Computer Architecture

Arquitectura de Computadores

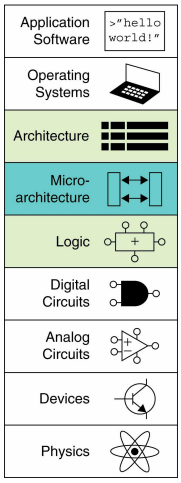
MicroArquitectura

Basado en texto: "***Digital Design and Computer Architecture*, 2nd Edition**", David Money Harris and Sarah L. Harris

Computer Architecture

Tópicos

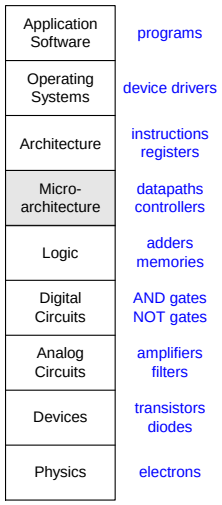
- **Introducción**
- **Análisis de Rendimiento**
- **Procesador Mono-Ciclo**
- **Procesador Multi-Ciclo**
- **Procesador con Pipeline**
- **MicroArquitectura Avanzada**



Computer Architecture

Introducción

- **Micro-arquitectura:** como implementar una arquitectura en hardware
- Procesador:
  - **Camino de Datos (Datapath):** bloques funcionales
  - **Control:** señales de control



Computer Architecture

Microarquitectura

- Múltiples implementaciones de una misma arquitectura:
  - **Mono-ciclo:** Cada instrucciones se ejecuta en un solo ciclo
  - **Multi-ciclo:** Cada instrucción es dividida en una serie de pasos mas cortos
  - **Con Pipeline:** Cada instrucción es dividida en una serie de pasos & múltiples instrucciones se ejecutan a la vez

Procesador MIPS

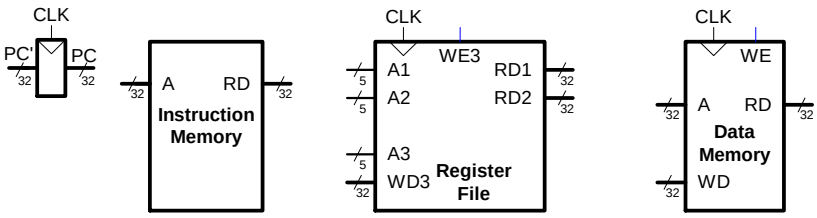
- Considere un subconjunto de las instrucciones MIPS :
  - Instrucciones tipo R: and, or, add, sub, slt
  - Instrucciones de Memoria: lw, sw
  - Instrucciones de Bifurcación: beq

Estado de la Arquitectura

- Determina todo acerca del procesador:
  - PC
  - 32 registros
  - Memoria

Computer Architecture

Elementos de Estado de MIPS



Computer Architecture

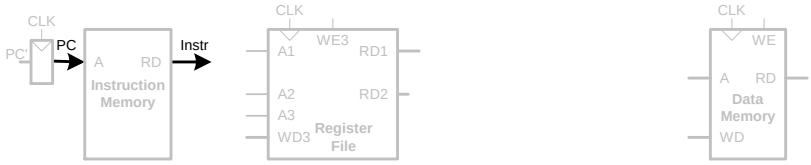
Procesador MIPS Mono-ciclo

- ¿Como diseñar el camino de Datos?
- ¿Como se diseña la Unidad de Control?

Computer Architecture

Camino de Datos Mono-ciclo: lw fetch

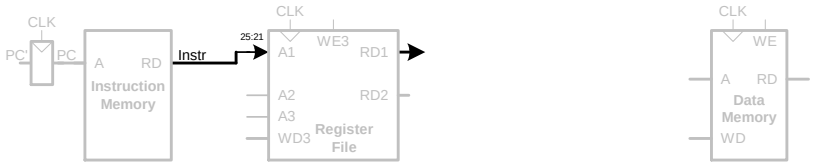
PASO 1: instrucción fetch (ir a buscar)



Computer Architecture

Camino de Datos Mono-ciclo: lw Lectura Registro

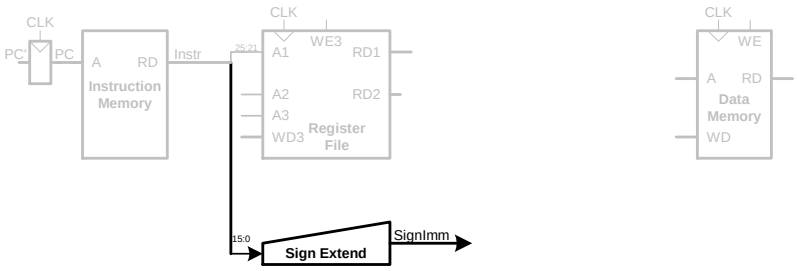
PASO 2: Leer operandos fuentes de RF (Archivo de Registros)



