TEMA III: ELEMENTOS DE MEMORIA

Como ya hemos comentado, la diferencia básica entre la lógica combinacional y secuencial radica en la propiedad de almacenamiento de esta última. Esta propiedad puede ser alcanzada de dos formas diferentes:

- De forma implícita, a través de lazos de realimentación directa (con o sin elementos de retraso)
- De forma explícita, a través de elementos de memoria.

Por lo tanto, en este tema nos centraremos en las principales características y tipos de estos elementos, que nos podemos encontrar en los sistemas secuenciales.

También presentaremos los grandes sistemas de almacenamiento, y sus características, que nos podemos encontrar en sistemas complejos como sistemas informáticos.

1. Introducción. Definiciones y Clasificaciones.

Entre las muchas definiciones que podemos encontrar de elemento de memoria, vamos a elegir la siguiente:

> Un **elemento de memoria** es aquel elemento capaz de almacenar un estado durante un tiempo determinado.

Los dos elementos, mostrados en la figura 3.1, son elementos de memoria.

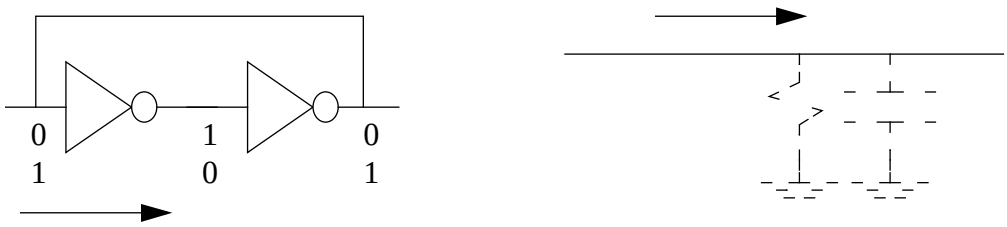

Figura 3.1.- Ejemplos de elementos de memoria

El primer elemento está formado por dos inversores realimentados de tal forma que el valor a la entrada del primer inversor es el mismo que a la salida del segundo inversor, estando

de acuerdo con el lazo de realimentación. Por lo tanto, mientras que el dato de entrada no cambie, el dato de salida permanecerá sin cambiar, es decir, quedará almacenado. De igual forma podemos comprobar que una simple línea de conexión muestra el mismo comportamiento, de tal forma que la tensión es almacenada en el condensador parásito asociado a dicha línea.

La diferencia entre ambos elementos se encuentra en el tiempo que permanece almacenado el dato, característica que se suele denominar **duración de la información**. En el primer elemento, la información permanecerá almacenada indefinidamente hasta que el dato de entrada cambie su valor. En cambio, en el segundo caso, de la misma forma que hay un condensador parásito, también existe una resistencia parásita, creando un camino de descarga a través de la resistencia. Como el dato no es regenerado por ningún elemento (como sucede con los inversores en el primer elemento), cuando se sobrepasa un determinado tiempo, que se denomina tiempo de descarga y suele considerarse proporcional al producto RC, la tensión almacenada no es lo suficiente alta como para identificar un nivel lógico o cambia su valor. A este tipo de almacenamiento se denomina **almacenamiento dinámico**; mientras que cuando el dato permanece durante un tiempo indefinido, el almacenamiento se denomina **almacenamiento estático**. En el caso del almacenamiento dinámico, para evitar la pérdida de la información es necesario volver a almacenar la información de forma periódica (antes de superar el tiempo de descarga), lo cual se conoce como **ciclo de refresco**.

Otra propiedad que podemos encontrar en los ejemplos anteriores consiste en un almacenamiento instantáneo. Cuando el dato de entrada cambia, el valor almacenado en el elemento de memoria cambia de forma instantánea (después de que se haya superado el retraso impuesto por el elemento), como podemos ver en la figura 3.2. A esta propiedad se la conoce con el nombre de **transparencia**, diciéndose entonces que estamos considerando un **elemento de memoria transparente**.

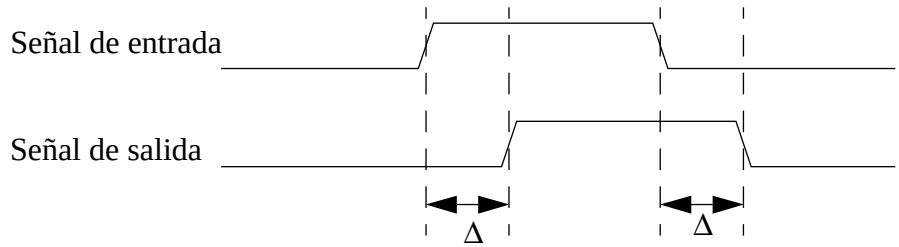

Figura 3.2.- Formas de onda correspondiente a un elemento de memoria transparente

En contraposición a este tipo de elementos podemos encontrar elementos de memoria no transparentes. En este tipo de elementos, los cambios correspondientes a los datos almacenados no obedecen directamente a los cambios de los datos de entrada, sino que solamente se producirán cuando lo indique un señal de control. Así, los elementos mostrados en la figura 3.3(a) pertenecen a este grupo.

Como podemos ver en la figura 3.3(b), la señal de control C controla un conmutador que permite o no el paso del dato de entrada al resto del elemento. Así, mientras C tenga un valor bajo, evitando el paso del dato de entrada, el elemento mantiene almacenado el dato anterior (en el caso de las formas de onda se ha supuesto un nivel bajo). En esta fase de operación se dice que el elemento es opaco o está en su **fase opaca**, de tal forma que se pierde toda influencia respecto a los datos de entrada. En cambio, cuando la señal C permite el paso del dato de

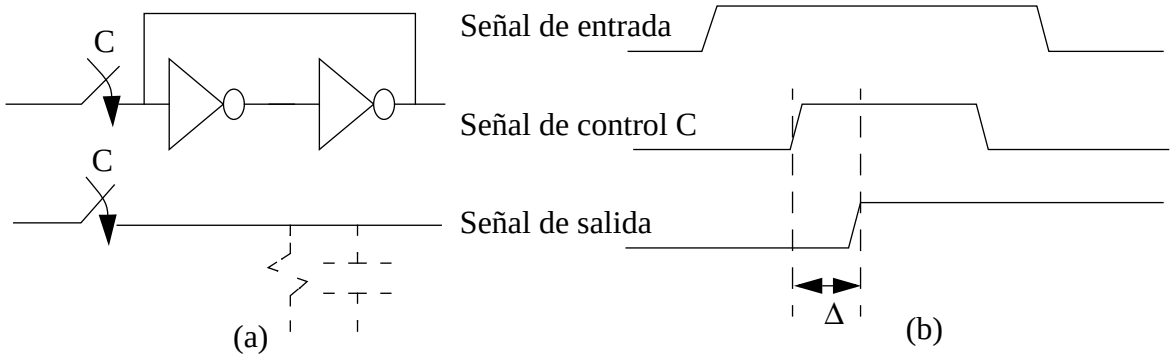

Figura 3.3.- (a) Elementos de memoria controlados o no transparentes y (b) su formas de onda.

entrada, el elemento obedece a este último. En esta fase de operación se dice que el elemento es transparente o está en su **fase transparente**.

En función de esta señal de control o de criterios de transparencia, podemos clasificar a estos elementos de memoria en:

- Elementos sensibles al nivel o **latches**.- Son los elementos en los que la fase de transparencia se corresponde con el intervalo en el que la señal de control tiene su nivel activo.
- Elementos de memoria sensibles a la transición o **flip-flops**.- Son los elementos en los que la fase de transparencia se corresponde con una transición de la señal de control, por lo que también se dice que carece de esta fase (por ser un instante y no un intervalo) y por tanto de esta propiedad.

Este comportamiento se puede apreciar en la figura 3.4. De ambos tipos de elementos, los flip-flops son los que muestran una mayor independencia con respecto a los datos de entrada, o lo que es lo mismo, una menor ventana de transparencia.

