Universidad Francisco de Vitoria
UFV Madrid

Fundamentos de la ingeniería informática

Ingeniería de sistemas industriales

Curso 2019-2020

Sistemas secuenciales

1. Introducción

Los sistemas combinacionales se caracterizan porque la/s salida/s en el instante t es/son una combinación de las entradas en el mismo instante t. Lo que se podría expresar de la siguiente forma:

$$z_i(t) = f_i[a_1(t), a_2(t), a_3(t) \dots a_n(t)]$$

Al contrario de ellos, los sistemas secuenciales no sólo dependen de las entradas en el instante t si no también de las entradas en instantes anteriores:

$$z_i(t) = f_i \begin{bmatrix} a_1(t), a_2(t), a_3(t) \dots a_n(t), \\ a_1(t-1), a_2(t-2), a_3(t-2) \dots a_n(t-2), \\ \dots \\ a_1(t-Q), a_2(t-Q), a_3(t) \dots a_n(t-Q) \end{bmatrix}$$

Esta expresión indica explícitamente que z_i depende de las de entradas en un numero finito (Q) de instantes anteriores.

También es posible obtener una función de salida que dependa de las entradas actuales y de las salidas en un número finito (R) de instantes anteriores:

$$z_i(t) = f_i \begin{bmatrix} a_1(t), a_2(t), a_3(t) \dots a_n(t), \\ z_1(t-1), z_2(t-2), z_3(t-2) \dots z_m(t-2), \\ \dots \\ z_1(t-R), z_2(t-R), z_3(t) \dots z_m(t-R) \end{bmatrix}$$

Lo anterior expresa (implícitamente) que la salida depende de un número infinito de entradas en instantes anteriores.

Y finalmente, pudiera ser que las salidas se puedan expresar como una combinación de las entradas actuales, las de un numero finito de instantes anteriores, y de las salidas en un numero finito de instantes anteriores.

En cualquier caso, los sistemas secuenciales, de una forma u otra necesitan de elementos de memoria que recuerden los valores pasados.

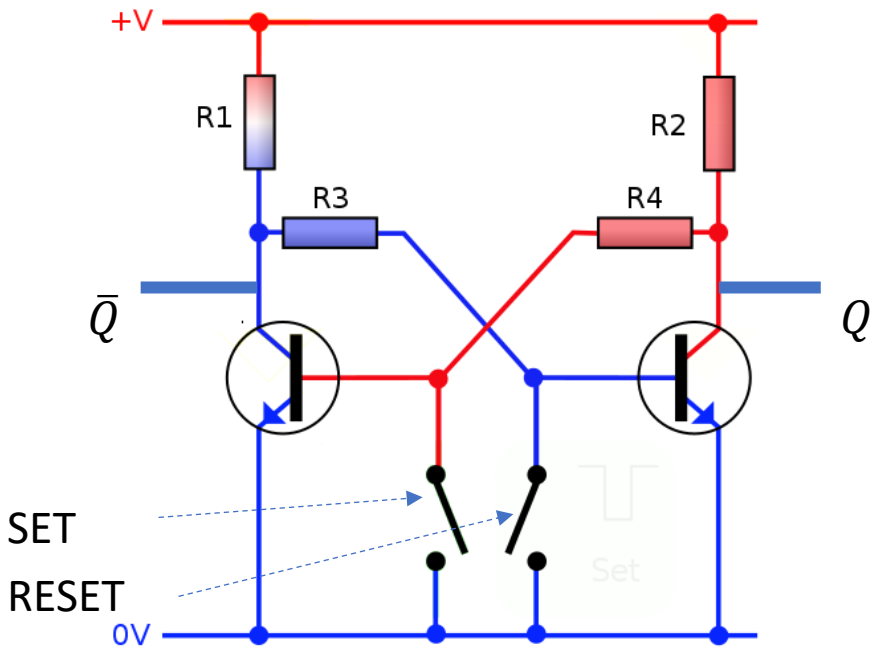
Evidentemente, aunque desde el punto de vista matemático es posible hacer depender la salida actual de entradas y/o salidas futuras, carece de sentido desde el punto de vista físico.

2. Células de memoria

Con este término queremos referirnos a circuitos que son capaces de retener un valor de un instante pasado.

Yendo a lo elemental: un biestable es una célula de memoria que puede recordar el valor de un bit en un instante anterior. El término biestable proviene del hecho de que sea cual sea el estado que toma la célula (0 ó 1) permanece inalterado indefinidamente (si la alimentación del circuito no se interrumpe) hasta que se provoca su cambio.

2.1. Biestable SR



El circuito anterior en el correspondiente a un biestable SR (set & reset) realizado con transistores NPN.

Si se analiza el circuito se descubre que cuando se activa (circuito cerrado) el pulsador SET (S), en Q aparece una tensión positiva (que interpretamos como 1), dada la simetría del circuito, al mismo tiempo en $\bar{Q}$ aparece una tensión nula (que interpretamos como 0). Dado este estado, aunque el pulsador SET se desactive (circuito abierto) el valor de Q (y /Q) no cambia.

Dada la simetría del circuito, podemos deducir que cuando el pulsador RESET (R) se activa (circuito cerrado) Q toma el valor 0 (y $\bar{Q}$ el valor 1), que se mantienen cuando se desactiva el pulsador RESET.

Esto supone que el circuito puede ser obligado a tomar un estado que se recuerda hasta que se le obliga a cambiarlo, lo que significa recordar un valor durante un tiempo virtualmente ilimitado.

Asumiendo que $\bar{Q}$ tiene siempre el valor contrario a Q podemos establecer la tabla de verdad del comportamiento de este circuito de la siguiente forma:

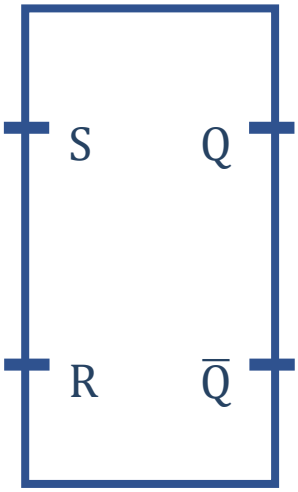
Q(t-1)	S(t)	R(t)	Q(t)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	?
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	?

Lo que se puede expresar de una forma más resumida del siguiente modo:

S(t)	R(t)	Q(t)
0	0	Q(t-1)
0	1	0
1	0	1
1	1	x

Esta información se puede leer de la siguiente forma: cuando S y R toman el valor 0 simultáneamente la salida no cambia, cuando sólo R se activa la salida toma el valor 0 independientemente de su estado anterior, cuando sólo S se activa la salida toma el valor 1 independientemente de su estado anterior. Finalmente, cuando S=1 y R=1 el valor de la salida no está predefinido.

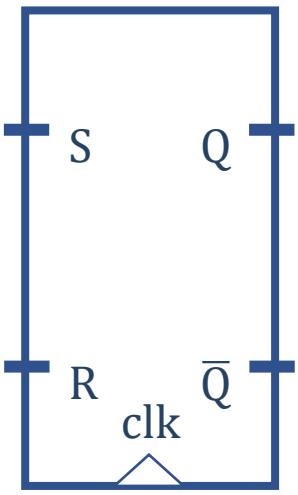
La representación gráfica habitual del Biestable SR es



En este circuito queda un poco confusa la definición de instante de tiempo, esto es debido a que los cambios en la salida ocurren cuando ocurren cambios en las entradas, no quedando claro el tiempo entre ellos son o no instantes de tiempo distintos.

