SISTEMAS SECUENCIALES

Un Sistema Secuencial es un Sistema Digital cuyos vectores de salida dependen no sólo del vector de entrada actual sino también del anterior o los anteriores. En otras palabras un Sistema Secuencial debe ser capaz de "memorizar" la evolución de los vectores de entrada y determinar el vector de salida en función de la misma.

Es posible interpretar el mismo concepto indicando las relaciones de vectores de entrada y salida de la siguiente manera:

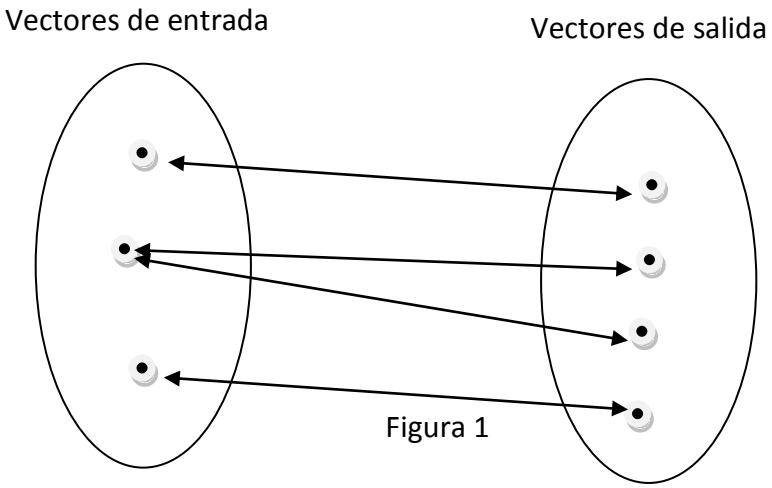


Figura 1

Este tipo de relación en donde al menos a un vector de entrada le corresponde dos o más vectores de salida nos permite deducir que las salidas no pueden obtenerse como funciones lógicas de las entradas como en el caso de los Sistemas Combinacionales y representa una herramienta para determinar la característica Combinacional o Secuencial de un problema en particular.

El realización de Sistemas Secuenciales se basa en realimentar un Sistema Combinacional es decir: conectar sus salidas como entradas. Para comprender porqué esta realimentación permite lograr comportamiento secuencial, planteemos un ejemplo:
Ejemplo: Construir un Sistema Digital cuya salida comande un LED que indique la situación de una puerta, si la puerta se ha abierto alguna vez el LED debe encenderse (aún si la puerta se ha cerrado).

Las entradas para este sistema son:

- X0 : indica el estado de la puerta, 0 cerrada, 1 abierta
- X1: entrada de inicialización, 1 apaga el LED (inicialización), 0 no implica acción.

La salida es la señal que comanda el LED, z = 0 led apagado, z=1 led encendido.

Solución:

Planteo de la Tabla de Verdad

X1	X0	Z
0	0	0 ó 1
0	1	1
1	0	0
1	1	No Definido (ND)

Figura 2

En la primera fila se ve que la salida será 0 ó 1 dependiendo de cuál ha sido el vector anterior a 00.

Si el vector de entrada anterior fue 01, la salida para 00 será 1.

Si el vector de entrada anterior fue 00, la salida para 00 será 0.

Se aprecia la correspondencia entre entradas y salidas de naturaleza secuencial, también se ve que este tipo de correspondencia incluye la relación de las salidas con las entradas mencionada al principio (las salidas no sólo dependen del la entrada actual sino también de de las anteriores).

Una forma de tener en cuenta el estado anterior de las entradas es considerar la salida como entrada adicional, por ej.: si la salida anterior era 0 y la entrada es 00, entonces la salida deberá seguir en 0.

Se propone la siguiente tabla de verdad:

INSTANTE ANTERIOR T			INSTANTE POSTERIOR T+1
Zt	X1	X0	Zt+1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	ND
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	ND

Figura 3 – TVT Tabla de Verdad Temporal

Esta tabla la llamamos TABLA DE VERDAD TEMPORAL (TVT) ya que el concepto de tiempo está incluido en la misma: a la izquierda el instante anterior (el Sistema tiene los valores indicados en el instante T), a la derecha el posterior (los valores anteriores de Sistema implica el nuevo valor de la salida)

Si esta TVT será usada para implementar Sistemas Secuenciales, se deberá reconsiderar el concepto de función lógica y ampliarlo. Una función lógica es una variable binaria que depende de otras variables binarias relacionadas por las operaciones lógicas. Ahora una función podrá además depender de sí misma:

$$P = f(a,b,c,..,p,..)$$

Se ve que si no se agrega el concepto de tiempo se caerá en el absurdo de que una variable binaria puede adoptar dos valores a la vez. Esta inclusión del tiempo nos lleva a esta notación

$$pt = f(a,b,c,\ldots,pt+1,..)$$

Así, en la línea sombreada de la TVT se acepta que z tenga dos valores distintos, uno para t y otro para t+1.

Como lo Sistemas combinacionales implementan funciones lógicas, es entonces posible implementar un Sistema Secuencial realimentando un Sistema Combinacional pero teniendo en cuenta el tiempo. El esquema propuesto es:

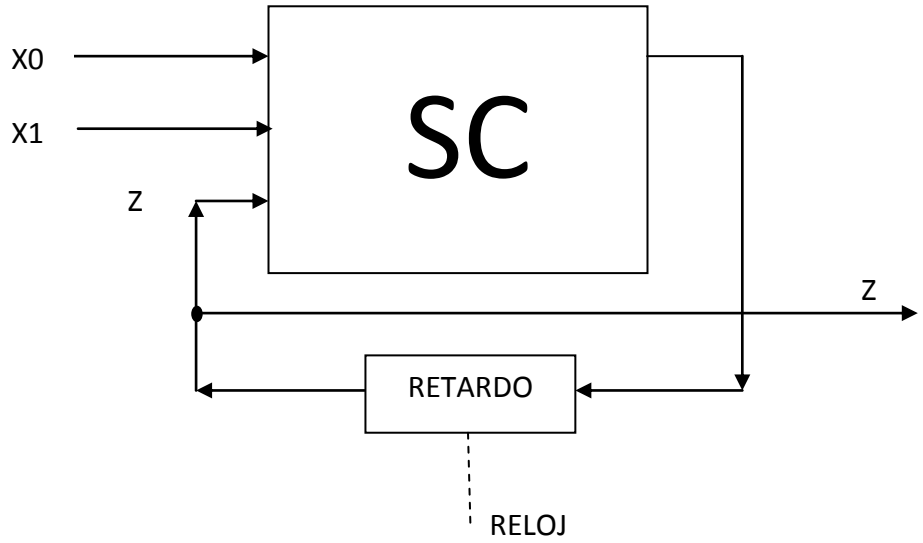


Figura 4

El RETARDO implementa el concepto de tiempo, es un circuito que retarda la realimentación de la salida z, evitando el absurdo de que z adopte dos valores a la vez.

Dependiendo de cómo se implementa el RETARDO resultará un Sistema Secuencial Asíncrono (SSA) si es un circuito que simplemente retarda un cierto tiempo Td a la señal en cuestión, o un Sistema Secuencial Síncrono (SSS) si es un circuito que permite que la señal de entrada pase a la salida en el momento de un flanco de una señal externa de sincronismo llamada reloj.

En el caso de nuestro ejemplo de la puerta, se tiene:

$Zt+1=f(xo,x1,zt)= \Sigma(1,4,5)+ \Phi(3,7)$

Z \ X1X0	00	01	11	10
0			X	1
1	1		X	1

Figura 5

Si no se consideran los minterms 3 y 7:

$Zt+1 = x1´x0 + x1´zt = \overline{x1´(x0+ zt)} = \overline{x1 + x0+zt}$

Si se consideran los minterms 3 y 7:

$Zt+1 = \overline{x0 + x1´zt} = \overline{x0´ x1´zt}$

Uno de los circuitos obtenidos es:

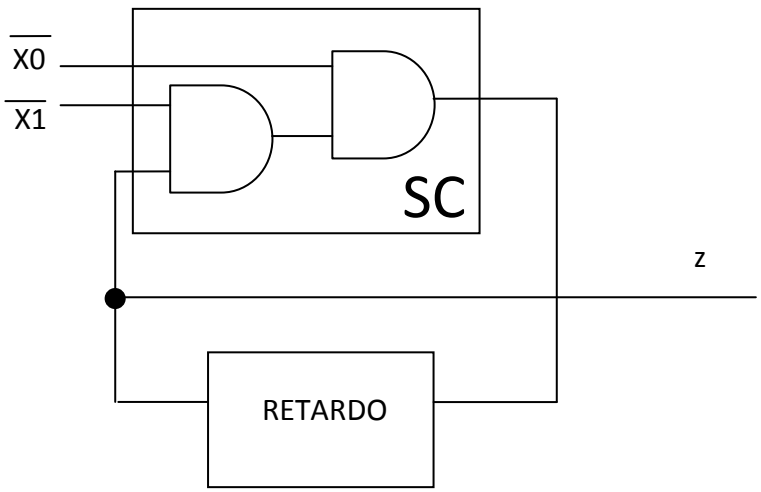


Figura 6

El RETARDO puede implementarse con el tiempo de propagación propio de las compuertas lógicas reales. De esta manera no sería necesario dibujar el RETARDO ya que estará automáticamente considerado.

El circuito obtenido se llama Memoria de un bit o también Biestable Set-Reset (SR) asíncrono y se simboliza así:

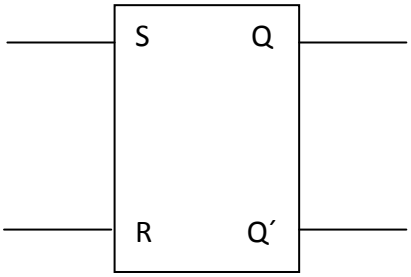


Figura 7

Donde:
S = x0, R = x1 y Q = z

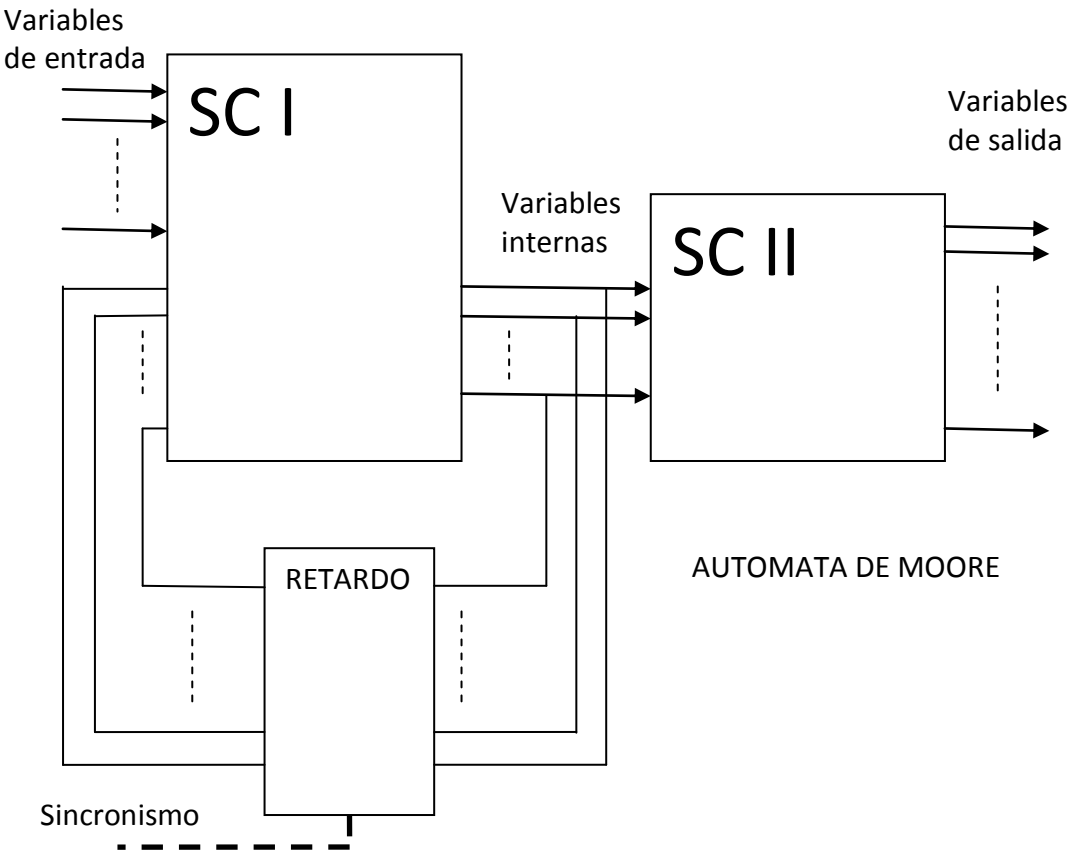
Generalizando el planteo de Sistemas Secuenciales podríamos suponer que las salidas son funciones lógicas de las variables que se realimentan y no necesariamente estas.

Surge de lo expuesto distinguir tres tipos de vectores en un Secuencial:

- Vectores de entrada
- Vectores internos
- Vectores de salida

Aparecen, con respecto a los Sistemas Combinacionales, un nuevo tipo de variables: variables internas. Estas variables internas tienen por objeto implementar el concepto de memoria propio de los secuenciales.

Se propone entonces los siguientes diagramas en bloques de Sistemas Secuenciales:


Variables
de entrada
SC I
Variables
de salida
Variables
internas
SC II
RETARDO
AUTOMATA DE MOORE
Sincronismo

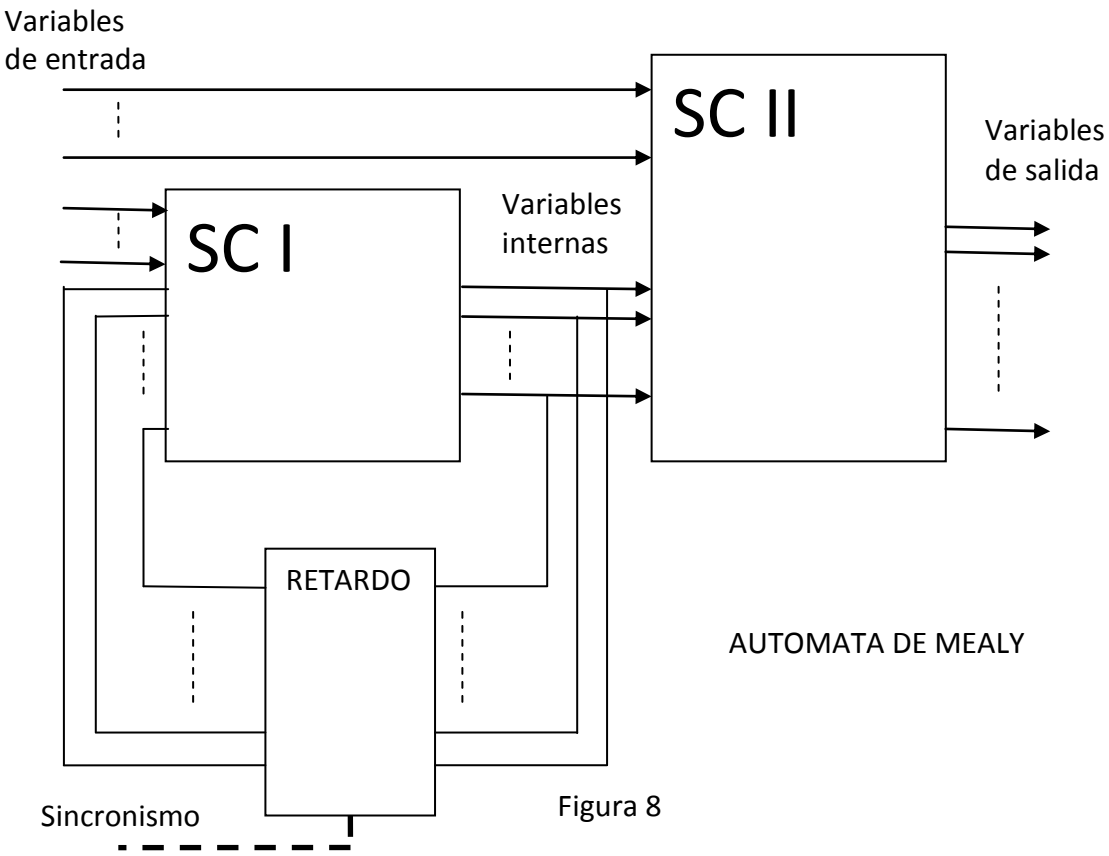

Variables
de entrada
SC II
Variables
de salida
SC I
Variables
internas
RETARDO
AUTOMATA DE MEALY
Sincronismo

Figura 8

Las salidas del autómata de Moore son funciones lógicas de las variables internas (o ellas mismas), en un Autómata de Mealy las salidas son funciones lógicas de las variables internas y las de entrada. Se verá que siempre es posible obtener los dos Autómatas. La palabra *autómata* se aplica porque se puede ver a un secuencial como una máquina que evoluciona automáticamente entre una cantidad finita de estados internos.

