UNIVERSIDAD POLITECNICA DE VALENCIA

etsinf

DISCA

Fundamentos de los computadores

Tema 5. SISTEMAS SECUENCIALES (Segunda Parte)

Objetivos

- Primera parte
  - Conocer los sistemas secuenciales básicos más importantes
    - Registros, Banco de registros, Memoria, Contadores
  - Aprender a representar formalmente un sistema secuencial síncrono sencillo a partir de una especificación
  - Analizar el comportamiento de un sistema secuencial
    - Para determinar las salidas del mismo
- Segunda parte
  - Conocer la metodología de diseño de los sistemas secuenciales síncronos
    - Para obtener un circuito a partir de una especificación

Índice de la segunda parte

- Diseño de autómatas de Moore
  - Objetivos
  - Etapas del diseño
    - Formalización
      - Diagrama de estados
      - Tabla de estados
    - Codificación
    - Cálculo de las funciones de excitación y salida
    - Circuito final, reuniendo las funciones de excitación y salida con las variables de estado en un autómata de Moore
- Ejemplos de diseño de autómatas de Moore

Diseño de autómatas de Moore

FCO

- Objetivos del diseño
  - Transformar unos requerimientos en lenguaje natural en un autómata de Moore (un tipo de SSS)
  - El reloj juega un papel fundamental
    - El sistema evoluciona en función del reloj
    - Se dice que el reloj es "*free-running*" (siempre en marcha)
  - Ejemplos:
    - Diseñar un SSS que active una salida S durante 3 ciclos de reloj en cuanto se detecte un cambio de valor cualquiera en una entrada A
    - Diseñar un SSS que cuente los ciclos de reloj que pasan desde que se activa una entrada A hasta que esta entrada se desactiva (limitando el número máximo de ciclos)

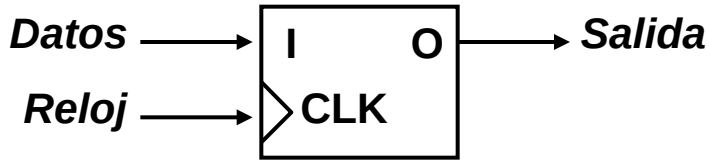
Diseño (etapas): Formalización

FCO

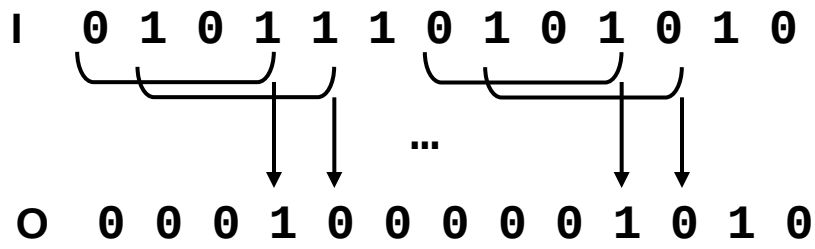
- El primer paso para diseñar un circuito es entender el problema a resolver a partir de la especificación
  - Determinar las entradas y salidas nos permite diseñar la interfaz del circuito
  - Entender el comportamiento nos va a permitir obtener una representación formal del mismo
    - Representaciones conocidas
      - Diagrama de estados
      - Tabla de estados
    - La tabla de estados es el punto de partida para llegar al circuito
      - A partir del diagrama de estados
      - Directamente de la especificación

Diseño (etapas): Formalización (ii)

- Ejemplo de diseño con diagrama de estados:
  - Detección de un patrón
    - Típico problema a resolver en comunicaciones de alta velocidad
      - El patrón corresponde a la marca de inicio de trama
  - En el sistema a diseñar llega un dato por ciclo de reloj a una entrada I. El sistema debe activar una salida O durante un ciclo de reloj cuando se detecte en la entrada el patrón 0101. El flanco activo del reloj es el flanco de subida
    - Interfaz



Ejemplo de evolución



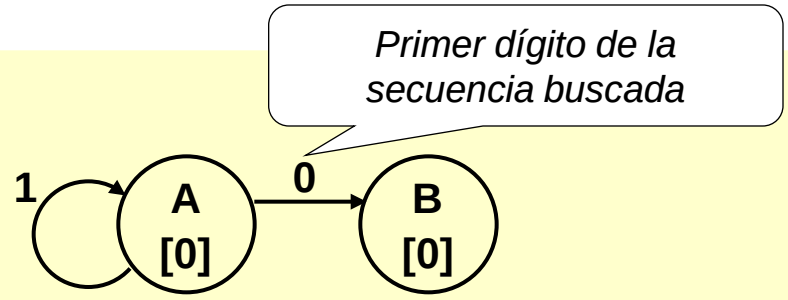
Diseño (etapas): Formalización (iii) FCO

- Ejemplo de diseño de diagrama de estados (cont.):
  - Necesitaremos varios estados
    - Como inicialmente no sabemos cuántos estados hacen falta ni qué información deben almacenar, utilizaremos nombres simbólicos
    - Un estado inicial o de reposo
      - Le llamaremos estado A, y la salida en dicho estado será 0
    - Un estado cuando se haya detectado el primer dígito de la secuencia, otro para el segundo, etc.
      - El sistema ha recibido 0 (estado B, salida 0)
      - El sistema ha recibido 0 – 1 (estado C, salida 0)
      - El sistema ha recibido 0 – 1 – 0 (estado D, salida 0)
      - El sistema ha recibido 0 – 1 – 0 – 1 (estado E, salida 1)

