TEMA V: SÍNTESIS DE SISTEMAS SECUENCIALES

Una vez que hemos visto como se puede resolver el problema de análisis de sistemas secuenciales, vamos a abordar el problema complementario, es decir, el diseño de sistemas secuenciales. Si recordamos, cualquier problema de diseño se podía definir de la siguiente forma:

> Dados un comportamiento y una funcionalidad, **el diseño de un circuito** consiste en determinar un circuito que cumpla dichas condiciones.

1. Introducción.

Para determinar un circuito secuencial que cumpla una determinada descripción funcional, existen una serie de pasos que debemos seguir. Estos pasos se suelen denominar flujo de diseño, y se muestran en la figura 5.1.

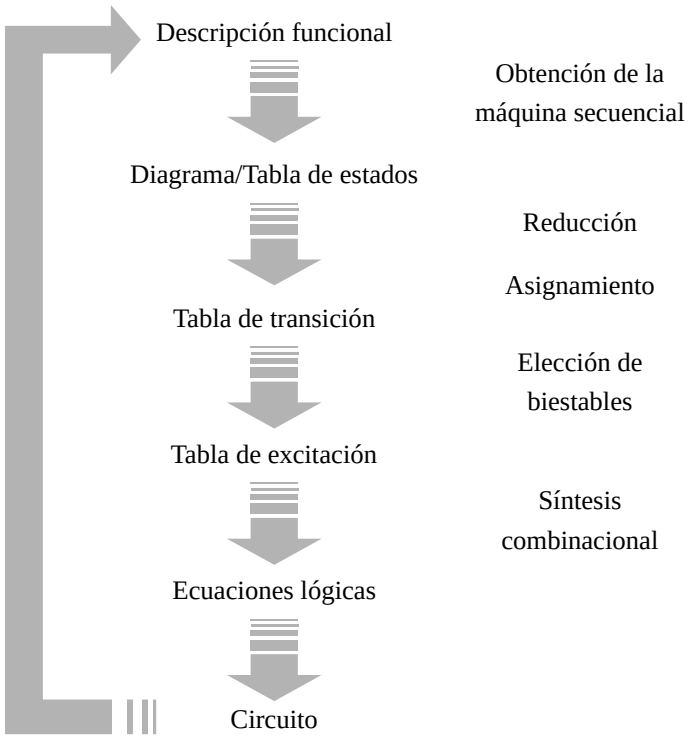


Figura 5.1.- Flujo de diseño típico de un sistema secuencial.

El punto de partida del problema de diseño consiste en la descripción del funcionamiento y comportamiento que debe mostrar nuestro sistema, es decir, las especificaciones de diseño. Esta descripción, generalmente verbal, debe ser traducida a una máquina secuencial en forma de diagrama y/o tabla de estados.

El objetivo final del diseño consiste en obtener un circuito con coste mínimo, el cual está íntimamente relacionado con el número de estados. Por lo tanto, el diagrama y/o tabla de estados debe ser reducido para seguir con la garantía de obtener un diseño mínimo. Después de tener el diagrama y/o tabla mínima, debemos afrontar el problema de asignamiento de estados, o lo que es lo mismo, asignar a cada estado una determinada combinación binaria.

Sustituyendo el alfabeto de estados por su codificación, la tabla de estados pasará a denominarse **tabla de transición**. Esta tabla solamente contiene datos binarios y es la situación más adecuada para elegir los elementos de memorias que estarán presentes en el circuito final. El tipo de biestable elegido será aquel que necesite menos lógica combinacional para generar las entradas de los biestables.

Una vez elegido el tipo de biestable, debemos cambiar las variables de próximo estado por las entradas de dichos biestables. Con este paso, la tabla de transición pasa a denominarse **tabla de excitación**. En este momento hemos separado la lógica combinacional de los biestables. Por lo tanto, y llegado a este punto, el problema pasa a ser el diseño de un circuito combinacional que deberá generar las señales de entrada a los biestables y las señales de salida, tomando las señales de estado presente como entradas independientes.

Por último, y con el circuito ya diseñado, se debe realizar un proceso de análisis para garantizar que el circuito diseñado cumple todas las especificaciones de comportamiento y funcionalidad de las que se partió.

2. Obtención de la máquina secuencial.

El resultado de este paso consiste en obtener una descripción de la máquina secuencial en modo de diagrama y/o tabla de estados, que muestre el mismo comportamiento que se desea implementar.

Debido a la existencia de multitud de máquinas equivalentes, no existe ningún procedimiento sistemático que se pueda seguir para realizar esta traducción. Luego la pericia del diseñador será un factor con muy alta influencia en la calidad de la traducción obtenida. No obstante, existen una serie de guías que se pueden seguir. Entre estas guías podemos encontrar las siguientes:

- Pasar de una descripción verbal a una secuencia de entrada/salida.
- Si se conoce alguna máquina válida para nuestro sistema, se suele tomar como máquina de partida.
- Siempre se comienza por un estado conocido; en el caso de que exista un estado inicial, se empezará por él.
- Para cada estado se asigna una transición por cada combinación de entradas, indicando el próximo estado y el valor de las salidas.

- No hay que temer introducir nuevos estados en caso de duda, ya que durante el proceso de reducción se eliminarán todos los estados sobrantes.
- Una vez obtenido el diagrama, se aplica la secuencia de entradas para comprobar que la secuencia de salida es correcta.