Se considera que el retardo puede estar implementado con el tiempo de propagación del SCI, o bien puede consistir en circuitos que dejan pasar el valor lógico de sus entradas ante la presencia de un flanco en una señal de sincronismo externa (indicada con línea de puntos). Lo mencionado permite clasificar a los Secuenciales en.

SISTEMAS SECUENCIALES
- ASÍNCRONOS
- SÍNCRONOS

Cada valor del Vector Interno representa un ESTADO INTERNO del secuencial, que llamaremos simplemente ESTADO. Ante la aparición de un nuevo vector de entrada, el secuencial evolucionará entre estados pudiendo llegar a un estado estable, en el cual permanecerá hasta la aparición de un nuevo vector de entrada. Entonces:

ESTADO: distintos valores de las variables internas.
ESTADO ESTABLE: Un estado es estable, para un vector de entrada (que suponemos permanece estable durante un tiempo) cuando el próximo estado hacia el cual evoluciona el secuencial es el mismo.

Observando el diagrama en bloques se puede aclarar el concepto: Supongamos que el Secuencial está “quieto”, es decir que para una entrada estable se encuentra en un estado interno y este no cambia (es decir: es estable). Ahora la entrada adopta un nuevo valor (que permanece durante un tiempo), cambiarán las variables internas (salidas del SCI), se realimentarán y provocarán otro valor de las variables internas, estos cambios continuarán hasta que el vector interno anterior y posterior coincidan, se ha llegado a un estado estable.

El diseño del secuencial implica que existirán estados estables, justamente este método de realimentación apunta a poder memorizar la evolución de los vectores de entrada mediante estados internos estables.

CONSIDERACIONES:

- Se concluye por lo expuesto que el diseño de un secuencial puede realizarse planteando la TVT particularizada según se trate de un SSA o de un SSS. La TVT involucra el tiempo anterior T y el tiempo posterior T+1, como se ve en el ejemplo de la puerta, y también conviene encolumnar las salidas. Una TVT tendrá entonces el siguiente aspecto:

TIEMPO ANTERIOR T		TIEMPO POSTERIOR T+1	
Variables de entrada	Variables internas	Variables internas	Variables de salida
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.

Figura 9 - TVT

Las Variables de salida pueden ser funciones de las variables internas en T+1 (autómata de MOORE) o bien de las variables internas en T+1 y las de entrada (autómata de MEALY).

La evolución desde el tiempo T al T+1 ocurrirá espontáneamente (SSA) o en sincronismo con el flanco de una señal de reloj externa (SSS).

- Para que este esquema funcione los vectores de entrada deberán permanecer estables mientras el sistema evoluciona en pos de un estado estable. Si el secuencial es un SSA este tiempo de permanencia del vector de entrada deberá ser mayor que el que tardan todas las evoluciones intermedias (estados inestables) de las variables internas. Es objeto del diseño lograr que estas evoluciones intermedias se reduzcan a sólo una, de esta forma, la permanencia del vector de entrada, deberá ser mayor al tiempo de propagación del SCI. Si el secuencial es un SSS, evoluciona en sincronismo a los flancos del reloj externo y la entrada no debería cambiar durante un tiempo mayor al del periodo del reloj.
- Si el secuencial es un SSA puede ocurrir que las variables internas tengan que cambiar a un valor no adyacente al actual (transiciones no adyacentes de las variables internas TNA), en este caso las variables internas pasarán por estados intermedios que se realimentan antes de alcanzar el estado final, esta realimentación no prevista podría llevar al secuencial a estados no deseados (transiciones no adyacentes críticas TNAC). Estas TNAC deben evitarse. Si el secuencial es SSS, esto no ocurre siempre que el tiempo entre flancos se mayor que el tiempo de propagación de SCI.
- En un secuencial SSA un cambio no adyacente de las variables de entrada puede provocar un malfuncionamiento por razones similares al punto anterior. Estas transiciones deben ser consideradas en el diseño. En los secuenciales SSS, estas no adyacencias no implican un problema ya que suponemos que el transitorio habrá desaparecido a la salida del SCI cuando llegue el próximo flanco del reloj.
- Las salidas de un MOORE presentan menos fluctuaciones indeseadas que un MEALY ya que sólo dependen de las variables internas, como contrapartida un MOORE tiene más estados internos que un MEALY lo que hace más complejo al circuito SCI.
- En algunos casos debemos diseñar un SSA que responda a los flancos de las variables de entrada, en realidad en los casos prácticos se presentan los problemas que implican tratar

variables de entrada que actúan por nivel y otras por flancos. Para resolver este problema veremos un método de diseño que permite caracterizar al SSA para algunas variables de entrada que actúan por flancos y otras por nivel, o sólo por flanco.
El concepto de caracterizar un SSA por los flancos de las variables de entrada representa una abstracción que resulta útil ya que permite un planteo más sencillo para muchos problemas prácticos, además hay casos en los que sólo pueden resolverse aplicándola. Considérense los siguientes ejemplos:
Ej1: Se desea realizar un secuencial cuya única salida cambie de estado cuando su única entrada experimenta un flanco de subida
Diagrama de tiempo:

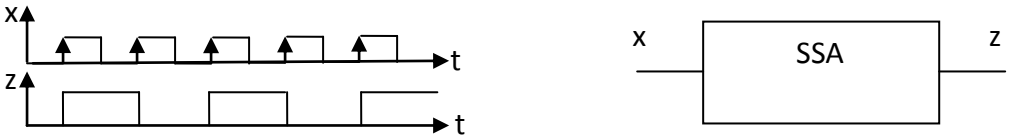


Figura 10

Se ve que la caracterización es por flancos ya que son los que generan cambios en la salida, en este ejemplo no es posible caracterizar al secuencial aplicando el concepto de nivel.
Ej2: Ejemplo de la puerta dado anteriormente:
Diagrama de tiempo:

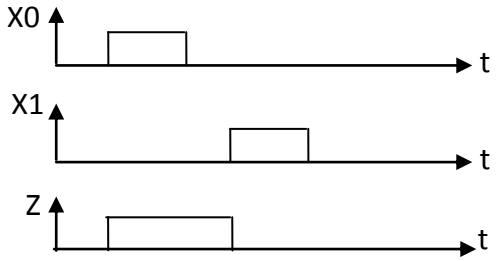


Figura 11

Este es un ejemplo de caracterización por niveles. Si quisiéramos caracterizar por flancos este ejemplo, deberíamos especificar los requerimientos del secuencial de la siguiente manera:
Especificaciones: Realizar un secuencial que tenga dos variables de entrada (x0 y x1) y una de salida (z):

- cuando se produce un flanco de subida en x0, la salida debe ser z=1.
- Cuando se produce un flanco de subida en x1, la salida debe ser z=0.

Diagrama de tiempo:

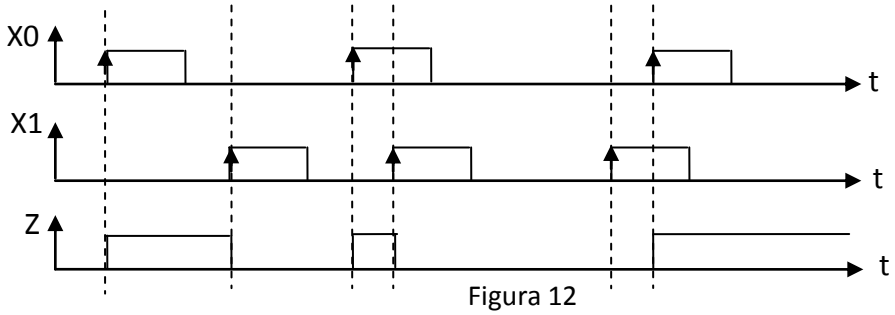


Figura 12

Se puede suponer que el secuencial resultante será más complejo y seguramente tendrá más de dos estados internos.
Para diseñar SSA caracterizados por nivel veremos que puede usarse el planteo por TF, pero frecuentemente resultan TF complicadas en las cuales no es difícil equivocarse. Por lo tanto si el secuencial es simple (como el Ej.1) es recomendable la utilización de TF, si el secuencial es más complejo se recomienda la utilización de GF caracterizado por flancos.

Aplicación de conceptos

Aplicaremos los conceptos vistos al problema de la puerta:

- El secuencial obtenido es un SSA
- Posee 2 estados internos ya que tiene 1 variable interna. Por esto lo llamamos biestable.
- Se trata de un autómata de MOORE en el cual el SCII no está, la variable de salida coincide con la interna.
- No hay problemas de TNAC de variables internas ya que sólo tiene una.
- El caso de no adyacencia de los vectores de entrada puede verse en la TVT que se repite abajo
  Supongamos que el vector de entrada es X1X0 = 01 (FILA6) y cambia a 10 , lo que ocurrirá es que pasará por algunos de los estados intermedios 00 ó 11.

	INSTANTE ANTERIOR T			INSTANTE POSTERIOR T+1
	Zt	X1	X0	Zt+1
FILA 1	0	0	0	0
FILA 2	0	0	1	1
FILA 3	0	1	0	0
FILA 4	0	1	1	0
FILA 5	1	0	0	1
FILA 6	1	0	1	1
FILA 7	1	1	0	0
FILA 8	1	1	1	0

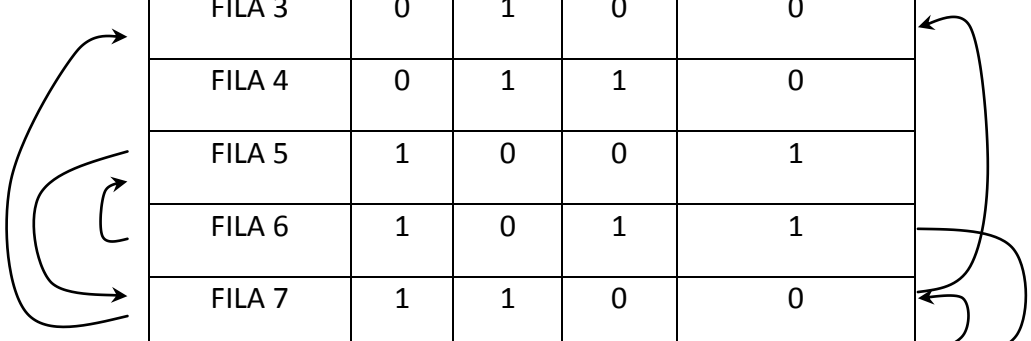


Figura 13

Flechas de la izquierda de la figura 13

- Supongamos que pase por 00, la salida vale 1 (FILA 5)
- Luego pasa a 10 , la salida vale 0 (FILA 7)
- Finalmente queda con salida 0 (FILA3)

Conclusión: La TNA de entrada no es crítica

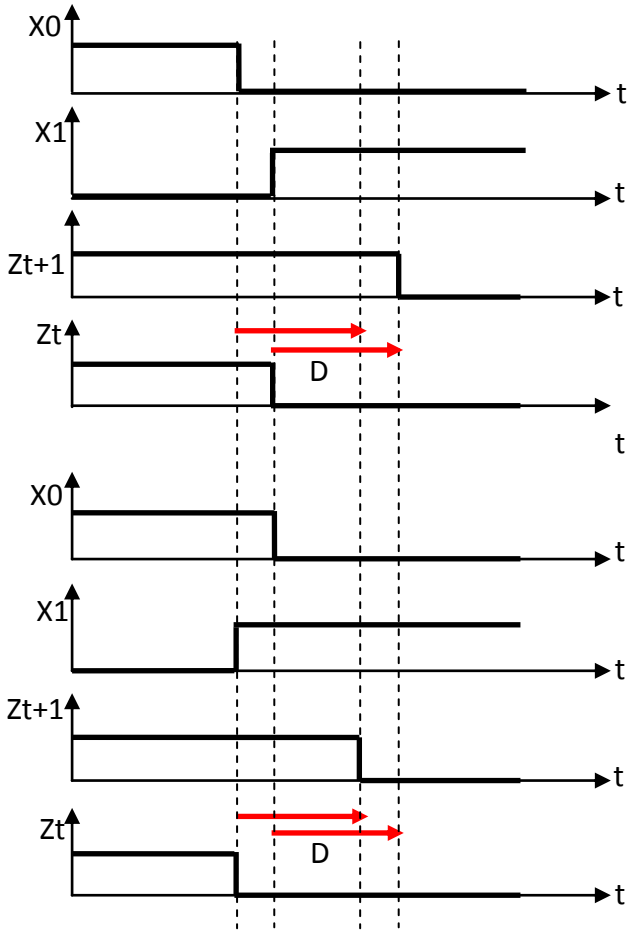
Flechas de la derecha de la figura 9

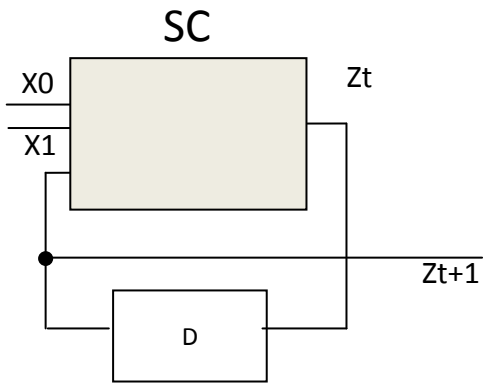
- Supongamos que pasa por 11, si NO hemos considerado presente el Minterms correspondiente, la salida vale 0 (FILA 8)
- Luego pasa a 10, la salida vale 0 (FILA 7)
- Finalmente queda con salida 0 (FILA3)

Conclusión: La TNA de entrada no es crítica.

La diferencia entre los dos caminos intermedios posibles es que la salida llega antes a 0 en el segundo caso, pero en ninguno es crítica.

Este razonamiento puede realizarse en Diagramas de tiempo:





Suponemos que el SC no tiene tiempo de propagación y el retardo D está concentrado en D. Todas las transiciones no adyacentes pueden considerarse como tales si ocurren dentro de un $\Delta t < D$. Los retardos D están indicados con una flecha en los diagramas de tiempo.

Como conclusión de lo expuesto es recomendable evitar las TNA de entrada en los SSA ya que considerarlas implica un análisis que se complica cuando el número de variables de entrada aumenta y las distancias de las no adyacencias es mayor a 2.

Figura 14

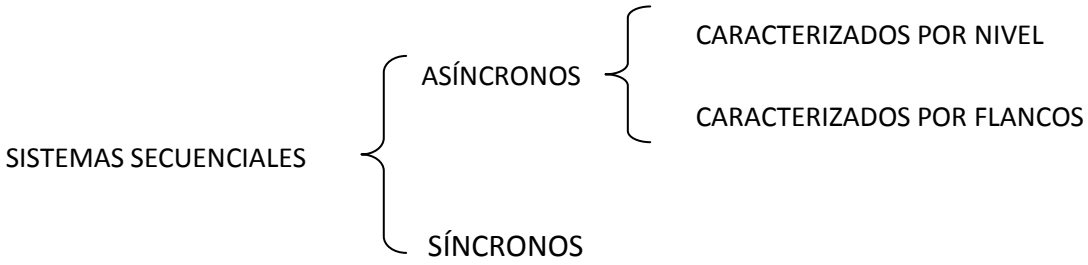
En

El ejemplo anterior es un secuencial con un solo Estado Interno (codificado mediante una variable interna) por lo tanto no puede tener TNA de las variables internas, entonces, ya que las no adyacencias de entrada evolucionan sin depender de ellas mismas, una TNA de entrada

no puede provocar un mal funcionamiento. Si el Secuencial tiene más de una variable interna pueden aparecer TNA internas y habrá que, necesariamente, evaluar las TNA de entrada.

