La performance depende de 3 caracteristicas:

* Clock rate
* CPI
* Cantidad de Instrucciones

3 tipos de instrucciones:

Que tienen en común?

PC 🡪 dirección de la instrucción, la ejecuto y luego sumo 4 lugares y ejecuto esa y asi

R-type:

Señales de control: información que sale del binario de las instrucciones, cuando descifro que instrucción es, estas señales me dicen que hacer en el procesador

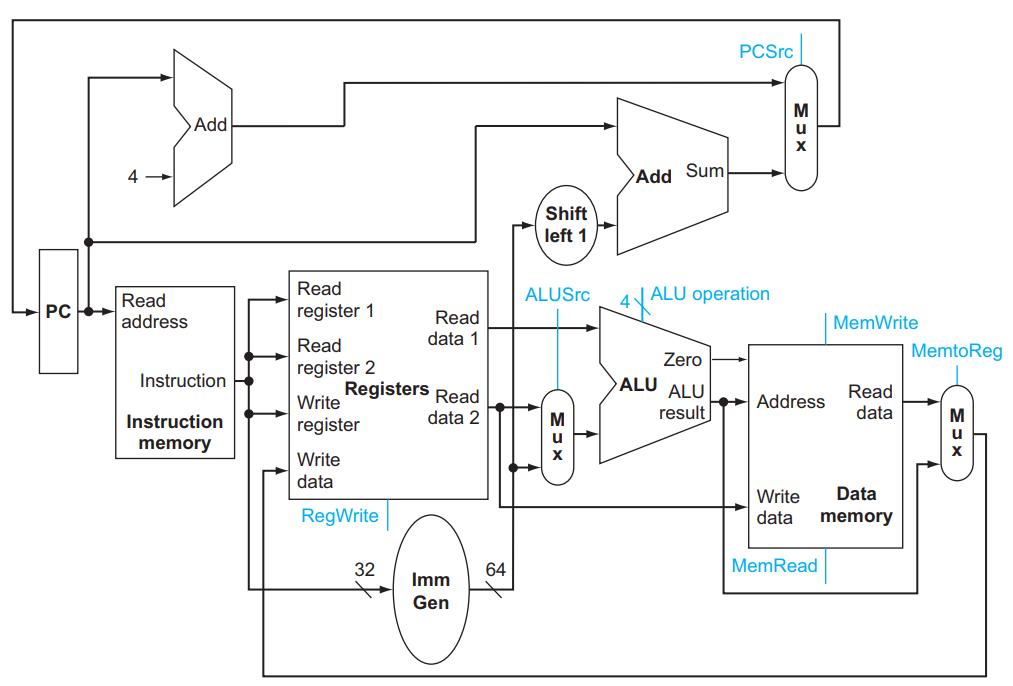
Zero: sirve para los condicionales

Load-Store:

Branch:

2 ALUs, una para evaluar la condición y otra para ver a donde salto

El offset es desde la instrucción misma de salto



PCSrc es la única señal que no viene de la instrucción, lo que necesitamos saber aca es si hay un salto u . Lo vamos a tener que sacar del opcode

RegWrite:

ALUSrc: esta en 0 cuando es inmediato

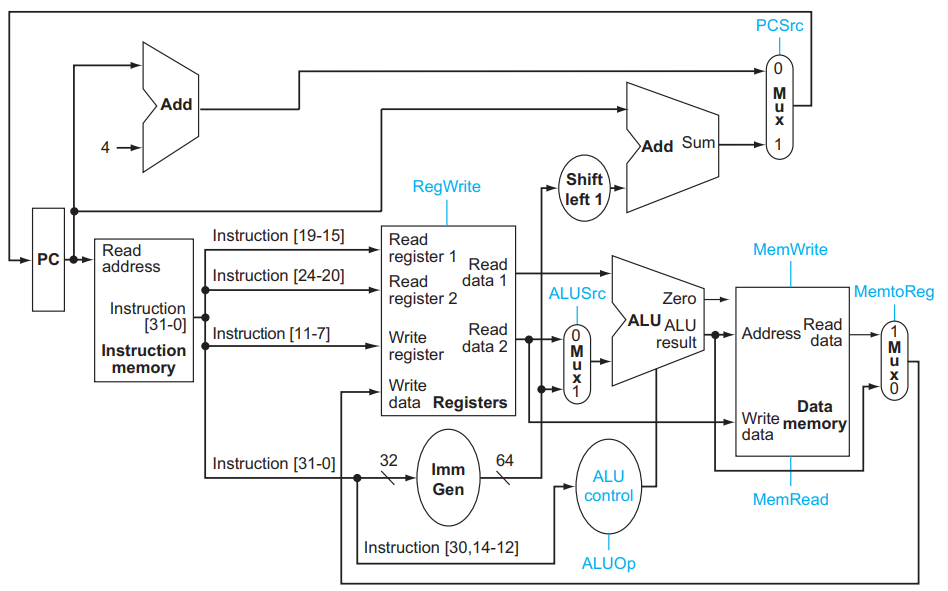
PCSrc: O la dirección de salto o el PC que le sume 4

MemtoReg: el valor que saque de memoria

ALU control:

Vamos a decir que solo tenemos 4 operaciones aritmético-logicas asi que le asignamos un binario a cada una

En el load-store y en el Branch tengo que calcular la dirección



Implementación Multiciclo:

Hasta ahora teniamos muchos secuenciales, memorias y combinacionales que me estaban determinando un T el cual me decia la frecuencia maxima a la cual puedo exigir el procesador para que PC no se Adelante

Vamos a agrear latches para dividir en etapas y mi frecuencia maxima va a ser 1/T donde ese T es el tiempo mas grande de una de esas etapas la cual es la de registers

1. Instruction Fetch
   1. Comun para todas las instrucciones
2. Instruction decode and register fetch
   1. Cargo el rs1 y el rs2
   2. Calculo la direccion de salto en caso de que sea un branch
3. Execution, mem
   1. Load-store: calculo el offset con el inmediato y la base del registro
   2. Arimetica logica:
   3. Branch: comparo los registros para ver si es zero
4. Mem access or R-type completion
   1. Load-store: debo escribir en el registro lo que busque en memoria
   2. Arimetica logica: debo escribir el registro con el resultado
5. Mem read completion
   1. Solo caso del load: valor que traje de memoria lo cargo en el registro

CPI = 4.1

Cada programa en promedio va a tardar esos ciclos por instruccion

En el monociclo el CPI es 1 pero cada ciclo es mas largo al tener la frecuencia mas baja

Pipeline el objetivo es implementar un multiciclo pero sacando una instruccion por ciclo, ya que al tener etapas

Este multiciclo es un oaso teorico intermedio para llegar a la realidad que es el pipeline