Laboratorio #6:
Sumadores

Escuela de Ingeniería Eléctrica

Universidad de Costa Rica

1 Introducción

1.1 Objetivos del Laboratorio

En este laboratorio, las personas estudiantes explorarán el diseño e implementación de circuitos sumadores, fundamentales en sistemas digitales y aplicaciones de procesamiento de datos. A través del desarrollo práctico, se enfocarán en el diseño modular utilizando Verilog y la comparación de diferentes topologías de sumadores para comprender su impacto en el desempeño y la complejidad.

A través de esta actividad, los estudiantes lograrán:

- Comprender el funcionamiento de un sumador completo (full adder) y su integración en estructuras más complejas.
- Diseñar e implementar sumadores en cascada y analizar su funcionamiento.
- Explorar nuevas topologías de sumadores, como los sumadores de acarreo anticipado, y comprender sus diferencias con el diseño en cascada.
- Desarrollar habilidades prácticas en diseño y programación en Intel Quartus Prime Lite.

2 Conceptos teóricos necesarios

2.1 Sumador Completo

El sumador completo (*full adder*) es un circuito que recibe 3 entradas de un bit cada una, dos de estas (`A` y `B`) representan dígitos binarios y la tercer entrada (`C_in`) corresponde a un acarreo de entrada. Adicionalmente, el circuito posee dos salidas que en conjunto representan el resultado de la suma: `S` es el bit menos significativo del resultado y `C_out` es el bit más significativo del resultado y representa si hubo un acarreo en la suma.

La suma efectuada por este circuito puede verse de la siguiente manera:

$$\begin{array}{rr} & A \\ & B \\ + & C_{in} \\ \hline C_{out} & S \end{array}$$

Note que para una suma de entradas de **un** bit se requiere una salida de **dos** bits. En general, el resultado de la suma de dos números de **n** bits, necesita **n+1** bits para representarse.

Adicionalmente, la tabla de verdad para este circuito corresponde a la mostrada en la Tabla 1.

Y su funcionamiento se puede describir en Verilog mediante el siguiente módulo:

A	B	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Tabla 1: Tabla de verdad para un sumador completo

fulladder.v

```
//Declaracion del modulo fulladder
module fulladder (A, B, Cin, Cout, S);
  //Declaracion de entradas y salidas
  input A, B, Cin;
  output reg Cout, S;

  //Cada vez que cambie alguna entrada
  always @(*) begin
    //Realice la suma A+B+Cin
    //Y asigne el resultado a {Cout, S}
    //Cout contiene el bit mas significativo del resultado
    //S contiene el bit menos significativo del resultado
    {Cout, S} = A + B + Cin;
  //Termina always
  end
//Termina modulo
endmodule
```

2.2 Sumador de Acarreo en Cascada

Al igual que en una suma de dos o más dígitos decimales, el procedimiento para una suma de números binarios de varios bits consiste en sumar entre sí los dos dígitos para una misma posición de significancia y, en caso de ser necesario, a dicho resultado sumarle el acarreo producido por la suma de la posición anterior.

Note que las entradas y salidas del *full adder* permiten fácilmente replicar y expandir este procedimiento pues la suma de los bits `A` y `B` se puede efectuar para cada par de bits con la misma significancia en los sumandos y mediante la salida `C_out` y la entrada `C_in` se puede asignar el acarreo producido por un par de bits al sumador con los siguientes bits en significancia.

El Sumador de Acarreo en Cascada, o Ripple Carry Adder (RCA), es una red iterativa que se utiliza varias sumadores completos para llevar a cabo la operación de suma de múltiples bits. Por ejemplo, para un sumador de 4 bits, la estructura descrita se puede ver representada en la Figura 1, donde el resultado de la suma estaría dado por $\{C_{out,3}, S_3, S_2, S_1, S_0\}$.

En la Figura 1 las señales azules representan entradas al sumador de 4 bits, las salidas del sumador se representan en rojo y en negro se muestras conexiones internas entre los sumadores completos.

Lo anterior no está limitado a una cantidad específica de bits sino que se puede extender para sumandos de cualquier tamaño. El resultado de la suma de los dos números de n bits es la concatenación del acarreo de salida del último sumador y el resultado de la salida `S` de cada sumador, manteniendo su orden.