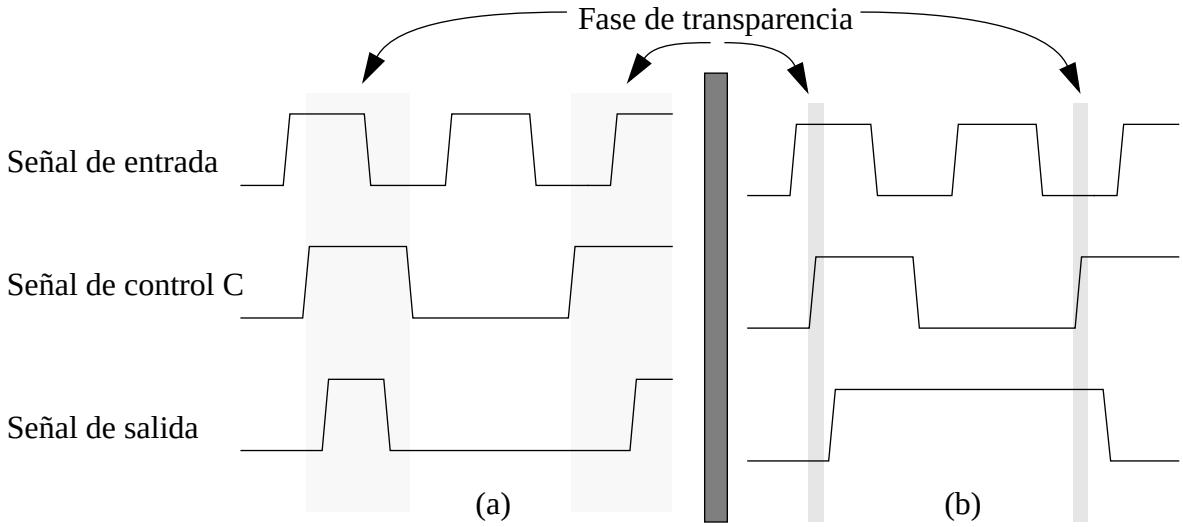

Figura 3.4.- Formas de onda correspondientes a (a) los latches y a (b) los flip-flops.

2. Elementos de Memoria.

No podemos olvidar que estamos estudiando una parte de la Electrónica Digital, y por lo tanto, nuestros datos podrán tener dos valores: ‘1’ ó ‘0’ lógico. Entonces, los elementos de memoria deben poder almacenar dos estados, correspondientes a los dos valores lógicos. Este es el motivo por el cual a los elementos de memoria que utilizamos en los sistemas secuenciales también se les conozca como **biestables**. A partir de ahora utilizaremos los conceptos de elemento de memoria o biestable de forma totalmente equivalente.

Una vez realizada esta puntualización vamos a estudiar los principales tipos de biestables que podemos encontrar. Para ello, se han dividido, utilizando los criterios de transparencia, en biestables transparentes, latches y flip-flops.

2.1. Elementos de Memoria Transparentes.

En primer lugar consideraremos los elementos de memoria transparentes, es decir, los biestables que carecen de señal de control.

Estudiemos el circuito mostrado en la figura 3.5.

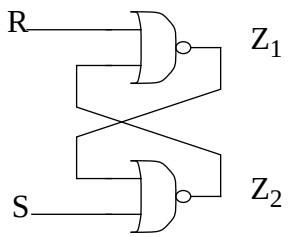

Figura 3.5.- Circuito bajo estudio.

Como las salidas también se realimentan como entradas, las ecuaciones lógicas correspondientes a dicho circuito son:

$$Z_1 = (R + Z_2)'$$

$$Z_2 = (S + Z_1)'$$

Por lo tanto, la tabla de verdad (que en este caso se denominará tabla de transición) será la mostrada en la tabla 3.1.

$Z_1\ Z_2$	R S: 00	01	11	10
00	11	10	00*	01
01	01*	00	00	01*
11	00	00	00	00
10	10*	10*	00	00

$Z_1\ Z_2$

Tabla 3.1. Tabla de estados del circuito bajo estudio

Podemos comprobar en dicha tabla que existen tres estados estables (marcados con un asterisco), los cuales son “00”, “01” y “10”, ya que para alguna combinación de entrada el próximo estado coincide con el presente.

Consideremos el estado estable “00”, por lo que partiremos de la combinación de entrada RS=”11”. A continuación vamos a ver el comportamiento de dicho circuito para cualquiera de las posibles transiciones de las señales de entrada RS, los cuales se muestran en la figura 3.6.

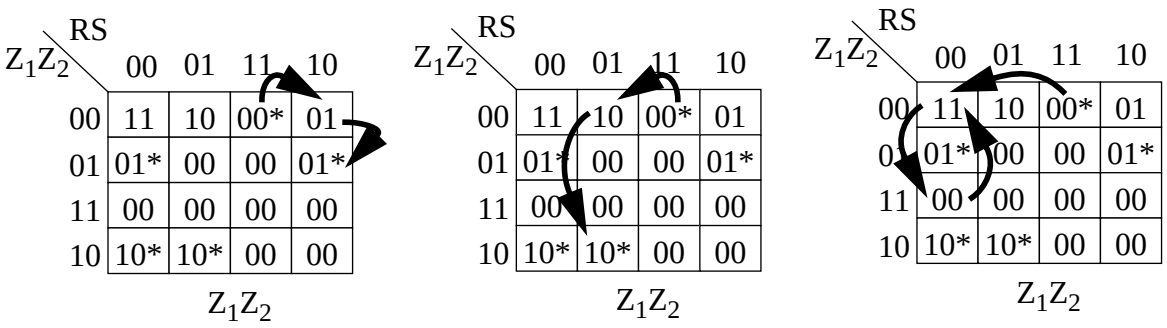

Figura 3.6.- Comportamiento del circuito bajo estudio, para todas las transiciones partiendo desde la combinación de entrada “11”.

Se puede observar que en el caso de las dos primeras transiciones se llega a un estado estable, “01” y “10” respectivamente. No obstante, en la última transición, es decir, en el cambio simultáneo de las señales de entrada, nunca llegamos a un estado estable sino que el circuito entra en un comportamiento cíclico, del cual solamente se saldrá cuando se produzca otra transición en las señales de entrada. Esta situación es problemática ya que no podemos saber cual será el estado, “00” ó “11”, a partir del cual se realizará la transición, por lo que no queda determinado el comportamiento del circuito. Por lo tanto, será aconsejable eliminar la posibilidad de que se produzca dicha transición. Una posible forma de eliminarla será evitar que se puede producir alguna de dichas combinaciones de entrada. La combinación de entrada que se forzará a no producirse será la “11” ya que muestra un solo estado estable, mientras que la combinación “00” muestra dos.

Una vez realizada esta consideración, solamente contaremos con dos estados estables reales: “01” y “10”, lo cual está de acuerdo con el término de biestable (elemento que almacena dos estados estables). Debido a estos dos estados estables, en las situaciones de estabilidad, la salida siempre será “01” ó “10”, por lo que podemos decir que $Z_1 = Z_2$’. Si consideramos la señal Z_1 como señal de salida, podemos describir el comportamiento del circuito de la siguiente forma:

- Cuando R = 1, la salida alcanzará un cero lógico, por lo que a dicha entrada se la conoce como **reset**.
- Cuando S= 1, la salida alcanzará un cero lógico, por lo que a dicha entrada se la conoce como **set**.
- Cuando ambas entradas tengan un nivel bajo, la salida permanecerá sin cambiar de tal forma que se almacena el estado anterior.

Si se va a considerar que las salidas Z_1 y Z_2 no son independientes entre sí, en la tabla de funcionamiento sólo deberá aparecer una de ellas. Según las consideraciones anteriores, sólo tendremos que considerar las filas donde Z_1 y Z_2 son diferentes por ser ambas señales complementarias entre sí. Este comportamiento se muestra en la figura 3.7, pero es necesario comentarla ya que la nueva tabla no se obtiene simplemente eliminando las filas donde Z_1 y Z_2 son iguales. En primer lugar hay que considerar que una tabla muestra el comportamiento estacionario, por lo que el valor de salida debe ser estable para dicha combinación de entradas. Así en la condición de set (RS = "01"), el valor al que llega la señal q (o lo que es lo mismo Z_1) es '1' independientemente desde el valor que parta; por lo que para cualquier condición de entrada debe tener dicho valor. No obstante, en la condición de almacenamiento (RS = "00"), la señal q es estable en cualquier valor que tenga, por lo que deberá permanecer en dicho valor. Finalmente, para que dicho circuito tuviese el comportamiento requerido, había que prohibir la combinación RS = "11".

