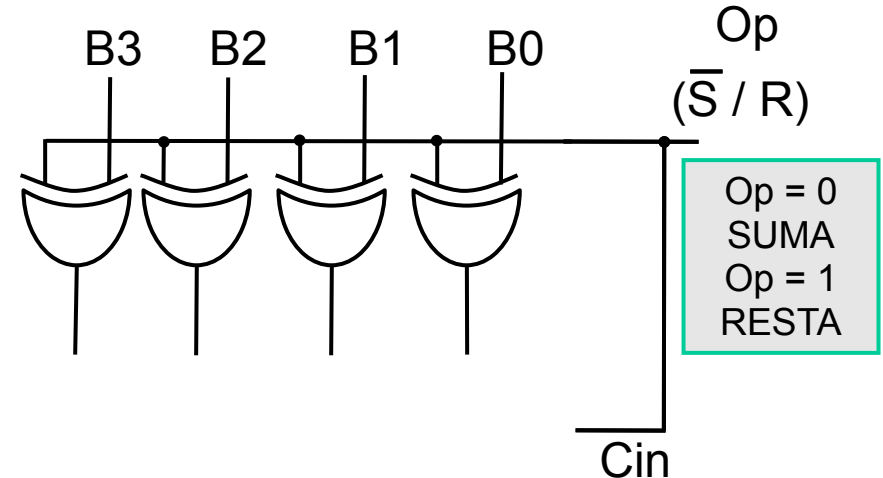
Operaciones de números con signo.

Circuito sumador / restador de 4 bits utilizando un sumador 4008

Esta parte del circuito realiza de ser necesario,
el complemento a 2 del segundo sumando

si $Op = 1$:

Las XOR invierten los bits y
poniendo $Cin = 1$, está sumando 1.



Si una de las dos entradas de una XOR está en 1, la entrada restante aparece invertida en la salida: funciona como un inversor.

Si en cambio fijo una entrada en 0, la otra se refleja en la salida. No invierte.

Los indicadores (“flags”) C, V, Z, N son comunes, y muy utilizados en microprocesadores y microcontroladores.

Suma de números BCD.

Se suman igual que los binarios naturales.

Si el resultado de un dígito es mayor a 9 (no se puede representar)
se corrige sumando 6 (0110)

$$\begin{array}{r} +5 \quad +0101 \\ +9 \quad +1001 \\ \hline 14 \quad +1110 \\ \quad +0110 \\ \hline 1 \quad 0100 \\ \hline \boxed{1} \quad \boxed{4} \end{array}$$

No se puede
representar en
BCD, se suma 6.

$$\begin{array}{r} +15 \quad +0001 \quad 0101 \\ +9 \quad +0000 \quad 1001 \\ \hline 24 \quad 0010 \quad +1110 \\ \quad \quad +0110 \\ \hline 0010 \quad \boxed{1} \quad 0100 \end{array}$$

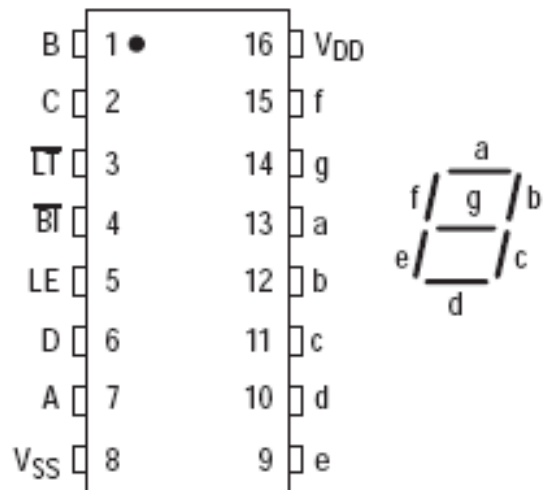
Esto se llama "Acarreo
intermedio" (Half Carry)

En algunos microprocesadores y microcontroladores, existe
un flag para Half carry: el "H"

Conversor BCD a 7 segmentos. 4511.

Permite visualizar un dígito BCD natural (4 bits) en un display de siete segmentos, del tipo cátodo común.

4511



Inputs							Outputs							Display
LE	BI	LT	D	C	B	A	a	b	c	d	e	f	g	
X	X	0	X	X	X	X	1	1	1	1	1	1	1	8
X	0	1	X	X	X	X	0	0	0	0	0	0	0	Blank
0	1	1	0	0	0	0	1	1	1	1	1	1	0	0
0	1	1	0	0	0	1	0	1	1	0	0	0	0	1
0	1	1	0	0	1	1	1	1	1	1	0	0	1	2
0	1	1	0	0	1	1	1	1	1	1	0	0	1	3
0	1	1	0	1	0	0	0	1	1	0	0	1	1	4
0	1	1	0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	1	1	0	0	0	1	1	1	1	1	6
0	1	1	0	1	1	1	1	1	1	0	0	0	0	7
0	1	1	1	0	0	0	1	1	1	1	1	1	1	8
0	1	1	1	0	0	1	1	1	1	0	0	1	1	9
0	1	1	1	0	1	0	0	0	0	0	0	0	0	Blank
0	1	1	1	0	1	1	0	0	0	0	0	0	0	Blank
0	1	1	1	1	0	0	0	0	0	0	0	0	0	Blank
0	1	1	1	1	1	0	0	0	0	0	0	0	0	Blank
0	1	1	1	1	1	1	0	0	0	0	0	0	0	Blank
1	1	1	X	X	X	X	*							*

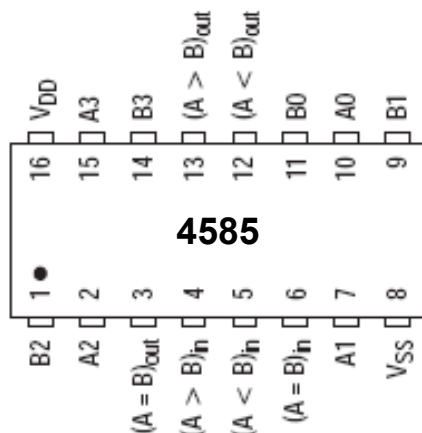
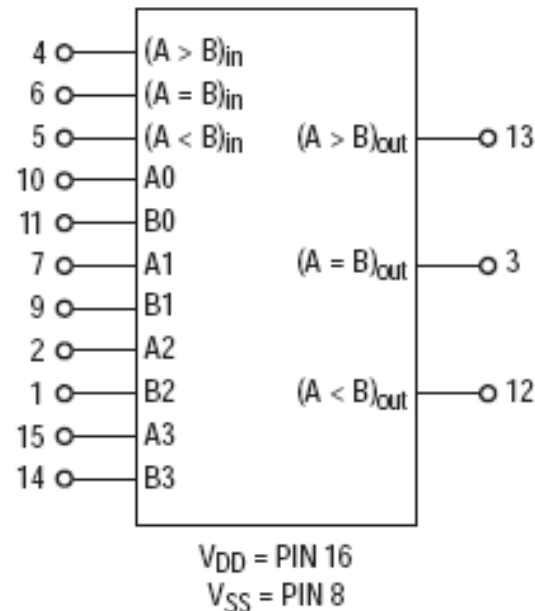
Con la entrada LE, si es 1, se memoriza el número. (LE = Latch Enable)

BL, si es 0 apaga todos los segmentos del display. (BL = Blank)

Si LT = 0, enciende todo el display. (LT = Lamp Test)

Comparador de magnitud. 4585.

Es un circuito que recibe 2 números de 4 bits como entrada y en sus salidas informa si son iguales, o cuál es mayor de los dos.



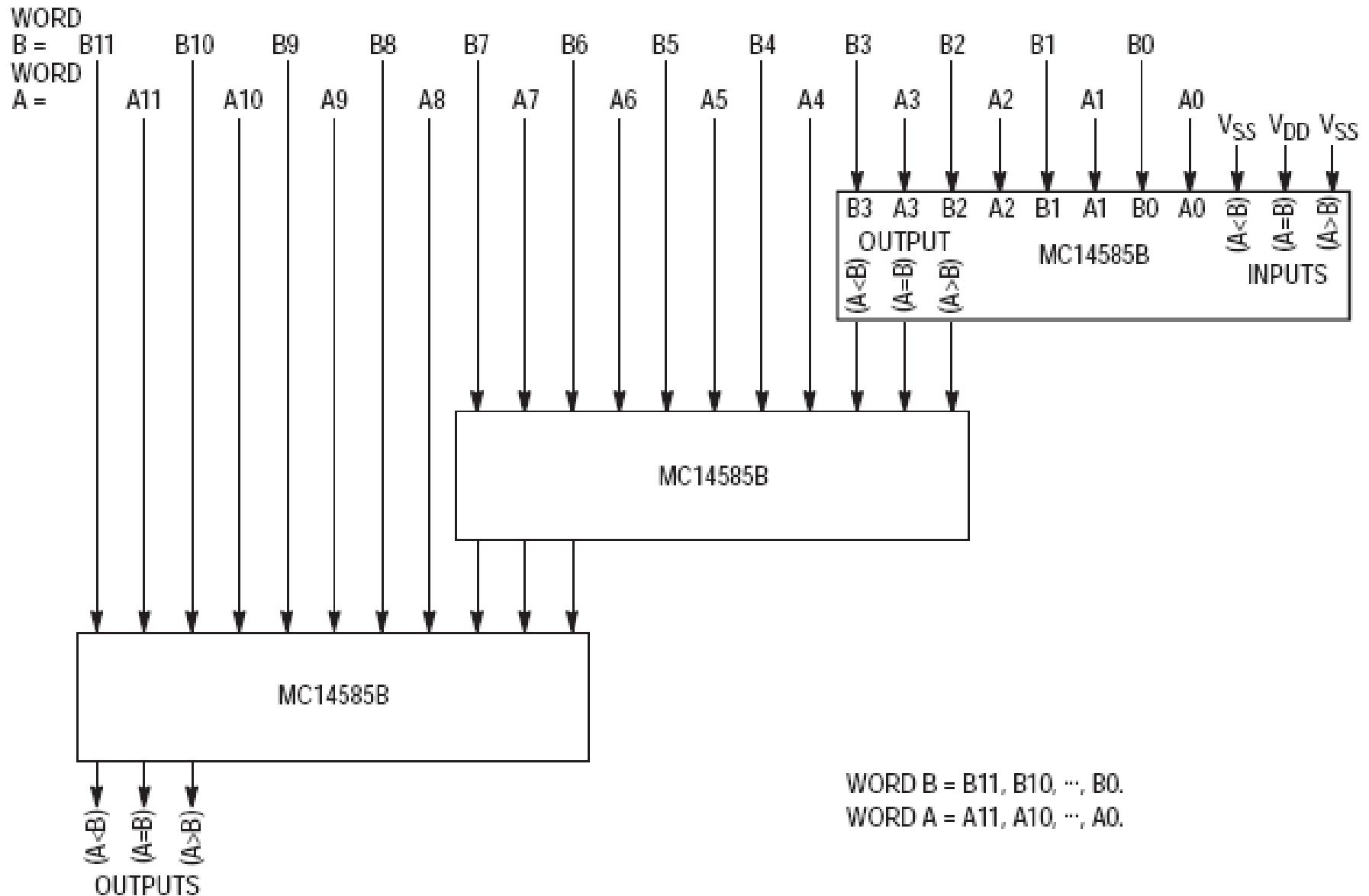
Inputs							Outputs		
Comparing				Cascading					
A3, B3	A2, B2	A1, B1	A0, B0	A < B	A = B	A > B	A < B	A = B	A > B
A3 > B3	x	x	x	x	x	x	0	0	1
A3 = B3	A2 > B2	x	x	x	x	x	0	0	1
A3 = B3	A2 = B2	A1 > B1	x	x	x	x	0	0	1
A3 = B3	A2 = B2	A1 = B1	A0 > B0	x	x	x	0	0	1
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	0	x	0	0	1
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	1	x	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	1	0	x	1	0	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	1	1	x	1	1	0
A3 = B3	A2 = B2	A1 = B1	A0 < B0	x	x	x	1	0	0
A3 = B3	A2 = B2	A1 < B1	x	x	x	x	1	0	0
A3 = B3	A2 < B2	x	x	x	x	x	1	0	0
A3 < B3	x	x	x	x	x	x	1	0	0

