Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Eléctrica
IEE2783 – Laboratorio de Sistemas Digitales

Experiencia 2: Generador de funciones digital

2 Semanas

---

1. Objetivos

Esta experiencia busca que el alumno se familiarice con el uso del FPGA y la tarjeta de desarrollo Basys 2 mediante el diseño e implementación de un generador de señales digital y el uso de algunos de los periféricos de la tarjeta.

2. Desarrollo

Esta experiencia consiste primariamente programar en verilog [1]. De todas maneras, antes de comenzar su trabajo en el laboratorio, procure tener todos los materiales necesarios y que su mesón de laboratorio se encuentre ordenado y libre de objetos que puedan dañar los instrumentos (principalmente líquidos).

2.1. Generador de señales digital

El generador de señales consiste en un contador que indexa cíclicamente los valores almacenados en una memoria, los cuales posteriormente son convertidos al dominio analógico por un DAC de ocho bits. Los valores almacenados en la memoria corresponden a muestras de ocho bits de la señales a generar.

Su diseño debe cumplir con las siguientes especificaciones generales:

- Tres formas de onda: sinusoidal, triangular y cuadrada. Para seleccionar la señal de salida se debe utilizar un botón que, al ser presionado, cambie la forma de onda a la salida.
- Por medio de un display de siete segmentos se debe indicar con un número cuál es la forma de onda seleccionada, por ejemplo, 1: sinusoidal, 2: triangular y 3: cuadrada.
- Su programa debe encender y apagar los LEDs de la placa de acuerdo al movimiento de la señal de salida (como los LEDs de un ecualizador).
- Frecuencia variable (al menos cuatro frecuencias distintas). La selección de frecuencia debe hacerse por medio de switches (al menos dos). Las frecuencias deben ser lo suficientemente lentas como para poder ver bien cómo se mueven los LEDs.
- Switch de encendido y apagado.

[1] A estas alturas se espera que cada alumno ya cuente con el programa ISE Webpack y el programa Digilent Adept.

2.2. Detalles de implementación

A continuación se especifican detalles sobre una implementación posible del generador de señales. Para todos los efectos considere el diagrama de bloques de la Fig. 1 [2].

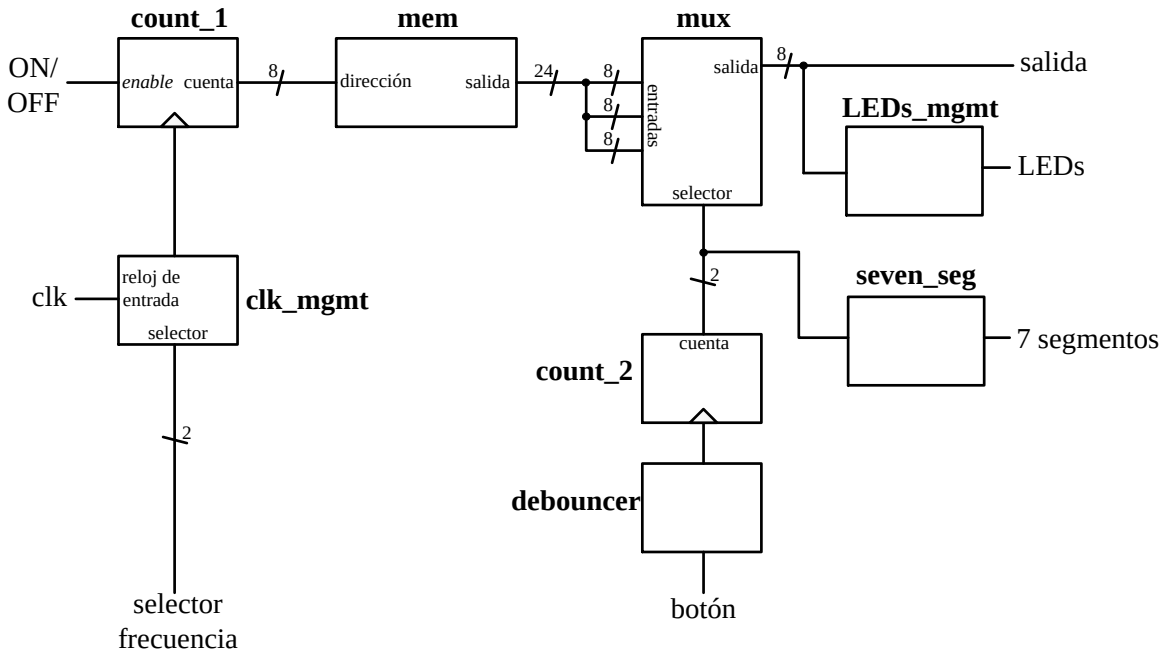

Figura 1: Diagrama de bloques del generador de señales.