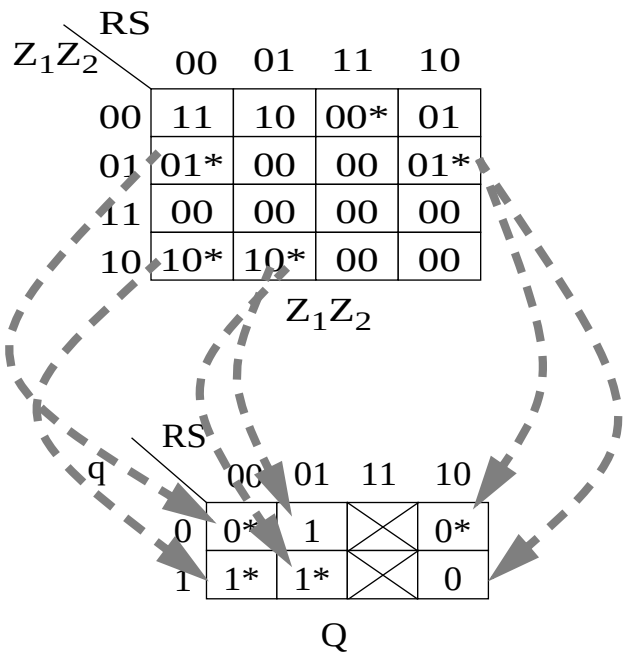

Figura 3.7.- Tabla de comportamiento de un biestable RS utilizando puertas NOR.

Debido a todas estas características, a este elemento de memoria se le conoce como **biestable RS**.

Además de este elemento existen más tipos de biestables dependiendo de su funcionalidad. Dentro de los biestables más empleados, junto al RS, podemos encontrar los siguientes:

- Biestable D.- Almacena el valor de la entrada D.
- Biestable T.- Cuando la entrada T vale '0', se almacena el estado anterior; mientras que cuando T = 1, se cambia el estado almacenado
- Biestable JK.- Es un biestable equivalente al RS, salvo que en la combinación de entrada problemática ("11") se produce un cambio de estado.

En la figura 3.8 mostramos las tablas de estados correspondientes a cada uno de estos biestables, así como la ecuación lógica que siguen.

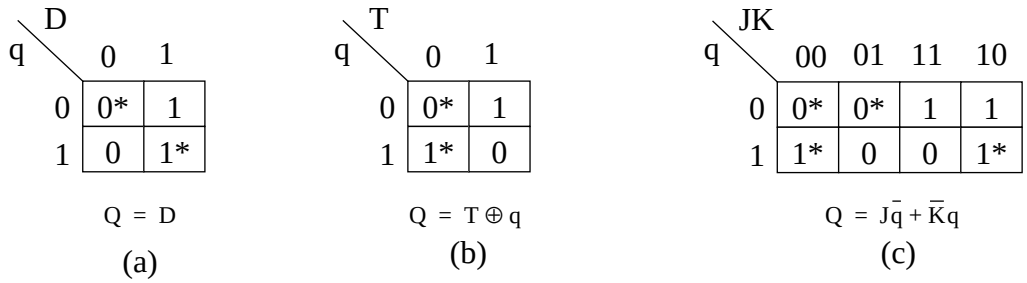

Figura 3.8.- Tablas de estado y ecuaciones lógicas correspondientes a los biestables (a) D, (b) T y (c) JK.

Todos estos biestables tienen una misión en común, almacenar información, por lo que deberíamos ser capaces de poder realizar transformación entre los diferentes tipos. Una razón más evidente es que todos estos elementos son circuitos secuenciales, por lo que debería poder obtener el comportamiento de uno de ellos utilizando otro diferente con el modelo genérico mostrado en la figura 1.3. Veamos como ejemplo el paso de un biestable tipo T utilizando biestables RS, según se muestra en la figura 3.9. Como podemos observar, el problema se reduce a determinar cuanto deben valer las señales R y S (entradas del biestable del que disponemos) en función de la señal T (entrada del biestable que queremos obtener) y la señal de estado (realimentación secuencial). Para determinar dichos valores, realizamos una comparación entre las tablas de comportamiento de los biestables RS y T, de tal forma que obtengamos cuanto deben valer las señales RS para una determinada transición de estados (desde q hasta Q). Una vez obtenida la nueva tabla que tendrá como entradas las señales T y q, y como salidas las señales R y S, se obtienen las fórmulas de conmutación correspondientes que serán equivalentes a la lógica combinacional que hay en el esquema. Por lo tanto, el resultado final se muestra en la figura 3.9(c).

2.2. Latches.

En el apartado anterior hemos considerado los biestables transparentes, por lo que no se encuentran presentes ningún tipo de señal de control. Ahora vamos a considerar los latches, en los que el cambio de estado viene dictado por una señal de control, siendo sensibles al nivel de dicha señal.

La única diferencia existente entre los latches y los biestables transparentes consiste en que durante el nivel inactivo de la señal de control se produce el almacenamiento del estado en el que se encontraba el biestable, a pesar de los posibles cambios en las señales de entrada. Consideremos la siguiente variante del biestable RS, al cual se le han añadido unas puertas AND para unir las señales de entrada RS y la señal C, mostrado en la figura 3.10. Cuando la señal C tenga el valor de un “1” lógico, nos encontraremos ante un biestable transparente RS como el considerado en el apartado anterior. En cambio, cuando la señal C tenga el valor de un “0” lógico, nos encontraremos ante un biestable transparente RS equivalente en el que sus entradas tienen un valor bajo, produciéndose el almacenamiento del estado anterior, aunque la entradas reales RS sufran transiciones. Por lo tanto, el comportamiento de este circuito es el de un latch RS (ya que se basa en un biestable transparente RS) en el que la señal de control será C con el nivel alto como nivel activo.

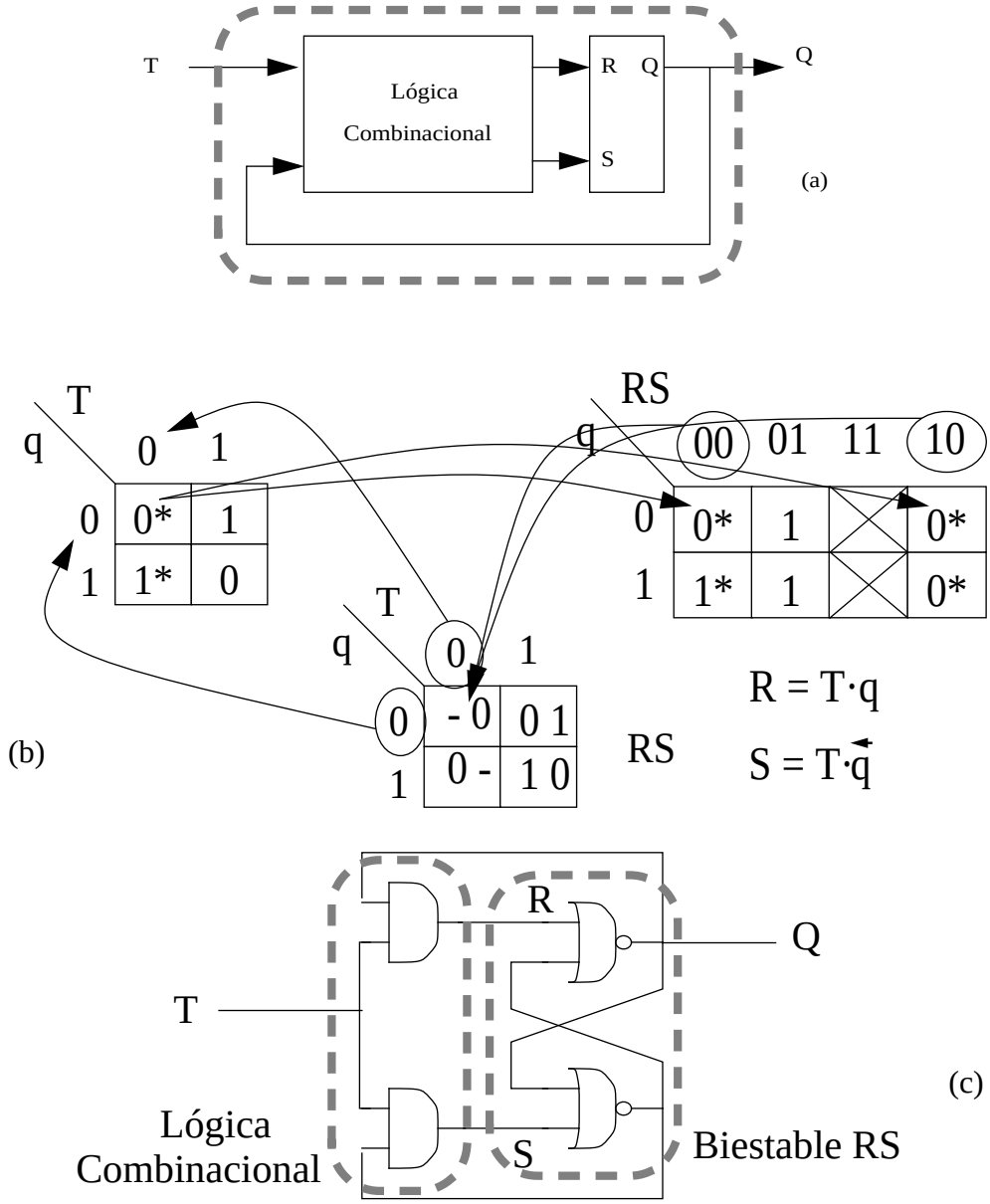

Figura 3.9.- (a) Esquema del biestable convertido en función del utilizado. (b) Determinación de los valores de las señales R y S para la operación como biestable tipo T. (c) Esquema lógico del biestable tipo T.

