

9. Circuitos secuenciales asíncronos

angel@uc.cl

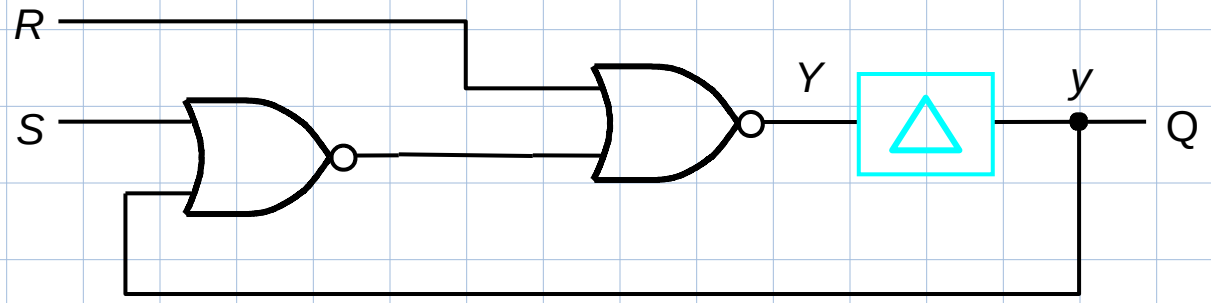
IEE2713 Sistemas Digitales

Motivación

- Ya aprendimos circuitos secuenciales síncronos
  - Los flip-flops son controlados por un reloj periódico
  - Todos los flip-flops que conmutan lo hacen al mismo tiempo
  - Operan en **modo pulso** porque son controlados por pulsos
- En este capítulo veremos circuitos secuenciales asíncronos, que no operan en modo pulso
  - Los cambios ocurren dependiendo del valor de las entradas
  - Para lograr operación confiable, **las entradas deben cambiar una a la vez**
    - Este supuesto es muy importante
    - Dos entradas simultáneas producirían resultados impredecibles
  - Los cambios en la entrada tienen que ocurrir de forma suficientemente lenta para que las salidas alcancen a establecerse (**estado estable**)
  - Esta modalidad de operación se denomina **modo fundamental**

9.1 Comportamiento asíncrono

- Consideremos un latch SR, donde el retardo de las compuertas NOR es representado por un bloque Δ
- Q es el estado presente, *y*, que toma el valor de *Y* luego de Δ



Present state *y*	Next state			
	SR = 00	01	10	11
	Y	*Y*	*Y*	*Y*
0	(0)	(0)	1	(0)
1	(1)	0	(1)	0

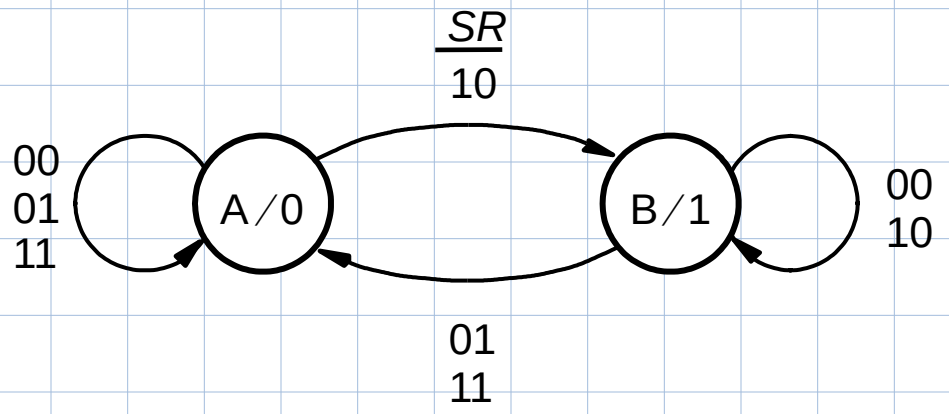
Si la salida inmediata 'Y' es distinta a la salida después del delay 'y', entonces se habla de un **estado inestable** en el sentido de que cambia (y no necesariamente que oscila)

Los círculos representan condición de estado siguiente igual a estado actual – decimos que el circuito es **estable** en esta condición y la notación del círculo es estándar

9.1 Comportamiento asíncrono

- A partir de la tabla de estados definidos anterior derivamos la tabla de estados y el diagrama de estados de una máquina de Moore
- Decimos que los estados A y B corresponden a los estados presentes y = 0 e y = 1

Present state	Next state				Output Q
	SR = 00	01	10	11	
A	(A)	(A)	B	(A)	0
B	(B)	A	(B)	A	1



- Vemos que los circuitos secuenciales asíncronos pueden ser representados de manera similar a los síncronos
- Sin embargo, la síntesis de circuitos asíncronos más complicados es considerablemente más difícil que la de circuitos síncronos

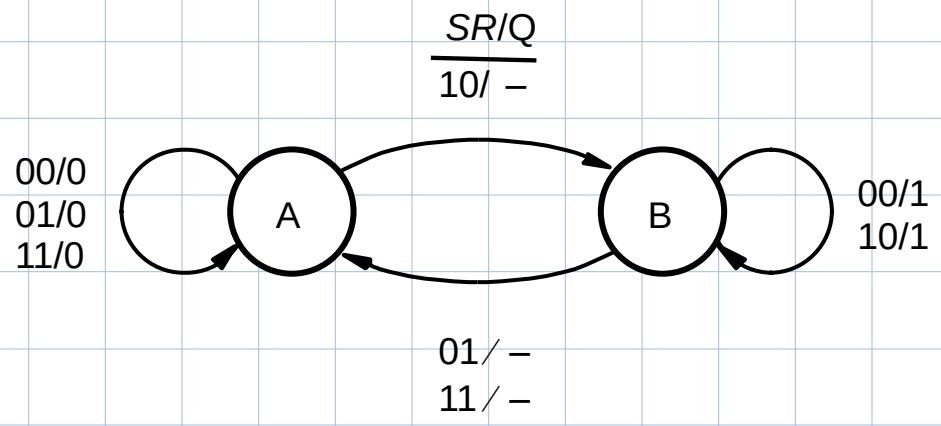
Cálculo del siguiente estado

$$Y = \overline{R} \cdot (S + y)$$

9.1 Comportamiento asíncrono

Present state	Next state				Output, Q			
	SR = 00	01	10	11	00	01	10	11
A	(A)	(A)	B	(A)	0	0	–	0
B	(B)	A	(B)	A	1	–	1	–

(a) State table



(b) State diagram

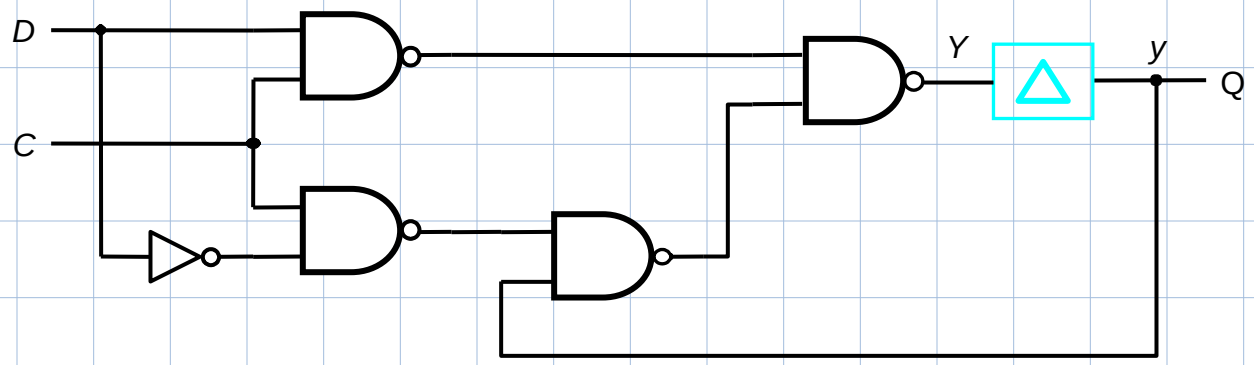
- Para completar el análisis, se puede plantear el latch como una maquina de Mealy
- En estado estable, la salida es 0 para el estado A y 1 para el estado B
- Algunas salidas Q no fueron especificadas:
  - En el modelo de Mealy la salida deberia ser afectada directamente por los cambios en las entradas
  - Para simplificar los circuitos se decidió dejar la salida como indefinida para casos inestables, pues el estado igual cambiará pasado un tiempo pequeño
    - No importa si la salida cambia antes o después de la transición de estado

Terminología importante

- Al lidiar con circuitos secuenciales asíncronos se usa terminología análoga, pero diferente a la de circuitos secuenciales síncronos
  - Se habla de tabla de flujo en lugar de tabla de estados
  - En lugar de hablar de tabla de estados asignados se habla de tabla de excitación o de transición
  - La tabla de flujos definirá los cambios en los estados y en las salidas
  - La tabla de excitación muestra las transiciones en términos de las variables de estado

9.2 Análisis de circuitos asíncronos: veamos un ejemplo – latch D

Nuevamente asumimos compuertas instantáneas y un retardo Δ en la rama de realimentación y que concentra los retardos de todas las compuertas



(a) Circuit

Latch tipo D: Esta vez asumimos asincronía, o bien, el reloj y la señal D no cambian al mismo tiempo

La mínima expresión para el siguiente estado es (simplificando Dy que es redundante y sirve para eliminar riesgos):

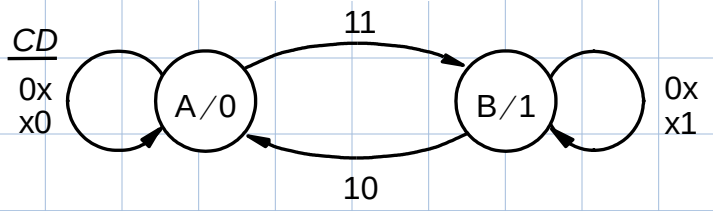
$$Y = CD + \overline{C}y$$

Present state y	Next state CD = 00	01	10	11	Q
	Y	Y	Y	Y	
0	(0)	(0)	(0)	1	0
1	(1)	(1)	0	(1)	1

(b) Excitation table

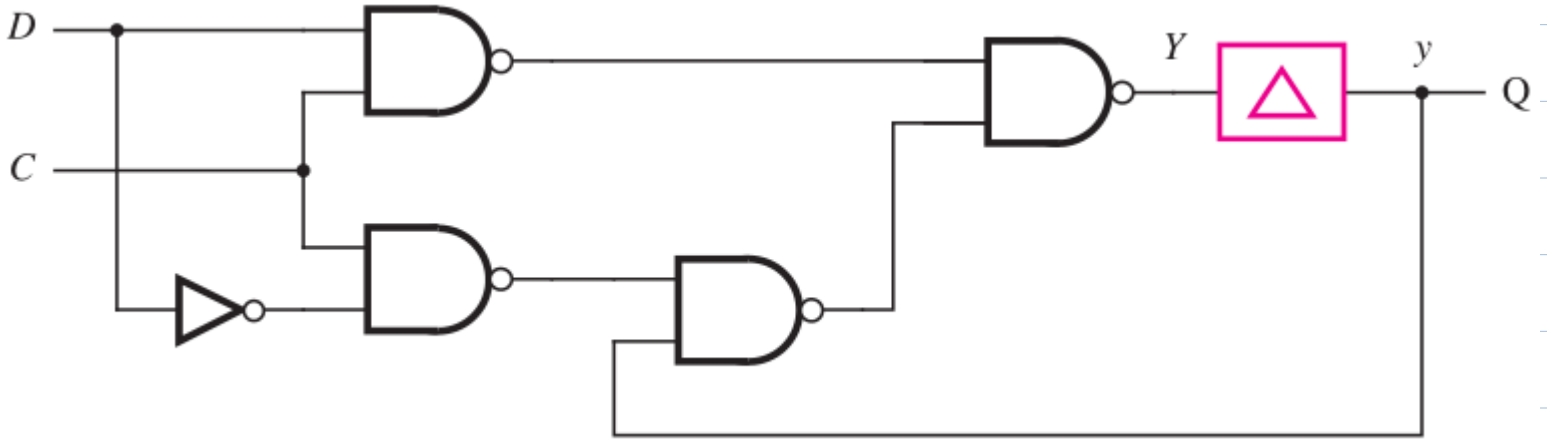
Present state	Next state CD = 00	01	10	11	Q
A	(A)	(A)	(A)	B	0
B	(B)	(B)	A	(B)	1

