PRÁCTICA 2
CIRCUITOS SECUENCIALES

Objetivos

- Comprender el funcionamiento de un circuito secuencial modo reloj y la señal de reloj.
- Comprender las magnitudes de frecuencia que se utilizan.
- Reutilizar circuitos ya diseñados para implementar un circuito de mayor complejidad.
- Diseñar y comprobar el funcionamiento de circuitos digitales secuenciales.

En esta práctica, primeramente se verá el diseño de un contador ya implementado y cómo obtener un reloj más lento a partir del reloj de la placa DE0, seguidamente se realizará el diseño de un circuito modo reloj sencillo y por último se diseñará un circuito modo reloj más complejo.

Parte a)

En esta parte se implementará un contador en BCD circular y se probará con un reloj de frecuencia 0.745 Hz, suficientemente lento como para poder visualizar los cambios en un display de 7 segmentos. Para obtener un reloj de 0.745 Hz a partir del reloj de 50 MHz que provee la placa DE0, se dispone de un divisor de frecuencia por 8192, **Div_8192**.

Se pide:

- Estudiar el funcionamiento de **Div_8192** (ver notas y sugerencias).
- Crear un proyecto en el **Quartus II** con el circuito **ContadorBCD** de la figura 1. Analizar el circuito y simular en forma exhaustiva. El bloque **INC_BCD_1** es el diseñado en la practica 1.
- Crear el símbolo **ContadorBCD**.
- Para probar en la placa, generar un proyecto con el circuito indicado en la figura 2, donde las entradas y salidas se conectarán a las señales del mismo nombre en la placa DE0. Analizar el circuito, programar la placa y verificar el funcionamiento.

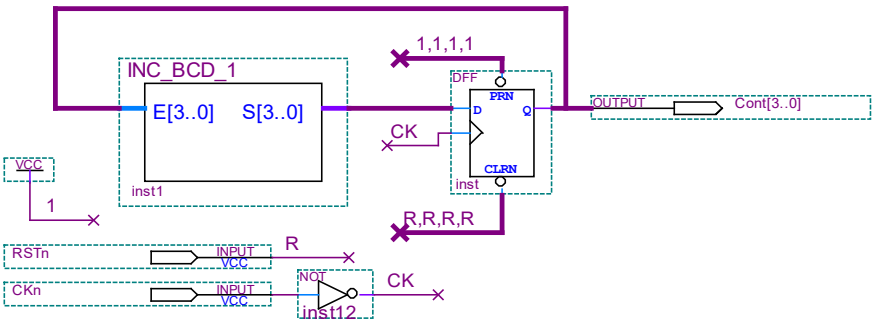

Figura 1. *Circuito ContadorBCD*

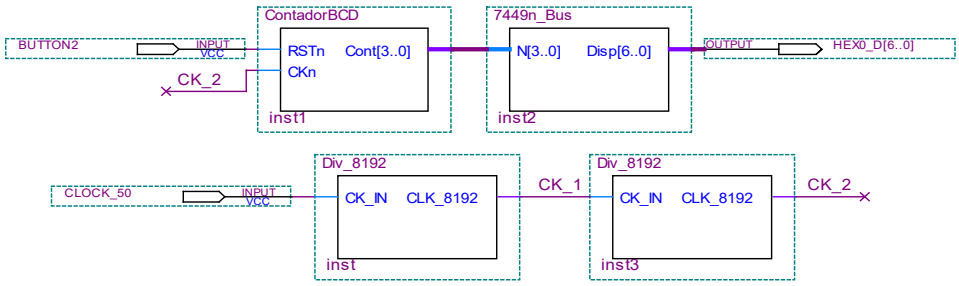

Figura 2. *Circuito de prueba para la parte a).*

Notas: - 0.745 Hz = 50 MHz / 8192 / 8192 que equivale a un período de reloj de 1.3 segundos.

