Tecnología de Computadores
Práctica 9
Sistemas secuenciales: Contadores

Universidad Miguel Hernández [1]

6 de mayo de 2016

Resumen

Este documento describe la práctica número 9 de la asignatura Tecnología de Computadores de 2º del Grado en Ingeniería Informática en Tecnologías de la Información de la Universidad Miguel Hernández. Este material docente ha sido desarrollado enteramente, a menos que se indique lo contrario, por el profesor P. Pablo Garrido Abenza.

Índice general

Capítulo 1

Introducción

1.1. Objetivos

Tras completar esta práctica el alumno deberá ser capaz de:

- Buscar y manejar la hoja de características (*datasheet*) de algún circuito integrado (IC).
- Diseñar contadores a partir de biestables y otros componentes combinacionales, tanto asíncronos (ascendentes y descendentes) como síncronos, comprendiendo la diferencia entre ambos.
- Combinar circuitos combinacionales y secuenciales para diseñar algún circuito con aplicación práctica.

1.2. Software necesario

El software necesario para la realización de las siguientes prácticas es *Simulín* v5.61 o posterior, el cual se encuentra ya instalado en los ordenadores del aula de informática. También puede se puede instalar en cualquier otro ordenador personal, ya que este software es de libre distribución. Está disponible para Windows, Linux y Mac OS X, y se puede descargar desde el material de la asignatura, o también, desde el siguiente enlace.

Capítulo 2

Descripción

Se comenzará diseñando un contador asíncrono, y después se diseñará un contador síncrono. Finalizaremos la práctica realizando un diseño para dar una aplicación práctica al diseño de contadores.

2.1. Consejos para analizar los circuitos

Para **analizar el comportamiento** de estos circuitos, y de cualquier otro circuito secuencial, se aconseja mostrar el cronograma previamente (Simulación > Cronograma). Después, puede iniciarse la simulación (Simulación > Iniciar), lo cual pondrá en marcha la señal de reloj con el periodo establecido en sus propiedades. En cualquier momento, se podrá detener la simulación (Simulación > Detener), para ir paso a paso mediante la modificación manual de las entradas o reloj del circuito (Click-dcho. > Cambiar valor). Posteriormente se puede reanudar de nuevo la simulación (Simulación > Iniciar), continuando por el estado actual. Finalmente, siempre se puede volver a empezar desde cero la simulación (Simulación > Reiniciar).

Se aconseja **nombrar las salidas** conectadas a los biestables del siguiente modo: `Q0 (LSB)`, `Q1`, `Q2`, ... `Qn (MSB)`. De esta forma se facilita la interpretación del cronograma cuando se haga la simulación. Además, es conveniente **ordenar las entradas y salidas** con la opción 'Simulación > Cambiar orden I/O' del simulador, antes o durante la simulación si fuera necesario.

2.2. Contador asíncrono

Se pide diseñar un **contador asíncrono ascendente % 10** utilizando biestables J-K activados por flanco de subida, es decir, que siga la secuencia ascendente y correlativa: 0,1,2,3,4,5,6,7,8,9 y vuelta a empezar (archivo: `cont_asinc_10.sim`).

El contador dispondrá de las siguientes E/S (ver figura 2.1):

- **Entradas** la señal de reloj `CLK`, y la señal `RESET` para puesta a 0 del contador cuando `RESET=1`.
- **Salidas**: serán las salidas `Q` de los biestables (Q_0, Q_1, Q_2, Q_3) y una señal `inicio` que se indica cada vez que se llega al final de la cuenta y vuelve a empezar.

Recuerda que para el correcto funcionamiento del contador no debemos detectar el estado 9 sino el 10, el cual se considera un estado transitorio que será imperceptible, es decir, al detectar el estado 10_{10} (1010_2) se actuará sobre las entradas asíncronas y prioritarias de los biestables (`PRESET'` y/o `CLEAR'`) que sean necesarias para pasar al estado 0_{10} (0000_2) a partir de la salida de una puerta `NAND`. En este punto se aconseja comprobar el correcto funcionamiento del circuito.

Para reinicializar el contador en cualquier momento podemos incluir una señal `RESET`, es decir, para que pase al estado inicial 0_{10} (000_2) cuando `RESET=1`. Deberemos tener en cuenta esta señal a la hora de averiguar la lógica necesaria para las entradas `PRESET'` y/o `CLEAR'` de cada uno de los biestables en función de la señal `RESET` y la salida de la puerta `NAND` del contador necesaria para reinicializar la cuenta. Como ayuda se proporciona la siguiente tabla.