Las entradas (A>B)_{in}, (A=B)_{in}, (A<B)_{in} sirven para poder comparar números más grandes, usando más comparadores.

Si no se utilizan, se pueden conectar a 0V.

Comparador de magnitud. 4585.

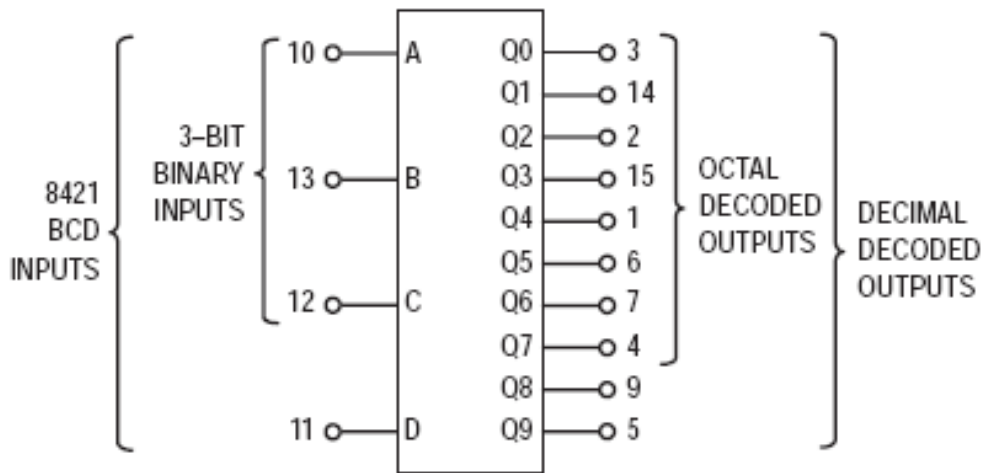
Una expansión de comparadores: En el siguiente diagrama, comparador de 12 bits.



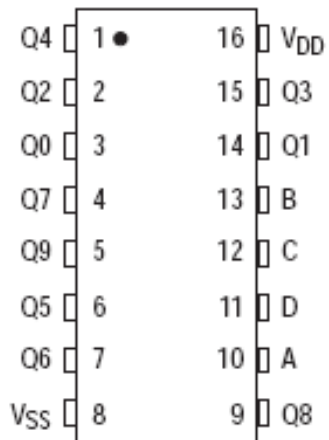
Decodificador. 4028.

Un decodificador activa la salida seleccionada por un número en su entrada.
La cantidad de salidas de un decodificador es 2^n , donde n es el número de entradas.

4028



V_{DD} = PIN 16
 V_{SS} = PIN 8



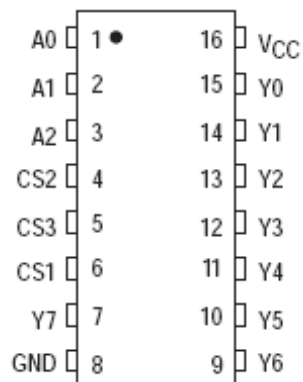
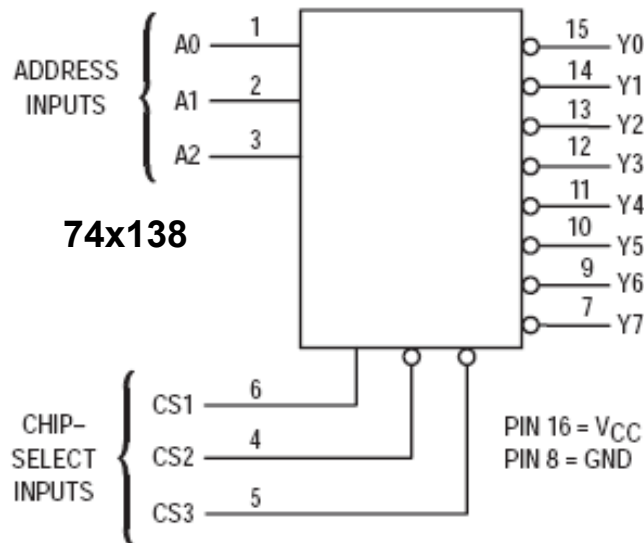
D	C	B	A	Q9	Q8	Q7	Q6	Q5	Q4	Q3	Q2	Q1	Q0
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0

Ej.: Si en la entrada (DCBA) colocamos el número 7 (0111) activamos la salida Q7. Sólo funciona con entrada BCD.

Otro decodificador. 74x138

Este dispositivo cuenta con terminales de “Habilitación” y funciona con binario natural. Las habilitaciones sirven para realizar expansiones y / o para que todas las salidas presenten alta impedancia (Hi-Z)

Las salidas, en este caso, son “activas en nivel bajo”: al ser seleccionadas, tienen “0”.



Inputs						Outputs							
CS1	CS2	CS3	A2	A1	A0	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	X	H	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	L	H	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	L	H	H	H
H	L	L	H	H	L	H	H	H	H	H	L	H	H
H	L	L	H	H	H	H	H	H	H	H	H	L	H

En algunas tablas de verdad se usa H y L, en lugar de 1 y 0 respectivamente.

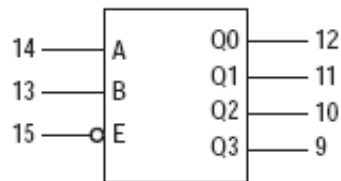
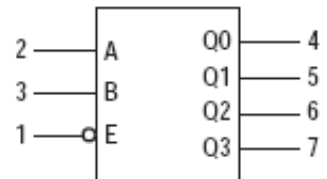
Este circuito activa (pone en cero) la salida seleccionada por los terminales A2, A1, A0 si las tres habilitaciones CS3, CS2, CS1 tienen 0, 0, 1 respectivamente.

Decodificadores. 4555 y 4556.

Cada uno de estos integrados incluye dos decodificadores, de 4 salidas y un terminal de habilitación cada uno .

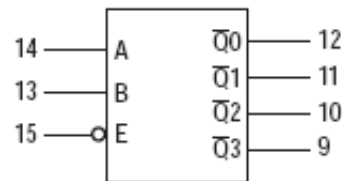
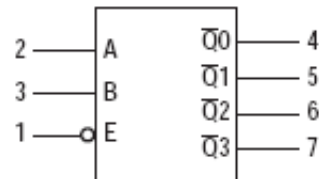
El 4555 tiene salidas activas en 1 y el 4556 tiene salidas activas en 0.

4555

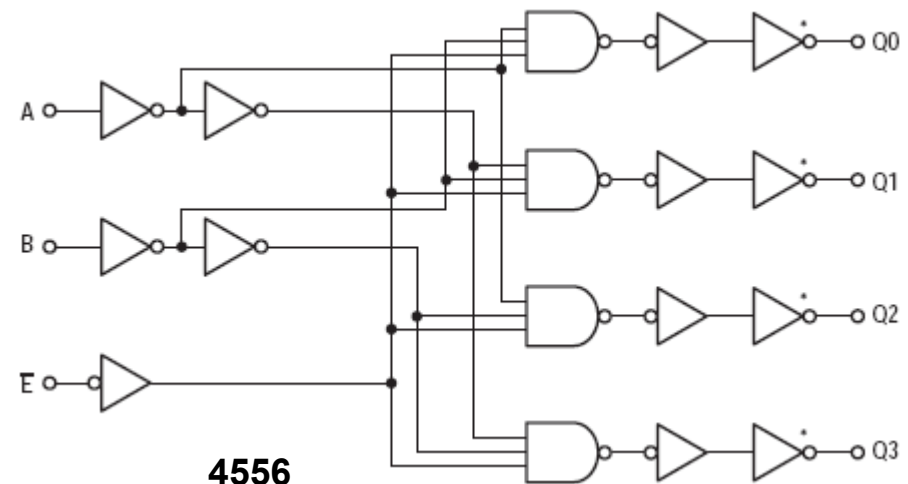
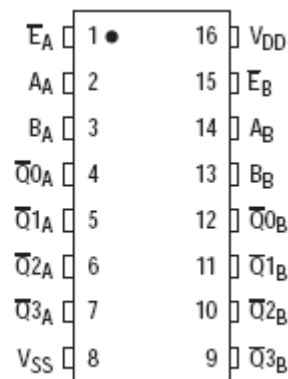
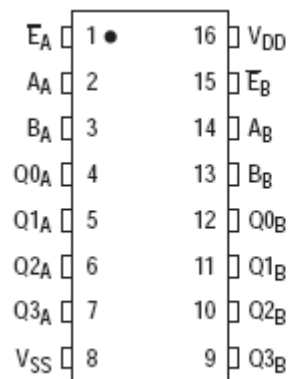


V_{DD} = PIN 16
 V_{SS} = PIN 8

4556



Inputs			Outputs							
Enable	Select		MC14555B				MC14556B			
$\bar{E}$	B	A	Q3	Q2	Q1	Q0	$\bar{Q}3$	$\bar{Q}2$	$\bar{Q}1$	$\bar{Q}0$
0	0	0	0	0	0	1	1	1	1	0
0	0	1	0	0	1	0	1	1	0	1
0	1	0	0	1	0	0	1	0	1	1
0	1	1	1	0	0	0	0	1	1	1
1	X	X	0	0	0	0	1	1	1	1



Decodificadores. Aplicaciones.