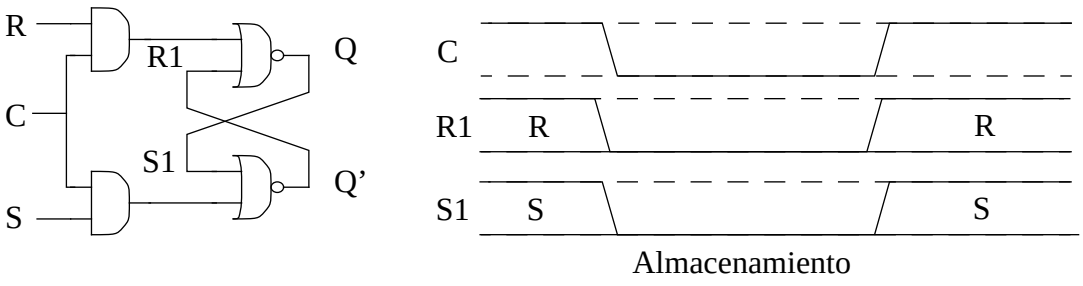

Figura 3.10.- Variante bajo estudio del biestable RS

La aparición de la señal de control provoca un mayor control sobre la propiedad de transparencia de los latches. No obstante, puede que nos interese que algunas señales no estén controladas por dicha señal, como puede ser una señal para inicializar el estado del latch de la forma más rápida posible. Por lo tanto, en los latches podemos encontrar dos tipos de señales:

- **Señales síncronas**.- Son señales de entrada que son controladas por la señal de control, que suele ser denominada **reloj**, de tal forma que la influencia de estas señales depende de la señal de control.
- **Señales asíncronas o directas**.- Son señales de entrada que tienen la propiedad de transparencia, es decir, su influencia no está controlada por ninguna señal adicional (ni siquiera por el reloj), por lo que un cambio en dichas señales provoca una respuesta instantánea (después del retraso de la lógica combinacional) en el comportamiento de los latches. Entre las señales más representativas de este tipo podemos encontrar la señal de *preset* o *set* cuya función es colocar un '1' lógico en la salida del biestable; y la señal de *clear* o *reset*, cuya función es colocar un '0' lógico en la salida del biestable.

En la figura 3.11 se puede ver un biestable RS con ambos tipos de señales. Como ya hemos visto anteriormente, las señales R y S sólo actúan cuando lo indica la señal de control (en este caso, cuando dicha señal toma el valor '1'). No obstante la señal *clear* tiene un efecto directo sobre las señales de salida, de tal forma que cuando dicha señal toma el valor '0', las salida también toma dicho valor (ya que $\overline{q}$ = '1'), independientemente de valor o transición de la señal de control.

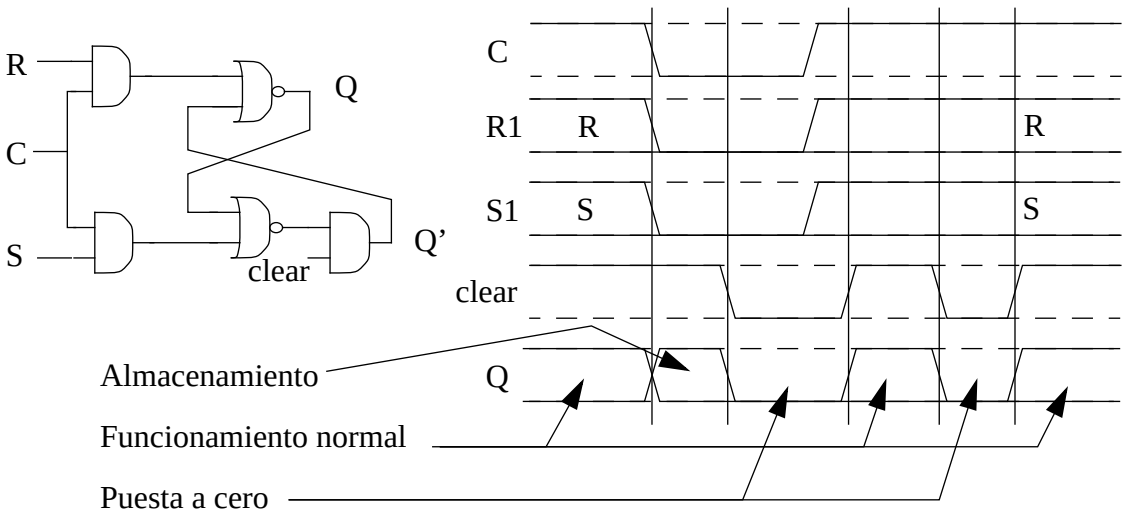

Figura 3.11.- Biestable RS con señales síncronas (R y S) y asíncronas (clear).

De igual forma que hemos encontrado un latch RS, podemos encontrar diferentes tipos de latches partiendo de cada uno de los tipos de biestables transparentes. Por lo tanto, podemos encontrar latches D, T y JK. Las tablas de estado serán las mismas que las correspondientes a sus biestables con la salvedad de que los cambios de estado debidos a las transiciones de las señales síncronas, solamente se podrán producir durante el nivel activo de la señal de control.

2.3. Flip-flops.

Una forma de obtener flip-flops, es decir, biestables sensibles a las transiciones de la señal de control, consiste en utilizar latches en configuración **maestro-esclavo**. Esta configuración consiste en la utilización de dos latches en serie, cuyas señales de control están complementadas.

Estudiemos el siguiente circuito formado por latches tipo D, que se muestra en la figura 3.12. El primer latch es sensible al nivel bajo, mientras que el segundo es sensible al nivel alto. De esta forma el primer latch muestra una dependencia con respecto al dato de entrada, por lo que se denomina maestro. En cambio, el segundo latch no va a tener transiciones de entrada durante el periodo activo de su señal de control ya que el primer latch mantiene constante su salida, que es la entrada al segundo latch, durante dicho periodo. Este comportamiento se muestra en la figura 3.12(b).

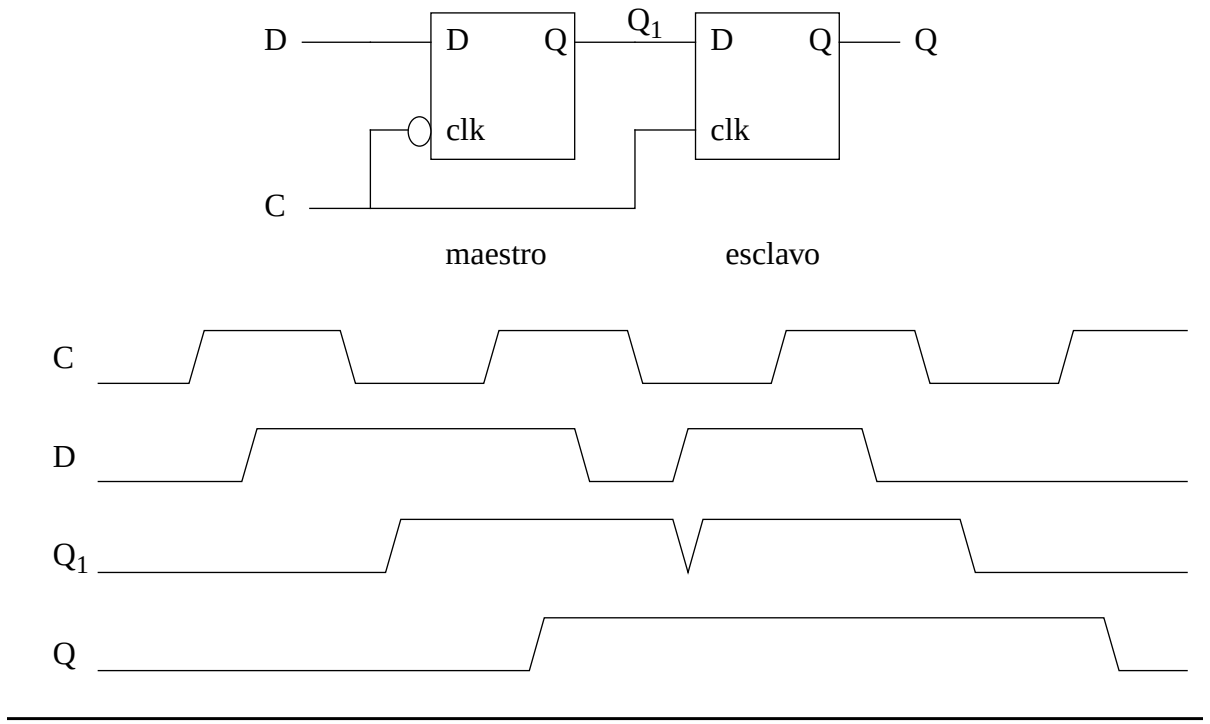

Figura 3.12.- (a) Biestable tipo D en configuración maestro-esclavo y (b) sus formas de onda.

