13 SINCRONISMO Y BIESTABLES SÍNCRONOS SECUENCIADORES LÓGICOS PROGRAMABLES

13.1. Sincronismo y configuración amo/esclavo: biestables síncronos
13.2. Registros síncronos
13.3. Tiempos funcionales e inicialización de los biestables
13.4. Dispositivos lógicos programables: PAL con biestables
13.5. Los biestables en VHDL

«Dividir en partes» es la forma típica de abordar la complejidad. El sincronismo es una partición del tiempo que lo divide en unidades: supone que la variable tiempo deja de ser continua y pasa a variable discreta, fraccionada en unidades numerables.

Un sistema secuencial síncrono es aquel cuyos cambios de estado se producen solamente en los momentos de paso de una unidad de tiempo a la siguiente; tales instantes vendrán definidos por los flancos de una señal digital que cuantifica el tiempo en unidades y a la que, genéricamente, denominaremos reloj.

Para construir sistemas secuenciales síncronos necesitamos biestables síncronos, cuyos cambios de estado se produzcan solamente en el flanco activo de la señal de reloj. Aparece un nuevo «elemento de entrada» que no es ni el 0 ni el 1, sino el paso de 0 a 1 (flanco de subida); para conseguir un biestable «habilitado por flancos» es necesario utilizar dos biestables asíncronos, uno de ellos habilitado por valor 0 y el otro por 1, de forma que el conjunto responda al flanco de la señal: configuración amo/esclavo.

A partir de la combinación amo/esclavo (biestable síncrono tipo D) puede construirse una amplia diversidad de tipos de biestables y, en particular, el biestable JK que sustituye directamente al biestable RS en el diseño síncrono de sistemas secuenciales. Por otra parte, la conexión en serie de biestables D da lugar a registros de desplazamiento que permiten la conversión serie-paralelo (y, también, paralelo-serie) de una palabra binaria.

El correcto funcionamiento de un diseño síncrono exige que se respeten los tiempos funcionales de los biestables que lo conforman (tiempos de propagación, de anticipación y de mantenimiento) y que, en el momento inicial, se establezca en ellos el valor adecuado.

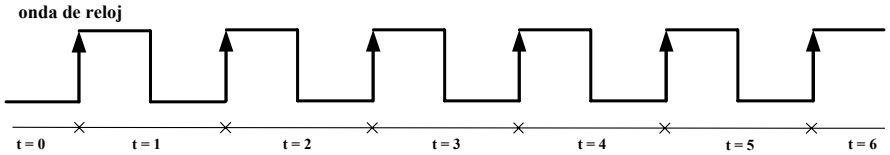
Recordemos que los bloques programables tipo PAL permiten configurar en su interior conjuntos de funciones booleanas en forma de suma de términos producto. Análogamente, se construyen bloques PAL + biestables (añadiendo un biestable tipo D en la salida de cada módulo PAL) que permiten recoger la evolución de las variables de estado; tales bloques reciben el nombre de PLS (secuenciadores lógicos programables).

Diversas mejoras sobre el módulo PAL + biestable han dado lugar al concepto de «macrocelda» programable; actualmente se dispone de circuitos integrados con múltiples macroceldas y canales de interconexión entre ellas (CPLDs: dispositivos lógicos programables complejos), capaces de configurar, por programación, sistemas digitales completos (pues el número de macroceldas en un integrado puede ser muy alto).

Tomás Pollán Santamaría. Tecnología Electrónica. E.U.I.T.I.Z. Universidad de Zaragoza. tpollan@posta.unizar.es

13.1. Sincronismo y configuración amo/esclavo: biestables síncronos

El sincronismo implica una *cuantificación del tiempo* en unidades discretas, definiendo con precisión el instante en que se pasa de una unidad a la siguiente: las transiciones (los cambios de estado) tendrán lugar en dichos instantes y sólo en ellos.

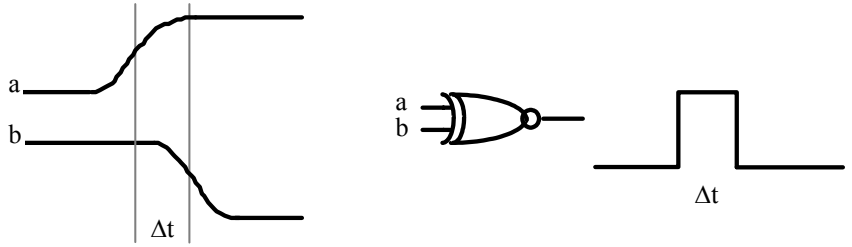


El paso de una unidad temporal a la siguiente vendrá definido por uno de los flancos de la onda de *reloj*; en la figura se ha considerado como activo el flanco positivo ↑ pero de igual forma podría tomarse el flanco negativo ↓.

La sincronización, además de coordinar temporalmente el funcionamiento de los biestables, proporciona una gran seguridad de funcionamiento.

Consideremos, por ejemplo, un biestable **RS** asíncrono que ha de ser activado cuando el valor booleano de dos variables intermedias de un circuito sea idéntico: $S = a.b + \bar{a}.\bar{b}$ supongamos que en un determinado momento la variable **a** que estaba en **0** pasa a **1** y que, a la vez, **b** pasa de **1** a **0**: el biestable no debe activarse.

Puede suceder que, por diferencia de tiempos de propagación, las ondas correspondientes a las variables **a** y **b** que lleguen a las entradas de la lógica de activación del biestable sean del tipo de la figura, superponiéndose durante un pequeño intervalo de tiempo Δt, lo cual da lugar a un pulso espurio de breve duración (*glitch*), que pudiera ser suficiente para disparar el biestable.



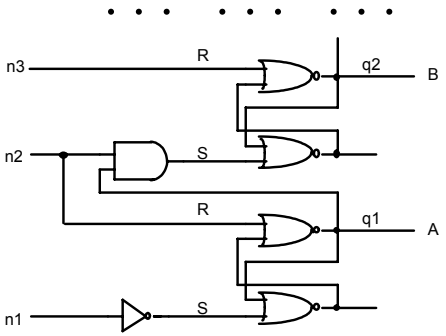
Este marcado no deseado puede ser evitado mediante la utilización de biestables síncronos: con ellos se consigue que todas las variables del circuito conmuten a la vez (en los flancos activos del reloj). Y, lo que es más importante, que los valores que determinan las transiciones sean los presentes en el momento anterior al flanco activo del reloj; no importa, pues, que existan retrasos desiguales en la llegada de las señales a los diversos biestables, ya que son los valores consolidados previos a la transición (y no los que se producen durante ella) los que la controlan.

Otro ejemplo relativo a la seguridad de funcionamiento: en el caso del depósito con mezcla de 4 líquidos, en la codificación con «un solo uno» (apartado 11.4, página 30), simplificando al máximo las condiciones de marcado y de borrado resulta que:

q_1: $S = \overline{n_1}$ $R = n_2$

q_2: $S = q_1.n_2$ $R = n_3$

el primer biestable se borra con **n_2** y el segundo se marca con **n_2** cuando el primero está en **1**: la misma señal **n_2** que borra al primer biestable debe marcar al segundo de ellos.



