UNIDAD 3: ELECTRÓNICA DIGITAL

3.A. Funciones lógicas.

Constantes y variables Booleanas.

En el álgebra booleana las variables solo pueden tomar dos valores posibles: 0 y 1. Estos se conocen como **niveles lógicos**. Al nivel lógico 0 se lo denomina a también: falso, apagado, abierto, bajo, y al nivel 1: verdadero, encendido, cerrado, alto. Electrónicamente se representan mediante niveles de voltaje, tradicionalmente los "niveles TTL" 0 V y 5 V respectivamente, aunque con la evolución de la tecnología digital es común circuitos de 3,3V , 2,7V, 1.8V. Los niveles tienen un margen de tolerancia, según la tecnología, por ejemplo para TTL, de 0 a 0,8 es un '0' lógico y mayor de 2V es un '1' lógico. Para tecnología CMOS, de 0 a 0,3xVdd es un '0' y mayor de 0,7xVdd es un '1' lógico (siendo Vdd el voltaje de alimentacióon).

Se utiliza el álgebra booleana como medio para representar la entrada y la salida en un circuito lógico. Las entradas suelen ser designadas por letras.

Funciones lógicas.

Cualquier circuito lógico puede describirse por completo mediante el uso de las tres operaciones básicas.

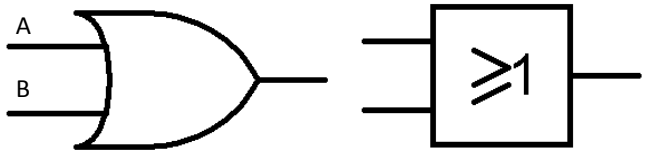
Postulados básicos.

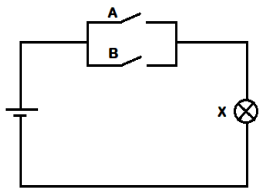
Compuertas lógicas.

1. **Operación OR**

La tabla de verdad muestra la operación OR: "x" es un 1 lógico cuando una o más entrada vale 1. "x" se vuelve 0 únicamente cuando tanto "A" como "B" están en 0. Se puede decir que basta que una de las entradas sea verdadera para que la salida sea verdadera.

La expresión booleana para la operación OR es: $\boldsymbol{x=A+B}$

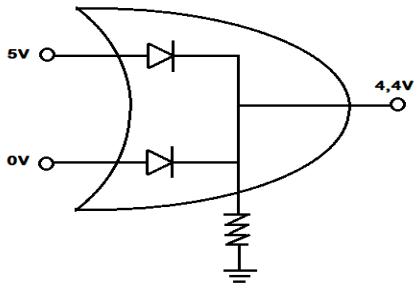




B	A	X
0	0	0
0	1	1
1	0	1
1	1	1

Símbolo IEEE	Símbolo IEC	Lógica de llaves	Tabla de verdad

La implementación electrónica más sencilla de una compuerta OR es con diodos, como muestra la figura. Basta con que a una de las entradas se le aplique tensión para que en la salida haya tensión. La resistencia en la salida se denomina *pull down* ya que "tira hacia abajo" el potencial. Suele ser de entre 1000 y 10.000 ohms.



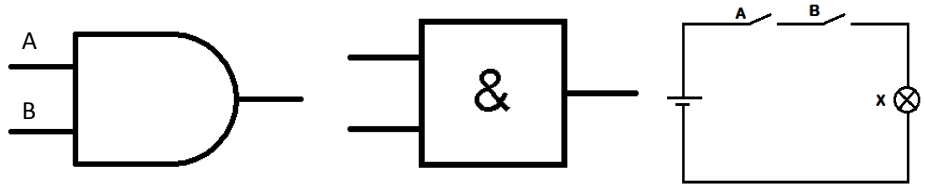
Es importante señalar que, en la tabla de verdad, las combinaciones de las posibles entradas **deben seguir un orden determinado** ya que la salida es una identidad de la función. Las entradas deben agregarse de derecha a izquierda, quedando el mayor subíndice o ultima letra en la primera columna a la izquierda. Las entradas se nombran con letra mayúscula y cuando son varias del mismo tipo se las denomina con la misma letra y distinto subíndice.

Se debe tener en cuenta que la cantidad de combinaciones posibles es 2^E siendo E la cantidad de entradas. Para completar la tabla la lógica es la siguiente. La columna de la primera entrada se completa alternando 0 y 1. La columna de la segunda entrada se completa alternando a la mitad de la frecuencia, es decir dos ceros, luego dos unos. La columna de la tercera entrada a la mitad de la frecuencia anterior, es decir 4 ceros y 4 unos. Y así sucesivamente.

2. **Operación AND**

La tabla de verdad muestra la operación AND. "x" es un 1 lógico cuando tanto "A" como "B" se encuentran en nivel 1. Para cualquier caso en donde "A" o "B" valgan 0 entonces "x" valdrá 0. Se puede decir que, para que la salida sea verdadera, todas las entradas deben ser verdaderas. O, que basta que una de las entradas sea falsa para que la salida sea falsa.

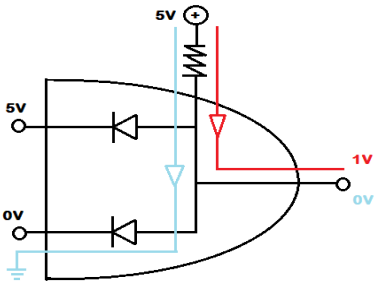
La expresión booleana para la operación AND es: $\boldsymbol{x=A.B}$



B	A	X
0	0	0
0	1	0
1	0	0
1	1	1

Símbolo IEEE	Símbolo IEC	Lógica de llaves	Tabla de verdad

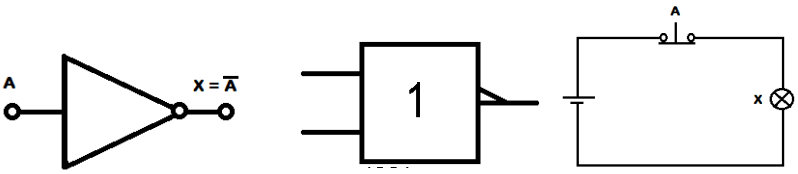
La implementación electrónica más sencilla de una compuerta AND es con diodos. En color rojo, la tensión sigue su camino hacia la salida obteniéndose un 1. Es importante señalar que aplicar 0V a una entrada, es mandarla a masa, no dejarla abierta. La resistencia a la salida se denomina *pull up* ya que "tira hacia arriba" el potencial. Suele ser de entre 1000 y 10.000 ohms.



Si una de las entradas está a '0' (a masa), su correspondiente diodo quedará polarizado en directo y conducirá, por lo que en la salida (punto entre la resistencia R y el diodo) la tensión caerá a 0,6 volts, es decir un '0' lógico.

3. **Operación NOT**

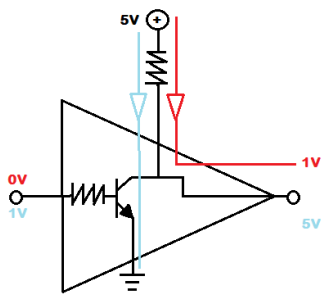
La operación NOT se realizar sobre una sola variable: "x" será el inverso, el complemento o el valor opuesto de "A". **X= A***(usaremos una línea sobre la letra, o un asterisco como superíndice)



A	X
0	1
1	0

Símbolo IEEE	Símbolo IEC	Lógica de llaves	Tabla de verdad

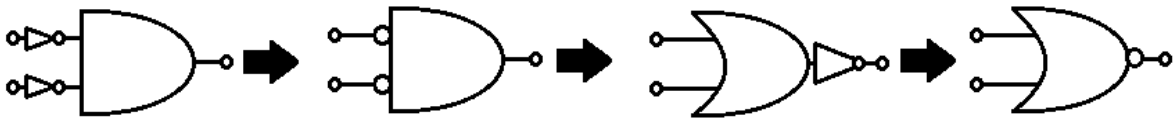
En la compuerta NOT también es necesaria una fuente de alimentación. Cuando se aplican 5V en la entrada, es decir un 1, el transistor se satura y actúa como llave cerrada por lo que la tensión se descarga por la resistencia a masa. Esto se observa en el camino azul de la imagen. Al aplicar 0V, el transistor actúa como llave abierta y la tensión llega a la salida. Esto se observa en el camino rojo de la imagen.



