UNIVERSIDAD DE COSTA RICA
Escuela de Ingeniería Eléctrica
IE0523 – Circuitos Digitales II

Tarea 7

Yeison Rodríguez Pacheco, B56074

08/05/19

Resumen

En esta tarea se utilizarán for loop y genvar para generar múltiples instancias de algunos circuitos. También se creará una máquina de estados que analiza paquetes entrantes.

1. Contabilización del tiempo

Tabla 1: Contabilización del tiempo

Sesiones	Sesión 1	Sesión 2	Sesión 3
Búsqueda y estudio de información	1h	10 min	0 min
Mejora del Makefile	0min	0 min	0 min
Programación del código	3h	3h	0h
Ejecución y pruebas	0min	35 min	40 min
Reporte	0 min	0 min	1.5h

2. Descripción arquitectónica del circuito

Este circuito consiste en una máquina de estados finitos, que tiene un bus de datos de entrada de un tamaño parametrizable, así como tamaño para cada palabra del bus. El circuito se puede dividir en tres etapas, la primera consiste en un circuito que invierte el orden de las palabras del bus de entrada, para esto se utilizó un genvar y un for, con el fin de que todo estuviese parametrizado.

La segunda parte del circuito consiste en la salida de control que tiene como tamaño la cantidad de palabras del bus de datos de entrada, y para el cálculo de esta salida se le debe hacer la operación bitwise OR a cada una de las palabras del bus, para esto también se utilizó un genvar y for.

Por último se tiene la máquina de estados finitos, que consiste en un seleccionador de paquetes, los paquetes tienen que cumplir que la palabra más significativa del bus este conformada solo por 1s, además de que el bit menos significativo de la entrada debe ir aumentando como un contador, si esto no se cumple se presentan errores.

3. Plan de pruebas

Las pruebas consisten en probar el funcionamiento del circuito que calcula el bitwise OR de cada una de las palabras, así como verificar que se invierta la entrada en la salida data_out, y verificar el comportamiento correcto de la máquina de estados, asegurando que se comporte como se espera.

También se probará el funcionamiento del circuito variando los parámetros de entrada, para demostrar su correcto funcionamiento.

4. Instrucciones de utilización de la simulación

Para correr la prueba con los parámetros bus_size = 16, word_size = 4 simplemente se debe ejecutar el comando

```
make
```

Inmediatamente se presentarán las simulaciones en GTKwave y se podrá acceder a las señales del modelo estructural y conductual.

Para correr la prueba con los parámetros bus_size = 20, word_size = 5 simplemente se debe ejecutar el comando

```
make prueba1
```

En la tabla 2 se dejará una lista detallada de comandos del makefile, muchos de estos solo se programaron para la creación del laboratorio, y no para las pruebas en sí.

Tabla 2: Comandos Makefile

Comandos del Makefile para pruebas (usuario)	
make	Compila y ejecuta la prueba estandar, también abre el GTKwave con las simulaciones
make gtk	Ejecuta nuevamente el gtkWave, por si necesita abrirlo de nuevo sin compilar
make prueba1	Ejecuta el archivo prueba2
make pdf	Abre el pdf del informe
Comandos de Makefile para facilitar generación de código (diseñador)	
make generar_archivos	Crea los archivos conductual, probador y banco de prueba según variables de los nombres deseados especificadas en el Makefile (Comando parametrizado)
make generar_archivo_yosys	Genera el archivo con los comandos de Yosys con respecto a los nombres elegidos por el diseñador en las variables del Makefile (Comando parametrizado)
make llenarBancoPrueba	Este comando hace los include e instancias necesarias para que el banco de pruebas funcione correctamente. (Comando parametrizado)
make ejecutar_yosys	Este comando ejecuta el archivo .ys y además se encarga de cambiar el nombre del módulo generado por la síntesis, y hace automaticamente el include del archivo que contiene la librería de compuertas a utilizar (Comando parametrizado)

5. Discusión y Análisis de resultados

5.1. Máquina de estads finitos

En la figura 1 vemos la respuesta estructural y conductual de la máquina de estados, a continuación se procederá a explicar cada una de las señales.

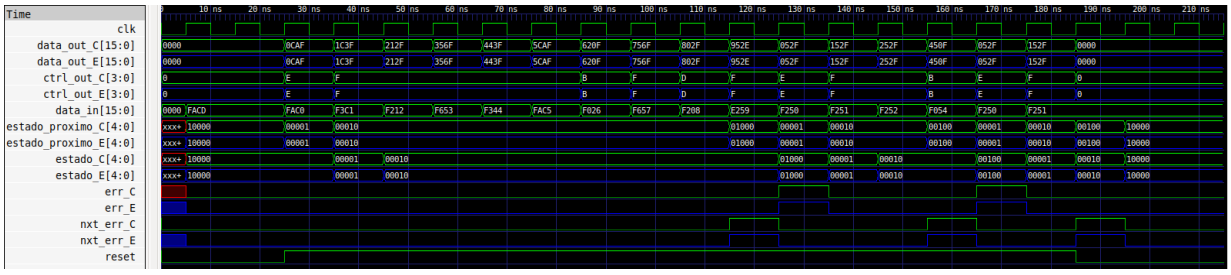

Figura 1: Respuesta conductual y estructural del circuito con bus_size = 16, word_size = 4. Hecho en GTKwave