Este tipo de circuito se denomina asíncrono.

Existe otro tipo de Biestable RS que resuelve esta indefinición y que por supuesto se denomina biestable RS síncrono. Su grafismo es el siguiente:

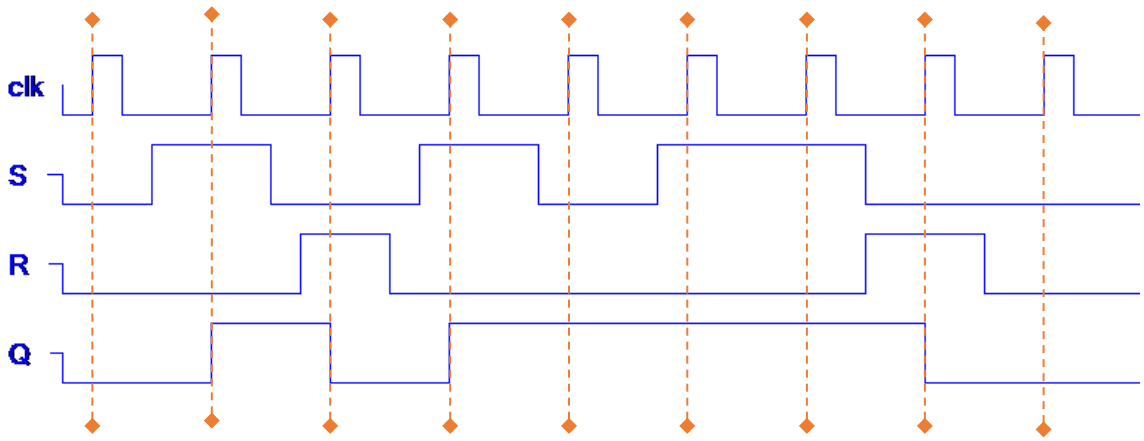


El circuito síncrono tiene una entrada más denominada clk (del inglés clock, reloj). Esta señal permite sincronizar el comportamiento de este dispositivo con el de otros del mismo hardware, pues los cambios sólo ocurren (dependiendo del estado de S y R) solo en el momento concreto en que clk cambia de 0 a 1. De este modo los cambios en los distintos dispositivos ocurren a la vez (síncronamente).

El término reloj se aplica a señales que de forma periódica cambian de 0 a 1 y viceversa. Sin embargo, los biestables no necesitan que eso sea así, por ello en múltiples casos, esa entrada recibe otros nombres como **enable**, **strobe, etc.**, algunos de los cuales iremos utilizando a lo largo de estos apuntes según la funcionalidad que dicha señal realice en el circuito.

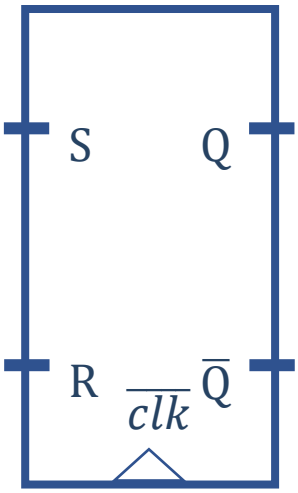
La transición de 0 a 1, se denomina flanco de subida y que el dispositivo sólo funcione en los flancos de subida se explicita con el triángulo que marca la entrada clk.

Aquí podemos presentar dos términos referidos a los tipos de señal, **de impulso** aquellas que funcionalmente son útiles en sus transiciones, como el clk): **señales de nivel** aquellas que funcionalmente son útiles por el estado estable en el que se encuentren. Las señales de impulso son utilizadas para transmitir ordenes mientras que las de nivel lo son para transmitir información.

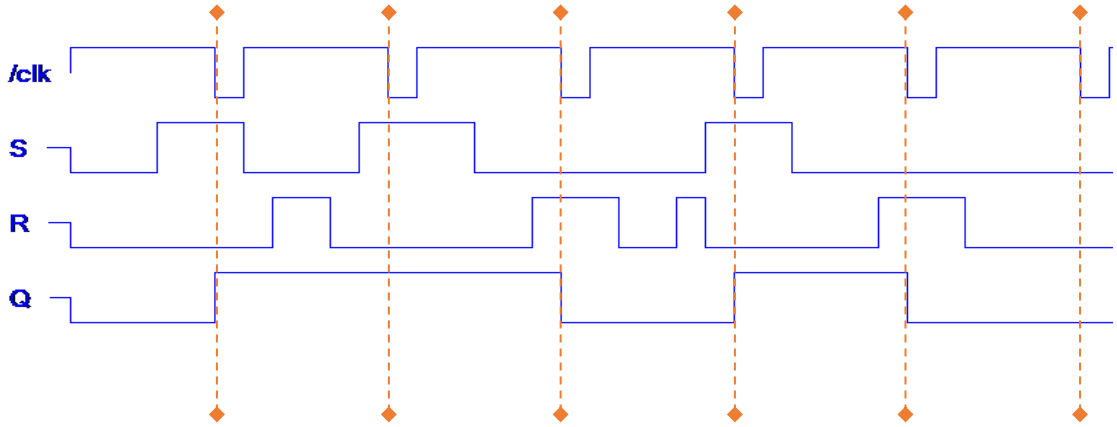


En la gráfica anterior el eje horizontal representa el tiempo. Cada cierto tiempo, de forma periódica aparece un flanco de subida en clk y es entonces cuando pueden ocurrir cambios siguiendo el comportamiento definido para un biestable SR.

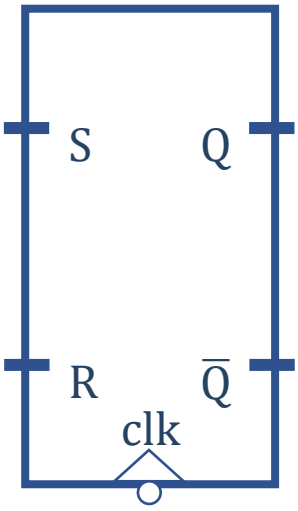
Existe otra posibilidad de funcionamiento del biestable SR



Se trata de que los cambios sólo puedan ocurrir en el flanco de bajada (transición 1 a 0) de la señal de reloj, cosa que se expresa negando la señal clk.



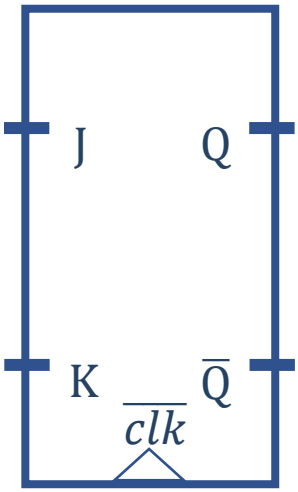
Esto mismo también se puede expresar con el grafismo que sustituye el negado en el nombre de la entrada por un pequeño circulo en la misma



2.2. Biestable JK

El comportamiento del biestable JK es muy similar al del SR, con la excepción de que resuelve la indefinición que tiene la salida cuando las dos entradas toman el valor 1. En este caso el biestable JK cambia al estado opuesto al actual.

