Laboratorio 2. Lógica Combinacional y Aritmética

Brayan Alpízar Elizondo
Mariana González Sanabria
Escuela de Ingeniería en Computadores
Instituto Tecnológico de Costa Rica

I. DISEÑOS

A. Full adder 1 bit

Propuesta 1

La primera propuesta consiste en implementar un Full Adder de 1 bit utilizando variables internas para calcular la suma parcial y el acarreo. Se definen:

$$X1 = A \oplus B$$

$$T1 = A \cdot B$$

$$T2 = Cin \cdot X1$$

$$Sum = X1 \oplus Cin$$

$$Cout = T1 + T2$$

La tabla de verdad con estas variables intermedias es:

TABLE I
TABLA DE VERDAD DEL FULL ADDER DE 1 BIT CON VARIABLES INTERMEDIAS XOR/AND

A	B	Cin	$X1$	$T1$	$T2$	Sum	$Cout$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	1	0	0	1	0
0	1	1	1	0	1	0	1
1	0	0	1	0	0	1	0
1	0	1	1	0	1	0	1
1	1	0	0	1	0	0	1
1	1	1	0	1	0	1	1

—

Propuesta 2

La segunda propuesta consiste en implementar el Full Adder directamente mediante compuertas lógicas, sin variables intermedias. Las ecuaciones utilizadas son:

$$X1 = A \oplus B$$

$$T1 = A \cdot B$$

$$T2 = Cin \cdot X1$$

$$Sum = X1 \oplus Cin$$

$$Cout = T1 + T2$$

Físicamente no se implementan las variables intermedias pero estas sirven para visualizar de mejor forma el comportamiento en la tabla de verdad, la cual es la siguiente:

TABLE II
TABLA DE VERDAD DEL FULL ADDER DE 1 BIT CON VARIABLES INTERMEDIAS XOR/AND

A	B	Cin	$X1$	$T1$	$T2$	Sum	$Cout$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	1	0	0	1	0
0	1	1	1	0	1	0	1
1	0	0	1	0	0	1	0
1	0	1	1	0	1	0	1
1	1	0	0	1	0	0	1
1	1	1	0	1	0	1	1

Diseño Elegido

Tras analizar ambas propuestas, se elige la Propuesta 2, la implementación directa con XOR, AND y OR. Esta opción es más eficiente, ya que evita el uso de variables intermedias, reduciendo el retardo de propagación y simplificando el circuito. Además, facilita la replicación para construir sumadores de múltiples bits, manteniendo un diseño claro y compacto.

Aunque la Propuesta 1 con variables p y g permite visualizar los cálculos internos y es útil para depuración, la Propuesta 2 ofrece un mejor rendimiento y menor consumo de recursos lógicos en hardware.

B. Restador

Propuesta 1

La primera propuesta consiste en implementar el restador de 4 bits a partir de un *full adder* (sumador completo) como bloque base. 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.

La operación lógica del restador de 1 bit puede representarse como:

$$D = A - B - Bin$$

$$Bout = \text{Préstamo de salida}$$

Definiciones de las variables intermedias para la visualización en la tabla:

$$X1 = A \oplus B$$

$$T1 = \overline{A} \cdot B$$

$$T2 = Bin \cdot \overline{X1}$$

$$D = X1 \oplus Bin$$

$$Bout = T1 + T2$$

Como se dijo anteriormente, físicamente no se implementan las variables intermedias. La tabla de verdad quedaría de esta forma:

TABLE III
TABLA DE VERDAD DEL RESTADOR DE 1 BIT (PROPUESTA ELEGIDA)

A	B	Bin	$X1$	$T1$	$T2$	D	$Bout$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1
0	1	0	1	1	0	1	1
0	1	1	1	1	0	0	1
1	0	0	1	0	0	1	0
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	1	1	1

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 son:

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

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

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.

