Taller #1

Brayan Esteban Alpízar Elizondo
Mariana Gonzalez Sanabria

Taller de Diseño Digital
Instituto Tecnológico de Costa Rica
16 de agosto de 2025

1. Problema #2

Diseñe un restador completo de 4 bits con modelo de estructura en VHDL. Parta del diseño un restador completo de 1 bit.

1.1. Propuesta 1

La primera propuesta consiste en implementar el restador de 4 bits a partir de un *full adder* (sumador completo) como bloque base [1]. Este método se fundamenta en el principio de que una operación de resta puede transformarse en una operación de suma si se utiliza el complemento a 2 del sustraendo. Para lograrlo, cada bit del segundo operando se invierte y se introduce un *carry in* inicial con valor lógico '1'. De esta forma, el circuito del sumador completo puede emplearse directamente para realizar restas, sin modificar su lógica interna fundamental. El proceso de diseño inicia con la creación de un módulo de restador de 1 bit basado en un *full adder*, el cual recibe como entradas un bit del minuendo, un bit del sustraendo y el bit de acarreo/préstamo de la etapa anterior. Sus salidas corresponden al bit de diferencia y al acarreo/préstamo hacia la siguiente etapa. Una vez validado el módulo de 1 bit, se interconectan cuatro instancias idénticas en cascada para formar el restador completo de 4 bits.

Este diseño presenta algunas ventajas como la modularidad que permite trabajar con bloques pequeños y reutilizables, lo que facilita la depuración y el mantenimiento del código VHDL. Otra ventaja es la escalabilidad ya que aumentar el número de bits del restador implica únicamente replicar el módulo base y ajustar la conexión de los acarreos, sin necesidad de rediseñar la lógica interna. Además ya que si se dispone de un *full adder* previamente verificado, se reduce el riesgo de errores y se optimiza el tiempo de desarrollo. La verificación también se simplifica, puesto que validar un módulo elemental asegura que su comportamiento se mantendrá en el diseño completo.

1.2. Propuesta 2

La segunda propuesta plantea la implementación directa de un restador de 1 bit a partir de las ecuaciones lógicas mínimas de la operación de resta. En este diseño, se utilizan compuertas XOR, AND y OR para generar las señales de diferencia y de

préstamo (*borrow*) sin depender de un sumador completo. Las ecuaciones booleanas utilizadas pueden expresarse como:

$$D = A \oplus B \oplus Bin$$

$$Bout = (\overline{A} \cdot B) + (Bin \cdot \overline{A \oplus B})$$

donde D representa el bit de diferencia, $Bout$ es el préstamo de salida, A y B son los bits de entrada, y Bin es el préstamo de entrada. Una vez diseñado este módulo de 1 bit, se interconectan cuatro instancias para obtener el restador de 4 bits.

Aunque esta propuesta permite un control más directo sobre la cantidad y tipo de compuertas utilizadas, lo que en algunos casos puede reducir el retardo de propagación o el consumo de recursos lógicos, también presenta algunos problemas. El diseño y verificación resultan más complejos, ya que no se parte de un bloque previamente validado. Además que la reutilización es limitada, pues las ecuaciones están adaptadas a un fin específico, y cualquier modificación del comportamiento requiere revisar y ajustar la lógica. Si bien es posible replicar este módulo para aumentar el número de bits, no es tan escalable y es menos eficiente que en la primera propuesta.

1.3. Diseño Elegido

Tras analizar ambas alternativas de diseño, creemos la primera propuesta como la más adecuada. Su estructura modular basada en un *full adder* es mas util en cuanto a la reutilización, la escalabilidad y la facilidad de mantenimiento, permitiendo construir un sistema mas robusto a partir de componentes probados y con un menor riesgo de errores.

2. Problema #3

2.1. Propuesta 1

Para el diseño del contador con botón y reset asíncrono se plantearon dos posibles alternativas. La primera consiste en utilizar una señal de enable que se activa únicamente cuando se presiona el botón. En este diseño la señal del botón pasa por un circuito de debounce que elimina los rebotes mecánicos que se puedan crear y posteriormente se genera un pulso en un solo ciclo de reloj. En este pulso funciona el enable del contador, lo que asegura que el valor se incremente solamente cuando se presiona el botón y no de forma automática. Este enfoque mantiene todo el sistema bajo el mismo dominio de reloj de la FPGA, lo cual simplifica el diseño y mejora su confiabilidad.

2.2. Propuesta 2

La segunda idea de diseño se basa en una captura asíncrona del evento del botón. Su lógica es que cualquier cambio en el estado del botón, incluso si es muy pequeño activa un evento que luego se procesa en el siguiente ciclo de reloj para aumentar el

contador. La ventaja de esta propuesta es que evita perder pulsos muy cortos, aunque presenta la desventaja de tener un mayor cuidado con los rebotes del botón y problemas de sincronización por la lógica asíncrona involucrada.

2.3. Diseño Elegido

Se eligió la primera opción, la del enable sincrónico, ya que el reloj de la FPGA es rápido como para asegurar que los pulsos generados por un botón que duran mucho más que un ciclo de reloj no se pierdan. Además este diseño resulta más simple de implementar.

Referencias

[1] S. L. Harris y D. M. Harris, *Digital Design and Computer Architecture: ARM Edition*, 2nd. Cambridge, MA: Morgan Kaufmann, 2021.