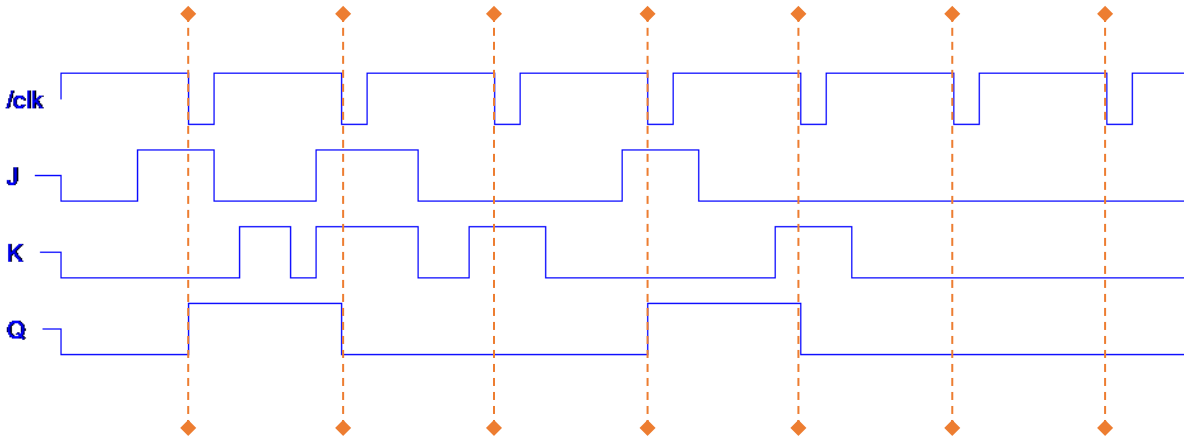
La simbología gráfica de este biestable es la siguiente:



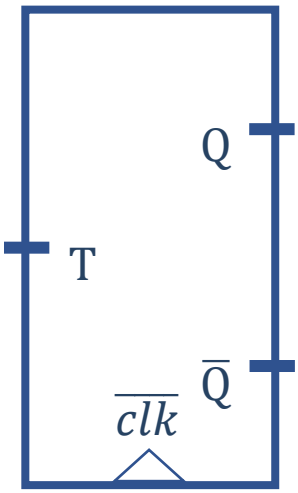
J(t)	K(t)	Q(t)
0	0	$Q(t-1)$
0	1	0
1	0	1
1	1	$\overline{Q(t-1)}$

En este caso se ha decidido utilizar el valor negado del reloj, lo que significa que los cambios ocurren sólo en el flanco de bajada (1→0) de la señal.

Así un ejemplo de diagrama de tiempo sería:



2.3. Biestable T

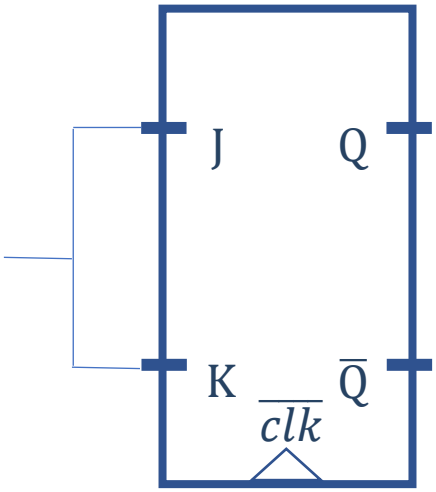


El comportamiento de un biestable T (transition) produce el cambio del estado siempre que la entrada toma el valor 1 en el flanco activo del reloj.

$\mathbf{T}(t)$	$\mathbf{Q}(t)$
0	$\mathbf{Q}(t-1)$
1	$\overline{\mathbf{Q}(t-1)}$

$Q(t-1)$	T	$Q(t)$
0	0	0
0	1	1
1	0	1
1	1	0

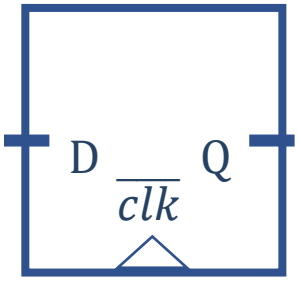
Una forma de construir este biestable es la siguiente:



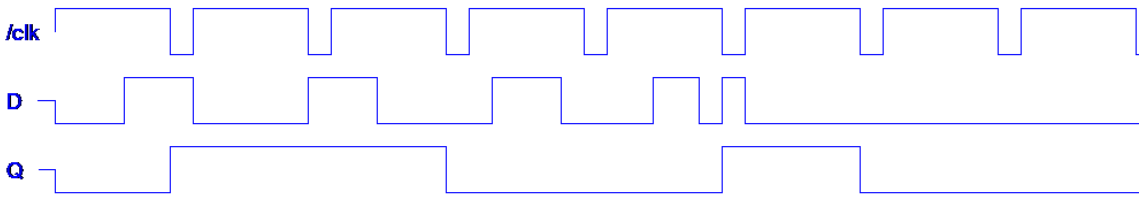
Y un ejemplo de diagrama de tiempo del comportamiento es:

2.4. Biestable D

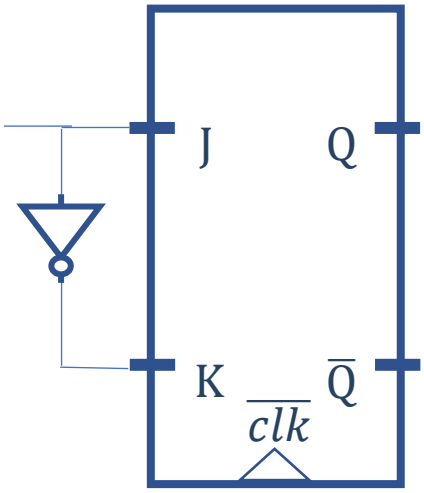
El biestable D (Data) copia el valor de la entrada a la salida en el flanco activo del reloj.



$\mathbf{D}(t)$	$\mathbf{Q}(t)$
0	0
1	1



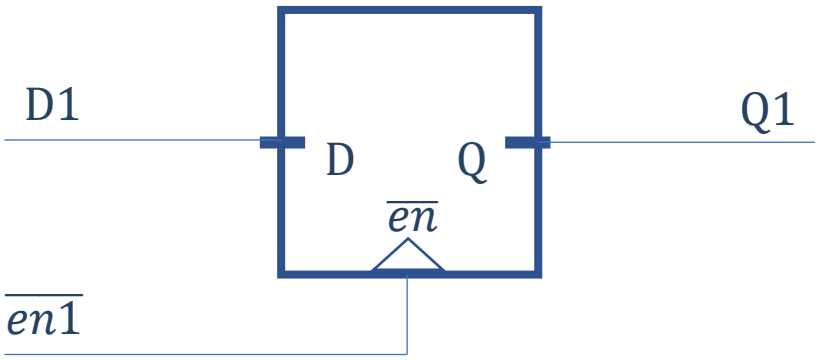
Una forma de construir este biestable es la siguiente:



3. Memoria

3.1. Memoria de 1 bit

Una versión básica de una memoria de un bit puede realizarse utilizando directamente un biestable D en un circuito como el de la figura:



En este caso se ha optado por etiquetar como “en” de (enable) a la señal de sincronismo, al suponer que esta no es periódica