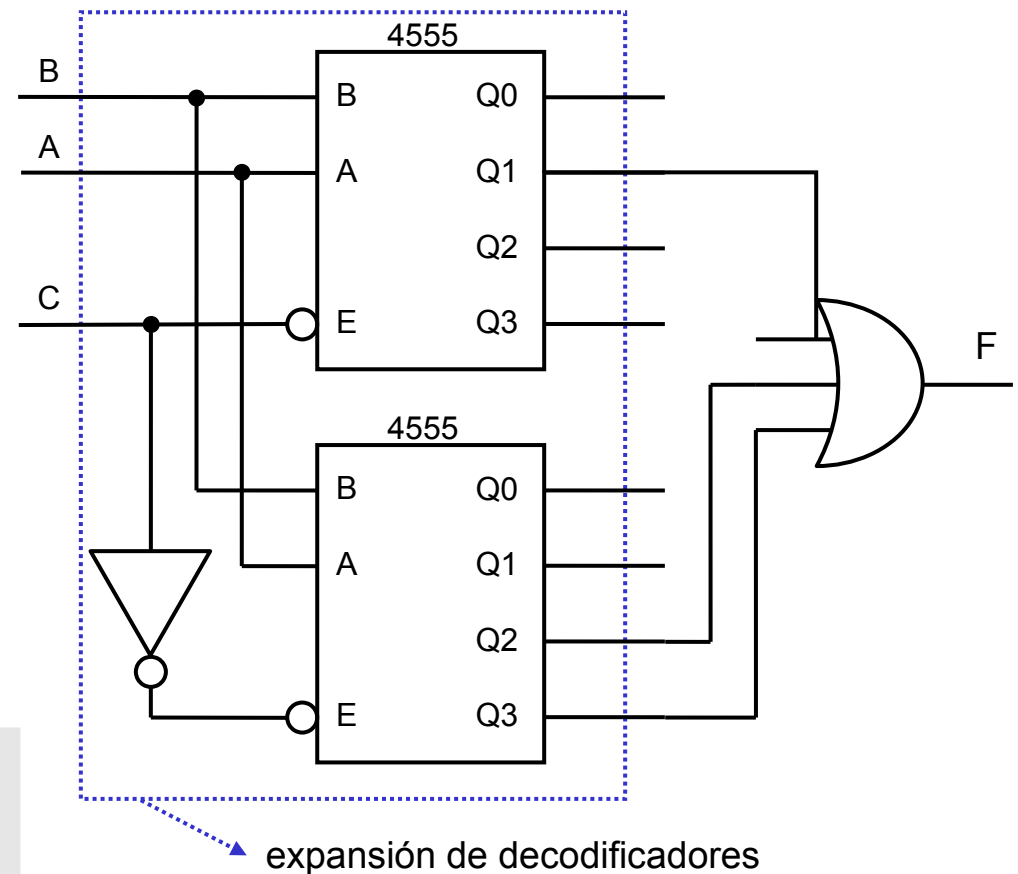
Con un decodificador se puede hacer un circuito que cumpla con una tabla de verdad, utilizando una cantidad mínima de compuertas.

Para implementar la función siguiente:

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Se necesitan 3 entradas ($2^3=8$).
Se puede utilizar por ejemplo **dos** 4555
(cuatro salidas cada uno)

Utilizamos las salidas correspondientes a los renglones de la tabla que deben ser 1 conectados a una compuerta OR.

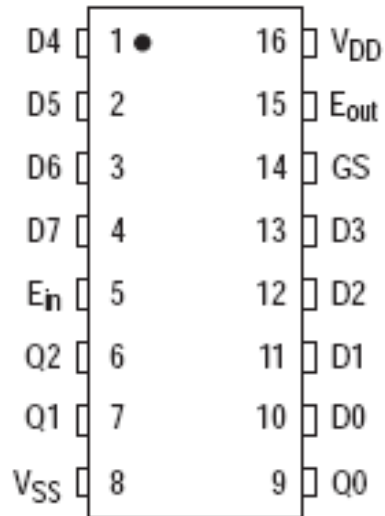


Codificador con prioridad. 4532.

Un codificador con prioridad presentará en la salida el número de la entrada (la de número más grande) a la que se aplique un 1.

Este integrado en particular presenta 8 entradas, y por lo tanto 3 salidas.

4532



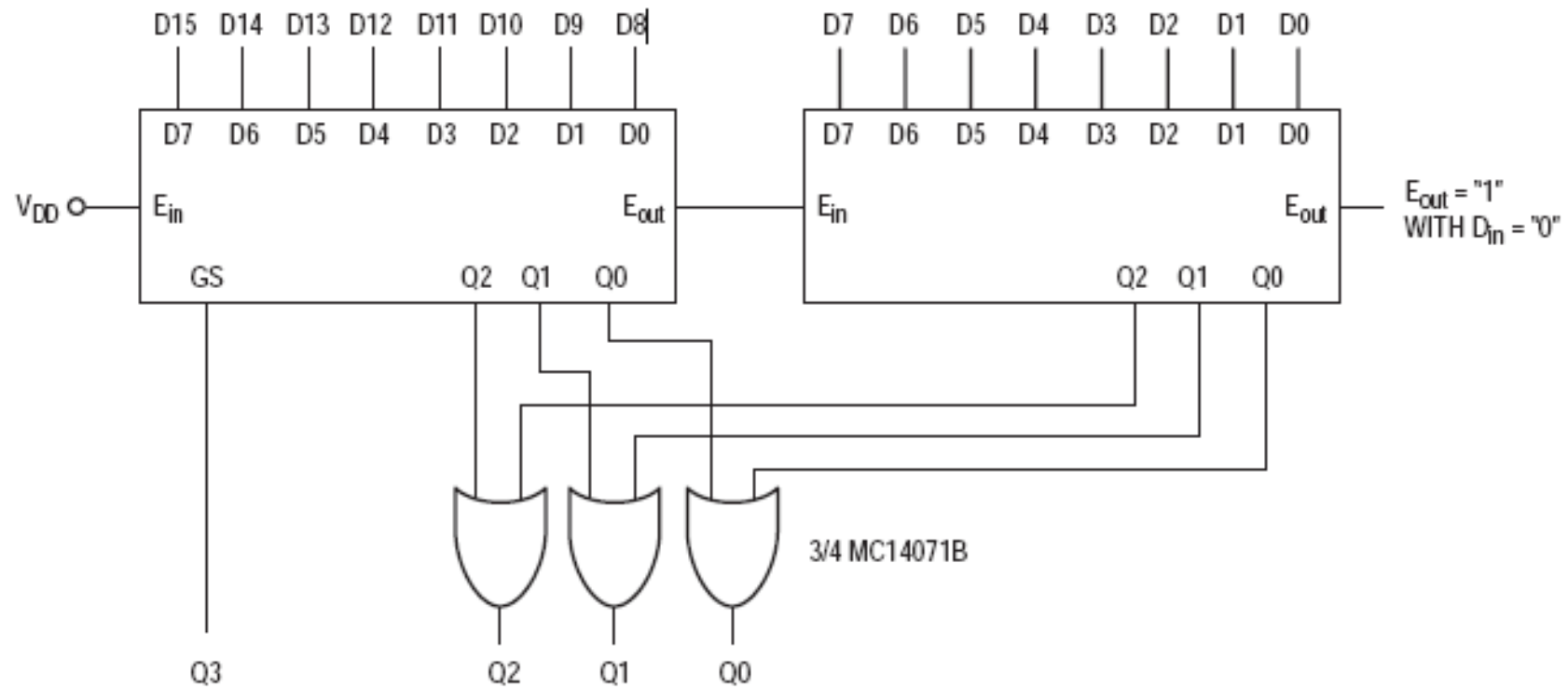
Input									Output				
E _{in}	D7	D6	D5	D4	D3	D2	D1	D0	GS	Q2	Q1	Q0	E _{out}
0	X	X	X	X	X	X	X	X	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	1
1	1	X	X	X	X	X	X	X	1	1	1	1	0
1	0	1	X	X	X	X	X	X	1	1	1	0	0
1	0	0	1	X	X	X	X	X	1	1	0	1	0
1	0	0	0	1	X	X	X	X	1	1	0	0	0
1	0	0	0	0	1	X	X	X	1	0	1	1	0
1	0	0	0	0	0	1	X	X	1	0	1	0	0
1	0	0	0	0	0	0	1	X	1	0	0	1	0
1	0	0	0	0	0	0	0	1	1	0	0	0	0

La salida GS presenta 1 si hay alguna entrada activa. Esto es útil para diferenciar cuando tengo a la salida 0000 si es porque no hay ninguna entrada con un 1, o la entrada 0 es la que tiene aplicado 1.

Ein y Eout se utilizan en caso de querer realizar expansiones.

Codificador con prioridad. 4532.

Se puede realizar una expansión para lograr más entradas y salidas.



Este conjunto permite 16 entradas, y entrega el número correspondiente a la entrada activa con 4 bits.

Se puede utilizar uno de estos dispositivos para obtener la tecla presionada en un teclado.

Circuitos secuenciales

Circuitos secuenciales.

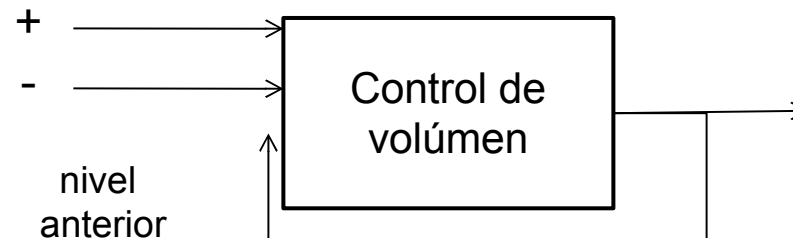
Es un circuito en el cual el conjunto de estados actuales dependen de los estados anteriores.

Las salidas dependen de las entradas y del estado previo del circuito.

Para retener el estado previo es necesario que estos circuitos tengan “memoria”.

Ejemplo de circuito secuencial:

Control de volúmen con dos botones: + y –
Si se presiona el + se sube el volúmen... Desde el nivel previo.



