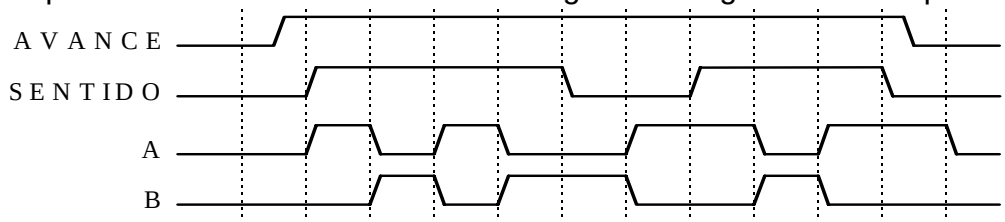
Repaso Parcial

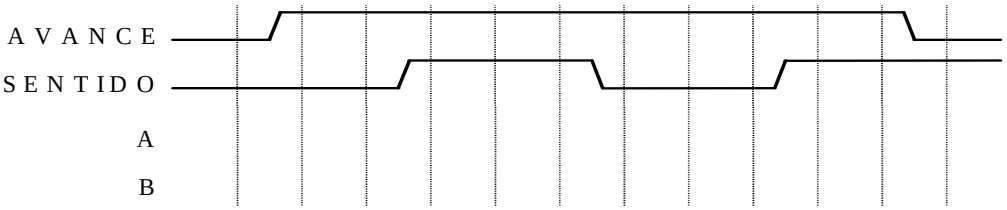
Ejercicio 1. *(ex. Marzo 1998)*
Se desea diseñar el driver de un motor que tiene dos entradas **AVANCE** y **SENTIDO**, y dos salidas **A** y **B**. La entrada **SENTIDO** está **sincronizada** con el reloj y la entrada **AVANCE** no lo está. El funcionamiento del driver es el siguiente:

- Mientras **AVANCE** = 0, las salidas deben valer **A** = **B** = 0 para que el motor esté detenido.
- Mientras **AVANCE** = 1, las salidas **A** y **B** deben oscilar en contrafase entre 0 y 1 en cada ciclo de reloj para hacer girar el motor. Cada pulso de **A** y **B** debe durar un ciclo entero de reloj.
- El comienzo y la finalización de la generación de pulsos en **A** y **B** se producen en el flanco de reloj posterior al cambio correspondiente en **AVANCE**.
- Los valores iniciales en **A** y **B** luego de la subida de **AVANCE** son a elección (**AB** = 01 o **AB** = 10).
- En caso de producirse un cambio en la entrada **SENTIDO**, las salidas **A** y **B** deben permanecer constantes repitiendo el valor que tenían en el ciclo anterior durante un ciclo de reloj más, a los efectos de que el motor cambie de sentido de giro.

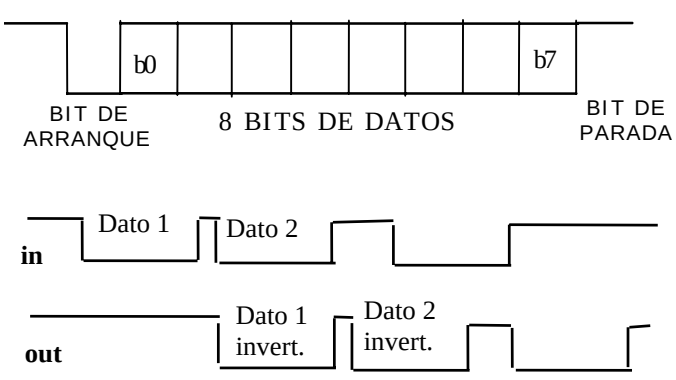
Un ejemplo del funcionamiento se da en el siguiente diagrama de tiempos:



a) Diseñar completamente un circuito **modo reloj** que funcione según la especificación descrita.
b) Dado el siguiente diagrama de tiempos, donde no se cumple la hipótesis de que la entrada **SENTIDO** es síncrona, indicar el estado y el valor de la las salidas **A** y **B** en cada instante para el circuito diseñado en (a).



Ejercicio 2. *(Ex. Febrero 1999).*
Diseñar un circuito RTL que invierta el orden de transmisión de los bits en una comunicación serial. El circuito tendrá una entrada de datos serie **in** y una salida de datos **out**. La recepción y transmisión se hace con un bit de arranque en cero, 8 bits de datos y al menos un bit de parada en uno.
Se supondrá que la señal **in** cambia sincronizada con una señal de reloj que se utilizará como reloj del sistema. El



circuito deberá tener una entrada de RESET.
Se deberá tener en cuenta que la transmisión de un byte y el siguiente pueden ser contiguas (separados por el bit de parada), con lo cual el sistema podría estar recibiendo un byte y al mismo tiempo trasmitiendo el anteriormente recibido.

