Sistemas Digitales

Ejercicios de Examen, Circuitos Secuenciales

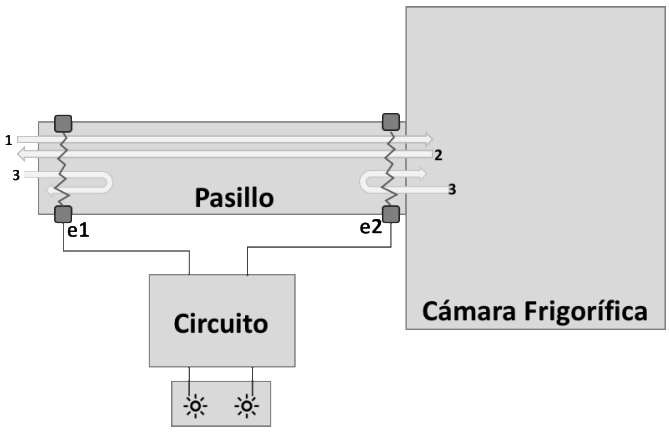
Para solucionar cada ejercicio, abre un nuevo proyecto en LogiSim. Dentro de cada proyecto, crea los circuitos que indique el enunciado del ejercicio.

Ejercicio 1. Parcial 2017-18 (Proyecto Ejercicio1.circ) Se supone una cámara frigorífica como la de la figura inferior, dentro de una pescadería, en la que trabajan dos personas. Por medidas de seguridad, se desea disponer de una monitorización permanente del número de trabajadores que hay dentro de dicha cámara. Para su acceso hay un pequeño pasillo, por el que solamente cabe una persona y en el que se han colocado dos sensores, *e1* y *e2*, de barrera, cuyo haz de luz es interrumpido cuando alguien lo atraviesa. Dichos sensores están colocados al principio y al final de dicho pasillo.
Se desea diseñar el circuito de control capaz de encender un número de leds igual al número de personas (0, 1 ó 2) que hay en el interior (El pasillo no se considera el interior). Los únicos movimientos permitidos son los mostrados en la figura:

1. Entrar y salir normalmente, es decir, atravesar *e1*, recorrer el pasillo y atravesar *e2*. Y viceversa.
2. Ir a entrar y arrepentirse. Es decir, atravesar *e1*, volverse a la mitad de pasillo atravesar *e1* de nuevo.
3. Ir a salir y arrepentirse. Es decir, atravesar *e2*, volverse a mitad del pasillo y volver a atravesar *e2*.

Los demás posibles supuestos no se permiten (no caben dos personas por el pasillo al tiempo, no se pueden arrepentir sin haber atravesado completamente cualquiera de los sensores, no se pueden atravesar dos sensores en el mismo ciclo, etc.)
Realiza las siguientes tareas:

a. Obtén el grafo de estados del circuito de control descrito.
b. Determina las tablas de verdad (Transiciones y Salida) correspondientes a dicho grafo.
c. Utiliza *LogiSim* para crear la *ROM* correspondiente al CLC de estado siguiente e implementa el CLC de salida como suma de minterms. Conecta los circuitos creados con los *flip-flops* necesarios para crear el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda el circuito completo como **E1a**.
d. A la vista del circuito resultante, y suponiendo que hay *flip-flops* conectados en las entradas y salidas, calcula el tiempo de ciclo mínimo de tu implementación (Valores de retardo: NOT=10u.t., OR/AND/XOR=30u.t., FLIP_FLOP=100u.t., ROM=70u.t.).



Ejercicio 2. Febrero 2017-18 (Proyecto Ejercicio2.circ). Queremos implementar un circuito a cargo de la gestión de volumen de un aparato de música. Como puedes observar en la figura inferior, nuestro aparato de música cuenta con un botón de *Power* y una ruleta de gestión de volumen, así como un *display* que indica el volumen actual (5 niveles). El circuito consta de dos partes, un primer CLC que se encarga de codificar el estado de acuerdo a los botones accionados (Ver Tabla), y un CLS que gestiona el nivel de volumen, indicado a través del *display* de *LEDs*.
A la vista de la tabla asociada al CLC, realiza las siguientes tareas:

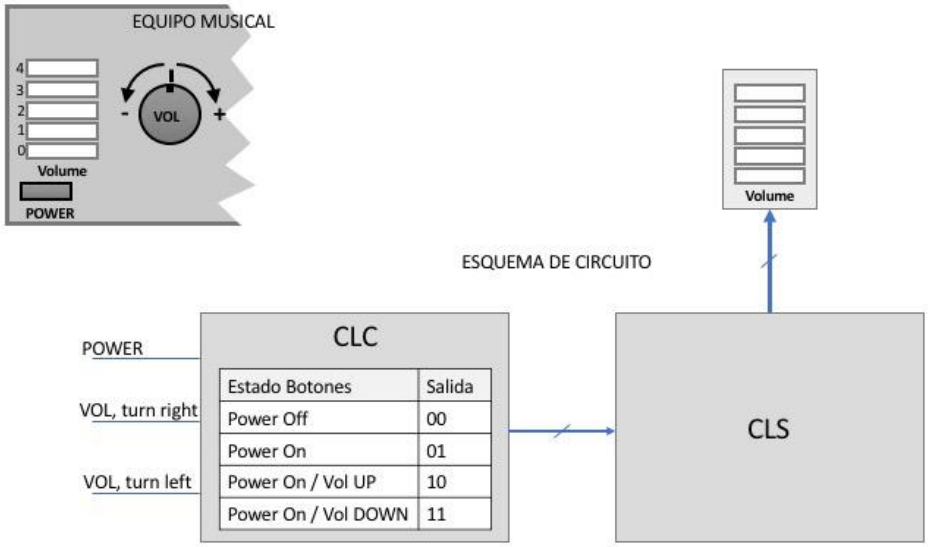
a. Determina la tabla de verdad para dicho CLC y lleva a cabo una implementación en suma de *minterms*. Guarda dicho circuito en tu proyecto como **CtrlVolumen.**