Con esta solución conseguimos que la entrada al latch esclavo no cambie durante el nivel activo de su señal de control, eliminando la propiedad de transparencia en la señal de salida excepto en la transición de la señal de control.

Obviamente esta solución puede ser aplicada a cualquier tipo de latch. No obstante, para mantener de forma sencilla la tabla de estados correspondiente, el latch esclavo se suele utilizar en configuración de latch tipo D. De esta forma, el flip-flop RS en una configuración maestro-esclavo, puede ser el mostrado en la figura 3.13.

La distinción de los símbolos de los latches y flip-flops consiste en que en los últimos, la entrada de control tiene un ángulo (>), y en los latches no. La transición (o nivel) activa de la señal de control se suele identificar de dos formas: con un signo + ó - en el terminal de la señal

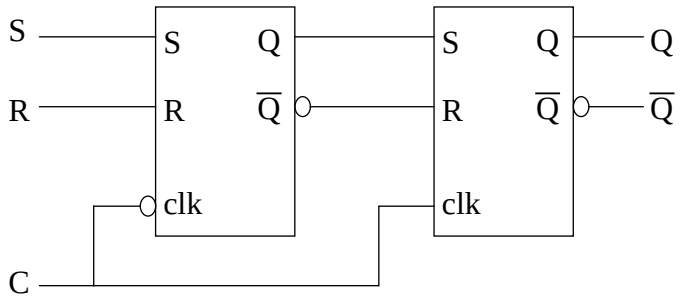

Figura 3.13.- Flip-flops tipo RS en configuración maestro-esclavo.

de control para las transiciones de subida o de bajada respectivamente; o la presencia o no de un círculo para las transiciones de bajada o de subida respectivamente.En la figura 3.14 mostramos los símbolos correspondientes a los latches y flip-flops tipo D.

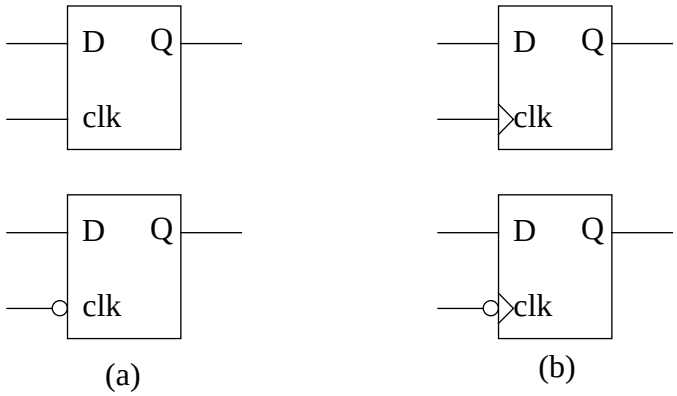

Figura 3.14.- Símbolos correspondientes a los (a) latches y (b) flip-flops tipo D.

2.4. Restricciones Temporales.

Si todos los elementos fuesen ideales, es decir, no tuviesen retrasos de propagación, las transiciones simultáneas (o casi) producirían un comportamiento correcto. No obstante, debido a la no idealidad de estos elementos, debemos imponer una serie de restricciones temporales para asegurar un comportamiento correcto.

Los únicos elementos de memoria que no presentan este tipo de restricciones son los biestables transparentes, ya que en todo momento las salidas siguen a las entradas. O lo que es lo mismo, este tipo de biestables carecen de señales de control.

En el caso de los latches ya tenemos una señal de control, y por lo tanto debemos imponer una serie de restricciones temporales. Básicamente existen tres restricciones temporales de importancia:

- **Tiempo de setup** (t_{setup}).- Es el tiempo necesario que los datos de entrada deben permanecer estables antes de que la señal de control entre en su nivel activo.

- **Tiempo de hold** (t_{hold}).- Es el tiempo necesario que los datos de entrada deben permanecer estables después de que la señal de control haya alcanzado su nivel inactivo.
- **Anchura de pulso de control** (t_w).- Es el tiempo necesario que la señal de control debe permanecer en su nivel activo.

Estos tiempos se ven representados en la figura 3.15.

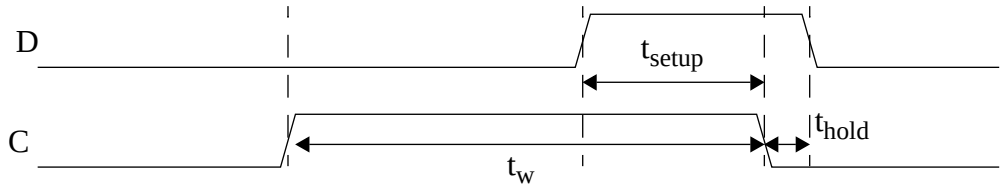

Figura 3.15.- Representación esquemática de las principales restricciones temporales correspondientes a los latches.

En el caso de los flip-flops, el periodo transparente se limita a la transición activa de la señal de control. Por lo tanto, únicamente se definen los tiempos de setup y de hold, correspondiendo a la misma transición de la señal de control. Este caso se representa en la figura 3.16.

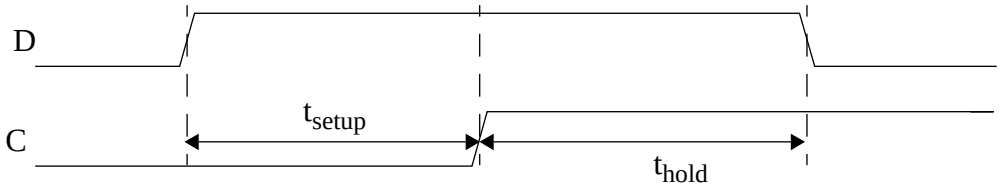

Figura 3.16.- Representación esquemática de las principales restricciones temporales correspondientes a los flip-flops.

3. Sistemas de Almacenamiento. Memorias de Semiconductores.

Cuando nos encontramos con sistemas complejos como pueden ser microprocesadores o microcontroladores, necesitamos disponer de sistemas de almacenamiento masivo. La misión de estos sistemas difiere de la función de los biestables en el caso de los sistemas secuenciales. En estos últimos, los biestables debían almacenar únicamente información relativa a la historia de la ejecución del sistema a través de las variables de estado. En cambio, los sistemas de almacenamiento de sistemas complejos suelen almacenar los datos involucrados en el proceso, ya sean de entrada, salida o intermedios, y las instrucciones que hay que seguir en este proceso.

Tanto los datos como las instrucciones seguirán un tipo de codificación que involucrará a más de un bit. Por lo tanto,

> Una **palabra** se denomina al conjunto de bits que pertenecen a la misma instrucción o al mismo dato,

es decir, el **tamaño de una palabra** es el número de bits a los que se accede para obtener una sola información, ya sea un dato o una instrucción. Obviamente, en estos sistemas que en ade-

lante los denominaremos simplemente memorias, debemos poder almacenar más de un dato o instrucción, por lo que necesitaremos un identificador para poder distinguir las diferentes palabras.

> Se conoce como **dirección** al conjunto de bits que identificarán unívocamente a cada palabra de la memoria.

De esta forma, la estructura básica de una memoria estará formada por una matriz de elementos de memoria, en la que se guardará la información, y un sistema que decidirá la palabra a la que queramos acceder mediante su dirección, como esta dirección también estará codificada, dicho sistema será un decodificador. Por lo tanto, un posible esquema de una memoria se puede ver en la figura 3.17.

