Universidad de Costa Rica | Facultad de Ingeniería | Escuela de Ingeniería Eléctrica

Tarea 5: Descripción estructural de circuito digital generada por Yosys

Roberto Sánchez Cárdenas - B77059

San José, 10 de septiembre de 2020
Digitales II

Índice

1. Introducción

Hasta el momento de creación de esta tarea se ha venido trabajando en la síntesis manual de circuitos digitales en Verilog, sin embargo, en proyectos a gran escala es común utilizar herramientas que realizan una descripción estructural automática a partir de una descripción conductual. Puesto que la descripción de comportamiento es más sencilla de programar, se ve reducido el tiempo empleado en el trabajo.

En este caso se va a usar Yosys para tomar una descripción conductual y pasarla a estructural, esto utilizando las compuertas de una librería pre-armada y disponible en el git de Yosys.

2. Procedimiento

A partir del código creado en la Tarea 2, se va a realizar la descripción estructural del circuito mostrado en la figura 1. Esto nos es muy útil puesto que se pueden comparar los resultados obtenidos en ambas tareas y comprobar el correcto funcionamiento.

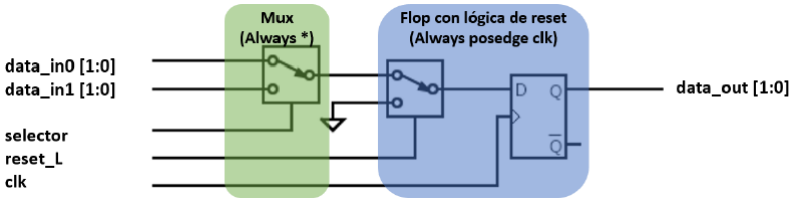

Figura 1: Circuito deseado

2.1. Plan de pruebas

Como se mencionó previamente, se va a reciclar el código de las tareas previas. En este caso se van a usar los pulsos mostrados en la siguiente figura para probar los resultados. Se espera que los resultados sean idénticos y que en el caso del que tiene delay, solo debe haber un desfase.

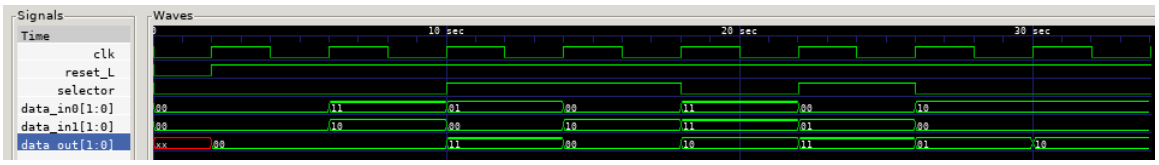

Figura 2: Respuesta esperada

2.2. Resultados y análisis

Inicialmente se comenzó buscando las librerías cmos_cells, en las cuales tenemos una serie de componentes que van a ser utilizados para el mapeo de los diseños. Para el primer punto de la tarea se construyó una síntesis estructural genérica (RTLIL), la cual no está relacionada a una tecnología específica. Al mostrar el diagrama gráfico generado por yosys se obtuvo lo mostrado en la siguiente figura. Los resultados de esta sección se pueden ver usando el comando "**$make parteA**.

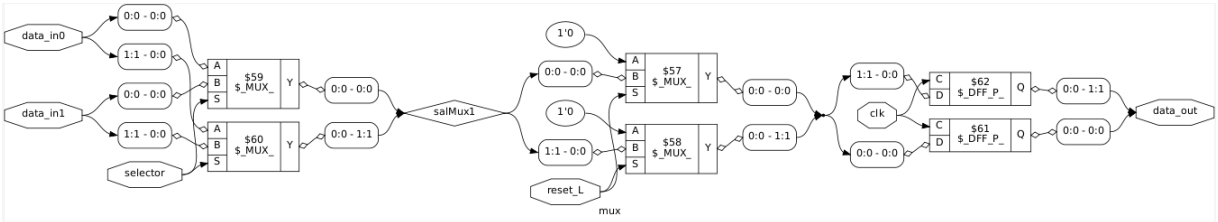

Figura 3: Diagrama de la descripción RTLIL

Se usó el comando sed para cambiar el módulo a un nombre que tenemos por defecto en el banco de pruebas para la prueba de este módulo. A correrlo y abrir los resultados en el software de GTKwave se observa que los resultados son los esperados, la descripción generada (estructural) coincide con la conductual. Para obtener estos resultados basca con correr el comando "$make parteA.[en] el directorio respectivo a esta tarea.

