Laboratorio #5:
Verilog Conductual

Escuela de Ingeniería Eléctrica

Universidad de Costa Rica

1 Introducción

1.1 Objetivos del Laboratorio

En este laboratorio, las personas estudiantes explorarán los multiplexores y demultiplexores, circuitos combinacionales clave en el diseño de sistemas digitales. A través del desarrollo práctico, se implementarán multiplexores de distintas capacidades y se emplearán topologías de árbol para manejar señales más complejas. Además, se integrarán conceptos de Verilog conductual para modelar estos circuitos, extendiendo su uso a aplicaciones prácticas en la tarjeta de desarrollo.

A través de esta actividad, las personas estudiantes lograrán:

- Comprender el funcionamiento básico de los multiplexores y demultiplexores.
- Implementar multiplexores de mayor capacidad utilizando configuraciones en árbol.
- Diseñar circuitos digitales combinacionales utilizando Verilog conductual.
- Programar y probar el comportamiento de los circuitos diseñados en hardware mediante una FPGA.

2 Conceptos teóricos necesarios

2.1 Multiplexores (Mux)

Un multiplexor es un elemento que permite seleccionar una una de las entradas para transmitir a la salida. Esta selección se realiza mediante un conjunto de señales de selección, donde la cantidad de señales de selección es función de la cantidad de entradas del multiplexor.

En la Figura 1 se muestra el ícono que se utiliza para representar a los multiplixores, mientras que en la Tabla 1 se muestra la tabla de verdad que muestra su funcionamiento.

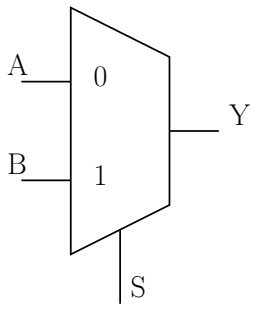

Figura 1: Ícono de un Multiplexor 2:1

S	A	B	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Tabla 1: Tabla de verdad del Multiplexor 2:1

Como se aprecia en la tabla de verdad, el multiplexor selecciona cuál entrada transmitir a su salida mediante la la señal S. Cuando $S = 0$, se tiene que $Y = A$, mientras que cuando $S = 1$, se tiene $Y = B$.

Dicha tabla se puede simplificar utilizando condiciones "no importa", las cuales se representan con una X. Observe que si $S = 0$, la salida sigue el valor de A, sin importar qué valor tenga B. Esto se representa como se muestra en la Tabla 2.

S	A	B	Y
0	0	X	0
0	1	X	1
1	X	0	0
1	X	0	0

Tabla 2: Tabla de verdad del Multiplexor 2:1 con No Importa

De la misma forma, el multiplexor 2:1 se puede extender a seleccionar entre un mayor número de entradas, para esto es necesario aumentar la cantidad de bits de selección. Por ejemplo, con dos bits de selección, se podría escoger una entrada entre cuatro opciones distintas.

2.2 Árboles de multiplexores

Los árboles de multiplexores son estructuras jerárquicas formadas por múltiples multiplexores más pequeños, organizados de manera que permiten manejar una gran cantidad de señales de entrada con un número reducido de líneas de selección. Este enfoque modular facilita la implementación de multiplexores de mayor tamaño utilizando bloques más simples y reutilizables.

Los árboles de multiplexores se utilizan cuando se necesita construir multiplexores de mayor tamaño con componentes más simples, pues es más eficiente fabricar multiplexores 2:1 o 4:1 y ensamblarlos en estructuras más grandes. Además, facilitan la implementación en FPGA y ASIC, reutilizando módulos pequeños en una estructura escalable.

Para ilustrar la idea de un árbol de multiplexores, considere un multiplexor 4:1 (resulta más sencillo de construirlo utilizando otras metodologías, pero es un caso simple para ilustrar la idea). Cuya tabla de verdad se muestra en la Tabla 3. Note que, para poder seleccionar entre cuatro entradas, es necesario incluir dos bits de selección (S1 y S0).

En la Figura 2 se muestra cómo construir un multiplexor 4:1 a partir de un árbol de multiplexores 2:1. El árbol se compone de tres muxes (M0, M1, M2), dispuestos en dos niveles distintos, con M0 y M1 en el primer nivel (controlados por S0) y M2 en el segundo nivel (controlado por S1). Note que el segundo nivel (M2) selecciona (utilizando S1) cuál de los dos nodos intermedios es el que pasará a la salida. En caso de que $S1 = 1$, se utilizará la salida de M0, caso contrario se utilizará la salida de M1. Por otra parte, el primer nivel (M0 y M1) realiza las selecciones locales, dependiendo del bit menos significativo.

Esta misma idea se puede expandir para construir árboles con más niveles, capaces de realizar la selección de aún más entradas.