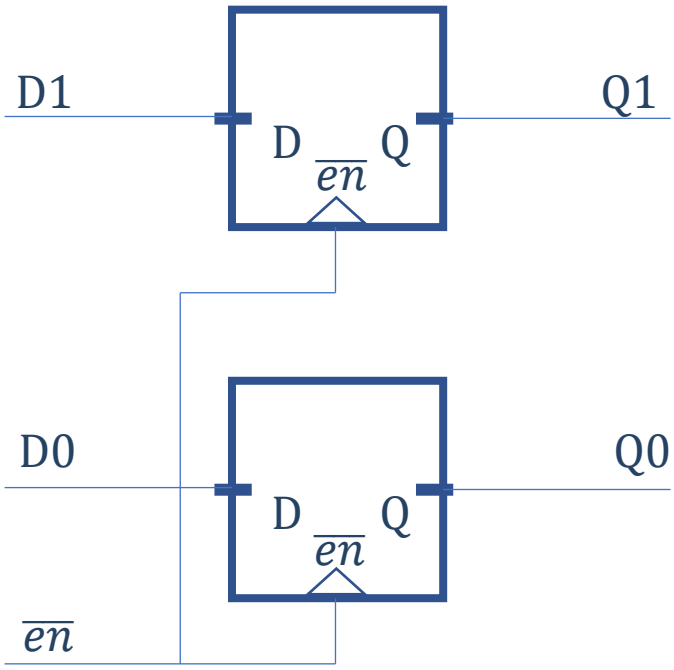
La operación de escribir en memoria se reduce a que en la entrada D1 se encuentre el valor deseado y cuando sea así activar el enable. En ese momento Q1 tomará el valor de la entrada y permanecerá estable con independencia de los cambios en aquella entrada.

La operación de lectura consiste en recoger el estado de la salida.

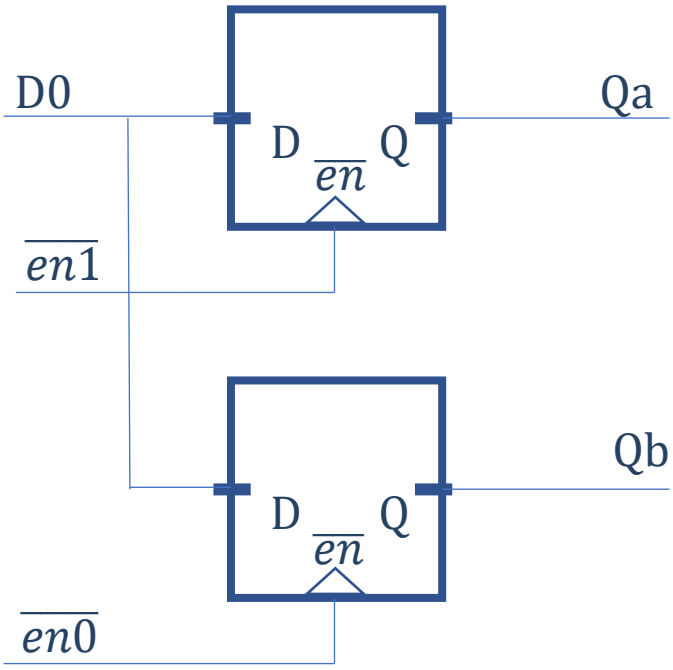
3.2. Memoria de 1 célula de 2 bits

Escalando el circuito anterior podemos tener dos bits en paralelo. El objetivo es que la escritura ser realice simultáneamente en ambos.

Repitiendo lo dicho antes, la operación de lectura se reduce a tener los valores deseados en D0 y D1 y cuando estos sean estables activar la señal **enable**.

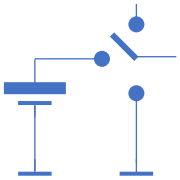


3.3. Memoria de dos células de 1 bit



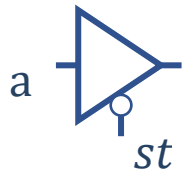
El problema en el circuito anterior es que no se puede cortocircuitar Q1 con Q0 al tratarse de dos salidas.

La solución es utilizar una puerta buffer con salida tristate. Este tipo de salida puede tomar no dos sino tres estados, el 0 y el 1 habituales y un tercer estado en el que la puerta ofrece un estado de alta impedancia. Algo parecido al circuito eléctrico de la figura que puede estar conectado a la batería (1), a masa (0) o desconectado (Hi-Z).

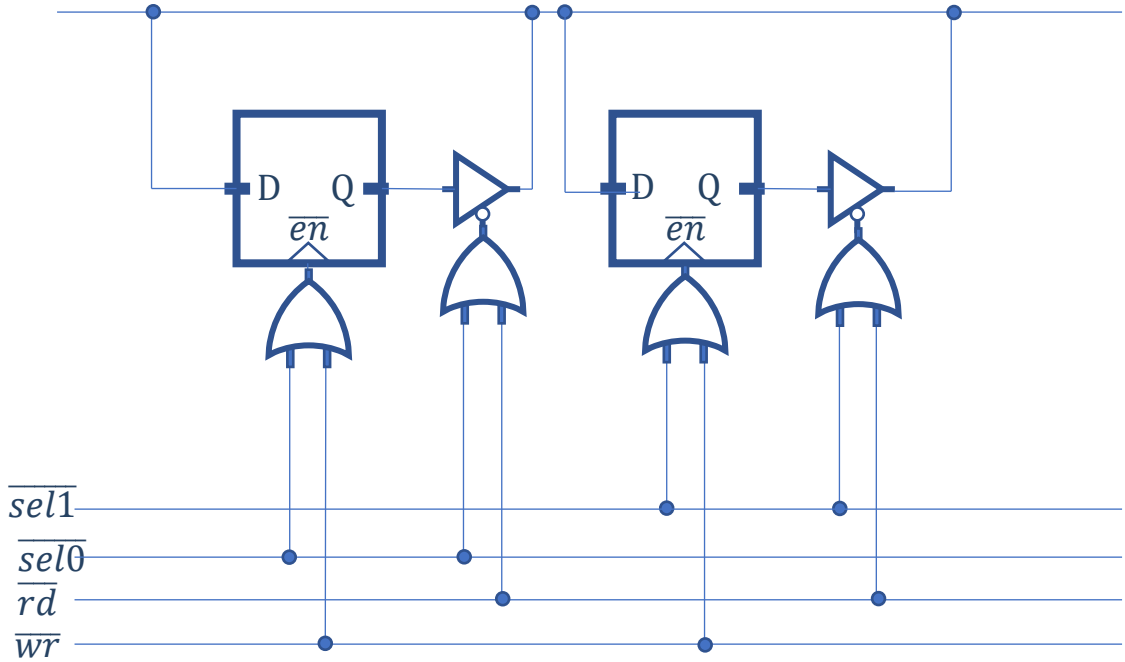


Si se asegura que como máximo una no está en estado Hi-Z, varias celdas de este tipo pueden estar conectadas por sus salidas.