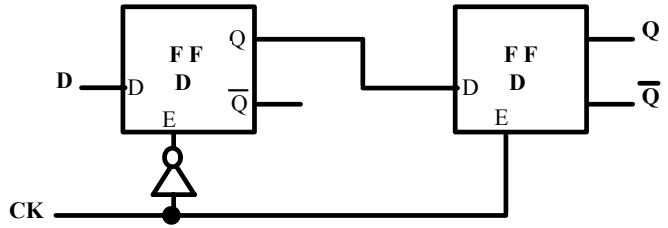
Ahora bien, el borrado de **q_1** es más rápido que el marcado de **q_2** ya que éste debe atravesar una puerta "**y**" adicional (que introduce el correspondiente retraso); es fácil (y así ocurre normalmente si se monta este circuito en laboratorio) que se produzca el borrado del primer biestable y, al hacerse **$q_1 = 0$**, desaparezca la condición de marcado del segundo biestable y no se llegue a marcar.

En la solución dada en la página 30 se evitaba esta situación haciendo que el borrado de **q_1** también pasara por dicha puerta "**y**" (lo cual es correcto). Pero la posibilidad de error por diferencia en el tiempo de marcado/ borrado de las diversas variables de estado está siempre presente y ha de tenerse en cuenta, salvo que se utilice código Gray (en cuyo caso nunca se modifican dos variables de estado a la vez).

De nuevo cabe decir que este tipo de errores, debidos a diferencia de tiempo en la propagación de señales que deben actuar simultáneamente, se evita con la utilización de biestables síncronos: en un diseño síncrono son los valores previos al flanco activo del reloj los que determinan las transiciones (y no les afecta el transitorio de conmutación de las variables de estado en dicho flanco).

Biestables síncronos son aquellos en que los cambios de estado, es decir los cambios del valor booleano memorizado, se producen solamente en el flanco activo de la señal de *reloj*.

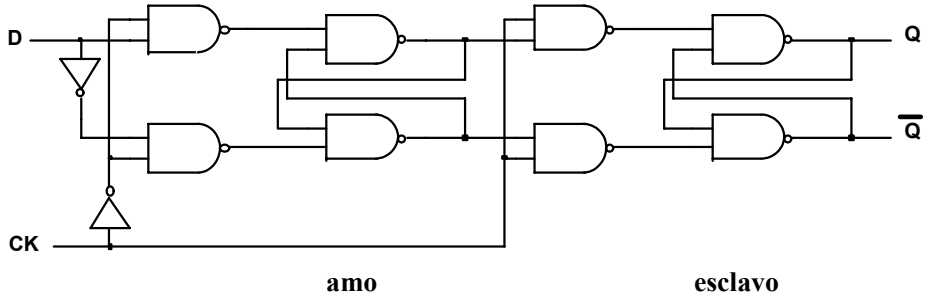
La actuación «por flancos» no corresponde directamente al álgebra booleana: los sistemas lógicos actúan por niveles, correspondientes a los valores booleanos **1** / **0**. Para conseguir que un biestable «síncrono» modifique su salida solamente en los flancos de la señal de reloj es preciso utilizar dos biestables conectados en serie (uno detrás del otro) y de manera que el primero de ellos se habilite con el valor **0** de la onda de reloj y el segundo con el valor **1** de dicha onda.



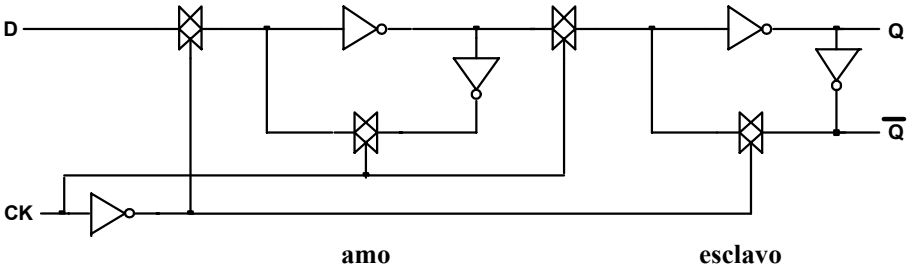
De esta forma para **CK=0** el primer biestable recoge el valor lógico presente en su entrada **D** y lo almacena y cuando **CK=1** el primer biestable transmite el valor almacenado al segundo: el conjunto actúa globalmente cuando **CK** pasa de **0** a **1**, es decir, en las subidas de la señal de reloj **CK**.

Este tipo de estructura que agrupa dos biestables (tipo **D**, con habilitación por niveles **E**) para obtener un biestable síncrono (tipo **D**, con habilitación por flancos **CK**) se denomina configuración *amo/esclavo* (*master/slave*: el segundo biestable actúa como «esclavo» del primero, repite su mismo valor booleano: es «la voz de su amo»).

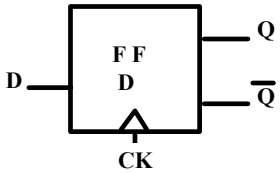
La configuración *master/slave* puede construirse con puertas "**y-negada**" (*Nand*) o con puertas "**o-negada**" (*Nor*).



Asimismo, la construcción de biestables síncronos puede hacerse utilizando puertas de transmisión [en tecnologías MOS la configuración de biestables con puertas de transmisión requiere menor número de transistores, por lo cual ocupa menor área de integración y presenta menores tiempos de propagación]:

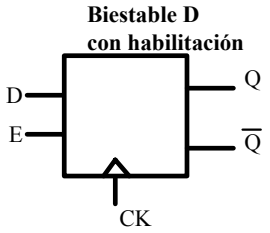


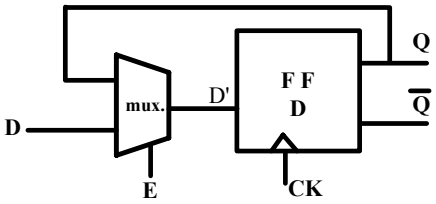
La combinación amo/esclavo da como resultado un biestable tipo **D** síncrono; su funcionamiento es análogo al biestable **D** con habilitación por niveles **E**, pero la adquisición del dato presente en la entrada **D** se produce solamente en los flancos activos de la onda de reloj (entrada **CK**, *Clock*).

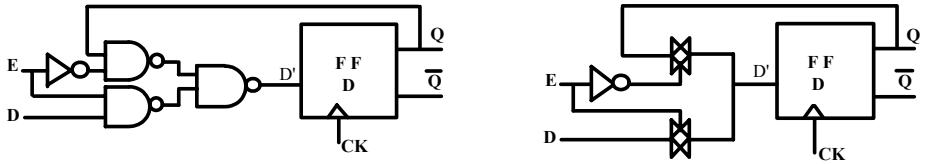


La entrada de reloj, como señal de sincronismo, se distingue mediante un pequeño «triángulo» que indica que dicha señal actúa «por flancos» (y sirve, a la vez, para diferenciar a los biestables síncronos). Hemos supuesto que el flanco activo corresponde a la «subida» (paso de valor **0** a valor **1**), pero de igual modo puede serlo la «bajada» de la onda (paso de valor **1** a valor **0**); en tal caso, indicaremos que el reloj actúa con flancos descendentes mediante un cÍrculo (inversor) en la entrada de reloj.