- 8192 = 2^{13}, observar que cada divisor tiene 13 flip-flops.
- Se dispone del bloque **7449n_Bus,** que es un 7449 en formato BUS y su salida negada.

Sugerencias:

– Para agregar el valor de registros internos de un circuito a una simulación se debe indicar "*Registers: Pre-Syntesis"* en el campo "*Filter"* de la ventana "*Node Finder"*.
– Si se desea realizar la simulación en el circuito de prueba, para evitar que la PC demore mucho, no utilice los bloques divisores de frecuencia.

Parte b)

Cuando se oprime una tecla en un teclado PS2, este envía el código D[7..0] de la tecla oprimida y cuando se suelta dicha tecla, envía el valor F0h y a continuación, nuevamente el código de esta tecla.

Nos va a interesar detectar cuando se suelta una tecla que corresponde a un número de la parte numérica de un teclado PS2. Para esto, en la figura 3 se presenta el circuito, **KEY_VALID**, con entradas **Rdy**, **D[7..0]** y salidas **Kv** y **Q** tal que :

- **Rdy**: Indica con un pulso a 1 de un período de reloj de duración que D[7..0] es válida. Entre dos pulsos consecutivos de **Rdy** hay más de 20 períodos de reloj.
- **D[7..0]**: Es un código de 8 bits enviado por el teclado. Su valor es válido si **Rdy** = 1.

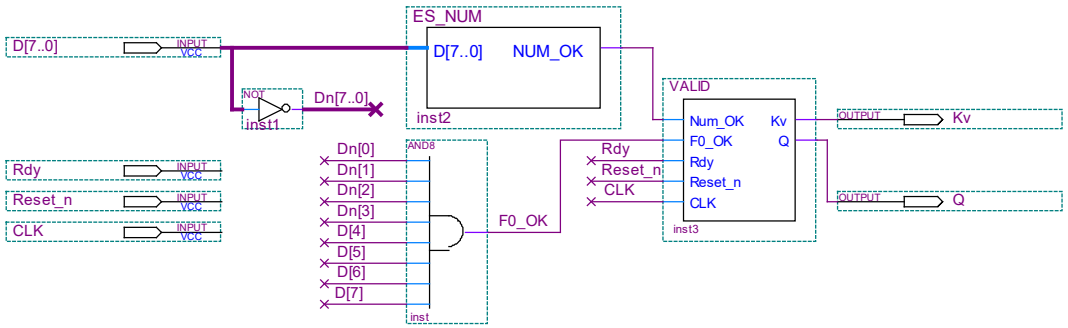

Figura 3. *Circuito KEY_VALID*

El circuito debe funcionar de forma tal que, al recibir un pulso a 1 en **Rdy**, con **D[7..0]** = F0h (F0_OK = 1) y luego, en el siguiente pulso de **Rdy**, **D[7..0]** es un código PS2 del teclado numérico (Num_OK = 1), entonces **Kv** debe ser 1 junto con este segundo pulso de **Rdy**. En otro caso **Kv** = 0. Luego de un reset, el circuito debe entender que no ocurrió ningún pulso en **Rdy**. Nunca ocurre que, en 2 pulsos consecutivos de **Rdy, D[7..0]** = F0h en ambos pulsos.

El bloque **ES_NUM** es el diseñado en la practica 1. Recordar que **Num_OK** = 1 si **D[7..0]** es un código PS2 del teclado numérico. Observar además que, por construcción, las señales **Num_OK** y **F0_OK** no pueden ser simultáneamente 1.

Para completar el circuito **KEY_VALID** es necesario diseñar el circuito modo reloj **VALID**, sensible al flanco de subida. La solución obtenida tendrá 2 estados y la salida **Q** del circuito debe corresponder con la salida Q del flip-flop utilizado. Esta última salida solo será a efectos de simulación y pruebas.