A continuación, queremos diseñar el circuito de control del CLS, a cargo del *display* de *LEDs* que indican el volumen. Los segmentos del visualizador se pueden implementar con el dispositivo *LED Bar*, disponible en la carpeta Entrada/Salida de LogiSim (analiza el funcionamiento de dicho componente).
Las especificaciones de funcionamiento del CLS se proporcionan a continuación:

1. Cuando el equipo se encuentra apagado, ningún *display* de volumen se encuentra encendido.
2. Al encender el equipo, el volumen siempre comienza a un valor de volumen medio.
3. Como máximo, el volumen varía 1 nivel por ciclo.
4. Al alcanzar el nivel máximo, éste se mantiene si seguimos girando la ruleta para aumentar el volumen (lo mismo sucede para el nivel mínimo).

Realiza las siguientes tareas:

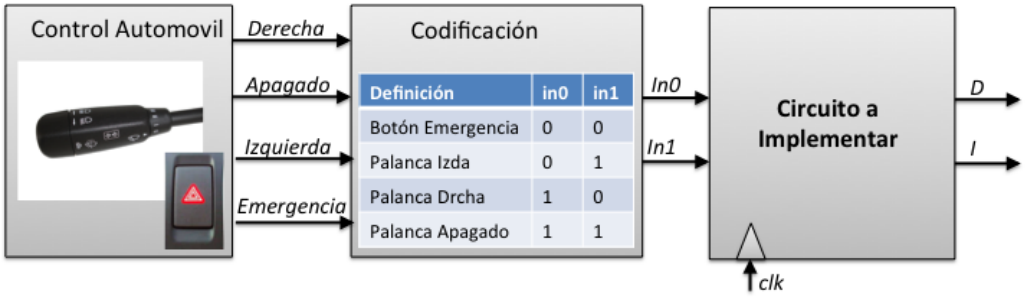
a. Obtén el grafo de estados del circuito descrito. Determina las tablas de verdad (Transiciones y Salida) correspondientes al grafo.
b. Implementa mediante una ROM el CLC de estado siguiente y otra ROM para el CLC de salida Conecta todos los circuitos creados con los *flip-flops* necesarios para crear el circuito completo (el de la imagen inferior). Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. En ese punto, guarda el circuito completo como **EquipoMusical**.



Ejercicio 3. Parcial 2015-16 Para este ejercicio, usa el proyecto denominado **Ejercicio3.circ**. Para el circuito CLS de dicho proyecto, deberás llevar a cabo un análisis mediante simulación y obtener el grafo de estados completo. Ten en cuenta que ***x*** e ***y*** representan las entradas del circuito secuencial, y ***w*** es la salida del mismo.

Ejercicio 4. Febrero 2016-17 (Proyecto Ejercicio4.circ) Se pretende diseñar el sistema de encendido de intermitencia de un coche. Para ello hay que implementar, de acuerdo con la figura adjunta, un circuito secuencial que cumpla las siguientes especificaciones:

- Cuando la palanca se coloque en la posición DERECHA, se deberá encender la luz identificada como D. Cuando la palanca se coloque en la posición IZQUIERDA, se deberá encender la luz identificada como I.
- Cuando la palanca se coloque en la posición central (APAGADO) no se encenderá ninguna luz.
- Cuando se active el interruptor de EMERGENCIA, se activarán ambas luces simultáneamente, independientemente de la posición de la palanca (la entrada de emergencia tiene prioridad absoluta).



Realiza las siguientes tareas:

1. Determina el grafo de estados del circuito, así como su tabla de transiciones/salidas.
2. implementa el circuito secuencial utilizando dos *ROMs*, una para las transiciones y otra para las salidas. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda tu circuito solución como **Ej4Sol**.

Ejercicio 5. Parcial 2015-16 (Proyecto Ejercicio5.circ) Deseamos crear un circuito que controle el pago en una máquina de refrescos. Cada refresco cuesta 1'50 euros, y la máquina acepta monedas de 1 euro y 50 céntimos. La máquina no devuelve monedas ni da cambio. Utilizaremos 2 entradas [COIN, EURO] y una salida [VALID].

- El valor COIN=1 indica que la moneda que se ha introducido es válida, mientras que un valor 0 indica que la moneda introducida no es válida, o no se ha introducido moneda.
- La señal EURO sólo tiene sentido si COIN=1 y se ignora en cualquier otro caso. Cuando su valor es 1, indica que la moneda introducida es una moneda de 1 euro, si el valor es 0, la moneda introducida es de 50 céntimos.
- VALID toma valor 1 cuando el refresco se considere pagado (1,50 euros o más) durante un único ciclo, y el resto del tiempo tendrá valor 0.