Otros:

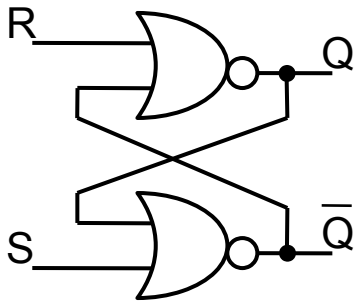
Cambio de canal en un TV con dos botones.
Encendido y apagado de un artefacto con un pulsador .
Un contador que aumenta el número si recibe un impulso eléctrico.

Circuitos secuenciales. Biestables.

Los circuitos biestables pueden tomar dos estados estables.

Son los elementos básicos de un sistema secuencial.

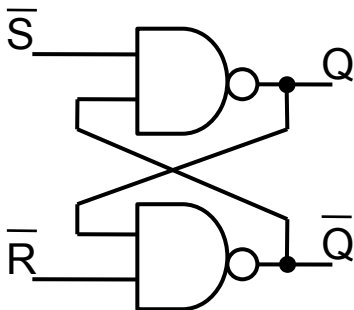
Biestable RS utilizando compuertas NOR



S	R	Q^t	$/Q^t$
0	0	Q^{t-1}	$/Q^{t-1}$
0	1	0	1
1	0	1	0
1	1	0	0

- Si S y R son 0, mantiene el estado.
- Si R=1, Q pasa a 0.
- Si S=1, Q pasa a 1.
- Si S y R son 1, se produce un estado lógicamente inválido: $Q = /Q$

Biestable RS utilizando compuertas NOR



$/S$	$/R$	Q^t	$/Q^t$
0	0	1	1
0	1	1	0
1	0	0	1
1	1	Q^{t-1}	$/Q^{t-1}$

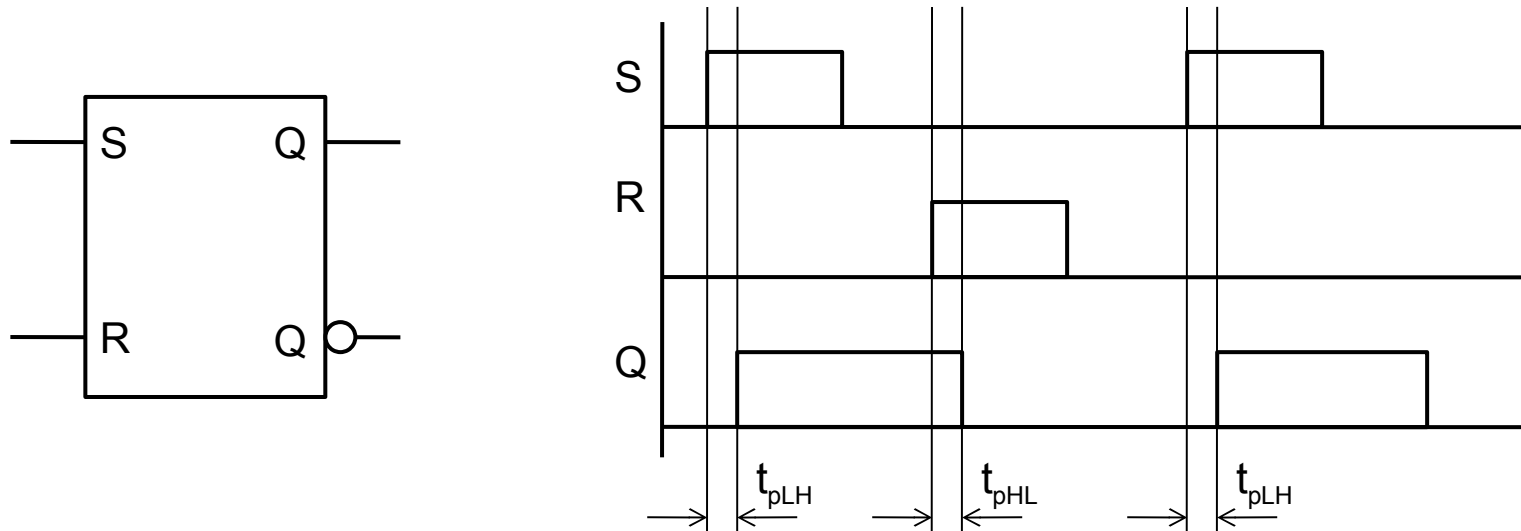
- Si $/S$ y $/R$ son 1, mantiene el estado.
- Si $/R=0$, Q pasa a 0.
- Si $/S=0$, Q pasa a 1.
- Si $/S$ y $/R$ son 0, se produce un estado lógicamente inválido: $Q = /Q$

Latch RS.

A cualquiera de los biestables anteriores se los denomina Latch RS, o SR.

S por “Set” y R por “Reset”

Símbolo y Diagrama temporal del funcionamiento de un latch SR:



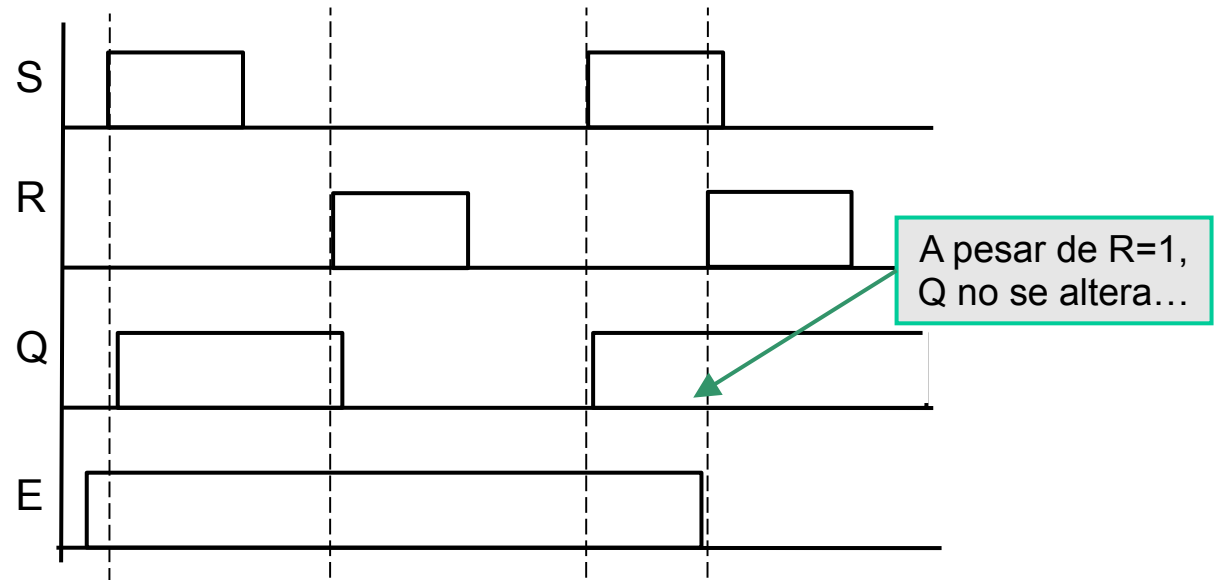
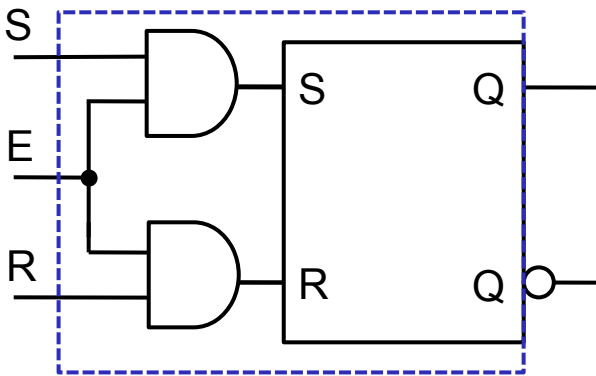
- Se asume que el latch tiene inicialmente **Q=0** lo cual **no siempre es verdadero**.
- Al encender un circuito el **estado inicial** es **aleatorio**, salvo que contenga elementos que fijen un estado inicial determinado.
- Se observa un “retraso” entre las señales R y S y la respuesta en Q. Se debe al **tiempo de propagación** del latch.

Latch RS con Habilitación.

El latch RS se reacciona ante cambios en las entradas S y/o R en cualquier momento.

En muchas aplicaciones es útil evitar este comportamiento.

Esto se logra agregando una Habilitación.



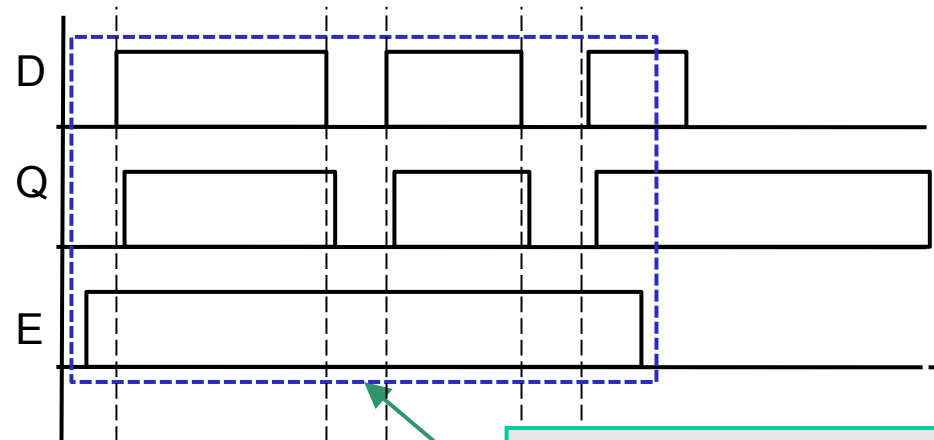
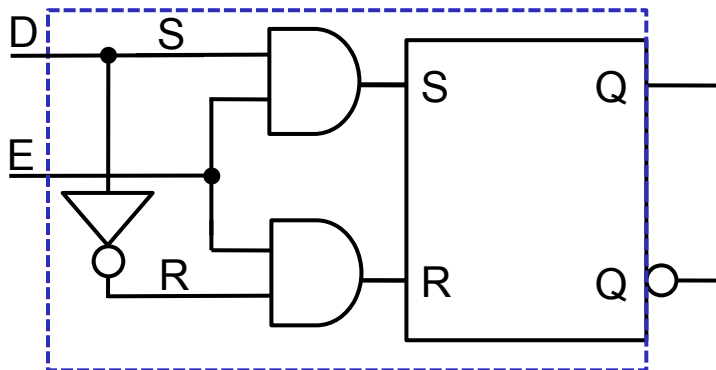
E	S	R	Q^t	$/Q^t$
1	0	0	Q^{t-1}	$/Q^{t-1}$
1	0	1	0	0
1	1	0	1	0
1	1	1	-	-
0	X	X	Q^{t-1}	$/Q^{t-1}$

- Si $E=1$, el funcionamiento es como el de un RS de los vistos anteriormente.
- Si $E=0$, el latch mantiene el estado de sus salidas, independientemente del estado de R y S.