En el caso de los SSS, las TNA de entrada no afectan el funcionamiento del secuencial ya que entre los flancos de la señal de sincronismo las TNA se resolverían. En estos SSS, si el problema a resolver implica TNA que deben considerarse, podría ser una solución sincronizar las variables de entrada en el flanco opuesto al de evolución del secuencial, así tosas las variaciones del vector de entrada que sucedan dentro del periodo de la señal de sincronismo no afectarían el funcionamiento. Si fuera necesario considerarlas habría que aumentar la frecuencia de la señal de sincronismo y paralelamente utilizar una tecnología más rápida.

Lo mencionado nos lleva a completar el cuadro de clasificación de los Sistemas Secuenciales:



DISEÑO

Como se mencionó anteriormente el diseño de secuenciales se basa en la obtención de la TVT cuya interpretación se particulizará para cada tipo de Secuencial. Seguidamente se presenta una metodología de diseño para SSA y SSS resolviendo un ejemplo para todos los casos.

A fin de tener presente que existen alternativas de diseño, se presenta un diseño para SSA híbrido que representa una solución diferente para las TNAC.

Los pasos a seguir son los siguientes:

I) ESPECIFICACIONES LITERALES

II) ELECCIÓN DEL MÉTODO DE SÍNTESIS PARA OBTENER EL DIAGRAMA DE FLUJO (MEALY, MOORE)

- **Utilizando Tabla de Fases (TF)**
- **Utilizando Grafo Reducido (GR)**
  - **Caracterización por niveles**
  - **Caracterización por flancos (sólo Asíncronos)**

III) CODIFICACIÓN DE ESTADOS INTERNOS

- **SECUENCIAL ASÍNCRONO**
- **SECUENCIAL SÍNCRONO**

IV) OBTENCIÓN DE LA TABLA DE VERDAD TEMPORAL (TVT)

V) OBTENCIÓN DE LAS FUNCIONES LÓGICAS INTERNAS

- **SECUENCIAL ASÍNCRONO**
- **SECUENCIAL SÍNCRONO**

VI) OBTENCIÓN DE LAS FUNCIONES LÓGICAS DE SALIDA

VII) SÍNTESIS

Problema:

Se desea implementar un Sistema Digital que permita seleccionar 3 tamaños diferentes de cajas que viajan en fila sobre una cinta transportadora y derivarlo a 3 cintas transportadoras para su posterior procesamiento. El Sistema Digital, para detectar el tamaño, consta de 3 sensores ópticos ubicados longitudinalmente y separados una distancia adecuada a los tamaños a determinar. Los productos que viajan por la cinta lo hacen con una separación tal que ingresan a la zona de detección cuando el anterior ya ha pasado por la misma.

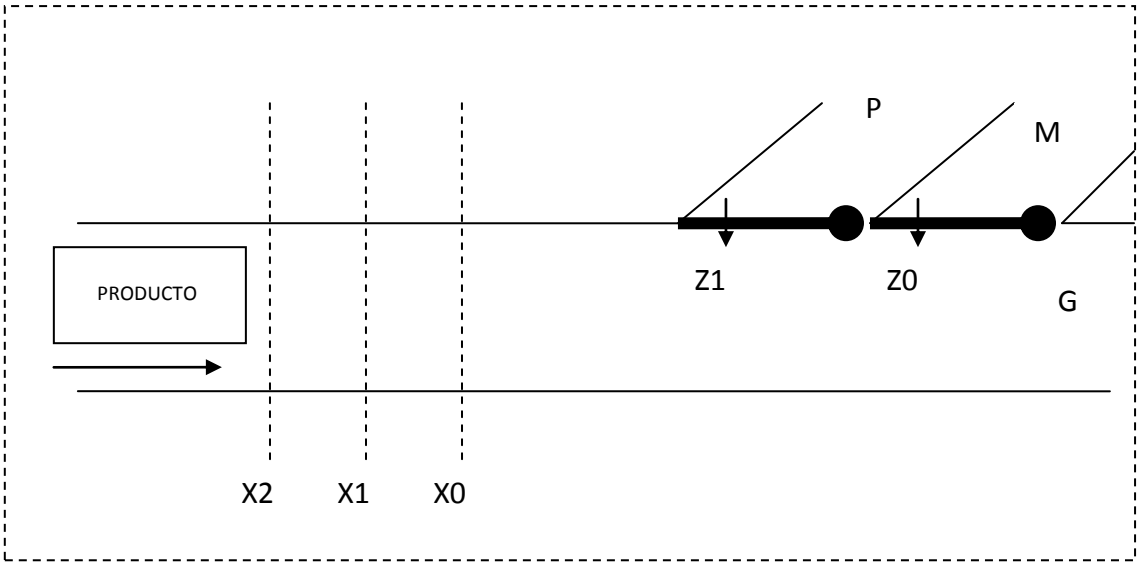


Figura 15

Nuestro Sistema será entonces:

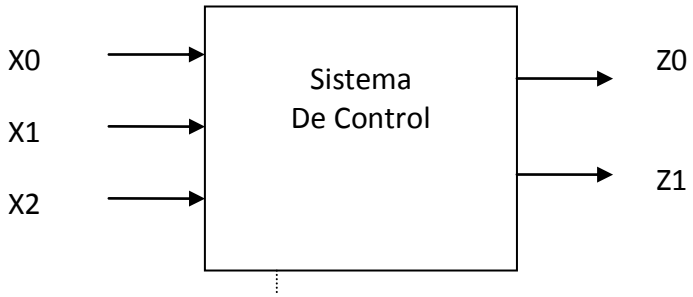


Figura 16

SÍNTESIS

I. ESPECIFICACIONES LITERALES

El problema a resolver en general está especificado literalmente. Aquí es necesario definir los vectores de entrada, los de salida y sus correspondencias básicas, determinada la naturaleza del Sistema Digital, es decir si se trata de un Sistema Digital Combinacional o un Sistema Digital Secuencial.

En nuestro problema vemos tres posibles secuencias de vectores de entrada:

Pequeño		Mediano		Grande	
Entrada	Salida	Entrada	Salida	Entrada	Salida
000	00	000	00	000	00
100	00	100	00	100	00
000	10	110	00	110	00
010	10	010	01	111	00
000	10	011	01	011	00
001	10	001	01	001	00
000	10	000	01	000	00

Observando las correspondencias entrada-salida vemos que se trata de un Secuencial ya que un mismo vector de entrada se corresponde con más de uno de salida.

II. ELECCIÓN DEL MÉTODO DE SÍNTESIS:

A) Tabla de fases TF
B) Grafo Reducido GR

A) Utilizando Tabla de Fases (TF)

Planteo de la tabla de fases

Consiste en dibujar una tabla donde las columnas son los vectores de entrada y de salida. Se entiende por Fase del Sistema a cada una de las combinaciones posibles de variables de entrada y salida en las que permanece. Las filas indican la evolución de fases del secuencial:

X2X1X0	000	001	010	011	100	101	110	111	Z1Z0
F1	1				2				00
F2	11				2		3		XX
F3			7				3	4	XX
F4				5				4	XX
F5		6		5					XX
F6	1	6							XX
F7			7	8					XX
F8		9		8					XX
F9	10	9							XX
F10	10				2				01
F11	11		12						XX
F12	13		12						XX
F13	13	14							XX
F14	15	14							XX
F15	15				2				1X

Figura 17

- Se ha indicado con flechas la evolución las fases del secuencial para el caso que un producto grande es detectado.
- A fin de lograr un secuencial lo más simple posible, se han marcado con X los casos para los cuales no interesan los valores de salida (ya que suponemos que el producto está transitando por los detectores ópticos). Esto nos permitirá asignar los valores que resulten convenientes más adelante.
- Los cuadros sombreados indican una Fase Estable, los no sombreados una Fase inestable y los vacíos una situación imposible. Por ejemplo: *la F1 es estable para el vector de entrada 000 ya que mientras permanezca en ese valor, el sistema no cambiará de fase*. Si el vector de entrada cambia a 100, la fase F1 es inestable, el sistema cambia a la F2. El 2 no sombreado en la columna 100 está indicando dos cosas, que la F1 es inestable para ese vector de entrada y que el sistema evolucionará hacia la F2 (que es estable para ese vector de entrada).
- En este tipo de descripción tabular es muy posible definir fases redundantes. Para eliminarlas se utilizan dos herramientas:

1) Determinación de fases estables equivalentes

Dos o más fases son equivalentes si:

- Tienen el mismo vector de entrada
- Generan la misma salida
- Si a partir de las mismas se evoluciona, para los mismos vectores de entrada, hacia las mismas fases destino originarias.

2) Determinación de fases estables fusionables

Dos o más fases son fusionables si:

- No son estables para el mismo vector de entrada.
- Evolucionan hacia las fases estables destino para todos los vectores de entrada posibles.

Reducción de la Tabla de Fases

Consiste en determinar las fases equivalente y luego las fusionables

Equivalencias

En la Tabla se observa que las fases F11, F13 y F15 son equivalentes esto es porque tienen el mismo vector de entrada, generan las mismas salidas y es posible a partir de una sola fase evolucionar hacia las mismas fases destino. También las fases F12 y F7

F15,F13 → F11 donde aparezca un 15 o un 13 colocamos un 11
F12 → F7 donde aparezca un 12 colocamos un 7

X2X1X0	000	001	010	011	100	101	110	111	Z1Z0
F1	1				2				00
F2	11				2		3		XX
F3			7				3	4	XX
F4				5				4	XX
F5		6		5					XX
F6	1	6							XX
F7	11		7	8					XX
F8		9		8					XX
F9	10	9							XX
F10	10				2				01
F11	11	14	7		2				1X
F14	11	14							XX

Figura 18

Fusiones

Dos o más fases son fusionables si al juntarlas en una misma fila implican las mismas fases estables e inestables, No es requisito que las salidas coincidan para la misma fila ya que pueden asignarse a la fase correspondiente y al vector de entrada. Si como consecuencia de la fusión, resulta necesario asignar distintos vectores de salida a una misma fila, resultará un Autómata de Mealy. Por lo mencionado, si es nuestra intención obtener un Autómata de Moore, es necesario evitar fusionar las fases que provoquen lo mencionado.

- Se observa que las fases F1, F3, F4, F5 y F6 son fusionables.
- Se observa que las fases F2, F7 y F11 y F14 son fusionables
- Se observa que las fases F8, F9 y F10 son fusionables

X2X1X0	000	001	010	011	100	101	110	111	Z1Z0
F1	1	6	7	5	2		3	4	00
F2	11	14	7	8	2		3		1X
F8	10	9		8	2				01

Figura 19

La figura 5 es la ***Tabla de Fases reducida***, podemos ahora introducir el concepto de **Estado Interno del Secuencial** asignando un estado a cada fase como se indica en la figura 6. Cada estado interno debe asignarse a un vector interno, por lo tanto habrá que codificar variables internas. Esta codificación puede realizarse de distintas formas como veremos más adelante.

Por el momento conviene definir como **Estado Interno Estable** el estado que, para un vector dado de entrada, no genera una transición a otro Estado Interno.

De acuerdo a lo mencionado vemos, por ejemplo en la fig. 6 que el Estado Interno E0 es estable para los vectores de entrada 000, 001, 011, 110 y 111.

X2X1X0	000	001	010	011	100	101	110	111	Z1Z0
E0	1	6	7	5	2		3	4	00
E1	11	14	7	8	2		3		1X
E2	10	9		8	2				01

Figura 20

Obtención del diagrama de flujo

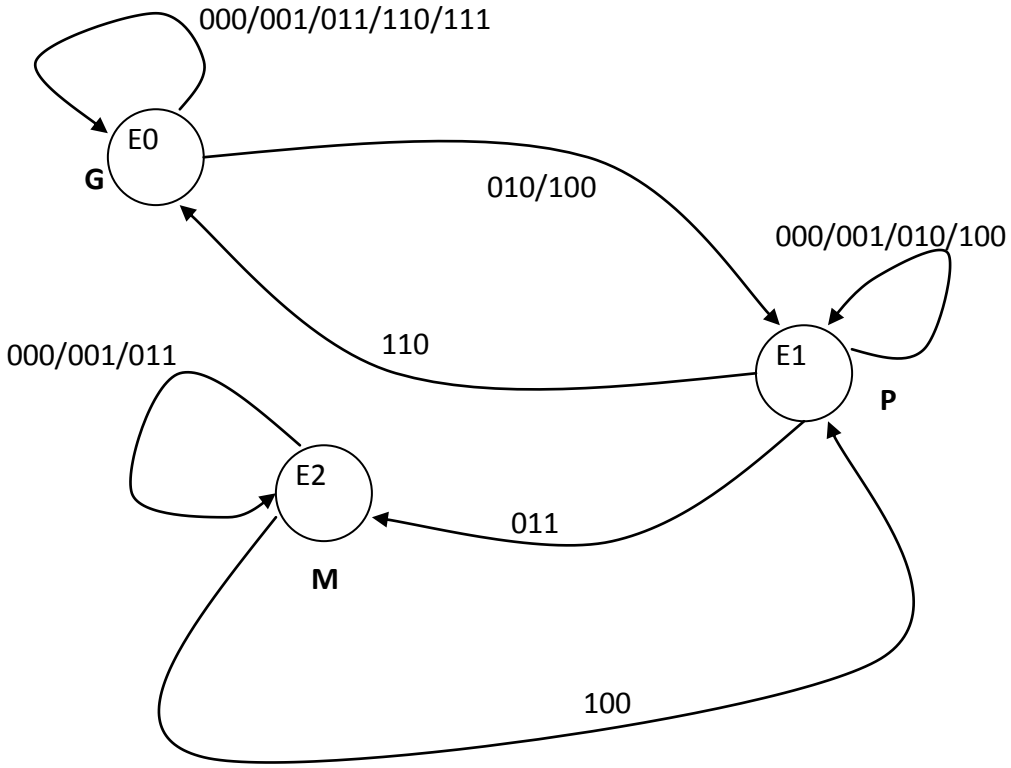


Figura 21

B) Utilizando Grafo Reducido (GR)

Los Grafos Reducidos (GR) consisten en dibujar un círculo y asignárselo al Estado inicial del Secuencial. Detenidos en ese Estado inicial evaluamos la receptividad del mismo. Se dice que un estado es receptivo (receptividad uno) para un determinado vector de entrada si el secuencial debe evolucionar hacia otro estado. En estos términos, un Estado será estable mientras que su receptividad sea cero dado un cierto vector de entrada. Una vez evaluada su receptividad se dibuja otro círculo para el estado siguiente. Este proceso se continúa hasta agotar todas las receptividades.

Si el Sistema a implementar es *asíncrono*, es posible caracterizar el Grafo considerando que los vectores de entrada actúan por **niveles** o por **flancos**.

Si el Sistema a implementar es *síncrono*, sólo es posible caracterizar el Grafo considerando que los vectores de entrada actúan por niveles.

GR - Caracterización por niveles

Debemos definir un estado inicial y evaluar su receptividad a vectores de entrada. Cuando se habla de estado se infiere un vector de salida que estará asociado al estado en cuestión o bien al estado en cuestión y posibles vectores de entrada que no impliquen activar su receptividad. Si este último es el caso se dice que el estado es sensible a uno o más vectores de entrada que no implican activar su receptividad. El concepto de sensibilidad está relacionado con la obtención de un autómata de Moore o de Mealy, el primero no tiene estados sensibles. Siempre es posible obtener cualquiera de ellos.

Vamos a nuestro ejemplo. Podemos definir un estado inicial E0 el cual será receptivo a algún vector de entrada. Para obtener la receptividad veamos la siguiente tabla que indica la evolución de los vectores de entrada para los 3 casos: producto pequeño (P), mediano (M) y grande (G).