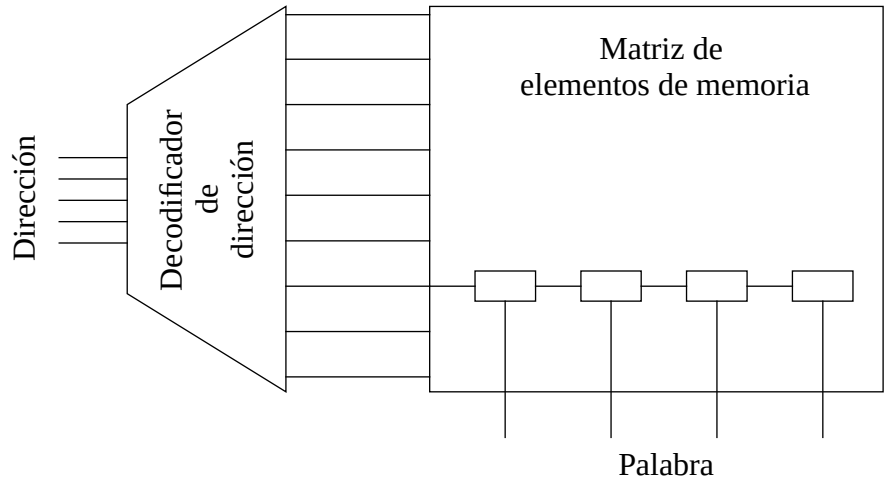

Figura 3.17.- Esquema básico de un sistema de almacenamiento (memoria).

El funcionamiento de una memoria puede ser el siguiente. En primer lugar se le suministra la dirección de la palabra a la cual se quiere acceder. Esta dirección activará solamente una de las salidas del decodificador. Esta activación habilitará la conexión entre su palabra asociada y la salida de la memoria, permitiendo el acceso a dicha palabra. Como es lógico, las únicas operaciones que puede realizar una memoria son operaciones de lectura o escritura. En el caso de escritura, se tiene que poner en la entrada la información que se quiere almacenar.

Una definición más formal de memoria podría ser la siguiente:

> Una **memoria** se define como un sistema capaz de almacenar información, la cual será suministrada en cualquier momento que un elemento conectado a ella la solicite.

3.1. Características de las Memorias.

En una memoria de semiconductores, los principales parámetros serán la cantidad de información que es capaz de almacenar, la velocidad con la que se accede a la información y su coste. Particularmente, estos parámetros estarán entre las principales características de las memorias, cuyas nomenclaturas son capacidad de almacenamiento, tiempo de acceso y coste por bit.

Una posible definición de la capacidad de almacenamiento podría ser la siguiente:

La **capacidad de almacenamiento** se define como la cantidad de información que podemos almacenar en nuestro sistema de memoria.

Esta propiedad está directamente relacionada con el tamaño de la palabra y de la dirección, ya que el primero nos indica el número de bits que tiene la palabra y el segundo nos indica el número de palabras que tiene nuestro sistema. Por lo tanto, si tenemos un sistema con una palabra de 4 bits y una dirección de 10 bits, tendremos un sistema con una capacidad de

$$2^{10} \text{ (palabras)} * 4 \text{ (bits por palabra) bits} = 4 \text{ Kbits}$$

No obstante, y en función de nuestro sistema, puede que toda la información que almacenemos no sea útil para el desarrollo de las instrucciones, es decir, no sean ni datos ni instrucciones. Por lo tanto, podemos definir una **capacidad bruta o total**, que correspondería a la cantidad total de información, sea o no útil, y una **capacidad neta o útil**, que correspondería a la cantidad de información útil que se puede almacenar.

El tiempo de acceso podría definirse de la siguiente forma:

El **tiempo de acceso** será el tiempo transcurrido desde que se suministra la dirección hasta que se accede a la palabra deseada.

En función de nuestro dispositivo, este tiempo puede ser el mismo para todas las palabras, por lo que el **acceso** se denomina **aleatorio**, o puede ser diferente para todas las palabras, por lo que el **acceso** se denomina **secuencial**. Existe una situación intermedia entre ambos tipos en la cual el tiempo es el mismo para un determinado grupo de palabras, denominándose entonces **acceso directo**. En nuestro caso particular de las memorias de semiconductores, se verifica que todas muestran un acceso aleatorio.

Existe otro tipo de acceso que no se diferencia por el tiempo de acceso. El tiempo de acceso es aleatorio en el sentido de que a todas las palabras se accede con el mismo tiempo. No obstante en el caso de las lecturas, no se identifica la palabra por la dirección, sino por cierta parte del contenido, de ahí que sea conocido como **acceso por contenido**. Así, la palabra de este tipo de memoria se divide en dos partes: un **descriptor**, que hará las veces de la dirección, y los **datos**, que será la parte realmente útil. Para llegar a seleccionar una palabra se van comparando los descriptores de todas las palabras almacenadas, de forma paralela, con el descriptor de la palabra a la que se quiere acceder. El tipo de memoria que utiliza este acceso es la **memoria cache**. En el caso de la escritura, se utiliza el mismo mecanismo de la memoria RAM estática.

El coste por bit podría tener la siguiente definición:

El **coste por bit** será el precio que cuesta almacenar un bit de información.

O lo que es lo mismo, será el precio de una memoria dividido por la capacidad total de almacenamiento.

En función de estas características, una memoria ideal sería aquella que tendría la máxima capacidad de almacenamiento con el menor tiempo de acceso y coste por bit. No obstante, estas características no son independientes sino que están muy relacionadas entre sí, según las gráficas mostradas en la figura 3.18.

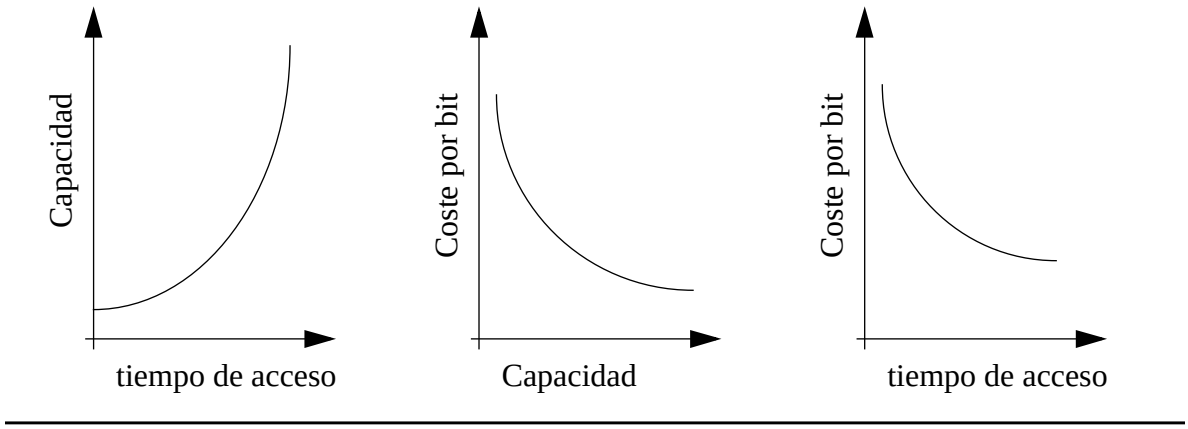

Figura 3.18.- Relación entre las principales características de las memorias.

Según estas gráficas, a medida que el tiempo de acceso aumenta, el coste por bit disminuye y la capacidad de almacenamiento aumenta. Esto implica que actualmente no podemos pensar llegar a la memoria ideal. Para tratar de acercarnos a dicha memoria se utiliza la **jerarquización de la memoria** que consiste en utilizar diferentes módulos con diferentes características.

La jerarquización de la memoria tiene como objetivo que el procesador vea una memoria ideal, es decir, con máxima capacidad y mínimo tiempo de acceso. Para ello se basa en la utilización de módulos de diferentes características, de tal forma que se aproveche únicamente la ventaja de cada uno de ellos. Este esquema se muestra en la figura 3.19. En este sistema, cada nivel es una copia de una parte del nivel inmediatamente inferior, tal que el procesador siempre (o en la mayoría de los casos) trate directamente con el módulo de nivel superior, con un tiempo de acceso menor. En cuanto a la capacidad, como la información va pasando de nivel en nivel, según las necesidades, la capacidad que ve el procesador es la del nivel inferior, con una capacidad mayor. Con respecto al coste por bit, también disminuye debido al coste de los niveles inferiores, con un coste menor. Por último, el controlador de la memoria debe ser capaz de decidir cuando la información debe fluir a través de la jerarquía, así como un esquema para que este flujo sea lo menor posible disminuyendo el tiempo de acceso global.