Diseño (etapas): Formalización (iv) FCO

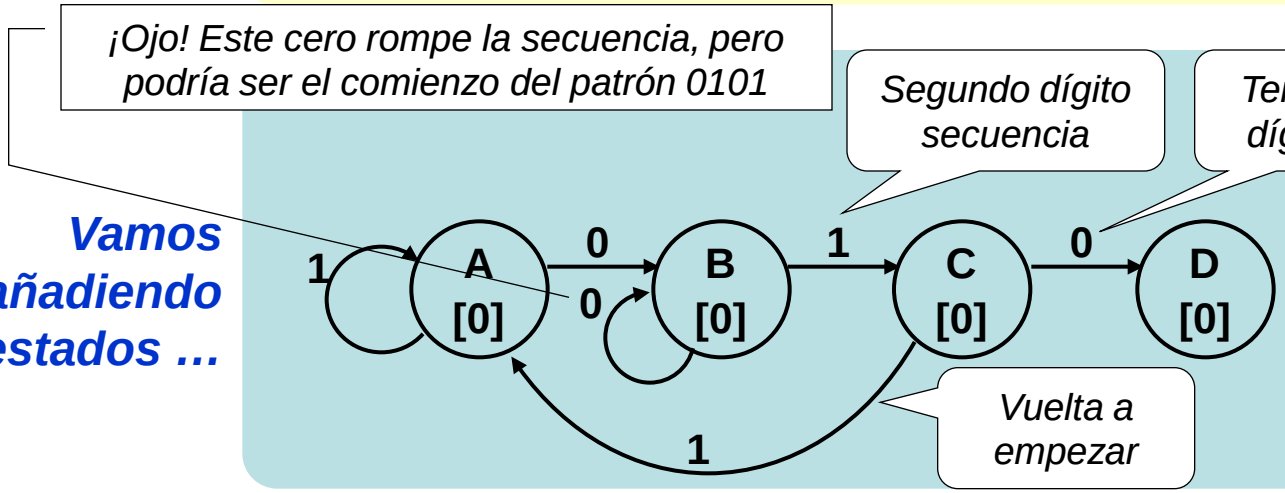
- Ejemplo de diseño de diagrama de estados (cont.):
  - Diagrama de estados (utilizando vectores)

Inicio …



Dibujamos el primer estado (A - "inicial") y todas las transiciones que salen del mismo, para lo que necesitamos incluir un segundo estado B ("detectado primer dígito")

Vamos añadiendo estados …

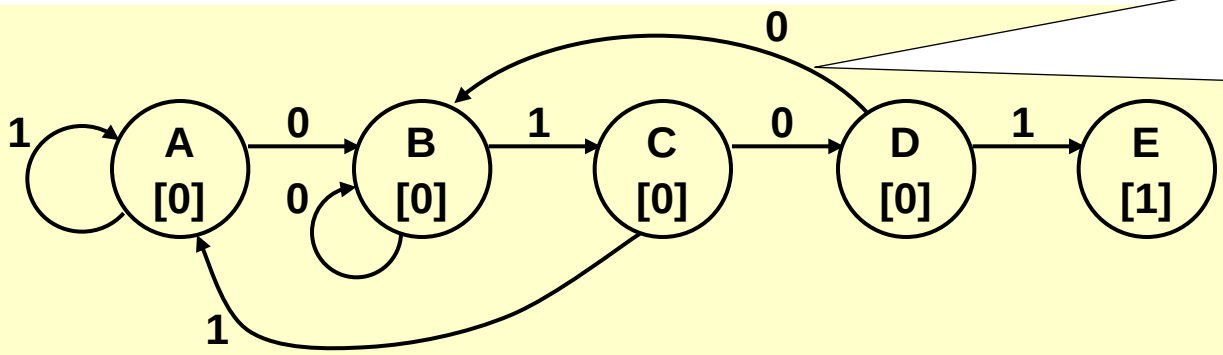


Añadimos estados (C y D, "detectado segundo dígito" y "detectado tercer dígito", respectivamente) a medida que dibujamos las transiciones que salen de los estados que ya tenemos

Diseño (etapas): Formalización (v) FCO

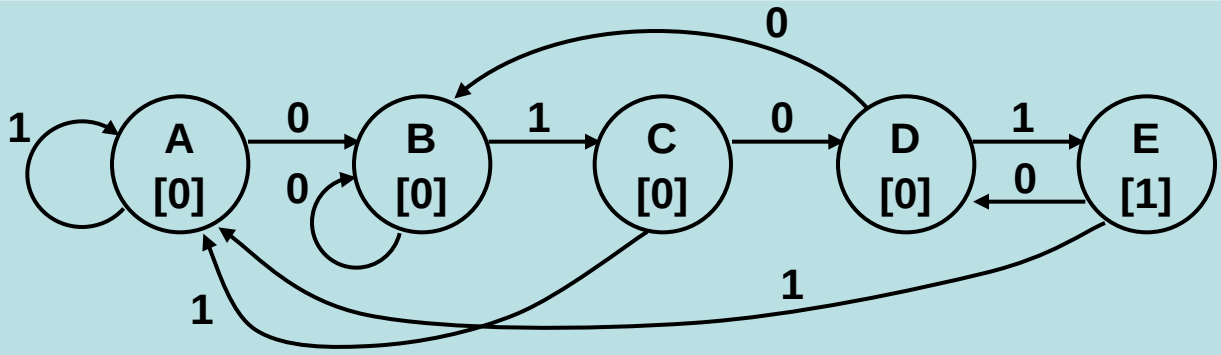
- Ejemplo de diseño de diagrama de estados (cont.):
  - Diagrama de estados (utilizando vectores)