S1	S0	A	B	C	D	Y
0	0	0	X	X	X	0
0	0	1	X	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1

Tabla 3: Tabla de verdad de un multiplexor 4:1

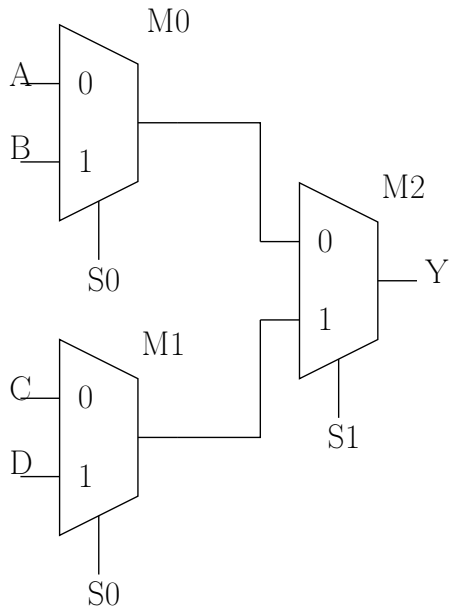

Figura 2: Multiplexor 4:1 utilizando un árbol de Multiplexores 2:1

2.3 Demultiplexores (Demux)

Un demultiplexor es un circuito combinacional que realiza la función inversa a un multiplexor. En lugar de seleccionar una de n entradas de datos para transmitir a su **única salida** (como lo hace un multiplexor), el demultiplexor transmite su **única entrada** a una de sus n salidas de datos.

En operación normal de un demultiplexor, todas las salidas son 0, excepto la que se está seleccionando a través de las señales de selección. La señal seleccionada es igual a la señal de entrada D. La Tabla 4 muestra su funcionamiento, mientras la Figura 3 muestra el ícono que se utiliza para representarlo.

S	D	Y0	Y1
0	0	0	0
0	1	1	0
1	0	0	0
1	1	0	1

Tabla 4: Tabla de verdad del Demultiplexor 2:1

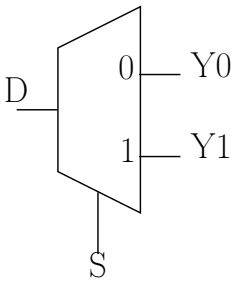

Figura 3: Ícono de un Demultiplexor 2:1

2.4 Display de 7 Segmentos

Un display de 7 segmentos es un dispositivo de salida ampliamente utilizado para representar números y ciertos caracteres alfanuméricos en sistemas digitales. Se compone de siete LEDs organizados en forma de un "8", junto con un punto decimal (dp). Cada segmento puede encenderse o apagarse de manera independiente para formar los diferentes dígitos del 0 al 9 e incluso algunas letras. Una fotografía de un display de 7 segmentos se muestra en la Figura 4.

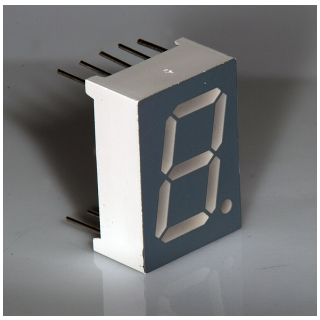

Figura 4: Fotografía de un display de 7 segmentos.

Los displays de 7 segmentos se utilizan en una amplia variedad de aplicaciones, incluyendo relojes digitales, temporizadores, cronómetros, como indicadores de medición como en voltímetros o amperímetros, para interfaces de usuario simples y económicas como en microondas, etc.

En el caso específico de la tarjeta que usaremos en este laboratorio, los segmentos están controlados como se muestra en la Figura 5. En este caso, HEX0 se refiere al display más a la derecha de la tarjeta (es decir, también existen HEX1, HEX2, etc). Para todos los casos, el bit 0 maneja el segmento de arriba, el 1 el superior del lado derecho, etc. El bit 7 maneja el punto decimal (DP).

2.5 Verilog Conductual

El diseño conductual en Verilog es un enfoque que permite describir el comportamiento lógico y funcional de un circuito digital mediante estructuras de control similares a las utilizadas en lenguajes de programación como C. A diferencia del Verilog estructural, que detalla las conexiones físicas entre los componentes del circuito, el Verilog conductual abstrae estos detalles y se enfoca en qué hace el circuito en lugar de cómo está construido.

La descripción conductual acelera el proceso de diseño de los circuitos, pues permite enfocarse en el comportamiento lógico del circuito, dejando de lado aspectos de mucho más bajo nivel de abstracción, como las conexiones entre compuertas.