P	M	G
100	100	100
000	110	110
010	010	111
000	011	011
001	001	001
000	000	000

Figura 22

Se observa que el estado inicial E0 tendría que ser receptivo a los vectores de entrada 111 (que implica la detección de un producto grande) y a 011 (producto mediano). El vector que aparezca primero definirá el próximo estado. Si no se detecta ninguno de los vectores mencionados se concluye que se trata de un producto pequeño y el secuencial no cambia de estado. Lo mencionado da lugar al siguiente diagrama de flujo:

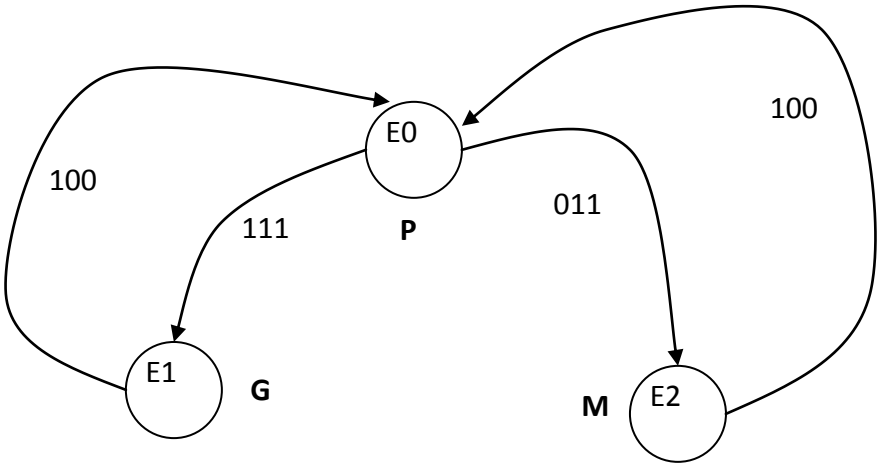


Figura 23

Notemos que el diagrama de flujo resultante de aplicar GR, no posee flechas que retornan al mismo estado (como ocurre con el obtenido a partir de la TF). Esto implica que no está definido el comportamiento del secuencial para todos los vectores de entrada y probablemente resulte un circuito menos sencillo que con el planteo de TF ya que habrá menos cruces en la TVT.

GR - Caracterización por flancos

Cuando el Secuencial a construir es asíncrono, es posible considerar que la receptividad de un estado es uno cuando se produce un flanco (cambio de nivel) en alguna (sólo una) variable de entrada y las demás (o parte de ellas) se mantienen en niveles determinados. Un caso particular sería la situación en la cual no importa el estado de las demás variables de entrada, basta que cambie de nivel (flanco) una de ellas para que el secuencial deba cambiar de estado. Otro caso particular sería aquel en el cual el secuencial deba cambiar de estado ante la ocurrencia de algún vector de entrada determinado sin importar como se llegó a él.

Esto último se parece a la caracterización con niveles, efectivamente la caracterización por flancos permite una caracterización por niveles cuando no hay flancos a considerar.

Implementar un secuencial asíncrono utilizando GR y caracterización por flancos, puede resultar menos complejo que las demás alternativas. Esto es para aquellos casos en los que la receptividad dependa mayoritariamente de los cambios de nivel (flancos) de las variables de entrada.

Se describen a continuación los pasos a seguir para implementar un secuencial asíncrono mediante GR caracterizado por flancos.

La receptividad (Rc) de un estado puede hacerse 1 en los siguientes casos:
Cambio de nivel de una única variable:

1) Rc = xi↑↓
2) Cambio de nivel de una variable y nivel de otras: Rc = xi↑↓.Xi
3) Nivel de variables: Rc = Xi (similar al planteo GR caracterizado por niveles)

Usando estas definiciones se puede caracterizar el diagrama de flujo del ejercicio ejemplo, de la siguiente manera:

Consideramos un Estado inicial que puede ser cuando aún ningún producto ha ingresado, lo llamamos E0 y lo asignamos a un producto mediano.

- Si lo primero que ocurre es que X2 baja y x1 en ese momento está en cero (este es el caso de receptividad que depende de un flanco y la presencia de un vector), entonces lo que está pasando es un producto pequeño (E1), el sistema permanecerá en E1 hasta que se detecte un nuevo producto en la barrera X2, es decir cuando suba X2.j
- Si chequeamos si X0 sube, entonces lo que está pasando es un producto grande (E2), el sistema permanecerá en E2 hasta que se detecte un nuevo producto en la barrera X2, es decir cuando suba X2.
- El paso de un producto mediano (E0) no provocará ningún cambio de estado en el secuencial.

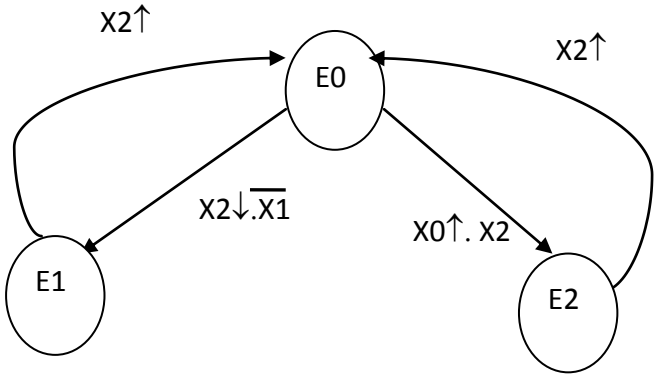


Figura 24

Los estados y las receptividades de cada uno de ellos son:

E0 (para producto mediano) Rc0 = x2↓$\overline{x1}$+ x2.x0↑
E1 (para producto pequeño) Rc1 = x2↑
E2 (para producto grande) Rc0 = x2↑

La codificación de los Estados del secuencial consiste en asignar a cada estado una CELDA ACTIVADA POR FLANCOS (CAF), esta celda estará en 1 si el secuencial está en el estado correspondiente, por lo tanto las demás deberán estar en cero. Cuando el secuencial cambia de

estado, se activará la celda que le corresponda y se desactivará la celda anterior. Los cambios de estado se producen cuando se presenta una receptividad distinta de cero.

III. CODIFICACIÓN DE ESTADOS INTERNOS:

La codificación de los estados internos determinará la cantidad de variables internas y su valor para cada estado interno. La codificación dependerá de que clase de secuencial implementaremos: asíncrono (caracterizado por niveles o por flancos) o síncrono.

Secuencial Asíncrono caracterizado por niveles - Codificación

Existen dos formas de codificar estados de un secuencial:

- Asignar una variable interna a cada estado interno
- Asignar más de una variable a cada estado interno

Si asignamos una variable interna a cada estado, todas las transiciones de Vectores Internos serán no adyacentes teniendo que analizar cada transición a fin de determinar si es crítica o no.

Si asignamos más de una variable interna a cada estado (m variables internas de tal forma que 2^m >= n, siendo n la cantidad de estados) pueden resultar transiciones no adyacentes críticas no evitables modificando la codificación.

Siempre es necesario verificar que las transiciones no adyacentes de las variables internas no provoquen evoluciones indeseadas del secuencial. Esto es porque si el vector interno tiene que cambiar a un valor no adyacente, pasará por valores intermedios que se convierten en nuevas entradas al combinacional realimentado, existiendo el riesgo que, ante estos valores no tenidos en cuenta al determinar las funciones internas, se llegue a un estado no previsto o el secuencial fluctúe entre estados.

A estas transiciones no adyacentes de los vectores internos se las llama Transiciones no Adyacentes Críticas (TNAC)

Para evitar las TNAC lo primero que debe hacerse es, en el momento de la codificación de los estados internos, evitar las no adyacencias. Por ejemplo, sea el diagrama de flujo obtenido con GR caracterizado por niveles siguiente:

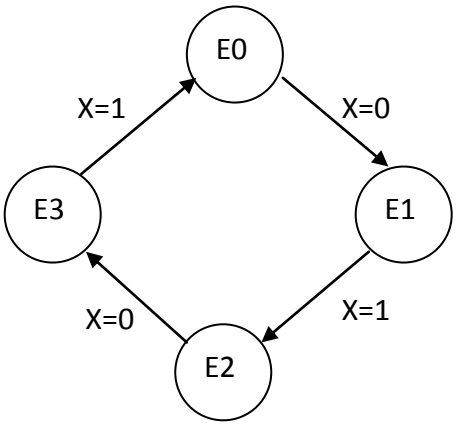


Figura 25

Elegimos la codificación 2^n = m, por lo tanto necesitamos 2 variables internas (y0 e y1).

Para evitar las TNAC es útil usar la siguiente tabla:

Y0 Y1	0	1
0	E0 →	E1 ↓
1	E3 ↑	E2 ←

Figura 26

Se ve que para este caso es posible elegir una codificación que evite las no adyacencias.

Supongamos ahora el siguiente diagrama de flujo con su TVT:

X1	X0	Y1	Y0	Y1	Y0
0	0	0	0	0	1
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	1	1	1
0	1	1	0	1	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	1	0	1
1	0	1	0	1	0
1	**0**	**1**	**1**	**0**	**0**
1	1	0	0	0	0
1	1	0	1	0	1
1	1	1	0	0	0
1	1	1	1	1	1

	Y1	Y0
E0	0	0
E1	0	1
E2	1	1
E3	1	0

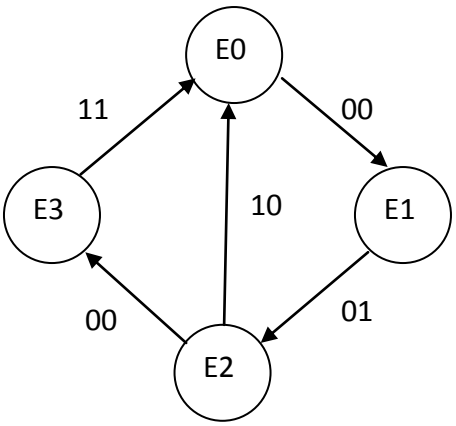


Figura 27

Se ve que no es posible codificar con dos variables internas y evitar transiciones no adyacentes, conviene verificar que las transiciones no adyacentes sean críticas.

Se observa en el diagrama que la transición de E2 (11) a E0 (00) que ocurre cuando el vector de entrada es 10, no es adyacente, será crítica?. Cuando las variables internas pasan de 11 a 00, pueden adoptar valores intermedios 10 ó 01. Si pasan primero por 10 el secuencial pasa al estado E3 el cual para el vector de entrada 10 es estable, es decir: el secuencial es posible que se quede en E3 al y no llegue a E0 que es el estado deseado. Para analizar en detalle la TNA vamos a considerar que el SC tiene un tiempo de propagación (Tsc), además suponemos que

cuando las salidas y0 e y1 cambian de forma no adyacente, por ejemplo de 11 a 00 (transición de E2 a E0) se requiere un tiempo (Ta) adicional para que alcancen 00 desde su estado intermedio 01 o 10. En la siguiente tabla el tiempo transcurre de arriba hacia abajo, como Tcs > Ta, supongamos que a Tsc le corresponden 4 renglones y a Ta 1 renglón. Asumimos que el secuencial está en el estado E2 (11) y el vector de entrada toma el valor 10.

	X1	X0	Y1	Y0
1	1	0	1	1
2				
3				
4				
5	1	0	1	0
6	1	0	0	0
7				
8				
9	1	0	1	0
10	1	0	0	0
11				
12				
13	1	0	1	0
14	1	0	0	0
15				
16				
17	1	0	1	0
18	1	0	0	0
19				

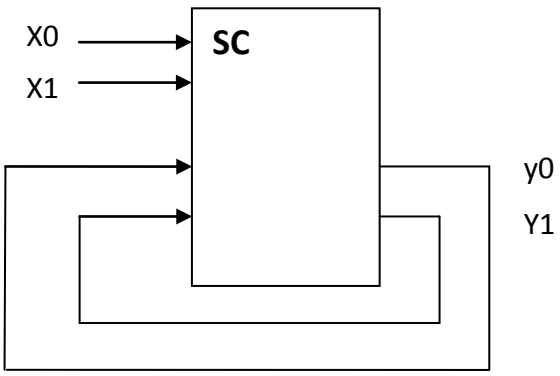


Figura 28

En la fila 1 el secuencial está en el estado E2 (11) y el vector de entrada adopta el valor 10.

En la fila 5 se indica los valores de las entradas al SC suponiendo que primero ha cambiado y0 (pasó de 1 a 0) y después de un tiempo, fila 6, Ta cambia también y1 (pasó de 1 a 0). La flecha desde la fila 1 a la 5 indica el tiempo Tsc.

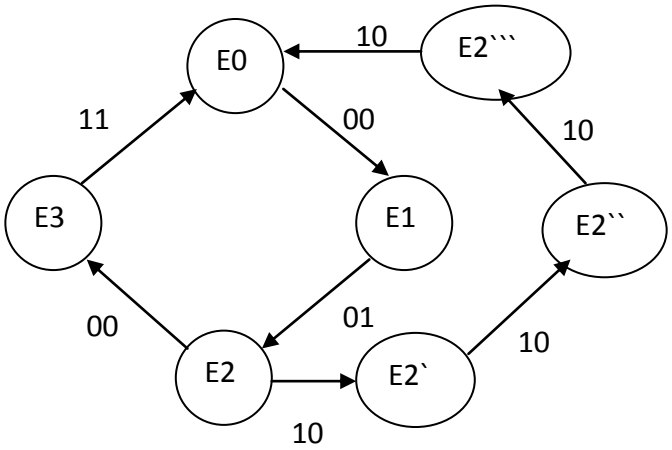
Las entradas al SC de la fila 5 provocan, después de Tsc, la salida indicada en la fila 9 y las de la fila 6 la 10.

Asumiendo Tsc similares para las entradas de la fila 5 y 6 (suposición que no necesariamente es cierta), el secuencial fluctuará entre E0 y E2 permanentemente mientras se mantenga invariable el vector de entrada. Este no es el comportamiento deseado, por lo tanto debemos considerar a la TNA como TNAC.

En esta situación se pueden plantear cinco alternativas:

1) Cambiar las codificaciones y analizar las no adyacencias para cada caso. Si en alguno no es crítica el problema está resuelto.
2) Agregar variables internas y nuevos estados de forma tal que la transición de E2 a E0 se realice por estados adyacentes. Por ejemplo, si agregamos una variable interna sería posible realizar una codificación como la siguiente:

Se ve que la transición TNAC de E2 a E0 se realiza a través de los estados E2', E2'' y E2''' que son adyacentes entre sí, resultando el siguiente diagrama de flujo modificado:



y1y0 / y2	0 0	0 1	1 1	1 0
0	E0	E1	E2	E3
1	E2'''	E2''	E2'	

Figura 29

Se ha salvado la TNAC pero se han agregado estados: el secuencial es más complejo.

Otra forma de encontrar la codificación adecuada agregando estados es utilizar los cubos N. Supongamos el siguiente diagrama de fluyo de cuatro estados y que son posibles las transiciones entre todos ellos, como se indica en la siguiente figura;

	Y2y1y0	Y2y1y0
E0/E2´	0 0 0	1 0 0
E1/E3´	0 0 1	1 0 1
E2/E0´	0 1 1	1 1 1
E3/E1´	0 1 0	1 1 0

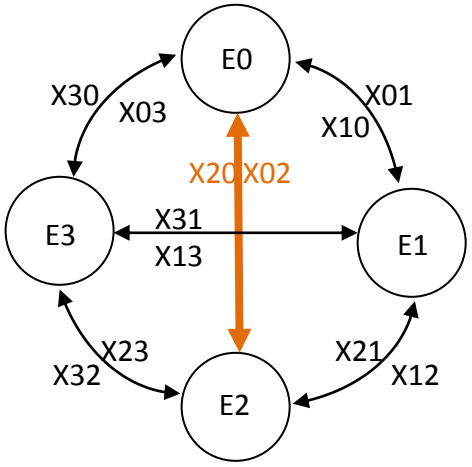


Figura 30

Se observa que si la transición es de E0 a E2, o de E1 a E3, se producirán no adyacencias. El método consiste en dibujar un cubo N con código cíclico y contínuo en una de sus caras, por ejemplo en la cara superior como muestra la figura y asignar en esa cara los cuatro estados del diagrama (E0 a E3). A la cara opuesta, es decir la inferior,

se le asignan estados de forma tal que se cubran las no adyacencias, ver tabla a la derecha del diagrama de flujo. Entonces, si el secuencial debe pasar de E0 a E2 (estados no adyacentes), se lo hace pasar a E2` y empezará a trabajar en la cara inferior hasta que tenga que sortear una no adyacencia en esa cara, situación en la cual volverá a la cara superior. Es rojo se indica cómo se resuelven las transiciones E0 a E2 (en realidad pasa a E2´, de E2 a E3 (en realidad de E2´a E3´) y de E3 a E1 (en realidad de E3´a E1).