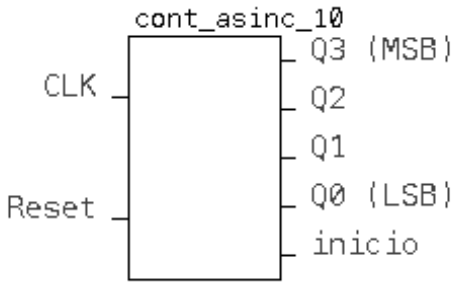


Figura 2.1: Contador asíncrono - Símbolo de bloque

RESET	NAND	CLR'3	CLR'2	CLR'1	CLR'0	PRE'3	PRE'2	PRE'1	PRE'0
0	0								
0	1								
1	0								
1	1								

Algunas funciones de salida de la tabla anterior puede que no sean necesarias, y otras pueden coincidir en más de un biestable. Si es posible, insertar una **nota de texto** en el circuito con la tabla rellena y cualquier otro comentario que se considere oportuno.

RECUERDA: las señales CLR' y PRESET' actúan a nivel bajo, es decir, cuando CLR'=0 es cuando asigna 0 al biestable, y cuando PRESET'=0 asigna 1 al biestable.

Para pensar:

- Si un contador asíncrono se debe implementar con biestables activados por flanco de bajada, ¿cómo se realizan las conexiones de la entrada de reloj para que sea ascendente? ¿Y para que sea descendente?.
- Si un contador asíncrono está implementado con biestables activados por flanco de subida, ¿sucede algo si se sustituyen por otros activados por flanco de bajada respetando las conexiones existentes?.
- Si en vez de utilizar biestables J-K se pidiera utilizar biestables T, ¿cuáles serían las entradas de datos de los nuevos biestables T?.

2.3. Contador síncrono módulo 8

En este apartado realizaremos un **contador síncrono módulo 8** ascendente (0..7), utilizando **biestables T** activados por **flanco de subida** (archivo: `cont_sinc_8.sim`).

El contador dispondrá de las siguientes E/S (ver figura 2.2):

- **Entradas**: la señal de reloj `CLK`, y la señal `RESET` para puesta a 0 del contador cuando `RESET=1`.
- **Salidas**: serán las salidas `Q` de los biestables (Q_0, Q_1, Q_2) y una señal `inicio` que se indica cada vez que se inicia una nueva cuenta.

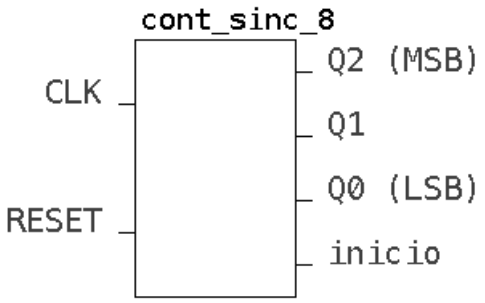


Figura 2.2: Contador síncrono - Símbolo de bloque

Este circuito lo necesitaremos insertar como bloque funcional en el último apartado de esta práctica (ver sección 2.4).

Para pensar:

- Si un contador síncrono está implementado con biestables activados por flanco de subida, ¿sucede algo si se sustituyen por otros activados por flanco de bajada?.
- Si en vez de utilizar biestables J-K se pidiera utilizar biestables T, ¿cuáles serían las entradas de datos de los nuevos biestables T?.

2.4. Aplicación práctica

Para finalizar esta parte de la asignatura, vamos a realizar una aplicación práctica en la que necesitaremos incluir tanto elementos combinacionales como secuenciales. Se trata de diseñar un circuito electrónico que simule el funcionamiento del display de un reproductor de CD (ver Fig. 2.3).

Tendremos tres pulsadores `Pause`, `Play`, y `Stop`, dos visualizadores de 7 segmentos, uno que, dependiendo del estado de los pulsadores, indica si se está reproduciendo el CD (simulando un CD que gira) o no, y el otro que indica la pista actual que se está reproduciendo. Ambos displays visualizarán únicamente símbolos proporcionados por el decodificador de 7-segmentos realizado en una práctica anterior, el del archivo `deco7_cd.sim`. Estos decodificadores, a su vez, reciben el código del símbolo a visualizar desde un contador; aquí se utilizarán los dos contadores implementados en los apartados anteriores, uno para cada display.