Propiedades de las compuertas.

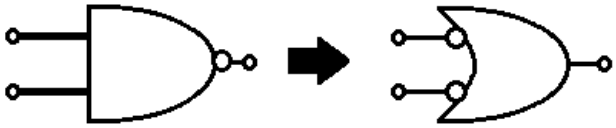
1. Cuando dos tablas de verdad, con las combinaciones construidas de la misma manera , son idénticas, es decir, poseen salidas idénticas, las funciones son equivalentes aunque internamente estén compuestas por compuertas distintas.
2. A+0=A
3. A+1=1
4. A.0=0
5. A.1=A
6. $A+A^{*}=1$
7. $A.A^{*1}=0$
8. $A^{*}.B^{*}=(A+B)^{*}$
9. $A^{*}+B^{*}=(A.B)^{*}$

Estas dos últimas propiedades se utilizan para sustitución de compuertas. Por ejemplo, en el caso de la propiedad 8 sería:



Esta compuerta es la compuerta NOR. Es la composición de una compuerta OR con un inversor en la salida y es equivalente a una compuerta AND con un inversor en cada entrada.

Lo mismo ocurre con la propiedad 9 una compuerta NAND y una OR con entradas negadas son equivalentes.

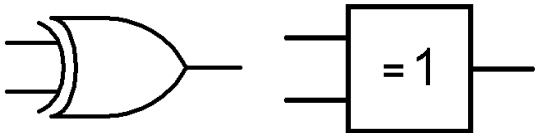


Las funciones OR, AND y NOT son funciones primitivas, mientras que la NAND y la NOR son funciones derivadas.

4. Operación XOR (or exclusiva)

Una de las compuertas derivadas más importantes es la OR EXCLUSIVA o XOR. La salida es la suma exclusiva de las entradas. La salida es cero cuando la cantidad de 1 es par.

La expresión booleana para la operación XOR es: $x=A \oplus B$

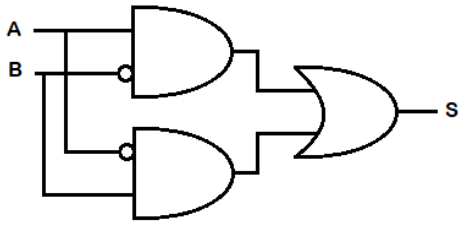


B	A	S
0	0	0
0	1	1
1	0	1
1	1	0

La OR EXCLUSIVA es una compuerta que puede armarse con compuertas AND y compuertas OR con lo que se llama SUMA DE PRODUCTOS.

A . B^* es 1 cuando A=1 y B=0.

A^* . B es 1 cuando A=0 y B=1.



La función estará formada por dos subfunciones:

$S= A . B^* + A^* . B = A \oplus B$

Obtengo una compuerta OR que conecta las salidas de dos compuertas AND con sus respectivos inversores.

3.B. Circuitos combinacionales

Las compuertas se pueden interconectar para que ante combinaciones de variables lógicas de entrada, se activen salidas siguiendo una lógica de funcionamiento deseada. Vemos a continuación algunas aplicaciones:

Síntesis de una función lógica a partir de la Tabla de Verdad:

Uno de los procedimientos para obtener la función a partir de la tabla de verdad es el siguiente: Se marcan aquellas combinaciones de entrada para las que la salida es **'1'**. Por cada combinación de entradas obtendré un término en la suma. Dentro de cada termino, se colocan negadas aquellas entradas que valen cero y sin negar las que valen uno.

Ejemplo: Si queremos que la salida S se active en las combinaciones indicadas (1, 4, 5 y 7), se realiza una suma (OR) de 4 términos, que serán funciones AND de las entradas A, B y C.

	C	B	A	S
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

La función que se activa en la combinación 1 es A.B*.C* (es '1' cuando A=1, B=0 y C=0)
Para la combinación 4 es:
A*.B*.C (es '1' cuando A=0, B=0 y C=1)
Para la combinación 5 es:
A.B*.C (es '1' cuando A=1, B=0 y C=1)
Para la combinación 7 es:
A.B.C (es '1' cuando A=1, B=1 y C=1)
Luego para que S se active en 1,4, 5 y 7:
S= A.B*.C*+ A*.B*.C + A.B*.C + A.B.C

Si se observa, la función XOR se puede obtener de este modo.

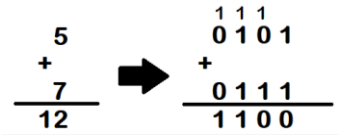
Generador de paridad.

La codificación de la unidad de información en las computadoras es el byte (1byte = 8 bits). Cuando la información se transmite en serie, se transmite de una computadora a otra bit por bit los paquetes de bytes. Una de las formas de verificar que la información llego correctamente es agregarle un bit extra al paquete de 8 bits, llamado bit de paridad. Entonces con este bit uno puede asegurarse de que la cantidad de '1' del byte sea por ejemplo, siempre par. Esta es una convención entre el equipo que recibe y el que transmite. Si, por ejemplo, por ruido eléctrico **uno** de los valores cambia durante la transmisión, la paridad con la que fue generado se va a romper, y el receptor podrá detectar el error.

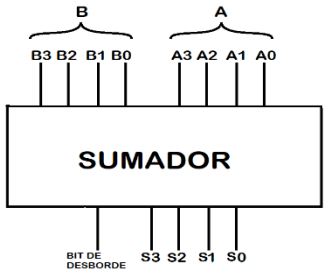
El generador de paridad puede ser E (even - par), O (odd . impar) o N (sin paridad).

Sumador.

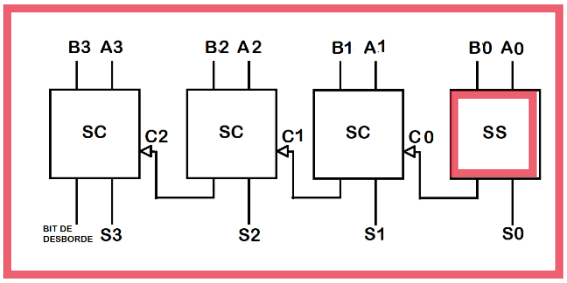
La idea central del sumador es, como su nombre lo indica, realizar la suma aritmética de números representados en binario.



La mecánica es igual que para decimales solo que en este caso el máximo número por columna es 1 y el excedente se acarrea.



Para la implementación del circuito tendremos dos conjuntos de entradas: Uno A y uno B para formar cada uno de los operandos. Si cada conjunto está formado por 4 bits podremos ingresar los números del 0 al 15. La salida tendrá un bit más, llamado bit de desborde, ya que el máximo resultado de la suma es 30.



Cada uno de los dígitos del resultado es la función suma de dos de los bits de las entradas. En la primera operación será la suma de dos bit pero a partir de la segunda deben sumarse 3 bit, ya que se agrega el acarreo de la operación anterior. Por lo que cada operación tendrá dos salidas, una de las cuales es el acarreo que estará conectado como entrada a la operación siguiente.

Estas operaciones se llevan a cabo en subsistemas: el sumador completo SC y el semisumador SS. El SS es un combinacional con 2 entradas y dos salidas y el SC posee 3 entradas y 2 salidas.

Implementación del semisumador

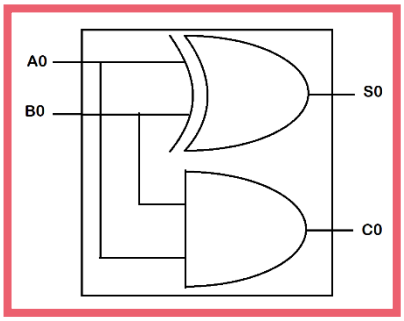
La tabla de verdad correspondiente es la siguiente.