Determina el grafo de estados completo para que la máquina funcione. Calcula la tabla de transiciones/salidas e implementa el circuito usando una ROM y los biestables que sean necesarios. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda tu circuito solución como **Ej5Sol**.

Ejercicio 6. Febrero 2015-16 (Proyecto Ejercicio6.circ) Deseamos construir un circuito secuencial con una señal de entrada A y dos señales de salida X,W. El valor inicial de X,W es cero (no ha llegado ningún 1). Tras la aparición del primer 1 en A, El valor de X se mantiene a 1 hasta que se reinicie la cuenta. W toma valor 1 si A ha sido 1 durante los dos últimos ciclos (consecutivos). La cuenta se reinicia el ciclo siguiente a que W tome valor 1.

a. Obtener el grafo de estados, la tabla de transiciones y la de salida.
b. Implementar la tabla de salida mediante suma de minterms y la tabla de transiciones mediante decodificador y puertas or (utiliza el decodificador suministrado en LogiSim). Añade pìnes de entrada/salida y guarda tus circuitos como **transiciones** y **salida**.
c. Une las cápsulas de los circuitos anteriores mediante Flip-Flops para construir el circuito secuencial correspondiente al grafo de estados. Guarda el circuito generado como **Ej6Sol**.
d. Determina el camino crítico y tiempo de ciclo del circuito generado (Valores de retardo: NOT=10u.t., OR/AND/XOR=30u.t., FLIP_FLOP=100u.t., Decoder=70u.t.).

Ejercicio 7. Septiembre 2015-16 (Proyecto Ejercicio7.circ). Deseamos construir un circuito secuencial contador de N ciclos, con una señal de entrada (Pulsador) y una señal de salida (Delay). El valor inicial de Delay es 0. Tras activar el Pulsador, se inicia la cuenta de ciclos y cuando se alcanzan los N ciclos, el valor de Delay cambia a 1. La señal delay se mantiene a dicho valor hasta que se vuelva a activar el Pulsador, que fija Delay a 0 y reinicia la cuenta de ciclos. Las activaciones del Pulsador durante la cuenta (Delay=0) y hasta que se alcancen los N ciclos son ignoradas.

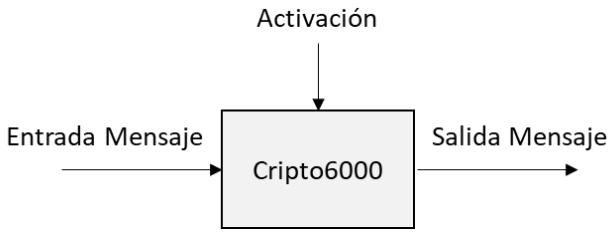
a. Obtener el grafo de estados del circuito. Indica la relación de N con el número de biestables necesarios para su implementación.
b. Implementa mediante una ROM y los biestables necesarios un circuito contador de 3 ciclos, repite el proceso para otro de 6 ciclos. Añade pines de entrada/salida y guarda dichos circuitos con el nombre Contador3 y Contador6 respectivamente.

Utilizaremos los circuitos generados (sus cápsulas) para el diseño de un semáforo de peatones con el siguiente funcionamiento: Mientras no se active el Pulsador (Pulsador=0), el semáforo permanece a VERDE por tiempo indefinido. Cuando se activa Pulsador, se encenderá en el siguiente ciclo de reloj la luz AMBAR, sin apagarse la VERDE, y tras 3 ciclos, se apagarán ambas y se encenderá la luz ROJA durante 3 ciclos, finalizados los cuales se volverá a la situación inicial, con sólo la luz VERDE encendida. El Pulsador solamente es atendido cuando está encendida la luz VERDE.

c. Obtener el grafo de estados del circuito, sus entradas serán 3 (Pulsador, Contador3, Contador6) y sus salidas 3 (VERDE, AMBAR, ROJO), indicando qué colores están activados en cada momento.
d. Haciendo uso de los circuitos creados en el apartado a y los LEDs de la librería para implementar la salida, construye con LogiSim el semáforo completo. Guarda el circuito con el nombre **E7Sol**.

Ejercicio 8. Parcial 2018-19 (Proyecto Ejercicio8.circ) Se desea diseñar un circuito secuencial que permita encriptar mensajes con un mecanismo muy simple. El circuito tiene una señal de activación, una entrada por la que entran los bits del mensaje (de uno en uno), y una única salida por la que sale el mensaje encriptado, tal y como se puede ver en la figura. Su funcionamiento es sencillo:

1) Mientras la señal de ***activación*** toma valor 1, el circuito debe cambiar el valor del bit de entrada en ciclos alternativos. Es decir, para una entrada 001001010, la salida debe ser **1**0**0**11**1**11 (los bits en negrita han sido cambiados)
2) Mientras la señal de ***activación*** toma valor 0, la salida del circuito debe ser igual que la entrada.



a. Obtén el grafo de estados del circuito descrito.
b. Determina las tablas de verdad (Transiciones y Salida) correspondientes a dicho grafo.
c. Utiliza *LogiSim* para crear la *ROM* correspondiente al CLC de estado siguiente e implementa el CLC de salida como suma de minterms.
d. Conecta los circuitos creados con los *flip-flops* necesarios para crear el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. En ese punto, guarda el circuito completo como **E8Sol**.
e. A la vista del circuito resultante, y suponiendo que hay *flip-flops* conectados en las entradas y salidas, calcula el tiempo de ciclo mínimo de tu implementación (Valores de retardo: NOT=10u.t., OR/AND=20u.t., FLIP_FLOP=100u.t., ROM=70u.t.).