Por ejemplo, la descripción verbal de la máquina de refrescos podría ser la siguiente:

Se desea diseñar un circuito secuencial que controle el funcionamiento de una máquina de refrescos con las siguientes especificaciones:

- *Disponga de un solo producto con precio único e igual a 60 c.*
- *Sólo se permitan monedas de 10, 20 y 50 c.*
- *El refresco se obtendrá al pulsar el botón B con un importe superior e igual a 60 c.*
- *No se devuelve cambio.*

En primer lugar debemos identificar las entradas y las salidas. La salida será la expulsión del refresco; mientras que las entradas serán la inserción de las diferentes monedas y pulsar el botón de obtención. Así, conociendo las entradas y salidas, las diferentes secuencias de entrada/salida podría ser la siguiente:

- Si no hay dinero y echamos 10 c., se almacenan 10 c.
- Si no hay dinero y echamos 20 c., se almacenan 20 c.
- Si no hay dinero y echamos 50 c., se almacena 50 c.
- Si hay 10 c. y echamos 10 c., se almacenan 20 c.
- Si hay 10 c. y echamos 20 c., se almacenan 30 c.
- Si hay 10 c. y echamos 50 c., se almacena 60 c.
- Si hay 20 c. y echamos 10 c., se almacenan 30 c.
- Si hay 20 c. y echamos 20 c., se almacena 40 c.
- Si hay 20 c. y echamos 50 c., se almacena 60 c.
- Si hay 30 c. y echamos 10 c., se almacena 40 c.
- Si hay 30 c. y echamos 20 c., se almacena 50 c.
- Si hay 30 c. y echamos 50 c., se almacena 60 c.
- Si hay 40 c. y echamos 10 c., se almacena 50 c.
- Si hay 40 c. y echamos 20 ó 50 c., se almacena 60 c.
- Si hay 50 c. y echamos 10 c., 20 c. ó 50 c., se almacena 60 c.
- Si hay 60 c. y echamos más monedas, se almacena 60 c.
- Si pulsamos el botón B sin que haya 60 c., no pasa nada.
- Si pulsamos el botón B y hay almacenado 60 c., obtenemos el refresco.

Como no conocemos ninguna máquina válida que se ajuste a estas especificaciones, pasaremos al tercer punto. Para lo cual partiremos del estado inicial en el que no hay dinero almacenado.

Como podemos tener dudas si el estado inicial es diferente al de obtener un refresco, se sitúan en dos estados diferentes. Por lo tanto, un posible diagrama de dicha máquina será el mostrado en la figura 5.2.

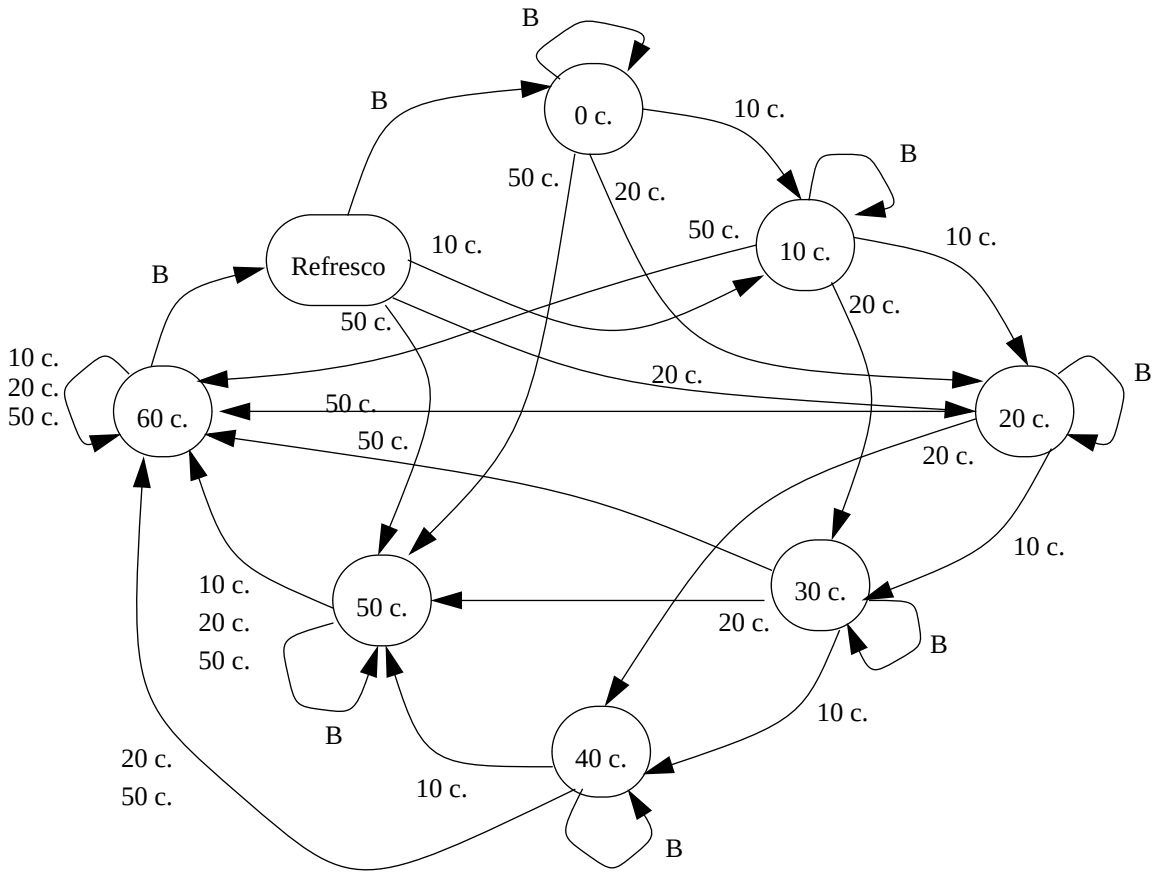


Figura 5.2.- Posible diagrama de estados de la máquina de refrescos.