Al abstraer la forma en la que se describe el circuito, facilita el modelado de sistemas complejos, como máquinas de estado (que se estudiarán más adelante en el curso). Además, es mucho más sencillo de mantener

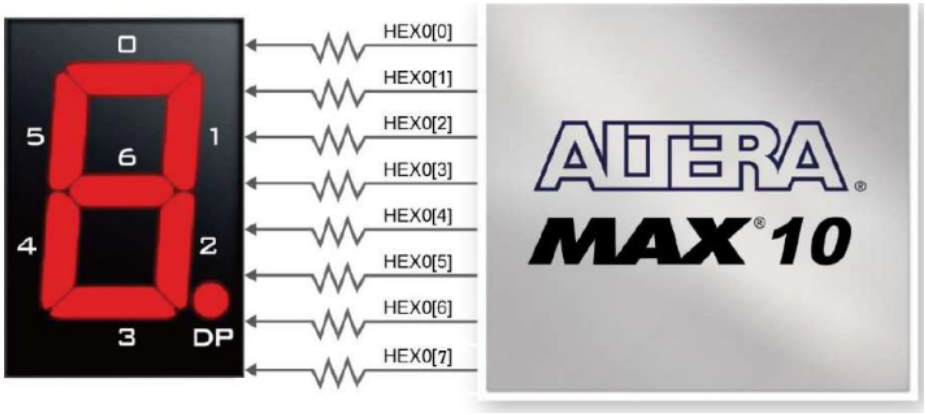

Figura 5: Displays de 7 segmentos de la tarjeta DE10-Lite.

el código e incluir cambios en la lógica, sin que sea necesaria la reconfiguración de conexiones.

Para más detalles sobre la sintaxis y semántica de Verilog conductual, refiérase al material compartido por su profesor sobre el tema, disponible en Mediación Virtual.

3 Ejemplo introductorio - Multiplexor 2:1

Recuerde que el dispositivo que se usará es el Max 10 10M50DAF484C7G.

En este ejemplo se desarrollará un módulo de Verilog que describirá el funcionamiento de un multiplexor 2:1

El multiplexor deberá seleccionar entre dos entradas distintas, estas se definirán como `A` y `B`, y deberá transmitir la señal seleccionada a la salida del multiplexor, que se definirá como `Y`. Para seleccionar una de ellas a la vez, se utilizará una señal de selección, designada como `S`.

Creación del módulo de *Verilog* del sistema digital

A continuación se presenta el módulo de Verilog que describe el funcionamiento de un multiplexor 2:1. Note que los puertos contenidos corresponden a los mencionados en la sección anterior y el uso del *if-else* para definir el valor asignado a la salida en función del valor de la entrada `S`. Utilice un editor de texto para crear este archivo.

mux_2x1.v

```
//Declaracion del modulo mux2x1
module mux_2x1 (A, B, S, Y);
  //Declaracion de puertos
  input A, B, S;
  output reg Y;

  //Cada vez que alguna entreada cambie
  always @(*) begin
    //Si S=0 la salida es A
    if (S == 0) begin
      Y = A;
    end
    //Caso contraro es B
    else begin
      Y = B;
    end
  //Termina always
  end
//Termina modulo
endmodule
```

Creación del proyecto de Quartus Prime

Cree un proyecto de Intel Quartus Prime siguiendo las indicaciones de la guía anterior. Añada el archivo `mux_2x1.v` al proyecto creado.

Asigne los puertos `A`, `B`, `S` y `Y` a los componentes `SW1`, `SW0`, `SW9` y `LEDR0`, respectivamente. Programe la tarjeta de desarrollo con el diseño y verifique que funciona como se espera.

4 Trabajo en Clase

1. Utilizando un árbol de multiplexores, implemente en Verilog un módulo nuevo que funcione como un multiplexor 4x1. Este módulo deberá contener múltiples instancias del módulo `mux_2x1` descrito en la sección 3.

   Asigne las entradas del multiplexor a los interruptores `SW3-SW0` de la tarjeta de desarrollo y la salida al LED `LEDR0`. Utilice los interruptores `SW8-SW9` para las señales de selección del multiplexor.

2. Implemente un multiplexor 4x1 utilizando únicamente Verilog Conductual.

3. Implemente un circuito que controle **un segmento particular** del *display* de siete segmentos `HEX0` (utilice los 7 segmentos y el punto decimal). Para esto debe utilizar un demultiplexor que utilice las señales de selección para elegir a cual segmento del *display* se le enviará el dato que indique el interruptor `SW0`. El segmento seleccionado deberá reflejar el estado actual del interruptor `SW0`: si está cerrado, el segmento seleccionado deberá iluminarse y si está abierto, el segmento deberá permanecer apagado. Los segmentos no seleccionados deberán permanecer apagados.

   Utilice los interruptores SW9, SW8 y SW7 para seleccionar el segmento a controlar, utilice la menor cantidad posible de interruptores.

   Por ejemplo, si el valor de los switches $\{SW9, SW8, SW7\} = \{1, 1, 0\}$, se debe controlar el segmento 6, es decir el segmento del centro.

> **Nota:** Tome en cuenta que los segmentos del *display* se encienden al suministrar un estado lógico bajo. Para más detalles, incluyendo los pines que deberá asignar, refiérase al manual de usuario de la tarjeta de desarrollo (**Terasic** o **Programa Académico de FPGAs de Intel**).

Para la DEMOSTRACIÓN deberá mostrar los diseños descritos funcionando de forma adecuada. Apenas finalice cada diseño, puede solicitar a la persona docente hacer el demo. La calificación se repartirá de la siguiente manera:

- Árbol de Multiplexores 4:1: 25%
- Multiplexor 4:1 Conductual: 25%
- Control de segmento: 50%