Ejercicio 9. Final 2018-19 Dado el circuito secuencial proporcionado con el nombre **Ejercicio9.circ**:

a) Obtén el grafo de estados haciendo uso de *LogiSim*.
b) Si el retardo de la ROM es de 70 u.t, los *flip-flop* 100 u.t y el multiplexor 60 u.t, ¿cuál es el tiempo de ciclo mínimo del circuito? Considera un tiempo de estabilización en entrada y salida de 90 u.t.

Ejercicio 10. Parcial 2019-20 (Proyecto Ejercicio10.circ). Vamos a implementar un circuito secuencial para el control de un robot seguidor de líneas muy sencillo, con un esquema como el que se muestra en la figura inferior. El robot cuenta con dos ruedas motrices y dos sensores ópticos para mantenerse sobre un carril negro. Cada rueda se controla con dos señales, una para indicar si debe haber giro (on=1, off=0) y la otra para indicar la dirección de giro (forward=1, reverse=0). Los sensores generan un valor "1" en presencia del color negro y un "0" para otro color. El objetivo es mantener la línea negra siempre entre los dos sensores.

Diseña el circuito secuencial de control de dicho robot atendiendo a las siguientes especificaciones:

- El robot avanza en línea recta si ningún sensor detecta color negro.
- Si se detecta una desviación (los sensores L o R se activan), las ruedas motrices deben hacer girar al robot. Para ello, solamente una de las ruedas avanzará, quedando la otra quieta, hasta volver a centrar el robot sobre la línea.
- Si se detecta línea bajo los dos sensores, el robot ha llegado a meta. En ese momento se quedará parado hasta que alguno de los sensores deje de detectar el color negro (el robot se desplaza de forma manual).

Lleva a cabo las siguientes tareas:

a. Determina el grafo de estados que controlará el robot. Toma todas las decisiones que consideres oportunas y explícalas en tu hoja de examen.
b. Utiliza *LogiSim* para crear la ROM correspondiente al CLC de estado siguiente. Implementa el CLC de salida como suma de minterms. Guarda la implementación en minterms con el nombre **minterms**.
c. Conecta los circuitos creados con los *flip-flops* mínimos necesarios para crear el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda el circuito completo como **E10Sol**.
d. A la vista del circuito resultante, y suponiendo que hay *flip-flops* conectados en las entradas y salidas, calcula el tiempo de ciclo mínimo de tu implementación (Valores de retardo: NOT=20u.t., OR/AND=40u.t., FLIP_FLOP=100u.t., ROM=70u.t.).

Ejercicio 11. Parcial 2019-20

El circuito proporcionado como **Ejercicio1.circ** es de tipo secuencial con 2 entradas y 2 salidas. Responde a las siguientes preguntas:

a. Obtén el grafo de estados del circuito. Anota tu resultado en la hoja de examen.
b. Determina si es un circuito tipo *Moore* o *Mealy*. Razona tu respuesta.
c. Determina el retardo máximo de la capsula **ROM** para que el tiempo de ciclo mínimo se corresponda con el camino biestable-salida. Calcula el tiempo de ciclo mínimo en ese caso si el *Flip Flop* tiene un retardo de 100 u.t., las puertas AND y OR de 20 u.t., la NOT de 10 u.t. y la XOR de 50 u.t. Asume que las entradas y salidas están conectadas a biestables.

Ejercicio 12. Parcial 2020-21 (Proyecto Ejercicio12.circ) El circuito secuencial con el que vamos a trabajar responde a las siguientes fórmulas, en las que t y t+1 representan los ciclos correspondientes al estado actual y siguiente:

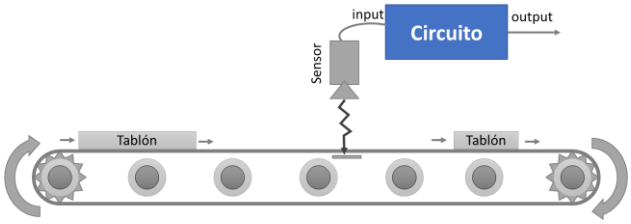
$$q_0(t+1) = q_0(t) + \overline{q_1(t)}$$
$$q_1(t+1) = IN + q_1(t)$$
$$OUT = q_1(t) \times q_0(t)$$

Para dicho circuito, lleva a cabo las siguientes tareas:

a. Con las puertas básicas (and, or, not) y *Flip-Flops* de tu librería Implementa el circuito secuencial completo, añadiendo pines de entrada/salida y la señal de reloj. Guarda dicho circuito con el nombre **E12Sol**.
b. Obtén el grafo de estados de dicho circuito haciendo uso de *LogiSim*.
c. Determina su camino crítico y tiempo de ciclo mínimo (Los *Flip-Flops* presentan un retardo de 100u.t., el retardo de las puertas es AND y OR de 20 u.t., la NOT de 10 u.t. y la XOR de 50 u.t). Asume un tiempo de estabilización para la entrada y la salida de 50 u.t.