3. Reducción y asignamiento de estados.

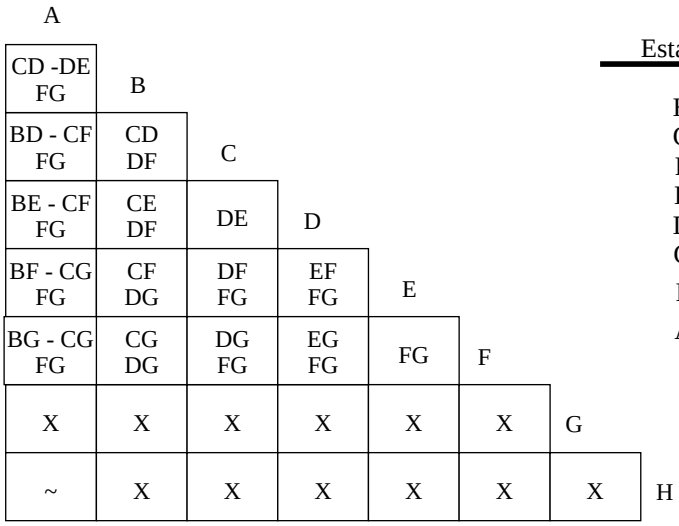
El problema de reducción del número de estados ya se abordó en un tema anterior, por lo que no lo volveremos a repetir, y nos centraremos en el ejemplo que nos ocupa.

En primer lugar tenemos que identificar las señales de entrada, para lo cual sabemos que tenemos cuatro combinaciones de entrada: pulsar el botón, echar 10c., echar 20c. y echar 50c. No obstante debemos considerar como distinguimos los intervalos temporales entre la inserción de dos monedas, es decir, cuando el sistema debe hacer algo. Podemos adoptar dos estrategias diferentes: identificar cuando se produce alguna de las condiciones anteriores, y entonces ejecutar las operaciones; o añadir una combinación de entrada más, que sería no echar ninguna moneda. En el resto del desarrollo consideraremos la segunda estrategia, por lo que tendremos tres señales de entrada: el botón B y las señales I_2 e I_1 con la codificación: “00” = 0 c., “01” = 10 c., “10” = 20 c. y “11” = 60 c. La tabla de estados será la mostrada en la tabla 5.1.

	000	001	010	011	100	101	110	111
Nada (estado A)	A, 0 *	B, 0	C, 0	F, 0	A, 0 *	A, 0 *	A, 0 *	A, 0 *
10 c. (estado B)	B, 0*	C, 0	D, 0	G, 0	B, 0 *	B, 0 *	B, 0*	B, 0 *
20 c. (estado C)	C, 0 *	D, 0	F, 0	G, 0	C, 0 *	C, 0 *	C, 0 *	C, 0 *
30 c. (estado D)	D, 0 *	E, 0	F, 0	G, 0	D, 0 *	D, 0 *	D, 0 *	D, 0 *
40 c. (estado E)	E, 0 *	F, 0	G, 0	G, 0	E, 0 *	E, 0 *	E, 0 *	E, 0 *
50 c. (estado F)	F, 0 *	G, 0	G, 0	G, 0	F, 0 *	F, 0 *	F, 0 *	F, 0 *
60 c. (estado G)	G, 0 *	G, 0 *	G, 0 *	G, 0 *	H, 1	H, 1	H, 1	H, 1
Refresco (estado H)	H, 0 *	B, 0	C, 0	F, 0	A, 0	A, 0	A, 0	A, 0

Tabla 5.1. Tabla de estados correspondiente al diagrama de la máquina de refrescos.

Estamos ante una máquina completamente especificada, por lo que la reducción se limita a hallar los máximos compatibles. Esta reducción se muestra en la figura 5.3, de donde obtenemos que todos los estados son incompatibles excepto los estados A y H. Luego, la tabla mínima tendrá cinco estados, siendo la mostrada en la tabla 5.2.



Estados	CEC	MC
H	--	H
G	--	H, G
F	--	H, G, F
E	--	H, G, F, E
D	--	H, G, F, E, D
C	--	H, G, F, E, D, C
B	--	H, G, F, E, D, C, B
A	H	AH, G, F, E, D, C, B

Figura 5.3.- Reducción de estados correspondiente a la máquina de la figura 5.2.

	000	001	010	011	100	101	110	111
Refresco (estado AH)	AH, 0 *	B, 0	C, 0	F, 0	AH, 0 *	AH, 0 *	AH, 0 *	AH, 0 *
10 c. (estado B)	B, 0*	C, 0	D, 0	G, 0	B, 0 *	B, 0 *	B, 0*	B, 0 *
20 c. (estado C)	C, 0 *	D, 0	F, 0	G, 0	C, 0 *	C, 0 *	C, 0 *	C, 0 *
30 c. (estado D)	D, 0 *	E, 0	F, 0	G, 0	D, 0 *	D, 0 *	D, 0 *	D, 0 *
40 c. (estado E)	E, 0 *	F, 0	G, 0	G, 0	E, 0 *	E, 0 *	E, 0 *	E, 0 *
50 c. (estado F)	F, 0 *	G, 0	G, 0	G, 0	F, 0 *	F, 0 *	F, 0 *	F, 0 *
60 c. (estado G)	G, 0 *	G, 0 *	G, 0 *	G, 0 *	AH, 1	AH, 1	AH, 1	AH, 1

Tabla 5.2. Tabla mínima de estados correspondiente al diagrama de la máquina de refrescos.

Una vez que hemos realizado la reducción de estados, y por lo tanto tenemos una tabla mínima, debemos dar a cada estado una codificación binaria.

El problema de codificar los estados según el sistema binario se conoce como **asignamiento de estados**.

