



Universidad Rey Juan Carlos

Escuela Técnica Superior de Ingeniería
Informática

Titulación: Grado en Ingeniería Informática

Asignatura: Fundamentos de Computadores

Práctica 1:

Diseño VHDL de sistemas combinacionales

Profesor: Isaac Martín de Diego

Importante:

En las siguientes páginas se presenta el enunciado de la práctica 1. Es imprescindible utilizar EXACTAMENTE los nombres que vienen indicados para las distintas entidades, arquitecturas, puertos de entrada y salida, etc. El no cumplimiento de esta premisa impedirá la correcta evaluación de la práctica, y quedará calificada como SUSPENSA.

Material a entregar como resultado de la práctica:

- Se debe entregar un archivo comprimido (rar, zip, . . .) que contenga dentro una carpeta de nombre “PrimerApellido_NombreAlumno_Prac1” con el siguiente material:
 - o Todos los archivos VHDL escritos durante la realización de los distintos apartados de la práctica.
 - o Una memoria escrita (Word, OpenOffice, Latex, PDF, . . .) en la que se expliquen todos los pasos seguidos para la realización de la práctica: tabla de verdad, dibujos de los circuitos, cálculos teóricos, etc, así como una captura de pantalla con los resultados comentados de la simulación que se pide en el punto f). Los dibujos de los circuitos se pueden hacer a mano y escanearlos, o se puede utilizar alguna herramienta como Xilinx o Microsoft Visio disponibles en las aulas de libre acceso.

Las prácticas se realizarán de forma INDIVIDUAL. La entrega de material no original supondrá una calificación de SUSPENSO en la práctica.

La entrega se realizará ÚNICAMENTE a través del campus virtual, con fecha límite igual al día del examen final de la asignatura (consultar con el profesor).

NO se aceptarán prácticas entregadas en mano, ni por correo electrónico.

Enunciado:

Se desea diseñar un circuito combinacional con 2 entradas de 2 bits cada una X_1X_0 e Y_1Y_0 , y una única salida Z que toma valor '1' si el número X_1X_0 es menor que el número Y_1Y_0 .

El circuito se diseñará de diversas maneras, y para cada una de ellas se hará una descripción en VHDL. Cada una de las descripciones en VHDL será una arquitectura asociada a la siguiente entidad:

```
entity practica_1 is
port (x, y: in std_logic_vector(1 downto 0),
      z: out std_logic);
end practica_1;
```

Se pide hacer lo siguiente:

- Obtener la expresión más simplificada de Z en forma de suma de productos. Describir en VHDL la expresión obtenida utilizando una asignación concurrente. El nombre de esta arquitectura debe ser **concurrente_sdp**.
- Obtener la expresión más simplificada de Z en forma de producto de sumas. Describir en VHDL la expresión obtenida utilizando una asignación concurrente. El nombre de esta arquitectura debe ser **concurrente_pds**.
- Implementar Z utilizando sólo puertas NOR (y opcionalmente inversores). Describir en VHDL el circuito obtenido utilizando las puertas lógicas proporcionadas en el archivo *puertas_basicas.vhd*. El nombre de esta arquitectura debe ser **estructural_nor**.
- Implementar Z utilizando un decodificador de 3 a 8 y puertas lógicas auxiliares. Describir en VHDL el circuito obtenido utilizando el decodificador proporcionado en el archivo *componentes.vhd* y las asignaciones concurrentes o puertas lógicas que se considere oportuno. El nombre de esta arquitectura debe ser **estructural_deco**.
- Implementar Z utilizando un multiplexor de 8 a 1 y puertas lógicas auxiliares. Describir en VHDL el circuito obtenido utilizando el multiplexor proporcionado en el archivo *componentes.vhd* y las asignaciones concurrentes o puertas lógicas que se considere oportuno. El nombre de esta arquitectura debe ser **estructural_mux**.
- Utilizando el test-bench que se proporciona en el archivo *test_bench_practica1.vhd* simular simultáneamente los distintos circuitos de los apartados a) a e). Si no se ha realizado alguno de los apartados anteriores se debe comentar la línea correspondiente en el archivo *test_bench_practica1.vhd* para que no de error. Es importante que en los apartados anteriores se respeten los nombres propuestos para las arquitecturas con el fin de que el test-bench no de errores.