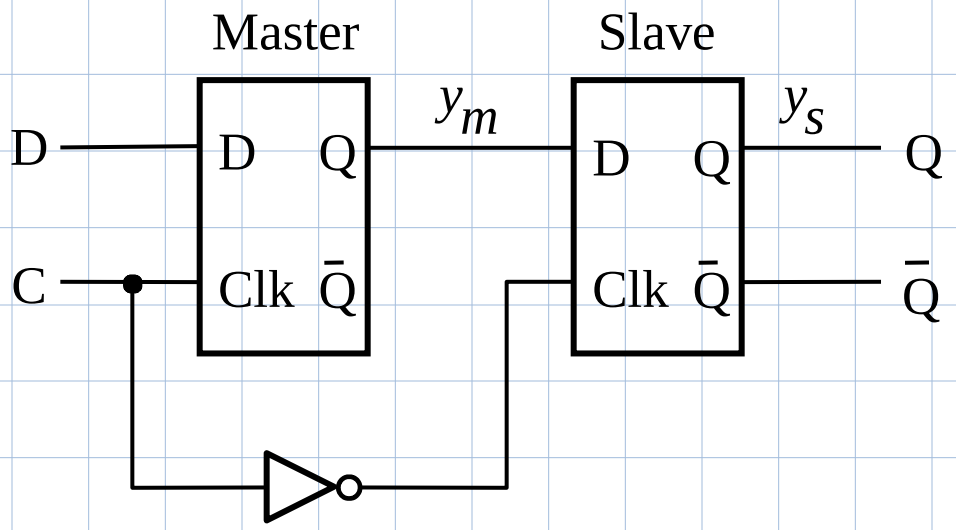
(c) Flow table



(d) State diagram

9.2 Análisis de circuitos asíncronos: otro ejemplo – FF D Master-slave





Podemos tratar este circuito como una conexión en cascada de dos latches D

Aplicando las expresiones vistas anteriormente:

$$Y_m = CD + \overline{C}y_m$$
$$Y_s = \overline{C}y_m + Cy_s$$

9.2 Análisis de circuitos asíncronos: otro ejemplo – FF D Master-slave

- La tabla de excitación resultante es:

Present state $y_m\ y_s$	Nextstate CD = 00	01	10	11	Output Q
	$Y_m\ Y_s$				
00	(00)	(00)	(00)	10	0
01	00	00	(01)	11	1
10	11	11	00	(10)	0
11	(11)	(11)	01	(11)	1

(a) Excitation table

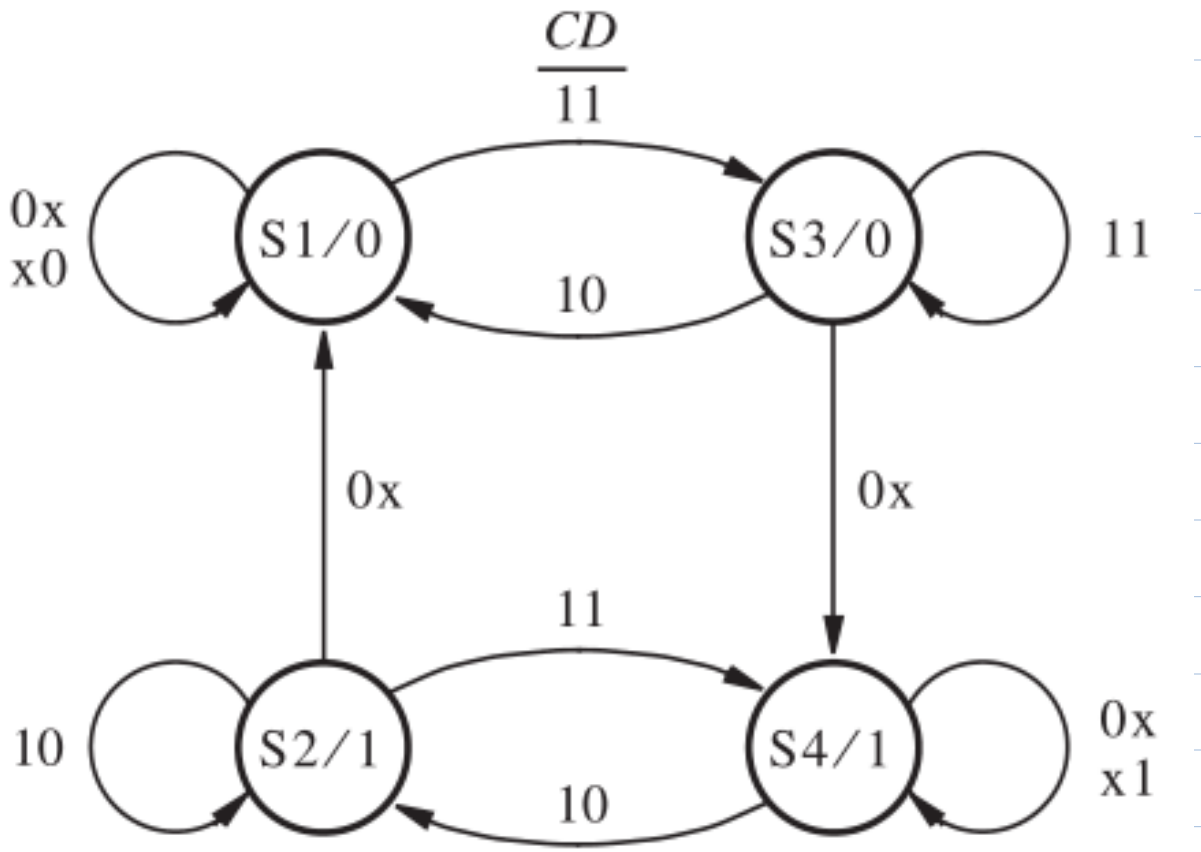
- La tabla de flujo es:

Present state	Nextstate CD = 00	01	10	11	Output Q
S1	(S1)	(S1)	(S1)	S3	0
S2	S1	S1	(S2)	S4	1
S3	S4	S4	S1	(S3)	0
S4	(S4)	(S4)	S2	(S4)	1

(b) Flow table

9.2 Análisis de circuitos asíncronos: otro ejemplo – FF D Master-slave

El diagrama de estados es:



Tiene 4 estados internos, aunque el FF Master-slave tiene sólo 2 estados externos y opera como un FF D gatillado por flanco negativo

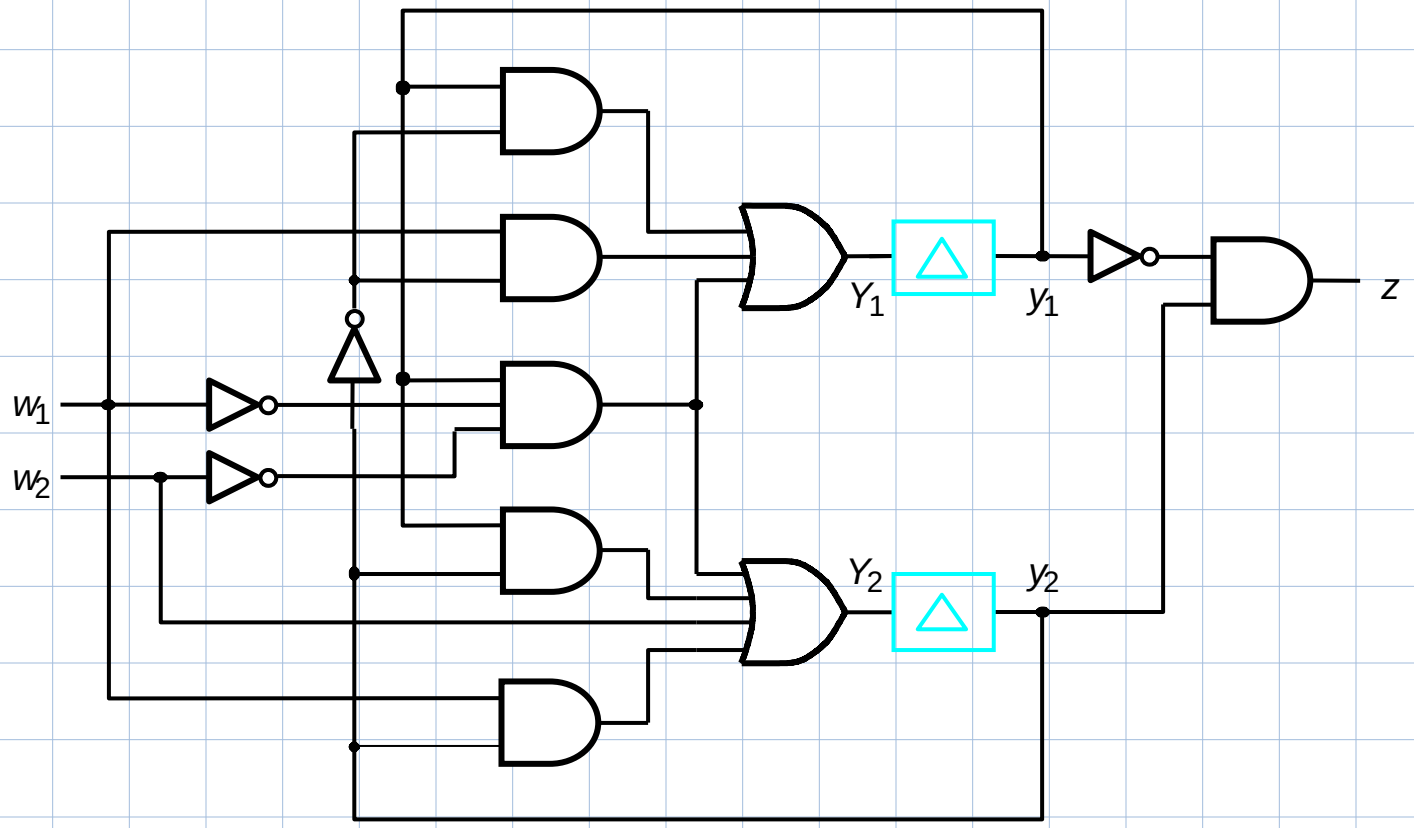
9.2 Análisis de circuitos asíncronos

- Dijimos que no puede haber cambios simultáneos en las entradas
  - Las entradas cambian una a la vez
- Entonces dejamos indeterminados los estados que requieren cambios simultáneos en las entradas

Present state	Next state				Output Q
	CD = 00	01	10	11	
S1	(S1)	(S1)	(S1)	S3	0
S2	S1	–	(S2)	S4	1
S3	–	S4	S1	(S3)	0
S4	(S4)	(S4)	S2	(S4)	1

(c) Flow table with unspecified entries

9.2 Análisis de circuitos asíncronos: Tercer ejemplo



$$Y_1 = y_1\bar{y}_2 + w_1\bar{y}_2 + \bar{w}_1\bar{w}_2y_1$$
$$Y_2 = y_1y_2 + w_1y_2 + w_2 + \bar{w}_1\bar{w}_2y_1$$
$$z = \bar{y}_1y_2$$

Las tablas en la página siguiente muestran que algunas transiciones de estado que requieren cambios simultáneos son válidas porque pasan por **estados inestables** que desembocan en un estado estable – esto es aceptable mientras no produzca glitches

9.2 Análisis de circuitos asíncronos

Present state y_2y_1	Nextstate w_2w_1 = 00	01	10	11	Output z
	Y_2Y_1	Y_2Y_1	Y_2Y_1	Y_2Y_1	
00	(00)	01	10	11	0
01	11	(01)	11	11	0
10	00	(10)	(10)	(10)	1
11	(11)	10	10	10	0

(a) Excitation table

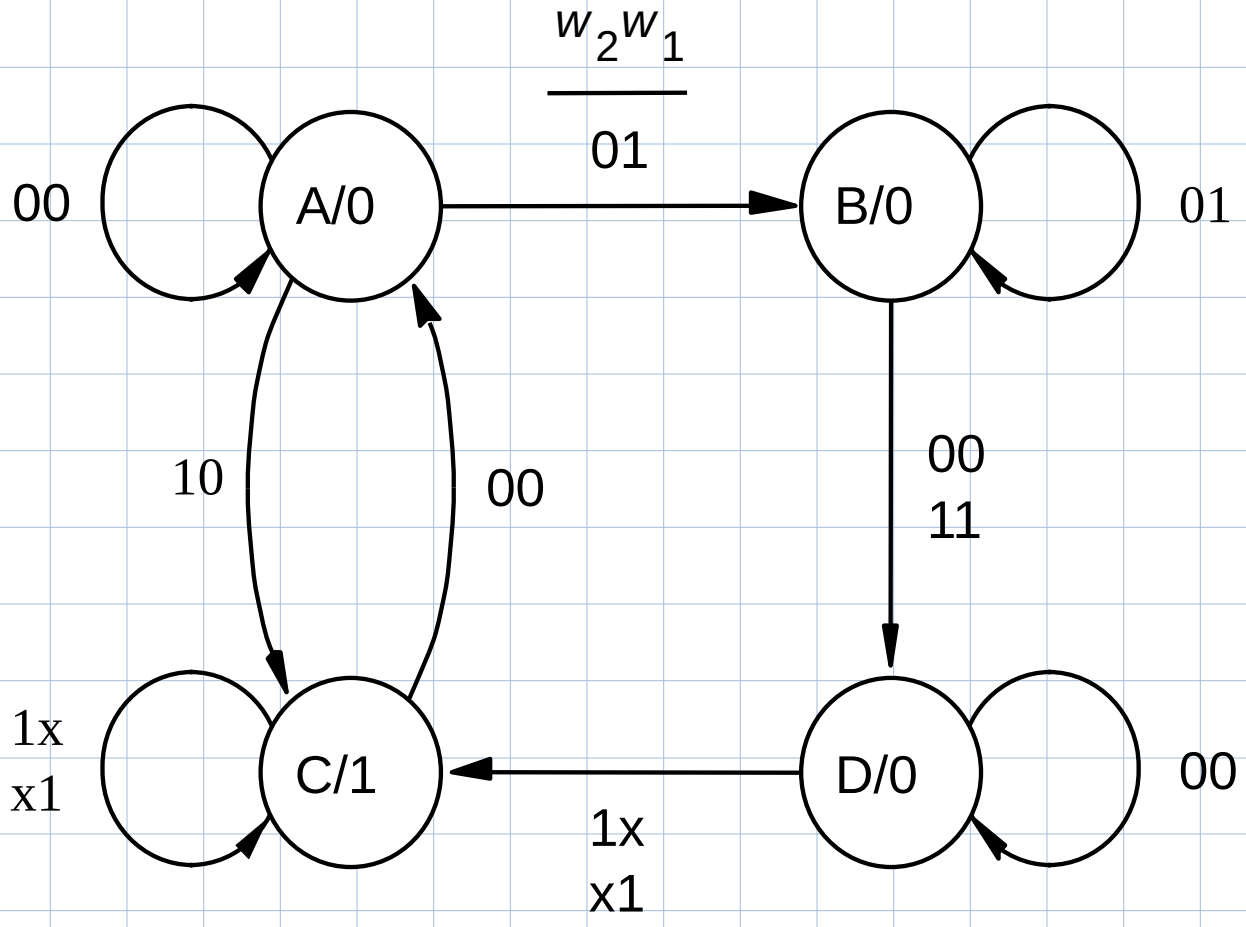
Present state	Nextstate w_2w_1 = 00	01	10	11	Output z
A	(A)	B	C	D	0
B	D	(B)	D	D	0
C	A	(C)	(C)	(C)	1
D	(D)	C	C	C	0