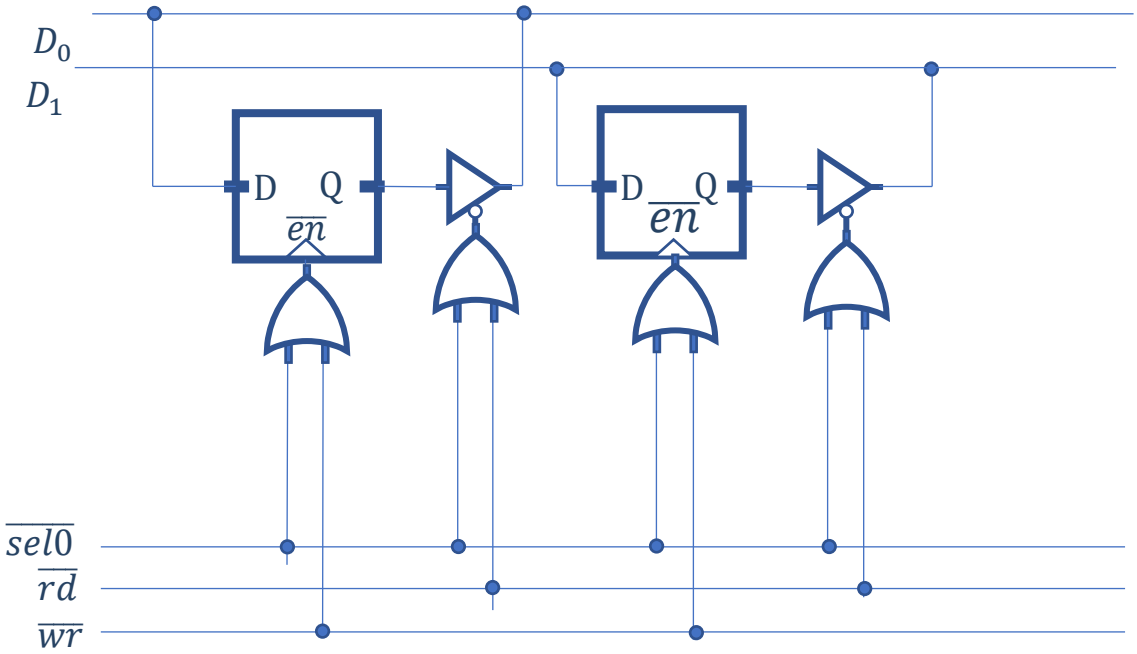
La señal que controla ese tercer estado se denomina strobe (st)



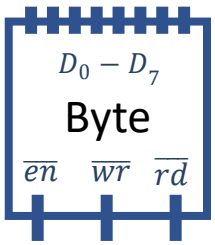
$\overline{st}$	a	z
0	0	Hi-Z
0	1	Hi-Z
1	0	0
1	1	1



3.4. Memoria 2 bits

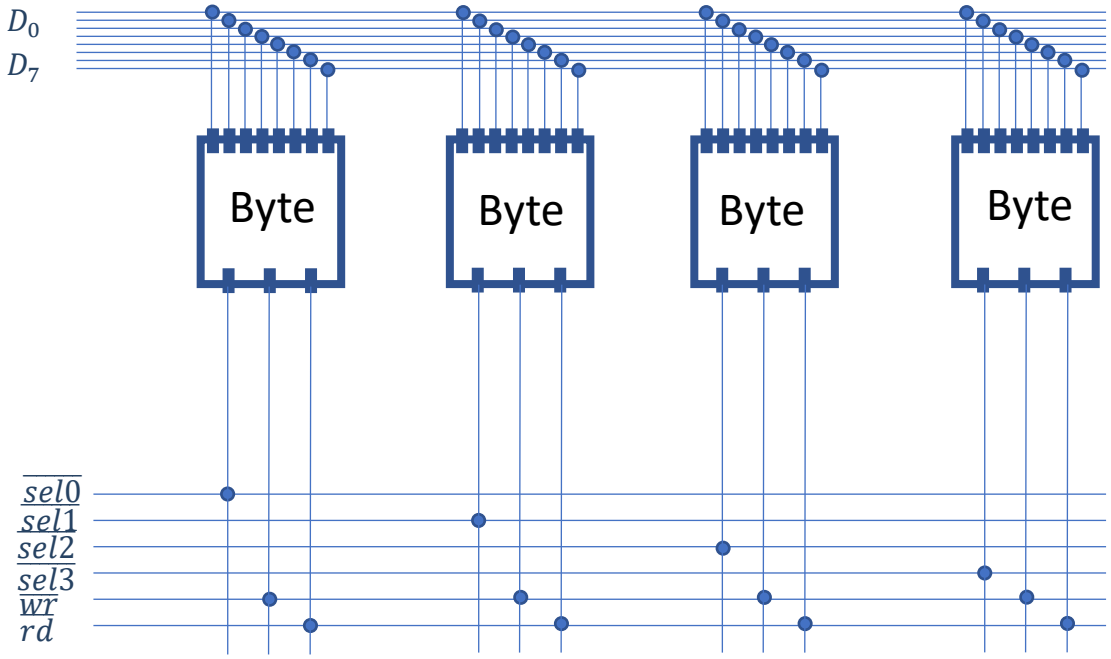


Podemos extender este esquema hasta tener una memoria de una palabra de 8 bits, y el resultado encapsularlo en una caja negra que vamos a llamar Byte. Tendrá 8 entrada/salidas de datos D_0-D_7, una señal de selección otra de lectura y otra de escritura comunes para todos los bits de la palabra.



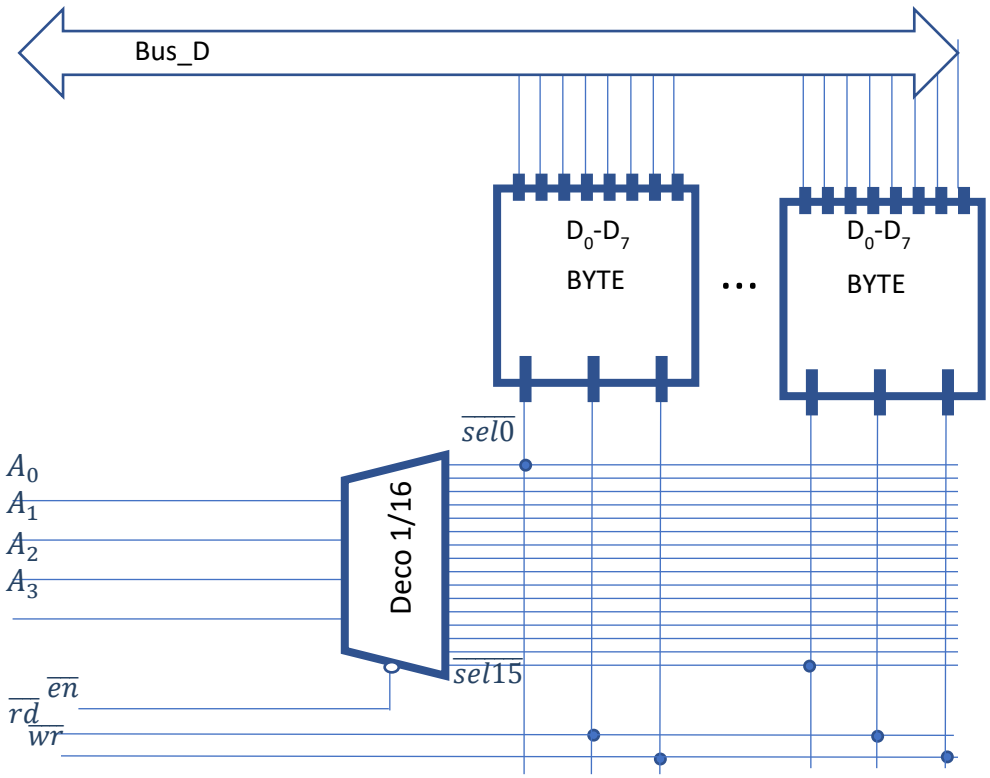
3.5. Memoria

Podemos reunir los conceptos descritos en las dos secciones anteriores para obtener memorias de múltiples palabras de un número bits también múltiple.



