Subsistemas Combinacionales

Circuitos Electrónicos Digitales
Ingeniería Informática. Tecnologías Informáticas
Dpto. de Tecnología Electrónica

1 Descripción y objetivos

En esta práctica tiene como principal objetivo el trabajo con subsistemas combinacionales de propósito general y específico. Para ello se considerarán dos posibilidades:

- El uso de Circuitos integrados de escala de integración pequeña y mediana (SSI-Small Scale Integration, MSI-Medium Scale Integration).
- El uso de dispositivos programables.

Se trabajan los siguientes conceptos y competencias:

- Diseño de circuitos combinacionales haciendo uso de subsistemas.
- Implementación mediante elementos discretos.
- Uso del instrumental habitual de laboratorio de electrónica.
- Uso de lenguajes de descripción de hardware (en concreto, Verilog) para simulación de circuitos combinacionales.
- Uso de Verilog para implementar un circuito haciendo uso de un dispositivo programable.

La práctica se estructura a partir de un proyecto inicial que es explicado íntegramente por el profesor a modo de ejemplo (bien en directo o mediante un vídeo pregrabado). A partir de este proyecto de ejemplo, se proponen varios proyectos adicionales que amplían o se basan en el proyecto inicial. El alumno debe reproducir el proyecto inicial y realizar uno o varios de los proyectos adicionales.

2 Material y documentación

2.1 Software y hardware

- Instrumentación del laboratorio de Electrónica Digital.
- Circuitos integrados
- Software para simulación de circuitos electrónicos a a partir de su descripción en Verilog (EDA Playground o Icarus Verilog)
- Ordenador con entorno Vivado 2024.2 instalado.
- Placa de desarrollo Digilent Basys3.
- Archivos iniciales de diseño (kit de laboratorio).

2.2 Documentación

- Vídeos explicativos.
- Manual del instrumental básico de laboratorio de la asignatura (https://www.dte.us.es/docencia/etsii/gii-ti/cedti/laboratorio).
- Temas de susbsistemas combinacionales.
- Diagrama de pines de los circuitos integrados. Documento patillaje en https://www.dte.us.es/docencia/etsii/gii-ti/cedti/laboratorio
- Manual de Vivado https://www.dte.us.es/docencia/etsii/gii-ti/cedti/laboratorio

3 Primera parte: Simulación (trabajo previo)

Actividades a realizar:

1. Estudiar el diseño del circuito descrito en el apartado 3.1.
2. Completar la plantilla de la descripción Verilog del MUX8:1 (archivo *mux8_1.v*)
3. Estudiar la descripción Verilog del comparador diseñado mediante multiplexor (archivo *comparador_mux.v*).
4. Utilizar los archivos de testbench para simular el comportamiento de los circuitos descritos (archivos *testmux.v* y *testcomparadorMux.v*).

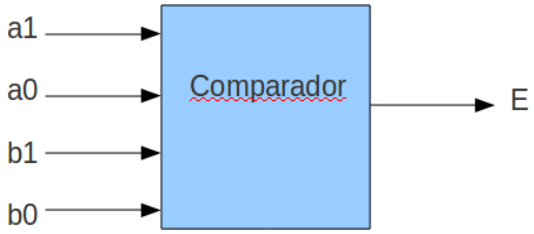
3.1 Diseño en papel de los circuitos a simular

En este apartado se explicará como diseñar el circuito que debe ser montado en laboratorio y también descrito y simulado en Verilog

3.1.1 Diseño de comparador mediante subsistemas

Descripción verbal:
Diseñe un comparador de dos números de dos bits que nos indique en su única salida si los números son iguales con un **multiplexor** de 8 canales e **inversores**.

Paso 1.
Se realiza el diagrama de bloques del circuito a diseñar y se especifican adecuadamente las entradas y salidas al mismo (*figura 1*).



ENTRADAS: 4 bits correspondientes a los dos números a comparar a_1a_0 y b_1b_0
SALIDA: informa sobre el resultado de la comparación E=0 si $a_1a_0 \neq b_1b_0$ E=1 si $a_1a_0=b_1b_0$

Figura 1: Diagrama de bloque

Paso 2.
Se representa la función de conmutación correspondiente a la descripción verbal. En nuestro caso utilizaremos un k-mapa (*tabla 1*).