(b) Flow table

Present state	Next state w_2w_1 = 00	01	10	11	Output z
A	(A)	B	C	—	0
B	D	(B)	—	D	0
C	A	(C)	(C)	(C)	1
D	(D)	C	C	C	0

De igual forma que en casos anteriores, se crea una tabla modificada con valores no especificados para cambios de estado asociados a cambios síncronos en las entradas

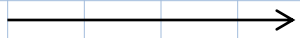
9.2 Análisis de circuitos asíncronos



- Este circuito busca representar una máquina expendedora, donde w_1 representa 5 centavos y w_2 representa 10 centavos
- el valor de un dulce es de 10 centavos
- Es imposible que entren al mismo tiempo una moneda de 5 y otra de 10 centavos
  - Entonces modificamos la tabla:

9.2 Análisis de circuitos asíncronos

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	B	C	—	0
B	D	(B)	—	D	0
C	A	(C)	(C)	(C)	1
D	(D)	C	C	C	0



Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	B	C	—	0
B	D	(B)	—	—	0
C	A	(C)	(C)	—	1
D	(D)	C	C	—	0

$w_2 \equiv$ dime $w_1 \equiv$ nickel

9.2 Análisis de circuitos asíncronos

A modo de resumen, podemos identificar los siguientes pasos:

- Para un circuito determinado, identificamos los lazos de realimentación e insertamos retardos en ellos
  - La entrada al retardo es la variable correspondiente al siguiente estado Y_i
  - La salida del retardo es el estado presente y_i
- Derivamos expresiones para el siguiente estado y la salida a partir del circuito
- Empleando las expresiones anteriores derivamos la tabla de excitación
- Obtenemos una tabla de flujo, asociando nombres arbitrarios a los estados
- Generamos un diagrama de estados

9.3 Síntesis de circuitos asíncronos

La síntesis de un circuito sencuencial asíncrono es similar al de un circuito síncrono:

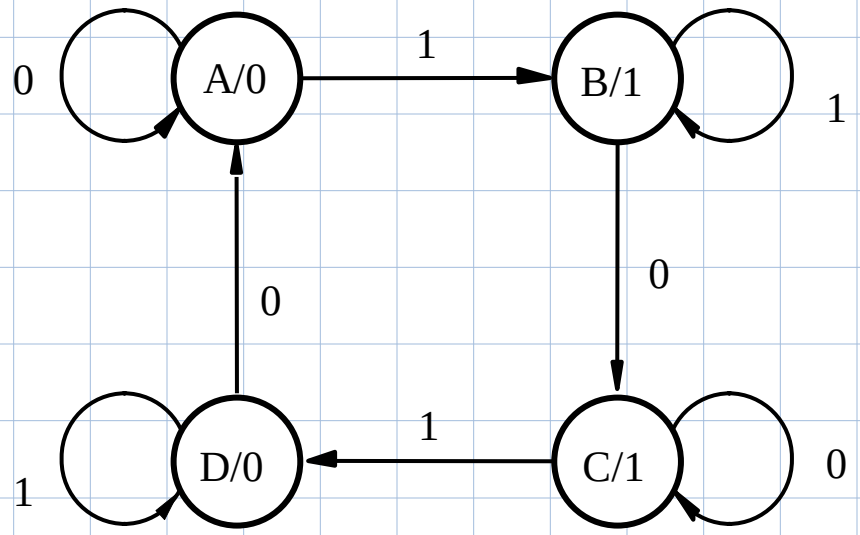
- Se dibuja un diagrama de estados para la FSM que produzca el comportamiento secuencial deseado
  - En estados estables, las salidas son correctas
  - En estados inestables transitorios, las salidas no deben presentar glitches
- Se deriva la tabla de flujos, reduciendo el número de estados de ser posible (sección 9.4)
- Se asignan estados y se genera una tabla de excitación
- Se obtienen expresiones para estados siguientes y salidas
- Se construye el circuito empleando esas expresiones

Ejemplo: Generador de señal de paridad

- Diseñemos un circuito de entrada w y salida z
- z = 0 si el numero de pulsos recibidos es par
- z = 1 si el numero recibido de pulsos es impar

Se definen 4 estados:

- A: w = 0; z = 0
- B: w = 1; z = 1
- C: w = 0; z = 1
- D: w = 1; z = 0



(a) State diagram

Present State	Next state		Output z
	$w = 0$	$w = 1$	
A	(A)	B	0
B	C	(B)	1
C	(C)	D	1
D	A	(D)	0

(b) Flow table

Ejemplo: Generador de señal de paridad

A continuación se deben realizar las asignaciones de estado a los estados presentes

Asignación de estados deficiente

Present state $y_2 y_1$	Next state $w = 0$	Next state $w = 1$	Output z
	$Y_2 Y_1$		
00	(00)	01	0
01	10	(01)	1
10	(10)	11	1
11	00	(11)	0

Asignación de estados mejorada

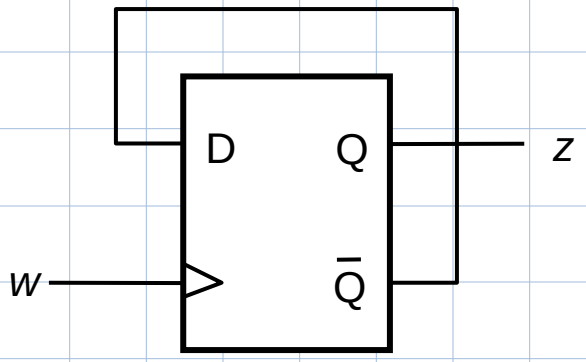
Present state $y_2 y_1$	Next state $w = 0$	Next state $w = 1$	Output z
	$Y_2 Y_1$		
00	(00)	01	0
01	11	(01)	1
11	(11)	10	1
10	00	(10)	0

La asignación de la izquierda es una mala idea, pues requiere de un cambio simultáneo de y_2 e y_1 al pasar de D a A, contraviniendo los principios de un circuito asíncrono (sólo una entrada puede cambiar a la vez)

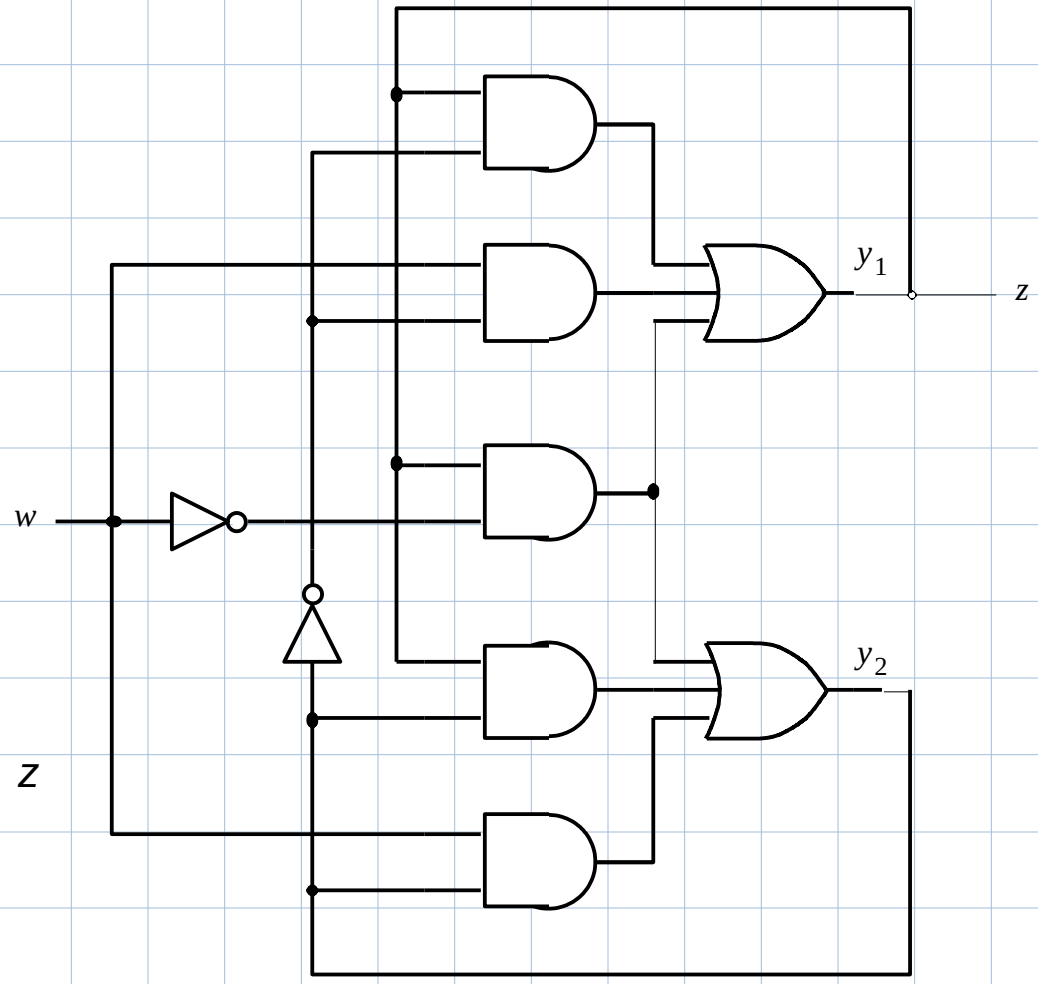
Ejemplo: Generador de señal de paridad

$$Y_1 = w\bar{y}_2 + \bar{w}y_1 + y_1\bar{y}_2$$
$$Y_2 = wy_2 + \bar{w}y_1 + y_1y_2$$
$$z = y_1$$

Ecuaciones obtenidas de la tabla de excitación



Solución síncrona



Solución asíncrona

A partir de la tabla podemos encontrar las ecuaciones correspondientes, y a partir de ahí el circuito.

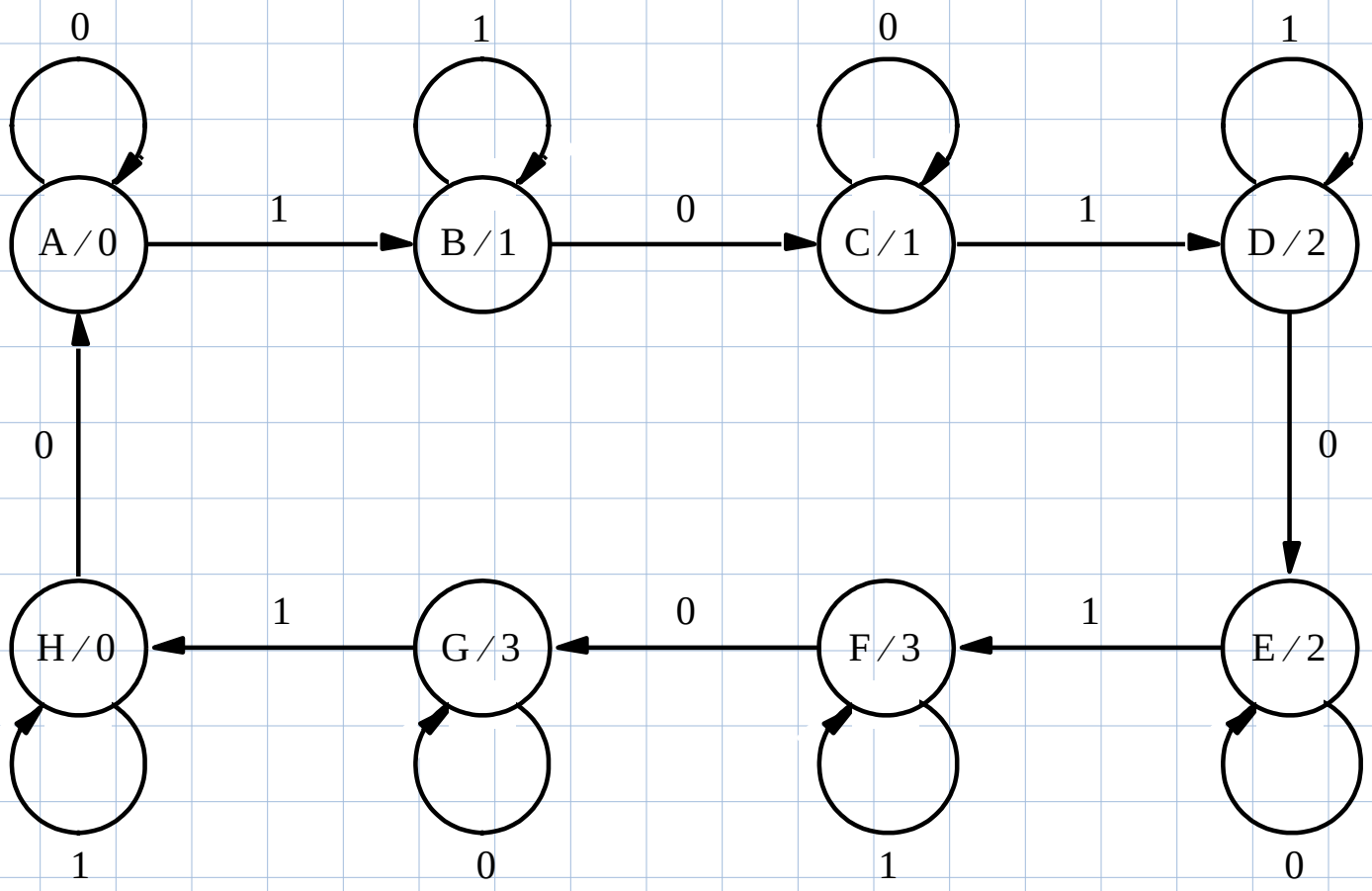
Se incluye la solución síncrona para comparación.

