TA147

TALLER DE SISTEMAS DIGITALES

Implementación de Sistema Secuencial en VHDL

Primer Trabajo Práctico

Alumno
Klöckner, Martin Javier

Correo electrónico
mklockner@fi.uba.ar

Índice

Índice de figuras

En el presente trabajo se realiza el diseño, simulación y síntesis de un sistema digital destinado al control de dos semaforos en una intersección. Se emplea el lenguaje de descripción de hardware VHDL para modelar el sistema y la herramienta de linea de comandos GHDL para la compilación y simulación del diseño. Por ultimo se utiliza Vivado para realizar una síntesis sobre el dispositivo xc7a15tftg256-1.

1. Introducción

El control de tráfico en intersecciones es un problema clásico en sistemas digitales. La implementación mediante hardware programable permite obtener soluciones eficientes, determinísticas y además flexibles.

El objetivo del trabajo, entonces, es diseñar e implementar un sistema digital que controle el estado de dos semaforos en la intersección de dos calles.

2. Diseño

Para la implementación del control de los dos semáforos se modela al sistema como una maquina de estados finita compuesta de seis estados cíclicos, en la que cada estado representa una configuración de luces. Se considera que el estado de cada semáforo es, ordenados en orden secuencial: solo luz roja encendida, luz roja y amarilla encendida al mismo tiempo, solo luz verde encendida y solo luz amarilla encendida.

El diagrama de estados de la maquina de estados diseñada se muestra en la figura 1.

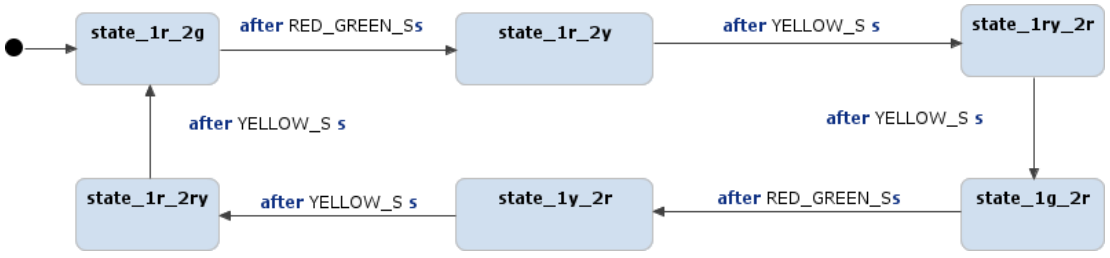

Figura 1: Diagrama de estados del control de semaforos

Donde los estados de la maquina de estados representan la configuración de luces como se muestra a continuación:

- state_1r_2g: primer semáforo en rojo, segundo en verde.
- state_1r_2y: primer semáforo en rojo, segundo en amarillo.
- state_1ry_2r: primer semáforo en rojo y amarillo, segundo en rojo.
- state_1g_2r: primer semáforo en verde, segundo en rojo.
- state_1y_2r: primer semáforo en amarillo, segundo en rojo.
- state_1r_2ry: primer semáforo en rojo, segundo en rojo y amarillo.

A cada estado del semanforo se asigna un valor de tiempo en segundos, para los estados solo rojo o solo verde se asigna 30 segundos y para los estados que involucran amarillo se asignan 2 segundos.

Dado que el dispositivo programable en el cual se sintetiza el diseño solo dispone de una señal de reloj de frecuencia 50 MHz, se implementa un modulo de reloj de tiempo real (Real Time Clock, RTC) que permita determinar la cantidad de segundos reales. Luego este modulo RTC se combina con un modulo que cuenta la cantidad de segundos en cada estado y genera una señal bandera (flag) que indica que se debe cambiar de estado.

3. Implementación

Para la implementación del control de los dos semáforos, se divide la lógica en distintos submódulos que interactúan entre sí. Y se integran estos submódulos en un módulo principal denominado `intersection_ctrl`.

El diseño se compone de las siguientes partes fundamentales, modeladas mediante el lenguaje de descripción de hardware VHDL: `semaphore_pkg`, `rtc`, `cnt`, `semaphore` y `intersection_ctrl`.

3.1. semaphore_pkg

En este paquete se definen los tipos de datos utilizados en el sistema. En particular, se define el tipo enumerativo `state_t`, el cual representa los seis estados secuenciales y cíclicos de la máquina de estados.

3.2. rtc

Este módulo se utiliza para contar la cantidad de segundos reales, activando la bandera `seg_flag` por un ciclo de reloj en cada segundo. Para determinar la cantidad de segundos se utiliza un contador que incrementa su valor en cada ciclo de reloj hasta alcanzar el valor de `MAX_CNT`.

3.3. cnt

Este modulo es el responsable de medir la duración de cada estado incrementando un contador interno únicamente cuando es habilitado por la señal `seg_flag` que proviene del modulo RTC. El tiempo objetivo depende del estado actual: en los estados con amarillo se utiliza una constante de 3 segundos (`YELLOW_S`) y para los estados en verde o rojo se utiliza una constante de 30 segundos (`RED_GREEN_S`). Al finalizar la cuenta se activa la señal `state_flag` para solicitar un cambio de estado.