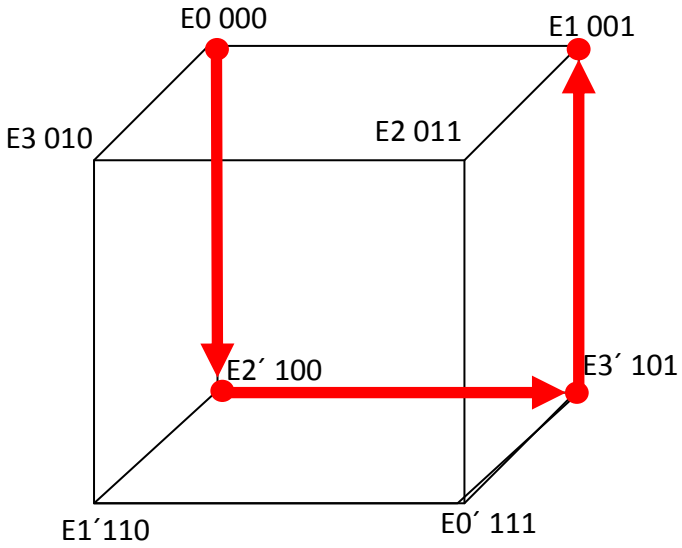


Figura 31

En la figura de abajo se indica el diagrama de flujo resultante, y las transiciones puestas en juego (indicadas en naranja) cuando el secuencial debe cambiar de E0 a E2 y viceversa

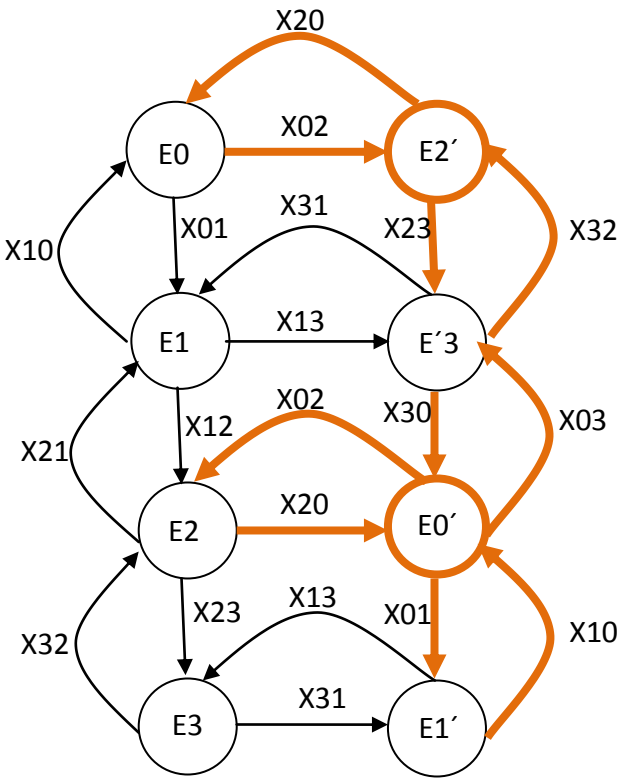


Figura 32

A) Codificación para el SSA resultante del diagrama de flujo obtenido por TF
Codificación asignando m variables internas 2^m >= n estados
n = 3 → m = 2

	Y1	Y0
E0	0	0
E1	0	1
E2	1	1

Figura 33

Esta codificación no presenta transiciones no adyacentes

Codificación asignando una variable interna a cada estado

	Y2	Y1	Y0
E0	0	0	1
E1	0	1	0
E2	1	0	0

Figura 34

Las transiciones no adyacentes no son críticas para nuestro ejemplo, supongamos una transición entre E0 y E1, el vector interno pasaría de 001 a 010 por el valor intermedio 011 ó 000. Si pasa por 011 el sistema quedaría momentáneamente en dos estados a la vez (E0 y E1), en esta situación si el vector de entrada pasa a 000, el secuencial permanecerá en los dos estados mencionados. Que el vector de entrada pase a 000 en la situación mencionada, antes de que el sistema se acomode en el estado E1 no es posible debido a la velocidad de la cinta transportadora. Por lo tanto, si bien el secuencial se encontrará en dos estados simultáneamente, será por un tiempo no significativo y llegará correctamente al estado E1. Las mismas consideraciones pueden hacerse para todos los casos.

B) Para el diagrama de flujo obtenido por GR (caracterización por niveles)

Codificación asignando m variables internas (2^m >= n) para n estados
n = 3 → m = 2

	Y1	Y0
E0	0	0
E1	0	1
E2	1	0

Figura 35

Codificación asignando 1 variable interna a cada estado

	Y2	Y1	Y0
E0	0	0	1
E1	0	1	0
E2	1	0	0

Figura 36

Secuencial Asíncrono caracterizado por flancos

La codificación de los Estados del secuencial consiste en asignar a cada estado una CELDA ACTIVADA POR FLANCOS (CAF), esta celda estará en 1 si el secuencial está en el estado correspondiente, por lo tanto las demás deberán estar en cero. Cuando el secuencial cambia de estado, se activará la celda que le corresponda y se desactivará la celda anterior. Los cambios de estado se producen cuando se presenta una receptividad distinta de cero.

El diagrama en bloques de una CAF es el siguiente:

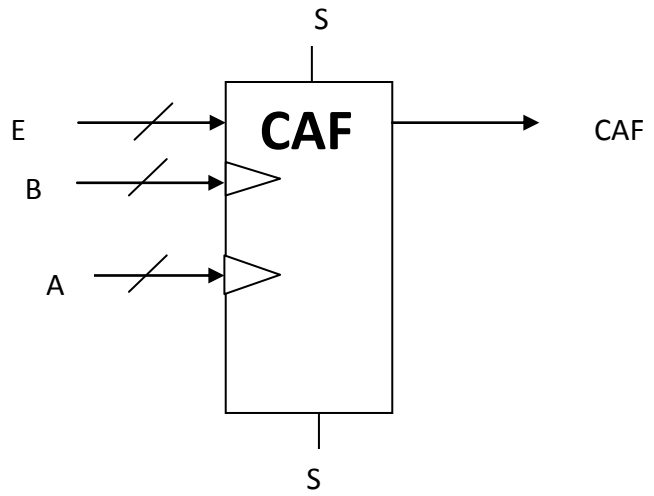


Figura 37

Donde:
A: un flanco de subida en A activa la CAF siempre que E esté en 1
B: un flanco de subida en B borra la CAF
E: un 1 permite que un flanco en A active la celda

La codificación consistirá será entonces:

	CAF2	CAF1	CAF0
E0	0	0	1
E1	0	1	0
E2	1	0	0

Figura 38

Las entradas de puesta a cero y puesta a uno de las CAFs (no conectadas en la figura) se utilizan para asegurar que al iniciar el sistema, sólo la CAF0 este en 1.

Secuencial Síncrono caracterizado por niveles - Codificación

En este caso no es necesario evaluar las TNAC, por lo tanto la codificación es libre.

Codificación asignando m variables internas 2^m >= n estados

n = 3 → m = 2

	Y1	Y0
E0	0	0
E1	0	1
E2	1	1

Figura 39

Codificación asignando una variable interna a cada estado

	Y2	Y1	Y0
E0	0	0	1
E1	0	1	0
E2	1	0	0

Figura 40

IV. OBTENCIÓN DE LA TABLA DE VERDAD TEMPORAL (TVT)

La TVT es una tabla donde a la izquierda encolumnamos los Vectores de Entrada y los Vectores Internos en un instante *t*. A la derecha los Vectores Internos en un instante posterior *t+1*

La TVT obtenida nos servirá para implementar un SSS o un SSA indistintamente. En el caso de los SSS el instante *t* y el *t+1* están separados por el franco del reloj de sincronismo.

Cuando el diagrama de flujo ha sido obtenido a partir de la tabla de fases (TF) o mediante Grafo Reducido, la TVT se plantea partiendo del diagrama de flujo. Nos ubicamos en un estado del mismo y analizamos la evolución del secuencial para los distintos vectores de entrada, así hasta agotar los estados. Si quedan codificaciones no usadas, están se llenan con Xs ya que el secuencial nunca las alcanzará.

A continuación planteamos la TVT para cada caso y cada codificación.

TVT desde TF

T					T+1	
X2	X1	X0	Y1	Y0	Y1	Y0
0	0	0	0	0	0	0
0	0	0	0	1	0	1
0	0	0	1	0	X	X
0	0	0	1	1	1	1
0	0	1	0	0	0	0
0	0	1	0	1	0	1
0	0	1	1	0	X	X
0	0	1	1	1	1	1
0	1	0	0	0	0	1
0	1	0	0	1	0	1
0	1	0	1	0	X	X
0	1	0	1	1	X	X
0	1	1	0	0	0	0
0	1	1	0	1	1	1
0	1	1	1	0	X	X
0	1	1	1	1	1	1
1	0	0	0	0	0	1
1	0	0	0	1	0	1
1	0	0	1	0	X	X
1	0	0	1	1	0	1
1	0	1	0	0	X	X
1	0	1	0	1	X	X
1	0	1	1	0	X	X
1	0	1	1	1	X	X
1	1	0	0	0	0	0
1	1	0	0	1	0	1
1	1	0	1	0	X	X
1	1	0	1	1	X	X
1	1	1	0	0	0	0
1	1	1	0	1	X	X
1	1	1	1	0	X	X
1	1	1	1	1	X	X

Figura 41

TVT desde GR (por nivel)

T					T+1	
X2	X1	X0	Y1	Y0	Y1	Y0
0	0	0	0	0	0	0
0	0	0	0	1	0	1
0	0	0	1	0	1	0
0	0	0	1	1	X	X
0	0	1	0	0	0	0
0	0	1	0	1	0	1
0	0	1	1	0	1	0
0	0	1	1	1	X	X
0	1	0	0	0	0	0
0	1	0	0	1	0	1
0	1	0	1	0	1	0
0	1	0	1	1	X	X
0	1	1	0	0	1	0
0	1	1	0	1	0	1
0	1	1	1	0	0	1
0	1	1	1	1	X	X
1	0	0	0	0	0	0
1	0	0	0	1	0	0
1	0	0	1	0	0	0
1	0	0	1	1	X	X
1	0	1	0	0	0	0
1	0	1	0	1	0	1
1	0	1	1	0	1	0
1	0	1	1	1	X	X
1	1	0	0	0	0	0
1	1	0	0	1	0	1
1	1	0	1	0	1	0
1	1	0	1	1	X	X
1	1	1	0	0	0	1
1	1	1	0	1	1	1
1	1	1	1	0	1	0
1	1	1	1	1	X	X

Figura 42

TVT desde GR (por flancos)

	T					T+1		
X2	X1	X0	CAF2	CAF1	CAF0	CAF2	CAF1	CAF0
↓	0	x	0	0	1	0	1	0
X	x	↓	0	0	1	1	0	0
↑	x	X	0	1	0	0	0	1
↑	x	x	1	0	0	0	0	1

Figura 43

V. OBTENCIÓN DE LAS FUNCIONES LÓGICAS INTERNAS

En este punto debemos hacer la diferencia de SSA y SSS.

Sistema Secuencial Asíncrono (SSA) – Por nivel

Se proponen dos maneras de obtener las funciones lógicas de las variables internas en un SSA:

- Utilizando combinacionales SSI y/o MSI.
- Utilizando combinacionales SSI y/o MSI y bietables SR asíncronos.
- Combinacional Programable

Utilizando SSI y/o MSI.

Las funciones a implementar, tomando como ejemplo la TVT obtenida a partir de la TF, son:

$Y0 = S_5\,(1,3,5,7,8,9,13,15,16,17,19,25) + Q_5\,(2,6,10,11,14,18,20,21,22,23,26,27,29,30,31)$

$Y1 = S_5\,(3,7,13,15) + Q_5\,(2,6,10,11,14,18,20,21,22,23,26,27,29,30,31)$

Estas funciones pueden implementarse minimizando con el método de multifunciones (SSI), o bien usando decodificadores o multiplexores (MSI).

Obtenemos entonces el siguiente combinacional realimentado:

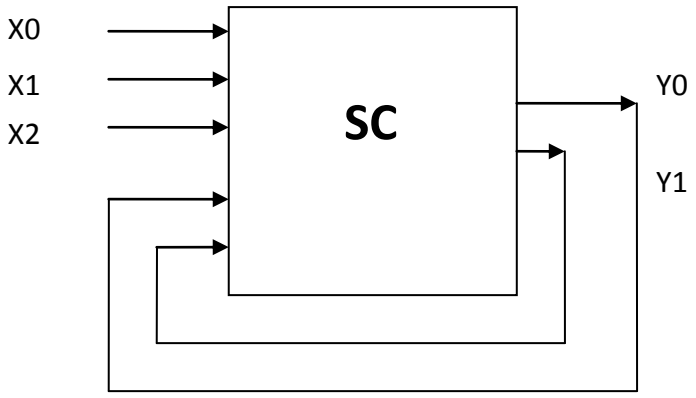


Figura 44

Utilizando SR asíncronos

Consiste en asignar un biestable SR a cada variable interna. La programación de las entradas del biestable se realiza combinado la tabla de verdad del biestable con la TVT. Así, las distintas programaciones de Ri y Si, serán:

Condición	R	S
Mantener un cero	X	0
Poner un cero	1	0
Mantener un	0	X

uno		
Poner un uno	0	1

Figura 45

En la fig. 18 aparece la programación correspondiente a la TVT a partir de la TF, a la que se agregan las columnas correspondientes a R0, S0, R1 y S1.

Las funciones lógicas son:

$R0 = Q_5 (0,3,4,6,10,11,12,14,18,20,21,22,23,24,26,27,28,29,30,31)$

$R1 = S_5 (19) + Q_5 (0,1,2,4,5,6,8,9,10,11,12,14,16,17,18,20,21,22,23,24,25,26,27,28,29,30,31)$

$S0 = S_5 (8,16) + Q_5 (1,2,3,5,6,7,9,10,11,13,14,15,17,18,19,20,21,22,23,25,26,27,29,30,31)$

$S1 = S_5 (14) + Q_5 (2,3,6,7,10,11,14,15,18,20,21,22,23,26,27,29,30,31)$

Estas funciones pueden implementarse minimizando con el método de multifunciones (SSI), o bien usando decodificadores o multiplexores (MSI).