La siguiente figura representa un biestable síncrono tipo **D** dotado de una entrada de habilitación **E**: cuando **E=0** el biestable conserva el valor anterior (para ello, la entrada ha de recibirlo de la salida **Q**), mientras que para **E=1** el biestable recibe un nuevo dato en el flanco activo del reloj: $D' = Q(t+1) = \overline{E}.Q + E.D$



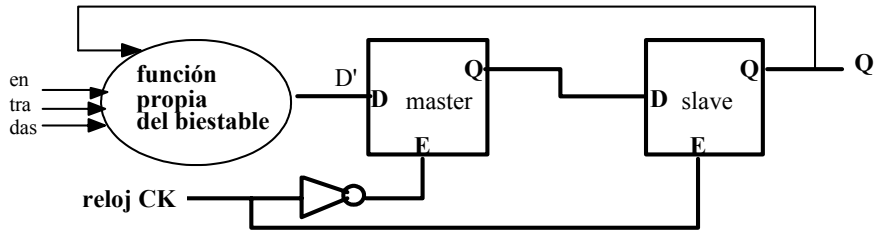




Biestable síncrono D con habilitación E

Dado que para **E = 0** este biestable no debe cambiar de estado, podría pensarse en un diseño «más simple», interrumpiendo el paso de la señal de reloj mediante una puerta "**y**" controlada por la entrada de habilitación **E**. Tal diseño sería incorrecto, ya que nunca se debe «hacer lógica» sobre la señal de reloj: conformar funciones booleanas con dicha señal rompe el sincronismo, pues provoca la aparición de flancos espurios en la entrada de reloj. [Véase el apartado 15.5. *Precauciones relativas a la señal de sincronismo*].

Como en el caso anterior, a partir de la configuración amo/esclavo (biestable síncrono básico, tipo **D**) pueden construirse otros tipos de biestables síncronos: para ello, delante de la entrada **D'** de la configuración *master/slave* se añadirá la función que corresponda al comportamiento que se desea para el biestable; función que puede obtenerse a través de su tabla funcional: entradas, **Q(t)** $\rightarrow$ **Q(t + 1)**



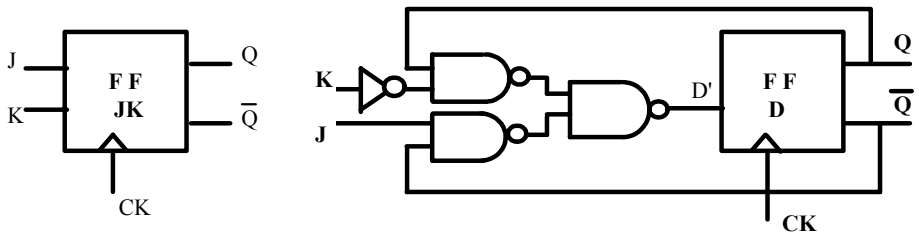
El biestable **JK** corresponde, en forma síncrona, al biestable **RS**, pero su tabla de operación incluye también el caso de activación de ambas entradas a la vez (**J=K=1**): la entrada **K** actúa para el borrado (**≡ R**) y la **J** para la puesta a **1** (**≡ S**) y cuando se activan ambas a la vez el biestable cambia de estado. Bien entendido que las transiciones se efectúan sólo en los flancos activos de la onda de reloj y lo hacen conforme a los valores booleanos de **J** y **K** inmediatamente anteriores a dichos flancos.

J(t)	**K(t)**	**Q(t+1)**
0	**0**	no cambia
0	**1**	**0**
1	**0**	**1**
1	**1**	cambia de estado (**0→1**, **1→0**)

La función booleana previa que ha de añadirse a la configuración amo/esclavo corresponde a la siguiente «tabla de verdad»:

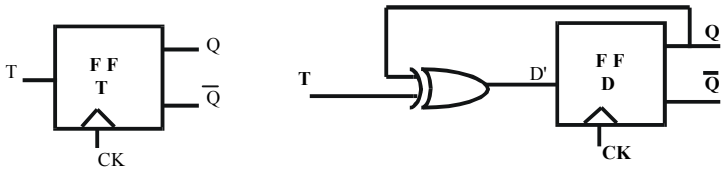
Q(t)	J	K	Q(t+1)
0	**0**	**0**	**0**
0	**0**	**1**	**0**
0	**1**	**0**	**1**
0	**1**	**1**	**1**
1	**0**	**0**	**1**
1	**0**	**1**	**0**
1	**1**	**0**	**1**
1	**1**	**1**	**0**

$$D' = Q(t+1) = \overline{K}.Q + J.\overline{Q} = (\overline{K} * Q) * (J * \overline{Q})$$



El biestable síncrono tipo **T** (*Toggle*) resulta muy útil para la construcción de contadores: cuando su entrada **T=0** el biestable no modifica su estado, mientras que para **T=1** el biestable conmuta (cambia de estado) en cada flanco activo del reloj:

$$D' = Q(t+1) = \overline{T}.Q + T.\overline{Q} = T \oplus Q$$

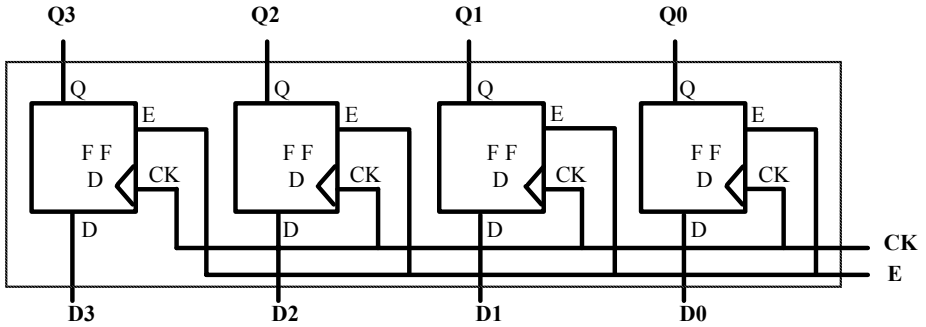


También podemos considerar el biestable tipo **T** sin entrada que cambia de estado cada vez que recibe un flanco activo del reloj: $D' = \overline{Q}$

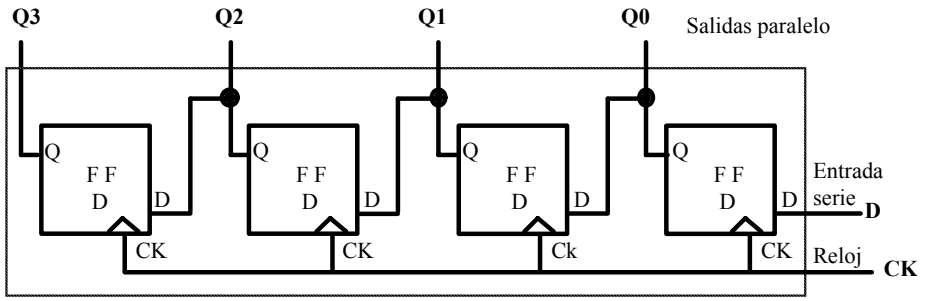
Como complemento a este tema dedicado al «buen manejo del tiempo» (mediante el sincronismo), el apéndice A4 (*Temporizadores: osciladores y monostables*) describe los monostables y los astables, circuitos que, junto con los biestables, forman el grupo de los *multivibradores*. Los biestables son las celdas básicas de memoria y, también, de sincronización de los circuitos secuenciales; astables y monostables son muy útiles en el diseño secuencial, para generar ondas de reloj y como temporizadores, respectivamente.

13.2. Registros síncronos

Agrupando **n** biestables **D** en paralelo, con sus entradas de reloj y de habilitación comunes, se configura un registro síncrono, capaz de almacenar una palabra de **n** dígitos.