3.6. Escalado de memoria

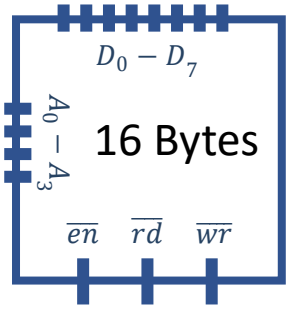
Podemos encadenar 16 de esas cajas negras que hemos llamado **bytes**, necesitaríamos 16 señales que activen los 16 **enable**. Dado que se utilizan para seleccionar cada uno de los bytes independientemente las hemos llamado **select** (sel0 … sel15). Debemos asegurar que como máximo uno de los sel está activo. Para ello podemos incorporar un circuito **decodificador 1:16**. Como habíamos visto, un decodificador es un circuito de cuyas salidas sólo hay una activa (ninguna si su enable está desactivado). La salida activa es aquella cuyo número de orden coincide con el código indicado en las entradas, en este caso 4 (A_0-A_3).



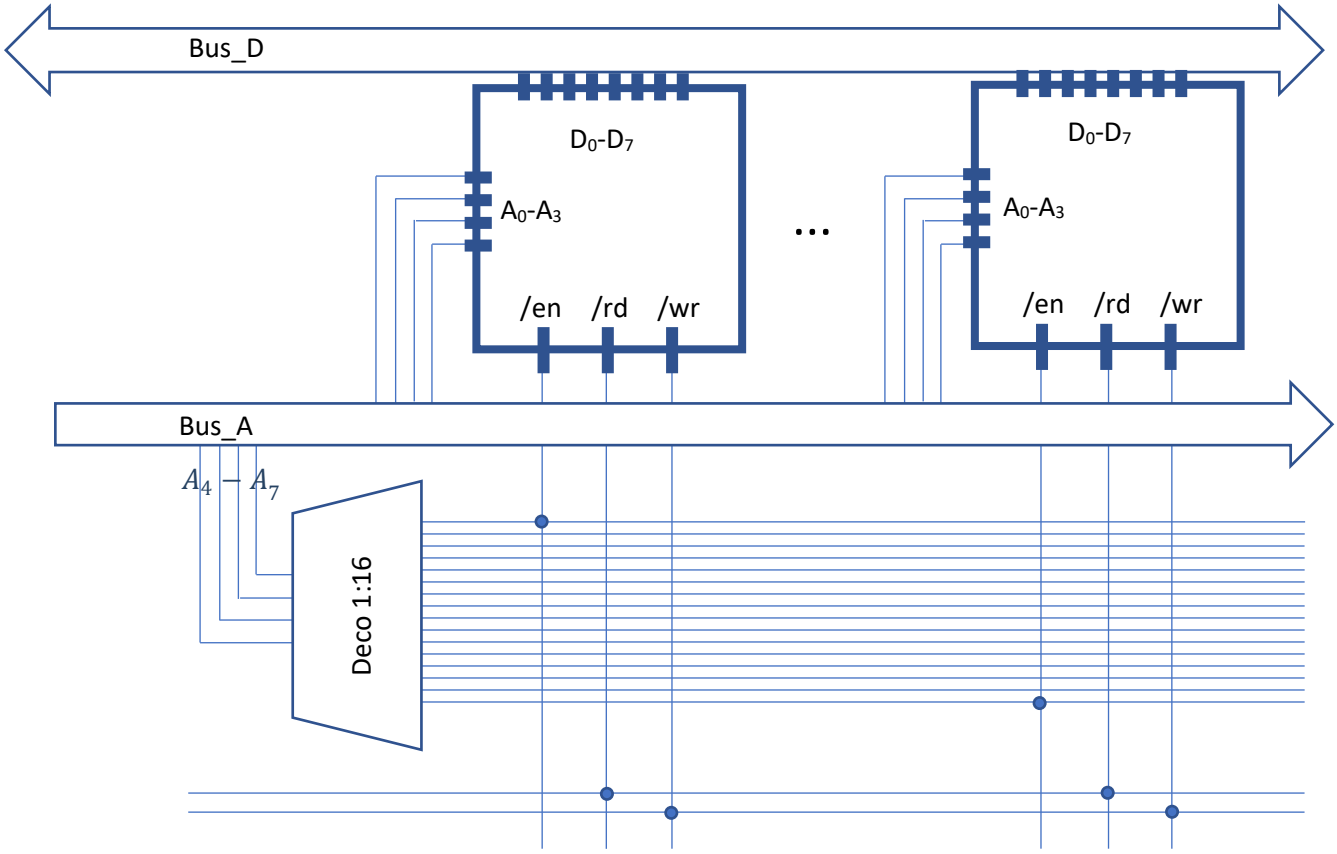
Al conjunto de señales D_0-D_7 se le denomina **bus de datos** (BUS-D).

A las señales A_0, A_1, A_2, A_3 las llamamos señales de dirección pues codifican la posición en la que se encuentra el byte seleccionado.

Podemos repetir el proceso y encapsular el circuito anterior en una caja negra:



Y encadenar 16 de ellas para alcanzar una memoria de 256B



Utilizamos otras cuatro señales de dirección (A_4-A_7) para determinar la posición lógica del bloque de 16 entre sus homólogos.

Al conjunto de todas las señales de dirección se le denomina **bus de direcciones** (address bus, Bus-A)

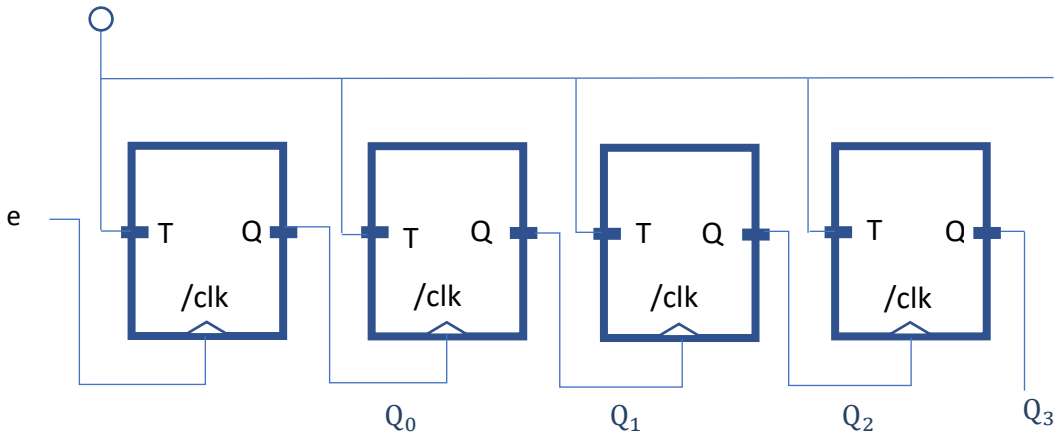
Esta forma de escalar la memoria la podemos extender hasta alcanzar el volumen de memoria deseado.