En función de este asignamiento, el circuito obtenido puede tener mayor o menor coste. No obstante, este problema no tiene una solución sistemática que garantice que el resultado final tendrá el mínimo coste, así como la eliminación de los problemas de temporización vistos en el tema anterior. Las soluciones que existen se pueden clasificar en tres grandes grupos:

- Soluciones exhaustivas
- Soluciones algebraicas
- Soluciones heurísticas

Las soluciones exhaustivas se basan en considerar todas las posibles codificaciones para todos los estados. Se obtienen todos los circuitos asociados a cada asignamiento, y nos quedamos con aquel que presente un coste menor. El problema de esta solución radica en que, a medida que aumenta el número de estados, aumenta el número de codificaciones diferentes, y por lo tanto, el número de casos a considerar. Se han obtenido que estas soluciones son inviables cuando el número de estados supera la cifra de 16.

Las soluciones algebraicas se basan en la teoría de particiones, de tal forma que el diagrama de estados se divide en partes aplicando soluciones exhaustivas a cada una de las partes. No obstante, al igual que en el caso anterior, el incremento del número de estados es un factor limitante para poder utilizar esta solución. Se ha obtenido que esta solución es inviable cuando el número de estado supera la cifra de 32. Además del problema anterior nos encontramos ante el problema de la elección de las particiones, las cuales son difíciles de encontrar.

Ante la situación de no tener un método sistemático para todos los problemas de asignamiento, los métodos más utilizados son los métodos heurísticos. Estos métodos se basan en la inspección de la tabla de estados y el seguimiento de una serie de directrices obtenidas por la experiencia. Entre estas directrices podemos encontrar las siguientes:

- Se asignan códigos adyacentes a estados presentes y próximos estados.
- Se asignan códigos adyacentes a estados que tengan el mismo próximo estado para una misma entrada.
- Se asignan códigos adyacentes a estados que son próximos estados del mismo estado presente.
- Se asignan códigos adyacentes a estados que tengan el mismo valor de salida para una misma entrada.

donde los códigos adyacentes son los que tienen una distancia de Hamming igual a 1. Como es lógico todas estas directrices no se pueden seguir, debiendo seguirse la mayoría de ellas. Con la primera directriz eliminaremos en la mayor medida de lo posible la existencia de carreras, debiendo considerar en primer lugar las carreras críticas.

Siguiendo con el ejemplo anterior, podemos decir que:

- Siguiendo la primera directriz:
  - El estado AH debe ser adyacente a los estados B, C y F.
  - El estado B debe ser adyacente a los estados C, D y G.

  - El estado C debe ser adyacente a los estados D, E y G.
  - El estado D debe ser adyacente a los estados E, F y G.
  - El estado E debe ser adyacente a los estados F y G.
  - Los estados F y G deben ser adyacentes.
  - Los estados G y AH deben ser adyacentes.
- Siguiendo la segunda directriz.
  - Los estados B, C, D, E, F y G deben ser adyacentes.
- Siguiendo la tercera directriz.
  - Los estados B, C y F deben ser adyacentes.
  - Los estados C, D y G deben ser adyacentes.
  - Los estados D, E y G deben ser adyacentes.
  - Los estados E, F y G deben ser adyacentes.
  - Los estados F y G deben ser adyacentes.
- Siguiendo la cuarta directriz.
  - Los estados AH, B, C, D, E y F deben ser adyacentes.

Como tenemos un total de siete estados necesitaremos un mínimo de tres señales de estado: Q_2, Q_1 y Q_0, para completar las combinaciones necesarias. Un posible asignamiento puede ser el siguiente:

- AH = “011”
- B = “001”
- C = “010”
- D = “101”
- E = “100”
- F = “111”
- G = “000”

En este caso no es posible mantener todas las condiciones de adyacencia, por lo que sólo se deben permitir la presencia de carreras no críticas. Luego la tabla de transición con la codificación de estados anterior se muestra en la tabla 5.3.

	000	001	010	011	100	101	110	111
000	000, 0 *	000, 0 *	000, 0 *	000, 0 *	011, 1	011, 1	011, 1	011, 1
001	001, 0 *	010, 0	101, 0	000, 0	001, 0 *	001, 0 *	001, 0 *	001, 0 *
010	010, 0 *	101, 0	111, 0	000, 0	010, 0 *	010, 0 *	010, 0 *	010, 0 *
011	011, 0 *	001, 0	010, 0	111, 0	011, 0 *	011, 0 *	011, 0 *	011, 0 *
100	100, 0 *	111, 0	000, 0	000, 0	100, 0 *	100, 0 *	100, 0 *	100, 0 *
101	101, 0 *	100, 0	111, 0	000, 0	101, 0 *	101, 0 *	101, 0 *	101, 0 *
111	111, 0 *	000, 0	000, 0	000, 0	111, 0 *	111, 0 *	111, 0 *	111, 0 *

Tabla 5.3. Tabla de transición correspondiente al diagrama de la máquina de refrescos.

4. Elección de los biestables.

Una vez que hemos obtenido la tabla de transición, tenemos que cambiar las funciones de próximo estado por las funciones de las señales de entrada de los biestables. En esta elección hay que considerar una serie de aspectos, los cuales van a determinar tanto el tipo de biestable como la temporización utilizada. Una vez que hayamos realizado esta sustitución en la tabla de transición, ésta recibirá el nombre de **tabla de excitación**.

4.1. Aspectos lógicos.

Este aspecto va a ayudarnos a elegir el tipo lógico de biestable, es decir, si serán biestables tipo D, T, RS o JK. En función de la transición de señales de estado que se deba realizar, las señales de entrada de los biestables deben tener un valor determinado. Estos valores se muestran en la tabla 5.4.