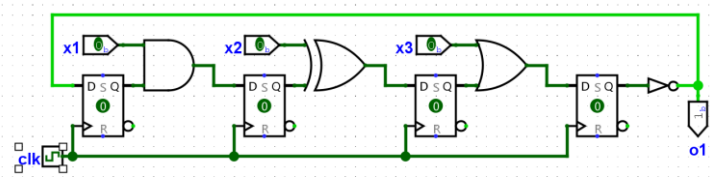
Ejercicio 13. Parcial 2020-21 (Proyecto Ejercicio13.circ) Se dispone de un sistema industrial como el que describe la figura inferior. Se trata de una cinta transportadora de tablones de madera. Los tablones son medidos haciendo uso de un detector, calculando la longitud del tablón en función del número de ciclos que el detector da señal. Cuando los tablones tienen la longitud correcta (3 ciclos con el detector activo), el tablón es dado por válido y se activa una señal que envía el tablón a seguir por la cadena. En caso contrario, si el tablón es demasiado corto o demasiado largo, el tablón sigue hasta ser descartado. El sistema nos garantiza que la distancia entre tablones es suficiente como para que pasen al menos dos ciclos de reloj sin que se produzca detección.
Diseña la máquina de estados que controla este sistema, sabiendo que:

- La máquina de estados tiene como entrada la señal enviada por el detector, que toma valor 1 cuando hay tablón, y 0 si no detecta nada.
- La máquina de estados tiene que proporcionar una salida que toma valor 1 si el tablón tiene la longitud correcta, y se envía únicamente en el ciclo en el que el tablón deja de pasar por el detector. La salida tendrá valor 0 en cualquier otro caso (no hay tablón, o el tablón es incorrecto).
- El tablón se considera correcto únicamente si su medida se corresponde con 3 ciclos de reloj. Debe ser considerado incorrecto si es más corto o más largo.



a) Obtén el grafo de estados y determina las tablas de verdad (transiciones y salidas) correspondientes a dicho grafo.
b) Utiliza *LogiSim* para implementar el CLC de salida con Decodificador y puertas Or. Guarda dicho circuito como **CLCSalida**.
c) Para el CLC de estado siguiente vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada actuará como selector del Multiplexor. Conecta los circuitos creados con los *Flip-Flops* necesarios para obtener el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda el circuito completo como **E13Sol**.

Ejercicio 14. Parcial 2020-21 Dado el circuito de la figura inferior, determina su camino crítico y tiempo de ciclo mínimo (Los *Flip-Flops* presentan un retardo de 100u.t., el retardo de las puertas es AND y OR de 20 u.t., la NOT de 10 u.t. y la XOR de 50 u.t). Asume un tiempo de estabilización para la entrada y la salida de 50 u.t.



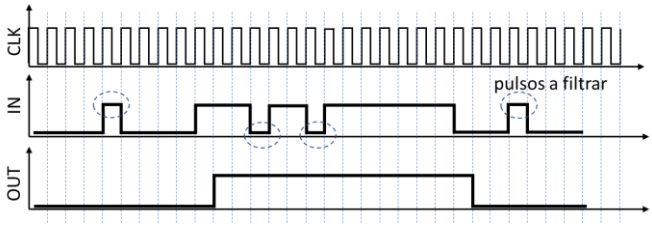
Ejercicio 15. Parcial 2020-21 (Proyecto Ejercicio15.circ) Vamos a diseñar una máquina de estados con dos señales de entrada (A, B) y una señal de salida (C) que funciona de acuerdo a las siguientes especificaciones (TRUE=1, FALSE=0):

- La máquina arranca en un estado inicial que solamente abandona si A×B=TRUE.
- Una vez se abandona el estado inicial, la máquina avanza por N estados intermedios cada vez que A+B=TRUE. En caso contrario retrocede al estado anterior.
- Al llegar al último estado, la máquina simplemente retorna al estado inicial y comienza de nuevo.
- La salida C se activa cuando el circuito se encuentra en sus estados inicial y final, y se desactiva mientras se recorren los estados intermedios.

Realiza las siguientes tareas:

a. Obtén el grafo de estados de la máquina descrita cuando N=3 (tres estados intermedios) y determina las tablas de verdad (transiciones y salidas) correspondientes a dicho grafo.
b. Utiliza *LogiSim* para implementar el CLC de salida con Decodificador y puertas Or. Guarda dicho circuito como **CLCSalida**.
c. Vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada A actuará como selector del Multiplexor. Utiliza *LogiSim* para crear la *ROM* correspondiente al CLC de estado siguiente para este tipo de síntesis, conecta los circuitos creados con los *Flip-Flops* y componentes necesarios para obtener el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda el circuito completo como **E15Sol**.

Ejercicio 16. Final 2020-21 (Proyecto Ejercicio16.circ) Vamos a diseñar un circuito para eliminar pulsos de ruido en una señal. Nuestra máquina de estados de tipo Moore tendrá una única señal de entrada (IN) y la señal de salida (OUT) con el ruido eliminado. Se define un pulso de ruido como un cambio de valor que solamente dura un ciclo (dicho pulso puede tener valor 0 o valor 1). Observa las señales de la figura inferior para entender el problema.