B0	A0	S0	C0
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Entonces:

$S0 = B0^{*} . A0 + B0 . A0^{*}$

$C0 = B0 . A0$



Podemos reconocer esta función como una OR Exclusiva y la función de Carry (acarreo) como una AND.

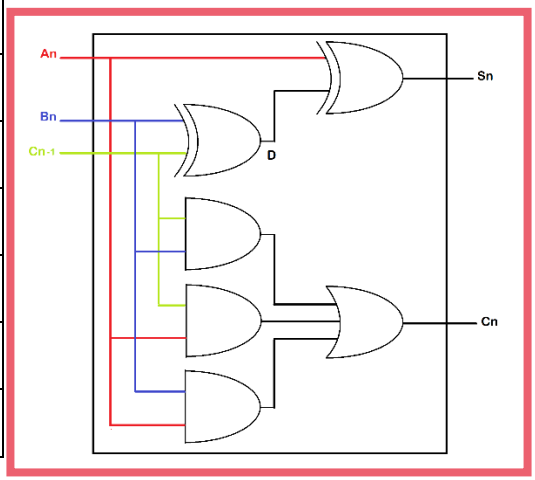
Implementación del sumador completo.

El sumador completo es un bloque con dos funciones de 3 entradas cada una.

La tabla de verdad correspondiente es la siguiente.

Cn-1	Bn	An	Sn	Cn
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

En principio, según la tabla, deberíamos usar 4 compuertas AND para cada salida S y C pero existen técnicas de minimización basadas en las propiedades de las funciones que permiten utilizar menos compuertas.



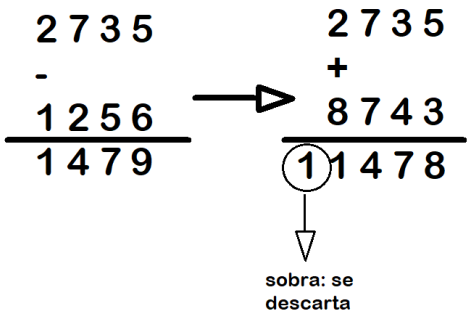
Nota: Este es un circuito que con los circuitos LUT (*look up table* o tabla de búsqueda) ha perdido sentido. La tabla permite buscar rápidamente un resultado almacenado mientras que el sumador debe realizar la operación cada vez que se ingresen los operandos. Es decir, que el sumador debe esperar a que el resultado se propague hasta el último sumador.

Restador.

La resta, electrónicamente, es más compleja que la suma.

Si analizamos cómo resolvemos una resta, cuando “el de arriba” es más chico que “el de abajo”, el primero “le pide uno” al de “al lado” y éste último debe “saber” que se le restó uno.

Para resolver ésta situación se aplica una técnica que se llama complemento a la base.



El complemento a la base menos 1, en sistema decimal siendo la base 10, es el complemento a 9. Es lo que le falta a cada digito del número que resto para llegar a 9. Se reemplaza cada digito por su complemento a 9 y, en lugar de restarlo, se lo suma.

El resultado obtenido es muy similar pero no igual. Siempre va a faltar una unidad y esto se corrige sencillamente sumando uno. Además, el primer digito debe truncarse.

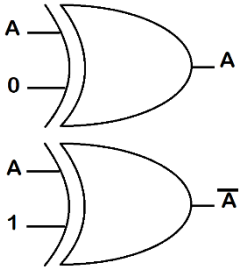
En números binarios esto es muy sencillo ya que el complemento a 1 de cualquier número es directamente el número inverso. Obsérvese el ejemplo.

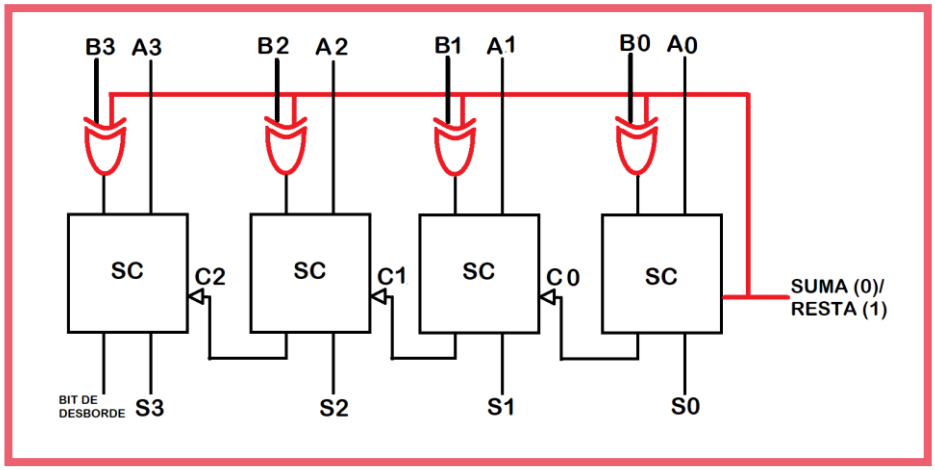
Sistema decimal	Sistema binario		
10	1010	Aplicando	1010
-	-	complemento	+
6	0110		1001
4	0100		10011
			+
			0001
			0100

Implementación del circuito

Sencillamente se puede resolver agregando inversores en las entradas del número binario B y sumándole uno al resultado o directamente al primer semisumador se le asigna un uno volviéndolo un sumador completo.

Al mismo circuito se le puede agregar un inversor a B si se quiere restar y quitarlo si se quiere sumar. Se puede aprovechar la propiedad de la compuerta or exclusiva de que, si una de las entradas es cero, en la salida tendré el valor de la otra variable, pero si es uno, tendré el inverso de la otra variable.





Comparador.

El circuito comparador revela cuál es la condición entre dos números (mayor, igual o menor). A continuación, se muestra su tabla de verdad para 2 bits.

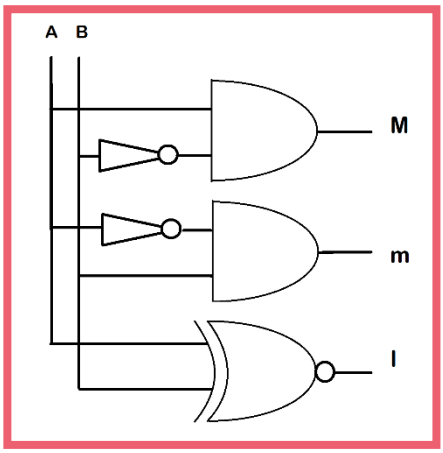
B	A	M	m	I
0	0	0	0	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

Con el método de suma de productos, las salidas son:

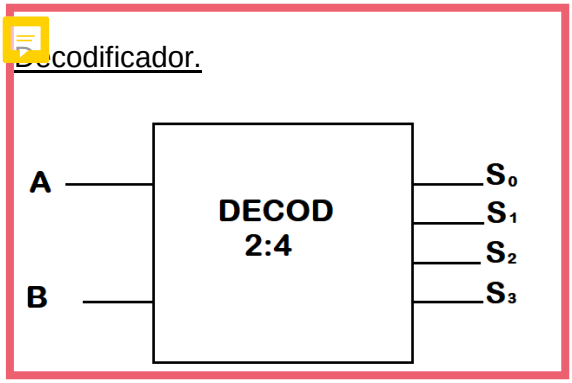
$$M = A\bar{B}$$

$$I = \bar{A}\bar{B} + AB$$

$$m = \bar{A}B$$

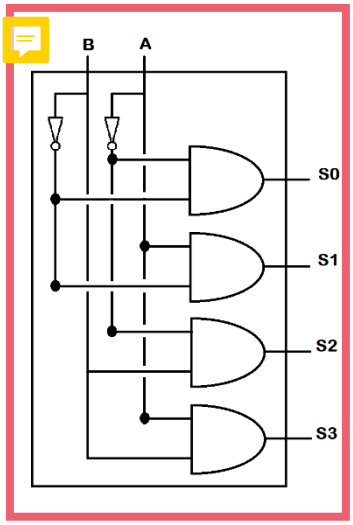


Decodificador.