Transición (q -> Q)	D	T	R	S	J	K
0 -> 0	0	0	0	--	0	--
0 -> 1	1	1	1	0	1	--
1 -> 0	0	1	0	1	--	1
1 -> 1	1	0	--	0	--	0

Tabla 5.4. Valores de las señales de entradas de los biestables para cada transición de salida.

Veamos un ejemplo de cómo se ha obtenido la tabla anterior. En la FIGURA mostramos la tabla de función de un biestable JK. En ella observamos la transición de ‘0’ (valor del estado presente q) a ‘1’ (valor del próximo estado Q). Podemos observar que podemos encontrarnos en dos casos diferentes, para los cuales las señales JK deben tomar el valor “11” o “10”. Por lo tanto, la única condición que es necesaria cumplir para realizar esta transición será que la señal J tome el valor ‘1’, independientemente del valor de K.

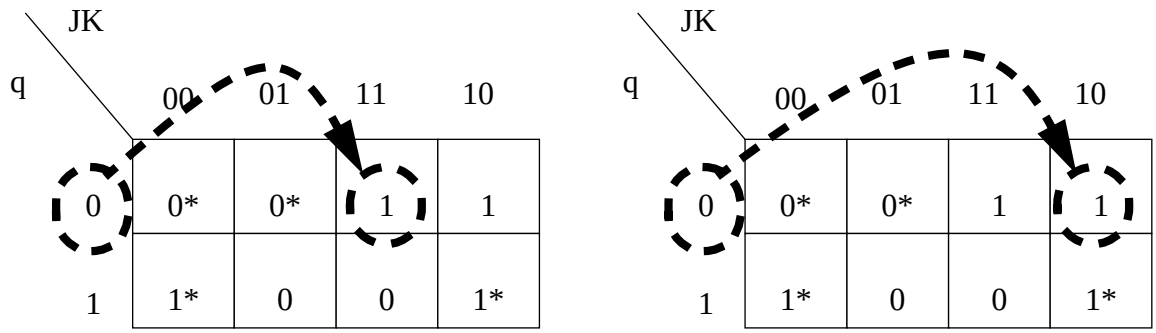


Figura 5.4.- Transición de ‘0’ a ‘1’ en un biestable JK.

Según esta tabla, podemos extraer las siguientes directrices:

- Si se utilizan biestables tipo D, la tabla de excitación coincidirá con la tabla de transición.
- El uso de biestables tipo D muestra un coste menor cuando el estado presente es muy parecido al próximo estado.

- El uso de biestables tipo T muestra un coste menor cuando el estado presente es muy parecido al complemento del próximo estado.
- El uso de biestables de tipo JK disminuye el coste del circuito combinacional.
- El uso del biestable RS es menos costoso.
- Un único tipo de biestable permite conseguir uniformidad en el diseño.

4.2. Aspectos temporales.

Un segundo grupo de aspectos muy importante en la elección del tipo de biestable es la temporización que se va a utilizar. Estos aspectos tendrán todos una naturaleza temporal.

Por ejemplo, si se va a diseñar un circuito asíncrono en modo fundamental, no será necesario ningún biestable al utilizarse las realimentaciones directas.

Así mismo, en el caso de utilizar biestables, habrá que decidir cómo serán disparados. Por lo tanto, podemos tener biestables transparentes, latches o flip-flops.

4.2.1. Principales diferencias entre un sistema síncrono y asíncrono.

A continuación veremos las principales diferencias entre un sistema síncrono y otro asíncrono. Estas diferencias se basan en las diferentes formas de operación, básicamente debido a la existencia de una señal global que controla la operación, denominada generalmente reloj, o a la implementación de un protocolo de comunicaciones.

En primer lugar, en los sistemas síncronos se debe garantizar que la señal de reloj llegue a todos los biestables de forma simultánea (ya que todos los biestables deben almacenar la información en el mismo instante temporal para empezar la operación con los datos correctos de la historia), y el rango inactivo de la señal de reloj debe ser superior al retraso de la lógica combinacional asociada (para garantizar la finalización de la operación en el momento oportuno). Según esta restricción, para circuitos grandes la frecuencia de la señal de reloj, y por tanto la velocidad del sistema, puede llegar a ser excesivamente baja. Para solucionar este problema se utiliza una técnica denominada ***pipeline***, que también puede ser utilizada en los diseños asíncronos con el mismo propósito, aumentar la velocidad del sistema.

Esta técnica consiste en dividir el diseño en varias partes más pequeñas e intercalar entre ellas elementos de memoria, como se puede ver en la figura 5.5. De esta forma la velocidad estará limitada por los bloques más pequeños y con menos retraso en lugar del bloque total con un mayor retraso. En el esquema con pipeline los datos de salida corresponderán a los datos del bloque B2, que excepto el primer dato, saldrá con la velocidad limitada por B1 o B2, a diferencia del esquema sin pipeline que la velocidad viene limitada por el bloque total.

