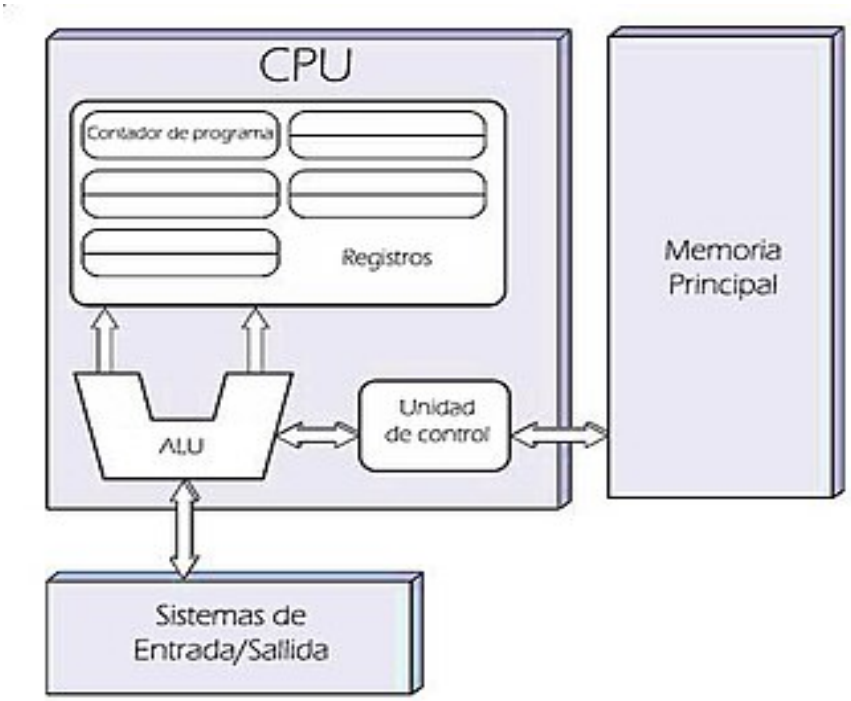
Arquitectura en computadoras

Taller 1



Integrantes: Sandy Perez, Martin Aburto
Profesores: Iván Pedro Nomdedeu, Juan Ignacio Casse
Materia: Arquitectura de Computadoras
Fecha de entrega: 15 de abril 2024

Introducción

La arquitectura de una computadora es fundamental porque define cómo los componentes del hardware se conectan e interactúan entre sí, una arquitectura bien diseñada favorece entre otras cosas: la compatibilidad del software, la seguridad y el rendimiento de una computadora.
En este taller nos adentraremos un poco en el mundo del diseño, comenzando desde la construcción de una ALU de 8 bits con registros y buses de datos, hasta el diseño de una arquitectura con su propio set de instrucciones. Además, también calcularemos la performance de la arquitectura planteada.

Parte 1- Alu de 8 bits

Creamos una ALU de 8 bits con registros y buses a partir de la ALU de 4 bits que teníamos hecha en electrónica digital. Al comienzo parecía que no tendríamos que hacerle tantas modificaciones, ya que solo había que cambiar el tamaño de las entradas, pero lo cierto es que mientras avanzamos descubrimos que era un poco más complejo de lo que pensábamos.

Lo primero que tuvimos que modificar fueron los sumadores, ya que para la ALU de 4 bits usábamos sumadores de cuatro bits pero para esta necesitábamos que fueran de 8, así que, al igual que decidimos construirlo utilizando los sumadores de 2 bits que ya teníamos armados.

También tuvimos que modificar el complementador A2 para que tenga entradas de 8 bits y funcione a base del sumador completo de 8 bits, de esa forma podíamos continuar con la implementación de la unidad aritmética.