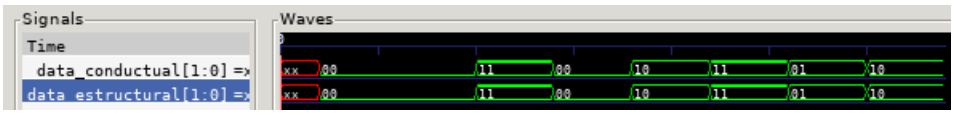

Figura 4: Respuesta de la descripción RTLIL

Usando las librerías mencionadas previamente se procedió a obtener una descripción estructural automática. Para este paso se usó el comando dfflibmap, que mapea los flipflops del circuito y abc, que termina de mapear el resto de los componentes. Este mapeo se hace con la librería original de yosys, es decir, no posee ningún retraso. Los resultados de esta sección se pueden ver usando el comando **"$make parteC"** en el directorio respectivo a esta tarea.

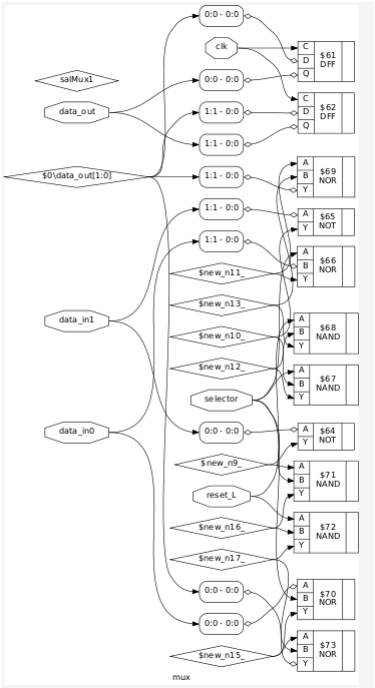

Figura 5: Diagrama de la descripción usando librería cmos_cell

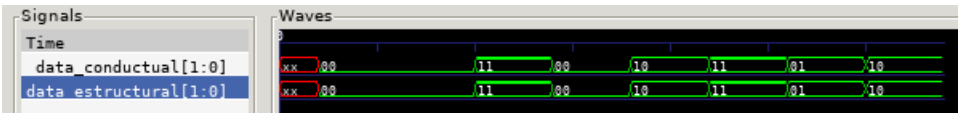

Figura 6: Respuesta de la descripción usando librería cmos_cell

En la figura 5 se observa el diagrama con compuertas obtenido con Yosys. La librería usada solo posee compuertas NAND, NOR, NOT y FLIPFLOP tipo D, por lo tanto el mapeo solo usó esta tecnología, sin embargo si se usan otras librerías como my_cells, se puede hacer mapeo en otras compuertas. Cabe destacar que el software da el mapeo con compuertas mínimas. También se observa que ambos comportamientos son iguales, por lo tanto se concluye que se hizo una descripción automática correcta.

Finalmente se modificó la librería cmos_cells para añadir los retardos d las compuertas según las hojas de fabricante extraídas de la página de Texas Instruments. Los componentes utilizados serán mencionados en la siguiente sección. La descripción estructural de esta sección es idéntica a la anterior, mas cambian las respuestas de salidas, pues en esta se genera un desfase por el retardo de los flip flops y tiene una frecuencia máxima de operación debido al camino más largo del diseño.

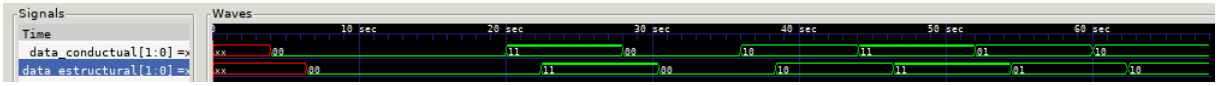

Figura 7: Respuesta de la descripción usando librería cmos_cell con retardos

Se obtuvo el periodo experimental de este circuito y se obtuo que es de 6.6ns, lo que equivale a una frecuencia máxima de $f_{máx}$ = $151MHz$.