La operación en este sistema de pipeline será como sigue. En un primer lugar, el dato identificado como D1 entra en el bloque B1, mientras que en el segundo no hay ningún dato. Una vez que haya realizado la operación el primer bloque, el resultado del dato D1 pasa al bloque B2 en el siguiente ciclo de operación, liberando el primer bloque. Como el primer bloque ya está libre, puede realizar la operación correspondiente al segundo dato D2. Por lo tanto, en el interior de este circuito en particular se encontrarán dos datos diferentes, cada uno en las

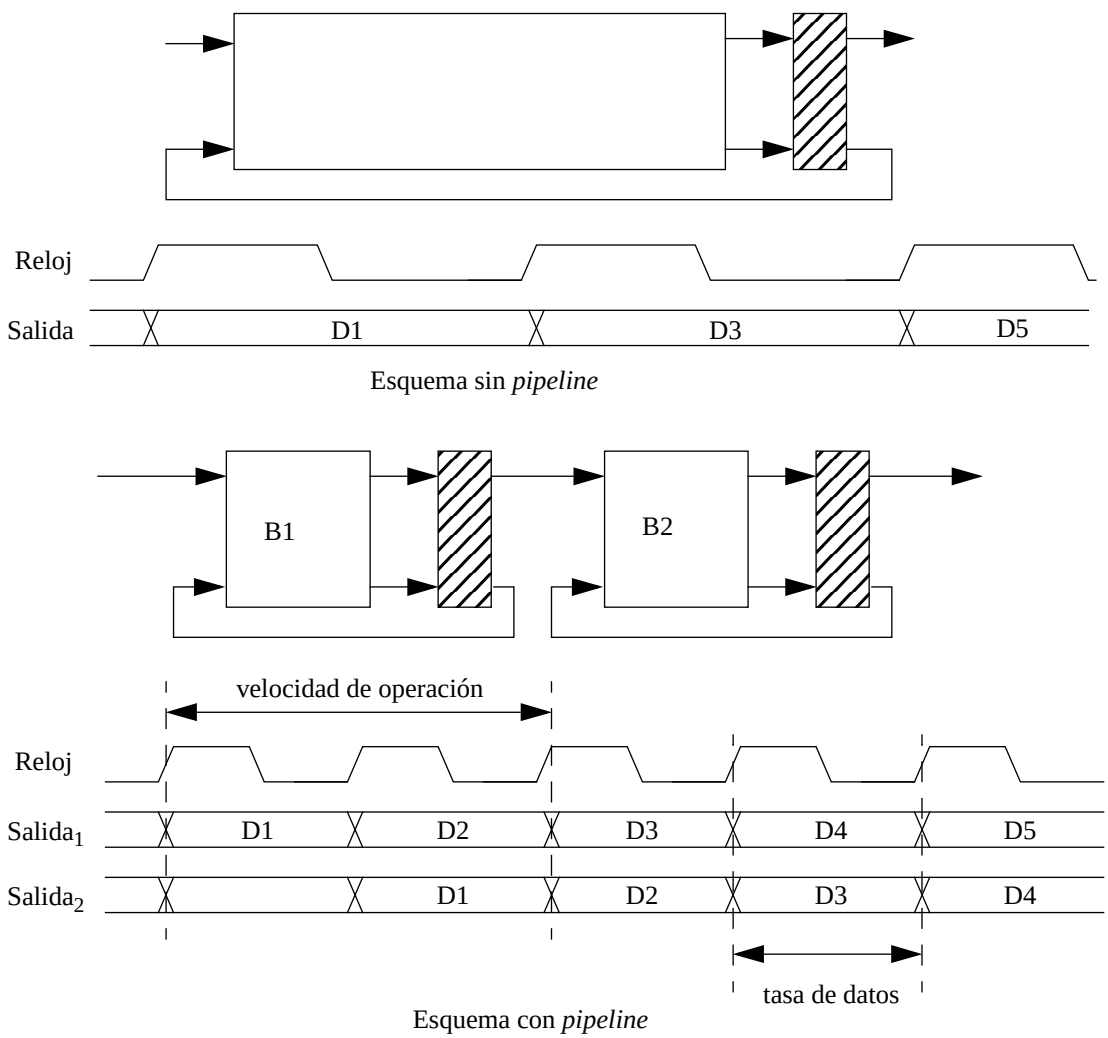


Figura 5.5.- Ilustración de la técnica de pipeline. Formas de onda para un circuito sin pipeline y otro con pipeline.

diferentes etapas del pipeline. Entonces encontramos dos parámetros temporales diferentes, los cuales los podríamos denominar como velocidad de operación y tasa de datos (también conocido como *throughput*).

> La **velocidad de operación** se definirá como la velocidad a la que se realiza toda la operación del circuito sobre un dato (desde que entra en el bloque B1 hasta que sale del bloque B2).
>
> La **tasa de datos** es la velocidad a la que el circuito admite la entrada de nuevos datos, o lo que es lo mismo, devuelve los resultados de cada operación (es decir, la velocidad a la que el bloque B1 o B2, el más lento, realiza su operación).

De los parámetros anteriores, el más importante es la tasa de datos porque es la velocidad que se ve desde el exterior, ya que se estarán realizando varias operaciones de forma concurrente.

Utilizando o no la técnica de *pipeline*, la señal de almacenamiento de un sistema síncrono, el reloj, es externa al sistema por lo que la velocidad vendrá supeditada al bloque de

mayor retraso (considerando todas las combinaciones de entrada), diciéndose entonces que este tipo de circuitos muestra la **operación del caso peor**. En cambio, en los circuitos asíncronos, el almacenamiento en los elementos de memoria es interna produciéndose cuando se hayan generados los datos de salida, es decir, cuando la operación haya finalizado. Esta situación provoca que la velocidad del sistema dependa de los datos de entrada, por lo que se toma como velocidad del sistema el valor medio de las velocidades para todos los patrones de entrada. En este caso se dice que estos circuitos muestran una **operación del caso medio**.