Tabla 1. K-mapa

$b_1\,b_0$ \ $a_1\,a_0$	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

E

Elegimos **a_1, a_0** y **b_1** como variables de selección. Señalemos para este caso los residuos sobre el mapa

$$E(0,0,0,b_0)=\overline{b_0} \qquad E(1,0,0,b_0)=0$$
$$E(0,0,1,b_0)=0 \qquad E(1,0,1,b_0)=\overline{b_0}$$
$$E(0,1,0,b_0)=b_0 \qquad E(1,1,0,b_0)=0$$
$$E(0,1,1,b_0)=0 \qquad E(1,1,1,b_0)=b_0\text{E}$$

$b_1\,b_0$ \ $a_1\,a_0$	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

E

De esta manera sabemos los valores que hay que conectar a las entradas de datos del MUX quedando el circuito que se muestra en la *figura 2.*

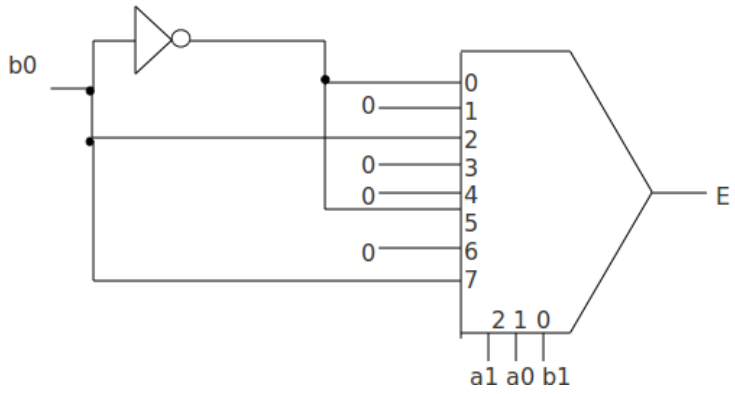


Figura 2. Diseño con subsistema de propósito general

3.2 Descripción Verilog del comparador

En este apartado se describen los pasos a seguir para realizar las simulaciones del comparador.

1. Complete la plantilla de la descripción Verilog proporcionada del multiplexor (archivo *mux8_1.v*) y pruebe su funcionamiento haciendo uso del testbench del mux (archivo *testmux.v)* .

2. Estudie la descripción Verilog del comparador diseñado mediante multiplexor (archivo *comparador_mux.v*). Observe que consta de una instancia del multiplexor al que se le han proporcionado las entradas adecuadas.

3. Realice la simulación utilizando el archivo de testbench del comparador (archivo *testcomparadorMux.v*).

3.3 Trabajo a entregar por el alumno correspondiente a la primera parte

Se deberá entregar un documento que contendrá:

- Descripciones Verilog del multiplexor y el comparador.
- Resultados obtenidos en la simulación y explicación de los mismos

4 Segunda Parte: Montaje del comparador mediante CIs

Montaje del circuito de la figura 3 utilizando un CI 74151 y un 7404.

1. Indique las conexiones realizadas sobre la la plantilla de la figura 3. **Tenga en cuenta que el multiplexor tiene una señal de habilitación activa en bajo denominada STROBE**.