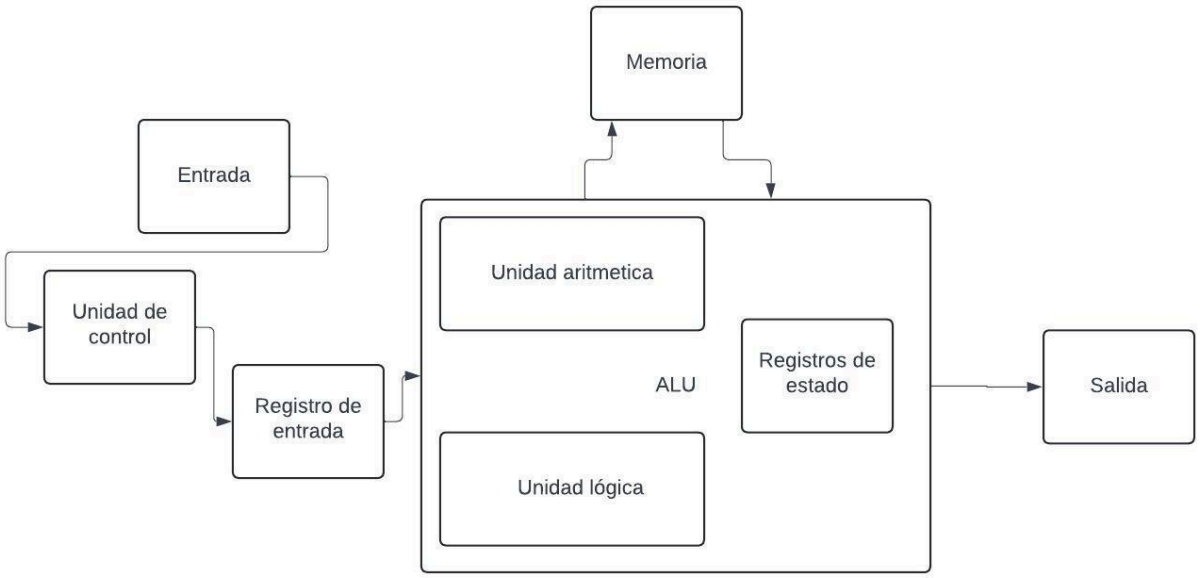
Para la unidad aritmética utilizamos todo lo antes planteado y le agregamos las flags de Zero, Overflow, Carry y Negativo, además agregamos un componente para que pueda realizar la multiplicación, este no lo diseñamos nosotros si no que usamos el módulo que trae por defecto Logisim.

Para la ALU simplemente tuvimos que usar la unidad aritmética modificada sumada a las puertas and y or y un multiplexor de 8 entradas con 3 bits de selección. De esta forma ya teníamos la ALU resuelta, solo nos quedaba agregarle los registros. Para hacer que la ALU de 8 bits también guarde en registros las entradas y las salidas, preparamos un registro de salida restringida de 8 bits (basándonos en lo que teníamos hecho del taller de electrónica digital), de esta forma podíamos almacenar las entradas y salidas y mostrarlas activando la opción “EN_OUT”. Finalmente,solo tuvimos que agregar los registros en ambas entradas y en la salida de la ALU, pero todavía nos quedaba un paso más, agregar los buses.

Para terminar la ALU de 8 bits debíamos agregarle buses de datos y dos registros adicionales, por lo que eso hicimos, conectamos cuatro registros de salida restringida a un bus de datos que contenía la información de una única entrada de 8 bits, a cada registro contar con una opción para habilitar la escritura (“W”) y la antes mencionada opción de hacer salir los datos (“EN_OUT”) se puede controlar el registro en el que se escribe la información, además, como la salida de la ALU también tiene un registro, se puede utilizar la salida de una operación anterior como componente de una siguiente, almacenando el resultado en alguno de los registros.

Parte 2 - Diseño de CPU

Diseño de la arquitectura.



Descripción de la arquitectura

La arquitectura cuenta con ocho entradas de datos de 8 bits, estas ocho entradas se almacenan en ocho registros de entrada diferentes.

Antes de que los datos se guarden en los registros de entrada, pasan por la unidad de control en donde se averigua la operación que se realizará en la ALU y se controlan los datos que entrarán ahí. Además en la unidad de control se encuentra el contador de programa que es un registro especial que almacena la dirección de la próxima instrucción de memoria, esta a su vez es almacenada en la memoria principal. El contador de programa (PC) es esencial para mantener el correcto flujo de operaciones en la CPU, permitiendo que estas se realicen de manera ordenada. Además, el PC permite realizar operaciones de tipo salto para así poder realizar condiciones o bifurcaciones o simplemente saltar hacia la dirección de instrucciones que correspondan a otro programa.

La ALU contiene una unidad lógica, encargada de hacer operaciones lógicas sobre todo de comparaciones como por ejemplo: OR, AND, XOR y XNOR y una unidad

aritmética, que como dice su nombre, se encarga de hacer problemas aritméticos de muchos tipos. La ALU tiene sus propios registros especiales para guardar información que saca de la memoria principal, de esta forma se aumenta la eficiencia al no tener que estar buscando en la memoria principal (que es mucho más extensa) cada vez que se quiere acceder a un dato.

La información entre los bloques se transmite mediante buses de datos de 32 bits, así siempre puede viajar por el bus una instrucción completa.