Latch D.

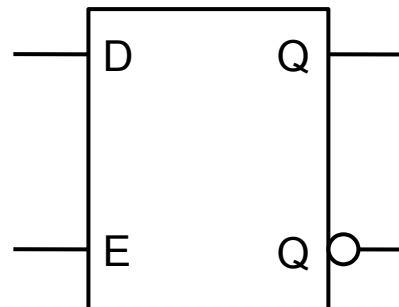
En el latch SR se “memoriza” el estado de dos señales.

En muchas aplicaciones se desea almacenar el estado de una sola señal.



$Q = D$ si $E=1$.
La salida “copia” la entrada,
si está habilitado.

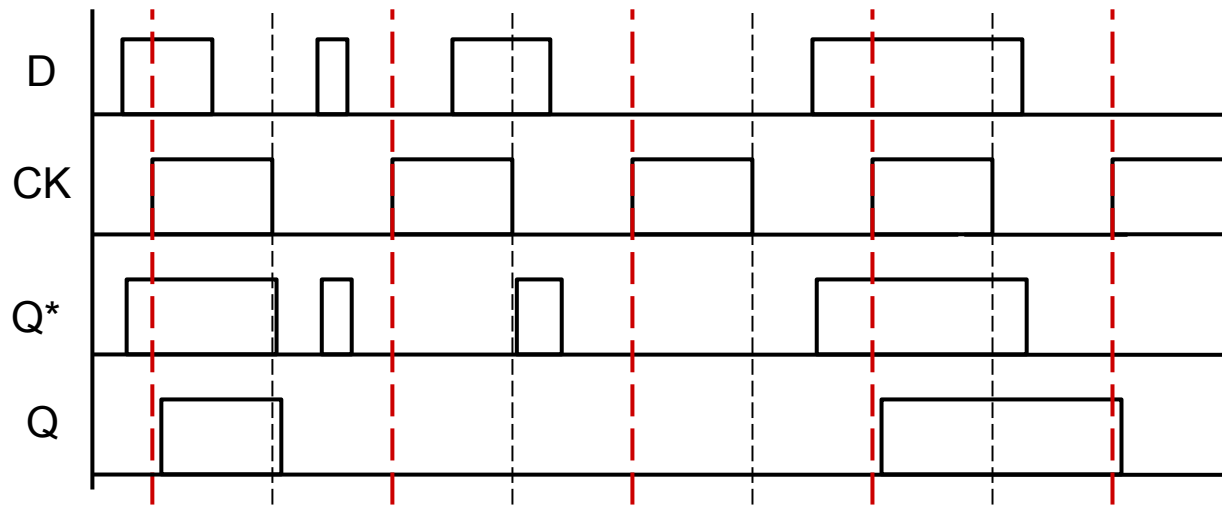
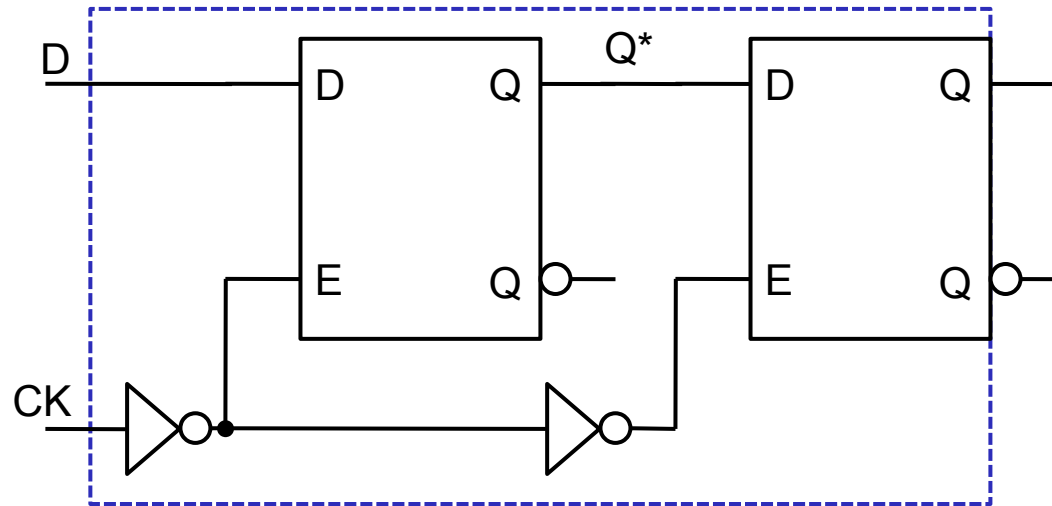
- Símbolo y tabla de comportamiento:



E	D	Q^t
0	X	Q^{t-1}
1	1	1
1	0	0

Por su funcionamiento se le llama “Latch transparente”.

Flip-Flop D



La entrada D se copia a la salida si se produce una **transición** de 0 a 1 (de estado bajo a alto) de la entrada CK

Flip-Flop D

- Los biestables activos solo durante una transición en la habilitación se llaman “Flip-Flop”
 - La transición se llama “flanco” o “Edge”
- Hay dispositivos activos por flanco ascendente o descendente. (Positive-edge y Negative-edge respectivamente)

Un flip-flop D copia la entrada a la salida sólo en el momento de producirse la transición adecuada en la entrada de Clock.

Símbolo

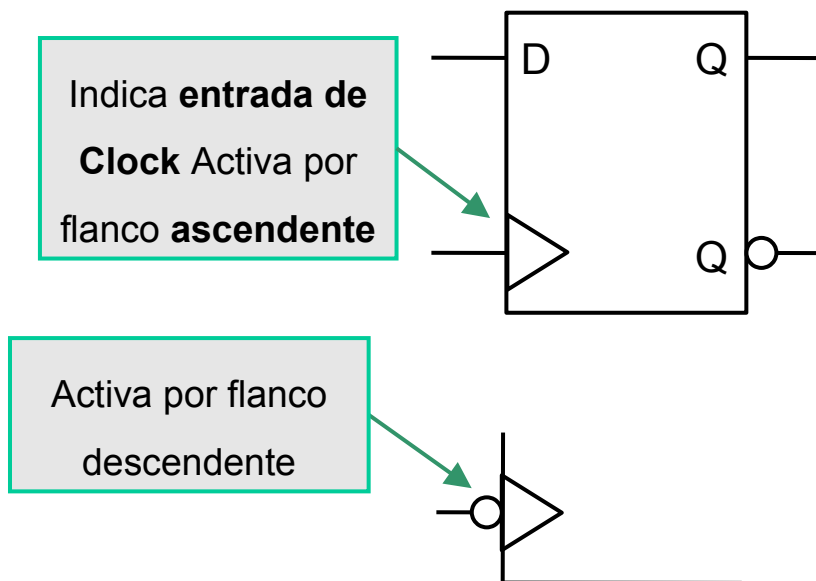


Tabla de comportamiento

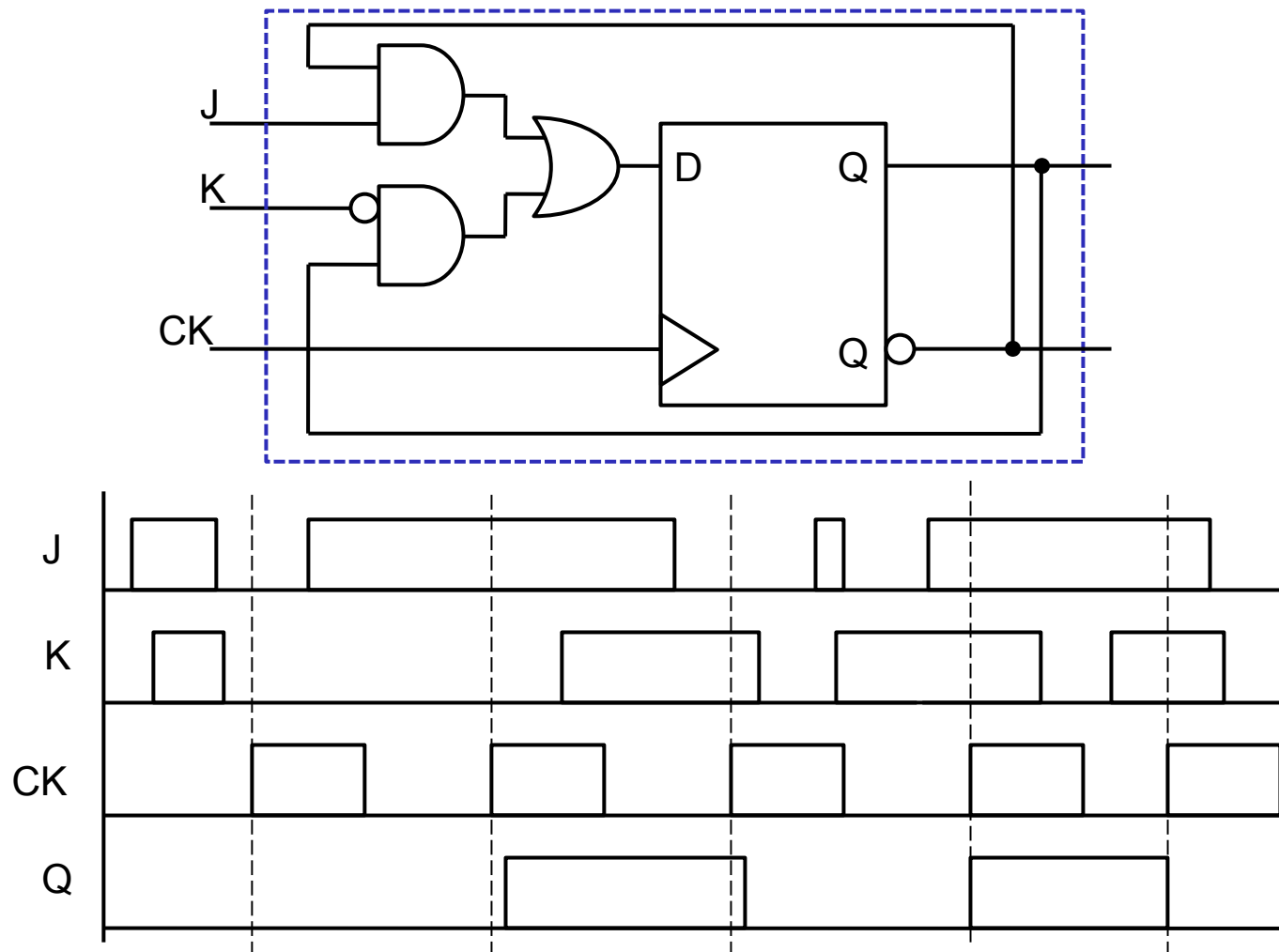
	CK	D	Q ^t
Indica flanco ascendente	↑	0	0
	↑	1	1
	0	X	Q ^{t-1}
	1	X	Q ^{t-1}
	↓	X	Q ^{t-1}

Flip-Flop JK

Se comporta en forma similar a un latch SR, activo por flanco.