Primero observamos las señales data_out, esta señal debe ser el inverso de data_in (inversas las palabras, que en este caso son de 4 bits), por lo que si un dígito hexadecimal está en la posición menos significativa de la entrada , debe ir a la posición más significativa de la salida, y esto se repite para cada dato. Vemos que tanto la respuesta conductual como estructural de estas señales se comporta como debería en todos los casos.

Ahora bien la señal contr_out es la representa el cálculo del bitwise OR de los bits de cada una de las palabras, por lo que tiene un tamaño de bus_size / word_size . Notamos que cuando todos los bits de una palabra son 0s como lo es el caso de 0xF026 la salida de control es B (1011), por lo que se demuestra el correcto funcionamiento tanto del modelo conductual como estructural para esta señal.

Por último se tiene el comportamiento de la máquina de estados, los estados son los siguientes:

- 00001 FIRST_PKT :Primer paquete
- 00010 REG_PKT :Paquete regular
- 00100 SEQ_ERR : Error en la secuencia
- 01000 F_ERR : Error en que no están todos los 1s de la palabra más significativa
- 10000 RESET : Estado de reset

Vemos que la máquina comienza con el estado de reset y en cuanto se levanta esta señal, próximo_estado cambia al estado FIRST_PKT, en el siguiente ciclo como el siguiente paquete cumple con todo lo especificado se pasa al estado REG_PKT, esto se repite hasta que se llega a la entrada 0xE259 que no cumple con que todos los 1s de la palabra más significativa sean 1s, por lo que se pasa al estado F_ERR, el siguiente paquete es válido por lo que se vuelve al estado FIRST_PKT y el siguiente paquete también, así que se pasa a REG_PKT, este estado perdura hasta que la entrada 0xF054 no cumple con la secuencia numérica, por lo que se pasa al estado SEQ_ERR. El siguiente paquete es válido por lo que se pasa a FIRST_PKT y sigue la secuencia.

Vemos que la máquina de estados se comporta como debería, tanto su respuesta estructural como conductuale.

En la figura 2 vemos la respuesta estructural y conductual de la máquina de estados con otros parámetros.

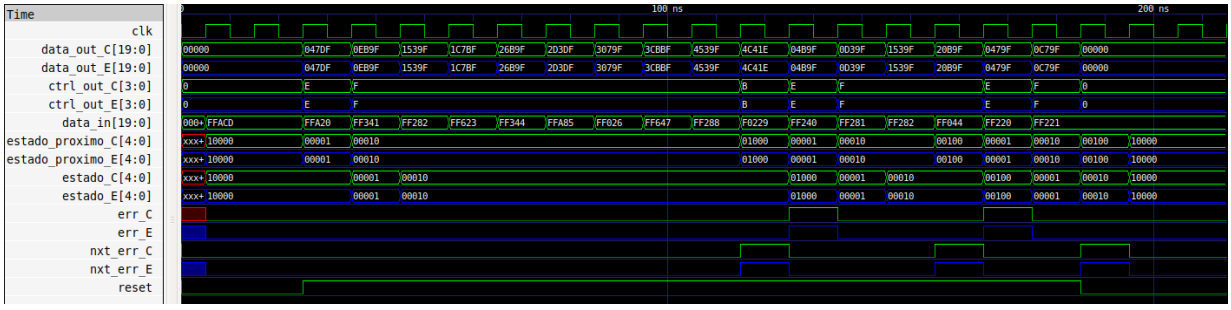

Figura 2: Respuesta conductual y estructural del circuito con bus_size = 20, word_size = 5. Hecho en GTKwave

Debido que ahora las palabras son de 5 bits se debe tener cuidado al revisar los resultados, pero podemos observar que ocurre el mismo comportamiento con los paquetes, si los 5 bits de la palabra

más significativa son 1 y la palabra menos significativa de los paquetes aumenta conforme el contador, todos van a ser correctos, notamos en cuanto se presenta un error (entrada 0xF0229) que se pasa al estado F_ERR, ya que no se cumple con que los bits sean 1. Ahora en la entrada 0xFF044 vemos que se da un error de secuencia por lo que se pasa al estado SEQ_ERR.

6. Conclusiones

- Fue posible implementar el genvar generate con un for loop para poder parametrizar diseños conductuales, lo cual es de gran utilidad.
- Se pudo crear una máquina de estados finitos que regula la entrada de paquetes de datos.
- Se logró realizar la síntesis correcta de for loops y circuitos conductuales que contenían genvar