2.3. Comprobación con el checker

Para comprobar el funcionamiento sin necesidad de ver las señales, se hizo un checker, que compara las señales cuando se da un flanco positivo. Para los casos mostrados en las figuras 4, 6 y 7 el checker mostró lo siguiente:

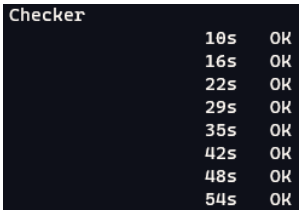

Figura 8: Resultado del checker en las pruebas mostradas en las figuras 4, 6 y 7

En el caso que se comprobó la frecuencia máxima con un retardo típico, se obtuvo los siguientes resultados al llegar a 6.4ns de periodo del clock.

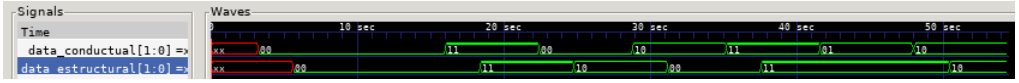

Figura 9: Respuesta de la descripción usando librería cmos_cell con retardos con un periodo de 6.4ns

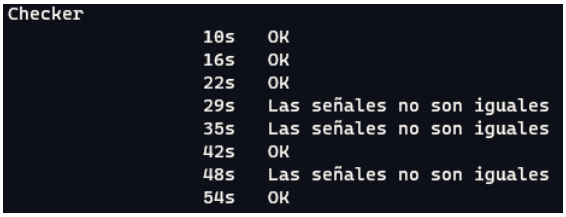

Figura 10: Checker de la respuesta con un clock de periodo 6.4ns

Como se observa, el checker identifica correctamente los errores. Cuando llega un flanco positivo el checkea el valor que tuvo traían ambas señales y si son iguales imprime OK, caso contrario muestra el error.

2.4. Uso de comando sed

Este comando fue de utilidad puesto que cada vez que se sintetiza un módulo, se usa el mismo nombre del módulo conductual original, lo cual puede generar inconvenientes a la hora de probar ambos resultados en un banco de prueba. El comando sed se implementó en el makefile de la siguiente manera: **sed -i "s/mux/mux_synth/g"$(SYNTH).v**, donde SYNTH en una etiqueta de path. Este comando cambió permanentemente la palabra mux por mux_synth en todo el archivo.

2.5. Componentes requeridos

Compuerta	Nombre	Ts [ns]	Potencia [μA]	Cantidad	Precio unidad [$]
NAND	SN74LVC1G38	(0.9:1.6:2.4)	10	4	0.129
NOR	SN74LVC1G02	(0.8:2.1:3.4)	10	4	0.063
NOT	SN74AHC1GU04	(1:3.7:6.5)	10	2	0.559
Flip Flop	SN74LVC1G80	(1.1:2.4:3.8)	10	2	0.086

Tabla 1: Componentes con sus características a un voltaje de alimentación de 5V

En la tabla 1 se muestran los precios de cada componente por unidad. Cabe destacar que si se compran estos componentes en grandes volúmenes los precios se reducen sustancialmente, sin embargo dado que en este caso se necesitan muy pocos únicamente se adjuntan los precios por unidad.

$$CostoComponentes = \$2{,}058$$

3. Conclusiones

Esta herramienta simplifica en gran medida el trabajo puesto que hace toda la síntesis con un par de líneas de comandos. Los resultados obtenidos con esta herramienta de síntesis fueron los esperados, todo se comportó respecto a las descripción conductual.

El costo es bastante elevado para los pocos componentes, si se quisiera construir un circuito de estos para la venta sería necesario recurrir a la compra de volúmenes grandes de componente que reduzcan estos números.

4. Uso del make

Para usar el make se debe utilizar un terminal de UNIX y moverse a la dirección en la que se tengan todos los archivos. Si se usa el comando make solo se corren todos los archivos sintetizados y se va abriendo el programa gtkwave para cada simulación. También se pueden correr los comandos:

- $make synth <- abre sintetizado sin mapeo
- $make nodelay <- sintetizado mapeado sin retardo
- $make delay <- sintetizado mapeado retardo

5. Tiempo empleado

- Búsqueda de librerías y componentes: 25 minutos
- Uso de comandos de Yosys: 1 hora
- Debuggeo debido a errores: 1.5 horas
- makefile: 20 minutos
- Reporte: 2 horas