4. Circuitos secuenciales

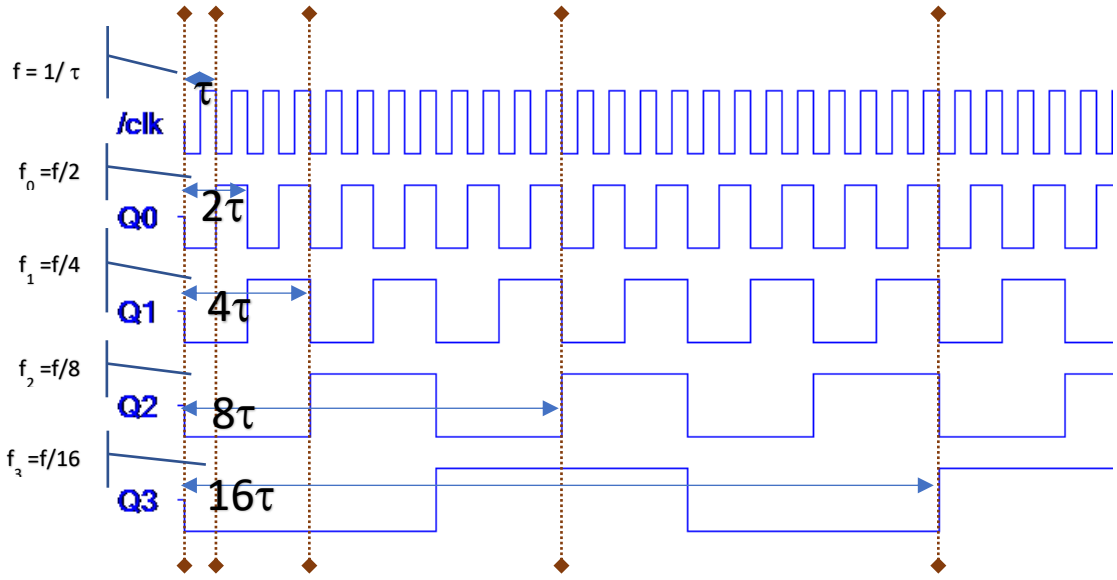
4.1. Divisor de frecuencia

Dada una señal de reloj de frecuencia f (periodo T) un divisor de frecuencia es un circuito secuencial que a la salida ofrece un reloj de periodo n·T (frecuencia f/n).

Una forma de construir un divisor de frecuencia con $n = 2^k$ es encadenar biestables T del siguiente modo



Si e es una función **cuadrada de reloj** (igual tiempo en 0 que en 1) como en la figura en la salida obtenemos Q_0 una señal cuadrada de mitad de frecuencia, en Q_1 otra de ¼ de la frecuencia inicial, en Q_2 una de 1/8 y en Q_3 una de 1/16.

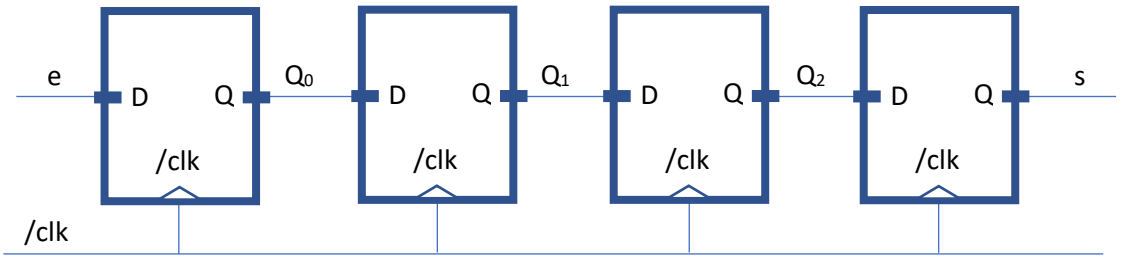


4.2. Registro de desplazamiento

Un registro de desplazamiento es un circuito que tiene una entrada (e) y una salida (s). En cualquier momento s(t) = e(t-N).

Esto implica que al menos es necesario recordar los últimos N valores de la entrada. Vamos a construir este circuito utilizando biestables D.

Si N es igual a 4 necesitaremos cuatro biestables.



Así, siendo el reloj quien marca los instantes t:

$$Q_0(t) = e(t-1)$$

$$Q_1(t) = Q_0(t-1) = e(t-2)$$

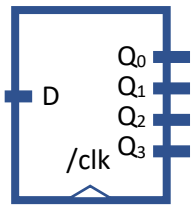
$$Q_2(t) = Q_1(t-1) = e(t-3)$$

$$s(t) = Q_2(t-1) = e(t-4)$$

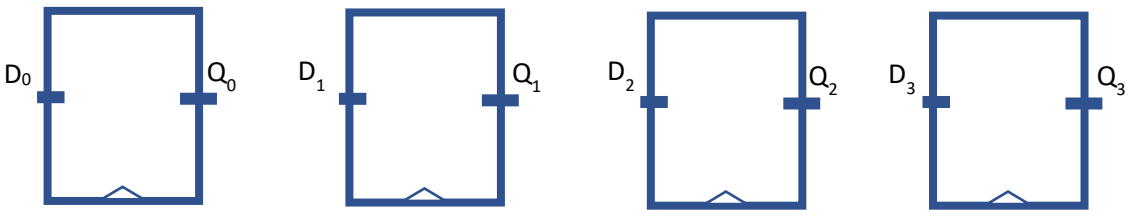
4.3. Contador

Un contador es un circuito que en sus salidas registra en binario el numero de pulsos (flancos) que se han producido a la entrada.

Para un contador de 16 estados serán necesarias cuatro salidas binarias.



Para tener cuatro salidas Q necesitaremos cuatro biestables D.



Tenemos que averiguar como conectar esos biestables. Planteemos la tabla de verdad. Incluimos la variable temporal t solo a modo de referencia

t	Q_3	Q_2	Q_1	Q_0		D_3	D_2	D_1	D_0
0	0	0	0	0		0	0	0	1
1	0	0	0	1		0	0	1	0
2	0	0	1	0		0	0	1	1
3	0	0	1	1		0	1	0	0
4	0	1	0	0		0	1	0	1
5	0	1	0	1		0	1	1	0
6	0	1	1	0		0	1	1	1
7	0	1	1	1		1	0	0	0
8	1	0	0	0		1	0	0	1
9	1	0	0	1		1	0	1	0
10	1	0	1	0		1	0	1	1
11	1	0	1	1		1	1	0	0
12	1	1	0	0		1	1	0	1
13	1	1	0	1		1	1	1	0
14	1	1	1	0		1	1	1	1
15	1	1	1	1		0	0	0	0

Es evidente que:

$$\boldsymbol{D_0 = \overline{Q_0}}$$

Para las demás salidas planteamos Karnough

Q_3Q_2 \ Q_1Q_0	00	01	11	10
D_1				
00		1		1
01		1		1
11		1		1
10		1		1

$D_1 = \overline{Q_1}Q_0 + Q_1\overline{Q_0}$

$$\boldsymbol{D_1 = Q_1 \oplus Q_0}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
D_2				
00			1	
01	1	1		1
11	1	1		1
10			1	

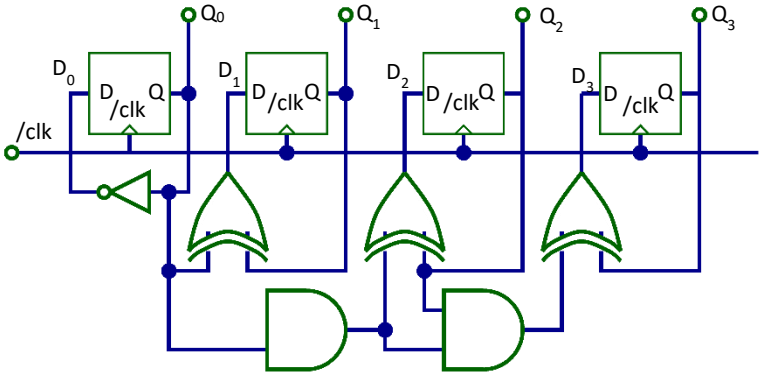
$D_2 = \overline{Q_2}Q_1Q_0 + Q_2\overline{Q_0} + Q_2\overline{Q_1}$

$$\boldsymbol{D_2 = Q_2 \oplus (Q_1 Q_0)}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00				
01			1	
11	1	1		1
10	1	1	1	1