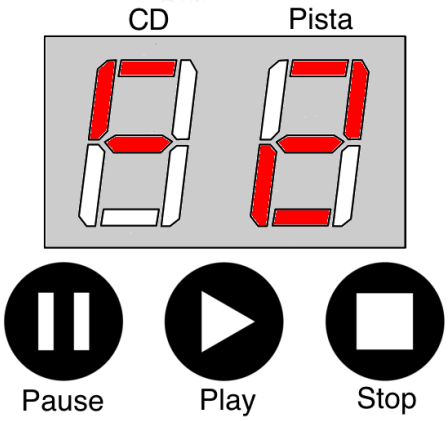


Figura 2.3: Reproductor de CD

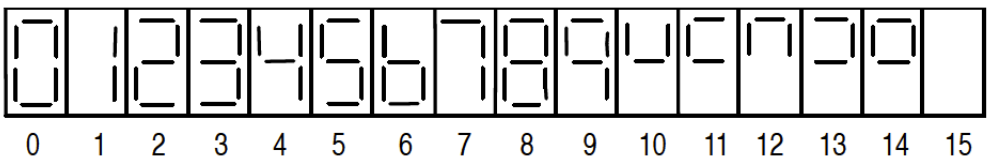


Figura 2.4: Segmentos para cada dígito del decodificador

2.4.1. Circuito `pauseplay.sim`

Para dividir la tarea en partes más manejables, comenzamos con la realización de un circuito combinacional sencillo que luego insertaremos en el circuito final. Se trata de un circuito (archivo: `pauseplay.sim`) que recibe como entradas la señal `Pause`, `Play`, y un valor binario de 2-bit (ver Fig. 2.5); según el valor de las entradas proporciona como salida el código binario que se enviará a un decodificador para mostrarlo en el display de 7-segmentos (A,B,C,D), que podrán ser los valores 10 a 13 (CD girando) o 14 a 15 (CD parpadeando), es decir, este circuito ignorará los valores 0 a 9 (ver Fig. 2.4).

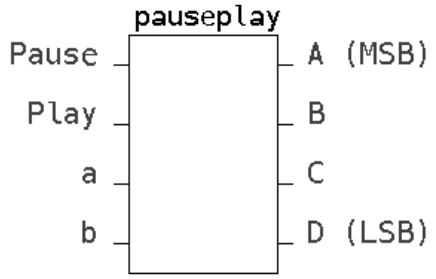


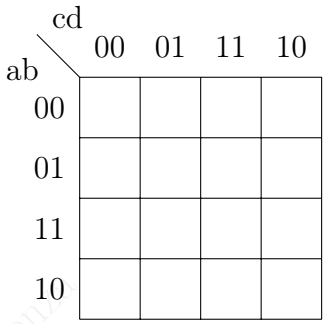
Figura 2.5: Símbolo de bloque del circuito `pauseplay`

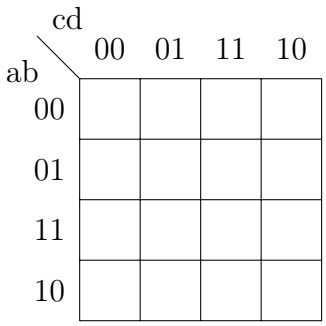
El **comportamiento** de este circuito será el siguiente:

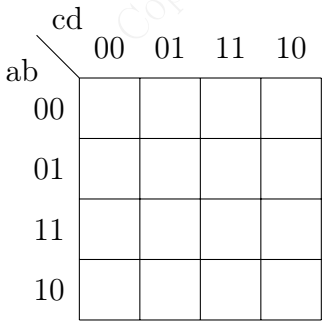
- Si **Pause** y **Play** están desactivados, el display de 7 segmentos estará en blanco (código 15).
- Si sólo **Play** está activado, el display mostrará uno de los 4 posibles valores de código 10 a 13, según el valor de 2-bits recibido (a,b). De este modo, conforme el contador vaya incrementando sucesivamente su valor, la imagen visualizada irá alternando entre esos 4 símbolos, dando la sensación de un CD que está girando. En otras palabras, habrá que sumar 10 al valor de 2-bits recibido (ab+1010_2)
- Si **Pause** está activado, independientemente de si Play está activado o no, el display mostrará imágenes correspondientes a los códigos 14 y 15 de forma alternada, según el valor de 2-bits recibido, dando la sensación de un CD parpadeando. Se puede hacer simplemente sumando 14 al bit LSB del valor de 2-bits recibido (b+1110_2).

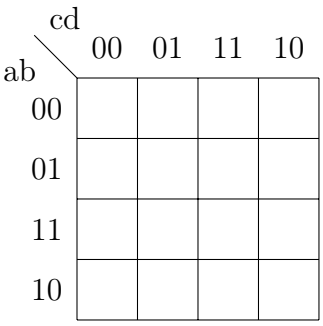
Se trata de completar la siguiente **Tabla de Verdad** y simplificar posteriormente las 4 funciones de salida.

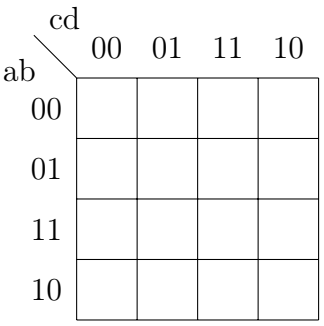
Pause	Play	a	b	A	B	C	D
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	0	1				
1	1	0	0				
1	0	0	1				
1	1	1	0				
1	0	0	1				











2.4.2. Circuito final

Una vez construido el circuito anterior, y teniendo también disponibles los circuitos de prácticas anteriores, procederemos a diseñar el circuito completo para el reproductor de CD (archivo: `cdplayer.sim`).

