UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Organización y Arquitectura de Computadoras
Practica 4

PRESENTA

Carlos Emilio Castañon Maldonado
Pablo César Navarro Santana

PROFESOR

José de Jesús Galaviz Casas

AYUDANTES

Ricardo Enrique Pérez Villanueva
Sara Doris Montes Incin
María Ximena Lezama Hérnandez

Organización y Arquitectura de Computadoras

Practica 4

Ejercicios

1 **Da una breve introducción a que son las ALUs en tu reporte.**
Una ALU es uno de los componentes principales de la CPU (Unidad Central de Procesamiento) de una computadora. Su función es realizar operaciones aritméticas y lógicas en los datos que se encuentran en los registros de la CPU.
Las operaciones aritméticas incluyen la suma, la resta, la multiplicación y la división, mientras que las operaciones lógicas incluyen la comparación, la negación y la conjunción/-disyunción.
La ALU es una parte fundamental de la arquitectura de una computadora, ya que todas las operaciones que realiza la CPU, desde el procesamiento de texto hasta la ejecución de videojuegos, dependen de su capacidad para realizar operaciones aritméticas y lógicas precisas y rápidas.

Preguntas

1 **¿Quien propuso por primera vez el diseño de una ALU? ¿Para que computador se propuso?**
Fue John von Neumann para la computadora EDVAC

2 **¿Que es VHDL? ¿Como se ve un full-adder en VHDL?**
VDHL, por su siglas en ingles Very High Speed Integrated Circuit (VHSIC) Hardware Description Language, es un lenguaje para describir circuitos electronicos propuesto por la IEEE.
Es como un lenguaje de programación, en el cual vamos declarando partes de nuestro circuito y como funciona entre ellos, así como su entradas y salidas.
La representación de un Full adder, sería la siguiente:

```
library IEEE;
use IEEE.STD\_LOGIC1164.ALL;
use IEEE.STD\_LOGIC\_ARITH.ALL;
use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity half\_adder is
    port(a,b:in bit; sum,carry:out bit);
end half\_adder;

architecture data of half\_adder is
begin
    sum<= a xor b;
    carry <= a and b;
end data;
```

Podemos ver como estamos importando librerías, usando variables y definiendo como funcionaran los operadores lógicos como lo son xor y and. Es por esto que mencionamos que se parece demasiado a un lenguaje de programación.

3 **¿Que operaciones aritméticas y lógicas son básicas para un procesador?**
Justifica tu respuesta. Las operaciones aritméticas y lógicas básicas que son esenciales para el funcionamiento de un procesador son:

Suma y Resta: Estas operaciones se utilizan para sumar o restar dos números y son fundamentales para realizar cálculos básicos.

Multiplicación y División: Estas operaciones son esenciales para realizar cálculos más complejos y son comunes en aplicaciones como gráficos y procesamiento de señales.

Operaciones lógicas: Estas incluyen AND, OR y NOT. Estas operaciones son importantes para realizar operaciones como la comparación o como verificar si dos números son iguales o si un número es mayor que otro.

Operaciones de comparación: Estas incluyen igualdad, mayor que, menor que, mayor o igual que y menor o igual que. Estas operaciones son importantes para tomar decisiones y ramificar el flujo de ejecución del programa.

4 **El diseño utilizado para realizar la adición resulta ser ineficiente, ¿por que?**
¿Que tipo de sumador resulta ser mas eficiente?
Recordando que nuestro Sumador de 8 bits usa Full Adders.
Es ineficiente debido a que requiere muchos componentes lógicos para su implementación, cada Full Adder requiere tres entradas (los dos bits a sumar y el bit de acarreo de entrada) y produce dos salidas (el bit de suma y el bit de acarreo de salida), ahora, para sumar dos números de n bits, se necesitan n Full Adders, lo que significa que se requieren $3n$ entradas y $2n + 1$ salidas.
El conflicto con esta implementación es que el número de componentes lógicos y las interconexiones necesarias aumentan con el tamaño de los números a sumar, esto puede llegar a ser un inconveniente para operaciones con números de gran tamaño, ya que la cantidad de componentes lógicos y la complejidad del diseño se incrementan significativamente.
Eso claro sumado a que el tiempo de propagación del acarreo es lento debido a que el acarreo debe ser propagado a través de todas las etapas de bits conectados en serie.

5 **Bajo este diseño, en la ALU se calculan todas las operaciones de forma simultanea pero solo se entrega un resultado, ¿se realiza trabajo inútil? ¿Toma tiempo adicional?**
¿Cual es el costo?
Sí, bajo ese diseño, es probable que se realice trabajo inútil en la ALU, ya que si se tienen todas las operaciones de la ALU calculándose al mismo tiempo con solo dos entradas, la(s) operaciones que no sean de nuestro interés solo estarán realizando cálculos innecesarios que aumentan el tiempo y el costo que nos representa nuestra ALU.
Por ejemplo, si se están realizando operaciones de suma y resta simultáneamente, es posible que se estén sumando los mismos dos operandos varias veces, lo que resultaría en trabajo y tiempo que podríamos habernos ahorrado.
Además, al realizar todas las operaciones simultáneamente, se puede generar un consumo de energía y un costo de hardware innecesario, ya que se están realizando cálculos que no se necesitan para la operación deseada.
En general, es más eficiente diseñar una ALU que realice solo la operación requerida en lugar de realizar todas las operaciones simultáneamente.

6 **¿Cuantas operaciones mas podemos agregar al diseño de esta ALU?**
¿Que tendriamos que modificar para realizar mas operaciones?
Para agregar más operaciones a la ALU, se necesitaría modificar su diseño para que esta pueda realizar las nuevas operaciones, esto daría pie a que debemos agregar nuevos componentes lógicos o modificar los existentes para implementar las operaciones adicionales.
Algunas operaciones que podríamos implementar son:
Multiplicación: El cual podría ser implementado si agregamos un circuito de multiplicación a la ALU que utiliza suma de sumas para realizar la operación.

Comparación de desigualdad (mayor que o igual que): El cual podría ser implementado si agregamos un circuito de comparación extra a la ALU que haga uso de nuestro circuito de igualdad y desigualdad que ya tenemos, obteniendo así una nueva operación que determina si un número es mayor o igual que otro.