

1 Ejercicio 6: Implementación de una ALU

Para este ejercicio se pidió implementar una ALU de 4 bits con las operaciones SUMA, RESTA, AND, OR, NOT, XOR, Complemento a dos y Shift Left. Para esto se decidió crear un modulo encargado de sumar 2 bits, este a su vez devuelve un bit de respuesta y un bit de carry. Utilizando ese modulo, se procedió a hacer un nuevo modulo que haga una suma de 3 bits. Esta decisión simplificó mucho el desarrollo del modulo de suma de la ALU de 4 bits, ya que como se puede ver en el código “sum.v” encontrado en la carpeta src, se puede ver comentado dentro del código como fue nuestro desarrollo sin la función sum3Bits y como quedo finalmente el código con la implementacion de sum3Bits.

Para el caso de la resta se decidió utilizar el modulo hecho en el ejercicio 4 del complemento a 2 y así, se utilizó en conjunto ese modulo con el modulo suma.

En el caso de las operaciones AND, OR, NOT y XOR, se opto por utilizar las funciones predefinidas por verilog y reutilizarlas bit a bit.

1.1 Definiciones

Esta unidad aritmetica lógica se implemento con 2 acumuladores (que llamaremos A y B en este orden) de 4 bits cada uno, tres bits de operaciones, 4 bits del acumulador de salida (que llamaremos Acumulador C) y un bit de carry, ordenados en el orden en que fueron mencionados. Para seleccionar que operacion se desea hacer, se deben encender los 3 bits de operaciones de la siguiente manera:

	Bit 0	Bit 1	Bit 2
AND	0	0	0
NOT	0	0	1
OR	0	1	0
XOR	0	1	1
SHIFT LEFT	1	0	0
SUM	1	0	1
SUBTRACTION	1	1	0
TWO'S COMPLEMENT	1	1	1

Table 1: Representacion de operaciones en los bits.

- AND: Realiza una operacion AND bit a bit entre acumuladores A y B y la devuelve en el acumulador C, a su vez, el carry se devuelve en cero.
- NOT: Realiza una operacion logica NOT bit a bit del acumulador A y devuelve la respuesta en el acumulador C, a su vez, el carry queda en cero.
- OR: Realiza una operacion OR bit a bit entre acumuladores A y B y la devuelve en el acumulador C, a su vez, el carry se devuelve en cero.
- XOR: Realiza una operacion XOR bit a bit entre acumuladores A y B y la devuelve en el acumulador C, a su vez, el carry se devuelve en cero.
- SHIFT LEFT: Se encarga de mover cada bit del acumulador A un espacio a la derecha e inserta un cero al bit menos significativo, la respuesta la devuelve en el acumulador C y el carry valdra 0 o 1 dependiendo del bit mas significativo de A

- SUM: Se encarga de hacer una suma numerica de los valores decimales de los acumuladores A y B y devuelve el resultado (en codigo binario) en el acumulador C. Dependiendo si es representable el resultado de la suma en un nibble se encendera o no el bit Carry.
- SUBTRACTION: Se encarga de hacer una resta numerica de los valores decimales de los acumuladores A y B y devuelve el resultado (en codigo binario) en el acumulador C. El carry se devuelve en 1.
- TWO'S COMPLEMENT: Se encarga de hacer el complemento a 2 del acumulador A y devuelve el resultado en el acumulador C.