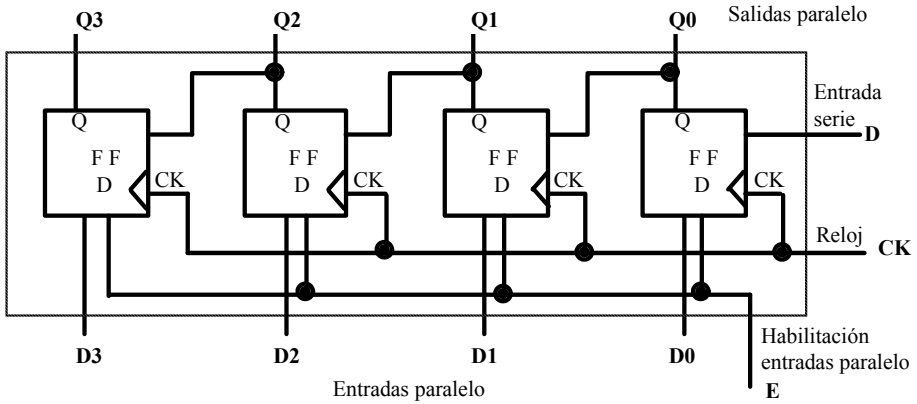


La conexión de **n** biestables síncronos tipo **D** en serie (la entrada de cada uno unida a la salida del anterior), con reloj común para todos ellos, configura un registro con una única entrada (la del primero) y **n** salidas en paralelo, en el que la información se desplaza de biestable a biestable «a golpes de reloj»: *registro de desplazamiento* (*shift-register*).



Este registro realiza una *conversión serie-paralelo* de la información que recibe en su entrada; con cada pulso de reloj el conjunto de los bits almacenados avanza un biestable y entra un nuevo bit en la cadena (operación de entrada en serie, presentación de valores booleanos en las salidas en paralelo).

Generalmente se añaden a tales registros de desplazamiento las correspondientes entradas paralelo junto con la entrada de habilitación de las mismas, es decir, una entrada **D** a cada uno de los biestables, controlada por una entrada de habilitación común **E**; activando la habilitación (**E=1**) los valores presentes en las entradas paralelo son almacenados en los biestables (*carga en paralelo*).



De esta forma puede recibirse información tanto en serie (entrada serie) como en paralelo (entradas paralelo) y transmitirse también en las dos formas: paralelo (en el conjunto de las salidas) y serie (en la salida del último biestable por desplazamiento).

Ello permite realizar la *conversión serie-paralelo* y la *conversión paralelo-serie* de una palabra binaria:

• *conversión serie-paralelo*: la palabra se recibe bit a bit por la entrada serie del registro de desplazamiento y, una vez recibida, sus dígitos están presentes en paralelo en las salidas de los biestables que configuran el registro;

• *conversión paralelo-serie*: la palabra se recibe a través de las entradas paralelo de los biestables del registro de desplazamiento y es enviada bit a bit por la salida del último de dichos biestables.

Otra aplicación de los registros de desplazamiento se basa en el hecho de que el desplazamiento de un número binario una posición hacía la izquierda equivale a multiplicar dicho número por 2, siendo sumamente útil para el diseño de algoritmos de producto aritmético; recíprocamente, el desplazamiento de una posición hacia la derecha equivale a dividir el número por 2 (registros de desplazamiento hacia la derecha). Para ello, se construyen registros de desplazamiento bidireccionales, con posibilidad de desplazarse en ambos sentidos, y, a su vez, con posibilidad de carga paralelo.

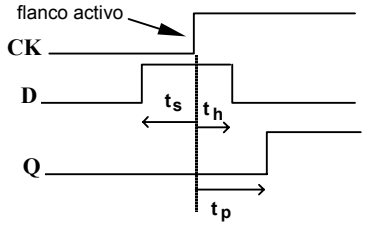
13.3. Tiempos funcionales e inicialización de los biestables

13.3.1. Tiempos de propagación, de anticipación y de mantenimiento de dato

El cambio de estado de un biestable síncrono se produce coincidiendo con el flanco activo de la onda de reloj; en el caso de un biestable tipo **D**, en ese momento (flanco activo del reloj), la salida efectúa una copia del valor presente en la entrada.

Obviamente existe un pequeño retraso entre el flanco activo de la señal de reloj y la consolidación del correspondiente estado en la salida: tiempo de propagación del dato $\mathbf{t_p}$. Además, para asegurar el correcto funcionamiento del biestable **D** cuando llega el flanco activo del reloj es necesario que el valor correcto del dato se encuentre presente en la entrada **D** con una cierta anticipación a dicho flanco (*setup:* $\mathbf{t_s}$) y que tal valor se mantenga durante un cierto intervalo posterior (*hold:* $\mathbf{t_h}$).

Supongamos, por ejemplo, que un biestable **D** que se encuentra con salida **0** debe recibir un **1**, dicho valor booleano **1** debe presentarse en la entrada del biestable, al menos, durante un intervalo $\mathbf{t_s}$ (*setup:* tiempo de anticipación) previo al flanco activo del reloj y debe permanecer dicho valor **1** en dicha entrada **D**, al menos, durante un intervalo $\mathbf{t_h}$ (*hold:* tiempo de mantenimiento) posterior al flanco activo.



Interesan, en general, tiempos de mantenimiento $\mathbf{t_h}$ muy pequeños o, al menos, que $\mathbf{t_h}$ sea menor que $\mathbf{t_p}$ (tiempo de propagación de los biestables «anteriores», es decir, de aquellos cuyas salidas influyen sobre la entrada del biestable), de manera que el propio retardo de propagación en los biestables sirva para cubrir el tiempo de mantenimiento del dato y evite errores en su captura por el biestable. En general, esto es lo que sucede al utilizar circuitos integrados estándar y, en tal caso, no es preciso prestar atención a los tiempos de mantenimiento (*hold*).

La suma de los tiempos de propagación y de preparación del dato $\mathbf{t_p+t_s}$ limita la velocidad máxima de trabajo de los biestables; el período de la onda de reloj deberá ser superior a dicha suma, para permitir que se forme el valor correcto en las salidas de los biestables (retraso $\mathbf{t_p}$) con suficiente anticipación (intervalo previo $\mathbf{t_s}$) para establecer el nuevo valor correcto en sus entradas. De forma que las «violaciones de *setup*» se resuelven disminuyendo la velocidad del reloj o, si es viable, utilizando biestables más rápidos (con menores tiempos de propagación o de anticipación).

El capítulo 15 desarrolla con mayor detalle el «análisis de tiempos» en los circuitos secuenciales síncronos, incorporando al mismo los tiempos de propagación debidos a la parte combinacional existente entre los biestables.

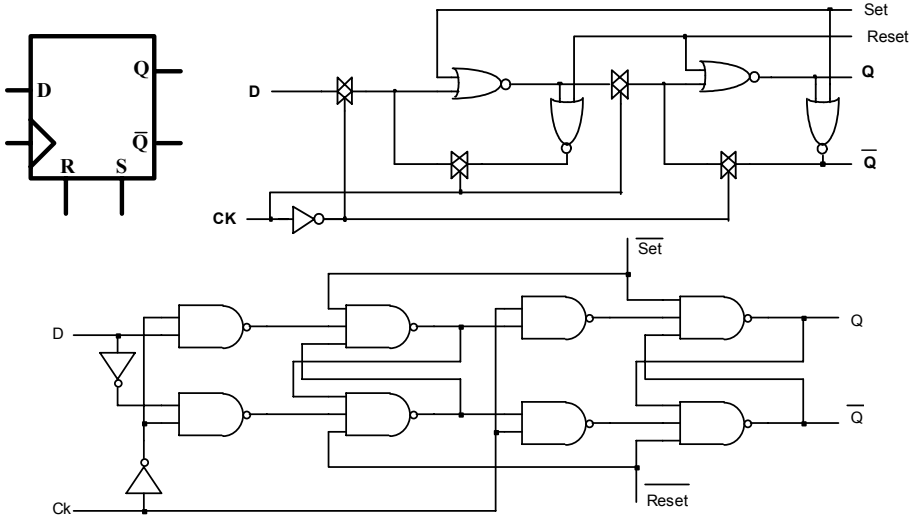
13.3.2. Inicialización de los biestables

Por lo general, un sistema secuencial, es decir, cualquier sistema configurado con biestables, requiere un estado inicial determinado, a partir del cual comienza la evolución del sistema: los biestables deben adoptar, en el primer momento, un estado booleano determinado, en muchos casos el estado 0 (**00…**).