Más estados…



¡Ojo! Este cero rompe la secuencia, pero podría ser el comienzo del patrón 0101. Por eso la transición es al estado B (“primer dígito encontrado”) y no al A

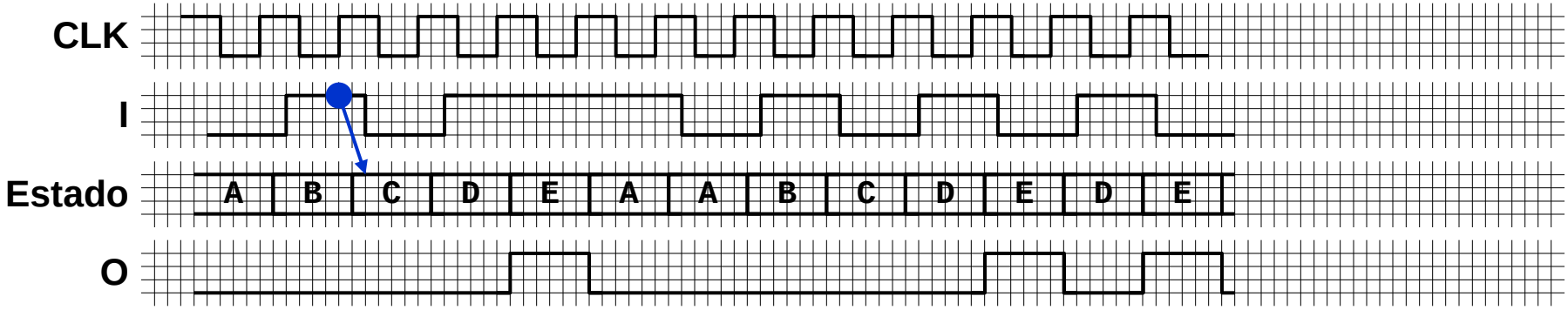
Diagrama final



Hemos finalizado cuando todos los estados tienen todas sus transiciones de salida

Diseño (etapas): Formalización (vi) FCO

- Ejemplo de diseño de diagrama de estados (cont.):
  - Nótese que hay un flanco de reloj que separa el cambio en la entrada I y el cambio en la salida O
    - **Latencia del sistema**
    - Imposible de evitar con un autómata de Moore
      - La salida sólo depende del estado
      - Para cambiar la salida debe cambiar el estado del autómata
      - Y el estado cambia con el flanco de reloj

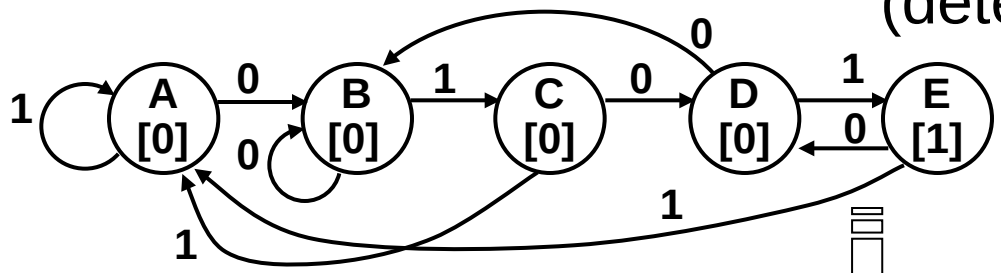


Diseño (etapas): Formalización (vii) FCO

- Tabla de estados
  - Obtenida directamente de la especificación o a partir del diagrama de estados
  - Generalmente utilizaremos nombres simbólicos en esta etapa del diseño
    - Aún no sabemos cuántas variables de estado
    - Ni tenemos asignados valores a los estados
    - Esto se resuelve en la etapa siguiente (codificación de estados), donde se sustituyen los nombres simbólicos por valores binarios

Diseño (etapas): Formalización (viii) FCO

- Ejemplo de diseño de tabla de estados
  - A partir del diagrama de estados del ejemplo anterior (detección del patrón 0101)



Estado actual (Q(t))	Estado siguiente (Q(t+1)) I = 0 ($\overline{I}$)	Estado siguiente (Q(t+1)) I = 1 (I)	Salidas O
A	B	A	0
B	B	C	0
C	D	A	0
D	B	E	0
E	D	A	1

Tabla de estados utilizando nombres simbólicos para los estados.