Nota: se admitirán soluciones en más de un bloque RTL, siempre que se especifiquen sus conexiones e interfaz.

Ejercicio 3. *(ex. Agosto 1996)*
a) Dado el circuito de la figura 1, completar en el diagrama de tiempos de la figura 2, las señales COUNT y FIN.
Suponiendo que la señal RESET está sincronizada con el flanco creciente de RELOJ, calcular el tiempo que transcurre entre el flanco creciente de RESET y el flanco creciente de FIN, en función del período de RELOJ (T_{CK}). Se supondrá que el valor de TIEMPO es constante e igual a 5.

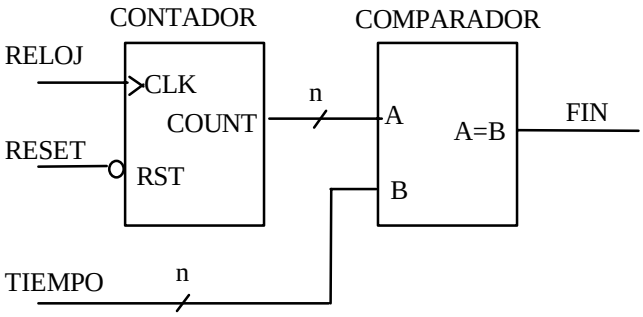


Figura 1

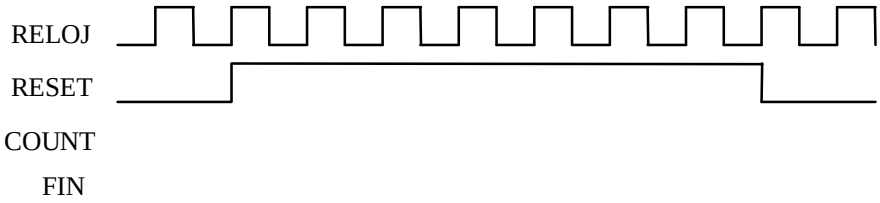


Figura 2

b) Se tiene una señal SC compuesta por pulsos a nivel cero espaciados un tiempo T_H constante. Aleatoriamente en la misma señal, aparecen pulsos a cero en la mitad del período T_H, llamados pulsos de preecualización, que deben ser eliminados. Para ello se plantea un circuito como el de la figura 3, donde el bloque TEMPORIZADOR corresponde al circuito de la parte (a).
Debe diseñarse el circuito **modo nivel** de manera de entregar a la salida la señal SH, que es igual a la señal SC pero sin los pulsos de preecualización (ver figura 4).

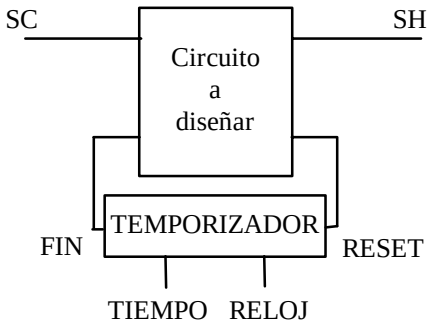


Figura 3

Este circuito manejará la entrada RESET del temporizador y tendrá como entrada la señal FIN. El valor de TIEMPO debe calcularse de manera de que el tiempo que mide el TEMPORIZADOR sea aproximadamente igual a 3/4 T_H. Indicar también la cantidad de bits mínima que debe tener el contador del TEMPORIZADOR.

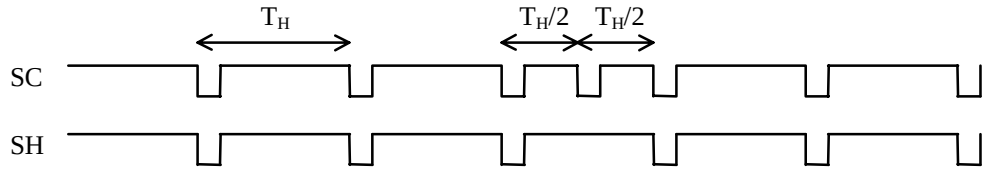
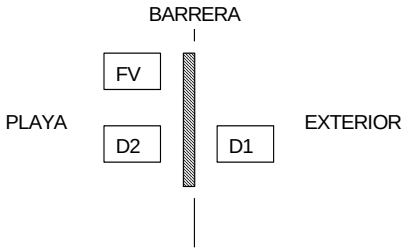


Figura 4

NOTA: para la parte b) se supondrá que T_{CK}=100 ns, T_H=64 μs, y el ancho de los pulsos a cero de la señal SC es del orden de T_H/10.