Set de instrucciones:

ISA Básica

Categoría	**Introducción**	**Ejemplo**	**Significado**	**Comentario**
Transferencia de datos	Dat	dat r0 = A	r0=Memory [A]	palabra de memoria a registro
Aritmética	Sum	sum r2-> r1,r0	r2=r1+r0	tres operando; datos en registro
Aritmética	Mul	mul r2-> r1,r0	r2=r1*r0	tres operando; datos en registro

Las instrucciones tienen un direccionamiento de tipo load/store, en nuestro programa solamente utilizamos dos tipos de instrucciones: una de tipo I (dat) y dos de tipo R (sum y mul).

Las instrucciones de tipo I siguen el siguiente formato:

100011 00100 01000 0000000000000000 (dat)

Los cinco primeros bits de la tira corresponden a la operación a realizar, en este caso es la operación dat, para asignar un valor a un registro. Los siguientes cinco bits se refieren a la dirección del registro en el que se guardará el valor, en este caso es en la dirección 100. En este caso se utiliza un direccionamiento indexado, por lo que los siguientes cinco bits se refieren a una dirección de memoria concreta mientras que los siguientes dieciséis bits simbolizan el desplazamiento, en este caso, como queremos el valor que hay exactamente en esa dirección de memoria, el desplazamiento es cero.

Para las instrucciones de tipo R el formato es el siguiente:

000000 00100 00101 01001 00000 000000 (sum)

Al igual que en las operaciones de tipo I, los cinco primeros bits se refieren a la operación que se desea realizar, en este caso la suma. Los siguientes cinco bits son la dirección del registro en donde se almacenará el resultado. En los siguientes diez bits se almacenan los operandos (con cinco bits de espacio cada uno). Los siguientes cinco bits sirven para operaciones en las que se necesite desplazamiento, en caso de necesitarlo se especifica ahí y por último, los 6 bits restantes indican la operación exacta que se realizará, en este caso la suma.

CPI de cada instrucción.

El CPI de las operaciones usadas es el siguiente:

- dat : cpi 3
- sum : cpi 1
- mul : cpi 2

Decidimos que dat tenga un CPI de 3 ya que las operaciones de carga suelen ser más costosas, esto se debe principalmente a que, a diferencia de las operaciones aritméticas que trabajan a partir de los datos de los registros de la cpu, estas operaciones generalmente necesitan acceder a la memoria para encontrar los datos que se desean cargar.

Ancho del bus de datos.
Los buses de datos tendrán un tamaño de 32 bits (4 bytes)

Evaluación de desempeño.

Un programa que resuelve la operación (A+B) * C + D utilizando el ISA propuesto anteriormente se vería así:

dat r0 = A;
dat r1 = B;
dat r2 = C;
dat r3 = D;
sum r4 -> r0, r1;
mul r4 -> r4, r2;
sum r4 -> r4, r3;

dirección	código	instrucción
100	100011 00100 01000 0000000000000000	dat r0 = A;

dirección	código	instrucción
101	100011 00101 01001 0000000000000000	dat r1 = B;
102	100011 00111 01010 0000000000000000	dat r2 = C;
103	100011 01000 01011 0000000000000000	dat r3 = D;
104	000000 00100 00101 01001 00000 000000	sum r4 -> r0, r1;
104	100100 01001 00111 01001 00000 000000	mul r4 -> r4, r2;
104	000000 01001 01000 01001 00000 000000	sum r4 -> r4, r3;

Con el CPI escogido para cada instrucción, la CPU necesitará un total de 16 ciclos de clock para completar la ejecución del programa. El CPI promedio es de 2,125.

Para calcular el tiempo de ejecución del programa, utilizamos la fórmula $t = (Nc)/F$. Donde Nc es la cantidad de ciclos que necesita la CPU para ejecutar el programa y F es la frecuencia del clock. En este caso, consideramos que la frecuencia del clock es de 1Mhz (0.001 ghz)

$$t = 16/0,001 * 10^9 = 16/10^{-3} * 10^9 = 16/10^6 = 0,000016$$

Osea que la CPU tarda 0,000016 segundos o 16 microsegundos en ejecutar el programa.

Observaciones finales

En resumen, este informe destacó la importancia de comprender las arquitecturas informáticas. Desde la Unidad Aritmética Lógica (ALU) hasta la Arquitectura del Conjunto de Instrucciones (ISA), examinamos cómo estos componentes afectan el rendimiento y la eficiencia de los sistemas informáticos. Al evaluar críticamente las características de diseño de una CPU, fortalecemos nuestra comprensión de cómo se diseñan y operan las computadoras en la actualidad.