Cuando utilizamos la técnica de pipeline, la señal de reloj debe llegar forzosamente a todos los elementos de memoria de forma simultánea para garantizar un correcto funcionamiento. En la figura 5.6 se muestra el paso de la secuencia 0 -> 1 a través de un circuito en *pipeline*. En esta figura, la secuencia correcta de datos debería ser tal que el último biestable no debería almacenar el resultado de la operación en el mismo ciclo que se inicia, sino en el siguiente. Esta secuencia solamente se sigue cuando $\Delta_1 > \Delta_2$, es decir, cuando se puede considerar que las señales de reloj de ambos biestables es la misma o la señal de reloj llega simultáneamente a todos los biestables. La situación anómala que puede producir fallos de operación se conoce como ***clock skew***. Esta situación es más crítica a medida que crece el tamaño del circuito por lo que el retraso de las líneas de conexión se hace cada vez menos despreciable. Para tratar de reducir este problema se requiere complicar el diseño, entre otros motivos debido a la adición de una lógica necesaria para distribuir de forma adecuada la señal de reloj. Este problema no es tan restrictivo en los diseños asíncronos al carecer de una señal global de reloj. Este hecho también viene motivado por la no necesidad de que todos los biestables deban almacenar la información de forma simultánea, es más, esta situación es la menos usual ya que solamente se produciría cuando el retraso de todos los bloques sea el mismo.

Otro problema de los diseños síncronos consiste en que los bloques realizarán una operación cuando cambien alguna de sus entradas, aunque esta situación no sea necesaria. Esta situación repercute en la existencia de un consumo de potencia innecesario. Este parámetro, el consumo de potencia, cada vez va adquiriendo mayor importancia debido al creciente auge observado en los denominados equipos “sin cable”, como pueden ser teléfonos móviles y ordenadores portátiles. En cambio, este consumo innecesario no está presente en los circuitos asíncronos debido al protocolo de comunicaciones, el cual garantiza que solamente se realizará la operación cuando ésta sea estrictamente necesaria.

Hasta ahora todos son ventajas para el diseño asíncrono. No obstante, esta situación no es del todo cierto ya que el protocolo de comunicaciones, que provoca todas las ventajas, debe ser implementado, mientras que en el diseño síncrono no se realiza dicha implementación (el protocolo es la señal de reloj). Esta implementación provoca:

- un mayor retraso, reduciendo las ventajas de la operación del caso medio,
- y un mayor consumo de potencia, reduciendo las ventajas de evitar las operaciones innecesarias.

Por lo tanto, podemos concluir que no existe un sistema ideal para todos los casos como se deduce del resumen mostrado en la tabla 5.5. De forma orientativa, podemos seguir las siguientes premisas:

- Los sistemas síncronos son los adecuados cuando las operaciones se suceden de forma más o menos periódica, por lo que pueden ser sincronizadas fácilmente por una señal periódica externa, el reloj.

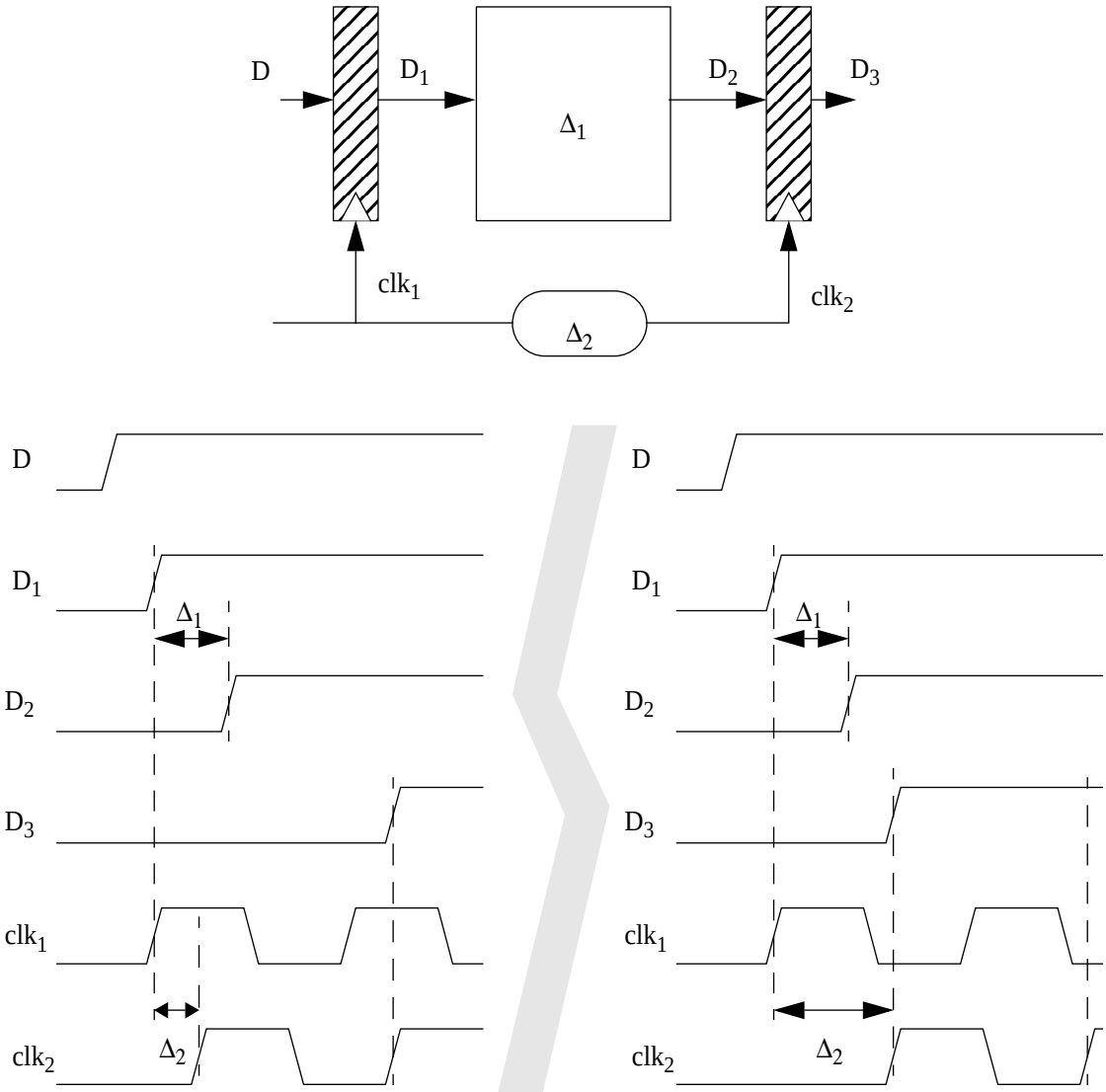


Figura 5.6.- Influencia del retraso sobre las líneas de las señales de reloj, *clock skew.*