Las entradas J y K son análogas a S y R.

Elimina el estado “prohibido” cuando J (S) y K (R) están activos al mismo tiempo.



Flip-Flop JK

Símbolo

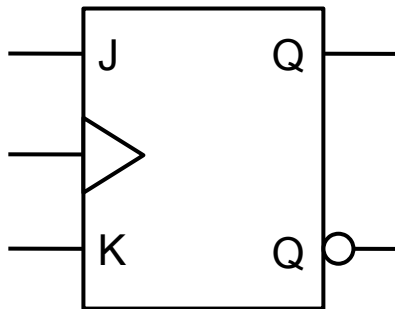


Tabla de comportamiento

CK	J	K	Q^t
$\uparrow$	0	0	Q^{t-1}
$\uparrow$	0	1	0
$\uparrow$	1	0	1
$\uparrow$	1	1	$\overline{Q^{t-1}}$
0	X	X	Q^{t-1}
1	X	X	Q^{t-1}
$\downarrow$	X	X	Q^{t-1}

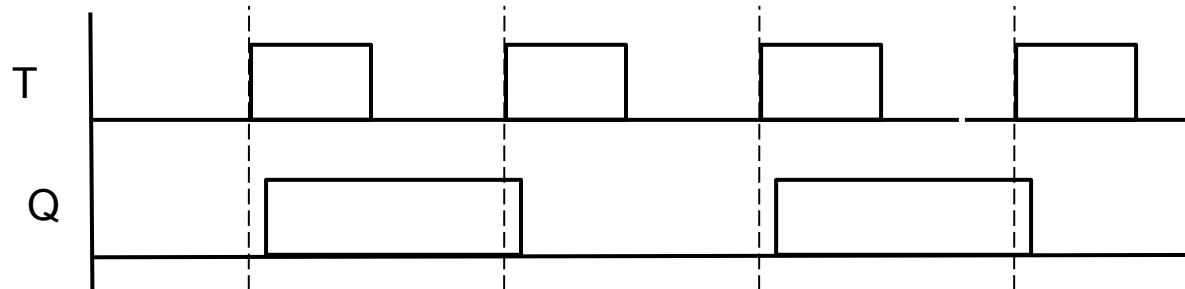
Invierte estado anterior de Q

Pone Q en 1 si J =1, y Q en cero si K = 1, cuando se produce una transición válida de CK.
Si J y K están activos, invierte el estado anterior de Q.

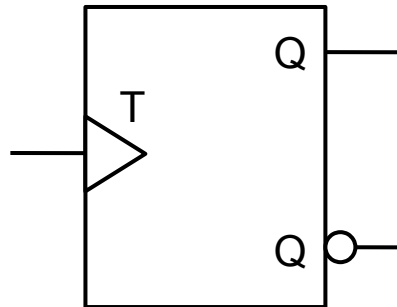
Flip-Flop T

Cambia el estado de Q en cada transición activa de clock.

Son muy utilizados en contadores y divisores de frecuencia.

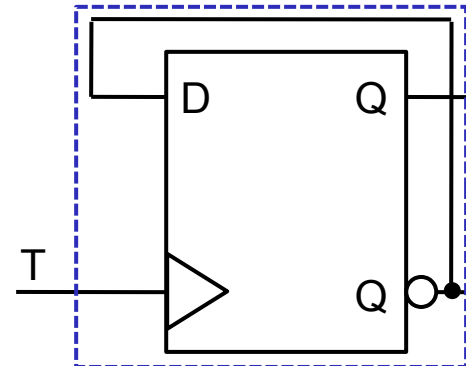
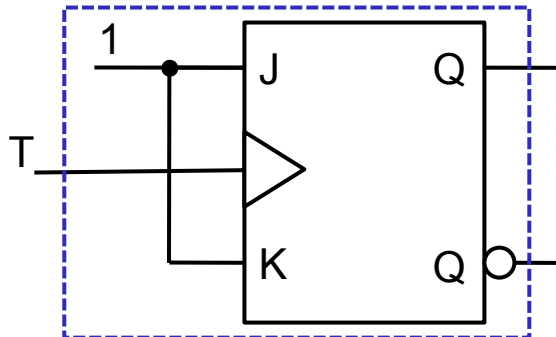


Símbolo

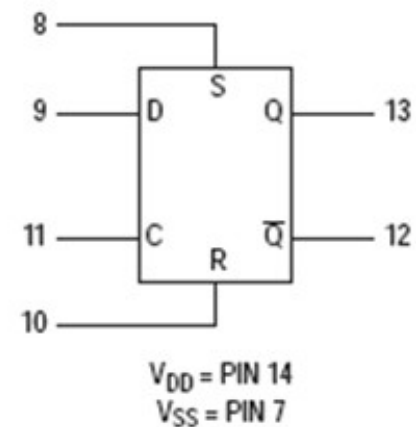
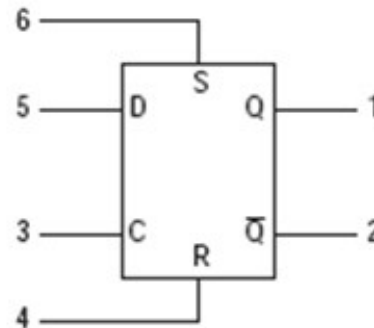
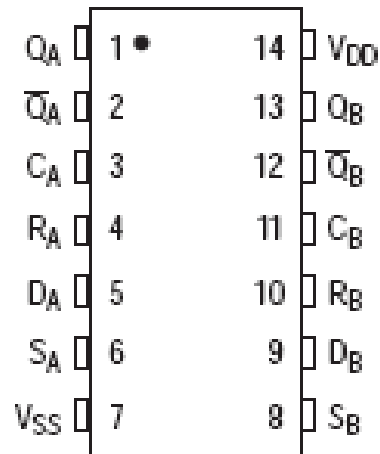


T	Q^t
$\uparrow$	$\overline{Q^{t-1}}$
0	Q^{t-1}
1	Q^{t-1}
$\downarrow$	Q^{t-1}

Tabla de
comportamiento



Flip-Flop D. 4013

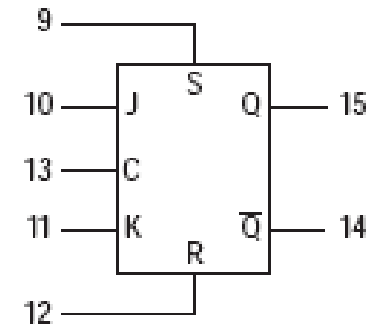
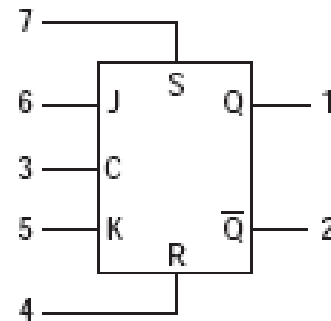
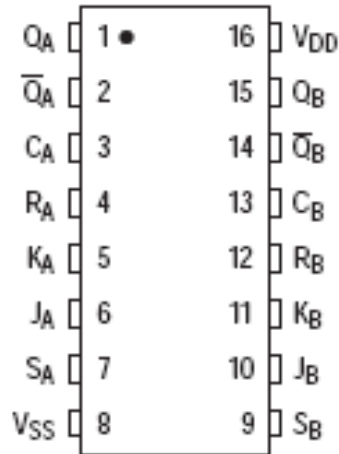


Inputs				Outputs	
Clock [†]	Data	Reset	Set	Q	Q-bar
	0	0	0	0	1
	1	0	0	1	0
	X	0	0	Q	Q-bar
X	X	1	0	0	1
X	X	0	1	1	0
X	X	1	1	1	1

No
Change

X = Don't Care
† = Level Change

Flip-Flop JK. 4027



V_{DD} = PIN 16
 V_{SS} = PIN 8

Inputs						Outputs*	
$C^\dagger$	J	K	S	R	$Q_n^\ddagger$	Q_{n+1}	$\overline{Q}_{n+1}$
	1	X	0	0	0	1	0
	X	0	0	0	1	1	0
	0	X	0	0	0	0	1
	X	1	0	0	1	0	1
	1	1	0	0	Q_0	$\overline{Q_0}$	Q_0
	X	X	0	0	X	Q_n	$\overline{Q_n}$
X	X	X	1	0	X	1	0
X	X	X	0	1	X	0	1
X	X	X	1	1	X	1	1

No
Change

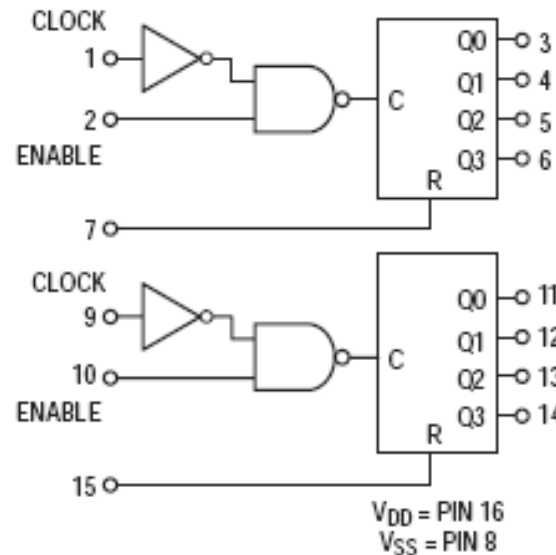
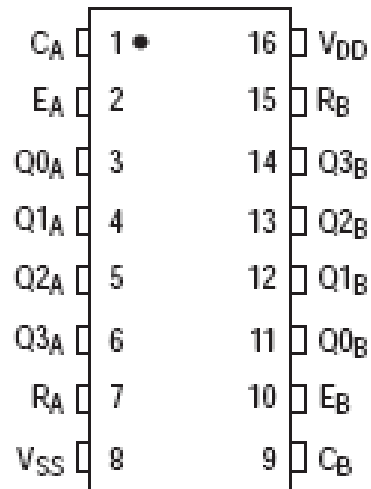
X = Don't Care
 $\dagger$ = Level Change

$\ddagger$ = Present State
 * = Next State

Contador BCD / Binario natural. 4518 / 4520

Cada integrado incluye dos contadores ascendentes, con Reset y Habilitación.
El 4518 cuenta en BCD, el 4520 en binario.