Diseño (etapas): Codificación FCO

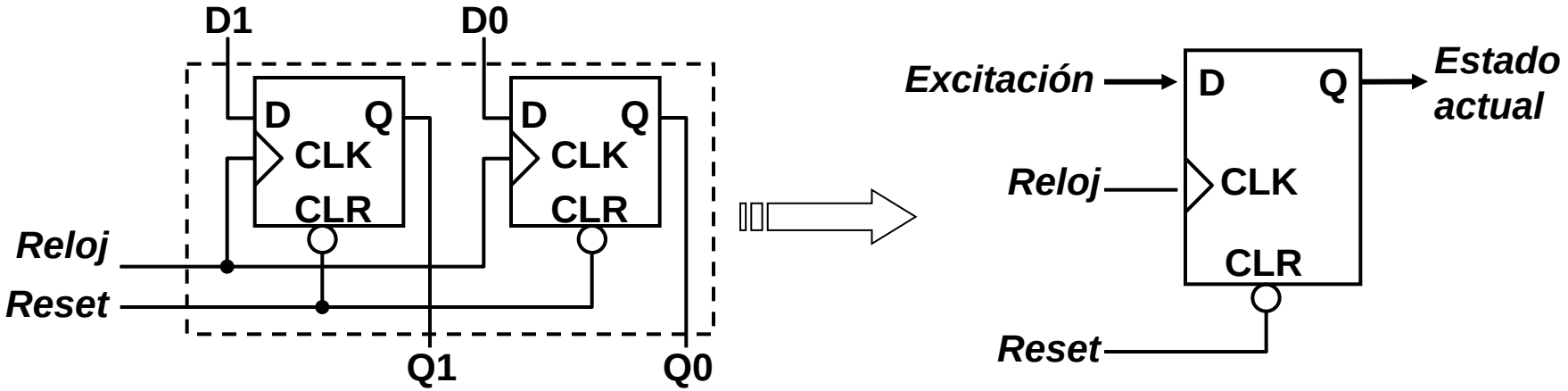
- Codificación de estados (cont.)
  - Determinar el número de variables de estado necesarias y asignar un código binario a cada estado
    - El mínimo es *ceil (log_2 (número estados))*
  - Opciones de **codificación compactas**
    - Utilizan el menor número posible de variables de estado
    - Ejemplos de codificación: binario, código grey
    - Suelen quedar códigos sin utilizar
      - Ejemplo: con 5 estados necesitamos 3 variables. Disponemos de 8 códigos binarios de los que sólo empleamos 5
      - ¿Qué pasa si las variables de estado se inician con un código no empleado?

Diseño (etapas): Codificación (ii) FCO

- Codificación de estados (cont).
  - Opciones de **codificación no compactas**
    - El objetivo puede ser, por ejemplo, simplificar las funciones de transición o salida
    - Ejemplo de este tipo de codificación: "***one-hot***"
      - Tantas variables de estado como estados
      - Sólo una variable vale 1 en cada código utilizado
      - Códigos para 5 estados: 00001, 00010, 00100, 01000, 10000
    - Por su construcción quedan (muchos) códigos sin utilizar
- Tabla de estados con los estados codificados
  - A partir de la tabla de estados con nombres simbólicos y la codificación escogida

Diseño (etapas): Codificación (iii) FCO

- Códigos sin utilizar ¿cómo evitarlos?
  - Se suelen emplear entradas asíncronas en los biestables
    - Para garantizar un determinado código binario al inicio
  - Se necesita una entrada adicional de inicialización (se suele denominar *"reset"*) en los elementos de memoria del estado
    - Ejemplo: El estado inicial (tras la activación de reset) es el valor 00



Diseño (etapas): Codificación (iv) FCO

- Problema de optimización
  - Existen múltiples opciones de codificación de estados
  - Para un tipo de codificación concreto, existen múltiples asignaciones distintas
    - Ejemplo: Cinco estados (A, B, C, D y E), codificación binaria (3 bits)
      - A = 000, B = 001, C = 010, D = 011, E = 100
      - A = 000, B = 001, C = 010, D = 011, E = 101
      - …
  - No hay un método de obtener la codificación óptima
  - Solución por fuerza bruta (¡que nunca haremos!)
    - Calcular las funciones de excitación y salida para cada codificación
    - Comparar los resultados y seleccionar la óptima

Diseño (etapas): Codificación (v) FCO

- Ejemplo de codificación de estados
  - A partir del ejemplo anterior (detección del patrón 0101)
  - Codificación binaria de 3 bits, ordenada (A=0, B=1, C=2, …)

	Estado siguiente (Q(t+1))		Salidas
Estado actual (Q(t))	I = 0 ($\overline{I}$)	I = 1 (I)	O
A	B	A	0
B	B	C	0
C	D	A	0
D	B	E	0
E	D	A	1

Estado	Codificación (Q2Q1Q0)
A	000
B	001
C	010
D	011
E	100

	Q2(t+1) Q1(t+1) Q0(t+1)		Salidas
Q2(t) Q1(t) Q0(t)	I = 0 ($\overline{I}$)	I = 1 (I)	O
000	001	000	0
001	001	010	0
010	011	000	0
011	001	100	0
100	011	000	1

Tabla de estados con los estados codificados.
Nos va a permitir obtener las tablas de verdad de las funciones de excitación y salidas

Diseño (etapas): Funciones

FCO

- Funciones de salida y excitación
  - Funciones combinacionales
    - Se implementan por métodos ya conocidos
    - A partir de la tabla de estados con los estados codificados
  - Función de salida
    - Calcula las salidas del sistema a partir del estado (Moore)
      - Tabla de verdad con los estados para determinar las salidas
  - Función de excitación
    - Calcula las entradas síncronas de los elementos de memoria que almacenan el estado, a partir del estado y de las entradas del sistema
      - Tabla de verdad con los estados y las entradas del sistema para determinar las entradas de los biestables (trabajando con biestables D, coinciden con el valor del estado siguiente)

Diseño SSS: Metodología – funciones (ii)

FCO

- Ejemplo de cálculo de las funciones de excitación y salida (del ejemplo anterior, detección patrón 0101)
  - Codificación binaria de 3 bits, ordenada

	Q2(t+1) Q1(t+1) Q0(t+1)		Salidas
Q2(t) Q1(t) Q0(t)	I = 0 ($\overline{I}$)	I = 1 (I)	O
000	001	000	0
001	001	010	0
010	011	000	0
011	001	100	0
100	011	000	1

Tabla de estados con los estados codificados.

Tabla de verdad de la función de salida. A partir de los valores de Q2Q1Q0, obtener O

Tabla de verdad de la función de excitación. A partir de los valores de Q2Q1Q0 e I, obtener D2, D1, D0 (teniendo en cuenta que Di = Qi(t+1))

Diseño SSS: Metodología – funciones (iii)

- Ejemplo de funciones de excitación y salida (cont.)

