UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Sistemas secuenciales síncronos: codificación de estados de un control de volumen

Apellidos, nombre	Martí Campoy, Antonio (amarti@disca.upv.es)
Departamento	Informàtica de Sistemes i Computadors
Centro	Escola Tècnica Superior d'Enginyeria Informàtica

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

1 Resumen de las ideas clave

En este artículo trabajarás una parte del proceso de diseño e implementación de un Sistema Secuencial Síncrono (SSS). El diseño de SSS incluye varios pasos, y en cada paso existen múltiples alternativas. Es imposible tratarlo todo en un documento breve, y además sólo nos llevaría a confusión.

Por eso, en este documento partiremos de un diagrama de estados ya diseñado y estudiaremos las diferentes alternativas para codificar los estados representados en ese diagrama de estados.

Para poder adquirir los conocimientos y habilidades presentadas en este artículo, debes tener los conocimientos previos presentados en la tabla 1.

Tabla 1. Conocimientos previos

Conocimientos previos
1. Funciones lógicas Booleanas.
2. Circuitos combinacionales.
3. Biestables y sus tablas de funcionamiento.
4. Conocer teóricamente los pasos necesarios para diseñar un SSS.

2 Objetivos

Una vez acabes de leer este artículo docente y reproduzcas los ejemplos presentados, serás capaz de **codificar** los estados de un sistema secuencial síncrono **eligiendo** entre varias alternativas y **construyendo** la tabla de codificación

3 Introducción

Cuando se construye el diagrama de estados de un SSS se suelen utilizar nombres simbólicos para identificar los estados. Esto se hace así principalmente porque facilita el diseño, ya que los nombres nos ayudan a identificar el significado del estado. Por ejemplo, si estamos diseñando el control de un ascensor, nombrar a los estados con “subiendo”, “bajando” o “cerrando puertas” facilita la compresión del diagrama que se está construyendo.

Pero el objetivo final de la construcción de un SSS es la obtención de un circuito digital que utiliza biestables para almacenar el estado actual del sistema. Y como bien sabes, los biestables no almacenan letras y caracteres alfanuméricos, sino bits. Por tanto, uno de los pasos en el diseño de un SSS es traducir los nombres de los estados a binario. Esto recibe el nombre de codificación de estados. El resultado de la codificación es una tabla donde se muestran los nombres simbólicos y junto a ellos el código binario asociado. Los bits del código binario o “estado” se

identifican por la letra Q, en referencia al nombre de las salidas de los biestables, que son los circuitos que almacenan el estado.

Existen múltiples alternativas para llevar a cabo la codificación de estados. Estas alternativas pueden agruparse en codificaciones mínimas y no mínimas. En la codificación mínima se puede encontrar a su vez diferentes variantes, usándose binario natural, código gray o prioridad a la salida. En las codificaciones no mínimas encontramos la codificación one-hot. No son estas las únicas posibilidades de codificación, pero sí son las más sencillas y utilizadas.

Para poder hacer ejemplos de aplicación de las diferentes maneras de codificar los estados, necesitamos partir de un SSS, especificado por su interfaz y diagrama de estados.

La Figura 1 muestra la interfaz del circuito. De este interfaz podemos extraer información muy interesante e importante: el sistema tiene tres entradas y dos salidas.

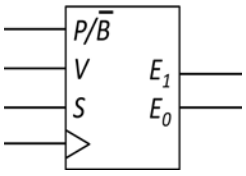

Figura 1. Interfaz del sistema

La Figura 2 muestra el diagrama de estados, una representación gráfica del comportamiento deseado para el circuito. En este diagrama podemos ver que hay cuatro estados, y que las transiciones entre estados están indicadas mediante expresiones algebraicas. Pero lo mejor de este diagrama de estados, y de cualquier diagrama de estados, es que ya no necesitas saber nada del comportamiento del circuito, ni del significado de las entradas y salidas. Al disponer del diagrama, porque tú mismo lo has construido o te lo han proporcionado como es el caso, no necesitas ninguna información en lenguaje "humano" sobre el circuito a construir.