Definiciones de las variables intermedias para la visualización en la tabla:

$$X1 = A \oplus B$$

$$T1 = \overline{A} \cdot B$$

$$T2 = Bin \cdot \overline{X1}$$

$$D = X1 \oplus Bin$$

$$Bout = T1 + T2$$

La tabla de verdad correspondiente es:

TABLE IV
TABLA DE VERDAD DEL RESTADOR DE 1 BIT (PROPUESTA 2)

A	B	Bin	$X1$	$T1$	$T2$	D	$Bout$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1
0	1	0	1	1	0	1	1
0	1	1	1	1	0	0	1
1	0	0	1	0	0	1	0
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	1	1	1

Diseño Elegido

Tras analizar ambas alternativas de diseño, se elige la segunda propuesta, es decir, implementar directamente el restador de 1 bit a partir de las ecuaciones lógicas mínimas. Esta decisión se basa en que este enfoque es más eficiente en cuanto a la cantidad de compuertas utilizadas y al retardo de propagación, ya que no depende de un *full adder* previamente construido.

Aunque la primera propuesta ofrece ventajas en modularidad y escalabilidad, la Propuesta 2 permite un control más directo sobre la lógica interna del restador, reduciendo el consumo de recursos y optimizando el rendimiento. Además, al replicar este módulo de 1 bit, se puede construir un restador de 4 bits eficiente, conservando la corrección funcional garantizada por las ecuaciones booleanas.

C. Multiplicador de 1 bit

La multiplicación de números binarios de 1 bit equivale a una operación AND, por lo que se utilizan compuertas AND para formar los productos parciales.

Propuesta 1

Se implementa el multiplicador de 1 bit directamente con compuertas AND para el producto y un bloque AND adicional para generar el Carry:

$$P = A \cdot B$$

$$Cout = P \cdot Cin$$

Tabla de verdad:

Propuesta 2

Se reutiliza un módulo de Full Adder para implementar el multiplicador, considerando el Carry de entrada y salida. Se definen variables intermedias:

TABLE V
TABLA DE VERDAD DEL MULTIPLICADOR DE 1 BIT CON CARRY (PROPUESTA 1)

A	B	Cin	P	$Cout$
0	0	0	0	0
0	1	0	0	0
1	0	0	0	0
1	1	0	1	0
0	0	1	0	0
0	1	1	0	0
1	0	1	0	0
1	1	1	1	1

$$P = A \cdot B$$

$$Sum = P \oplus Cin$$

$$Cout = P \cdot Cin$$

Tabla de verdad:

TABLE VI
TABLA DE VERDAD DEL MULTIPLICADOR DE 1 BIT CON FULL ADDER (PROPUESTA 2)

A	B	Cin	$X1$	Sum	$Cout$
0	0	0	0	0	0
0	1	0	0	0	0
1	0	0	0	0	0
1	1	0	1	1	0
0	0	1	0	1	0
0	1	1	0	1	0
1	0	1	0	1	0
1	1	1	1	0	1

Diseño Elegido

Se elige la Propuesta *, ya que es más eficiente en cuanto a la cantidad de compuertas y el retardo de propagación, pero mantiene el carry de entrada y salida, lo que permite conectar múltiples módulos para formar multiplicadores de varios bits de manera escalable y modular.

Implementación para N bits:

Para un multiplicador de N bits, se genera una matriz de productos parciales, donde cada bit del multiplicando se multiplica por cada bit del multiplicador usando el multiplicador de 1 bit elegido (Propuesta 1). Luego, los productos parciales se desplazan según la posición del bit y se suman utilizando el Full Adder (Propuesta 2) en cascada, propagando los carries correctamente. De esta forma, el multiplicador de 1 bit sirve como bloque base modular que se repite N×N veces para construir el multiplicador completo.

REFERENCES

[1] S. Harris y D. Harris, *Digital Design and Computer Architecture: ARM Edition*. Morgan Kaufmann, 2015.