La prueba de esta parte se realizará paso a paso, con un reloj manual, de forma de generar los flancos a conveniencia utilizando botones de la placa DE0. Durante cada transición de la señal proveniente directamente de un botón, se presenta un ruido conocido como "rebotes" (ver figura 4). El origen de este ruido está en la propia naturaleza mecánica de los contactos. Para eliminarlo se utilizará un flip-flop /S/R como eliminador de rebotes, conectado como se indica en la figura 5. En el **Quartus II** el flip-flop /S/R es una primitiva llamada "***nandltch***".

Figura 4. *Rebote de la señal.*

Se pide:

- Diseñar en forma mínima utilizando flip-flops tipo D el circuito secuencial modo reloj **VALID** sensible al flanco de subida.
- Generar un proyecto con **KEY_VALID** y comprobar su funcionamiento en el simulador.
- Crear el símbolo **KEY_VALID**.
- Analizar cómo utilizar las entradas del flip-flop S\R\ para lograr una señal de reloj sin rebotes.
- Generar un proyecto en el **Quartus II** con el circuito de la figura 5 donde las entradas y salidas se conectarán a las señales del mismo nombre en la placa DE0. Analizar el circuito, simular, programar la placa y verificar el funcionamiento en forma exhaustiva utilizando la tabla de verdad de la practica 1d).

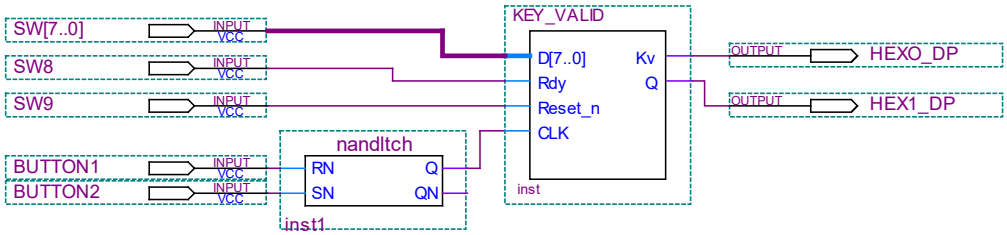

Figura 5. *Circuito de prueba para la parte b)*

Parte c)

El teclado envía los códigos de tecla en forma serial, proporcionando las señales:

- DATA para enviar los códigos de 8 bits.
- CLK_PS2 señal de sincronismo. DATA se debe leer en el flanco de bajada de esta señal.

Cuando el teclado está inactivo, ambas señales valen 1.
Para enviar un byte, el teclado primero baja DATA y luego baja CLK_PS2, generando un tren de 11 pulsos en esta última señal, los cuales son usados por DATA para transmitir en forma sincronizada 11 bits de la siguiente forma (ver figura 6):

- 1 bit de arranque en 0 (START)
- 8 bit de datos comenzando por el LSB (D[7..0]).
- 1 bit de paridad impar (PARIDAD) (no se va a chequear la paridad).
- 1 bit de parada (STOP) que siempre es 1.

El receptor debe leer cada bit en el flanco de bajada de CLK_PS2.

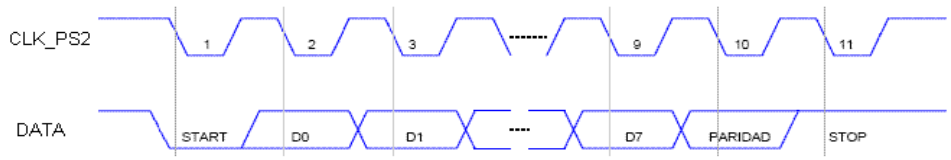

Figura 6. *Diagrama de tiempos de una transmisión PS2.*

El circuito **SaP_PS2** de la figura 7, se encarga de convertir el dato enviado por el teclado, de serie a paralelo, sin chequear la paridad, indicando que la salida **D[7..0]** es válida con un pulso a 1 de un período de CLK de duración en la señal **Rdy**. Este circuito está compuesto por **ContadorBCD**, un registro de corrimiento de 8 bits, **Ctrl_SaP** y compuertas.