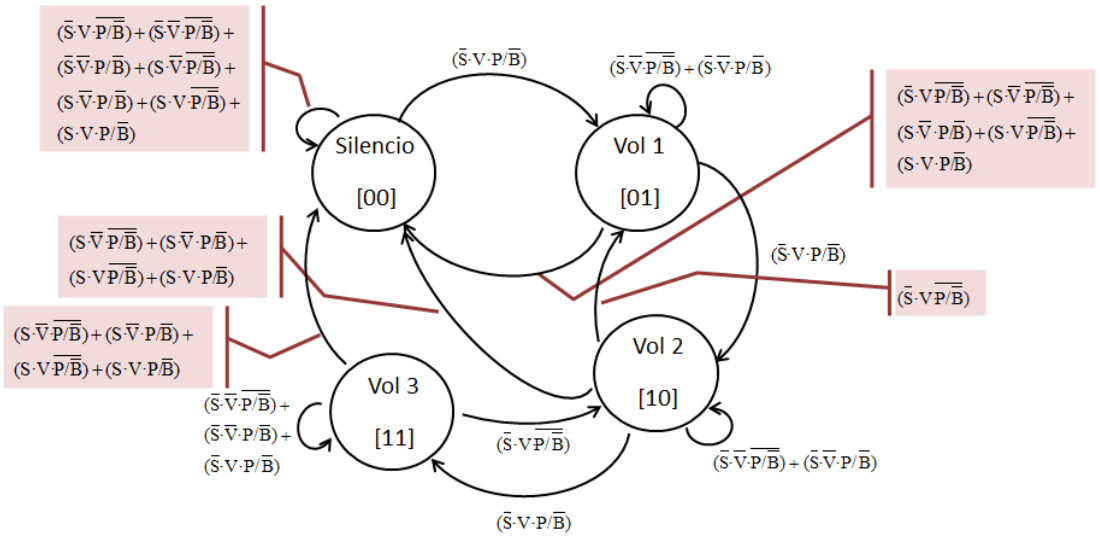

Figura 2. Diagrama de estados completo

4 Codificación de los estados

Elegir una codificación u otra tendrá impacto tanto en el proceso de diseño del SSS como en el circuito resultante. Según la codificación escogida, el circuito combinacional necesario para calcular el estado siguiente será más o menos complejo. Lo mismo sucederá con el circuito combinacional necesario para calcular las salidas. Y también afectará el número de biestables necesarios para almacenar el estado.

Las codificaciones mínimas tienen como objetivo reducir el número de biestables, y después intentan simplificar los circuitos combinacionales de estado siguiente y de salida. Por otro lado, las codificaciones no mínimas utilizan más biestables, pero simplifican el proceso de diseño y también los circuitos combinacionales resultantes. Empezaremos a estudiar la codificación no mínima *one-hot*, y luego veremos algunas codificaciones mínimas.

4.1 Codificación no mínima: *one-hot*

En la codificación *one-hot* el objetivo es simplificar el proceso de diseño de los circuitos digitales que serán necesarios para construir el sistema, y también obtener circuitos combinacionales de estado siguiente y de salida sencillos Pero el precio que se paga es el uso de un mayor número de biestables, ya que se utilizan tantos biestables como estados tiene el sistema.

A cada estado se le asigna un código donde sólo uno de los bits toma valor 1 dejándose el resto de bits a cero. No existe ningún criterio para decidir que código se asigna a cada estado. La única restricción es que no existan códigos duplicados, y que los códigos utilizados sólo tengan un bit con valor 1.

El diagrama de estados que nos han dado como ejemplo tiene cuatro estados, por lo que necesitaremos cuatro biestables que nos proporcionarán cuatro códigos diferentes: 0001, 0010, 0100 y 1000. La Tabla 2 muestra una posible codificación de los estados, pero recuerda que no es la única, cualquiera es válida siempre que no se repita ningún código y todos tengan sólo un bit a 1.

Tabla 2. Codificación de los estados con codificación *one-hot*.

Nombre simbólico	Código $Q_3Q_2Q_1Q_0$
Silencio	0001
Vol 1	0010
Vol 2	0100
Vol 3	1000

El principal inconveniente de esta codificación es si el sistema tiene muchos estados, ya que en ese caso se utilizan muchos biestables.

La principal ventaja es que se simplifica el proceso de construcción del SSS y es fácil corregir y modificar el diseño. Además, los circuitos combinacionales de estado siguiente y salida suelen tener una complejidad baja y un tiempo de respuesta rápido.