	Q2(t+1) Q1(t+1) Q0(t+1)		Salidas
Q2(t) Q1(t) Q0(t)	I = 0 ($\overline{I}$)	I = 1 (I)	O
000	001	000	0
001	001	010	0
010	011	000	0
011	001	100	0
100	011	000	1

Tabla de estados con los estados codificados.

Q2	Q1	Q0	O
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	x
1	1	0	x
1	1	1	x

$$O = Q2(t)$$

Q2	Q1	Q0	I	D2	D1	D0
0	0	0	0	0	0	1
0	0	0	1	0	0	0
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	0	1
0	1	1	1	1	0	0
1	0	0	0	0	1	1
1	0	0	1	0	0	0
1	0	1	x	x	x	x
1	1	0	x	x	x	x
1	1	1	x	x	x	x

*Tablas de verdad de las funciones de salida y excitación. **Nótese** cómo las combinaciones binarias que **no están presentes** en la tabla de estados codificados son **casos imposibles** en estas tablas de verdad.*

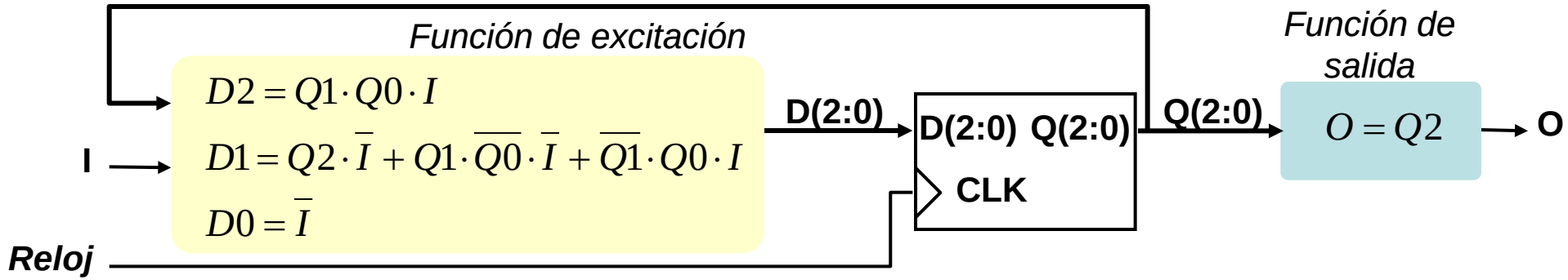
$$D2 = Q1 \cdot Q0 \cdot I$$

$$D1 = Q2 \cdot \overline{I} + Q1 \cdot \overline{Q0} \cdot \overline{I} + \overline{Q1} \cdot Q0 \cdot I$$