3.2. Tipos de Memorias de Semiconductores.

De todas las clasificaciones existentes, una de las más importantes es la regida según los criterios de permanencia de la información. Antes daremos las siguientes definiciones:

> Una memoria se considera **permanente** cuando, una vez haya sido almacenada la información, su contenido no se puede alterar.

Por lo tanto, en una memoria de este tipo sólo se podrán realizar operaciones de lectura, ya que solamente se podrá realizar una escritura.

> Una memoria se considera **volátil** cuando su contenido es destruido al desconectar la alimentación del sistema, en caso contrario de denomina **no volátil**.

Por lo tanto, este tipo de memoria no se puede utilizar para almacenar datos de forma permanente.

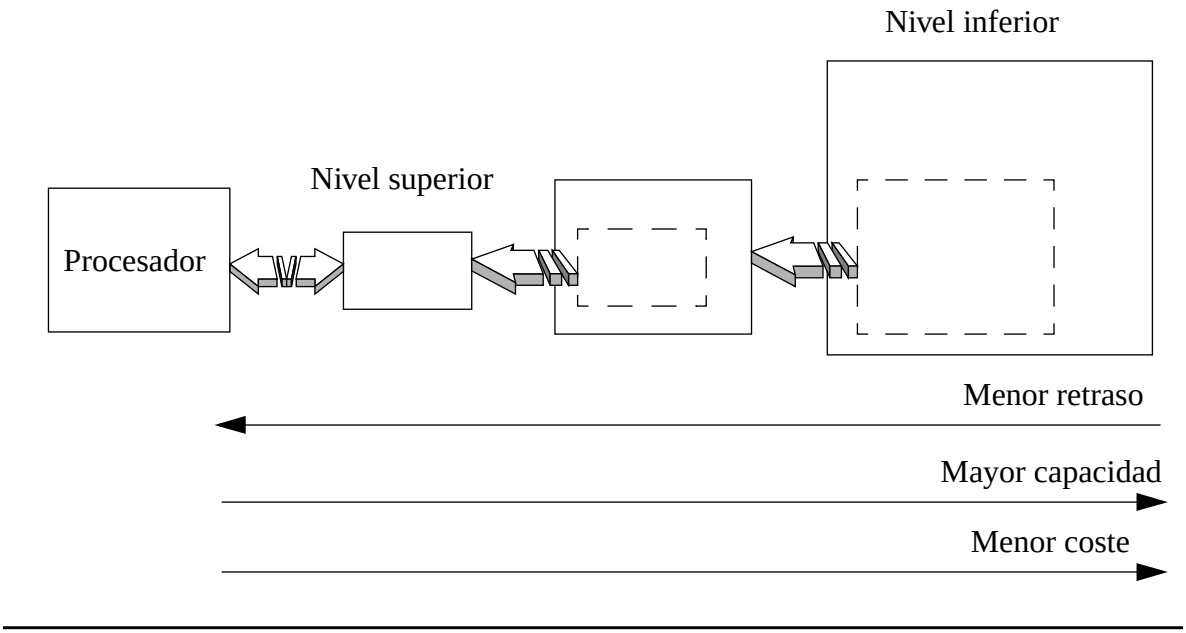

Figura 3.19.- Jerarquización de un sistema de memoria.

Una memoria se considera **dinámica** cuando su contenido es destruido cuando ha transcurrido un determinado intervalo temporal, aunque la fuente de alimentación no haya sido desconectada.

Por lo tanto, para evitar la pérdida de la información es necesario utilizar ciclos de refresco, ya que se basan en biestables dinámicos. Hay que notar que esta última definición es propia de las memorias de semiconductores. En el caso de memorias genéricas, la definición de memoria dinámica es otra diferente, la cual no es objeto de esta asignatura.

Según estos criterios, las memorias de semiconductores se pueden clasificar en:

- Memorias ROM y PROM.- Son memorias permanentes, y por lo tanto no volátiles. La diferencia entre ellas radica en el proceso de escritura de la información: en las memorias ROM, la información es almacenada por el fabricante; mientras que en las memorias PROM, la información es almacenada por el usuario a través de un programador.
- Memorias EPROM.- Son memorias no volátiles que permiten múltiples grabaciones. No obstante, el proceso de grabación es relativamente complejo y su uso principal es como memoria permanente. Dentro de este tipo podemos encontrar dos clases:
  - Memorias EPROM-FLASH (o solamente EPROM), en las cuales el proceso de grabación se realiza a través de radiación ultravioleta.
  - Memorias EEPROM, en las cuales el proceso de grabación se realiza de forma eléctrica a través de altos potenciales.
- Memorias NOVRAM (o RAM no volátiles).- Son memorias no volátiles con la ventaja de que tienen los mismos tiempos de acceso de la memoria RAM estática. De hecho, una memoria NOVRAM está formada por una memoria RAM estática y una memoria EEPROM, de tal forma que salvo en la conexión o desconexión de la alimentación está trabajado la memoria RAM. Estas memorias necesitan un controlador

especial que genera las señales de escritura y de lectura en la memoria EEPROM previo a la desconexión o conexión de la alimentación.

- Memorias RAM estáticas.- Son memorias volátiles estáticas, por lo que no necesitan ningún tipo de refresco.
- Memorias RAM dinámicas.- Son memorias volátiles dinámicas, por lo que son necesarios ciclos de refresco para mantener almacenada la información.

3.3. Diseño de un sistema de memoria.

En primer lugar debemos considerar que los sistemas de memoria de semiconductores se suelen utilizar en sistemas de procesado, ya sea con procesadores o microprocesadores. Por lo tanto, en la memoria se deben almacenar dos tipos de información bien diferenciadas:

- Instrucciones y datos necesarios para la inicialización del sistema
- Instrucciones y datos para la ejecución de las aplicaciones

Luego las memorias en las que se deben almacenar ambos tipos de informaciones tendrán características diferentes. Para el primer caso, necesario para la inicialización del sistema, se deben utilizar memorias no volátiles, ya que deben permanecer cuando se desconecta la alimentación. Por lo general, esta información no cambia a no ser que transcurra un tiempo relativamente alto; luego se suelen utilizar **memorias ROM o EPROM**, en función de si cambiarán o no. Para el segundo caso, la información deberá cambiar ya que los datos (como mínimo) cambiarán. Luego la memoria debe ser de lectura-escritura, es decir, no permanente; además suele necesitar que se realizan un número alto de lecturas y escrituras, por lo que el proceso de escritura no debe ser complejo, eliminado las memorias EPROM. Además no necesitaremos tener almacenado estos programas en cuanto se conecte la alimentación, por lo que serán utilizadas **memorias RAM**.

Una vez que hemos determinado el tipo de módulos que hay que utilizar, memorias ROM y RAM, deberemos situar cada tipo de memoria en una posición (dirección) bien determinada). Este proceso se denomina **configuración del sistema de memoria**. No existe ningún estándar de la colocación de los diferentes tipos de memoria sino que viene determinada por las especificaciones del diseño; lo que se suele tomar como estándar es que los módulos de un mismo tipo se ponen contiguos.

Consideremos la configuración mostrada en la figura 3.20. En él podemos observar las siguientes características:

- las direcciones están codificadas en hexadecimal para que las direcciones no contengan un número excesivo de dígitos
- dispondremos de 64K palabras ya que cada dirección tiene 16 bits.
- El espacio reservado para la inicialización del sistema (almacenado en memoria ROM) es de 16 K palabras en la parte inferior, ya que vamos de la palabra 0000h hasta 3FFFh.
- El espacio reservado para los datos y aplicaciones (memoria RAM) es de 32 K palabras siguientes a la inicialización, ya que tenemos 48 K menos las 16 K de la inicialización.

- El resto del espacio de memoria se considera para posibles ampliaciones de memoria. Este resto es de las últimas 16 K.

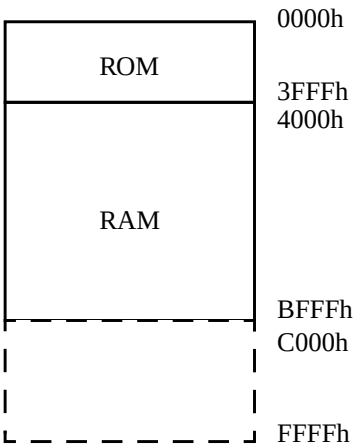

Figura 3.20.- Ejemplo de una configuración de un sistema de memoria.