4.2 Codificación mínima

Tal como te he dicho antes, en una codificación mínima se persigue que el número de biestables utilizados en el circuito final sea mínimo. Este número de biestables se corresponde con el valor de n que cumple la expresión $2^{n+1} > K \geq 2^n$ donde K es el número de estados. También puede calcularse como $n = \lceil \log_2 K \rceil$. Por tanto utilizaremos n bits para codificar los estados, asignando un número binario a cada uno de ellos.

Para el diagrama de ejemplo que estamos usando necesitamos $n = \lceil \log_2 4 \rceil = 2$, es decir, 2 biestables que representaremos con Q_1Q_0.

Una vez sabes cuántos biestables necesitas sólo tienes que asignar un código único a cada estado. Esta asignación puede hacerse de varias maneras, que producirán circuitos más o menos complejos. A continuación puedes encontrar tres opciones: binario, prioridad a la salida y código Gray.

4.2.1 Binario

En la codificación mínima en binario los estados se numeran en orden ascendente empezando por el cero. Si los estados no presentan un orden claro el diseñador puede repartir los códigos en el orden que quiera. La Tabla 3 muestra la codificación en binario donde se ha comenzado a numerar los estados por el estado "Vol 1" y se ha terminado con el estado "Silencio". Cualquier otra asignación es válida siempre y cuando no se repita ningún código.

El principal inconveniente de esta codificación es que la complejidad de los circuitos de estado siguiente y de salida que se obtienen es imprevisible. Es posible que el resultado sea sencillo o extremadamente complejo.

Tabla 3. Codificación de los estados con codificación mínima en binario.

Nombre simbólico	Código Q_1Q_0
Vol 1	00
Vol 2	01
Vol 3	10
Silencio	11

4.2.2 Prioridad a la salida

En la codificación mínima con prioridad a la salida se intenta simplificar el circuito combinacional que genera la salida. Recuerda que en un autómata de Moore la salida depende únicamente del estado en que se encuentra el sistema, por lo que si conseguimos encontrar una relación sencilla entre el código del estado y los valores de las salidas, obtendremos un circuito de salida muy sencillo.

Si te fijas en el diagrama de estados que estamos usando como ejemplo, cada estado tiene un valor diferente en las salidas: 00 para el estado "Silencio", 01 para el estado "Vol 1", 10 para el estado "Vol 2" y 11 para el estado "Vol 3". Si usamos el valor de las salidas cómo código de cada estado,

el circuito combinacional de salida será el circuito identidad, es decir, las salidas son iguales al estado, sin necesidad de utilizar ninguna puerta lógica, que es el circuito más sencillo que se puede obtener. La Tabla 4 muestra la codificación de los estados.

No siempre es posible encontrar una relación tan directa entre las salidas y los códigos de los estados, pues es bastante habitual que el número de biestables sea diferente al número de salidas, y también es habitual que diferentes estados tengan exactamente el mismo valor en las salidas. En cualquier caso, el circuito combinacional de las salidas será más sencillo que si usas simplemente codificación mínima en binario, pero se desconoce a priori la complejidad del circuito combinacional necesario para generar el estado siguiente.

Tabla 4. Codificación de los estados con codificación mínima y prioridad a la salida, haciendo coincidir el código del estado con su salida.

Nombre simbólico	Código Q_1Q_0	Salidas E_1E_0
Silencio	00	00
Vol 1	01	01
Vol 2	10	10
Vol 3	11	11

4.2.3 Código Gray

Esta codificación debe su nombre al código Gray o código reflejado, que es un sistema de numeración en el que dos valores consecutivos difieren sólo en un bit. Este código fue inventado por Frank Gray en 1947.

El objetivo de esta codificación es obtener un circuito combinacional de estado siguiente lo más sencillo posible. Para ello se aprovechan las propiedades del álgebra de Boole, que permiten simplificar expresiones algebraicas con términos que sólo difieren en un variable. Estos términos reciben el nombre de términos adyacentes. Para conseguir que en las expresiones del circuito de estado siguiente aparezcan términos adyacentes, hay que conseguir que los códigos de dos estados que sean origen y destino el uno del otro difieran sólo en un bit.

Por tanto, se utiliza el código Gray para numerar los estados siguiendo las transiciones entre ellos. La Tabla 5 muestra una posible codificación mínima Gray para el diagrama de estados ejemplo.