T					T+1					
X2	X1	X0	Y1	Y0	Y1	Y0	R1	S1	R0	S0
0	0	0	0	0	0	0	X	0	X	0
0	0	0	0	1	0	1	X	0	0	X
0	0	0	1	0	X	X	X	X	X	X
0	0	0	1	1	1	1	0	X	0	X
0	0	1	0	0	0	0	X	0	X	0
0	0	1	0	1	0	1	X	0	0	X
0	0	1	1	0	X	X	X	X	X	X
0	0	1	1	1	1	1	0	X	0	X
0	1	0	0	0	0	1	X	0	0	1
0	1	0	0	1	0	1	X	0	0	X
0	1	0	1	0	X	X	X	X	X	X
0	1	0	1	1	X	X	X	X	X	X
0	1	1	0	0	0	0	X	0	X	0
0	1	1	0	1	1	1	0	1	0	X
0	1	1	1	0	X	X	X	X	X	X
0	1	1	1	1	1	1	0	X	0	X
1	0	0	0	0	0	1	X	0	0	1
1	0	0	0	1	0	1	X	0	0	X
1	0	0	1	0	X	X	X	X	X	X
1	0	0	1	1	0	1	1	0	0	X
1	0	1	0	0	X	X	X	X	X	X
1	0	1	0	1	X	X	X	X	X	X
1	0	1	1	0	X	X	X	X	X	X
1	0	1	1	1	X	X	X	X	X	X
1	1	0	0	0	0	0	X	0	X	0
1	1	0	0	1	0	1	X	0	0	X
1	1	0	1	0	X	X	X	X	X	X
1	1	0	1	1	X	X	X	X	X	X
1	1	1	0	0	0	0	X	0	X	0
1	1	1	0	1	X	X	X	X	X	X

1	1	1	1	0	X	X	X	X	X	X
1	1	1	1	1	X	X	X	X	X	X

Figura 46

Obsérvese que el combinacional resultará de menor complejidad que en el caso anterior. El circuito es el siguiente:

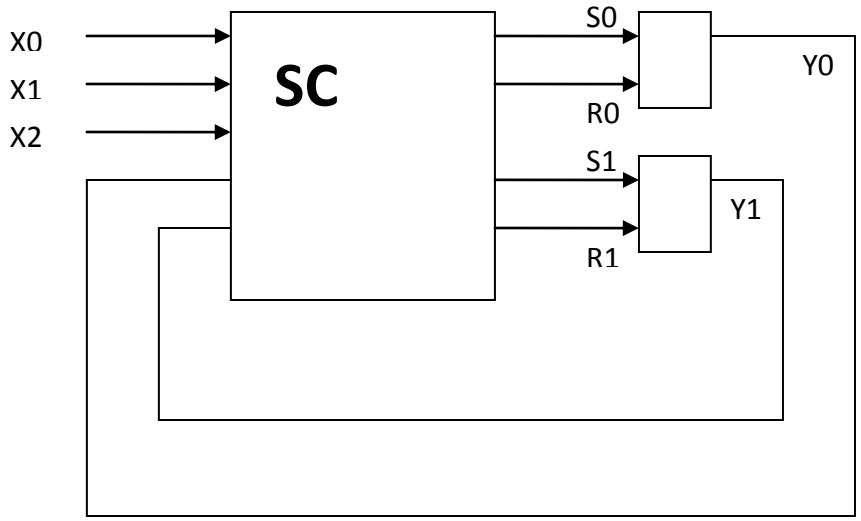


Figura 47

Consideración: Cuando se alimenta el circuito, los biestables pueden adoptar cualquier valor. Es necesario entonces verificar si el secuencial evoluciona correctamente sea cual sea el estado inicial de los biestables. Si la evolución no es correcta habrá que replantear el secuencial, o bien agregar entradas de puesta a uno y puesta a cero a cada biestable. Verifiquemos para nuestro caso:

- Si al alimentar, los biestables adoptan el valor 00, 01 o 11, no hay problema puesto que son estados posibles y el secuencial evolucionará de acuerdo al diagrama de flujo de la fig.7.
- Si al alimentar, los biestables adoptan el valor 10, debemos verificar qué Minterms marcados como X en la TVT han sido considerados como existentes en la implementación de la función. Si en todos los casos el secuencial evoluciona como es deseado, el problema está resuelto, sino debe asignarse un 1 ó 0 a la X para evitar evoluciones no deseadas.

Sistema Secuencial Asíncrono (SSA) – Por Flanco

	T					T+1		
X2	X1	X0	CAF2	CAF1	CAF0	CAF2	CAF1	CAF0
↓	0	x	0	0	1	0	1	0
X	x	↑	0	0	1	1	0	0
↑	x	X	0	1	0	0	0	1
↑	x	x	1	0	0	0	0	1

Figura 48

Las funciones lógicas que permiten intercotectar las CAF son:

E0 = CAF1 + CAF2 ; es la suma de las CAF anteriores multiplicada por el vector.
A0 = x2 ; es la suma de entradas que actúan por flanco.
B0 = CAF1 + CAF2; es la sumatoria de las CAF hacia donde va.

E1 = CAF0 x1´
A1 = x2´
B1 = CAF0

E2 = CAF0
A2 = x0´
B2 = CAF0

El circuito resultante es:

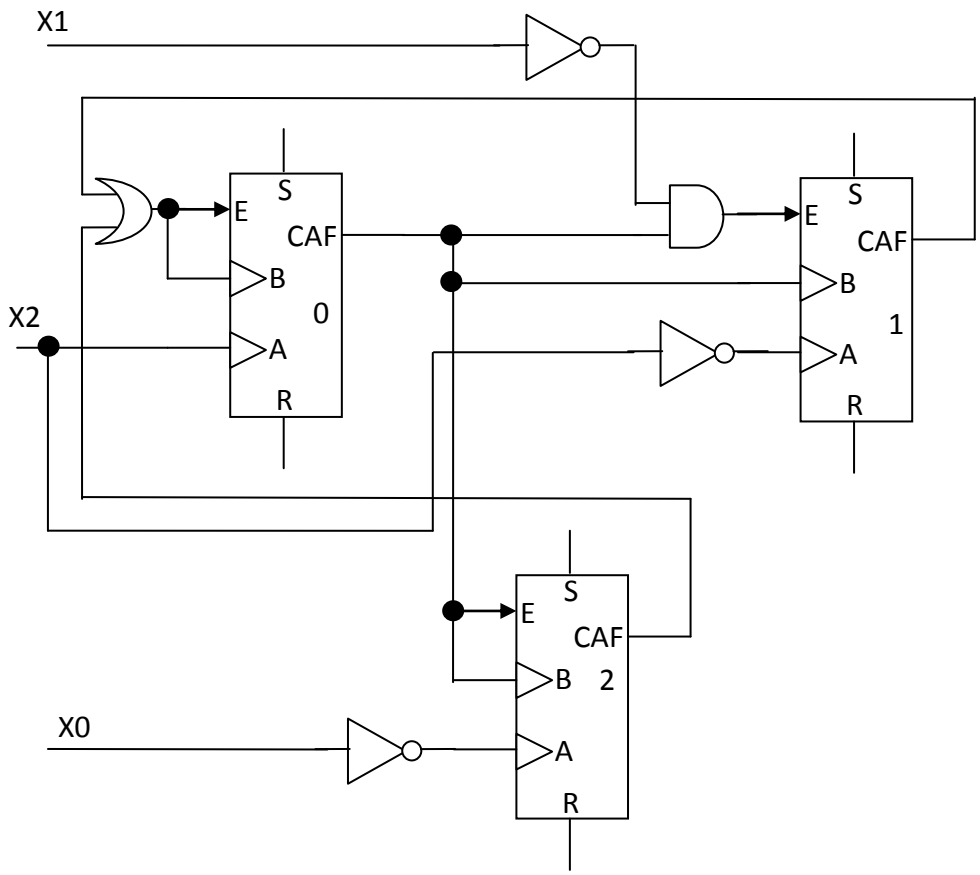


Figura 49

Sistema Secuencial Síncrono (SSS)

En un SSS, como vimos anteriormente, se sincronizan las variables internas con una señal de sincronismo externa llamada reloj del sistema. El secuencial evolucionará sincrónicamente con el reloj. A fin de eliminar el problema (propio de los SSA) del las transiciones no adyacentes críticas de los vectores internos, se usan biestables síncronos por flancos.

En nuestro ejemplo usaremos un biestable para cada variable interna con la codificación sugerida en el punto 3.

Se proponen tres maneras de obtener las funciones lógicas de las variables internas para un SSS:

- Utilizando combinacionales SSI y/o MSI y bietables SR.
- Utilizando combinacionales SSI y/o MSI y biestables JK.
- Utilizando combinacionales SSI y/o MSI y bietables D (Registro)

El procedimiento es similar para los distintos casos. Elegimos un bietable SR, cuya tabla de verdad es:

R	S	Qt+1
0	0	Qt
0	1	1
1	0	0
1	1	X

Figura 50

Para obtener las funciones lógicas que programan las entradas S0, R0, S1 y R1, se procede de las misma forma que lo hicimos en los SSA y los biestables SR.

Aplicando estos conceptos, resulta la tabla de la fig. 51, que es igual a la de la fig. 50.

T					T+1					
X2	X1	X0	Y1	Y0	Y1	Y0	R1	S1	R0	K0
0	0	0	0	0	0	0	X	0	X	0
0	0	0	0	1	0	1	X	0	0	X
0	0	0	1	0	X	X	X	X	X	X
0	0	0	1	1	1	1	0	X	0	X
0	0	1	0	0	0	0	X	0	X	0
0	0	1	0	1	0	1	X	0	0	X
0	0	1	1	0	X	X	X	X	X	X
0	0	1	1	1	1	1	0	X	0	X
0	1	0	0	0	0	1	X	0	0	1
0	1	0	0	1	0	1	X	0	0	X
0	1	0	1	0	X	X	X	X	X	X
0	1	0	1	1	X	X	X	X	X	X
0	1	1	0	0	0	0	X	0	X	0
0	1	1	0	1	1	1	0	1	0	X
0	1	1	1	0	X	X	X	X	X	X
0	1	1	1	1	1	1	0	X	0	X
1	0	0	0	0	0	1	X	0	0	1
1	0	0	0	1	0	1	X	0	0	X
1	0	0	1	0	X	X	X	X	X	X
1	0	0	1	1	0	1	1	0	0	X
1	0	1	0	0	X	X	X	X	X	X
1	0	1	0	1	X	X	X	X	X	X
1	0	1	1	0	X	X	X	X	X	X
1	0	1	1	1	X	X	X	X	X	X
1	1	0	0	0	0	0	X	0	X	0
1	1	0	0	1	0	1	X	0	0	X
1	1	0	1	0	X	X	X	X	X	X
1	1	0	1	1	X	X	X	X	X	X
1	1	1	0	0	0	0	X	0	X	0
1	1	1	0	1	X	X	X	X	X	X
1	1	1	1	0	X	X	X	X	X	X
1	1	1	1	1	X	X	X	X	X	X

Figura 51

El circuito es el de la figura 52. Para obtener el combinacional se puede usar SSI para lo cual minimizamos las funciones por el método de multifunciones, o bien MSI. Observemos que el circuito posee una entrada de inicialización I que permite ubicar al secuencial en el estado

inicial, para este caso E0. Aprovechamos para ello la entrada de puesta a cero asíncronas de los biestables.

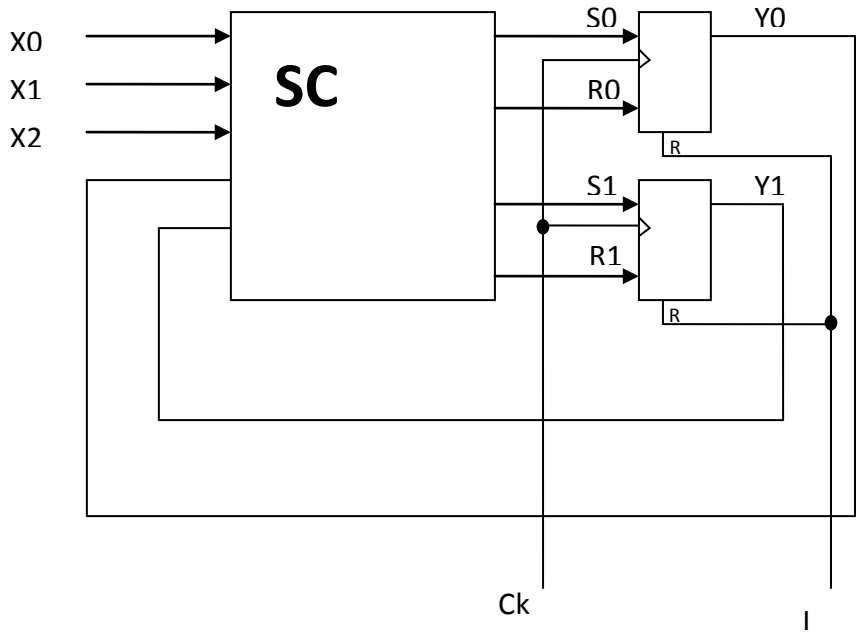


Figura 52

VI. OBTENCIÓN DE LAS FUNCIONES LÓGICAS DE SALIDA

En las figuras 7, 9 y 11 se observa que a cada estado interno del secuencial corresponde un vector de salida, por lo tanto se trata de Autómatas de Moore. Para el caso del diagrama de flujo originado por TF, tenemos las siguientes salidas:

	Y1	Y0	Z1	Z0
E0	0	0	1	X
E1	0	1	0	0
NO ASIGNADO	1	0	X	X
E2	1	1	0	1

Figura 53

$Z1 = S_2\,(0) + Q_2\,(2)$

$Z0 = S_2\,(3) + Q_2\,(2)$

Obtenemos el siguiente diagrama:

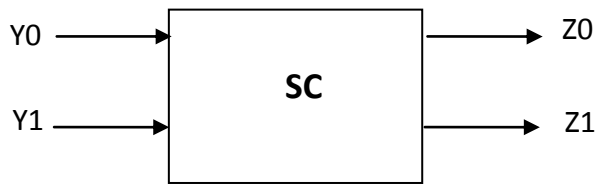


Figura 54

DISEÑO HÍBRIDO DE SECUENCIALES ASíNCRONOS CARACTERIZADOS POR NIVELES

Existe una alternativa para diseñar SSA por niveles que evita el problema de las TNAC siempre y cuando la distancia entre los Vectores internos no adyacentes no sea superior a 2.

También es posible su aplicación para distancias mayores pero es necesario verificar cada a de ellas lo que torna poco práctico el método. No obstante generalmente es posible limitar las TNA a 2.

El método se basa en el siguiente diagrama en bloques:

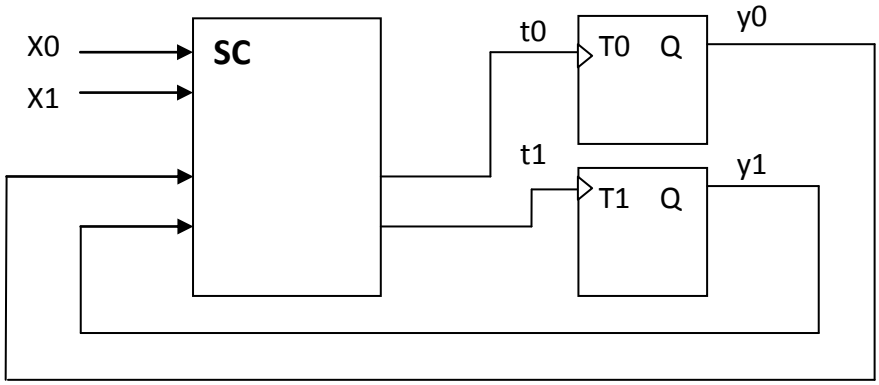


Figura 55

Los biestables son T asíncronos caracterizados por flancos (nótese que un T asíncrono por flancos puede implementarse con un JK haciendo J = K = 1 y el Ck = T).

X1	X0	Y1	Y0	Y1	Y0	T1	T0
0	0	0	0	0	1	NC	C
0	0	0	1	0	1	NC	NC
0	0	1	0	1	0	NC	NC
0	0	1	1	1	0	NC	C
0	1	0	0	0	0	NC	C
0	1	0	1	1	1	C	NC
0	1	1	0	1	0	NC	NC
0	1	1	1	1	1	NC	NC
1	0	0	0	0	0	NC	NC
1	0	0	1	0	1	NC	NC
1	0	1	0	1	0	NC	NC
1	**0**	**1**	**1**	**0**	**0**	**C**	**C**
1	1	0	0	0	0	NC	NC
1	1	0	1	0	1	NC	NC
1	1	1	0	0	0	C	NC
1	1	1	1	1	1	NC	NC

Supongamos que se trata del mismo ejemplo anterior. A la TVT se le agregan dos columnas para t1 y t0:

NC: el biestable no tiene que cambiar

C: el biestable tiene que cambiar.

Como t1 y t0 son salidas del SC, asignamos un 0 cuando está NC y un 1 cuando está C, de esta forma cuando, por ejemplo el secuencial está en el estado E0 (00) y el vector de entrada adopta el valor 00 (primera fila de la tabla) t1 =0 y t0 = 1 provocando que el biestable T0 cambie de estado.