ContadorBCD es el creado en la parte a). Si su entrada **RSTn** = 1, cuenta flancos de bajada de la señal CLK_PS2.

El registro de corrimiento está compuesto por 8 flip-flops. Notar que su salida cambia con cada flanco de bajada de **CLK_PS2**.

Ctrl_SaP es un circuito que se encarga de indicar en qué momento la salida del registro de corrimiento contiene un dato convertido de serie a paralelo válido. Este cuenta con dos entradas, **DATA** y **C**, y tres salidas **Rdy**, **R** y **Q[1..0].** La entrada **C** y la salida **R** son utilizadas para manejar el contador como indica la figura 7. **Ctrl_SaP** funciona de la siguiente forma:

- En estado de reposo, el teclado está inactivo con **DATA** = 1, el contador debe estar en estado de reset (**R**=0) y **Rdy** = 0.
- La transmisión comienza con un bit de arranque, baja **DATA**, debiéndose comenzar a contar con el contador (**R**=1) los flancos de **CLK_PS2**. Desde que baja **DATA** hasta que baja **CLK_PS2** hay varios períodos de reloj de **CLK**.
- Al llegar a 9 flancos (**C**=1) en **CLK_PS2**, **D[7..0]** es válido, por lo que se debe generar un pulso de 1 período CLK de duración en la salida **Rdy.** El resto del tiempo **Rdy** es 0.
- Tener en cuenta que el décimo bit corresponde al bit de paridad, que en este caso no se verifica, pero sí es necesario tenerlo en cuenta para determinar la finalización de la transmisión. Debido a ésto, la condición que debe considerarse para detectar que finalizó la transmisión es que hayan pasado al menos 10 flancos en **CLK_PS2** y **DATA** = 1, debiendo pasar nuevamente a estado de reposo.
- Luego de un reset se debe ir a estado de reposo.
- La salida **Q[1..0]** debe corresponder con las salidas Q de los flip-flops utilizados. Estas última salidas solo serán utilizadas a los efectos de simulación y pruebas.

Notas:
- Se puede saber si ya pasaron al menos 10 flancos de **CLK_PS2**, si **C** = 0 luego de haber sido 1.
- La frecuencia de CLK es mucho mayor a la de los pulsos de **CLK_PS2**.

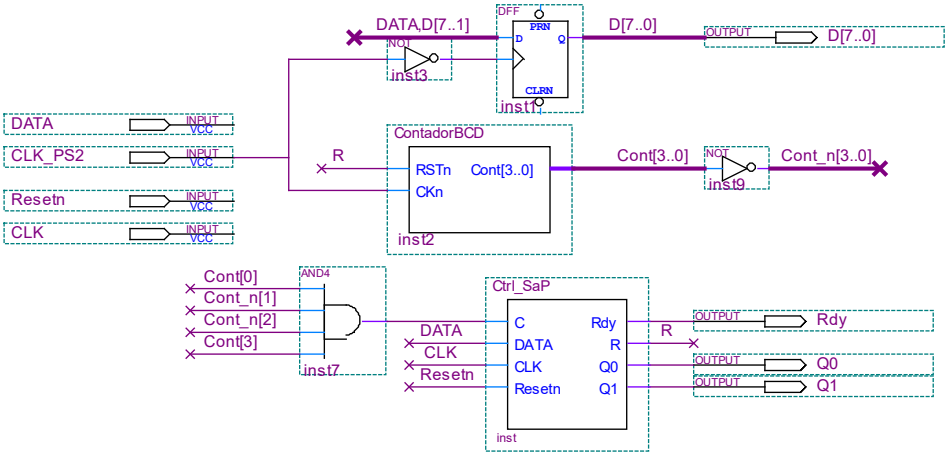

Figura 7. *Circuito del conversor **SaP_PS2***

Se pide:
- Estudiar el circuito de la figura 7.
- Diseñar de forma mínima el circuito secuencial modo reloj **Ctrl_SaP** que se describe arriba sensible al flanco de subida.
- Generar un proyecto en el Quartus II con el diseño **Ctrl_SaP** y verificar su funcionamiento en el simulador para todos los casos posibles.
- Generar un proyecto con el circuito **SaP_PS2**. Crear el símbolo correspondiente.
- Para probar el circuito en la placa, generar un proyecto en el Quartus II que implemente el circuito de la figura 8. Los divisores de frecuencia se utilizan para lograr enlentecer el reloj y apreciar lo que está sucediendo. Analizar el circuito, simular, programar la placa y verificar el funcionamiento. Se deberá probar en forma exhaustiva.

