Grado en Ingeniería de Computadores

Tecnología de Computadores

Práctica 2

%

Universidad Rey Juan Carlos

1. Objetivo

El alumno será capaz de realizar el diseño y la simulación de un circuito secuencial complejo utilizando Vivado. Además habrá adquirido los conocimientos de VHDL imprescindibles para poder diseñar circuitos secuenciales, detectar y corregir errores y simular circuitos utilizando testbenches.

2. Condiciones de entrega

Cómo.

- Todos los ficheros que se soliciten deben subirse agrupados en un fichero comprimido.
- El fichero debe tener el siguiente nombre:

  TC_grupo_Apellido1_Apellido2.zip

  Por ejemplo, un equipo del grupo 2, formado por Fran García y Sergio Hernández, debería tener el nombre *TC_2_Garcia_Hernandez.zip.*
- Si dentro falta alguno de los ficheros solicitados, se considerará como una práctica incompleta y por tanto suspensa.

Cuándo.

- La fecha límite se indica en la actividad habilitada en el Aula Virtual de la asignatura.

Qué.

- Memoria de la práctica, que consiste en:
  - un documento en PDF (así se puede leer en cualquier SO sin cambios de formato),
  - con una portada que incluya el nombre y DNI de los alumnos que forman el grupo,
  - y las respuestas a las preguntas de esta práctica.
- Ficheros de diseño y testbenches (.vhd) e imágenes con las ondas resultantes de la simulación (.bmp, .gif o .png).

Prohibiciones.

- NO se admiten entregas por medios que no sea la actividad habilitada en el Aula Virtual.
- NO se admiten entregas fuera del plazo marcado.
- Prácticas copiadas implican nota de 0 para el copiado y para el copiador.
- Las prácticas tienen que ser originales de los miembros del grupo. Si otra persona realiza la práctica se desencadenarán los procedimientos sancionadores correspondientes.

3. Lista de comprobación antes de que termine el plazo

Esta lista te sirve para repasar que cumples con todos los requisitos formales de la práctica.

- ¿Te has apuntado en un grupo de laboratorio a través de la actividad del aula virtual?
- ¿Has hecho una portada para la memoria como se indica arriba?
- ¿Has respondido a las preguntas formuladas?
- ¿Has metido todo lo necesario para evaluar la práctica en el fichero comprimido?
- ¿Tienen todos esos archivos el formato solicitado?
- ¿Has nombrado el fichero comprimido con el patrón indicado arriba?
- ¿Has subido el fichero comprimido?

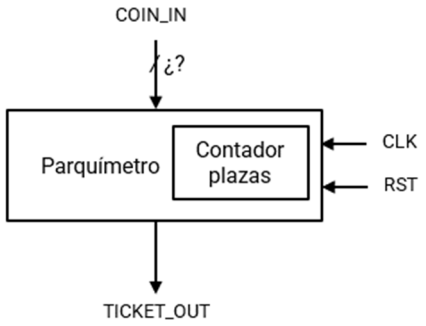
Si quieres, basta con que subas la práctica. No importa que se quede como borrador porque de todos modos queda la fecha. La ventaja es que, si te has equivocado en algo, estando como borrador puedes subir tantas veces como quieras el fichero.

4. Enunciado

Diseñar un sistema secuencial que realice el control de un parquímetro. Dicho sistema imprime un ticket válido para aparcar a un precio de 1€ siempre que haya espacios libres en el aparcamiento.

El parquímetro responde al siguiente comportamiento:

- Solo admite monedas de 1€.
- Devuelve un ticket solo si hay huecos libres.
- El aparcamiento dispone de 4 plazas.
- Por simplicidad, se asume que cuando se resetea el parquímetro todas las plazas de aparcamiento quedan libres.
- Tampoco se tendrá en cuenta que se puedan liberar plazas por otros medios diferentes al reseteo.



Para diseñar él parquímetro se recomienda utilizar dos máquinas de estados: (1) una que se encargue de leer monedas e imprimir el ticket si hay espacio y (2) otra que se encargue de contabilizar las plazas ocupadas. Asimismo, se recomienda realizar un análisis previo del parquímetro y su comportamiento esperado en cada caso.