Al activar la tensión de alimentación de un sistema (encendido: *power-on*), cada uno de los biestables presentes adoptará un estado booleano que, en principio, no es predecible, ya que depende de la configuración electrónica del biestable y del transitorio de encendido de la alimentación.

Para la inicialización de los biestables síncronos (fijación de valores iniciales en los mismos) y, también, para poder efectuar en cualquier momento un borrado o marcado de los mismos (con independencia del funcionamiento normal controlado por el reloj), interesa disponer de entradas asíncronas de marcado y borrado (*set* y *clear* o *reset*) que, al ser activadas, lleven directamente el biestable a estado **1** o estado **0**, respectivamente.

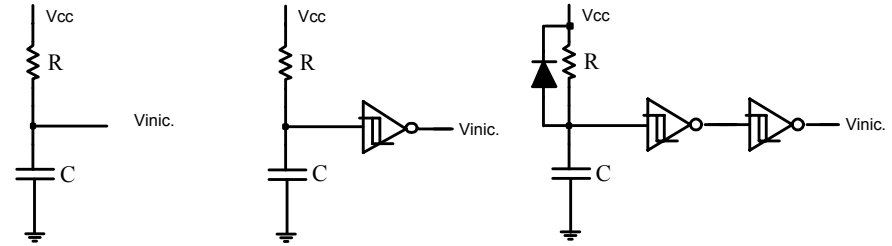
Es sencillo incluir entradas asíncronas de marcado y borrado en las configuraciones circuitales amo/esclavo representadas anteriormente:



Obsérvese (en las dos figuras) que es necesario actuar sobre ambos «semibiestables» (amo y esclavo) para asegurar que el borrado o el marcado permanecen hasta el siguiente flanco activo del reloj (si el borrado o marcado actuasen solamente sobre el esclavo y finalizasen cuando el reloj se encuentra con valor **1**, el esclavo recibiría inmediatamente el valor almacenado en el amo).

La inicialización de los biestables consiste en forzarles a adoptar el estado booleano que interese, mediante un primer pulso singular que se produzca tras el encendido (*power-on*) de la alimentación. La activación de las entradas de borrado o de marcado asíncrono (*reset* y *set*) por dicho primer pulso llevará al biestable a estado **0** ó **1** según proceda.

El pulso de inicialización puede generarse «automáticamente» mediante un circuito RC conectado a la tensión de alimentación; el retraso originado por la carga del condensador, a partir del propio «arranque» de la alimentación, determina la permanencia del valor **0** booleano durante un cierto intervalo posterior al encendido.



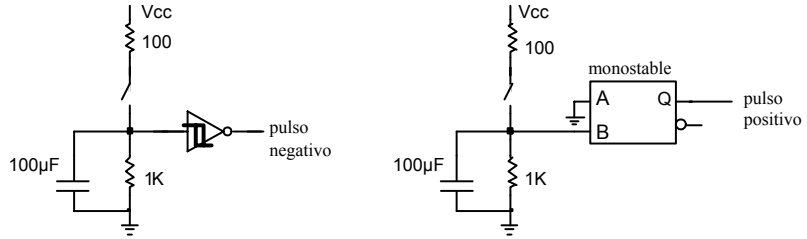
En el caso de que las entradas de borrado o marcado asíncrono se activen con valor **0**, bastará conectar la salida del primer circuito RC a la entrada que corresponda. Cuando su valor activo sea **1** será preciso invertir el pulso, mediante una puerta inversora, preferiblemente de entrada con histéresis (tipo *Schmitt*) para mejorar la definición del propio pulso y evitar los rebotes. Incluso, en activación con valor **0**, si las entradas de borrado o marcado de los biestables no son de tipo *Schmitt* es conveniente conformar el pulso a través de dos inversores cuyas entradas sí lo sean.

La anchura del pulso de inicialización en los circuitos anteriores será del orden de la constante de tiempo del circuito RC. Los valores del condensador y de la resistencia deben ser adecuadamente altos para asegurar una amplia anchura de pulso que lo prolongue más allá del transitorio de encendido (RC > 100 ms). En su caso, la inclusión de un monostable disparado por el circuito RC permite fijar con precisión la anchura del pulso de inicialización.

Suele incluirse un diodo en paralelo con la resistencia R, en polarización inversa, para descargar rápidamente el condensador C cuando hay caídas de tensión y asegurar que se produzca el pulso de inicialización en caso de «microcortes» (caídas de tensión de pequeña duración) de la alimentación.

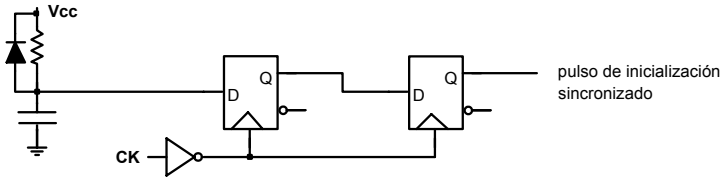
Existen circuitos integrados específicos que supervisan la tensión de alimentación y proporcionan un pulso de anchura fija en el encendido y, asimismo, activan la inicialización cada vez que la tensión de alimentación cae por debajo de un valor prefijado, prolongando después el pulso de inicialización al restablecerse la alimentación.

Caso de que interese una inicialización manual, será necesario conformar el pulso producido sobre el correspondiente pulsador mediante un circuito RC de constante de tiempo relativamente alta (a fin de evitar los rebotes del pulsador), seguido de una puerta con entrada *Schmitt*. La utilización de un monostable permite ajustar a un valor fijo la anchura de este pulso manual de inicialización.



Puede resultar conveniente sincronizar los pulsos de inicialización de los biestables con el reloj del sistema secuencial para evitar posibles coincidencias entre el final de la inicialización y el flanco activo del reloj; tales coincidencias podrían generar errores por cuanto que algunos biestables podrían responder al flanco de reloj y otros no.

Para sincronizar la inicialización basta pasarla a través de un biestable síncrono y, en tal caso, interesa conectar su entrada de reloj al flanco «no activo» del reloj para que la inicialización cubra por completo al anterior flanco activo del mismo; a fin de reducir la incidencia de situaciones «metastables» suelen utilizarse dos biestables sucesivos.

