Universidad Nacional Autónoma de México

Facultad de Ciencias

Práctica 04

Organización y Arquitectura de Computadoras

Johann Ramón Gordillo Guzmán - 418046090

José Jhovan Gallardo Valdez - 310192815

Proyecto presentado como parte del curso de **Organización y Arquitectura de Computadoras** impartido por el profesor **José de Jesús Galaviz Casas**.

18 de Septiembre del 2019

Link al código fuente: https://github.com/JohannGordillo/

1. Preguntas

1. ¿Qué operaciones aritméticas y lógicas son básicas para un procesador? Justifica tu respuesta.

   Hay un Teorema dentro de la Lógica Matemática que nos dice que cualquier conecctivo unario o binario es lógicamente equivalente a una expresión escrita en términos únicamente de los conectivos lógicos $\wedge$ y $\neg$. Es decir, que $\{\wedge, \neg\}$ es un sistema de conectivos completo.

   Así, las operaciones lógicas básicas para un procesador son el **Not** y el **And**. A partir de ellos, podemos realizar las operaciones operaciones lógicas y aritméticas básicas, como el Or, la suma y la resta. Aunque la resta la podemos realizar a partir de la suma, ya que $A - B$ es equivalente a $A + \overline{B} + 1$.

   Las comparaciones entre entradas, las podemos realizar a partir de la resta, como vimos en la presente práctica, y se muestra en el circuito entregado. Pero al ser la resta una suma, podemos decir que las comparaciones se realizan a partir de la operación aritmética suma. Por lo que las operaciones lógicas básicas son el And y el Not, y la operación aritmética básica es la suma. A partir de estas operaciones, podemos realizar todas las demás.

2. El diseño utilizado para realizar la adición resulta ser ineficiente, ¿por qué? ¿Qué tipo de sumador resulta ser más eficiente?

   El diseño utilizado corresponde al de un **Ripple-Carry Adder**. Este diseño es ineficiente, pues cada uno de los sumadores completos debe esperar por el acarreo de salida del sumador anterior, mismo que recibirá como acarreo de entrada.
   Un diseño más eficiente es el de un **Carry-Lookahead Adder**.

3. Bajo este diseño, en la ALU se calculan todas las operaciones de forma simultanea pero sólo se entrega un resultado, ¿se realiza trabajo inútil? ¿Toma tiempo adicional? ¿Cuál es el costo?

   Sí se realiza trabajo inútil, pero no se desperdicia tiempo ni toma tiempo adicional, ya que las operaciones se realizan al mismo tiempo (quizás podría provocarse algún retraso en el multiplexor si las opciones son demasiadas), pero sí se realiza un consumo energético en las compuertas que no estamos usando.

4. ¿Cuántas operaciones más podemos agregar al diseño de esta ALU? ¿Qué tendríamos que modificar para realizar más operaciones?

   Podemos agregar la división y la multiplicación. Para la división usaremos las comparaciones y la resta, además tendremos que agregar componentes que nos permitan formar el cociente de la división. Para el producto de bits, usamos And, pero tendremos que formar el resultado partiendo de los subproductos obtenidos.