Comenzamos haciendo una lista de los **elementos necesarios** (ingredientes):

- Las **entradas**: Pause, Play, Reset (inicializa los dos contadores), ...
- 1 señal de **reloj** `CLK` para el contador
- 2 **contadores**: uno asíncrono y otro síncrono, de los apartados 2.2 y 2.3, respectivamente.
- 1 **circuito** `pauseplay` del apartado anterior 2.4.1.
- 2 decodificadores de 7-segmentos (realizados anteriormente en la P5).
- 2 visualizadores o displays de 7-segmentos.
- Alguna puerta lógica básica, masa, tensión, biestables, ... también podrían ser necesarios (o no).

En relación a los **displays** de 7-segmentos, ambos necesitarán su correspondiente decodificador:

- El primer display (llamado 'CD') indicará el estado del CD: sensación de movimiento si `Play` está activo, CD parpadeante si `Pause` está activo, y totalmente apagado si no se ha activado ni `Pause` ni `Play`.
- Cuando el `Play` está activado, el primer display muestra un CD girando, mientras que el segundo display (llamado 'Pista'), indicará la pista que se está reproduciendo en cada momento, e irá incrementando su valor conforme el CD vaya dando vueltas. Puesto que el display sólo muestra un dígito, veremos las pistas 0 a 9 y luego volverá a empezar por el 0. En concreto, supondremos que cada canción 'dura' 2 vueltas, es decir, que cada 2 vueltas del CD en el display 'CD' se habrá reproducido una canción completa y se incrementará el número de pista.
- Cuando el `Pause` está activado, el primer display 'CD' muestra el CD parpadeando, y el display 'Pista' se quedará tal cual (no debe avanzar). Esto se realiza añadiendo alguna lógica en la señal de reloj del contador que cuenta el número de pista.

- Cuando se active `Stop`, el primer display 'CD' aparecerá en blanco (apagado), y el segundo display 'Pista' pondrá 0. Esto se puede conseguir simplemente conectado la entrada Stop a las entradas Reset de los dos contadores.

En cuanto a los **contadores**:

- Tenemos dos contadores, uno que cuenta 0..8 y otro 0..9. Debemos elegir cuál de ellos se conecta a cada display. El hecho de que sean síncronos o asíncronos no afecta en nada.
- Ambos contadores proporcionan una señal `inicio`, aunque sólo se utilizará en el primero, ya que se conectará a la señal de reloj del segundo para indicar cada vez que inicia la cuenta, momento en el que el segundo contador avanzará, es decir, la señal `inicio` del primer contador será la señal de reloj del segundo. Además, como se ha comentado antes, se deberá tener en cuenta que cuando `Pause` esté activo dejará de contar.

Con la descripción anterior, se deberá conectar todos los elementos para tener la funcionalidad completa.

Para pensar:

- ¿Cómo puede conseguirse que cada canción 'dure' 4 vueltas del CD en lugar de 2?. ¿Y si se quiere que sean 8 vueltas?.
- ¿Qué circuitos integrados (ICs) comerciales necesitarías para implementar físicamente el diseño realizado?

Capítulo 3

Entrega de la práctica

La forma de entrega de esta práctica se realizará de la siguiente forma:

1. Comprimir en un único archivo todos los circuitos *.sim implementados. Para ello se puede utilizar la opción 'Copia de seguridad' de Simulín, o programas como WinZip o 7z, admitiéndose los formatos: .zip, .7z, o .rar.

2. Subir el archivo comprimido a la tarea correspondiente antes del plazo fijado.

Los archivos a entregar para esta práctica se resumen en la tabla 3.

#	Descripción	Archivo
1	Cont. asíncrono % 10	`cont_asinc_10.sim`
2	Cont. síncrono % 8	`cont_sinc_8.sim`
3	Circuito auxiliar	`pauseplay.sim`
4	CD Player	`cdplayer.sim`

Cuadro 3.1: Archivos para los circuitos a implementar

NOTA: todos los circuitos adicionales que necesites de prácticas anteriores deberán estar en el mismo directorio que los de la práctica actual, y se deben incluir también en la entrega (decodificadores, sumadores, etc.).

RECUERDA: las prácticas serán corregidas por un programa informático, por lo que se insiste en que los archivos de los circuitos tengan el nombre que se especifica en las tablas, respetando incluso mayúsculas/minúsculas; el nombre del archivo comprimido que los contiene no importa.

Glosario de Acrónimos

Siglas	Significado
BCD	Binary-Coded Decimal
IC	Integrated Circuit
LED	Light-emitting diode
LSB	Least Significant Bit
MSB	Most Significant Bit