Al igual que sucedía en la codificación con prioridad a la salida, no siempre es fácil o posible encontrar una codificación óptima. Si, por ejemplo, existe un estado que tiene varios destinos posibles, puede ser imposible conseguir que todos los estados consecutivos difieran sólo en un bit.

En cualquier caso, el circuito de estado siguiente resultante será más sencillo que si se utiliza codificación binaria o con prioridad a la salida.

Tabla 5. Codificación de los estados con codificación mínima en código Gray.

Nombre simbólico	Código Q_1Q_0
Vol 1	00
Vol 2	01
Vol 3	11
Silencio	10

4.3 Cuando hay más códigos que estados

Es posible que, por su número, los biestables que vamos a utilizar nos proporcionen mayor número de códigos de estado que estados tiene realmente el sistema. Esto pasa siempre en codificación *one-hot*, pero puede pasar también en codificación mínima. Los códigos que no se asignan a ningún estado reciben el nombre de estados ilegales. Estos códigos no aparecen en la tabla de codificación.

5 Ejercicio

Para comprobar que lo tienes todo claro te propongo un ejercicio. La Figura 3 muestra la interfaz y el diagrama de estados de un SSS para controlar un semáforo con una entrada de prioridad al peatón. Observando el diagrama de estados, responde a las siguientes preguntas sobre la codificación de los estados:

- ¿Cuántos biestables son necesarios si usamos codificación *one-hot*? ¿Y si usamos codificación mínima?
- ¿Es buena idea utilizar codificación binaria con prioridad a la salida?
- Si usamos codificación mínima en código Gray, ¿obtendremos un circuito de estado siguiente mínimo?
- Construye la tabla de codificación de estados usando codificación mínima en código Gray.

Al final del documento tienes las respuestas a las preguntas, pero es muy importante que antes de mirarlas intentes resolverlas tú.

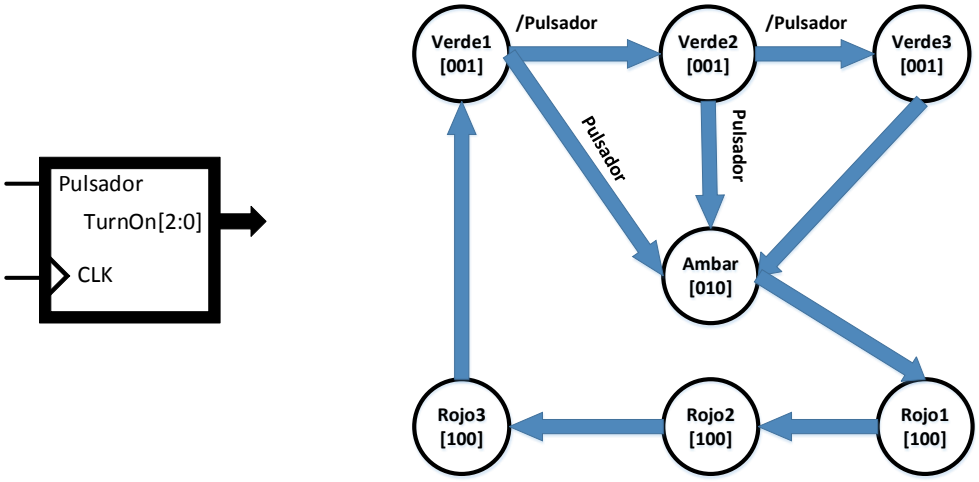

Figura 3. Interfaz y diagrama de estados del control de un semáforo con prioridad al peatón.

6 Conclusiones

Una vez tenemos la interfaz y el diagrama de estados de un SSS (sistema secuencial síncrono), bien porque lo hemos diseñado nosotros mismos o porque nos lo proporcionan, el siguiente paso hacia la implementación del sistema es la codificación de los estados.

La codificación de los estados da como resultado una tabla donde se emparejan los nombres simbólicos de los estados con un código binario. El único requisito que debe cumplir esta asignación es que sea una correspondencia biunívoca, es decir, que a cada estado simbólico se le asigna un único código y ningún código se utiliza para dos o más estados simbólicos.

Para asignar los códigos a los estados hay diferentes alternativas, incluso asignaciones *ad hoc* para sistemas específicos o en función del hardware que se vaya a utilizar para implementar los circuitos digitales.