Ejercicio 4. *(ex. Agosto 1998)*
Se desea construir un dispositivo que controle la barrera de una playa de estacionamiento. Recibirá como entrada las señales provenientes de 3 dispositivos, dispuesto según la figura:

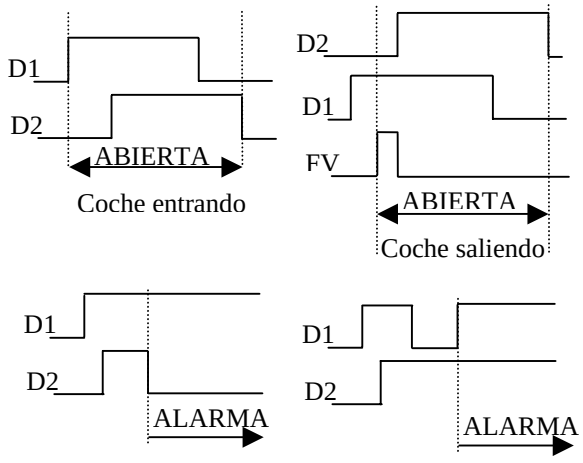
- D1: detector de coche que entra
- D2: detector de coche que sale
- FV: detector de ficha válida



El sistema contará con dos salidas:

- BA: control de la barrera (BA=1 levanta la barrera)
- AL: señal de alarma (activa también por nivel alto)

La figura 1 muestran la secuencia normal de activación de los detectores cuando un coche entra o sale. Cuando un coche entra se levanta la barrera durante el intervalo indicado en la figura y el sistema le entrega una tarjeta (esto no forma parte del problema). Cuando un coche sale se activa primero D2 y se debe esperar la señal de Ficha Válida (FV=1) para levantar la barrera.
En el estado inicial el sistema espera con la barrera baja a que se active uno de los detectores, si se activan ambos al mismo tiempo el sistema no cambia de estado. Para el caso de un coche entrando se pide que se detecten especialmente los siguientes casos:

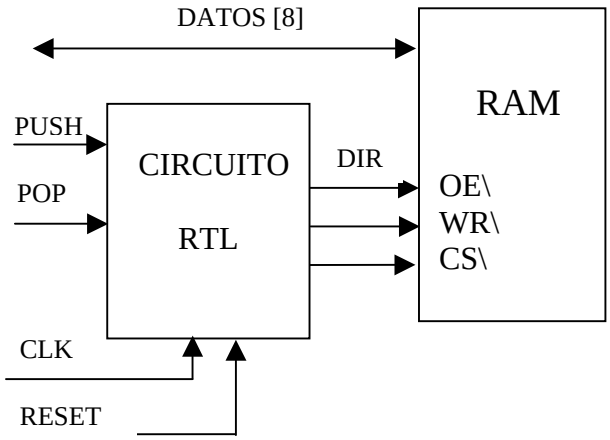


En las situaciones marcadas a partir de la línea punteada el sistema debe pasar a un estado "ALARMA" del que sólo puede salir con RESET. Durante dicho estado se bajará la barrera y se activará la salida AL.

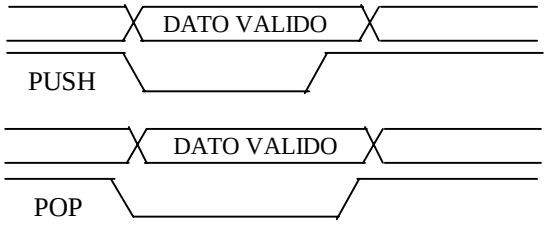
Se pide: a) Diagrama de estados del bloque de control.
b) Descripción RTL del circuito.
c) Bloque de datos y bloque de control.

NOTA: se supondrá que las únicas secuencias de entradas posibles son las indicadas en las figuras.

Ejercicio 5. *(Ex. Setiembre 1993)*
Diseñar un circuito RTL que maneje una RAM de modo de hacer un STACK hardware. La RAM es de 256x8.