Bin	B	A	S0	S1	S2	S3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
2	1	0	0	0	1	0
3	1	1	0	0	0	1

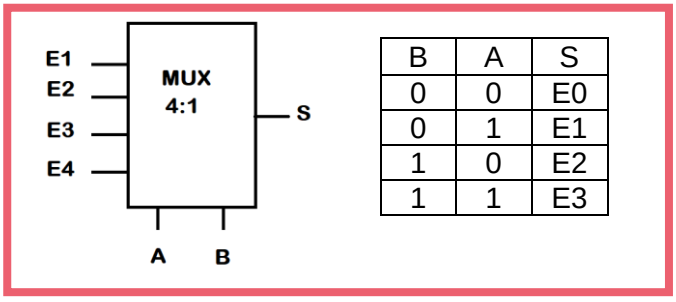
Es un sistema que tiene n entradas y 2^n salidas. Todas las salidas deben estar representadas en la tabla de verdad.



Las combinaciones de entradas corresponden a un número binario y cada salida se activa cuando su número binario correspondiente es ingresado en la entrada mientras que las demás permanecen desactivadas. Funciona como un selector de activación permitiendo acceder a los datos de las memorias.

Para obtener el circuito correspondiente a esta tabla de verdad observamos que la combinación de salidas de la S3 es idéntica a la de la función AND. Siguiendo el mismo procedimiento que para obtener la compuerta exclusiva, concluimos que el circuito será el siguiente.

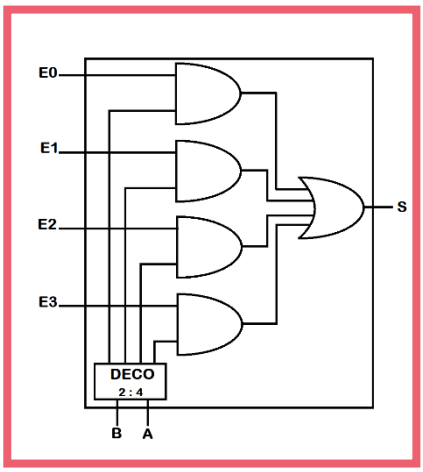
Multiplexor.



B	A	S
0	0	E0
0	1	E1
1	0	E2
1	1	E3

Es un circuito que permite que la salida S adopte el valor lógico de una entre 2^n entradas Ei, en función de los bits de selección A y B. Funciona como una llave selectora que pemite elegir cuá de las entradas estará presentes en la salida dependiendo de la combinación de A y B.

Para confeccionar la tabla de verdad se debe tener en cuenta que, en este caso, las entradas son 6: E0, E1, E2, E3, A y B. es decir la cantidad de combinaciones posibles a la salida son 2^6=64. Confeccionar la tabla de verdad completa sería muy extenso y se perdería de vista la función del multiplexor. Por esto se usa una tabla de verdad condensada o reducida.



Para el selector usamos un decodificador.

Si analizamos el comportamiento de este circuito se observa que el decodificador activara solo la salida indicada por A y B, es decir, solo esa salida valdrá uno y las demás valdrán cero. Por las propiedades enunciadas anteriormente, sabemos que aquellas compuertas AND que tengan una entrada cero, sin importar el valor de la otra entrada, la salida será cero y la que tenga un 1 en una entrada, la salida reflejará el valor de la otra entrada. De esta manera en la salida se reflejara el valor de la entrada seleccionada con el decodificador.

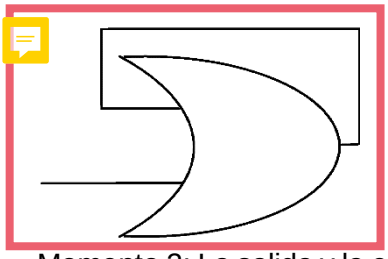
3.C. Circuitos secuenciales.

Los circuitos vistos hasta ahora tienen la característica de que, siempre que ingrese la misma combinación de entradas, en la salida se presentará el mismo valor. Por eso se denominan **combinacionales**.

En el caso de los circuitos **secuenciales** mediante el uso de realimentaciones desde una etapa hacia otra etapa anterior, se logra que la salida dependa no solo de los datos ingresados sino también del estado previo del sistema. Esto permite almacenar estados que van cambiando con el tiempo.

Los biestables son dispositivos capaces de almacenar un estado que puede ser alto o bajo.

Biestables: SR básico.

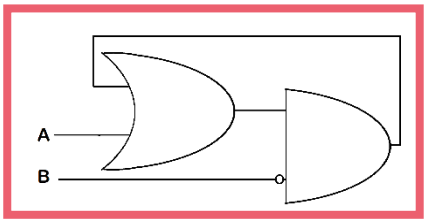


Se comenzará analizando un circuito muy simple: una compuerta OR con su salida realimentada hacia una de las entradas.

Momento 1: Suponiendo que la salida está en cero, la entrada superior toma el mismo valor. Al ingresar un cero por la entrada inferior, la salida continuará siendo cero y el estado se mantiene.

Momento 2: La salida y la entrada superior, como se dijo, están en cero. Se ingresa por la entrada inferior un uno, este se refleja en la salida y es realimentado a la entrada superior.

Momento 3: Al ingresar un cero por la entrada inferior nuevamente, como la entrada superior está dominada por la salida, la salida continúa valiendo uno. La compuerta “ya tiene historia”.



Al circuito de salida se le intercala una llave que abra el circuito para que deje de enviar el uno y envie un cero para poder volver a utilizar la compuerta. Para esto se utiliza una compuerta AND con la entrada inferior negada

De esta forma, si B=0, en la entrada de la AND se tendrá un 1 y lo que haya en la otra entrada va a pasar. Es decir que cuando B=0, el circuito es equivalente al anterior. Pero cuando B=1, por el inversor, la entrada valdrá cero, abriendo el circuito ya que, sin importar el valor que ingrese a la OR, en la salida siempre habrá un cero.

El estado normal de B será cero, es decir que estará apagado, dejando pasar la realimentación y al ingresar un uno por la entrada A este quedara memorizado. Cuando B=1, se cortará la realimentación. Lo primero se denomina **enclavamiento** y lo segundo **desenclavamiento**.

Los ceros y los unos se ingresan con pulsadores normal abierto (NA, como el de la imagen) o pulsadores normal cerrado (NC). El primero (NA) al pulsarlo **conecta** y el segundo (NC) **desconecta**. Una llave también se denomina pulsador con retención y tiene dos estados estables posibles. Es importante recordar que ingresar un cero no es dejar la entrada abierta ya que con esto lo que se logra es que ingrese ruido y no está garantizado ninguno de los dos valores. Para ingresar un cero se debe conectar con una resistencia a masa. Estas resistencias no son necesarias si la etapa previa es otra compuerta dogotal, que asegura los niveles de tensión de '0' y '1'.

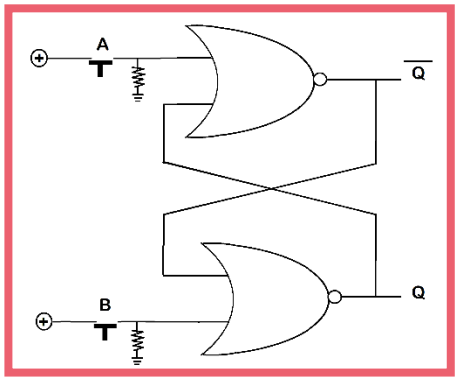
Este circuito es equivalente a dos compuertas NOR realimentadas mutuamente. Se demuestra de la siguiente forma. En la figura anterior:

1. Se colocan dos inversores seguidos entre la salida de la OR y la entrada de la AND. En principio el efecto es neutro.

2. Uno de los inversores se une a la compuerta OR formando una compuerta NOR.

3. Los dos inversores en la entrada de la AND, por la Ley de Morgan equivalen a una NOR.

4. Se observa que la salida de la NOR superior está conectada a la entrada de la inferior y viceversa.



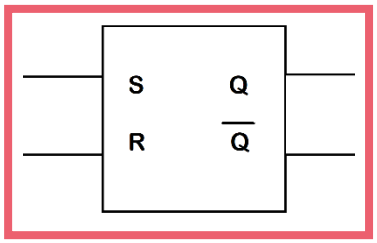
Al pulsar A, ingresa un 1 a la NOR superior, por lo que su salida se vuelve cero sin importar lo que haya en la otra entrada (al revés que con la OR). Si no se está pulsando B, entra un cero y por la otra entrada de retroalimenta el cero por lo que la salida de la NOR inferior será 1.