Figura 56

A fin de visualizar el comportamiento de este circuito para una TNA, analicemos la fila marcada en rojo en la TVT que implica un cambio no adyacente de las variables internas. Ahora podemos considerar que existen tres tiempos en juego, el tiempo de propagación del SC (Tsc), el tiempo de propagación de cada biestable (Tb) y el tiempo adicional que requiere el SC para que sus salidas T0 y T1 alcancen 11 (esto es porque t0 y t1 deben pasar de 11 a 00 necesariamente por el valor intermedio 01 o 10). En la siguiente tabla consideramos que el tiempo transcurre hacia abajo y asignamos a Tsc y Tb cuatro filas. A Ta le asignamos una fila. Veamos entonces qué pasa en el secuencial para la TNA mencionada.

Fila 1: E2 con entrada 11: Estable t1t0 = 00

Fila 2: E2 con entrada 10: Inestable t1t0 deben cambiar a 11, supongamos que primero cambia t0

	X1	X0	Y1	Y0	T1	T0
1	1	1	1	1	0	0
2	1	0	1	1	0	0
3						
4						
5						
6	1	0	1	1	0	1
7	1	0	1	1	1	1
8						
9						
10	1	0	1	0	1	1
11	1	0	0	0	1	1
12						
13						
14	1	0	0	0	1	0
15	1	0	0	0	0	0
16						
17						
18	1	0	0	0	0	0
19	1	0	0	0	0	0
20						
21						

Fila 6: Transcurrido Tsc cambia t0

Fila 7: Transcurrido Tsc y Ta cambia t1

Fila 10: Transcurrido Tb cambia y0

Fila 11: Transcurrido Tb y Ta cambia y1

Fila 14: Transcurrido Tsc t1 y t0 deben cambiar en función de las entradas al SC en el momento de la fila 10, según la TVT deberían pasar a 00, nuevamente suponemos que cambia primero t0.

Fila 15: Transcurrido Tsc t1 y t0 deben cambiar en función de las entradas al en el momento de la fila 11, según la TVT deberían pasar a 00. Además t1 debe pasar a 0 ya que el SC viene de la no adyacencia provocada en la fila 10.

Fila 18: Transcurrido Tsc t1 y t0 deben cambiar en función de las entradas al SC en el momento de la fila 14, según la TVT deberían pasar a 00.

Fila 19: Transcurrido Tsc t1 y t0 deben cambiar en función de las entradas al SC en el momento de la fila 15, según la TVT deberían pasar a 00.

Según se puede ver, el secuencial superó correctamente la TNA.

En la fila 15 es válido afirmar que t0 nunca deberá tener un valor distinto de 0 ya que el estado E0 es estable para el vector de entrada 10, si así no fuera implica un error en la TVT, es decir en el planteo del ejercicio.

Si la distancia entre las TNA es mayor a 2 puede resultar que las salidas del SC fluctúen entre 0 y 1 antes de volver a cero una vez pasado el transitorio. Estos flancos no deseados pueden llevar al secuencial a un estado no deseado o a inestabilidades. Si la distancia en 2, nunca puede aparecer un flanco no deseado a las salidas del SC ya que no hay posibilidad que, una salida del SC que pasó a 1 y luego a 0, vuelva a uno. Recordemos que un estado en un SSA siempre es estable para el vector de entrada que provocó la transición hacia él.

SISTEMAS SECUENCIALES SINCRONOS DE CONTROL

Existe una forma de plantear la implementación de SSS basándose en el siguiente esquema:

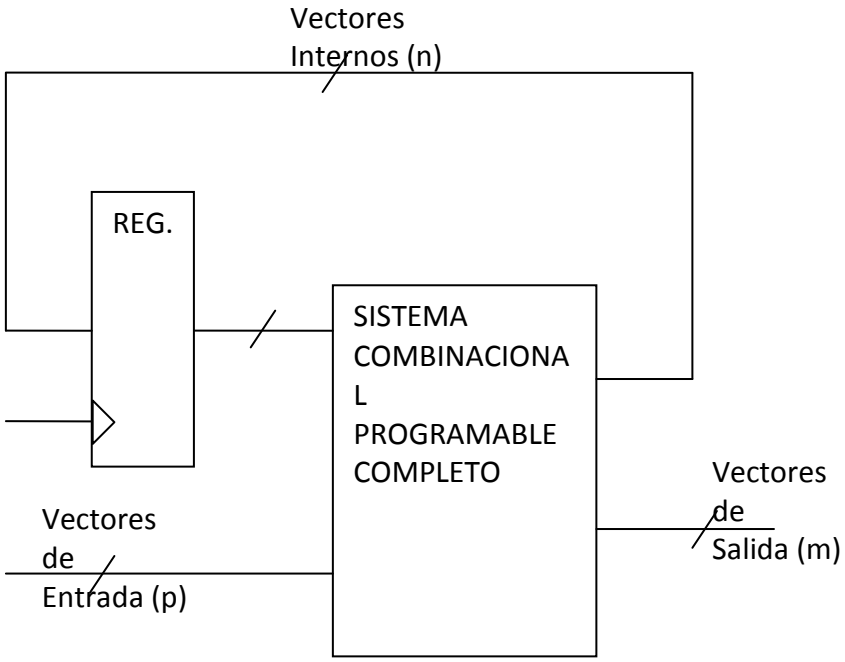


Figura 58

El Sistema Combinacional tiene n+p variables de entrada y n+m variables de salida. Esto implica que deberíamos implementar n+m funciones lógicas abarcando 2^{n+p} minterms posibles, se concluye entonces que podríamos usar una memoria RAM con n+p líneas de direccionamiento y con longitud de palabra de n+m bits, cada una de las 2^{n+p} posiciones serían los minterms y cada bit en la palabra serían las funciones lógicas. Una memoria RAM (de sólo lectura programable) se llama Sistema Combinacional Programable Completo.

El Registro se trata de un Registro de Desplazamiento Paralelo-Paralelo implementado con biestables D síncronos por flancos.

Un SSS implementado como se indica se llama **SSS de Control**, al contenido de la memoria se lo llama **microprograma** y al contenido de una posición de lo llama **microinstrucción.** Se observa que cada microinstrucción puede considerarse dividida en dos "campos": **variables de salida (m)** y **variables de estado (n).**

Variables de salida	Variables de estado

microinstrucción

Consideraciones:

1. En el diagrama en bloques se observa que se ha indicado en un solo bloque el Sistema Combinacional que, en el diagrama que vimos al presentar Sistemas Secuenciales, aparecía en dos bloques: SCI (en el cual realimentábamos las variables interna) y SCII (con el cual obteníamos las variables de salida), esto nos permitía clasificar los autómatas en Mealy o Moore. Para el caso actual ambos autómatas tienen el mismo diagrama, lo que implica que si el secuencial a implementar es un Moore las posiciones de memoria correspondientes a un mismo valor de las variables internas y distintos valores de las variables de entrada, tendrán el mismo contenido (es decir la misma microinstrucción). Lo mismo ocurrirá para un Mealy para aquellos estados en los cuales las variables de salida no dependen de las variables de entrada. Lo comentado supone un deterioro en la eficiencia del diseño ya que repetir microinstrucciones implica una RAM de mayor tamaño. También a este respecto es válido comentar que, dependiendo de la memoria usada, pueden aparecer fluctuaciones en las salidas cuando se direccionan posiciones de memoria que tienen el mismo contenido.
2. En la mayoría de los casos prácticos no es necesario considerar todos los minterms posibles, existen minterms que nunca se producen o si se producen no interesa considerarlos. Esto supone un deterioro en la eficiencia ya que tendríamos posiciones de memoria que nunca se direccionarían.
3. En la mayoría de los casos prácticos ocurre que para cambiar de un estado a otro solo interesa el estado de una variable y no el vector completo. Mirando el diagrama en bloques se observa nuevamente un deterioro en la eficiencia ya que si el sistema tiene que cambiar de un estado a otro en función de una única variable de entrada, todas las posiciones de memoria correspondientes a ese estado (n) y las 2^{n+p-1}, tendrán la misma microinstrucción. El mismo razonamiento puede hacerse para casos intermedios, es decir cuando el cambio de estado depende de más de una variable de entrada pero no de todo el vector. Lo comentado supone una RAM de mayor tamaño que el necesario.

Más adelante veremos cómo mejorar la eficiencia, no obstante es de considerar que este tipo de implementación presenta la ventaja de su simplicidad y permite modificar el secuencial solamente cambiando el microprograma.

Es oportuno clasificar los SSS en **cableados** y **microprogramados**.

Los cableados responden al diseño visto con anterioridad, es decir aquellos en los cuales los Combinacionales se implementan con puertas discretas o bloques funcionales. Para modificar el secuencial es necesario cambiar el cableado (de ahí su nombre).

Los microprogramados responden a la utilización de un combinacional programable.

DISEÑO

El diseño de estos SSS de Control (microprogramados) esencialmente es el mismo que ya hemos visto. Para estos casos debemos utilizar el método de la Tabla de Fases (TF) o los Grafos Reducidos (GR) por niveles para la obtención del Diagrama de Flujo.

Sólo vale comentar que la **TVT** es la que define el contenido de la RAM, así:

Líneas de direccionamiento (n+p) Antes del flanco (instante t)	Contenido (m+n) Microinstrucción Después del flanco (instante t+1)
Variables de estado (n) y de entrada (p)	Variables de estado (n) y de salida (m)
Ej 2^p minterms	Ej+1 salidas para Ej

Para el llenado de esta tabla debemos considerar lo siguiente:

- En la parte izquierda (instante t) suponemos que el secuencial está en un estado Ej (esto lo vemos en el Diagrama de Flujo), por lo tanto el valor de las variables de estado (n) es aquel que corresponde a la codificación del mismo. Las p variables restantes son 2^p combinaciones posibles de las variables de entrada, todas (aún las no posibles o que no interesen).
- En la parte derecha debemos colocar los valores de las variables de salida (m) correspondientes al estado Ej. Para el caso de las variables de estado debemos colocar la codificación correspondiente al estado Ej+1, es decir al estado al que deseamos que vaya el secuencial.
- Los dos puntos anteriores definen totalmente el contenido de la memoria.

VARIACIONES DEL PLANTEAMIENTO INICIAL

I)PAL o PLA

En la consideración 2) se indicó que en los casos prácticos es común que no todos los minterms sean posibles. La forma de tener en cuenta esto es reemplazar la RAM por una PAL o una PLA. La elección dependerá de cada caso, teniendo en cuenta cuantos minterms son posibles (es decir, cuales son los minterms a considerar en el caso). La programación de la PLA o PAL se realizará de acuerdo al contenido del la TVT cuyo llenado es similar al caso anterior sólo que los minterms que no importan no aparecerán en la misma.

El diagrama en bloques ahora es el siguiente:

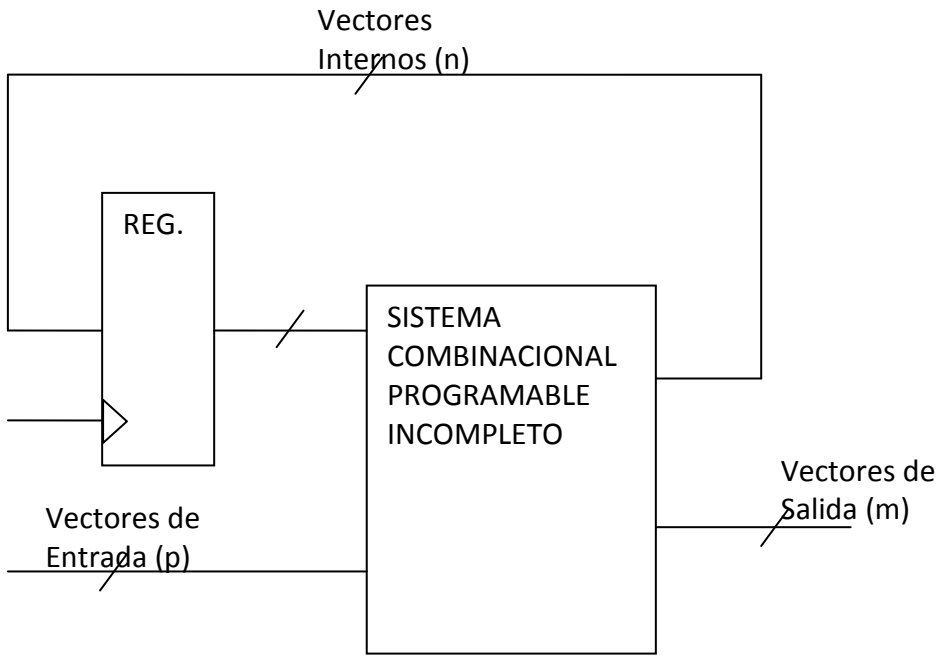


Figura 60

II)<u>DISMINUCIÓN DE VARIABLES DE ENTRADA</u>

En la consideración 3) se menciono que en muchos casos el cambio de estado sólo está condicionado al estado de algunas variables de entrada (no todas). Suponiendo el caso extremo que el secuencial cambia de estado en función de una única variable, es posible utilizar el siguiente diagrama:

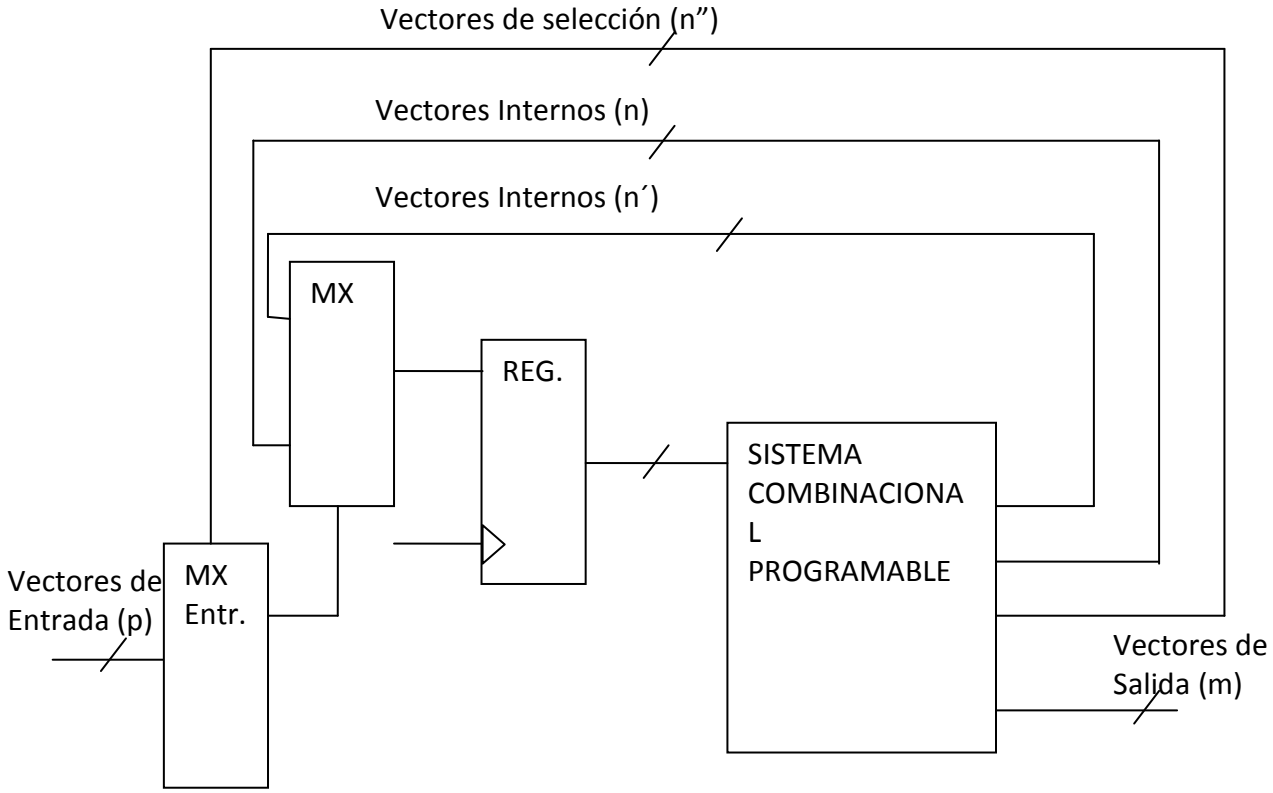


Figura 61

Ahora la microinstrucción tiene el siguiente aspecto:

Variables de salida (m)	Variables de estado (n)	Variables de estado (n´)	Variables de selección (n")

Donde:

- Variables de Salida (m bits): Salidas para el Estado actual del Secuencial
- Variables de Estado (n bits): Dirección de la próxima microinstrucción dependiendo del valor de la variable de entrada testeada. (n=n´)
- Variables de Estado (n´ bits): Dirección de la próxima microinstrucción dependiendo del valor de la variable de entrada testeada. (n=n´)
- Variables de Selección (n" bits): Estas variables son conectadas a las líneas de selección del multiplexor de entrada. Su cantidad depende de la cantidad de variables de entrada a testear.