4518 / 4520



Clock	Enable	Reset	Action
	1	0	Increment Counter
0		0	Increment Counter
	X	0	No Change
X		0	No Change
	0	0	No Change
1		0	No Change
X	X	1	Q0 thru Q3 = 0

Si $R = 0$ y $E = 1$, por cada flanco ascendente el número presente a la salida se incrementa.
Si el número es el máximo (9 en el 4518, 15 en el 4520), el próximo flanco vuelve a comenzar.
(Cuenta en forma continua, por ej. El 4518: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2...)

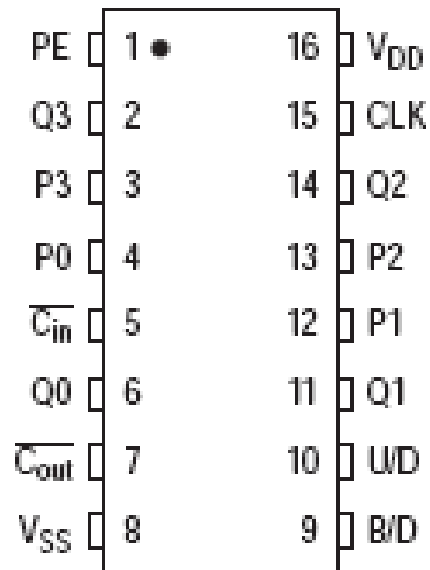
Contador BCD / Binario Natural ascendente / descendente. 4029.

Puede contar en forma ascendente o descendente.

Cuenta con entradas que posibilitan la expansión (C_{out} , C_{in})

Permite fijar un número arbitrario como inicio del conteo. (Preset)

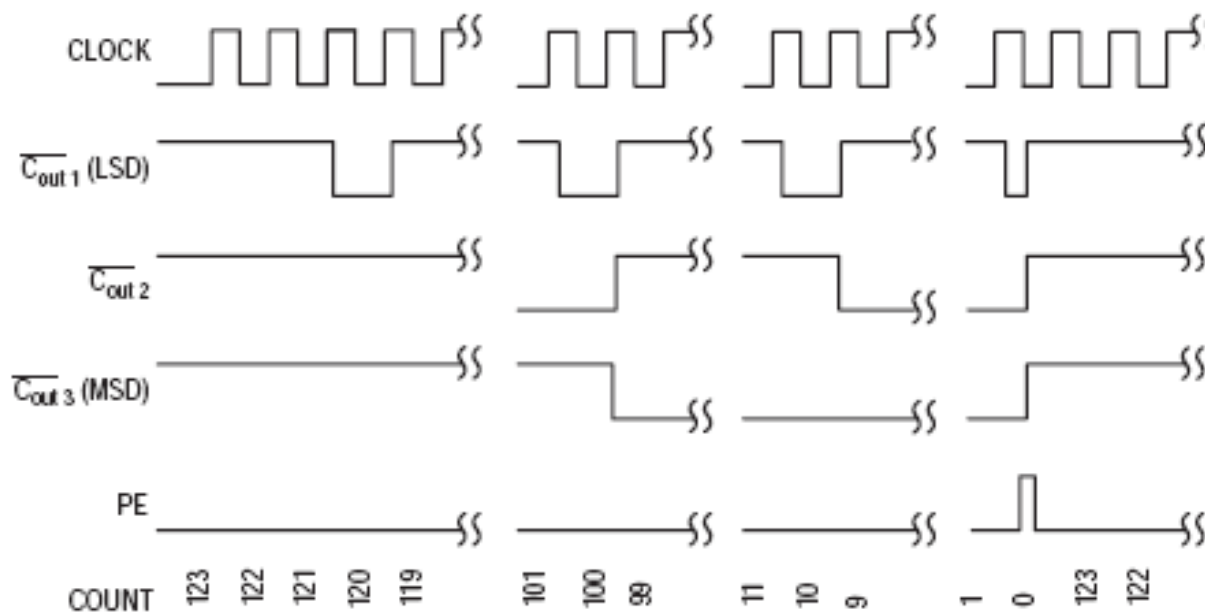
4029



$\overline{C}_{arry\ In}$	Up/Down	Preset Enable	Action
1	X	0	No Count
0	1	0	Count Up
0	0	0	Count Down
X	X	1	Preset

Cuenta hacia arriba o hacia abajo según el estado de las entradas mostradas en la tabla, en cada flanco activo.

Permite seleccionar además si cuenta en BCD o en Natural.

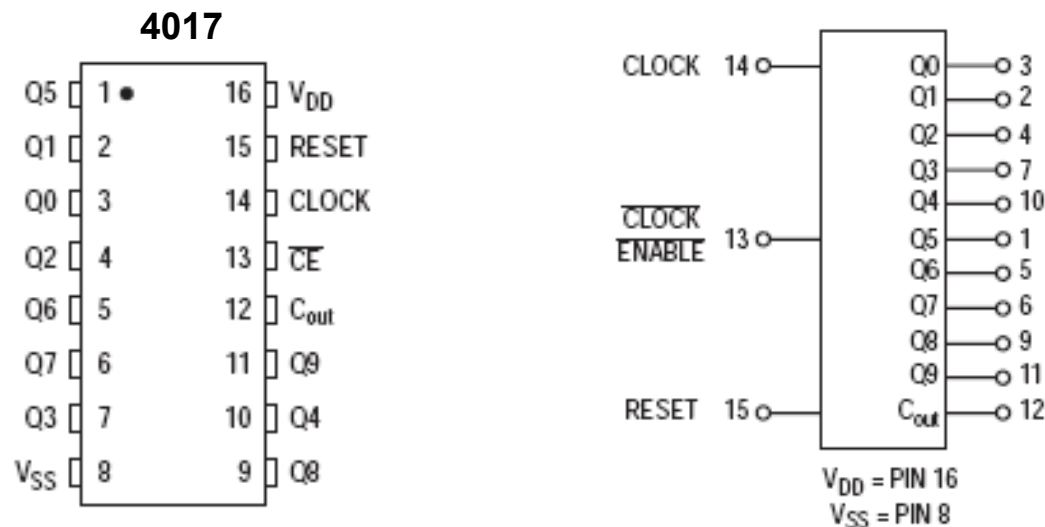


* $t_{WY} \cong 900$ ns @ $V_{DD} = 5$ V

Contador decimal. 4017.

Tiene 10 salidas que se van activando de a una por vez, según la ocurrencia de un flanco activo en la entrada de Clock.

Posee Reset, Habilitación de clock, y posibilidad de expansión (C_{out})



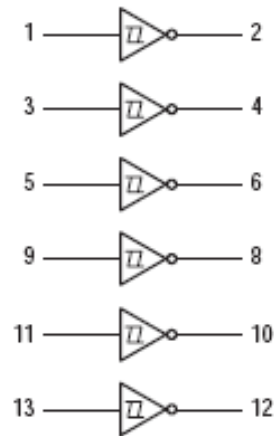
Clock	$\overline{\text{Clock Enable}}$	Reset	Decode Output=n
0	X	0	n
X	1	0	n
X	X	1	Q0
$\nearrow$	0	0	n+1
$\searrow$	X	0	n
X	$\nearrow$	0	n
1	$\searrow$	0	n+1

X = Don't Care. If $n < 5$ Carry = "1", Otherwise = "0".

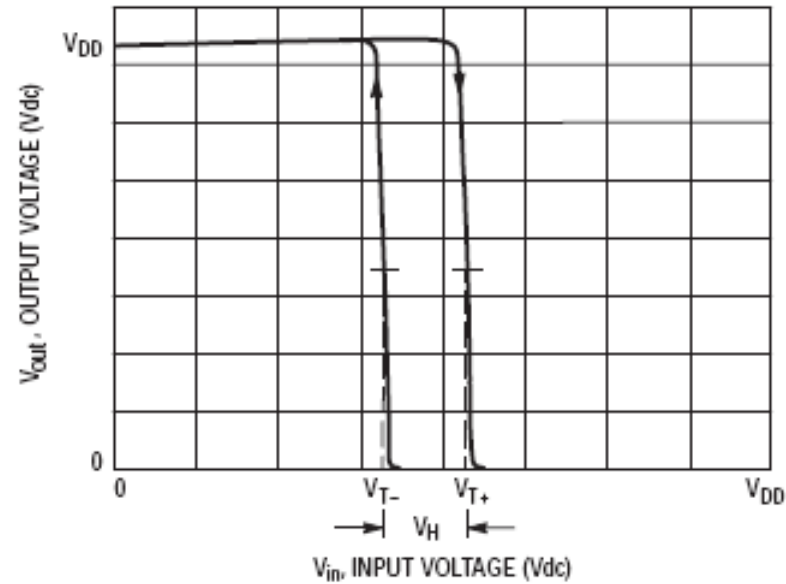
Tabla de salidas

Q9	Q8	Q7	Q6	Q5	Q4	Q3	Q2	Q1	Q0
0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0