$$D0 = \overline{I}$$

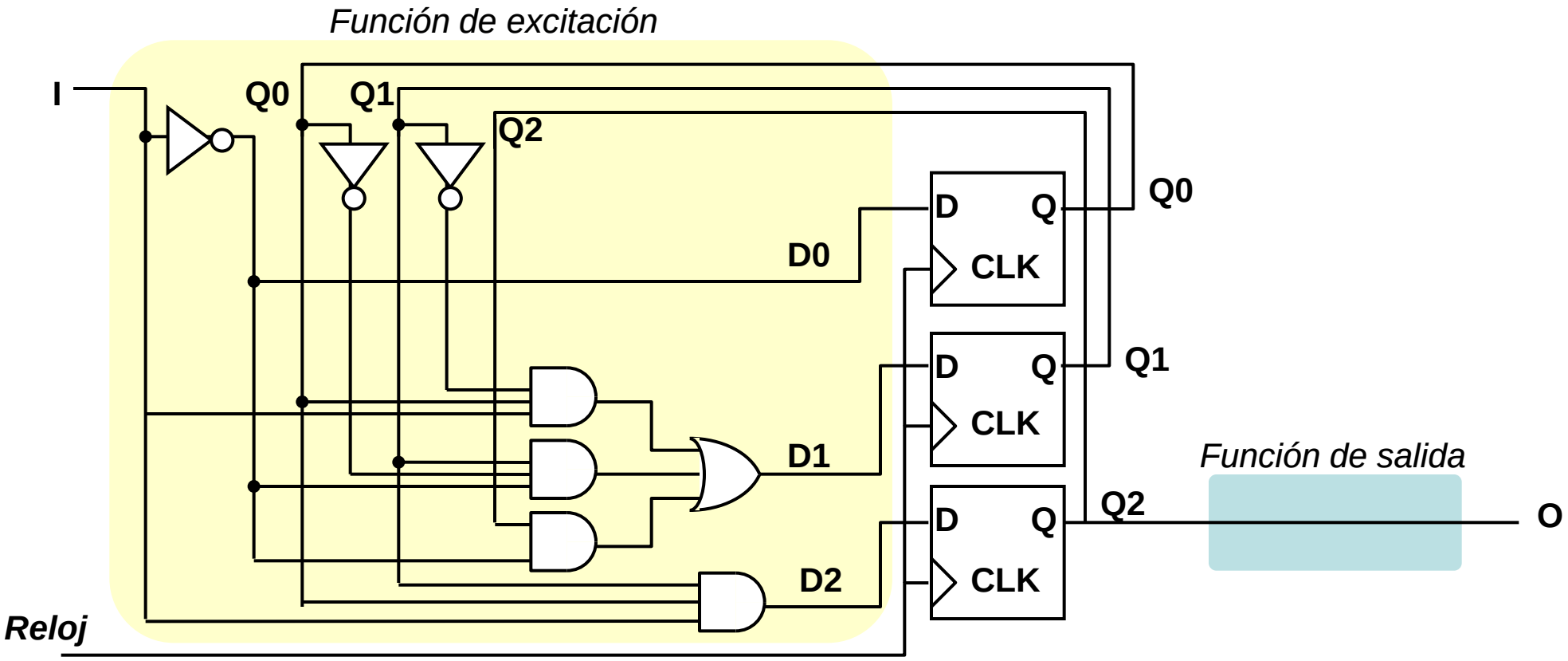
Diseño SSS: Metodología – circuito FCO

- Diseñar el circuito
  - Reunimos y conectamos las piezas siguiendo la estructura de un autómata de Moore
    - Las variables de estado almacenan el estado
    - La función de excitación calcula el estado siguiente
    - La función de salida calcula la salida del sistema
  - Esquema de bloques del circuito del ejemplo anterior (detección del patrón 0101)



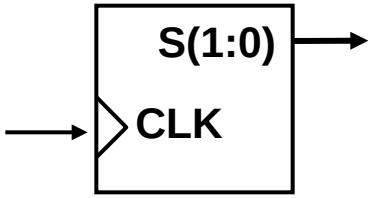
Diseño SSS: Metodología – circuito (ii) FCO

- Ejemplo de diseño del circuito
  - A partir del ejemplo anterior (detección del patrón 0101)

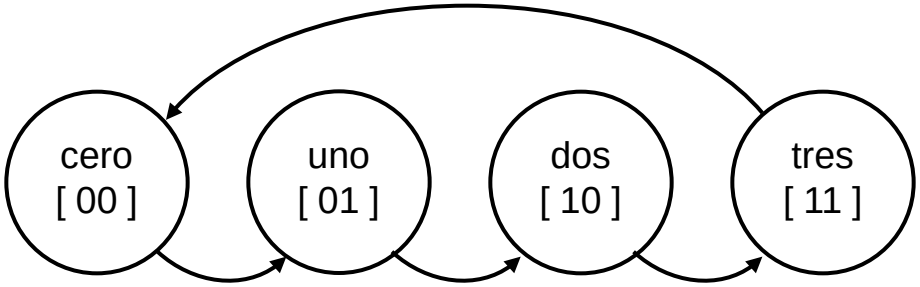


Ejemplos de diseño – 1.- Contador FCO

- Ejemplo 1: Contador binario ascendente (2 bits)
  - La cuenta puede empezar en cualquier valor
  - La salida debe ser 00 – 01 – 10 – 11 – 00 – …
  - No hay entradas (aparte del reloj)
  - Interfaz



  - Diagrama de estados
    - Transiciones sin ecuación, porque siempre (que llega el flanco de reloj) cambiamos de estado



Ejemplos de diseño – 1.- Contador (ii) FCO

- Ejemplo 1 (cont.)
  - Una buena codificación (muy obvia) nos permite eliminar la función de salida

Estado actual	Estado siguiente	Salida S1S0
cero	uno	00
uno	dos	01
dos	tres	10
tres	cero	11

Estado actual	Codificación (Q1Q0)
cero	**00**
uno	**01**
dos	**10**
tres	**11**

Q1(t)	Q0(t)	Q1(t+1)	Q0(t+1)	S1	S0
0	0	0	1	0	0
0	1	1	0	0	1
1	0	1	1	1	0
1	1	0	0	1	1

¿Porqué no se parece este circuito al que ya conocemos de contador de dos bits?

$$D1 = \overline{Q1} \cdot Q0 + Q1 \cdot \overline{Q0}$$

$$D0 = \overline{Q0}$$

$$S1 = Q1$$

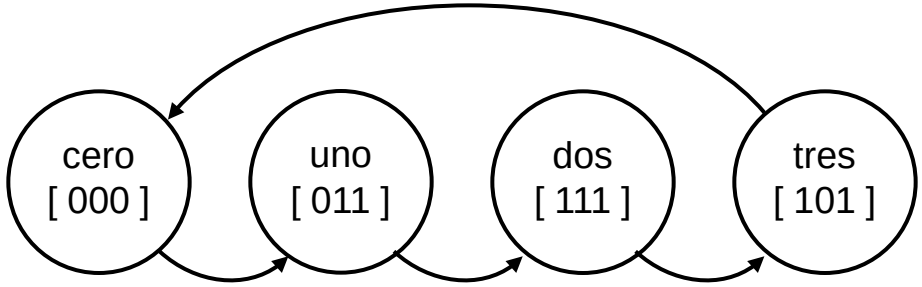
$$S0 = Q0$$

Q1	Q0	S1	S0
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1

Q1	Q0	D1	D0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

Ejemplos de diseño – 2.- Contador FCO

- Ejemplo 2: Contador 0 – 3 – 7 – 5 – 0 – …
  - Funcionamiento similar al contador binario, excepto en la salida
  - Diagrama de estados



  - Dos codificaciones rápidamente identificables
    - Con dos variables de estado y con función de salida
      - Calculando la salida (3 bits) en función del estado (2 bits)
    - Con tres variables de estado y sin función de salida
      - Haciendo coincidir la codificación del estado y la salida
      - ¿Porqué se puede emplear esta codificación en este caso?