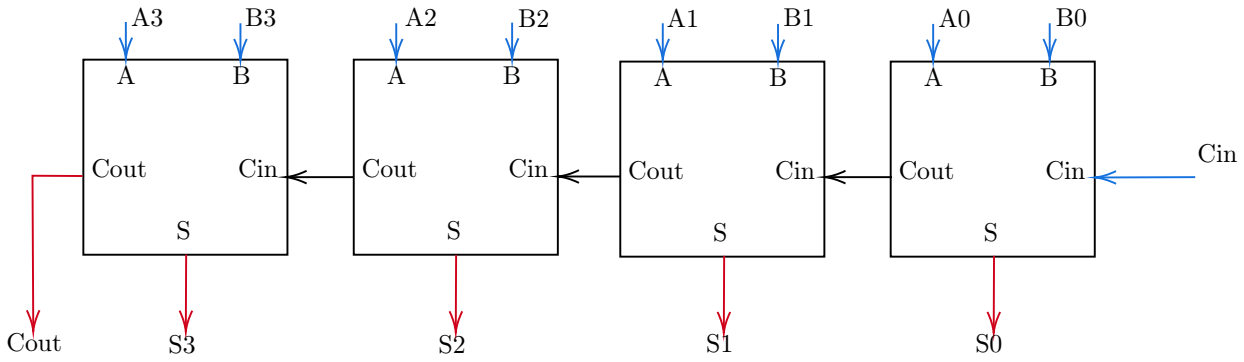

Figura 1: Diagrama de un CLA de 4 bits

2.3 Sumador de Acarreo Anticipado

El sumador de acarreo anticipado es una optimización del sumador de acarreo en cascada (RCA) que reduce el tiempo de propagación del acarreo. En un RCA tradicional, cada bit de la suma espera la propagación del acarreo desde el bit anterior, lo que genera un retardo lineal proporcional al número de bits, en sumador de 256 bits, el acarreo de entrada del sumador del bit más significativo debe pasar por 255 sumadores para poder calcularse. El Sumador de Acarreo Anticipado, o Carry Look-Ahead Adder (CLA) soluciona este problema generando los acarreos de forma simultánea mediante expresiones lógicas, lo que permite una ejecución más rápida. Sin embargo, a medida que se aumenta la cantidad de bits, la cantidad de hardware necesario crece de una forma mucho más rápida que en el RCA, lo que podría hacer inviable su implementación cuando los recursos son limitados.

En un sumador de acarreo en cascada, el acarreo del bloque correspondiente al $(i+1)$-ésimo bit se calcula utilizando la expresión:

$$C_{i+1} = (A_i \cdot B_i) + C_i \cdot (A_i \oplus B_i)$$

Es decir, es necesaria una propagación secuencial del acarreo. Para evitar este problema, el sumador de acarreo anticipado introduce dos términos auxiliares, la generación y la propagación:

- **Generación de acarreo**: Cuando ambas entradas (A_i y B_i) son 1, se *generará* un acarreo, sin importar el acarreo de entrada. Esto se expresa de la siguiente forma:

$$G_i = A_i \cdot B_i$$

- Si solamente una de las entradas (A_i o B_i) son 1 (pero no ambas), el acarreo de salida será igual al valor del acarreo de entrada. Es decir, el valor del acarreo de entrada se *propagará* al acarreo de salida. Esto se expresa de la siguiente forma:

$$P_i = A_i \oplus B_i$$

Entonces, la expresión de acarreo de salida se puede reescribir de la siguiente manera:

$$C_{i+1} = G_i + C_i \cdot P_i$$

Aún con esta expresión, se tiene un acarreo de salida que depende del acarreo del bloque anterior, sin embargo, se sabe que:

$$C_1 = G_0 + C_{in} \cdot P_0$$
$$C_2 = G_1 + C_1 \cdot P_1$$
$$C_3 = G_2 + C_2 \cdot P_2$$
$$C_{out} = G_3 + C_3 \cdot P_3$$

Por lo que las expresiones de acarreos intermedias se pueden reemplazar, para obtener las siguientes expresiones:

$$C_1 = G_0 + C_{in} \cdot P_0$$
$$C_2 = G_1 + (G_0 + C_{in} \cdot P_0) \cdot P_1$$
$$C_3 = G_2 + (G_1 + (G_0 + C_{in} \cdot P_0) \cdot P_1) \cdot P_2$$
$$C_{out} = G_3 + (G_2 + (G_1 + (G_0 + C_{in} \cdot P_0) \cdot P_1) \cdot P_2) \cdot P_3$$

En estas últimas ecuaciones, ninguna de las expresiones depende de acarreos intermedios, únicamente dependen de A, B (G y P sólo dependen de los sumandos) y C_{in}, por lo que todos los acarreos pueden calcularse en paraleo, sin necesidad de propagar señales intermedias.