Como ya vimos en la figura 3.17, un sistema de memoria está formado por un decodificador, que selecciona la palabra, y unas celdas de almacenamiento. Cuando el decodificador que necesitamos es mayor de los que disponemos, tenemos que implementar dicho decodificador utilizando más decodificadores menores. Esta situación se muestra en la figura 3.21. En ella podemos apreciar las siguientes características:

- los decodificadores son activos a nivel bajo, ya que las señales de selección suelen ser activas a dicho nivel.
- las señales de dirección de todos los decodificadores del mismo nivel son las mismas, ya que sólo se selecciona un decodificador por nivel a través del canal CS.
- Los decodificadores de los niveles superiores seleccionan a los decodificadores del siguiente nivel, excepto el último nivel que selecciona a la palabra en cuestión.

No obstante, no se suelen disponer de dispositivos que almacenan una sola palabra, sino que disponemos de módulos que almacenan un grupo relativamente grande de palabras. Estos módulos pueden ser vistos como el último nivel de decodificadores con los elementos de almacenamiento de las palabras. Estos módulos están caracterizados por el tipo de memoria, el número de palabras y el tamaño de la palabra; por lo que un módulo ROM 1Kx8 será un módulo del tipo ROM con un total de 1K palabras y cada palabra tiene 8 bits.

Por lo tanto, si disponemos de módulos ROM y RAM de 16 K con el mismo tamaño de palabra que nuestro sistema anterior (figura 3.20), necesitaríamos un módulo ROM (16K) y dos módulos RAM (32K). Para seleccionar uno de los módulos necesitaremos un decodificador 2:4, ya que 64K / 16K = 4.

El siguiente paso será conectar las señales de dirección. Para tener 64K palabras necesitaremos 16 señales de dirección (2^{16} = 64 K, donde 1K = 2^{10}). De estas 16 señales, 14 deben estar conectadas a los módulos (2^{14} = 16 K), mientras que las cuatro restantes deben conectarse al decodificador. Para que las palabras consecutivas pertenezcan al mismo tipo de memo-

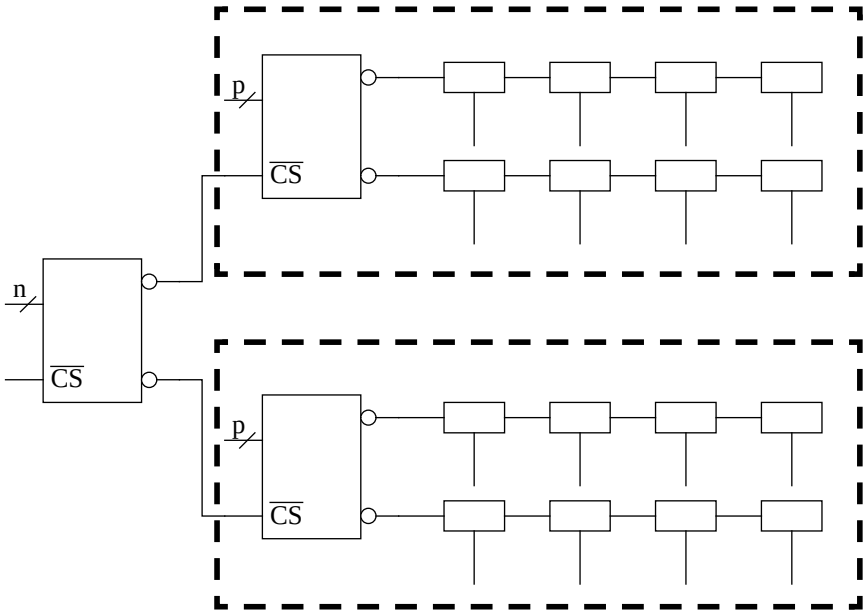

Figura 3.21.- Esquema de un sistema de memoria con varios niveles de decodificadores.

ria (módulo) las señales menos significativas deben conectarse a los módulos, mientras que las más significativas se conectarán al decodificador.

Las señales de datos de todos los módulos se conectan juntas ya que la codificación triestado permite esta conexión cuando solamente una palabra es seleccionada. La codificación triestado es aquella que tiene tres estados diferentes: ‘0’, ‘1’ y ‘Z’, donde ‘Z’ se conoce como estado de alta impedancia. El estado ‘Z’ es aquel que no existe ninguna conexión a ningún valor lógico. Luego, si una sola de las salidas tiene una conexión a un valor lógico y el resto están a ‘Z’, sólo hay una conexión a un valor lógico por lo que no hay ningún problema de la conexión directa. De hecho, este tipo de conexión funciona igual que un multiplexor.

Por último conectamos las señales de operación, lectura y escritura, a los terminales correspondientes. Debemos tener en cuenta que los módulos ROM sólo tendrán terminal de lectura y no de escritura.

Este sistema se muestra en la figura 3.22.

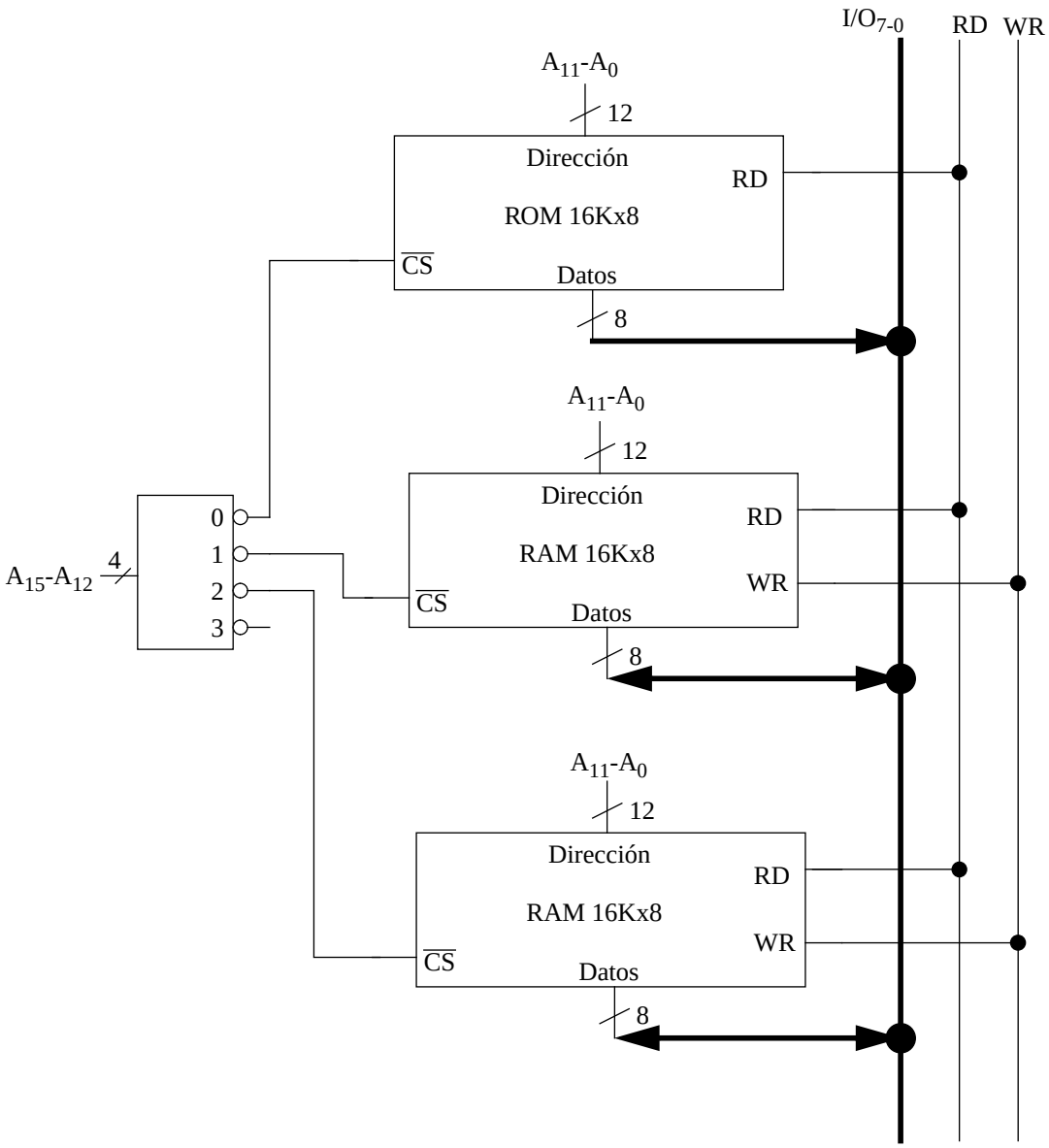

Figura 3.22.- Sistema de memoria correspondiente a la configuración de la figura 3.20 con un tamaño de palabra de 8 bits.