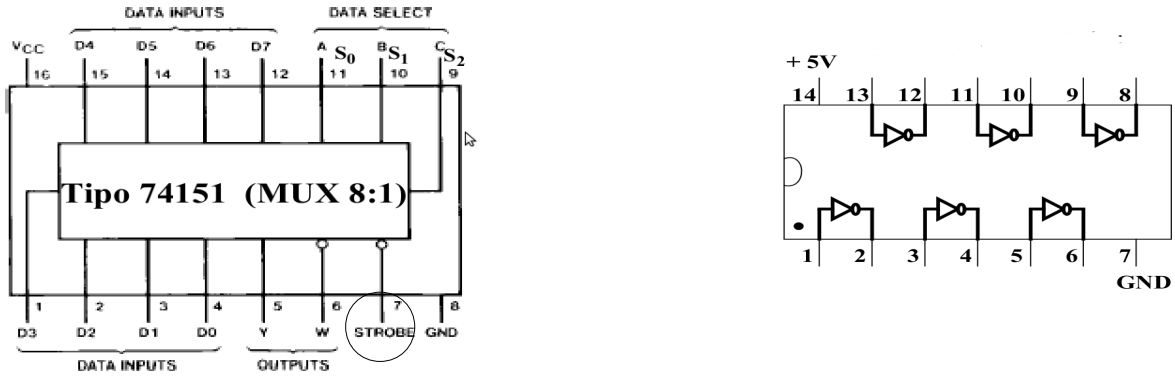
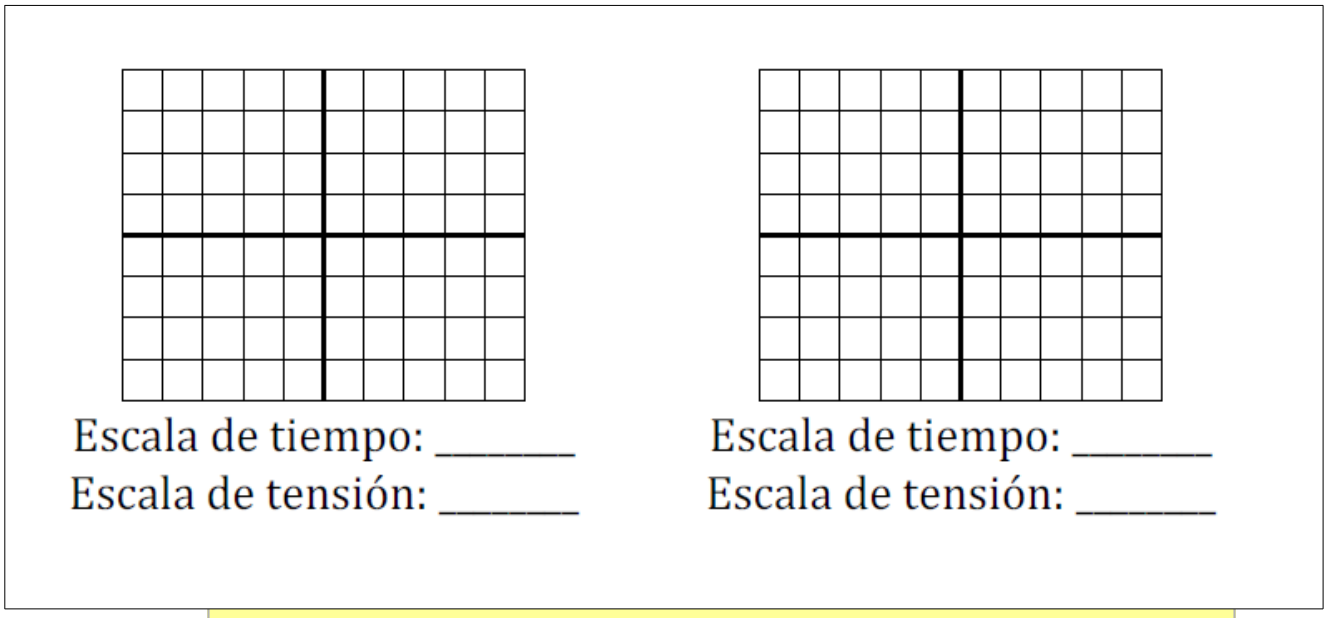


Figura 3 Plantilla para montaje

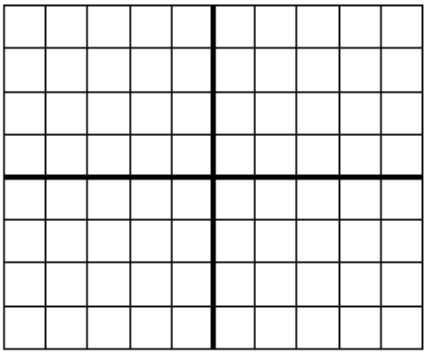
2. Compruebe el funcionamiento correcto de dicho circuito. Para ello siga el procedimiento habitual: cambie manualmente las entradas a_1, a_0 y b_1 y conecte a b_0 una señal cuadrada que varíe entre 0V y 5V.

3. Dibuje en los gráficos siguientes los casos $a_1a_0b_1 = 010$ y $a_1a_0b_1 = 101$.



☞ **Avise al profesor para que de el visto bueno al montaje una vez que haya comprobado su correcto funcionamiento.**

4. Mida el tiempo de retardo del circuito para el caso $a_1a_0b_1$= 111 y dibuje lo que observa en el osciloscopio cuando realiza dicha medida.



Escala de tiempo: _____

Escala de tensión: _____

t_{pHL} =

t_{pLH} =

☞ **Avise al profesor cuando tenga la configuración correcta para la medida del tiempo de propagación.**

5 Tercera parte: Implementación del comparador en FPGA

Para realizar esta parte se trabajará con las descripciones Verilog ya probadas en el apartado de simulación de esta práctica.

5.1 Creación de un proyecto en Xilinx Vivado

- Inicie el entorno Xilinx Vivado haciendo doble click en el icono “Vivado2024.2” que se encuentra en el escritorio.
- Cree un nuevo proyecto desde Quick Start → Create Project.
- En la siguiente ventana se solicita el tipo de proyecto. Se selecciona RTL Project y se marca “Do not specify sources at this time” y después se pulsa Next.
- El siguiente paso es especificar la FPGA (Part). Se selecciona Part y el dispositivo que tiene la placa Basys3 es (xc7a35tcpg236-1). Este se identifica utilizando los desplegables indicando los siguientes valores:
  - Category: General Purpose
  - Family: Artix-7
  - Package: cpg236
  - Speed grade: -1

  En Part: elegir xc7a35tcpg236-1

Tras ello se abre la interfaz para la gestión de diseños que ya conoce. Permite recorrer el flujo de diseño y analizar los resultados obtenidos.