En la Figura 2 se muestra un diagrama de un sumador de acarreo anticipado de cuatro bits. En el diagrama se puede apreciar un módulo nuevo, el cual se encarga de generar todos los acarreos a partir de los operandos. En azul se muestran las señales de entrada al sumador de acarreo anticipado, en rojo sus salidas, en verde las señales que los sumadores completos envían a la lógica de acarreo anticipado y en morado las señales que la lógica de acarreo anticipado le brinda a los sumadores completos.

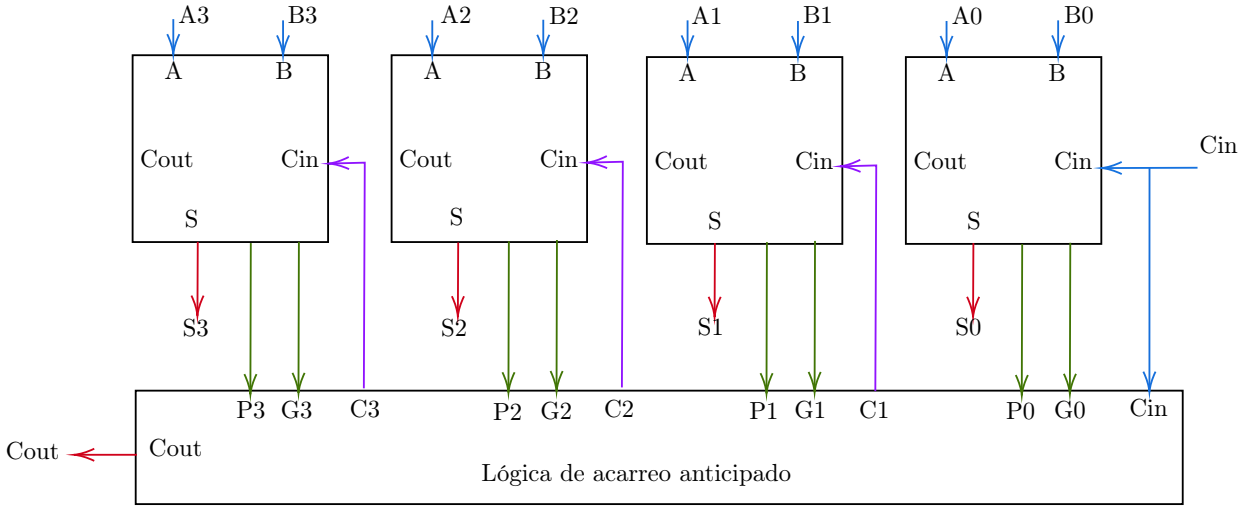

Figura 2: Diagrama de un CLA de 4 bits

Como se puede apreciar a partir de las ecuaciones, el CLA es eficiente para tamaños pequeños y medianos (4 a 16 bits), pero a medida que el número de bits aumenta, la complejidad del circuito se vuelve un problema. En arquitecturas de 32 y 64 bits, se utilizan híbridos como los sumadores Carry Skip o Carry Select para mantener un balance entre velocidad y costo en hardware.

3 Trabajo en Clase

Recuerde que el dispositivo que se usará es el Max 10 10M50DAF484C7G.

1. Implemente el sumador de acarreo en cascada de 4 bits mostrado en la Figura 1. Este módulo deberá contener múltiples instancias del módulo `fulladder` descrito anteriormente. Utilice el mapeo descrito a continuación, respetando el orden de significancia para los sumandos:

   Sumando 1 (A): {`SW3`, `SW2`, `SW1`, `SW0`}

   Sumando 2 (B) : {`SW9`, `SW8`, `SW7`, `SW6`}

   Acarreo de Entrada (C_{in}) : `Key1`

   Despliegue el resultado de la suma utilizando los LEDs que necesite de la tarjeta de desarrollo, asigne `LEDR0` como el bit menos significativo del resultado.

   Nota: Tome en cuenta que los botones `Key0` y `Key1` producen un nivel lógico **alto** cuando **no están siendo presionados** y que adoptan un estado lógico **bajo** cuando **son presionados**.

Para la DEMOSTRACIÓN deberá mostrar los diseños descritos funcionando de forma adecuada. Apenas finalice cada diseño, puede solicitar a la persona docente hacer el demo. La calificación se repartirá de la siguiente manera:

- Sumador RCA: 100k%