Nótese que hay términos que se podrían reducir en las ecuaciones, que no se eliminaron para evitar *hazards* (explicados en 9.6)

Ejemplo: Contador de módulo 4

Suponemos una entrada w

Ocupando una maquina de Moore se dibujan dos estados por cuenta



Present state	Next state $w = 0$	Next state $w = 1$	Output z
A	(A)	B	0
B	C	(B)	1
C	(C)	D	1
D	E	(D)	2
E	(E)	F	2
F	G	(F)	3
G	(G)	H	3
H	A	(H)	0

(a) Flow table

Ejemplo: Contador de módulo 4

Present state $y_3y_2y_1$	Next state $Y_3Y_2Y_1$ $w = 0$	Next state $Y_3Y_2Y_1$ $w = 1$	Output z_2z_1
000	(000)	001	00
001	011	(001)	01
011	(011)	010	01
010	110	(010)	10
110	(110)	111	10
111	101	(111)	11
101	(101)	100	11
100	000	(100)	00

(b) Excitation table

Mod-8 output $z_3z_2z_1$
000
001
010
011
100
101
110
111

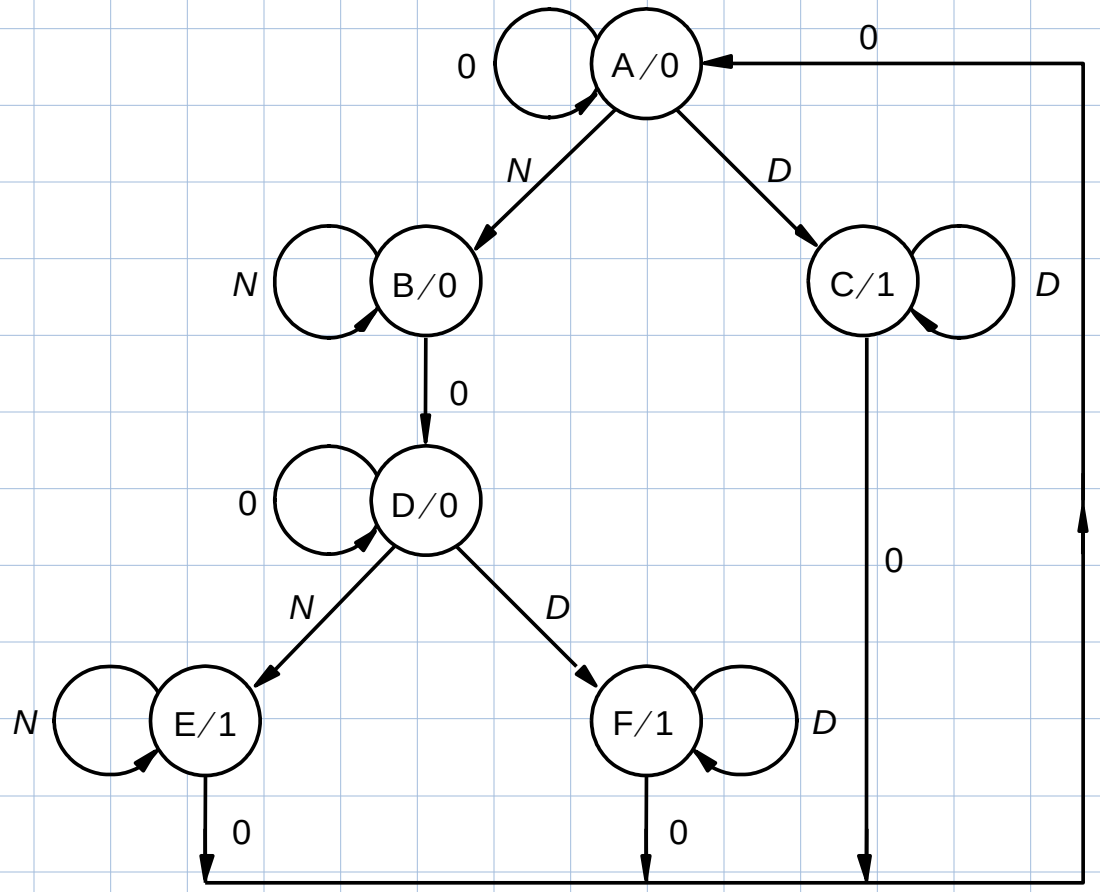
(c) Output for counting the edges

Esta salida alternativa cuenta flancos

$$Y_1 = \overline{w}y_1 + wy_2y_3 + w\overline{y}_2\overline{y}_3 + y_1y_2y_3 + y_1\overline{y}_2\overline{y}_3$$
$$= \overline{w}y_1 + (w + y_1)(y_2y_3 + \overline{y}_2\overline{y}_3)$$
$$Y_2 = wy_2 + \overline{w}y_1\overline{y}_3 + \overline{y}_1y_2 + y_2\overline{y}_3$$
$$Y_3 = wy_3 + y_1y_3 + \overline{y}_1y_2\overline{w} + y_2y_3$$
$$z_1 = y_1$$
$$z_2 = y_1y_3 + \overline{y}_1y_2$$

¿Por qué codificamos Gray?

9.4 Reducción de estados: ejemplo



- Similar a lo que hacemos con circuitos síncronos
- **Estudiemos la reducción de estados con un ejemplo**:
  - Diseñemos una máquina expendedora que recibe monedas de 5 (N) o monedas de 10 (D)
  - El snack cuesta 10
  - Si la máquina recibe más de 10, no da vuelto
- Diagrama de estados

9.4 Reducción de estados: ejemplo

- Obtenemos la tabla de flujo a partir del diagrama de estados
- Muchos don’t cares
- Cada fila tiene sólo un estado estable
  - Por lo tanto se denomina tabla de flujo primitiva
- Veremos una técnica de reducción de estados en dos pasos:
  - Mismo procedimiento de partición de 8.6.1
    - Para que dos filas sean potencialmente equivalentes, don’t cares deben estar en la misma columna
  - Unimos filas y aprovechamos don’t cares

Present State	Next state				Output
	DN = 00	01	10	11	*z*
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	A	–	(C)	–	1
D	(D)	E	F	–	0
E	A	(E)	–	–	1
F	A	–	(F)	–	1

9.4 Reducción de estados: ejemplo

- Apliquemos el procedimiento:
  - Estados A y D son estables para DN = 00, tienen salida 0 y tienen don't cares en la misma columna
  - Estados C y F son estables para DN = 10, tienen salida 1 y comparten don't cares
  - Estados B y E comparten don't cares y son estables para DN = 01, pero sus salidas son distintas
- Entonces $P_1 = (AD)(B)(CF)(E)$
- A y D desembocan en particiones diferentes, así que no son estados equivalentes
- Entonces $P_2 = (A)(D)(B)(CF)(E)$
- Finalmente comparando sucesores de C y F, concluimos que $P_3 = (A)(D)(B)(CF)(E)$
- C y F son equivalentes – F desaparece
  - Tratemos de unirlas aprovechando don't cares

Present State	Next state				Output
	DN = 00	01	10	11	*z*
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	A	–	(C)	–	1
D	(D)	E	F	–	0
E	A	(E)	–	–	1
F	A	–	(F)	–	1

9.4 Reducción de estados: ejemplo

- Ahora tratamos de aprovechar don't cares para unir algunas filas
  - A y C producen salidas distintas, sólo podríamos mezclarlas usando Mealy…
  - Podemos unir C y E y llegamos a esto

Present state	Next state				Output
	DN = 00	01	10	11	*z*
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	A	–	(C)	–	1
D	(D)	E	C	–	0
E	A	(E)	–	–	1

Present state	Next state				Output
	$w_2\,w_1$ = 00	01	10	11	*z*
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	A	(C)	(C)	–	1
D	(D)	C	C	–	0

$w_2 \equiv$ dime $w_1 \equiv$ nickel

9.4 Reducción de estados: Procedimiento de unión

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	H	B	–	0
B	F	–	(B)	C	0
C	–	H	–	(C)	1
D	A	(D)	–	E	1
E	–	D	G	(E)	1
F	(F)	D	–	–	0
G	F	–	(G)	–	0
H	–	(H)	–	E	0

Dos estados, S_i y S_j, son compatibles si no hay conflictos de estado para cualquier entrada. Entonces para cada valor de entrada, se debe cumplir una de las siguientes:

- S_i y S_j tienen el mismo sucesor, o
- S_i y S_j son estables, o
- El sucesor de S_i o S_j, o ambos, es don't care

Además, S_i y S_j deben tener la misma salida si ésta se especifica (no es don't care)

Ejemplo: tabla de flujos primitiva… Analicémosla

9.4 Reducción de estados: Procedimiento de unión - ejemplo

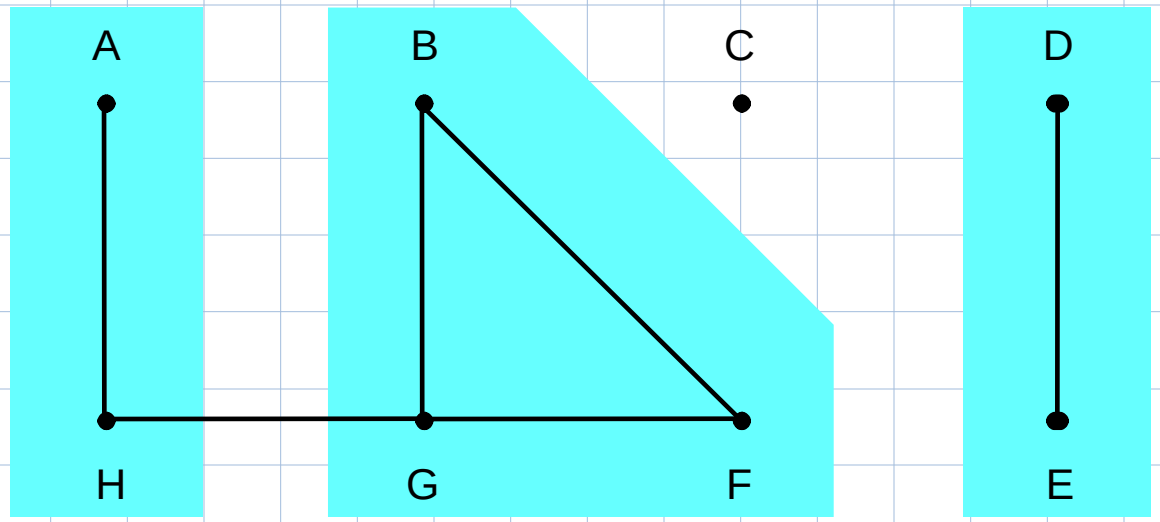
Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	H	B	–	0
B	F	–	(B)	C	0
C	–	H	–	(C)	1
D	A	(D)	–	E	1
E	–	D	G	(E)	1
F	(F)	D	–	–	0
G	F	–	(G)	–	0
H	–	(H)	–	E	0

- Analicemos compatibilidad:
  - A es compatible con H
  - B es compatible con F y G
  - C no es compatible con otro estado
  - D es compatible con E
  - F es compatible con G
  - G es compatible con H
- Entonces tenemos los siguientes pares compatibles: (A, H), (B, F), (B, G), (D, E), (F, G), (G, H)
- Podemos expresar esta compatibilidad con un diagrama de unión:

9.4 Reducción de estados

Cuando múltiples estados son compatibles entre sí, una estrategia corresponde en hacer un diagrama de unión:

- Cada estado se representa con un punto
- Si dos estados son compatibles entre si, entonces se traza una línea entre ambos:



Esta tabla es el resultado de unir (A, H), (B, F, G) y (D, E)