count_1 Contador de ocho bits (0 a 255) con *enable* controlado por la señal de encendido y apagado. Su función es indexar las muestras almacenadas en el bloque `mem` a una frecuencia dada por la salida del bloque `clk_mgmt`.

count_2 Contador de dos bits que cuenta sólo hasta 3. Su salida, que corresponde al selector del multiplexor `mux`, debe incrementar su cuenta con los flancos de subida del botón selector de forma de onda. Además, la salida de este bloque debe ser utilizada para manipular el *display* de siete segmentos (bloque `seven_seg`).

mux Multiplexor con tres entradas de ocho bits y una salida de ocho bits.

mem Memoria de 256x24 (256 palabras de 24 bits cada una). Cada palabra de memoria contiene una muestra de las tres formas de onda: los primeros ocho bits corresponden a la señal sinusoidal, los ocho bits del medio corresponden a la onda triangular, y los últimos ocho bits corresponden a la onda cuadrada. Hay dos alternativas para implementar este bloque: mediante la creación de una memoria hecha por usted mismo con un `case`, o utilizando el *Coregen Wizard* de ISE Webpack para crear una ROM. El *Coregen Wizard* asegura que la

[2] No es obligación guiarse por el diseño aquí propuesto.

ROM sea implementada en la región de memoria del FPGA, la cual está optimizada para estos propósitos, en vez de ocupar muchos registros y lógica de propósito general.

Para generar los valores de las muestras de cada señal utilice algún programa de computación científica como Matlab, Scilab u Octave. Específicamente, genere para cada forma de onda un ciclo de 256 muestras[3] con amplitud entre 0 y $2^8 = 256$. Redondee los valores obtenidos al entero más cercano e ingréselos en su código verilog de la manera más conveniente[4].

clk_mgmt Selector de frecuencia. Este bloque consiste en varios divisores de frecuencia y un multiplexor con el cual se selecciona la frecuencia de salida. La entradas de este bloque están dadas por el reloj de la tarjeta `clk` y por los dos bits selectores de frecuencia (provenientes de los interruptores de la placa), mientras que su salida está dada por `clk`, `clk/2`, `clk/3`, `clk/4`, `clk/5`, o cualquiera sean las frecuencias escogidas para su diseño. Averigüe cómo se pueden generar frecuencias que no correspondan a `clk` dividido por un número entero.

seven_seg Binario a siete segmentos. Este bloque toma como entrada un número binario (en este caso particular, de dos bits) y genera las señales de control del *display* de siete segmentos.

LEDs_mgmt Controlador de encendido y apagado de LEDs. Para implementar este bloque puede utilizar un decodificador termómetro.

debouncer Este bloque se encarga de limpiar los rebotes de la señal generada con el *push button* de la placa.

3. Informe

En su informe de experiencia debe incluir lo siguiente:

1. Diagrama de bloques de su máquina debidamente explicado.
2. Diagrama de input-output de su diseño (use layout de la tarjeta).
3. Discusión y conclusiones (comente sobre su impresión de la experiencia y proponga mejoras).

Debe enviar su código ordenado por email a los ayudantes.

4. Evaluación

Una vez finalizada la experiencia cada grupo debe realizar una demostración de su máquina y contestar preguntas sobre su diseño e implementación.

La nota de la experiencia se distribuye como:

- Funcionamiento del circuito: 60 %.
- Informe: 30 %.
- Preguntas hechas durante la demostración: 10 %.

[3] Debe procurar que no haya saltos en la señal al repetir cíclicamente los valores de los vectores generados.

[4] En el caso que haya decidido implementar la memoria con un case, lo más conveniente es generar inmediatamente el archivo mem.v con la sintaxis apropiada. En el caso contrario, averigüe en qué formato deben estar los datos para cargar la ROM creada con el *Coregen Wizard* de ISE Webpack.