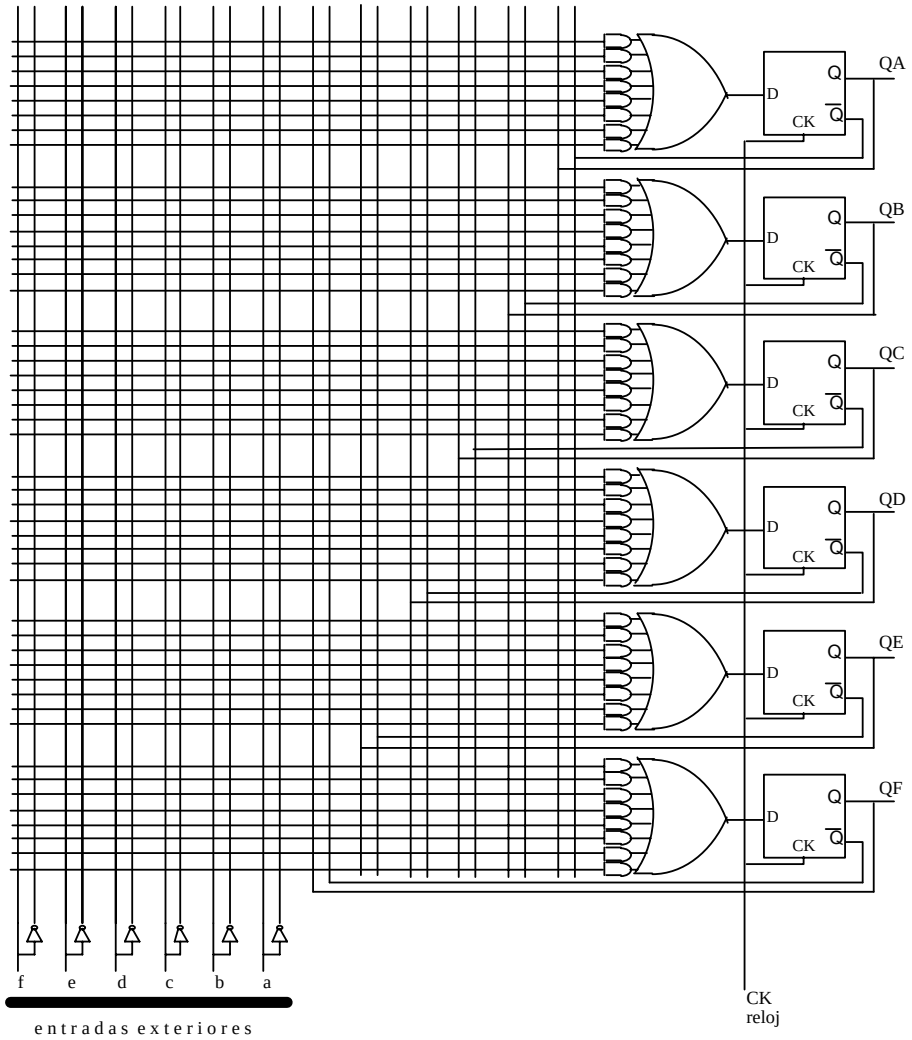


13.4. Dispositivos lógicos programables: PAL con biestables

Existen dispositivos programables basados en la configuración **PAL**, que añaden un biestable en cada una de las salidas; de esta forma, se proporciona a la estructura **PAL** capacidad de memoria y, por tanto, de realización de circuitos secuenciales. En este caso, los términos producto de la **PAL** han de admitir como variables, además de las propias variables de entrada, las salidas de los biestables; es decir, las variables de estado (almacenadas en los biestables) han de realimentarse como entradas sobre los módulos **PAL** que configuran las funciones de evolución del estado.

De esta forma tendremos un bloque con **n** módulos, formados, cada uno de ellos, por un biestable cuya entrada **D** es activada por una puerta "**o**" precedida por **q** puertas "**y**", cuyas entradas configuran una Matriz Y programable respecto a las **m** variables de entrada exteriores y sus negadas y a las **n** salidas de los biestables del bloque y sus negadas.

Este tipo de bloque programable recibe el nombre de *secuenciador lógico programable* **PLS** (*programmable logic sequencer*) o el de ***PAL con biestables*** **RPAL** (*registered* **PAL**).



PLS de 6 módulos con 8 términos producto en cada módulo y 6 variables de entrada

Los bloques **PLS** permiten la programación de las funciones de evolución de estado de un circuito secuencial; para ello habrá que obtener tales funciones de evolución del estado correspondientes a biestables tipo **D**, expresadas en forma de suma de productos.

Asimismo se pueden programar en el mismo bloque **PLS** las funciones de activación de las salidas. En este caso, las salidas pasarán a través de biestables que realizan un sincronismo de las mismas, lo cual suele ser beneficioso en la mayoría de los casos: se añade al sincronismo de las variables de estado el sincronismo de las salidas (que aplica a las mismas un retraso de una unidad de tiempo de reloj).

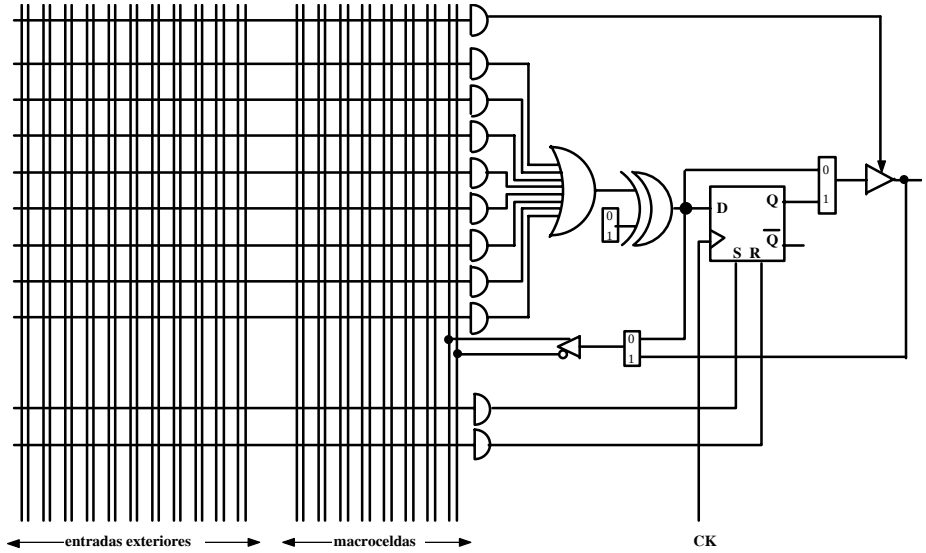
Hay también bloques **PLS** en que el biestable de salida de cada módulo puede utilizarse o no (puede *puentearse*) mediante programación, de forma que permiten la inclusión tanto de variables sincronizadas, como de funciones combinacionales; de esta manera pueden construirse las funciones de activación de las salidas sin añadirles el retraso correspondiente al sincronismo.

Así pues, en el interior de un bloque **PLS** puede configurarse un circuito secuencial completo, mediante programación; ésta es, hoy día, la forma habitual de diseño digital: un solo circuito integrado particularizado que contiene todo el sistema digital (sin necesidad de la conexión de múltiples circuitos integrados estándar).

Los circuitos programables han experimentado un extraordinario desarrollo en la última década (años 90), con la integración de un alto número de módulos en el mismo circuito y el aumento de prestaciones de tales módulos. El módulo básico **PLS** «suma de productos + biestable» ha incorporado diversos «selectores» programables, así como la posibilidad de salida «tri-estado» y recibe el nombre genérico de *macrocelda* (cuya configuración es la representada en la figura de la página siguiente).

Una macrocelda está compuesta por

- una suma de productos programables de múltiples entradas entre las que se encuentran las salidas de todas las macroceldas (realimentación), estando disponible cada entrada en su forma afirmada y negada;
- una puerta "**o-exclusiva**" programable que permite configurar la función de la macrocelda (suma de términos producto) en forma afirmada (y) o negada ($\overline{y}$), pudiendo optar por la que menor número de términos producto requiera (lo cual equivale a poder elegir entre resolver la función por «unos» o por «ceros»);
- un biestable que recibe dicha función;
- un selector que permite «puentear» el biestable y que configura, por tanto, la macrocelda como combinaciónal o como secuencial;
- y un adaptador triestado con capacidad para «desconectar» la macrocelda respecto del terminal de salida, lo cual da lugar a las siguientes posibilidades:
    - la simple desconexión (alta impedancia) del terminal
    - su utilización bidireccional (I/O)
    - su disponibilidad como entrada, caso de que la macrocelda no sea utilizada.