Pero recuerda que no importa que opción elijas, el sistema funcionará exactamente igual, pero el proceso de diseño de los circuitos y la complejidad de los mismos puede variar. ¿Cuál es mejor? Siento decirte que depende de cada sistema, y que tendrás que probar varias alternativas para encontrar la mejor.

7 Bibliografía

7.1 Libros:

John F. Wakerly *"Digital design : principles and practices"*, Prentice Hall. 2006

Antonio Lloris Ruiz; Alberto Prieto Espinosa; Luis Parrilla Roure *"Sistemas digitales"*, Aravaca, Madrid : McGraw-Hill/Interamericana de España. 2003

7.2 Referencias de fuentes electrónicas:

'One-hot', *Wikipedia, The Free Encyclopedia,* > [accessed 8 May 2015] <http://en.wikipedia.org/w/index.php?title=One-hot&oldid=659883289

'State encoding for low power', *Wikipedia, The Free Encyclopedia,* <http://en.wikipedia.org/w/index.php?title=State_encoding_for_low_power&oldid=660388530> [accessed 8 May 2015]

8 Soluciones al ejercicio

Antes de darte las respuestas debo decirte que este tipo de problemas no tienen una solución única. Las respuestas que se te dan son una posible solución,

- ¿Cuántos biestables son necesarios si usamos codificación *one-hot*? ¿Y si usamos codificación mínima? Con codificación *one-hot* necesitamos 7 biestables, ya que hay siete estados diferentes. Con codificación mínima necesitamos 3 biestables ($2^3=8>7$), sobrándonos un código de estado.
- ¿Es buena idea utilizar codificación binaria con prioridad a la salida? No parece una buena idea ya que hay varios estados con la misma combinación de

salidas, lo que provoca que no se puedan hacer relaciones biunívocas entre las salidas y el código de estado, impidiendo simplificar el circuito de salida.

- Si usamos codificación mínima en código Gray, ¿obtendremos un circuito de estado siguiente mínimo? Es posible que obtengamos un circuito de estado siguiente bastante sencillo, pero no será mínimo. Esto es porque hay estados que tienen varios destinos, un estado con varios orígenes y además sobra un código de estado, por lo que al menos habrá un salto de dos bits entre dos códigos.
- Construye la tabla de codificación de estados usando codificación mínima en código Gray. El objetivo es maximizar el número de transiciones en las que sólo cambia un bit de un estado al siguiente. La Tabla 6 muestra la tabla de codificación. Si añadimos los códigos binarios diagrama de estados (Figura 4), vemos que para pasar de Rojo3 a Verde1 tenemos que cambian dos bits ya que tenemos sólo 7 estados y no 8, y que de las tres transiciones de Verde a Ámbar dos tienes un cambio de un solo bit y sólo una de ellas tiene que cambiar dos bits.

Tabla 6. Tabla de codificación para el SSS descrito en la Figura 3.

Nombre simbólico	Código $Q_2Q_1Q_0$
Verde1	000
Verde2	001
Verde3	011
Ámbar	010
Rojo1	110
Rojo2	111
Rojo3	101

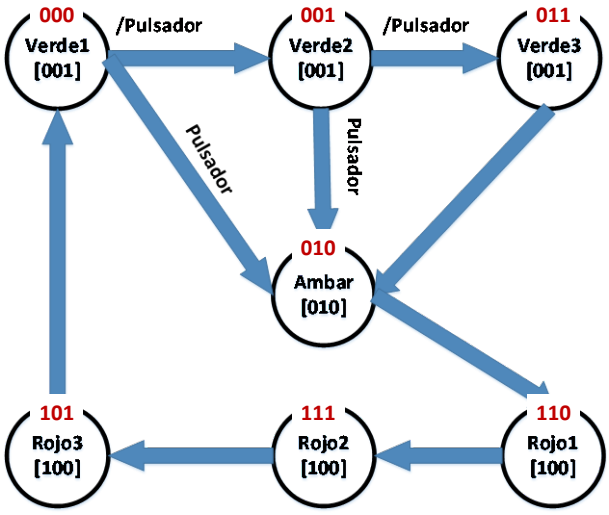

Figura 4. Diagrama de estados del SSS semáforo con los estados codificados según la Tabla 6