- Los sistemas asíncronos son los adecuados cuando las operaciones no siguen un flujo temporal periódico, sino que es totalmente aleatorio. En este caso, la sincronización con un señal externa es muy complicada.

Parámetro	Sistema síncrono	Sistema asíncrono
Modo de operación	Caso peor	Caso medio
Clock skew	Problemático	Efecto mínimo
Consumo de potencia	Alto	Necesario
Implementación del protocolo	Fácil	Complicada

Tabla 5.5. Resumen de las diferencias entre los sistemas síncronos y asíncronos.

4.2.2. Disparo de los elementos de memoria.

Otro aspecto temporal importante consiste en la elección del disparo de los elementos de memoria, es decir, si utilizamos elementos transparentes, latches o flip-flops. En la mayoría de los casos se utilizan flip-flops, ya que el periodo activo de la señal de control (y por tanto, el periodo en el que el circuito debe permanecer inactivo) está reducido al mínimo, produciendo un mejor aprovechamiento del tiempo.

4.3. Ejemplo.

Continuando con el ejemplo anterior, en las tabla 5.6 y tabla 5.7, mostramos las tablas de excitación para el uso de biestables tipo D y T.

	000	001	011	010	110	111	101	100
000	000, 0 *	000, 0 *	000, 0 *	000, 0 *	011, 1	011, 1	011, 1	011, 1
001	001, 0 *	010, 0	000, 0	101, 0	001, 0 *	001, 0 *	001, 0 *	001, 0 *
011	011, 0 *	001, 0	111, 0	010, 0	011, 0 *	011, 0 *	011, 0 *	011, 0 *
010	010, 0 *	101, 0	000, 0	111, 0	010, 0 *	010, 0 *	010, 0 *	010, 0 *
110	---, -	---, -	---, -	---, -	---, -	---, -	---, -	---, -
111	111, 0 *	000, 0	000, 0	000, 0	111, 0 *	111, 0 *	111, 0 *	111, 0 *
101	101, 0 *	100, 0	000, 0	111, 0	101, 0 *	101, 0 *	101, 0 *	101, 0 *
100	100, 0 *	111, 0	000, 0	000, 0	100, 0 *	100, 0 *	100, 0 *	100, 0 *

Tabla 5.6. Tabla de excitación correspondiente al diagrama de la máquina de refrescos, utilizando biestables tipo D.

	000	001	011	010	110	111	101	100
000	000, 0 *	000, 0 *	000, 0 *	000, 0 *	011, 1	011, 1	011, 1	011, 1
001	000, 0 *	011, 0	001, 0	100, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *
011	000, 0 *	010, 0	100, 0	001, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *
010	000, 0 *	111, 0	010, 0	101, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *
110	---, -	---, -	---, -	---, -	---, -	---, -	---, -	---, -
111	000, 0 *	111, 0	111, 0	111, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *
101	000, 0 *	001, 0	101, 0	010, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *
100	000, 0 *	100, 0	100, 0	100, 0	000, 0 *	000, 0 *	000, 0 *	000, 0 *

Tabla 5.7. Tabla de excitación correspondiente al diagrama de la máquina de refrescos, utilizando biestables tipo T.

5. Realización física del circuito.

Una vez que hemos obtenido la tabla de excitación, hemos eliminado toda dependencia temporal por lo que hemos simplificado el diseño a un problema completamente combinacional. Este problema se resuelve utilizando las técnicas que han sido estudiadas en el curso anterior.

Para el caso de utilizar biestables tipo D, cuya tabla de excitación se muestra en la tabla 5.6, la función que hay que implementar expresada como suma de mintérminos es:

$$S(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(32, 40, 48, 56) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$D_0(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(1, 3, 5, 7, 10, 11, 12, 17, 18, 21, 27, 32, 33, 35, 37, 39, 40, 41, 43, 45, 47, 48, 49, 51, 53, 55, 56, 57, 59, 61, 63) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$D_1(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(2, 3, 7, 9, 12, 18, 19, 21, 27, 32, 34, 35, 39, 40, 42, 43, 45, 48, 50, 51, 53, 56, 58, 59, 61) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$D_2(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(4, 5, 7, 10, 12, 13, 17, 18, 21, 27, 36, 35, 37, 44, 45, 47, 52, 53, 55, 60, 61, 63) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

Para el caso de utilizar biestables tipo T, cuya tabla de excitación se muestra en la tabla 5.7, la función que hay que implementar expresada como suma de mintérminos es:

$$S(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(32, 40, 48, 56) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$T_0(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(9, 10, 13, 15, 18, 19, 23, 25, 29, 31, 32, 40, 48, 56) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$T_1(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(9, 10, 11, 15, 21, 23, 26, 31, 32, 40, 48, 56) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$

$$T_2(B, I_1, I_2, Q_2, Q_1, Q_0) = \sum m(10, 12, 15, 17, 18, 20, 23, 27, 28, 29, 31) + \phi\ (6, 14, 22, 30, 38, 46, 54, 63)$$