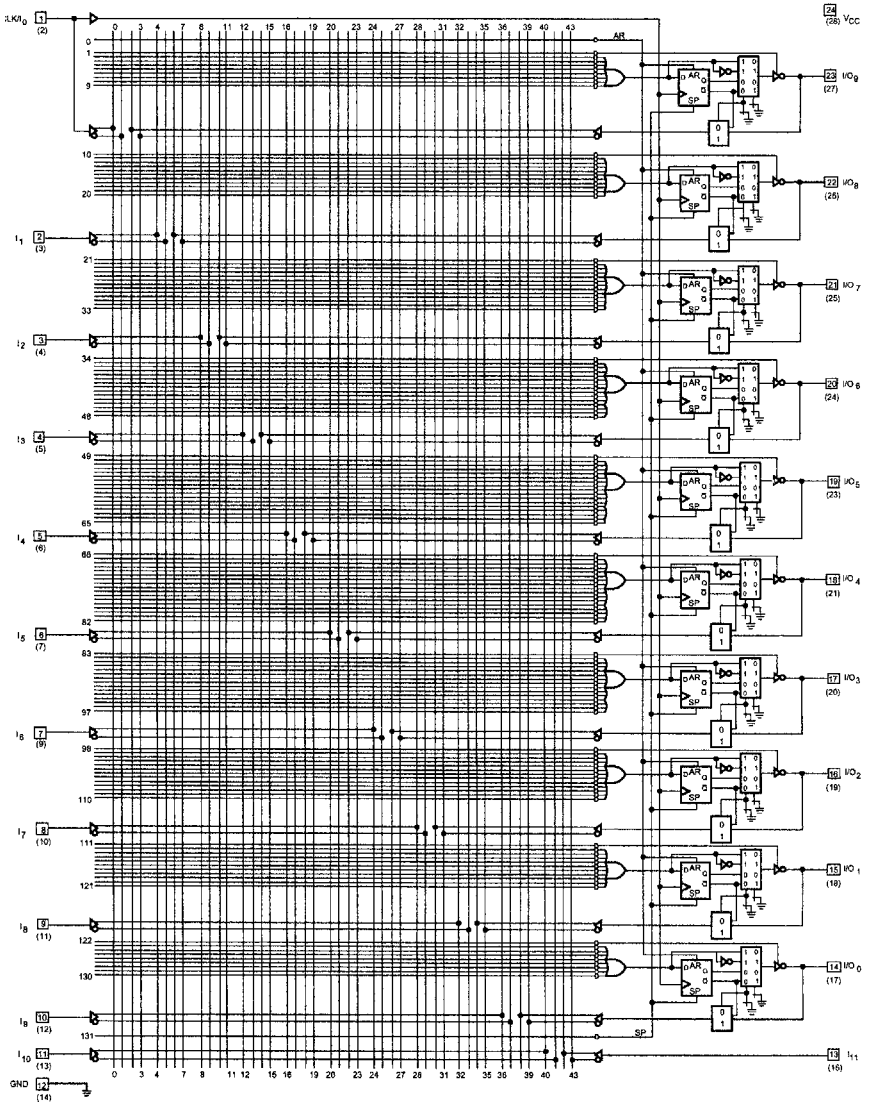
Configuración booleana de una macrocelda

Los circuitos integrados programables que utilizan *macroceldas* suelen ser aludidos con las siglas **PLD** (dispositivos lógicos programables).

Mientras el número de macroceldas contenido en un **PLD** no es alto, cada una de ellas recibe todas las entradas del dispositivo y la realimentación de todas las salidas del mismo, que coinciden directamente con las salidas de todas las macroceldas del dispositivo: en caso de que alguna de las macroceldas no sea utilizada, la salida correspondiente puede ser usada como entrada.

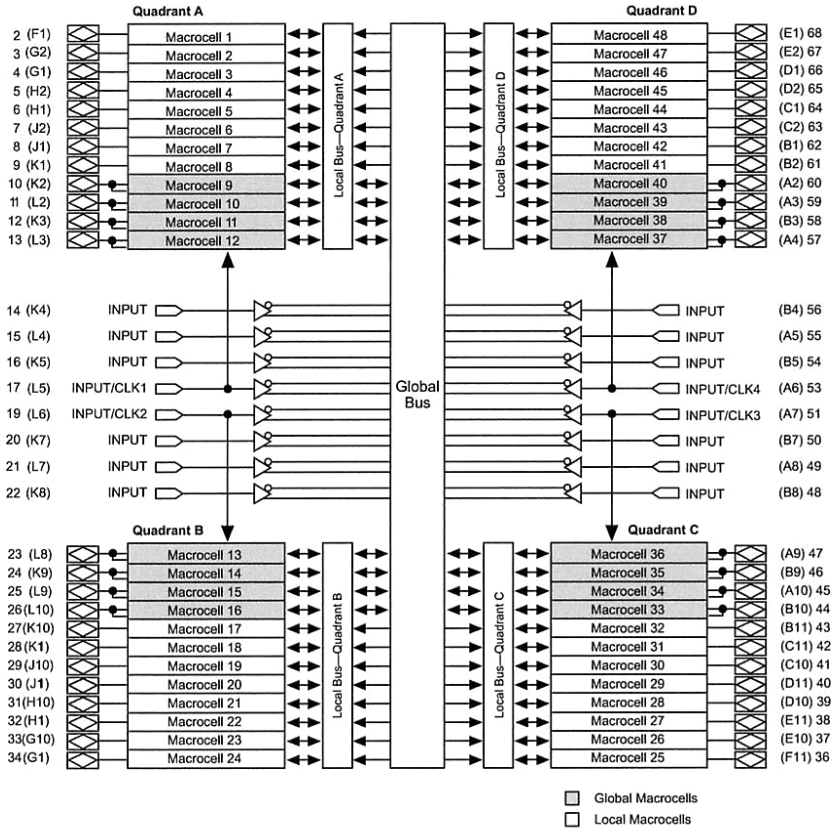
El más conocido de estos dispositivos es el 22V10 que contiene 10 macroceldas con diferente número de términos producto (hay 2 macroceldas con 8 términos producto, otras 2 con 10, 2 con 12, 2 con 14 y 2 con 16 términos producto). [Véase la figura de la página siguiente.]

El circuito integrado programable 22V10 ofrece 22 terminales: 12 de entrada (uno de ellos para el reloj) y 10 de tipo I/O (correspondientes a las 10 salidas de las macroceldas); cuando una macrocelda no es utilizada su correspondiente terminal de salida puede ser aprovechado como una entrada exterior más para el resto de las macroceldas.



Configuración del circuito integrado programable PLD 22V10

La figura siguiente representa un **PLD** aún mayor: el circuito integrado programable EP1800, que contiene 48 macroceldas con 16 entradas exteriores (encapsulado de 68 *pines*) y posibilidad de aprovechar también como entradas las salidas de las macroceldas que no se utilicen como tales.



Configuración del circuito integrado programable EP1800

Como el número de macroceldas de este circuito integrado ya es considerable, se dividen en cuatro cuadrantes; dentro de cada cuadrante hay cuatro macroceldas «globales» que se realimentan con todas las del integrado y ocho macroceldas «locales» que solamente se realimentan con las del propio cuadrante y con las globales.