5.2 Añadir archivos al proyecto

Para añadir los archivos previamente elaborados al proyecto, se selecciona Add Sources en el Project Manager de la ventana Flow Navigator o en el Menu File.

- Se elige el tipo de fuente que queremos añadir o crear:
  - Añadir o crear constrains (restricciones)
  - Añadir o crear fuentes de diseño
  - Añadir o crear fuentes de simulación (testbench)

Para incorporar el archivo donde ha descrito el módulo del multiplexor, se debe elegir Add or create design sources y se pulsa Add file. En la ventana emergente se elige el archivo con la descripción Verilog, a continuación Ok y, por último Finish.

5.3 Simulación funcional

Aunque en la primera parte de esta práctica ya haya realizado una simulación de **los circuitos (mux y comparador)**, deberá realizar una simulación en el entorno Vivado previamente a la implementación.

Para simular cada móduls con el testbench suministrado haga lo siguiente:

• Seleccione Add Sources en el Project Manager de la ventana Flow Navigator (o en el Menu File), se debe elegir en este caso Add or Create Simulation Sources y se selecciona Add File. En la ventana emergente se elige el archivo de test y se pulsa Finish. A continuación, el archivo se puede visualizar en la ventana Sources dentro de Simulation Sources.

• Para lanzar la simulación, en la ventana Flow Navigator se pulsa con el botón izquierdo sobre la opción Run Simulation y se elige “Run Behavioral Simulation”.

5.4 Añadir el fichero XDC para poder realizar la implementación

A continuación se implementará el circuito comparador

Para el comparador mapearemos las entradas *a1* y *a0* en dos switches de la placa (por ejemplo SW3 y SW2), las entradas *b1* y *b0* en otros dos switches (por ejemplo SW1 y SW0) y la salida *E* en uno de los LED (por ejemplo LD0).

- Añada el archivo XDC al proyecto (Add Sources → Add or create constraints)
- Edite el archivo XDC. El fichero suministrado es un fichero genérico para la placa Basys3 que tiene todas las líneas comentadas. Habrá que configurar 4 líneas correspondientes a las cuatro entradas y a la salida del circuito.
- Salve el archivo XDC.

5.5 Síntesis e Implementación

El proceso de síntesis e implementación culmina con la generación del conjunto de bits (bitstream) de configuración de la FPGA. Cuando estos bits sean cargados en los bloques correspondientes, la FPGA se comportará según el diseño realizado.

Para ello,

- Se selecciona Generate Bitstream en el menú de la ventana Project Manager. El programa advierte de que no se ha realizado aún la implementación, pero no hay problema porque la generación de bitstream incluye la síntesis y la implementación.
- Aparece una ventana emergente en la que se pregunta en qué directorio volcar el fichero, se mantiene el "por defecto" y se pulsa OK. En la carpeta del proyecto aparece un fichero con extensión .bit que es el bitstream necesario para programar (configurar) la FPGA.

Nota: el proceso toma un tiempo, puede ver que se está realizando la tarea ya que aparece un icono circular de color verde en la parte superior derecha. Al finalizar el proceso, en la ventana emergente, pulse Cancel.

5.5.1 Programación del diseño en la FPGA

Los pasos a seguir para programar la FPGA son:

- Se conecta la placa Basys3 al ordenador mediante el puerto USB mini.
- Se comprueba que el jumper JP1 de la placa (esquina superior derecha) está en la posición JTAG.
- Se enciende la placa (switch esquina superior izquierda de la placa).
- En Flow Navigator, se pulsa Open Hardware Manager -> Open Target -> Auto Connect
- Se acciona Program Device en Open Hardware Manager se abre una ventana y en ella se selecciona la FPGA usada.
- Tras ello, en la ventana emergente se selecciona el fichero .bit generado previamente y se acciona Program.

A partir de ese momento, la FPGA está programada según el diseño realizado.

Ahora pruebe su diseño introduciendo distintos valores en los datos y verificando que la salida E se activa solo en caso de igualdad.

6 Cuarta parte: proyectos adicionales

En esta parte el alumno deberá realizar la simulación e implementación en la FPGA de un circuito comparador mediante una descripción directa y mucho más sencilla que la desarrollada en la práctica.
Para ello sustituya el cuerpo de la descripción del archivo comparador_mux por la nueva descripción (no modifique el nombre de entradas y salidas para poder simular mediante el mismo testbench). Si utiliza un procedimiento recuerde que la salida debe ser declara como reg.