Realiza las siguientes tareas:

a. Obtén el grafo de estados de la máquina descrita y determina las tablas de verdad (transiciones y salidas) correspondientes a dicho grafo.
b. Vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada IN actuará como selector del Multiplexor. Utiliza *LogiSim* para crear la *ROM* correspondiente a este tipo de síntesis, conecta los circuitos creados con los *Flip-Flops* y componentes necesarios para obtener el circuito secuencial completo. Añade pines de entrada/salida y sondas que muestren los estados actual y siguiente para comprobar el funcionamiento de tu circuito. Guarda el circuito completo como **E16Sol.**

Ejercicio 17. Parcial 2021-22 Tenemos el encargo de diseñar el circuito secuencial (CLS) de control del alumbrado para unas luces de árbol de navidad. El proyecto Ejercicio17.circ (Circuito Árbol) contiene el esquema a realizar. Como puedes observar, el CLS (circuito CLSLuces) cuenta con dos entradas, la señal de reloj (CLK) y una entrada que determina la velocidad de parpadeo de las luces (SPEED). El CLS tiene una única señal de salida, que controlará si las luces se encienden o se apagan (LUZ)[1]. El CLS del circuito se encuentra vacío, y es necesario realizar su implementación. Para ello, hay que cumplir los siguientes requisitos:

- El botón de ON/OFF que se encuentra fuera del CLS tiene el funcionamiento esperado para un botón de encendido y apagado.
- El CLS puede operar a tres velocidades de parpadeo: Normal (2 ciclos luz on – 2 ciclos luz off), Rápido (1 ciclo luz on – 1 ciclo luz off), Permanente (Siempre luz on).

[1] Con Luz=1, las luces están encendidas, con luz=0, están apagadas.

- Cada vez que se pulsa el botón SPEED se modifica la velocidad de parpadeo, en el siguiente orden: Permanente – Normal – Rápido – Permanente – Normal – Rápido - …
- Cuando se reinicia el circuito (botón Reset de los flip flops) la velocidad de parpadeo inicial es Normal.

Realiza las siguientes tareas:

a. Obtén el grafo de estados del circuito descrito y determina las tablas de verdad (Transiciones y Salidas) correspondientes a dicho grafo.
b. Utiliza LogiSim para implementar el CLC de salida con Decodificador y puertas Or. Guarda dicho circuito como **CLCSalida**.
c. En el circuito CLSLuces vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada SPEED actuará como selector del Multiplexor. Utiliza LogiSim para crear la ROM correspondiente al CLC de estado siguiente para este tipo de síntesis y conecta los circuitos creados con los Flip-Flops y componentes necesarios para obtener el circuito secuencial completo.

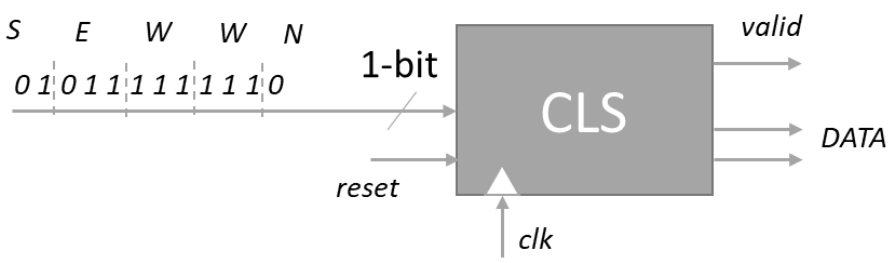
Ejercicio 18. Ordinaria 2021-22 (Proyecto Ejercicio18.circ). La codificación Huffman es un algoritmo de compresión de datos en el que los símbolos presentan códigos de longitud variable, siendo los símbolos más habituales representados con menos bits. Vamos a plantear un sistema de gestión de movimiento basado en los puntos cardinales. Podemos utilizar una codificación convencional (N:00, S:01, E:10, W:11), pero si sabemos que hay algunos puntos que se repiten con más frecuencia, podríamos codificarlo mediante el algoritmo de Huffman (N:0, S:10, E:110, W:111).

El objetivo de este ejercicio es implementar un circuito secuencial de tipo Moore capaz de hacer la conversión entre valores codificados con Huffman y su representación convencional. Éstas son las especificaciones del circuito a implementar:

- Nuestro circuito tiene dos señales de entrada. Una corresponde a los datos sobre los puntos cardinales (Huffman), que se reciben en serie (un bit por ciclo). La otra es una señal de reset (ver figura inferior).
- Las salidas del circuito serán tres bits, un bit de validez y el dato en formato convencional.
- Nuestro circuito cuenta con un estado inicial de “reposo” en el que no hay transmisión de datos.
- La señal de reset marca el comienzo de una transmisión de coordenadas válida cuando toma valor cero. En el momento que tome valor 1 se vuelve al estado de reposo.
- Cada vez que se complete la recepción de un valor de coordenada, el bit de validez toma valor uno y se muestra el valor de salida apropiado. En el siguiente ciclo ya aparece el primer bit del siguiente punto cardinal.
- Mientras el bit de validez sea cero, el valor de salida es irrelevante.

Lleva a cabo las siguientes tareas:

a. Obtén el grafo de estados del circuito descrito y determina las tablas de verdad (Transiciones y Salidas por separado) correspondientes a dicho grafo.
b. Utiliza LogiSim para implementar el CLC de salida con Decodificador y puertas Or. Guarda dicho circuito como **CLCSalida**.
c. Vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada reset actuará como selector del Multiplexor. Utiliza LogiSim para crear la ROM correspondiente al CLC de estado siguiente para este tipo de síntesis, conecta los circuitos creados con los Flip-Flops y componentes necesarios para obtener el circuito secuencial completo. Guarda dicho circuito como **E18Sol**.