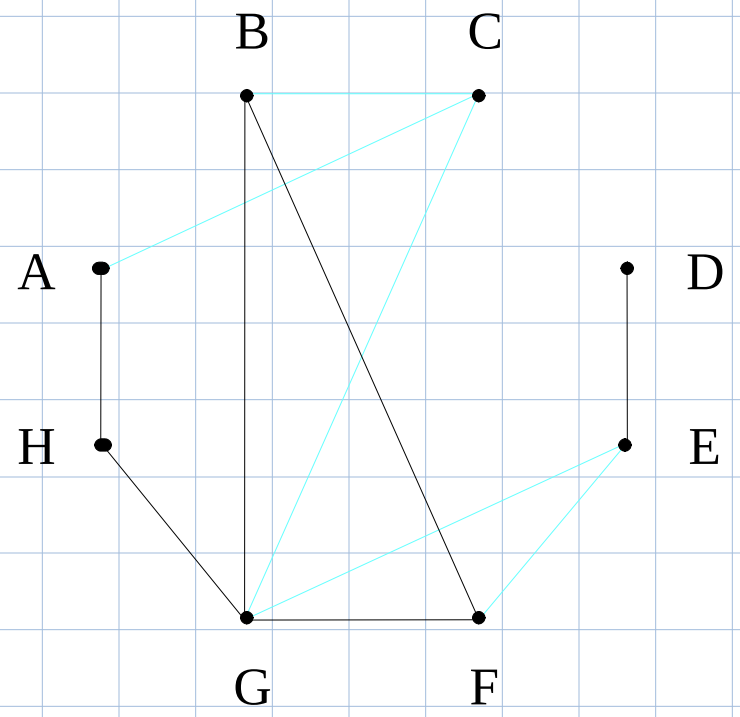
Present state	Next state w_2w_1 = 00	01	10	11	Output z
A	(A)	(A)	B	D	0
B	(B)	D	(B)	C	0
C	–	A	–	(C)	1
D	A	(D)	B	(D)	1

No podemos unir A y H si H además es unido a G, porque no hay línea que una A y G

9.4 Reducción de estados

En el ejemplo anterior quisimos mantenernos en modelo de Moore.

Otra opción habría sido pasar a estrategia de Mealy (diagrama). Las líneas negras son estados conectables a través de Moore y las líneas azules son estados conectables a través de Mealy (no es más eficiente en este caso, pero es una opción posible).



Receta para reducir estados:

1. Hacemos las particiones iniciales para eliminar estados equivalentes en tabla de flujo primitiva.
2. Dibujamos el diagrama de unión.
3. Hacemos subgrupos de estados que pueden unirse, minimizando subgrupos. Un estado sólo puede pertenecer a un subgrupo.
4. Reconstruimos la tabla de flujos uniendo filas en subgrupos elegidos.
5. Repetimos los pasos 2-4 hasta que no sea posible reducir más los estados.

Ejemplo 1

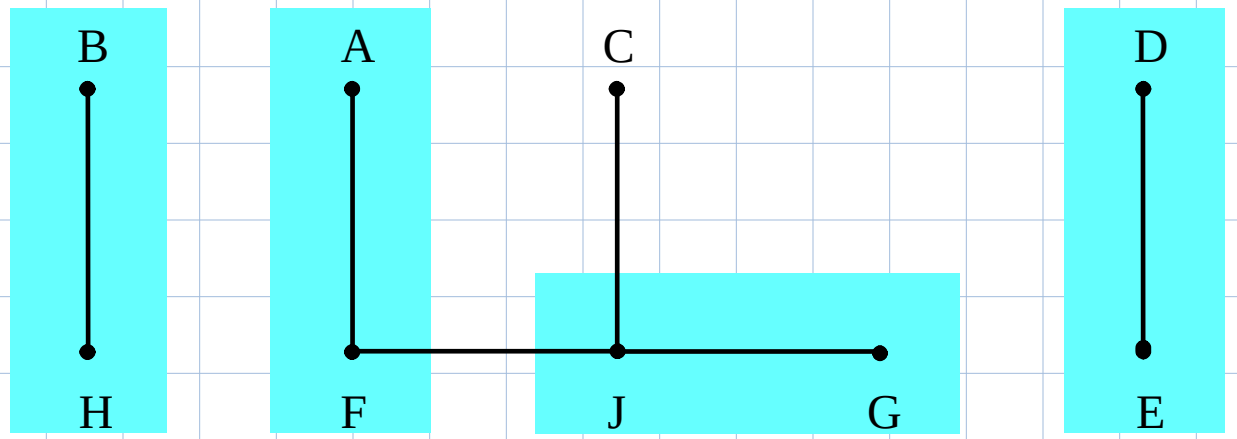
Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	F	C	—	0
B	A	(B)	—	H	1
C	G	—	(C)	D	0
D	—	F	—	(D)	1
E	G	—	(E)	D	1
F	—	(F)	—	K	0
G	(G)	B	J	—	0
H	—	L	E	(H)	1
J	G	—	(J)	—	0
K	—	B	E	(K)	1
L	A	(L)	—	K	1

- Vemos que los pares (A, G), (B, L) y (H, K) son potencialmente equivalentes, pues comparten salidas y don't cares
- P_1 = (AG)(BL) (C)(D)(E)(F)(HK)(J)
- (A, G) no son equivalentes, pues los sucesores no siempre están en el mismo bloque
- Entonces una segunda partición parece ser:
- P_2 = (A)(G)(BL)(C)(D)(E)(F)(HK)(J)
- Revisando más criterios se ve que los estados son efectivamente equivalentes:
- $P_3 = P_2$

Ejemplo 1

Reconstruimos la tabla de flujo y dibujamos el diagrama de unión

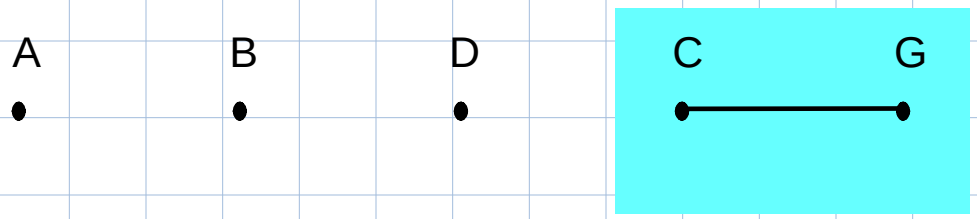
Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	F	C	—	0
B	A	(B)	—	H	1
C	G	—	(C)	D	0
D	—	F	—	(D)	1
E	G	—	(E)	D	1
F	—	(F)	—	H	0
G	(G)	B	J	—	0
H	—	B	E	(H)	1
J	G	—	(J)	—	0



- B y H debieran quedar unidos
- D y E debieran quedar unidos
- A y F pueden ser unidos pero esto impediría unir F y J
- C y J pueden ser unidos, o alternativamente, G y J
- La elección de uniones está sombreada

Ejemplo 1

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	(A)	C	B	0
B	A	(B)	D	(B)	1
C	G	—	(C)	D	0
D	G	A	(D)	(D)	1
G	(G)	B	(G)	—	0



Podemos repetir este paso una segunda vez, llegado a una **tabla de flujo** que ya no puede seguir siendo reducida

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	(A)	C	B	0
B	A	(B)	D	(B)	1
C	(C)	B	(C)	D	0
D	C	A	(D)	(D)	1

Ejemplo 2

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	G	E	—	0
B	K	—	(B)	D	0
C	F	(C)	—	H	1
D	—	C	E	(D)	0
E	A	—	(E)	D	1
F	(F)	C	J	—	0
G	K	(G)	—	D	1
H	—	—	E	(H)	1
J	F	—	(J)	D	0
K	(K)	C	B	—	0

Una partición inicial, ocupando lo mismo del ejemplo anterior entrega:

P_1 = (AFK)(BJ)(CG)(D)(E)(H)

P_2 = (A)(FK)(BJ)(C)(G)(D)(E)(H)

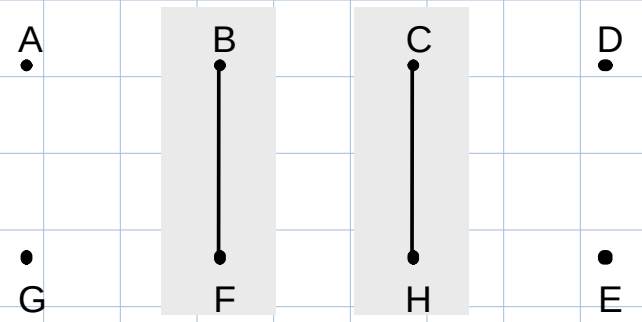
$P_3 = P_2$

Combinamos B y J, F y K y llegamos a la lámina siguiente:

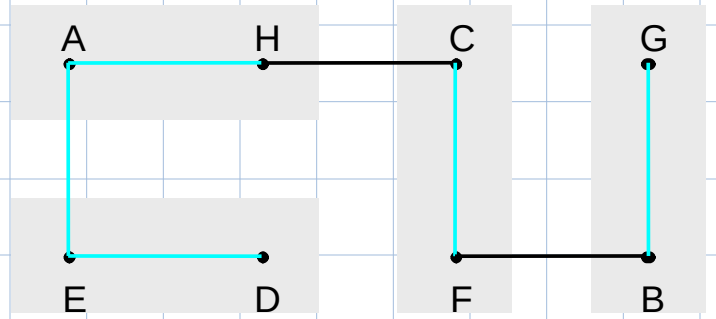
Ejemplo 2

Ahora, además de dibujar el diagrama de estados y la nueva tabla de flujos, podemos pensar en qué pasaría se cambiamos a estrategia de Mealy – debe generar mismas salidas y sin glitches

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	(A)	G	E	—	0
B	F	—	(B)	D	0
C	F	(C)	—	H	1
D	—	C	E	(D)	0
E	A	—	(E)	D	1
F	(F)	C	B	—	0
G	F	(G)	—	D	1
H	—	—	E	(H)	1



(a) Preserving the Moore model



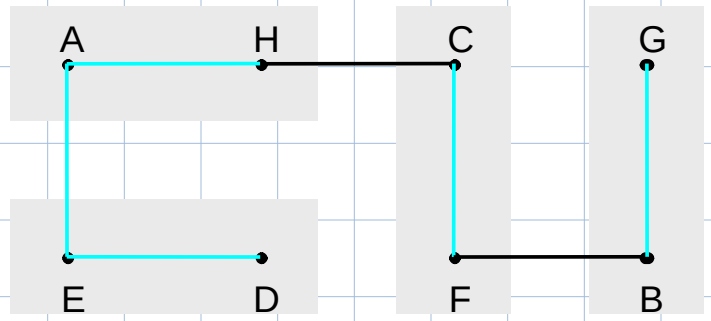
(b) Complete merger diagram

Ejemplo 2

Hacemos, por tanto, el cambio a máquina de Mealy, y seguimos con los pasos habituales, partiendo con la tabla de flujo

Present state	Next state				Outputz			
w_2w_1 =	00	01	10	11	00	01	10	11
A	(A)	G	E	–	0	–	–	–
B	F	–	(B)	D	0	–	0	0
C	F	(C)	–	H	–	1	–	1
D	–	C	E	(D)	–	–	–	0
E	A	–	(E)	D	–	–	1	–
F	(F)	C	B	–	0	–	0	–
G	F	(G)	–	D	–	1	–	–
H	–	–	E	(H)	–	–	1	1

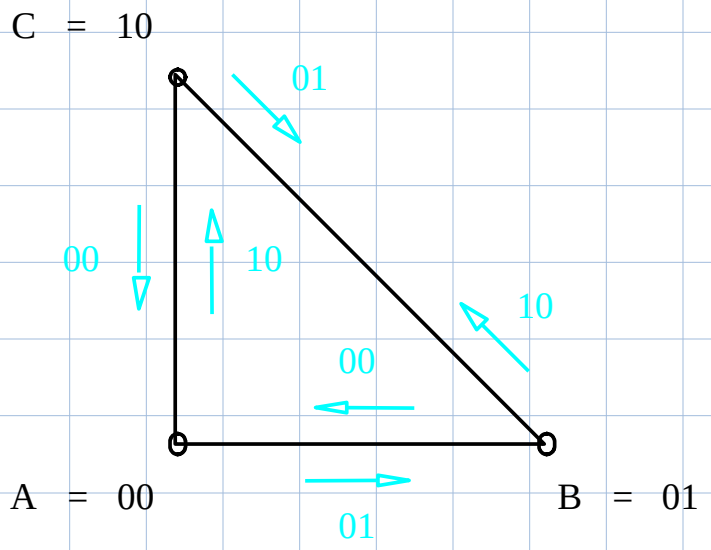
Reduciendo →



Present state	Nextstate				Outputz			
w_2w_1 =	00	01	10	11	00	01	10	11
A	(A)	B	D	(A)	0	–	1	1
B	C	(B)	(B)	D	0	1	0	0
C	(C)	(C)	B	A	0	1	0	1
D	A	C	(D)	(D)	–	–	1	0

9.5 Asignación de estados

- La asignación de estados debe ser tal que, de haber transición de un estado a otro, la distancia de Hamming debe ser 1, pues tratamos con circuitos asíncronos
- A veces es necesario **aumentar el número de variables de estado** para cumplir con esto



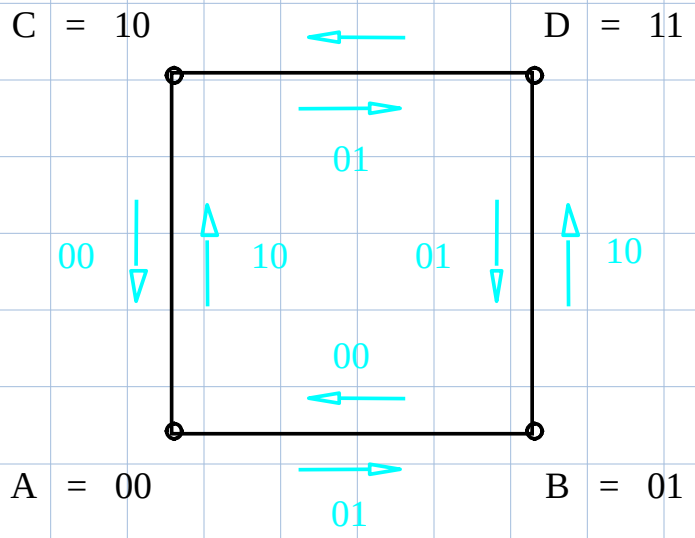
Present state	Next state $r_2 r_1$ = 00	01	10	11	Output $g_2 g_1$
A	(A)	B	C	–	00
B	A	(B)	C	(B)	01
C	A	B	(C)	(C)	10

El paso entre B y C no tiene distancia 1!!!