Este 1 se retroalimenta a la entrada de la NOR superior y mantiene la salida en cero aunque se suelte el pulsador.

Es decir: presionando A se logra que se encienda la salida de la NOR inferior (**Q**) y que la salida de la NOR superior (**!Q**) sea cero.

Si luego se acciona el pulsador B, de forma simétrica, la salida de la NOR inferior se vuelve cero y la NOR superior se vuelve uno.

Este biestable se denomina SET-RESET o SR o SR Asíncrono.



Es importante observar que las salidas están "cruzadas". Al presionar A o el SET se activa la salida inferior y al presionar B o el RESET se activa la superior. En el esquema las salidas se representan invertidas para que haya una correspondencia lógica. Lo que puede hacerse en el circuito es "cruzar los cables" por dentro.

S	R	Q
0	0	Q t-1
0	1	0
1	0	1
1	1	X

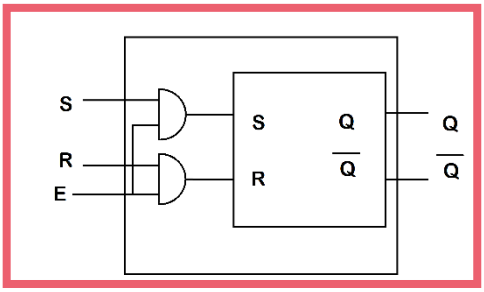
En la tabla de verdad solo se representa la salida Q ya que se asume que la otra simplemente tomara el valor opuesto.

Q t-1 es el estado anterior y X significa que hay una indeterminación: no puede saberse con seguridad que valor tomará la salida.

Este biestable permite guardar información: 1 bit. Esta es la unidad mínima de la memoria RAM. Al agrupar 8 se forma un byte. A diferencia de la memoria ROM, esta información se puede escribir y borrar. Además, es el núcleo de todos los biestables.

Biestables: SR activado por nivel.

También se denomina SR Síncrono.



En este caso se tienen 2 salidas pero 3 entradas.

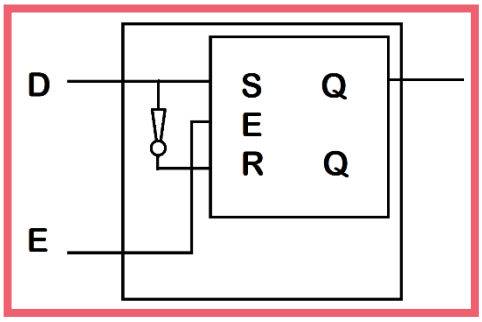
Si E=1, las AND dejan pasar los valores de S y R pero si E=0, sin importar los valores de S y R, no puede cambiarse la salida.

La entrada ENABLE se utiliza para sincronizar el dispositivo al momento de operarlo.

E	S	R	Q
0	X	X	Q t-1
1	0	0	Q t-1
1	0	1	0
1	1	0	1
1	1	1	X

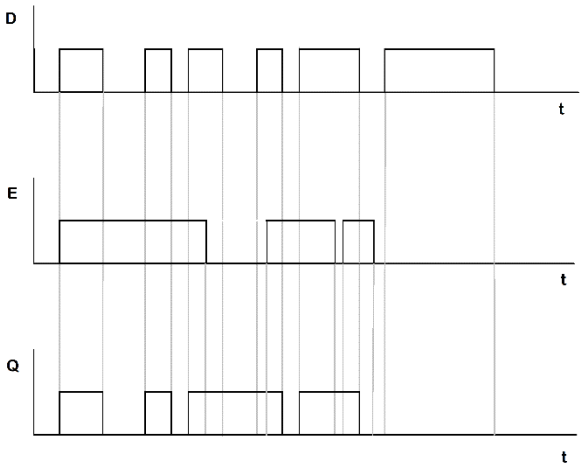
En la tabla de verdad se encuentran X en las entradas. Esto significa que no interesa el valor de la entrada.

Biestables: D activado por nivel.



Cuando E=0, mantiene el dato memorizado y no toma el dato de entrada. Cuando E=1, Q toma el valor de la entrada D, es decir: se puede escribir.

Una gran ventaja es que se anula la posibilidad de que exista la indeterminación de tener dos 1 en las entradas.



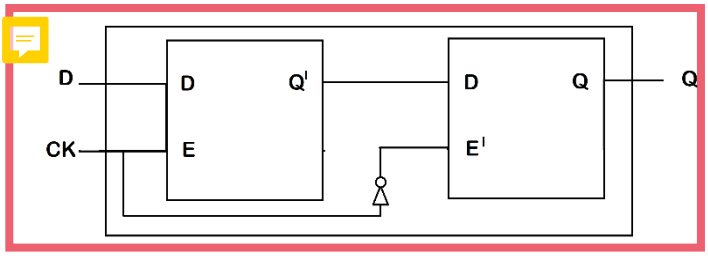
E	Q
0	Q t-1
1	D

Se puede decir que D_E es un circuito transparente si E=1. Es decir, que, si D cambia, Q cambia de la misma manera. Pero, cuando E=0, el ultimo valor de D queda memorizado en Q.

Biestables: D activado por flanco (maestro esclavo).

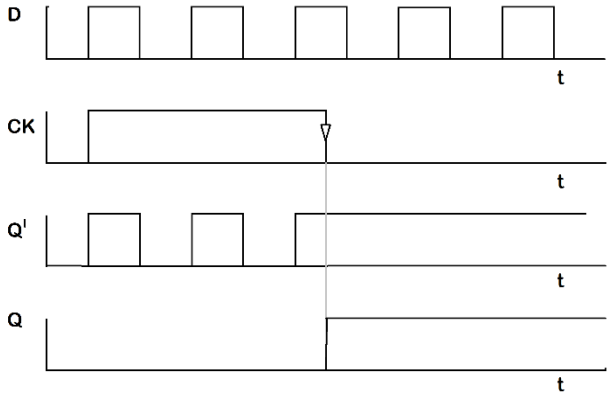
Este biestable sirve para aquellas aplicaciones en que se necesita que el dato se almacene en un instante único denominado flanco activo. Puede ser flanco de subida o flanco de bajada.

El biestable D_{MS} tiene dos entradas: el Clock y la entrada de datos.



Internamente está constituido por dos biestables D activados por nivel. Las habilitaciones de los biestables (E y E') están complementadas mediante el inversor. Cuando la del primero biestable vale 1, dejando pasar el dato, la del segundo vale cero y no permite que sea escrito permaneciendo en un estado de retención del dato anterior.

Cuando el clock pase a valer 0, será al revés. El primer biestable estará bloqueado y no se dejará escribir, pero el segundo sí.



El objetivo es que mientras el clock valga 1, aunque la entrada D esté cambiando, la salida permanezca constante. En el momento en que el clock pase a valer cero, la salida Q' quedará con el ultimo valor de D retenido y el segundo enable E' habilitará que Q tome éste último valor.

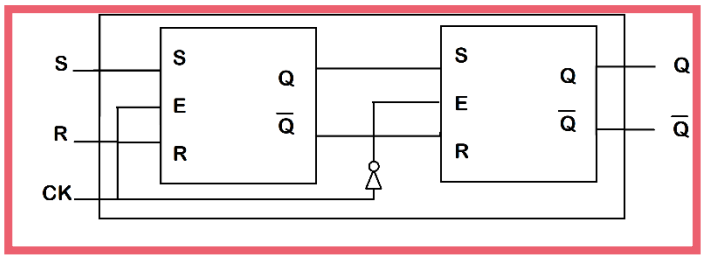
CK	Q
↓	D
X	Q_{T-1}

En resumen: la transferencia de datos de la entrada a la salida ocurre en el momento en que el clock pasa de valer 1 a 0. Es decir, su activación es por flanco de bajada.