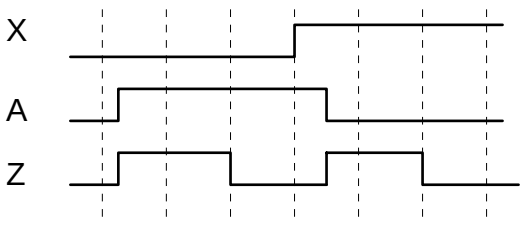
Las entradas al circuito serán **push** y **pop** que se activan cuando se quiere realizar alguna operación de escritura o lectura respectivamente.
Ciclo de push - Se supondrá que los datos están válidos alrededor del flanco de subida de la señal **push** un tiempo suficiente para satisfacer los requerimientos de la RAM.
Ciclo de pop - El dato correspondiente deberá estar disponible para que un dispositivo externo lo pueda leer con el flanco de subida de la señal **pop**.



El ancho de los pulsos en **push y pop** es el que determina la duración de los ciclos (no un cierto número de períodos de reloj del RTL). El circuito deberá funcionar correctamente con un ancho mínimo en dichos pulsos mayor que **dos** períodos de reloj.
Las líneas de datos se conectarán directamente a la RAM y se deberán generar las salidas de control hacia la misma.
El circuito deberá tener una entrada asíncrona de **reset** que lo lleve a su estado inicial.
Se podrá contar con bloques **combinatorios** que generan las funciones incremento (INC) y decremento (DEC) de un dato de 8 bits.

a) Especificación RTL del circuito.
b) Bloques de datos y de contol.
c) Para el circuito diseñado dibujar detalladamente las formas de onda de todas las señales para los ciclos de push y pop de la figura, indicando en qué paso se encuentra el circuito y los valores de los registros y las salidas en cada instante.

Ejercicio 6. *(Ex. Marzo 1996)*
Diseñar un circuito secuencial modo reloj que detecte cambios de nivel en una entrada A. El circuito tendrá una entrada de control X que le indicará si debe detectar flancos crecientes (X=0) o flancos decrecientes (X=1). La entrada X está sincronizada con el reloj.
La salida Z deberá ir a 1 inmediatamente después que se produzca el flanco que se quiere detectar en la entrada, y deberá bajar a cero con el segundo flanco de reloj a partir de esta situación independiente del valor de las entradas. (Mientras Z vale 1 no se detectarán los flancos en A).
Los tiempos en que la entrada A permanece en cero o en uno siempre son mayores que un período de reloj.
(a) Diagrama de estados.
(b) Circuito mínimo (utilizando FFJK).



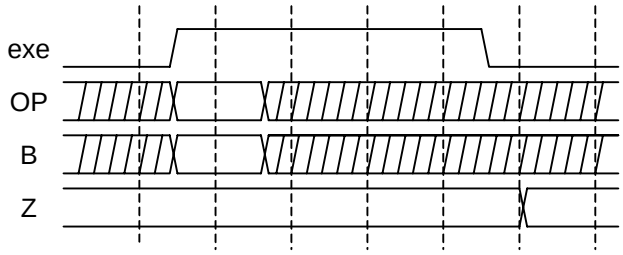
Formas de onda de ejemplo

Ejercicio 7. *(2do. Parcial 1999)*

Se desea diseñar un circuito RTL que realice operaciones lógicas y aritméticas entre un operando **B** de 8 bits que se ingresa al circuito y un registro interno **A** también de 8 bits; el resultado se almacena en el propio registro **A**. Para indicar cuál es la operación a realizarse se cuenta con una entrada **OP** de 2 bits, codificados de acuerdo a la tabla.

OP	operación a realizar
00	AND
01	OR
10	XOR
11	incremento de A

Además se cuenta con una entrada de control **exe** que indica en qué momento se debe realizar la operación (con un flanco de subida) y en qué momento se debe mostrar el resultado en la salida **Z** (con un flanco de bajada). La temporización de esta señal se observa en el diagrama de tiempos. Se garantiza que los valores de **OP** y **B** están válidos durante un tiempo T después de la subida de **exe**, siendo T el período de reloj del sistema. La salida **Z** deberá mostrar el resultado después de la bajada de **exe**, pero no podrá demorarse más de un tiempo T en cambiar su valor. Este valor en **Z** deberá mantenerse hasta que se muestre el resultado de la próxima operación.



Se pide una descripción RTL del sistema, bloque de datos y bloque de control.