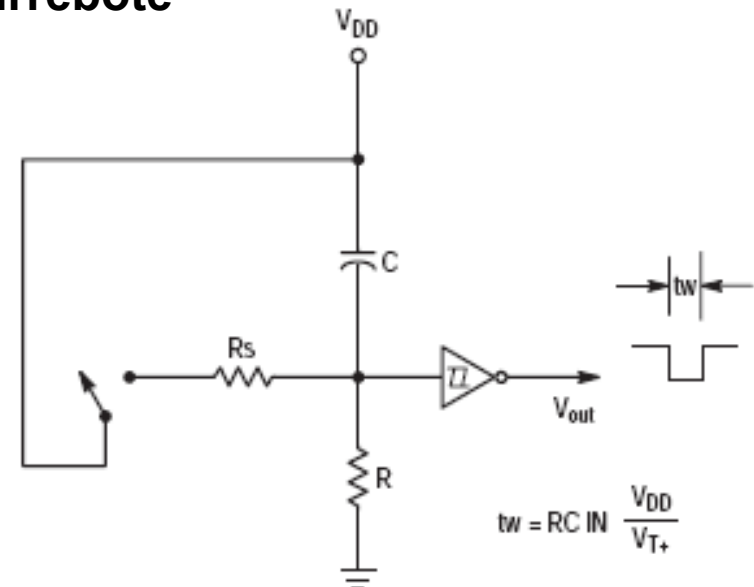
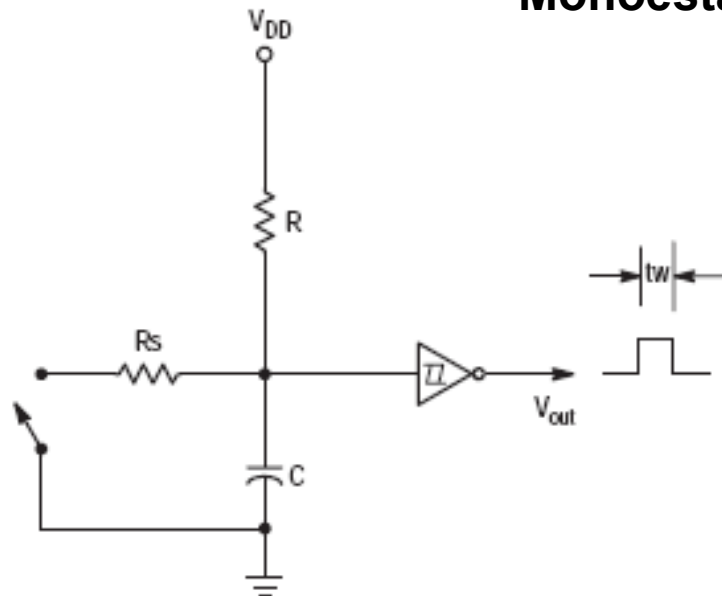
Schmitt Trigger. 4106.



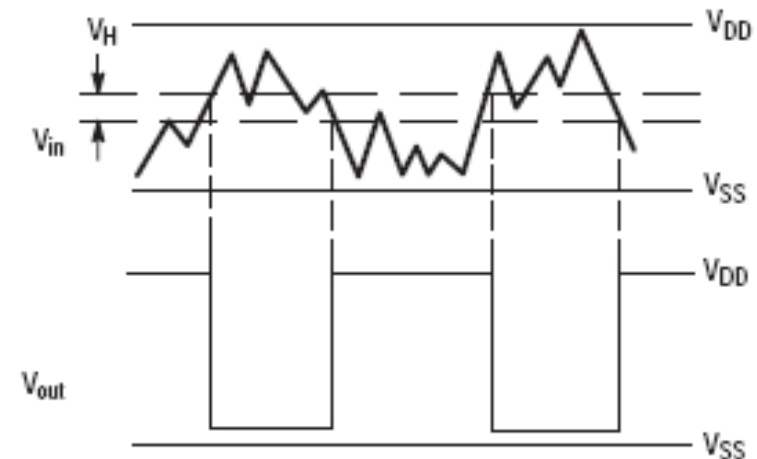
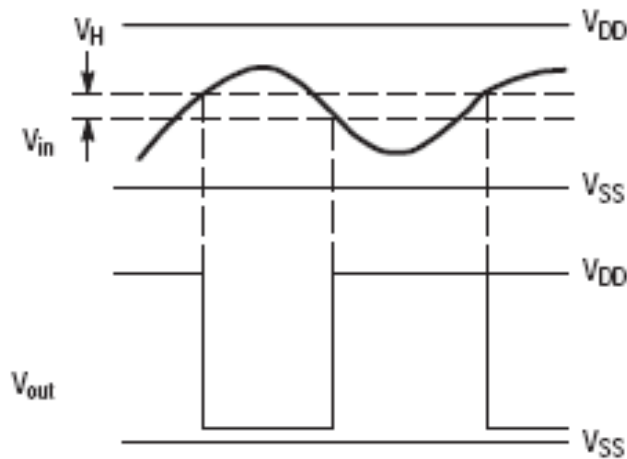
V_{DD} = PIN 14
 V_{SS} = PIN 7



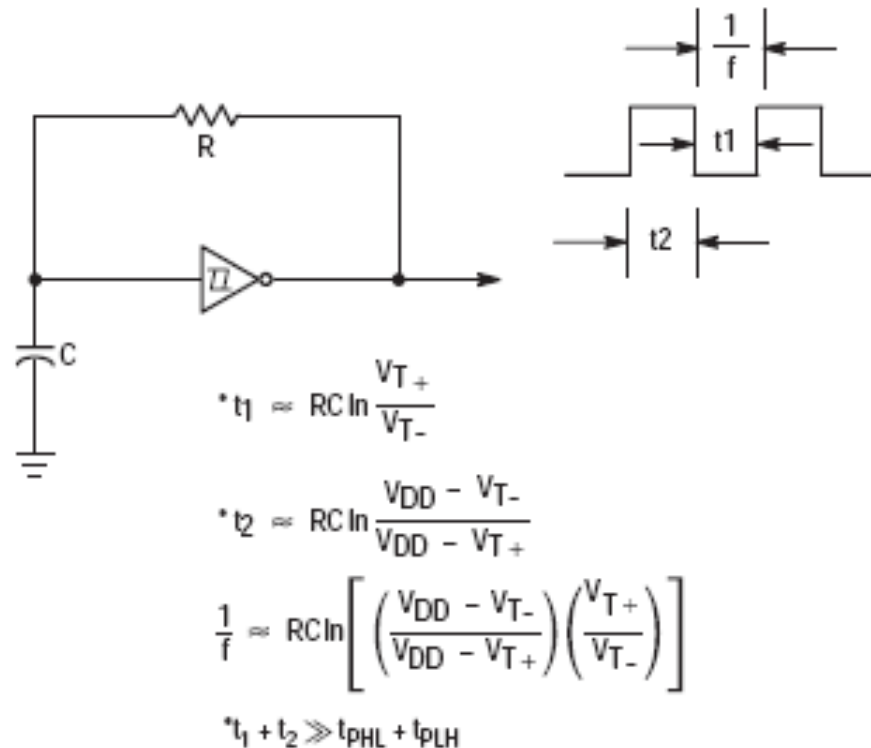
Monoestables - Antirrebote



Schmitt Trigger. 4106.

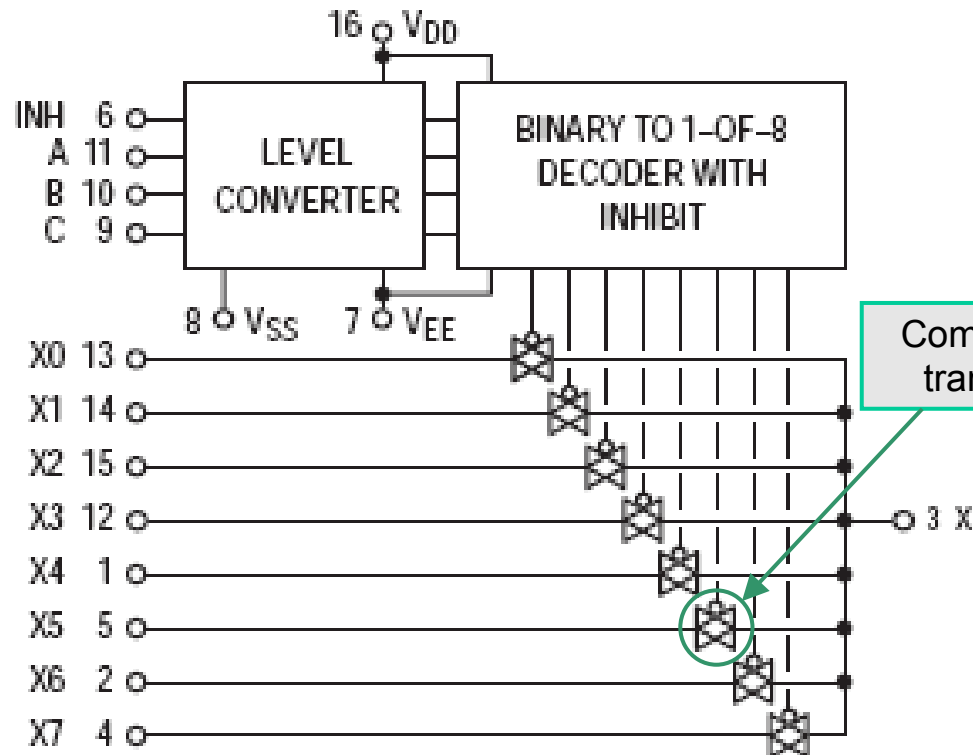
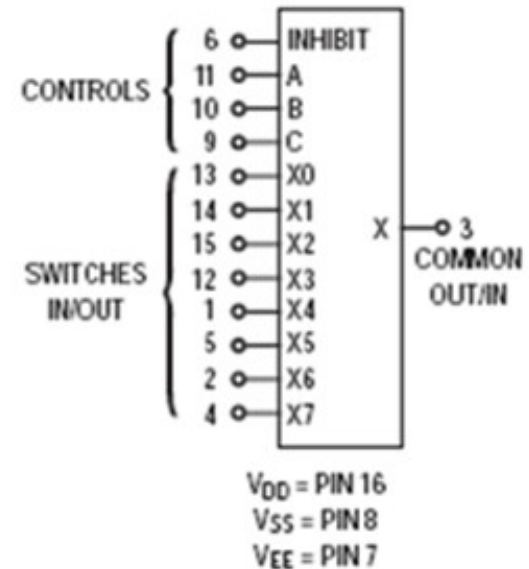
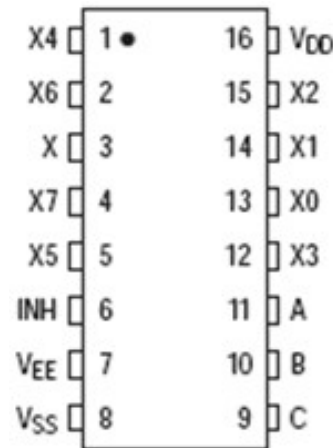


Astable



Multiplexor-Demultiplexor analógico. 4051.

Permite seleccionar señales analógicas (ej.: audio) o digitales. Son bidireccionales.



Control Inputs				
Inhibit	Select			MC14051B
	C*	B	A	
0	0	0	0	X0
0	0	0	1	X1
0	0	1	0	X2
0	0	1	1	X3
0	1	0	0	X4
0	1	0	1	X5
0	1	1	0	X6
0	1	1	1	X7
1	x	x	x	None