En un segundo momento, aunque el segundo biestable sea transparente, es decir, dejando pasar el dato, el primero estará bloqueado y no importará que varíe D.

De haber un nuevo flanco de subida, es decir que el CK pase a valer 1, el primer biestable volverá a ser transparente pero el segundo estará bloqueado sin que haya transferencia del dato.

Biestables: SR *master-slave*.



Éste no es un biestable que se utilice, si no que se estudia porque es un intermedio para analizar el biestable J_K.

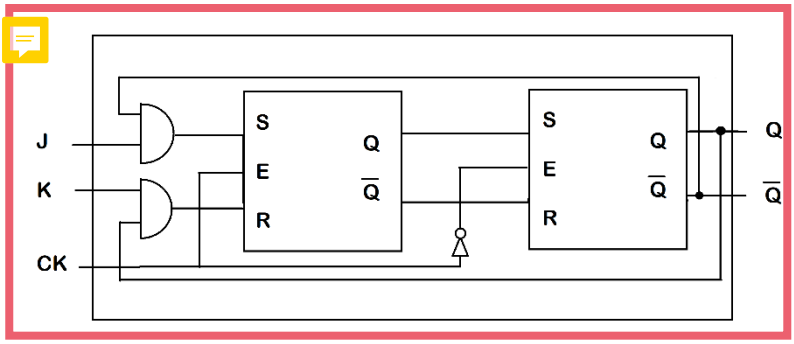
De la misma forma que para el biestable D_{MS}, se combinan dos biestables SR activados por nivel.

CK	S	R	Q
↓	0	0	Q_{T-1}
	0	1	0
	1	0	1
	1	1	X
X	X	X	Q_{T-1}

El comportamiento es el siguiente, si CK=1, el primer biestable es transparente y el dato que se ingresa en la entrada se transfiere a la salida del primer biestable y a la entrada del segundo. En el momento en que se ingresa un cero en el CK, el primer biestable se bloquea y el ultimo valor que ingresó al mismo, es el que ingresa al segundo.

En este caso se mantiene el problema de la indeterminación al ingresar dos 1.

Biestables: JK *master-slave*.



Esta configuración salva la indeterminación del biestable ficticio SR_{MS}.

El objetivo es no permitir que ingresen dos unos en simultaneo. Para esto, se busca que, si Q está encendido, permita que se pueda apagar y que, si está apagado, que se pueda prender.

CK	J	K	Q
↓	0	0	Q_{T-1}
	0	1	0
	1	0	1
	1	1	$\overline{Q_{T-1}}$
X	X	X	Q_{T-1}

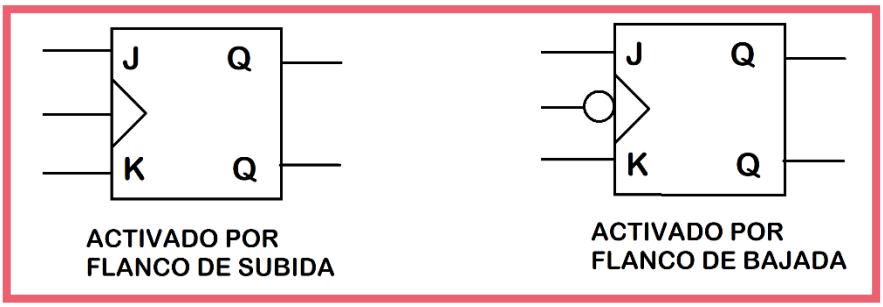
La idea es permitir mediante el uso de la realimentación con compuertas AND que se escriba 01 o 10 pero no ambas en simultaneo.

El comportamiento es el siguiente: si Q está encendido, K debe poder actuar para apagarlo y si está apagado, no es necesario que K actúe.

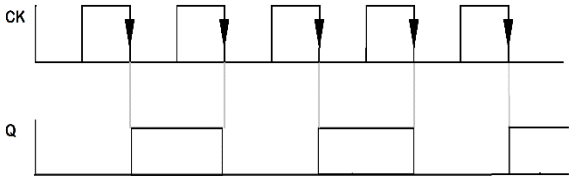
Cuando Q=0 y Q'=1, K no puede actuar sobre RESET, pero no importa porque ya está reseteado.

Al ingresar J=1 y K=1, en el flanco de bajada solo podrá pasar aquel que este apagado. Si Q=0, pasará J y lo encenderá y si Q=1, pasará K y lo apagará.

Los símbolos son los siguientes:



Como se observa en la tabla de verdad, en el flanco de bajada del CK, Q pasa a tener el valor opuesto al que tenía. El efecto obtenido es que a partir de una señal que tiene n pulsos/segundos, obtenemos una señal que tiene n/2 pulsos /segundos. Por lo que una de las aplicaciones de este biestable es como divisor de frecuencia.

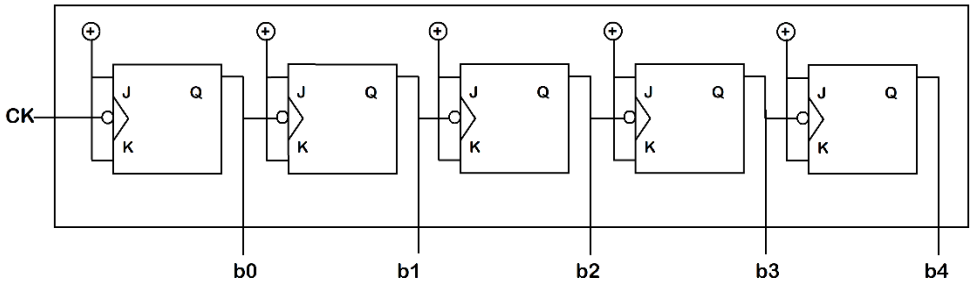


Registros.

Los registros son agrupaciones de biestables.

Contador asíncrono.

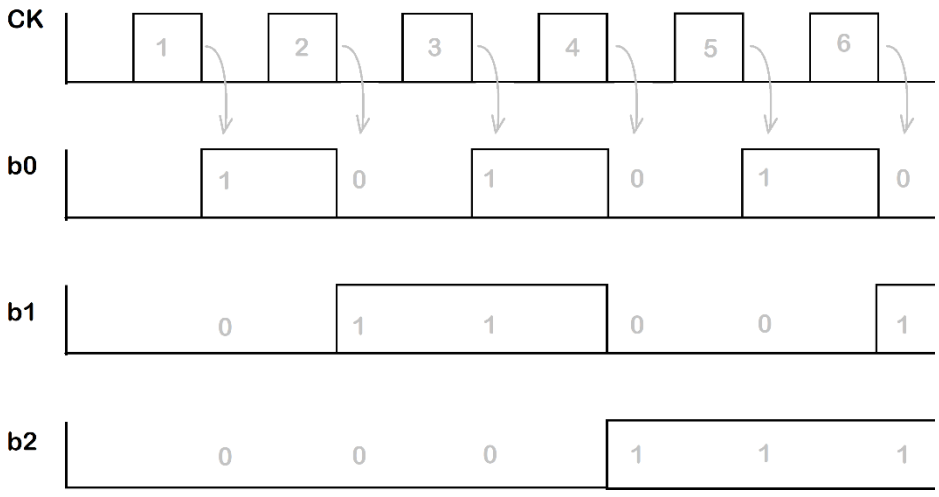
Es un contador binario. Consiste de una entrada donde ingresan pulsos y una serie de salidas denominadas bits y donde la onda de salida de un biestable funciona como el CK del siguiente biestable.