$D_3 = \overline{Q_3}Q_2Q_1Q_0 + Q_3\overline{Q_2} + Q_3\overline{Q_1} + Q_3\overline{Q_0}$

$\boldsymbol{D_3 = Q_3 \oplus (Q_2 Q_1 Q_0)}$

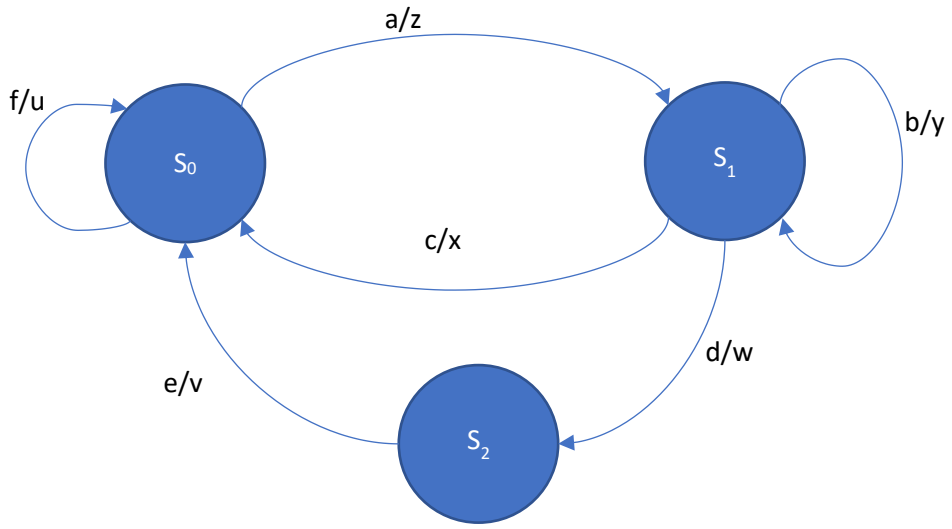


5. Máquina de estados

Una máquina de estados es una máquina secuencial cuyo comportamiento se define mediante un diagrama de estados.

Un diagrama de estados representa un conjunto de estados enlazados por transiciones. Una transición representa un cambio de estado asociado en nuestro caso a una entrada y una salida.

En el diagrama se presenta un ejemplo de diagrama de dos estados :



La lectura de este diagrama es la siguiente:

- Estando en el estado s_0
  - si se produce la entrada a se cambia al estado s_1 y se emite la salida z
  - si se produce la entrada f se cambia al propio s_0 y se emite u
- Estando en s_1
  - Con entrada d, se cambia a s_2 y se emite w
  - Con entrada b, se cambia a s_1 y se emite y
  - Con entrada c, se cambia a s_0 y se emite x
- Estando en s_2
  - Con entrada e, se cambia a s_0 y se emite v
- En cualquier otra situación ni es cambia de estado ni se emite nada.

5.1. Definición

La definición de una máquina de estados se realiza mediante la definición de las entradas, los estados, las salidas y finalmente las transiciones.

5.1.1. Entradas

Se trata de un conjunto de símbolos E que deben ser binarias o codificables en binario.

Siendo n el número de bits necesario para codificar las entradas I= (i_0, i_1, … i_{n-1})

5.1.2. Salidas

Se trata de un conjunto de símbolos O que deben ser binarias o codificables en binario.

Siendo m el número de bits necesario para codificar las entradas O = (o_0, o_1, … o_{m-1})

5.1.3. Estados

Un estado es cada una de las situaciones internas de la máquina. Se trata de un concepto abstracto y no tiene por qué corresponder a nada real.

Si P es el número de estados, p es el número de bits con el que se representa cada estado, tal que $P < 2^p$. De este modo cada estado $S = (s_0, s_1, \ldots, s_{p-1})$

5.1.4. Transiciones

Cada transición se trata de un cambio de estado de la máquina (aunque puede cambiar al mismo estado en el que se encontraba.

La transición se produce en respuesta a la recepción de un símbolo de entrada (I).

La transición suele implicar la emisión de un símbolo de salida.

Las transiciones de la máquina se definen en una tabla en el que a las parejas (I, S) corresponden parejas (O,S'). Siendo I el símbolo de entrada S el estado antes de la transición, O símbolo de salida y S' el estado después de la transición.

5.2. Construcción de la máquina

Definida una máquina de estados, el proceso de construir la máquina comienza por la elección de la codificación binaria de las entradas, las salidas y los estados.

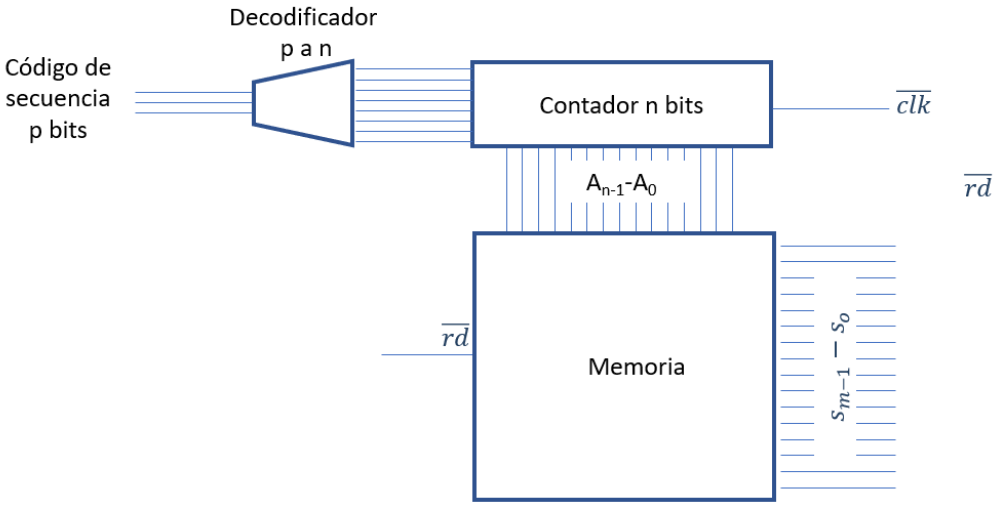
Transcribir la tabla de transcripciones en tabla de verdad.

Obtener las expresiones, simplificar e implementar con biestables utilizando un biestable D por cada bit de la variable de estado S.

6. Secuenciador

Un secuenciador es una máquina que ofrece una secuencia de salidas en respuesta a una entrada. Puede desarrollarse como máquina de estados, pero es más común implementarla en base a una memoria.

La memoria contiene todas las secuencias posibles, que comienzan en posiciones conocidas de la misma. Responder a una entrada implica emitir sucesivamente el contenido de la primera posición de la secuencia y las siguientes. Pueden establecerse distintos tipos de finalización de la secuencia: porque la longitud sea fija o porque una de las salidas emitidas sea la que provoque la finalización.


Decodificador
p a n
Código de
secuencia
p bits
Contador n bits
$\overline{clk}$
$\overline{rd}$
A_{n-1}-A_0
$\overline{rd}$
Memoria
$s_{m-1} - s_o$