3.4. semaphore

Este módulo implementa la lógica de control central, es decir, la maquina de estados. Esta maquina de estados transiciona secuencialmente a través de los seis estados únicamente cuando recibe la señal `state_flag`. Además, decodifica el estado actual para asignar los niveles lógicos a las salidas físicas, controlando las luces roja, amarilla y verde de ambos semáforos.

3.5. intersection_ctrl

Este es el modulo principal o de nivel superior (*top-level*) en el cual este se instancian todos los submódulos, asignando las señales de entrada y salida de cada modulo.. A través de este bloque se propaga el parámetro genérico `MAX_CNT` y se gestionan las señales internas `seg_flag` y `state_flag`, las cuales permiten la comunicación entre los módulos `rtc` y `cnt`; y la lógica de control final de las salidas físicas.

4. Simulación

Para realizar la simulación se creó un nuevo módulo `intersection_ctrl_tb` en el cual se encapsula el sistema diseñado, esto de manera de proporcionar señales ficticias que actúen como entradas y salidas. Para evitar tener que simular por mas de 30 segundos, se cambio el parámetro `MAX_CNT` de 50e6 (1 segundo con frecuencia de reloj 50 MHz) por 10, de esta forma se activa la bandera de que ha pasado un segundo cada 5 ciclos de reloj.

Como herramientas de simulación se utilizó *GHDL* y *make* para organizar el proceso de análisis, elaboración y simulación del sistema, luego para la síntesis se utilizó Vivado. Vivado también proporciona una opción de simulación utilizando los mismos módulos, por lo que también se hicieron pruebas allí para verificar el funcionamiento del diseño. Para la visualización de las señales por fuera de Vivado, resultantes de la simulaion en *GHDL*, se utilizó *surfer*, un programa similar a *GTKWave*. Una vista de las formas de onda resultantes de la simulación se muestra en la figura 2 a continuación, en la cual 1 segundo de tiempo real equivale a 0.1 us en las vistas de las formas de onda.

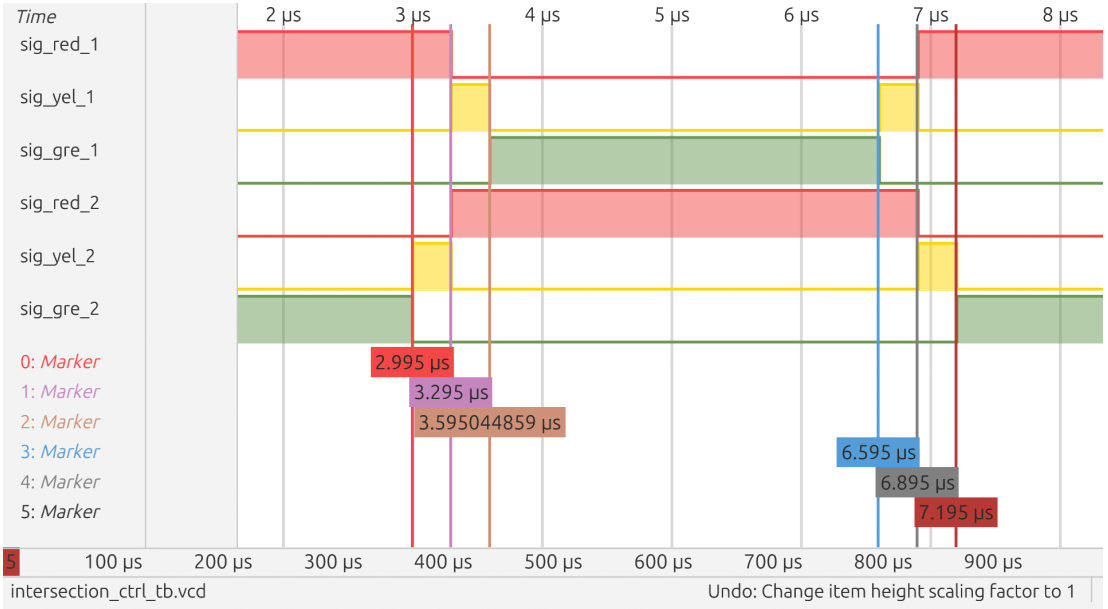

Figura 2: Resultado de las señales simuladas

5. Síntesis

Utilizando Vivado se realizo una síntesis del sistema sobre un dispositivo FPGA modelo xc7a15tftg256-1.

Luego de realizar la síntesis del sistema en Vivado para el dispositivo FPGA xc7a15tftg256-1, se validó la integridad estructural del diseño. Principalmente, se corroboró que las lineas de `clk` y `rst` fueran distribuidas de manera global y uniforme a todos los bloques lógicos (`rtc`, `cnt` y `semaphore`). Esta arquitectura garantiza un comportamiento puramente sincrónico en toda el sistema, asegurando que el sistema opere correctamente bajo restricciones temporales del hardware y además que este libre de bucles combinacionales.

6. Conclusión

En este trabajo se logró implementar un controlador de dos semáforos para la intersección de dos calles utilizando VHDL y validando el diseño mediante simulación en GHDL y en Vivado. Por último se realizo una síntesis sobre el dispositivo xc7a15tftg256-1. Con respecto a las maquinas de estados finitos, se concluye que resultan adecuadas y muy útiles para este tipo de aplicaciones.