La combinación binaria de la salida depende de la cantidad de pulsos ingresados por el clock.

b4	b3	b2	b1	b0	Cantidad de pulsos
0	0	0	0	0	1
0	0	0	0	1	2
0	0	0	1	0	3
0	0	0	1	1	4
0	0	1	0	0	5
0	0	1	0	1	6

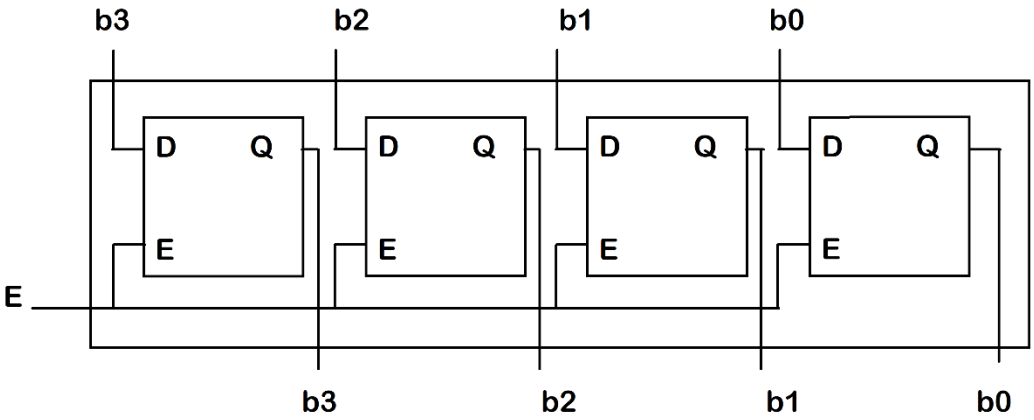
Se realiza la suposición de que, al comenzar, todos los bits están en 0. Al ingresar un pulso por el CK, el b_0 pasa a valer 1: número 1 en binario. Si se observa la tabla de numeración binaria, b0 cambia con cada flanco de bajada; b1 cada dos flancos de bajada; b2 cada 4 y asi sucesivamente. Si se lo interpreta en términos de pulsos, si b0 tiene una frecuencia; b1 tiene una frecuencia que es la mitad de la de b0 y así sucesivamente.



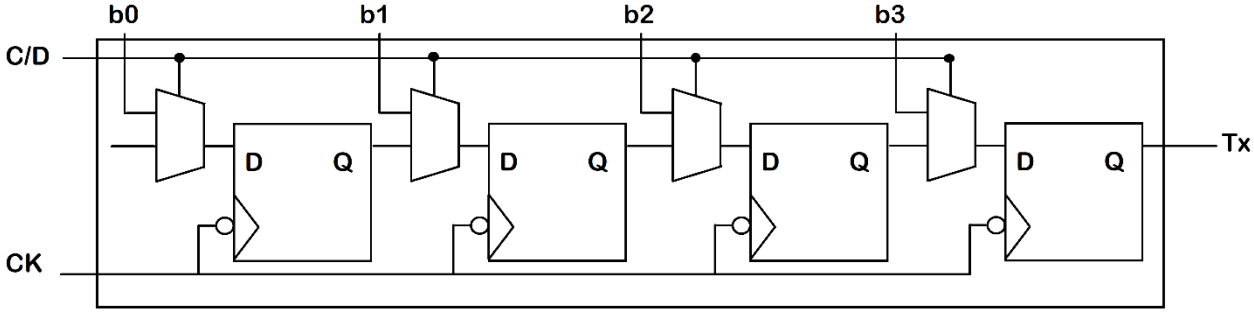
¿Qué ocurre cuando todos los bits coincidan en 1? El contador llega a su valor máximo posible y al siguiente pulso coinciden todos los flancos de bajada, reiniciándose en cero.

¿Se podría lograr que el contador cuente hacia atrás? Esto se logra utilizando JK activados por flanco de subida.

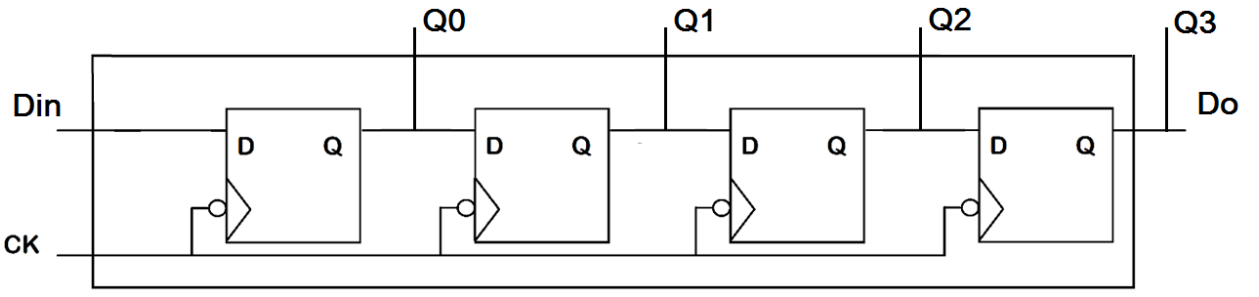
Registro paralelo-paralelo.



Registro de desplazamiento paralelo-serie.



Registro de desplazamiento serie-paralelo.



Aplicación en comunicación serie.

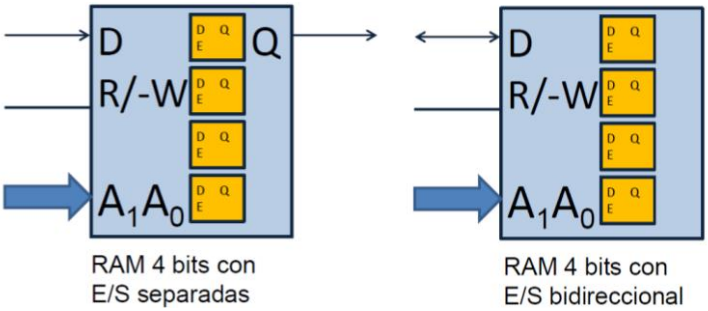
3.D. Memorias.

Memoria RAM estática de 4 bits.

La idea básica es poder almacenar o “escribir” un bit en un biestable determinado, seleccionado mediante una combinación binaria A1A0 denominada “dirección”, y luego poder recuperarlo. Con la entrada de control R/-W=0 se realiza la escritura, y con R/-W=1 la lectura.

En un esquema con entradas y salidas separadas el bit a escribir se presenta en la entrada D, y luego se lee por la salida Q.

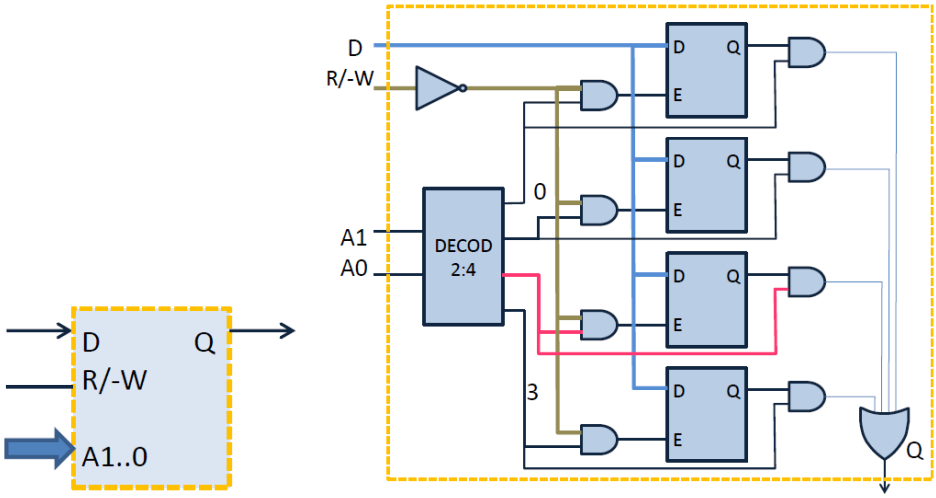
En un esquema con entrada y salida única (bidireccional), el bit se presenta para escribir o se lee por el mismo terminal. Es lo más habitual.