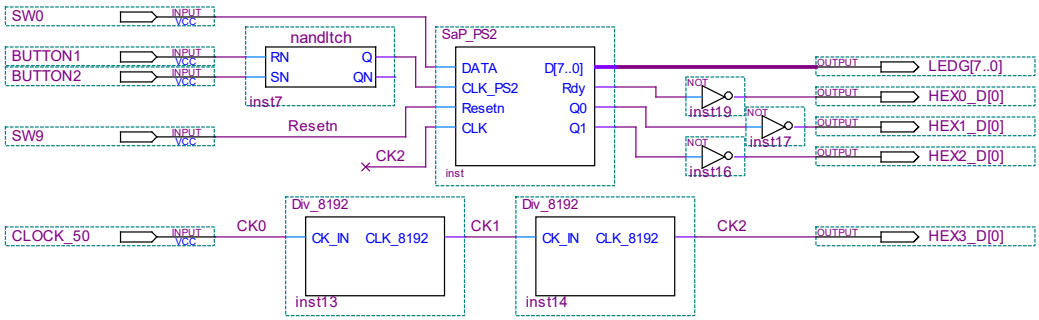

Figura 8. *Circuito de prueba para la parte c)*

INFORME

Uno de los integrantes del grupo deberá entregar en la tarea correspondiente en EVA, un **informe.pdf** utilizando la carátula disponible en EVA, que incluya:

Parte a)

- Diagrama del circuito **Div_8192** (para poder trabajar en la evaluación).
- Explicar brevemente el funcionamiento de **Div_8192**.
- Explicar brevemente el funcionamiento del **Contador**. Simulaciones.
- Explicar brevemente que es lo que hace la prueba (figura2). ¿Qué frecuencia tiene CK1?
- ¿Qué hay que modificar para que luego de oprimir **Boton2** la salida sea 5?.

Parte b)

- Diagrama de estados, tabla de estados, minimización de estados, asignaciones y mapas K del diseño **VALID**.
- Diagrama del circuito **VALID** implementado.
- Simulaciones (solo aquellos casos que se consideren de interés).
- Explicar como será utilizado el **nandltch** para evitar los rebotes.
- Breve explicación de cómo será probado el circuito de la figura 5 y qué deberá observarse en la placa DE0.

Parte c)

- Diagrama de estados, tabla de estados, minimización de estados, asignaciones y mapas K del diseño **Ctrl_SaP**.
- Diagrama del circuito **Ctrl_SaP** implementado.
- Simulaciones (solo aquellos casos que se consideren de interés).
- Breve explicación de cómo será probado el circuito de la figura 8 y qué deberá observarse en la placa DE0.

El día de la evaluación **el grupo deberá presentarse 10 minutos antes de la hora establecida** en el laboratorio de software del instituto de Ingeniería Eléctrica.
Además se deberán traer el KIT DE0-LAB y un “pen drive usb” con todos archivos de los proyectos indicados y las simulaciones realizadas, si se desea usar las pc de la facultad. Si utilizaron una laptop para hacer la tarea se recomienda traer la misma laptop para la defensa. En ambos casos probar que todos los programas funcionan antes de la práctica.

Puede ser de utilidad: Para “pegar” en el informe un circuito realizado en el editor gráfico del Quartus se debe seleccionar el circuito, “copiarlo” y realizar un “pegado especial”, indicando “mapa de bits”.