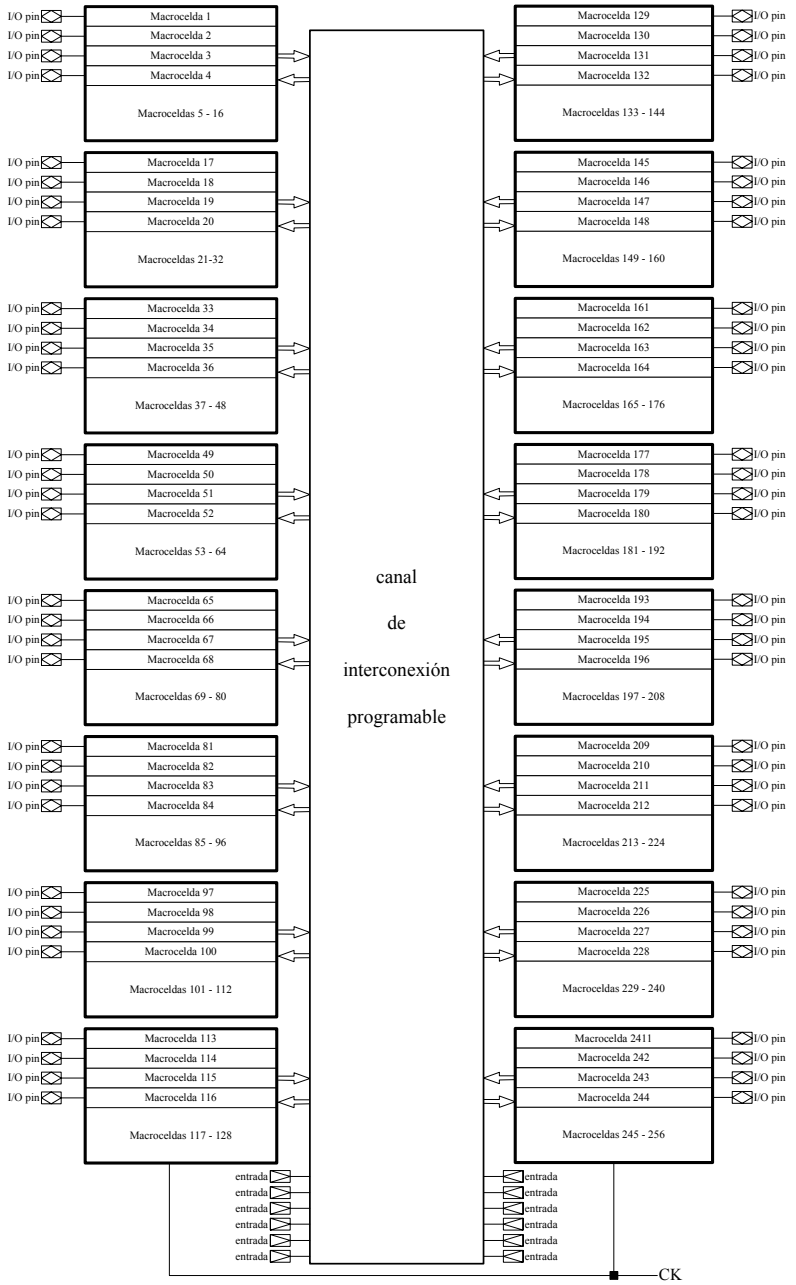
Cuando se trata de un gran número de macroceldas dentro de un mismo integrado no resulta adecuado que cada una de ellas reciba todas las entradas y se conecte con todas las macroceldas ya que la cantidad de conexiones programables sería sumamente elevada; tampoco es viable que las salidas de todas las macroceldas se conecten a terminales de salida del circuito integrado.

Para limitar el tamaño y el número de terminales del circuito, las macroceldas se agrupan en bloques disjuntos y reciben solamente las entradas propias del bloque y la realimentación de las macroceldas del mismo; no todas las salidas de las macroceldas están conectadas a terminales de salida sino solamente un número reducido de cada bloque. Los bloques se conectan entre sí a través de un bus central de conexiones programables conformando una estructura ramificada: un canal o eje de conexiones (*a manera tronco*) sobre el cual se insertan los bloques de macroceldas (*como ramas del mismo*). [Véase la figura de la página siguiente.]

Este tipo de circuitos integrados programables de amplio número de macroceldas con arquitectura ramificada es nombrado con las siglas **CPLD** (dispositivos lógicos programables complejos).

Para reducir la superficie de integración y evitar un alto desaprovechamiento de las estructuras PAL, el número de términos producto de cada macrocelda suele ser pequeño (4 ó 6) y para ampliarlo, cuando se necesite, se incluyen en cada bloque «expansores», en forma de términos producto adicionales que, sin pertenecer directamente a ninguna de las macroceldas del bloque, pueden ser utilizados por cualquiera de ellas.

La figura de la página siguiente representa un CPLD de 256 macroceldas, de las cuales solamente 64 pueden utilizarse como salidas exteriores (las otras 192 macroceldas son internas); el circuito dispone de una entrada específica de reloj y de 12 entradas exteriores, pudiéndose emplear también como entradas los terminales correspondientes a macroceldas que no se usen como tales.



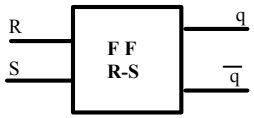
Configuración CPLD con 256 macroceldas

13.5. Los biestables en VHDL

Al igual que en los capítulos 1 y 4, se incluye aquí la descripción VHDL de diversos tipos de biestables a fin de desarrollar una aproximación gradual al diseño con lenguajes de descripción circuital:

Biestables asíncronos

biestable RS



a) q <= '0' when R = '1' else '1' when S = '1' else q;

b) y <= not R and (S or q);

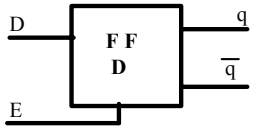
ambos casos con borrado prioritario

descripción utilizando proceso

```
process (R,S)
begin
if      R = '1'    then   q  <= '0';
elsif   S = '1'    then   q  <= '1';
end if;
end process;
```

(téngase en cuenta que *proceso* conserva los valores: por ello no es necesario añadir *else q <= q;*)

biestable D



a) q <= D when E = '1' else q;

b) y <= (D and E) or (q and not E)

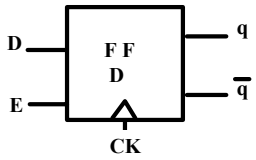
descripción utilizando proceso

```
process (D,E)
begin
if      E = '1'    then   q  <= D;
end if;
end process;
```

(no es necesario añadir *else q <= q;*)

Biestables síncronos

biestable D

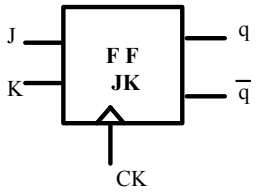



```
process
begin
wait on CK until CK = '1';
if E = '1'  then   q  <= D;  end if;
end process;
```

(el reloj se describe siempre dentro de un proceso; si afecta a todo el proceso se describe con un *wait* y el *until CK = '1'* indica flanco ascendente)

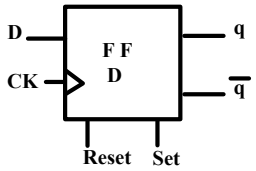
(no es necesario añadir *else q <= q;*)

biestable JK




```
process
begin
wait on CK until CK = '1';
if J = '1' and K ='1'   then     q <= not q;
elsif J='1'             then     q <= '1';
elsif K = '1'           then     q <= '0';
end if;
end process;
```

biestable D
con marcado/borrado
asíncronos




```
process(Reset,Set,CK,D)
begin
if Reset = '1'     then   q <= '0';
elsif Set = '1'    then   q <= '1';
elsif CK'event and CK = '1' then
                          q <= D;
end if;
end process;
```

(la parte asíncrona debe describirse antes del reloj; luego el reloj se describe dentro de un *elsif* con *event* y *CK = '1'* indica flanco ascendente)