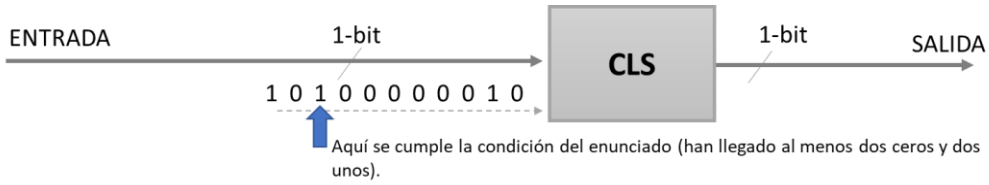


Ejercicio 19. Extraordinaria 2021-22 (Proyecto Ejercicio19.circ). El objetivo de este ejercicio es implementar un circuito secuencial de tipo Moore con las siguientes especificaciones:

- Nuestro circuito tiene una señal de entrada y otra de salida.
- La salida toma valor inicial 0 y se mantiene así hasta la llegada de al menos dos ceros y dos unos por la entrada, sin importar el orden de aparición (ver ejemplo en figura inferior). En dicho momento la salida toma valor uno y se mantiene así de manera indefinida.

Lleva a cabo las siguientes tareas:

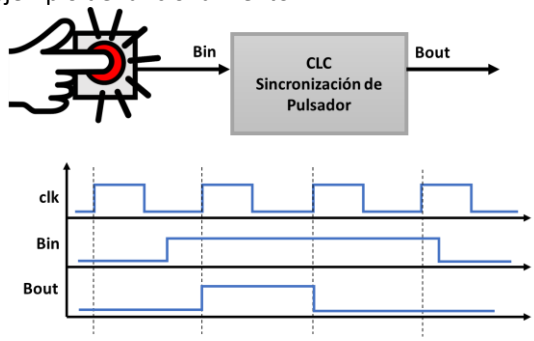
- Obtén el grafo de estados del circuito descrito y determina las tablas de verdad (Transiciones y Salidas por separado) correspondientes a dicho grafo.
- Utiliza LogiSim para implementar el CLC de salida con suma de minterms. Guarda dicho circuito como **CLCSalida**.
- Vamos a hacer una implementación de tipo ROM+Multiplexor, en la que la señal de entrada actuará como selector del Multiplexor. Utiliza LogiSim para crear la ROM correspondiente al CLC de estado siguiente para este tipo de síntesis, conecta los circuitos creados con los Flip-Flops y componentes necesarios para obtener el circuito secuencial completo. Guarda dicho circuito como **E19Sol**.



Ejercicio 20. Parcial 2022-23 En el proyecto **Ejercicio20.circ** encontrarás un circuito secuencial (CajaFuerte) que gestiona el mecanismo de apertura de una caja fuerte. La combinación consiste en una secuencia de tres dígitos en hexadecimal (Por ejemplo 3-A-F). Realiza las siguientes tareas:

1. Determina la combinación que abre la caja fuerte (Señal Open=1). Describe en tu hoja de examen el proceso que has seguido para hallar la solución.
2. Lleva a cabo un análisis temporal del circuito para determinar si funcionará de forma correcta con una señal de reloj de 200 u.t. Los tiempos de propagación a considerar serán los siguientes: And (20u.t.), Or(20u.t.) y Not(10u.t.)., FF=100u.t. Considera que tanto entradas y salidas están conectadas a Flip Flops.

Ejercicio 21. Parcial 2022-23 (Proyecto Ejercicio21.circ) En este ejercicio deberás implementar un circuito secuencial que limita las pulsaciones de un botón a una duración de un ciclo, independientemente del tiempo que se mantenga el botón pulsado (la pulsación de botón siempre dura al menos un ciclo). La figura inferior muestra un ejemplo de funcionamiento.



Lleva a cabo las siguientes tareas:

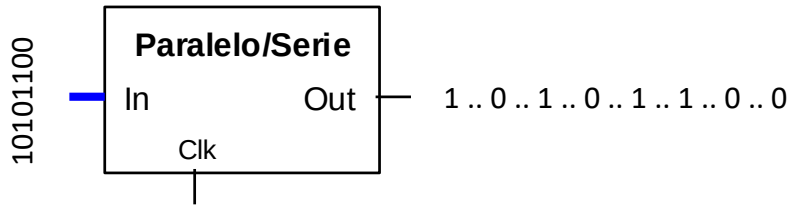
1. Dada la descripción de su funcionamiento, determina el grafo de estados y la tabla de transiciones/salidas (una sola tabla).
2. Usando la señal Bin como selector del multiplexor, lleva a cabo una implementación de tipo Minterms+Multiplexor (Haz síntesis en suma de minterms en vez de utilizar una ROM). Conecta los componentes necesarios para obtener el circuito secuencial completo y guarda dicho circuito como **E21Sol**.

Ejercicio 22. Final 2022-23 (Proyecto Ejercicio22.circ) En este ejercicio vamos a implementar un conversor de señales de paralelo a serie, cuyo esquema de funcionamiento puedes ver en la figura inferior. El conversor recibe 8 bits en paralelo, y los va sacando de uno en uno en ciclos consecutivos de reloj. Para ello, vamos a comenzar con la siguiente tarea:

1. Diseña un circuito secuencial sin señales de entrada y un único bit de salida que cumpla lo siguiente: el circuito deberá, de forma cíclica, tener un ciclo la señal de salida a 0 y 7 ciclos con la salida de valor 1 (Es decir, la salida tomará en ciclos consecutivos los valores 0-1-1-1-1-1-1-1-0-1-1-1-1-1-1-1…). Implementa el circuito secuencial con el método que creas adecuado y guarda el resultado como **CuentaCiclos**.