Ejemplos de diseño – 2.- Contador (ii) FCO

- Ejemplo 2 (cont.)
  - Codificación con **dos variables de estado**
    - Obviamos la tabla de estados con los estados codificados
    - Nota: Que S1 sea igual a D1 es una coincidencia

Estado actual	Estado siguiente	Salida S2S1S0
cero	uno	0 0 0
uno	dos	0 1 1
dos	tres	1 1 1
tres	cero	1 0 1

Estado actual	Codificación (Q1Q0)
cero	**00**
uno	**01**
dos	**10**
tres	**11**

Q1	Q0	D1	D0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

$$D1 = \overline{Q1} \cdot Q0 + Q1 \cdot \overline{Q0}$$

$$D0 = \overline{Q0}$$

Q1	Q0	S2	S1	S0
0	0	0	0	0
0	1	0	1	1
1	0	1	1	1
1	1	1	0	1

$$S2 = Q1$$

$$S1 = \overline{Q1} \cdot Q0 + Q1 \cdot \overline{Q0}$$

$$S0 = Q1 + Q0$$

Ejemplos de diseño – 2.- Contador (iii) FCO

- Ejemplo 2 (cont.)
  - Codificación con **tres variables de estado**
    - Existen códigos sin utilizar, lo que permite reducir la complejidad de las ecuaciones resultantes

Estado actual	Estado siguiente	Salida S2S1S0
cero	uno	0 0 0
uno	dos	0 1 1
dos	tres	1 1 1
tres	cero	1 0 1

Estado actual	Codificación (Q2Q1Q0)
cero	**0 0 0**
uno	**0 1 1**
dos	**1 1 1**
tres	**1 0 1**

Q2	Q1	Q0	D2	D1	D0
0	0	0	0	1	1
0	0	1	x	x	x
0	1	0	x	x	x
0	1	1	1	1	1
1	0	0	x	x	x
1	0	1	0	0	0
1	1	0	x	x	x
1	1	1	1	0	1

$$D2 = Q1$$

$$D1 = \overline{Q2}$$

$$D0 = \overline{Q2} + Q1$$

Q2	Q1	Q0	S2	S1	S0
0	0	0	0	0	0
0	0	1	x	x	x
0	1	0	x	x	x
0	1	1	0	1	1
1	0	0	x	x	x
1	0	1	1	0	1
1	1	0	x	x	x
1	1	1	1	1	1

$$S2 = Q2$$

$$S1 = Q1$$

$$S0 = Q0$$

Ejemplos de diseño – 3.- Genérico FCO

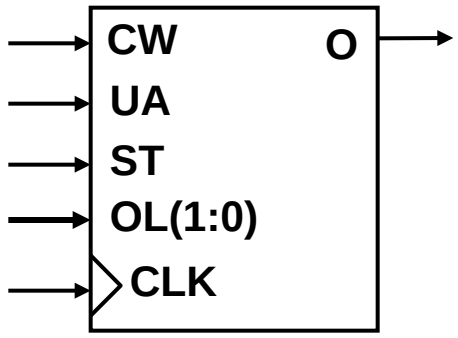
- Ejemplo 3: Barrera de entrada a un parking UPV
  - Simplificación
    - Sólo tratamos la apertura de la barrera de entrada al parking (no se contabilizan coches o leen los carnets UPV, eso lo hace otra parte del sistema)
  - Entradas (activas a nivel alto)
    - CW (*"Car Waiting"*): Se activa mientras hay un coche a la espera
    - UA (*"User Authenticated"*): Se activa si el conductor dispone de carnet UPV
    - ST (*"Student/Staff"*): El carnet UPV presentado (suponiendo UA activa) es de estudiante (ST = 0) o personal (ST = 1)
    - OL(1:0) (*"Occupation Level"*): Indica el nivel de ocupación del parking
      - Libre (00), Medio (01), Congestionado (10), Lleno total (11)

Ejemplos de diseño – 3.- Genérico (ii) FCO

- Ejemplo 3 (cont.)
  - Descripción del funcionamiento
    - En caso de que el parking esté libre, abrir en cuanto haya un coche a la espera
    - En caso de que la ocupación indique que el parking está medianamente ocupado, abrir únicamente a usuarios con carnet UPV
    - En caso de congestión, abrir únicamente al personal UPV
    - En caso de lleno total, no abrir

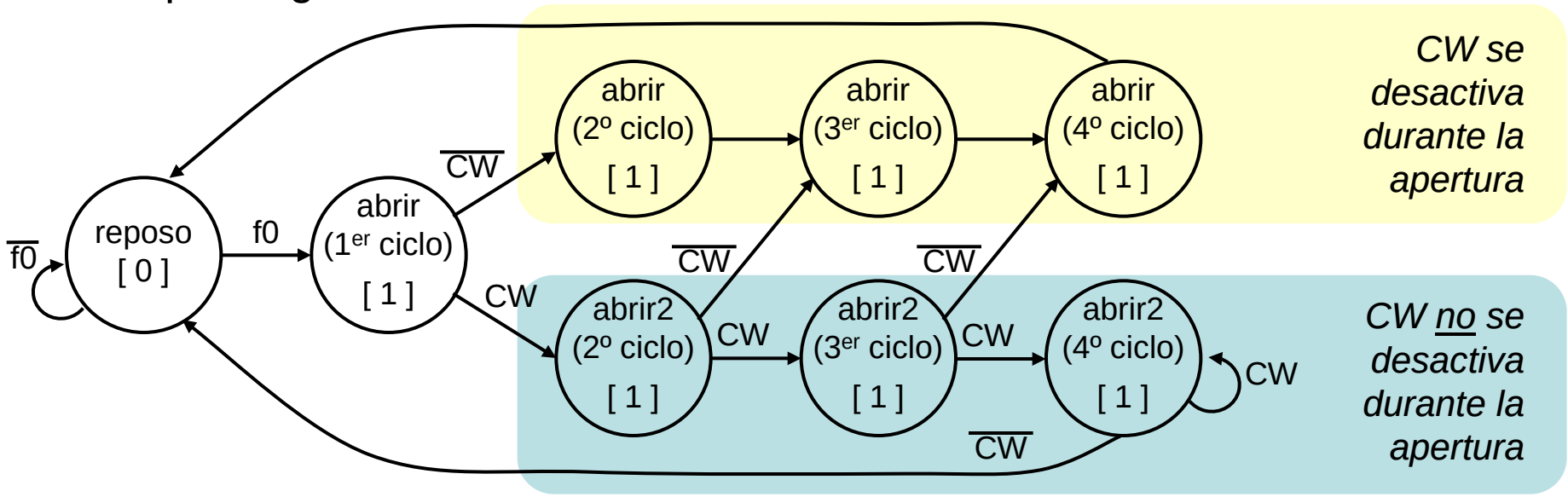
Ejemplos de diseño – 3.- Genérico (iii) FCO