(a) Transitions in Figure 9.21*a*

9.5 Asignación de estados

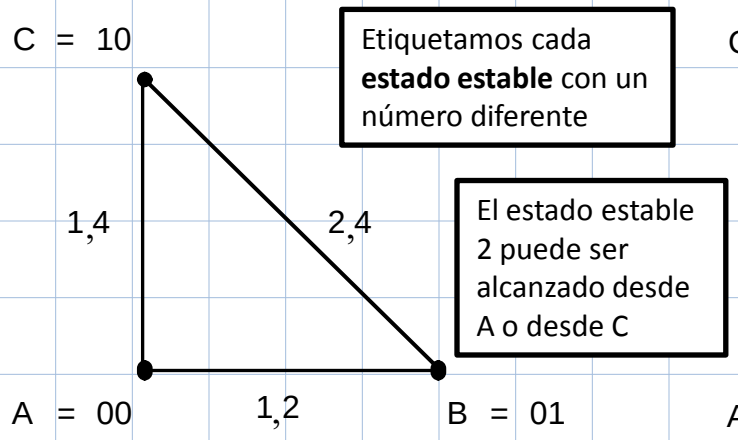
- El paso entre B y C no tiene distancia 1
- Para arreglarlo, agregamos estado extra D



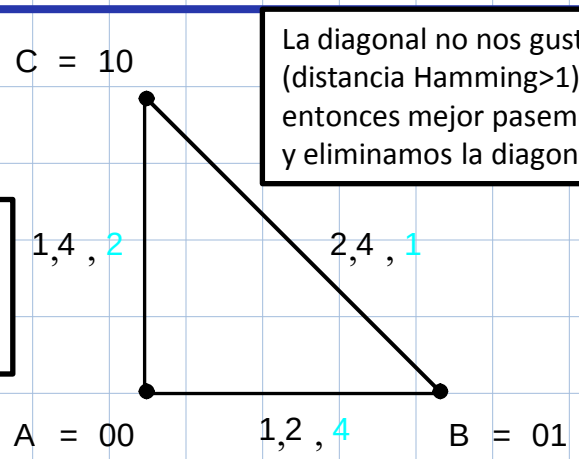
Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	(A)	B	C	—	00
B	A	(B)	D	(B)	01
C	A	D	(C)	(C)	10
D	—	B	C	—	10

(b) Using the extra state D

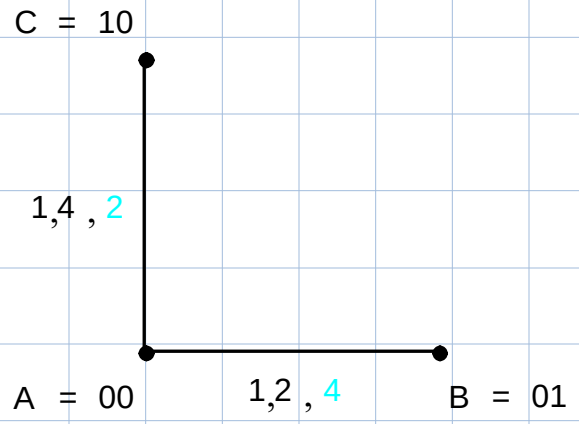
9.5.1 Diagrama de Transición



(a) Transitions in Figure 9.50



(b) Complete transition diagram



(c) Selected transition diagram

- Un diagrama que muestre los cambios de estados según la entrada se llama diagrama de transición
- Un diagrama bien diseñado para un circuito asíncrono no debería tener líneas diagonales (con distancia mayor que 1)
- El diagrama debiera caber en un cubo n-dimensional
  - n es el número de variables de estado

Present state	Next state $r_2 r_1$ = 00	01	10	11	Output $g_2 g_1$
A	(1)	2	4	—	00
B	1	(2)	4	(3)	01
C	1	2	(4)	(5)	10

(mismo que dos láminas atrás, pero etiquetando cada estado estable con etiqueta diferente)

9.5.1 Diagrama de Transición - Procedimiento

1. Determine la tabla de flujo re-etiquetada tal que, para un valor de la entrada, todas las transiciones que llevan al mismo estado estable tienen el mismo número. Las transiciones por estados inestables que llevan eventualmente a un estado estable llevan el mismo número que el estado estable.
2. Represente cada fila de la tabla de flujo por un vértice
3. Junte dos vértices V_i y V_j con una arista si tienen el mismo número en cualquier columna de la tabla de flujo re-etiquetada
4. Para cada columna en que V_i y V_j tienen el mismo número, etiquete la arista entre V_i y V_j con ese número. Usaremos etiquetas negras para transiciones directas a estados estables, y etiquetas azules cuando las entradas del siguiente estado para V_i y V_j en la tabla de flujo no son estables (sin círculo).

9.5.2 Aprovechando estados siguientes sin especificar

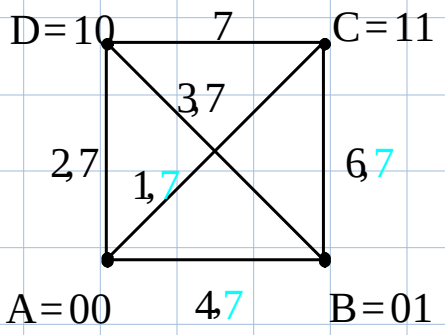
- Don’t cares en una tabla de flujo otorgan flexibilidad en una buena asignación de estados
- A continuación veremos un ejemplo con una posible metodología
- También veremos todos los pasos de un diagrama de transición
- Comencemos con la tabla de flujo
- Hay 7 estados estables
  - Etiquetemos cada uno de ellos con un número

Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	(A)	B	C	(A)	00
B	(B)	(B)	D	C	01
C	A	(C)	D	(C)	10
D	B	—	(D)	A	11

9.5.2 Aprovechando estados siguientes sin especificar

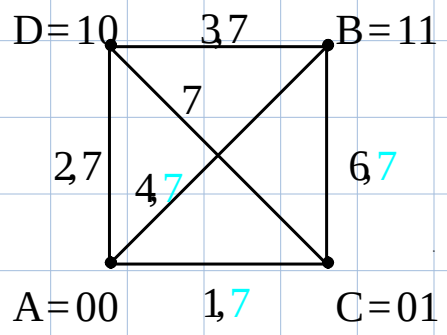
- Estados:
  - Estados 1 y 2 son el estado A
  - Estados 3 y 4 son el estado B
  - Estados 5 y 6 son el estado C
  - Estado 7 es el estado D
- Transiciones:
  - w_2w_1 = 00: C->A – transición 1 (estado estable sucesor)
  - w_2w_1 = 00: D->B – transición 3 (estado estable sucesor)
  - w_2w_1 = 11: B->C – transición 6 (estado estable sucesor)
  - w_2w_1 = 11: D->A – transición 2 (estado estable sucesor)
  - w_2w_1 = 01: A->B – transición 4 (estado estable sucesor)
  - w_2w_1: Estado C es estable (estado 5) y puede ser alcanzado desde el estado 6 estable
  - w_2w_1 = 10: estados inestables de paso:
    Por ejemplo, de 2 estable a 7 de transición (C) y luego D
- Veamos el diagrama de transiciones
  - Hay diagonales, no todas tienen distancia Hamming 1…
  - No podemos remover diagonales 1 y 3, no hay caminos alternativos
- Tratemos otra vez…

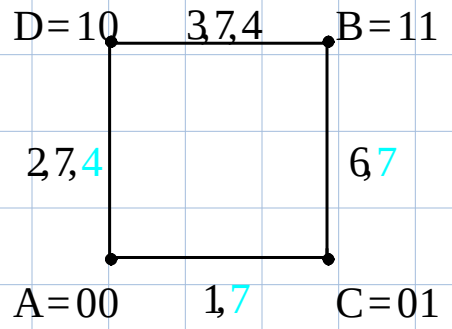
Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	(1)	4	7	(2)	00
B	(3)	(4)	7	6	01
C	1	(5)	7	(6)	10
D	3	—	(7)	2	11



9.5.2 Aprovechando estados siguientes sin especificar

- Invertimos los códigos de B y C
  - En este caso la diagonal 4 no puede ser eliminada…
- Pero podemos asignar un 4 al don't care de la tabla
  - Esto nos permite remover la diagonal
  - Esta asignación cabe en el cuadrado sin usar diagonales
  - Veamos cómo queda la tabla…





Present state	Next state $w_2 w_1$ = 00	01	10	11	Output $z_2 z_1$
A	(1)	4	7	(2)	00
B	(3)	(4)	7	6	01
C	1	(5)	7	(6)	10
D	3	—	(7)	2	11

9.5.2 Aprovechando estados-siguientes sin especificar

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output $z_2 z_1$ 00	01	10	11
A	(A)	D	D	(A)	00	00	11	00
B	(B)	(B)	D	C	01	01	11	01
C	A	(C)	B	(C)	–0	10	1–	10
D	B	B	(D)	A	–1	0–	11	00

(a) Modified flow table

	Present state $y_2 y_1$	Next state $w_2 w_1$ = 00	01	10	11	Output 00	01	10	11
		$Y_2 Y_1$				$z_2 z_1$			
A	00	(00)	10	10	(00)	00	00	11	00
B	11	(11)	(11)	10	01	01	01	11	01
C	01	00	(01)	11	(01)	–0	10	1–	10
D	10	11	11	(10)	00	–1	0–	11	00

(b) Excitation table

- En el don’t care original, ahora aparece el estado B
- El cambio de A a B para 01 pasa por D, así que ajustamos la tabla
  - Esto produce otro estado estable
- En 10, si estamos en C, va a D y puede hacerlo por A o B; elegimos B
- Esta tabla requiere movernos a modelo de Mealy y entonces debemos evitar glitches especificando bien las salidas

9.5.3 Asignación de estados ocupando variables de estado adicionales

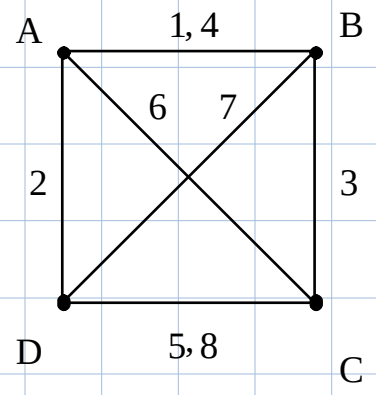
Present state	Nextstate w_2w_1 = 00	01	10	11	Output z_2z_1
A	(A)	(A)	C	B	00
B	A	(B)	D	(B)	01
C	(C)	B	(C)	D	10
D	C	A	(D)	(D)	11

(a) Flow table

Present state	Nextstate w_2w_1 = 00	01	10	11	Output z_2z_1
A	(1)	(2)	6	4	00
B	1	(3)	7	(4)	01
C	(5)	3	(6)	8	10
D	5	2	(7)	(8)	11

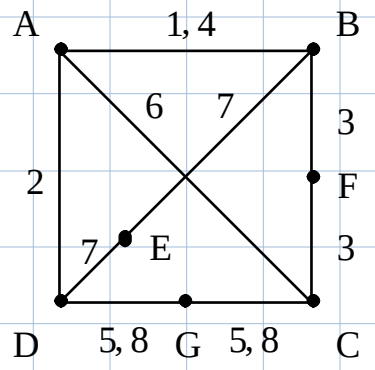
(b) Relabeled flow table

En este caso es imposible generar cambios a distancia Hamming 1 ocupando 2 o 3 variables de estado, y no tenemos don’t cares para aprovechar…