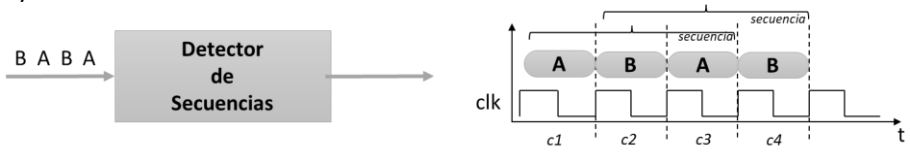
Carga el fichero LibEjercicio22.circ como librería en tu proyecto. En dicha librería encontrarás un circuito de Shift, que realiza dos operaciones distintas sobre números de 8 bits. Los ciclos de reloj que comienzan con el bit de selección (SL/In) a valor 0, se carga el valor de entrada de 8 bits (entrada) en los biestables de la unidad. Los ciclos que comienzan con el bit de selección a valor 1, se realiza un shift lógico a la derecha sobre el valor almacenado. Haciendo uso del circuito de shift y del diseñado en el apartado anterior, completa la implementación del conversor:

2. El circuito tendrá un comportamiento que se repite cada 8 ciclos: en el primer ciclo recibirá los 8 bits en paralelo que se desean transmitir (a través del bus "In" de la figura) y mostrará el bit menos significativo en la salida (Out), y los siete ciclos siguientes, debe ir sacando los bits de entrada uno a uno por el mismo cable de salida (Out) de menos significativo a más significativo. Guarda el circuito resultante como **E22Sol**



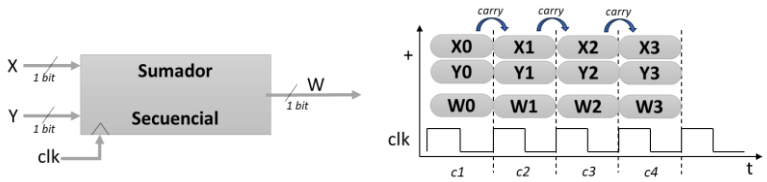
Ejercicio 23. Parcial 2023-24 (Proyecto Ejercicio23.circ) Queremos diseñar un circuito capaz de detectar secuencias del tipo ABA (mismo valor para las posiciones inicial y final y distinto a la posición intermedia). El circuito debe cumplir los siguientes requisitos:

- Los datos entran al circuito por una línea serie (1 bit/ciclo).
- La señal de salida del circuito indica si para los últimos tres valores se detecta la secuencia o no.
- Si los valores ABAB llegan al circuito en ciclos consecutivos, se detecta la secuencia en los ciclos 3 y 4.



Escribe en la hoja del examen el grafo de estados del circuito a implementar. A continuación, lleva a cabo una síntesis de tipo (Decodificador y puertas OR) + Multiplexor, guardando el circuito completo como **E23Sol**

Ejercicio 24. Final 2023-24 (Proyecto Ejercicio24.circ) Las implementaciones de sumador ripple-carry que hemos usado en clase consisten en encadenar circuitos Full-Adder (de 1 bit) mediante sus señales de Carry (Cout del bit n = Cin del bit n+1). Date cuenta de que, con un solo Full-Adder es muy sencillo hacer la misma operación de suma con un circuito secuencial que ciclo a ciclo vaya sumando desde el bit menos significativo al bit más significativo (La figura inferior describe el proceso).



Como primer paso, en este ejercicio debes implementar el sumador **secuencial** indicado (figura de la izquierda) con un único full-adder. Guarda tu circuito como **secuencial**. Determina el camino crítico y tiempo de retardo del circuito implementado (FF: 50u.t., AND: 30u.t, OR:20u.t., NOT: 10u.t.).
Haciendo uso del circuito anterior, vamos a llevar a cabo la implementación de un sumador genérico para números de N bits mediante circuitos secuenciales. Para controlar el número de bits de los sumandos vamos a implementar un circuito secuencial que cuente N ciclos, con una señal de entrada (Reset) y una señal de salida (Carry_on). Carry_on tiene un valor inicial de 0. Cuando la señal Reset toma valor 0, se comienza la cuenta de ciclos, y Carry_on toma valor 1 durante toda la cuenta. Cuando se alcanzan los N ciclos (estado N-1), el circuito vuelve al estado inicial (el estado con Carry_on a 0), de la misma forma que cada vez que se activa la señal de Reset. Para la implementación de este circuito lleva a cabo las siguientes tareas:

1. Obtén el grafo de estados para el caso específico de sumandos de 4 bits.
2. A continuación, lleva a cabo una síntesis de tipo (Suma de Minterms) + Multiplexor, guardándolo como **Contador**.
3. Finalmente, une tu contador con tu circuito secuencial de Suma. Para esta unión, El carry del sumador debe tomar cero al comienzo de la cuenta de ciclos (Estado inicial), mientras que en el resto de ciclos el carry de entrada será el generado por el sumador en el ciclo anterior. Guarda tu circuito como **E24Sol**.