Computer Architecture

Camino de datos mono-ciclo: lw Inmediato

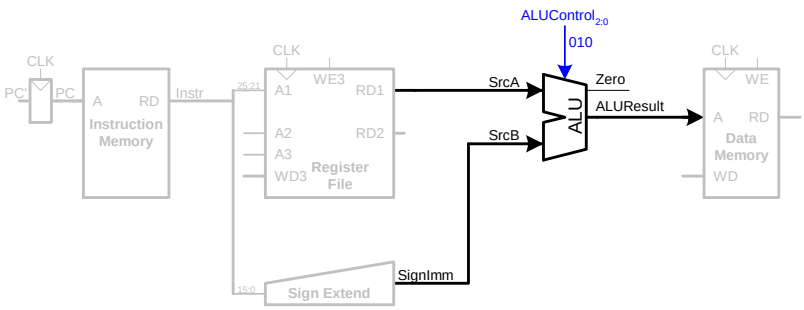
PASO 3: Extender signo del inmediato



Computer Architecture

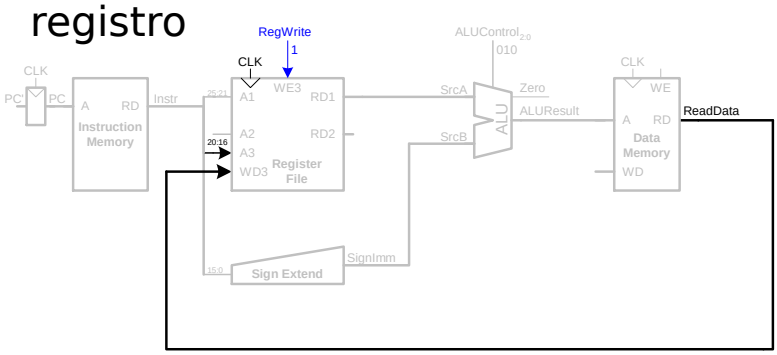
Camino de datos mono-ciclo: lw dirección

PASO 4: Calcular dirección de memoria



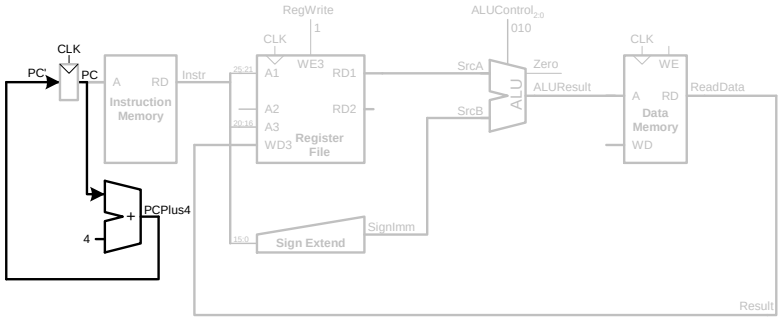
Camino de datos mono-ciclo: lw Leer Memoria

- **PASO 5:** Leer datos de memoria para luego escribirlos en un registro



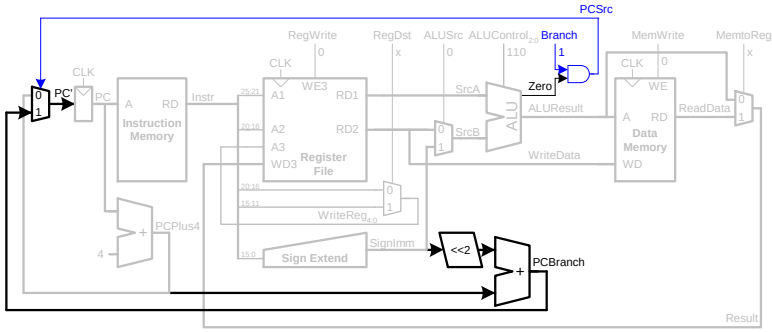
Camino de datos mono-ciclo: lw Incremento PC