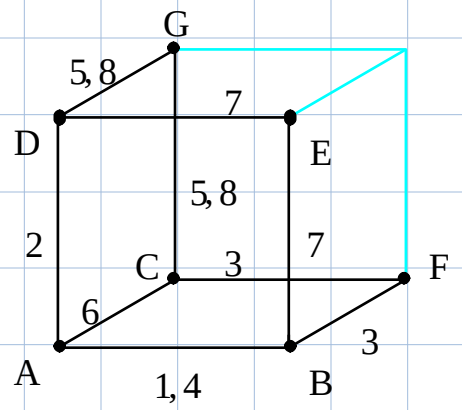


9.5.3 Asignación de estados ocupando variables de estado adicionales

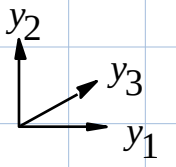
- La única opción es agregar una variable de estado adicional para pasar a un cubo
- Con 3 variables de estado, A queda a distancia Hamming 1 de B, C y D
- Igual es imposible tener pares (B, C), (B, D) y (C, D) a distancia 1…
  - Entonces agregamos estados adicionales
  - Vértices E, F y G cumplen el objetivo y son estados inestables



Esto puede ser representado por un cubo



En negro están los caminos considerados



9.5.3 Asignación de estados ocupando variables de estado adicionales

Present state	Nextstate w_2w_1 = 00	01	10	11	Output z_2z_1
A	(A)	(A)	C	B	00
B	A	(B)	E	(B)	01
C	(C)	F	(C)	G	10
D	G	A	(D)	(D)	11
E	–	–	D	–	–1
F	–	B	–	–	01
G	C	–	–	D	1–

(a) Modified flow table

	Present state $y_3y_2y_1$	Nextstate w_2w_1 = 00	01	10	11	Output z_2z_1
		$Y_3Y_2Y_1$				
A	000	(000)	(000)	100	001	00
B	001	000	(001)	011	(001)	01
C	100	(100)	101	(100)	110	10
D	010	110	000	(010)	(010)	11
E	011	–	–	010	–	–1
F	101	–	001	–	–	01
G	110	100	–	–	010	1–

(b) Excitation table

Por lo tanto, ocupando 3 variables de estado, se generan estados inestables que interconecten los que antes eran cambios "diagonales"

Otra opción correspondería a expandir estados ya existentes a dos estados equivalentes (A -> A1 y A2) tal de formar el cubo y mantener distancia Hamming de 1 – ver ejemplo 9.15 en el libro

9.5.4 Asignación de estados por codificación one-hot

Present state	Nextstate w_2w_1 = 00	01	10	11	Output z_2z_1
A	(A)	(A)	C	B	00
B	A	(B)	D	(B)	01
C	(C)	B	(C)	D	10
D	C	A	(D)	(D)	11

- Encontrar estados óptimos de distancia Hamming 1 puede ser complicado y tedioso
  - Ya lo hemos visto en este ejemplo…

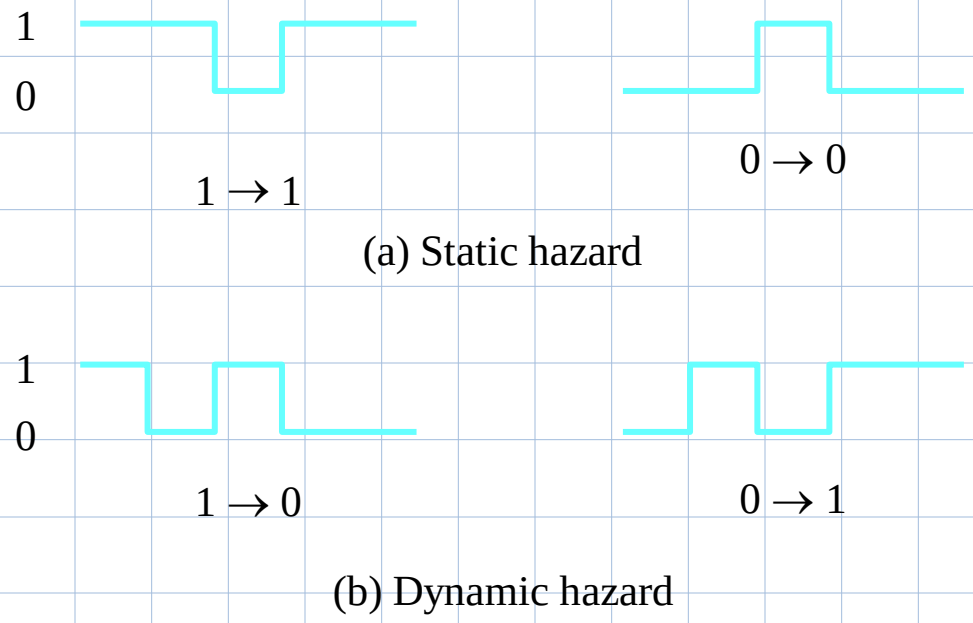
9.5.4 Asignación de estados por codificación one-hot

State assignment	Present State	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
0001	A	(A)	(A)	E	F	00
0010	B	F	(B)	G	(B)	01
0100	C	(C)	H	(C)	I	10
1000	D	I	J	(D)	(D)	11
0101	E	—	—	C	—	–0
0011	F	A	—	—	B	0–
1010	G	—	—	D	—	–1
0110	H	—	B	—	—	01
1100	I	C	—	—	D	1–
1001	J	—	A	—	—	00

- Una solución segura, aunque poco óptima, corresponde a codificar en one-hot cada estado estable, y crear estados inestables que mantengan esta distancia de 1 y enlacen estados estables
  - Ej: A = 0001, B = 0010, y hay un estado inestable que conecte ambos 0011

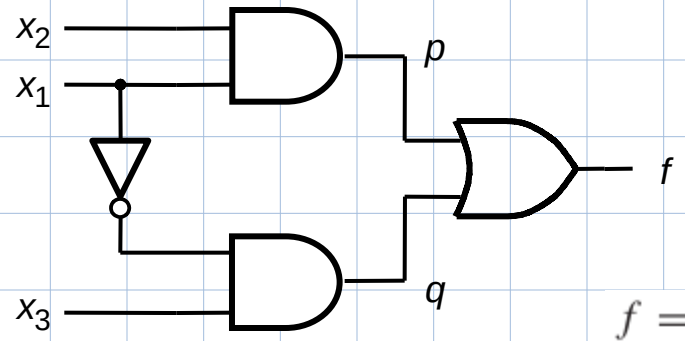
9.6 Riesgos

- Los riesgos son glitches producidos por la estructura de un circuito y los retardos de propagación
- Riesgo estático: Una variable se debería mantener en un valor, pero un cambio en la entrada produce un cambio breve de la variable a otro estado lógico
- Riesgo dinámico: Una variable oscila entre 0 y 1 antes de establecerse en su valor final
- Los circuitos asíncronos mal diseñados pueden producir riesgos



9.6.1 Riesgos estáticos

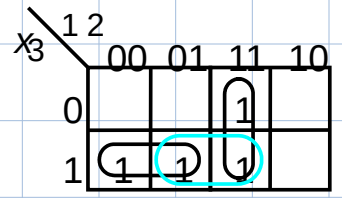
- El circuito de la figura sufre de un riesgo estático debido al retardo en la compuerta NOT



El cambio llega primero a p y luego a q

$$f = x_1x_2 + \bar{x}_1x_3$$

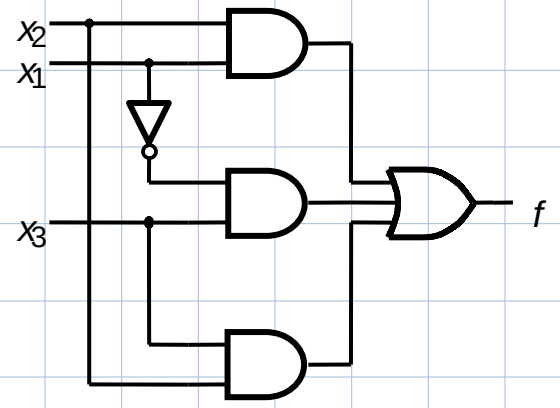
(a) Circuit with a hazard



(b) Karnaugh map

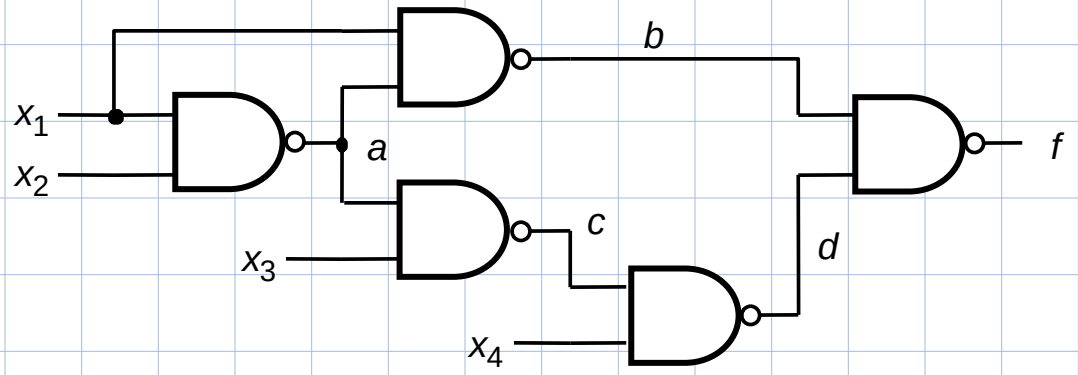
$$f = x_1x_2 + \bar{x}_1x_3 + x_2x_3$$

- Viendo el mapa de Karnaugh del circuito, se observa que éste implementa una solución óptima
- Si se busca eliminar este riesgo se debe implementar una solución no óptima
- En general, un riesgo aparece cuando dos 1's adyacentes no están cubiertos por un mismo minterm

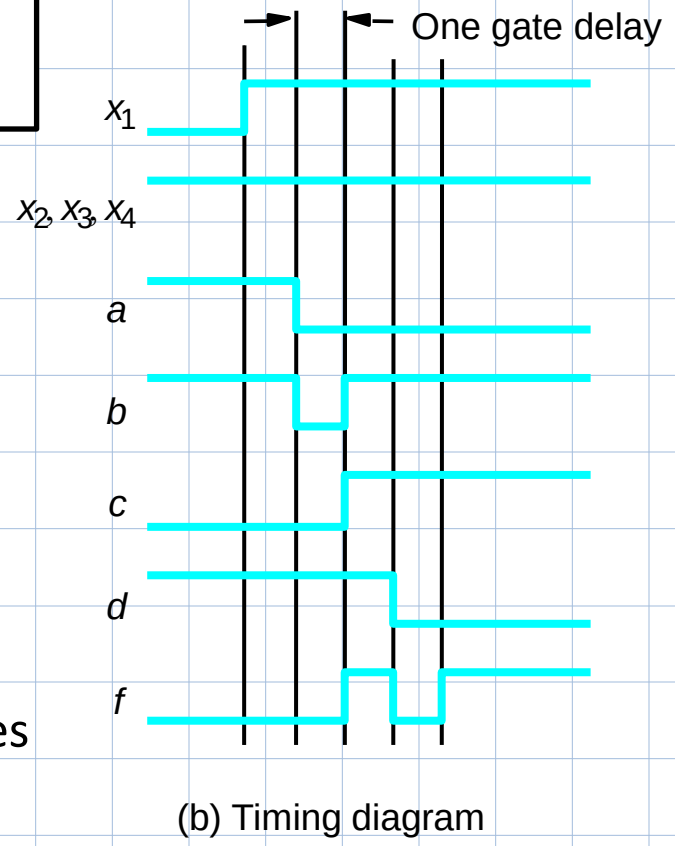


(c) Hazard-free circuit

9.6.2 Riesgos dinámicos



Un riesgo dinámico se produce cuando hay diferentes caminos para propagar un cambio particular



(b) Timing diagram

- El circuito de la figura tiene un riesgo dinámico
- Esto es observable al dibujar los diagramas de tiempo
  - Nótese en los diagramas que, de haber un riesgo dinámico, necesariamente existe un riesgo estático (en este caso para el nodo b)
- Los riesgos dinámicos no son fáciles ni de detectar ni de solucionar
- Suelen aparecer cuando se ocupan técnicas de factorización y descomposición lógica
- La forma más simple de solucionarlos es a través de circuitos de multiples niveles que esten libres de riesgos estáticos