Operación de Escritura: 1) Poner dirección en A1A0 y dato en D.
2) Aplicar pulso negativo en R/-W
3) Ya se puede quitar el dato

Operación de Lectura: 1) Con R/-W en 1, poner dirección en A1A0. En Q se tendrá el dato direccionado. (En la memoria E/S bidireccional, estará en D)

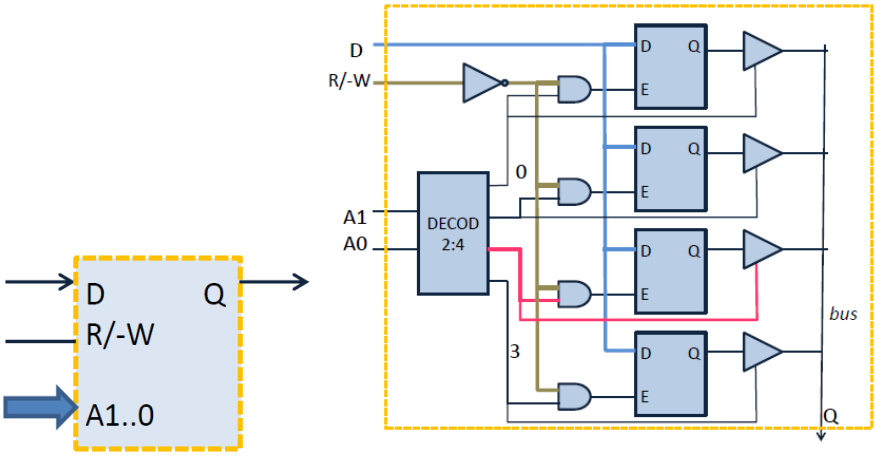
RAM con E/S separadas



Al poner un dato en D, éste se presenta en los 4 biestables. Las salidas del decodificador combinadas en las AND de entrada junto con la señal R/-W=0 habilitarán la escritura de uno de ellos, el direccionado por A1A0 .

El mismo decodificador forma – con las AND de salida y la OR de salida – un multiplexor que selecciona una de las salidas Q según los bits A1A0.

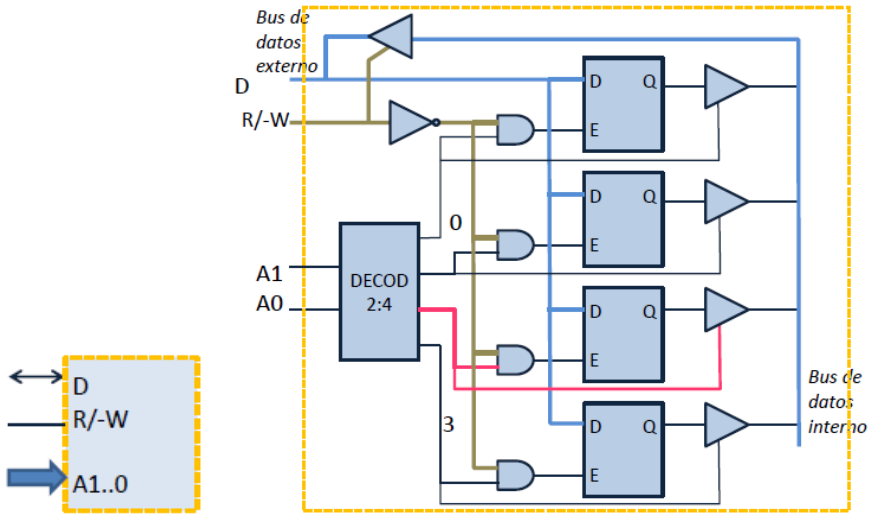
RAM con E/S separadas y buffers de Tercer Estado (Triestate)



La escritura es igual que antes.

En la salida, en vez de un multiplexor, el decodificadorhabilita uno de los buffers Triestate permitiendo que la salida Q correspondiente tome el control del bus.

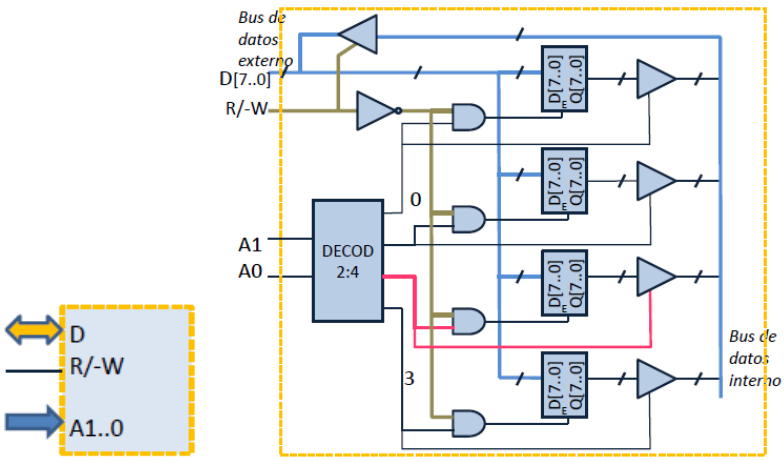
RAM con E/S bidireccional



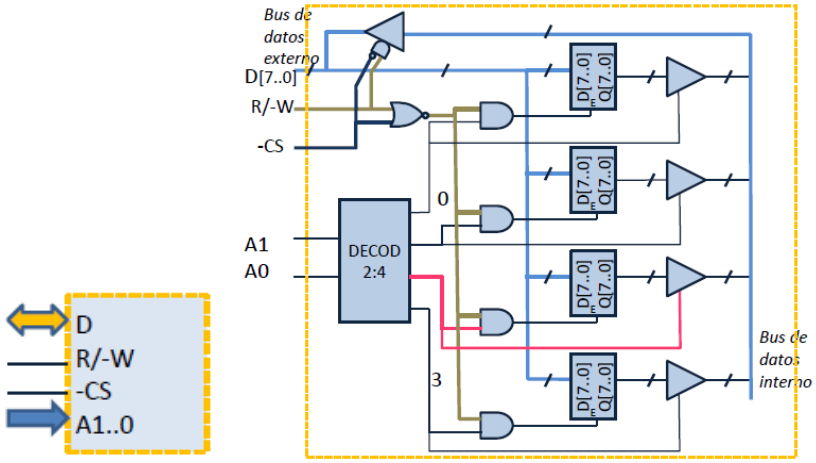
La escritura es igual que antes.

La misma entrada R/-W se utiliza para arbitrar qué señal toma el control del bus de datos externo.

Con R/-W=1, los datos salen de la RAM (lectura), con R/-W=0 (escritura) el buffer agregado impide la colisión.

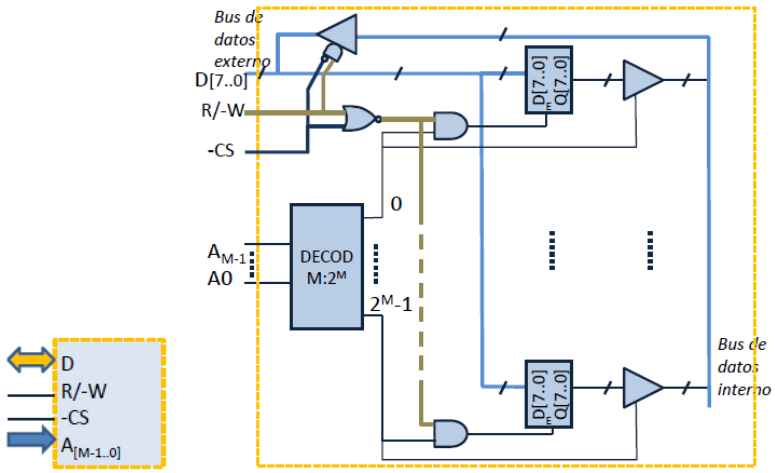


RAM estática de 4 Bytes con chip select.



El control Chip Select permite anular completamente las operaciones de lectura escritura, permitiendo aislar el chip completo. En el esquema, con -CS=1 se anulan las operaciones de lectura y escritura, con -CS=0 se habilitan.

RAM estática de 2M Bytes



Con un bus de direcciones de M bits se direccionan 2M bytes.

Memorias ROM/EPROM/EEPROM/Flash.

3.E. Tecnología: Esquemas de salida en puertas digitales.

Salidas tipo colector abierto y tipo complementaria. Análisis comparativo en velocidad, consumo y conectividad en bus. Tecnología de Tercer Estado.