- Ejemplo 3 (cont.)
  - Salida (activa a nivel alto)
    - O (*"Open"*): Activar para abrir la barrera de entrada al parking
      - Apertura mínima durante 4 ciclos de reloj
      - Si durante estos 4 ciclos de reloj el coche sigue a la espera, mantener abierto hasta que deje de estarlo
        - » *Si durante este tiempo CW = 0, una futura activación de CW significa que se trata de otro coche*
  - Interfaz



Ejemplos de diseño – 3.- Genérico (iv) FCO

- Ejemplo 3 (cont.)
  - Diagrama de estados
    - A la función lógica que determina si debemos abrir o no la barrera del parking la denominaremos f0



$$f0 = CW \cdot \overline{OL1} \cdot \overline{OL0} + CW \cdot UA \cdot \overline{OL1} \cdot OL0 + CW \cdot UA \cdot ST \cdot OL1 \cdot \overline{OL0}$$

Ejemplos de diseño – 3.- Genérico (v) FCO

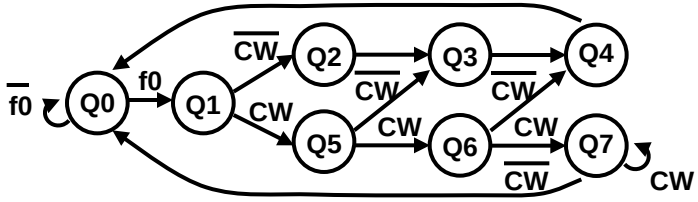
- Ejemplo 3 (cont.)
  - Función de excitación
    - Con 3 variables de estado … ¡la función de excitación tiene $2^8 = 256$ combinaciones de entrada!
      - Aunque no es imposible, el problema resulta intratable si se intentar resolver a mano
        - » *Simplificar por Karnaugh, por ejemplo, es imposible*
    - En este caso resulta más fácil utilizar una codificación *"one-hot"*
      - Tantas variables de estado como estados distintos
      - Sólo una variable de estado está activa (*"hot"*) en cada estado
      - Asociamos cada estado con su variable estado
      - Derivar ecuaciones (sin simplificar) es habitualmente más sencillo

Ejemplos de diseño – 3.- Genérico (vi) FCO

- Ejemplo 3 (cont.)
  - Codificación "*one hot*"
    - 8 variables de estado: Q7, Q6, Q5, Q4, Q3, Q2, Q1 y Q0
    - Estado Q0 "*hot*" (codificado 00000001): "reposo"
      - Cuando Q0 = 1 el autómata está en el estado "reposo"
    - Q1 (codificado 00000010) : "abrir (1er ciclo)"
    - Q2, Q3 y Q4: "abrir (…)" (2º, 3er y 4º ciclo, respectivamente)
      - Codificados 00000100, 00001000 y 00010000 respectivamente
    - Q5, Q6 y Q7: "abrir2 (…)" (2º, 3er y 4º ciclo, respectivamente)
      - Codificados 00100000, 01000000 y 10000000 respectivamente
  - Función de salida
    - La salida sólo es 0 en el estado de reposo, por lo que $O = \overline{Q0}$

Ejemplos de diseño – 3.- Genérico (vii) FCO

- Ejemplo 3 (cont.)
  - Función de excitación



    - Otra forma de entender la función de excitación es la siguiente: para alcanzar un estado (Qi(t+1) = 1 usando "*one-hot*") debe cumplirse la condición indicada para, al menos, una de las transiciones que tenga a ese estado como destino
    - Transitamos al estado de reposo utilizando 3 transiciones
      - Desde Q0 (si se cumple que f0 = 0), siempre que estemos en el estado Q4 y desde Q7 (si se cumple que CW = 0). Por tanto,

$$D0 = Q0 \cdot \overline{f0} + Q4 + Q7 \cdot \overline{CW}$$

    - Alcanzamos el estado Q1 desde Q0 si f0 = 1, por lo que

$$D1 = Q0 \cdot f0$$

    - Análogamente obtenemos …

$$D2 = Q1 \cdot \overline{CW} \qquad D5 = Q1 \cdot CW$$

$$D3 = Q2 + Q5 \cdot \overline{CW} \qquad D6 = Q5 \cdot CW$$

$$D4 = Q3 + Q6 \cdot \overline{CW} \qquad D7 = Q6 \cdot CW + Q7 \cdot CW$$

UNIVERSIDAD POLITECNICA DE VALENCIA

etsinf

DISCA

Fundamentos de los computadores

Tema 5. SISTEMAS SECUENCIALES (Segunda Parte)