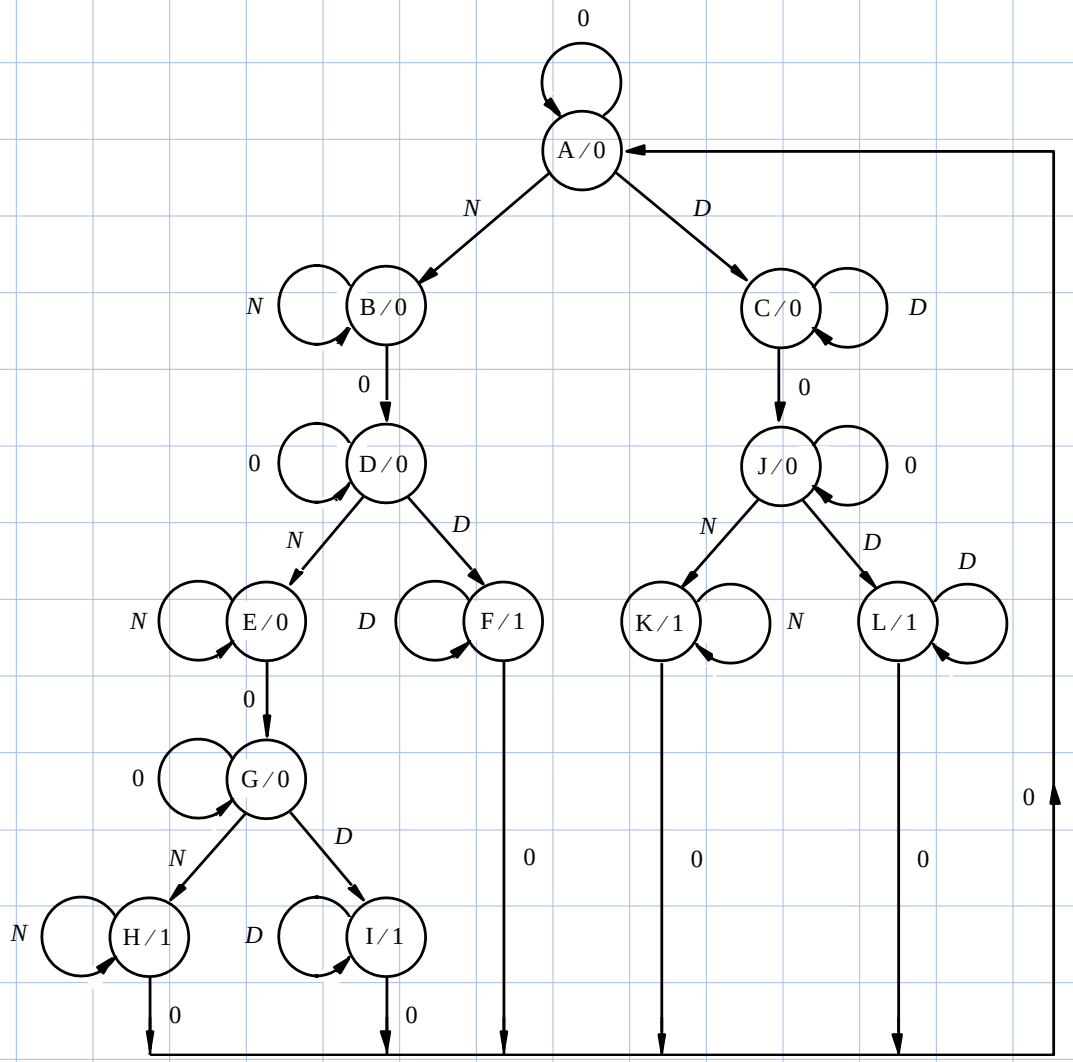
9.6.3 Importancia de riesgos

- Un riesgo puede hacer que un circuito asíncrono pase brevemente por estado incorrecto y estable, y se quede ahí
- En circuitos secuenciales hemos asumido que las entradas y variables de estado deben cambiar una vez
  - Entonces nos basta con eliminar riesgos debidos a una variable
- En los circuitos combinacionales la salida depende solamente de la entrada
- En circuitos secuenciales síncronos las entradas son estables respetando tiempo de setup y hold
  - Por este motivo los riesgos no son tan relevantes para estos casos si ocurren fuera del tiempo de setup y hold

9.7 Ejemplo de diseño completo

- Queremos diseñar un circuito para una máquina dispensadora con las siguientes especificaciones:
  - Acepta *nickels* y *dimes* (Monedas de 5 y 10 centavos de dolar respectivamente)
  - Entrega un dulce cuando recibe un total de 15 centavos
  - No entrega vuelto si recibe 20 centavos
- Se plantea como entrada una señal N y otra señal D, que representan un nickel y un dime siendo introducidos
  - nótese que es imposible que ambas señales se activen a la vez

9.7.1 Controlador para máquina expendedora



- Planteamos la FSM correspondiente
- El dulce es entregado en los estados F, H y K, al alcanzar los 15 centavos, y en los estados I y L, al alcanzar los 20 centavos

9.7.1 Controlador para máquina expendedora

Present state	Next state DN = 00	01	10	11	Output z
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	J	–	(C)	–	0
D	(D)	E	F	–	0
E	G	(E)	–	–	0
F	A	–	(F)	–	1
G	(G)	H	I	–	0
H	A	(H)	–	–	1
I	A	–	(I)	–	1
J	(J)	K	L	–	0
K	A	(K)	–	–	1
L	A	–	(L)	–	1

Inicial

Present state	Next state DN = 00	01	10	11	Output z
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	G	–	(C)	–	0
D	(D)	E	F	–	0
E	G	(E)	–	–	0
F	A	–	(F)	–	1
G	(G)	H	F	–	0
H	A	(H)	–	–	1

Primera minimización de estados

- A continuación se plantea una tabla de flujos, se plantea como reducción inicial:

P1 = (ADGJ)(BE)(C)(FIL)(HK)

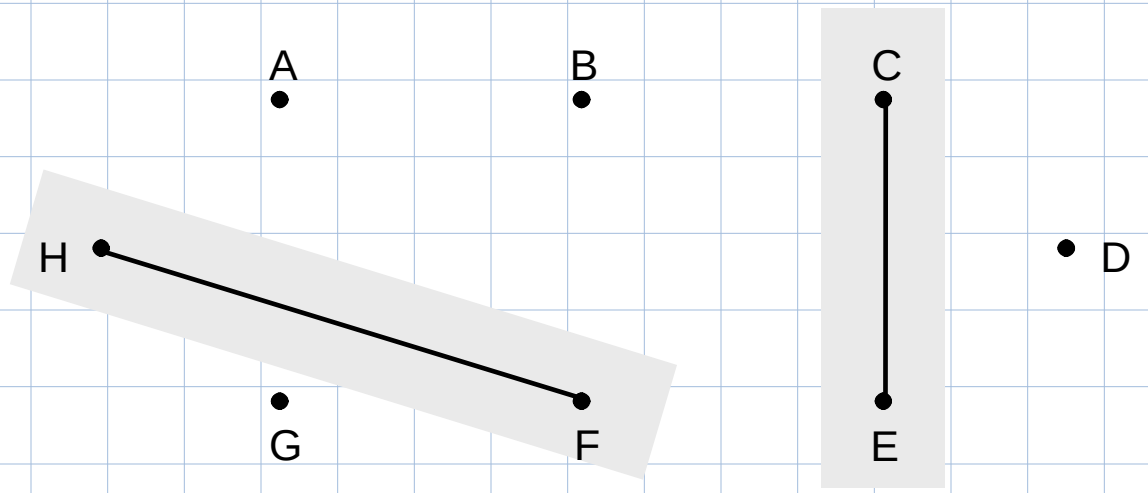
P2 = (A)(D)(GJ)(B)(E)(C)(FIL)(HK)

P3 = P2

- A partir de ahí se puede plantear una reducción de estados inicial

9.7.1 Controlador para máquina expendedora

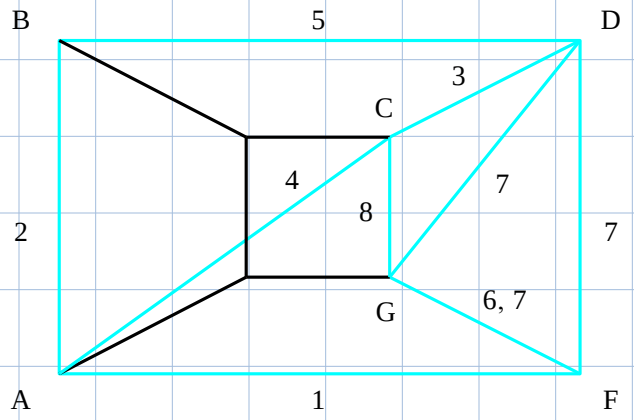
Luego dibujamos el diagrama de unión, vemos que hay más estados que pueden ser unidos:



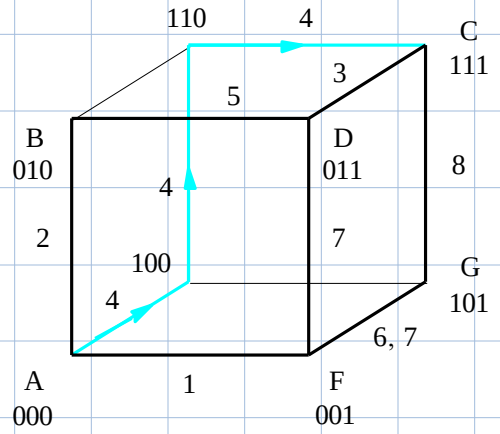
Present state	Next state				Output
	DN = 00	01	10	11	*z*
A	(A)	B	C	–	0
B	D	(B)	–	–	0
C	G	(C)	(C)	–	0
D	(D)	C	F	–	0
F	A	(F)	(F)	–	1
G	(G)	F	F	–	0

(a) Minimized flow table

9.7.1 Controlador para máquina expendedora



(a) Transition diagram



(b) Embedded on the cube

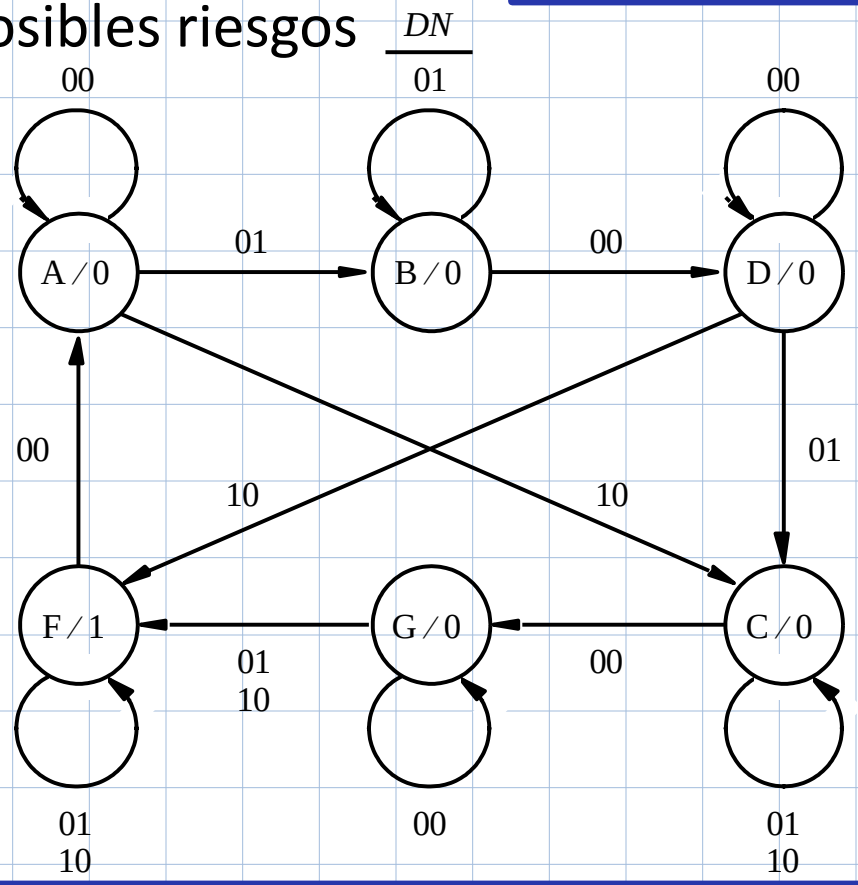
	Present state $y_3y_2y_1$	Nextstate $DN = 00$	01	10	11	Output z
		$Y_3Y_2Y_1$				
A	000	(000)	010	100	–	0
B	010	011	(010)	–	–	0
C	111	101	(111)	(111)	–	0
D	011	(011)	111	001	–	0
F	001	000	(001)	(001)	–	1
G	101	(101)	001	001	–	0
	100	–	–	110	–	0
	110	–	–	111	–	0

- Ahora procedemos con la asignación de estados a través del diagrama de transición
- Hay dos caminos diagonales:
  - D a G (no relevante, pues es un camino alternativo)
  - A a C, conectados a través de estados sin ocupar
- De esta forma podemos asignar estados y completar la tabla de excitación

9.7.1 Controlador para máquina expendedora

- Finalmente, hacemos los mapas de Karnaugh (ver en el libro) y llegamos al diseño final para el circuito
- Hay algunos términos extras para evitar posibles riesgos

$$Y_1 = \overline{N}y_2 + Ny_1 + Dy_1 + y_1y_3 + y_1y_2$$
$$Y_2 = N\overline{y}_1 + Ny_2 + \overline{y}_1y_3 + \overline{D}y_2\overline{y}_3 + Dy_2y_3$$
$$Y_3 = D\overline{y}_1 + y_2y_3 + Ny_1y_2 + \overline{D}y_3\overline{N}$$



9.8 Conclusiones

- La verdad es que todos los circuitos secuenciales son asíncronos, pero al emplear un reloj síncrono los sincronizamos
- Para diseñar circuitos asíncronos no se disponen de tantas herramientas CAD como es en el caso de circuitos síncronos
  - Por este motivo preferimos diseñar circuitos síncronos
- En general, los circuitos asíncronos son mas convenientes para algunas aplicaciones pues no dependen de un reloj y funcionan más rápido
- También son útiles para observar sistemas síncronos que funcionen con relojes de distinta frecuencia
  - permite a algunas secciones operar más rápido
- El análisis también es útil para comprender el funcionamiento de circuitos secuenciales, y permite entender conceptos como estado estable y la importancia de los riesgos