Un posible diagrama de la barrera de acceso se presenta en la imagen a modo de referencia. Las señales del diagrama se explican a continuación:

- CLK: señal de reloj para sincronizar el funcionamiento de la máquina
- RST: señal de reinicio
- COIN_IN: señal que indica si se ha introducido una moneda
- TICKET_OUT: señal que activa la impresión del ticket.
- FULL: señal interna que indica que se han ocupado todas las plazas.

Requisitos de diseño:

A continuación, se describen los requisitos para implementar el parquímetro y su contador de 2 formas distintas:

1. Realizar el diagrama de estados y la tabla de transición de estados de las dos máquinas del sistema.
2. Obtener la ecuación de la salida de la máquina de estados del contador. Crear con ella una entidad VHDL denominada **TC23_full** que implemente dicha ecuación.
3. Implementar el contador utilizando **flipflops de tipo D**, para ello, obtenga primero sus ecuaciones.
4. Crear una entidad VHDL denominada **TC23_ffD**, que implemente un flipflop tipo D.
5. Crear una entidad VHDL denominada **TC23_transicion_D**, que implemente las ecuaciones de transición de la máquina de estados del contador.
6. Crear una entidad VHDL denominada **TC23_FSM_D_count**, que implemente la máquina de estados del contador utilizando los ficheros VHDL anteriores.
7. Crear una entidad **TC23_FSM_JK_count**, que implemente la misma máquina de estados del contador, pero utilizando **flipflops de tipo J-K**. Para ello, obtenga primero las ecuaciones de excitación de dichos flipflops e implemente los códigos VHDL adicionales que considere necesarios.
8. Crear una entidad **TC23_FSM_D**, que implemente la máquina de estados del parquímetro utilizando únicamente información del diagrama de estados (sin utilizar ecuaciones) y la entidad **TC23_FSM_D_count**.
9. Crear una entidad **TC23_FSM_JK**, que implemente la máquina de estados del parquímetro utilizando únicamente información del diagrama de estados (sin utilizar ecuaciones) y la entidad **TC23_FSM_JK_count**.

Requisitos de simulación:

Crear un único *test bench* denominado **TC23_testbench** para simular las 2 implementaciones del parquímetro simultáneamente. Dicho *test bench* deberá simular al menos dos ciclos completos de uso del parquímetro, que comiencen con el aparcamiento vacío, finalicen cuando el aparcamiento esté lleno y que compruebe el caso en que ya no se pueden imprimir más tickets por falta de espacio.

La simulación deberá mostrar una señal CLK, RST y COIN_IN común para las 2 implementaciones, las señales TICKET_OUT, FULL, estado actual y estado siguiente de cada implementación.

5. Memoria de la práctica

En la memoria hay que presentar:

1. El diagrama de estados del parquímetro.
2. Las tablas y mapas de Karnaugh utilizados para poder obtener las ecuaciones de las salidas y de los flipflops D y J-K, así como las ecuaciones finales obtenidas.
3. Responder a las siguientes preguntas:
    a. Una vez simuladas las 2 implementaciones (flipflop D o J-K), ¿se comportan todas igual? ¿Por qué?
    b. ¿Qué aspectos del diseño final variarían si se implementara la máquina de estados utilizando el tipo de máquina contrario al que ha utilizado (Moore o Mealy)?
    c. **Explique** brevemente cómo implementaría un mecanismo que controle el vaciado del aparcamiento.

6. Tiempo estimado

Esta estimación presupone que se conocen los fundamentos teóricos de la asignatura.

Tarea a realizar	Tiempo estimado
1. Pensar y bocetar el diagrama de estados en papel. Obtener las ecuaciones de excitación y de salida.	1 hora
2. Implementar en VHDL el contador utilizando flipflops tipo D	2 horas
3. Implementar en VHDL el contador utilizando flipflops tipo J-K	2 horas
4. Implementar en VHDL el parquímetro a partir de su diagrama	1 hora
5. Escribir el código VHDL del *test bench*	30 minutos
6. Ejecutar la simulación del *test bench*	30 minutos
7. Escribir la memoria y responder a las preguntas propuestas	4 horas

..... **Total: Máximo de 11 horas de trabajo para una sola persona.**