La dirección de la próxima microinstrucción está contenida en la microinstrucción actual y depende del resultado del test de variables de entrada realizado por el multiplexor de entrada. Por ejemplo, si la variable de entrada testeada es “0”, la salida del multiplexor de entrada es “0” y seleccionará (en el otro multiplexor) las variables de estado (n) que serán cargados en el Registro cuando llegue el próximo flanco, así la próxima microinstrucción direccionada será aquella cuya dirección esté en el campo variables de estado (n) de la microinstrucción actual.

Se observa una disminución importante de la cantidad de posiciones de la RAM a costa de un aumento de la cantidad de bits de la palabra.
CONSIDERACIONES:

Esta variación planteada resuelve la siguiente situación:

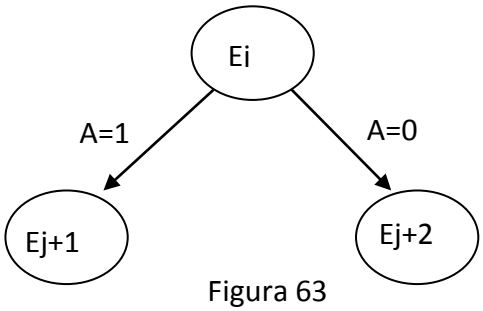


Figura 63

Donde, si el secuencial está en el estado Ej, pasa al Ej+1 si a=1 o al Ej+2 si A=0, que es un caso particular porque el cambio sólo depende de la variable de entrada A.

Si consideramos que el diagrama de flujo a resolver tiene casos más generales, como por ejemplo:

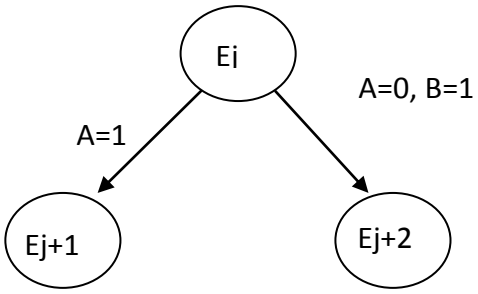


Figura 64

Podemos resolver el problema, conocido como **decisiones múltiples**, de dos formas:

- Agregando lógica en los canales de entrada del multiplexor de entrada, en este caso sería una AND que realice la función A`.B. Esto aumenta la lógica externa.
- Agregar líneas de direccionamiento a la RAM lo que agranda el tamaño de la memoria, como se ve en la figura:

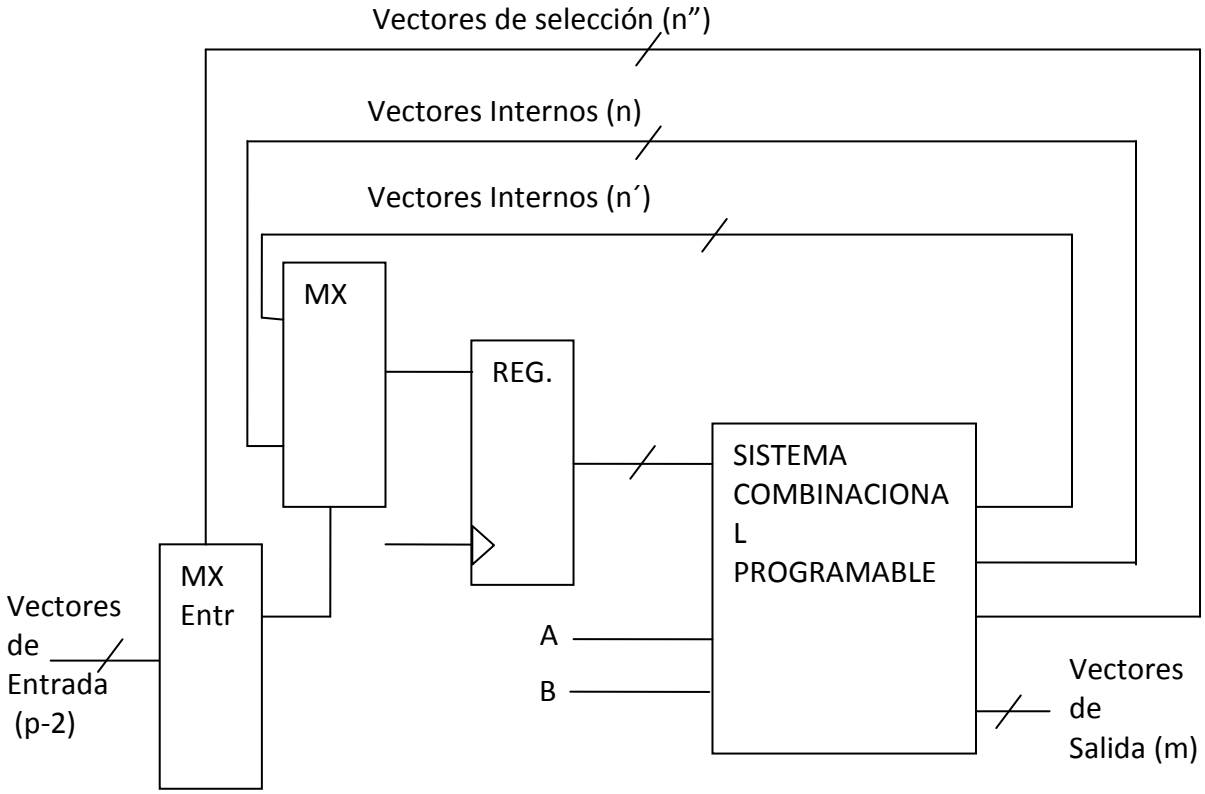


Figura 65

- En todos los caso es posible reemplazar la RAM con PAL o PLA como se comentó anteriormente.

III)DISMINUCIÓN DE LA LONGITUD DE PALABRA

Siempre es posible, al momento de realizar la codificación de estados internos, asignar al estado siguiente al actual (o al menos a uno de los siguientes) una codificación que represente la combinación siguiente en binario natural. Por ejemplo:

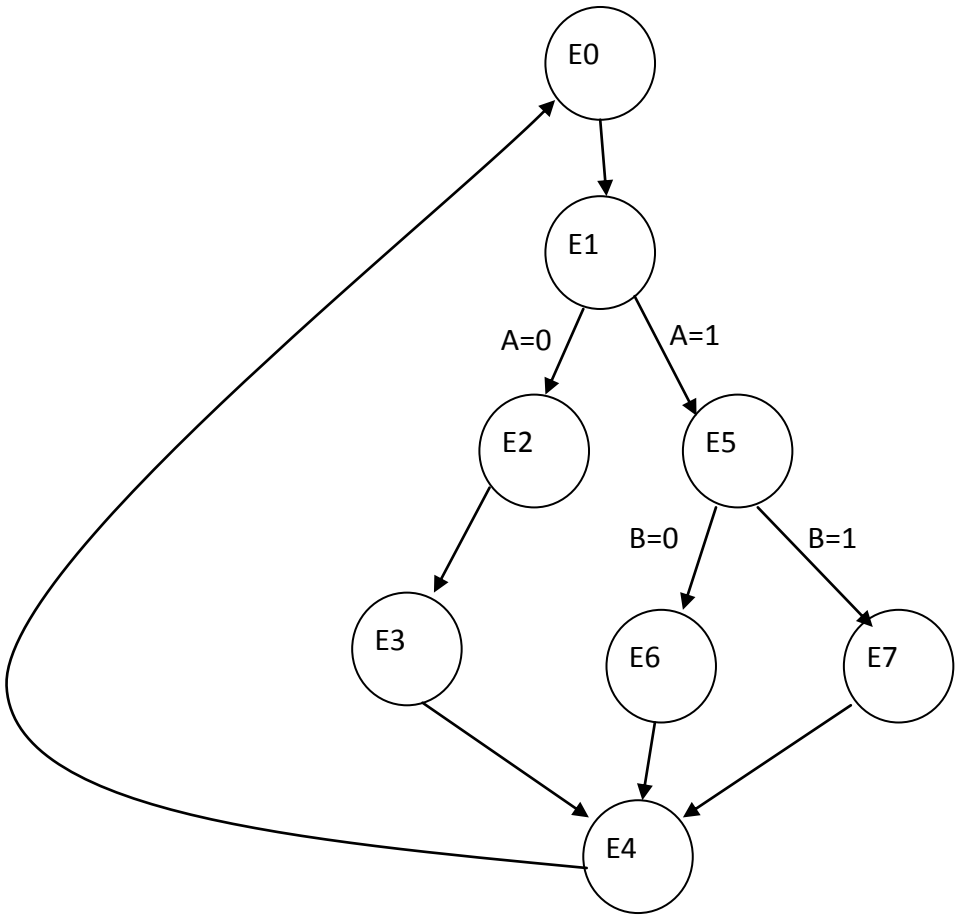
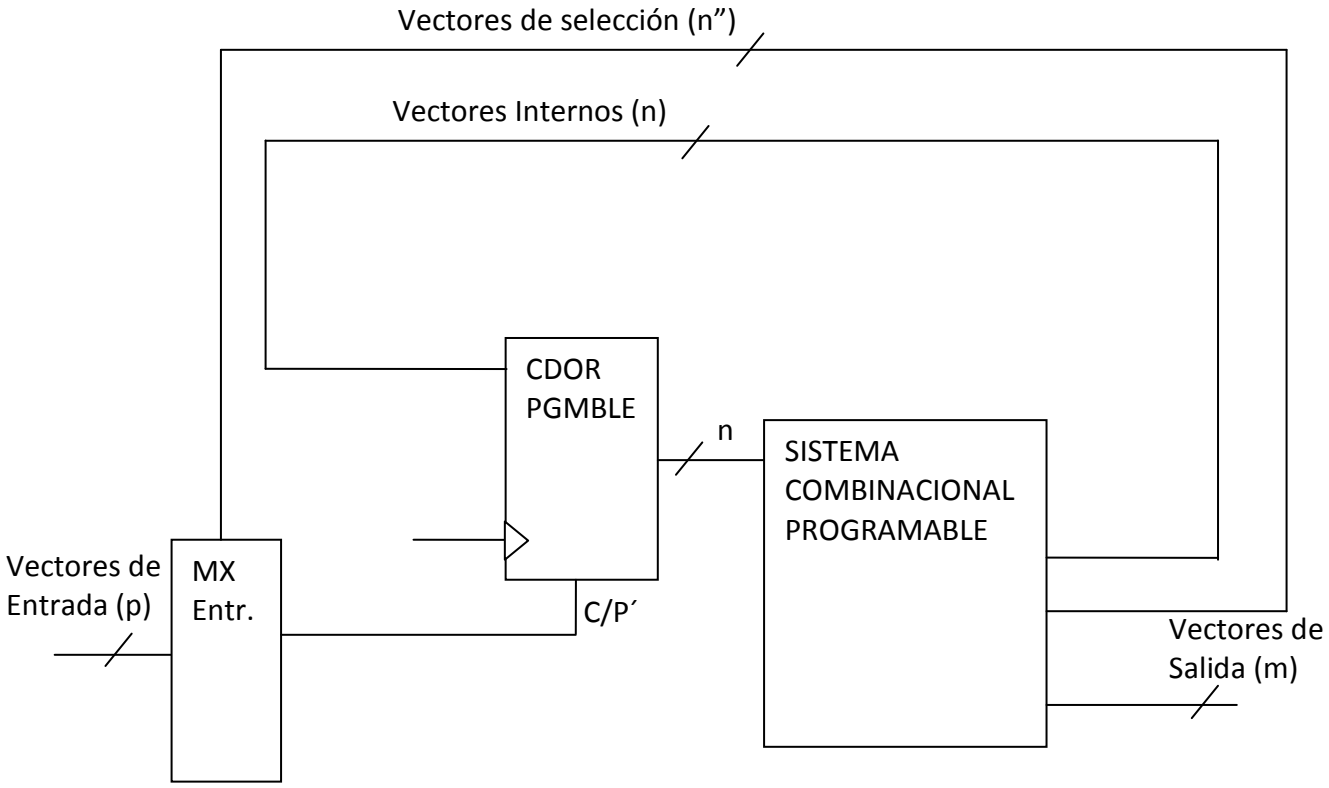


Figura 66

Se observa que si codificamos los estados del secuencial coincidentemente con los subíndices indicados en el diagrama de flujo, en muchos caso el próximo estado será el que tenga la combinación binaria siguiente (binario natural). Así podríamos usar un contador binario programable y hacer que el mismo cuente (para el caso del cambio E0 a E1 por ejemplo) o le cargamos el estado siguiente a través de sus líneas de carga paralela. De esta manera nos ahorramos el campo de Variables de Estado (n´) disminuyendo la longitud de palabra.

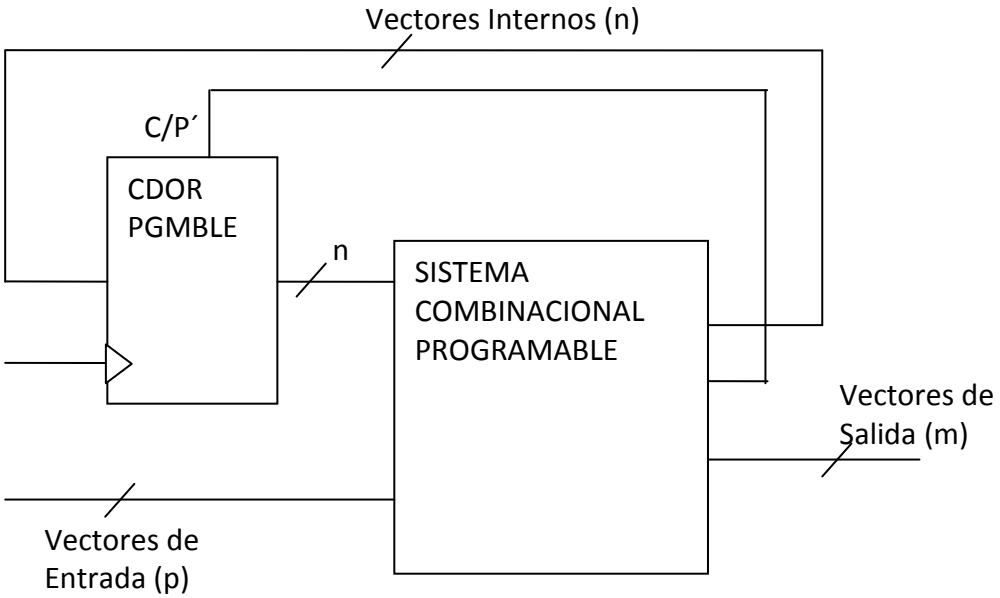
El diagrama en bloques resultante es, para el caso de usar multiplexor de entrada, el siguiente:



Variables de salida (m)	Variables de estado (n)	Variables de selección (n")

Figura 67

El diagrama en bloques si no usamos multiplexor de entrada, es el siguiente:



Variables de salida (m)	Variables de estado (n)	C/P´ (1)

Figura 68

Los esquemas de la página anterior plantean los casos extremos:

a) ninguna variable de entrada se usa como línea de dirección (lo que limita la decisión múltiple a una variable de entrada)

b) todas las variables de entrada son líneas de direccionamiento (no implica restricciones en las decisiones múltiples).

En la práctica es conveniente el planteo de configuraciones intermedias ya que dependerá del diagrama de flujo en particular.

En todos los caso es posible reemplazar la RAM con PAL o PLA como se comentó anteriormente.

_____