PASO 6: Determine dirección de la próxima instrucción

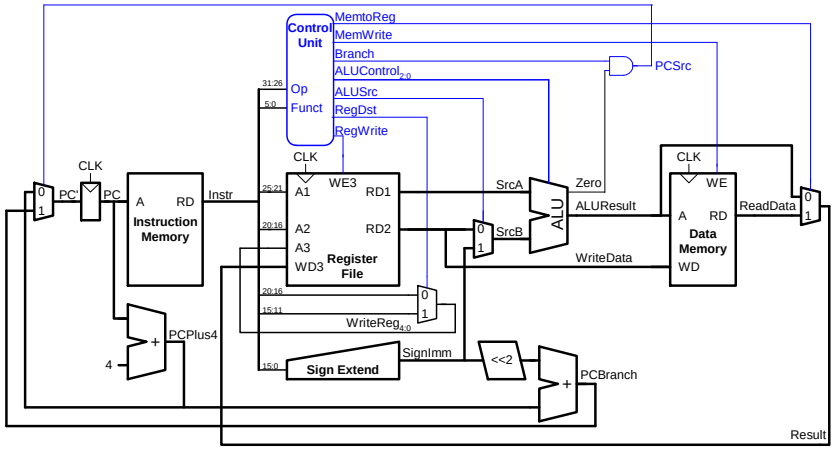


Camino de datos mono-ciclo: beq

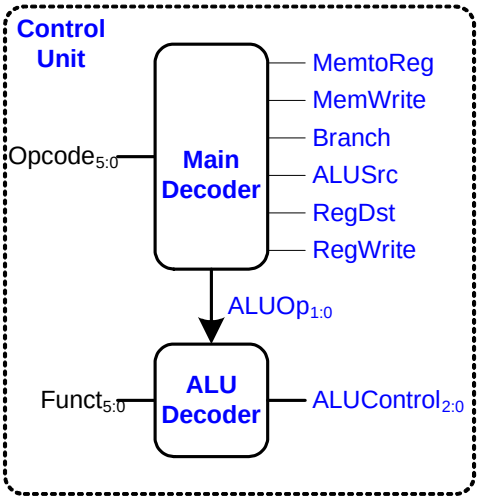
- Determine si valor en rs y rt son iguales
- Calcular dirección del salto al bifurcar:
  BTA = (inmediato extendido en signo << 2) + (PC+4)



Procesador Mono-ciclo

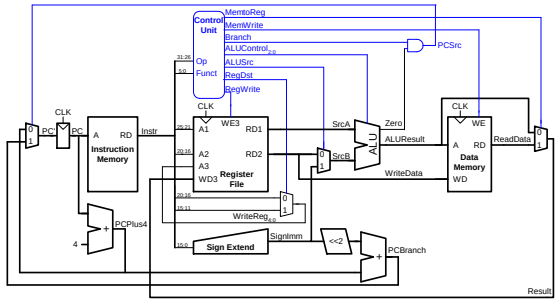


Control Mono-ciclo



Decoder Principal Unidad de Control

Instrucción	$Op_{5:0}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
Tipo R	000000							
lw	100011							
sw	101011							
beq	000100							



Computer Architecture

Unidad de Control: Decoder Principal

Instrucción	$Op_{5:0}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
R-type	000000	1	1	0	0	0	0	10
lw	100011	1	0	1	0	0	0	00
sw	101011	0	X	1	0	1	X	00
beq	000100	0	X	0	1	0	X	01

Computer Architecture

Ejemplo rendimiento mono-ciclo

Elemento	Parámetro	Retardo (ps)
Registro reloj-a-Q	t_{pcq_PC}	30
Setup Registro	t_{setup}	20
Multiplexor	t_{mux}	25
ALU	t_{ALU}	200
Leer memoria	t_{mem}	250
Leer arch. registros	t_{RFread}	150
Setup arch. registros	$t_{RFsetup}$	20

$T_c = t_{pcq\_PC} + 2t_{mem} + t_{RFread} + t_{mux} + t_{ALU} + t_{RFsetup}$

= [30 + 2(250) + 150 + 25 + 200 + 20] ps

= 925 ps

Computer Architecture

Ejemplo Rendimiento monociclo

Programa con 100 mil millones de instrucciones :

Tiempo Ejecución = # instrucciones x CPI x T_C

= $(100 \times 10^9)(1)(925 \times 10^{-12}$ s)

= **92,5 segundos**

Computer Architecture

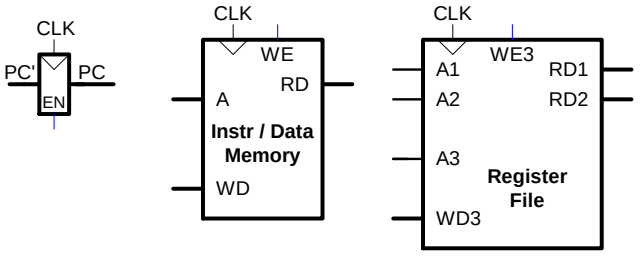
Procesador MIPS Multiciclo

- **Mono ciclo:**
  - + simple
  - - Tiempo de ciclo limitado por la instrucción mas larga (lw)
  - - 2 sumadores/ALUs & 2 memorias
- **Multiciclo:**
  - + velocidad del reloj es mas alta
  - +Instrucciones mas simples se ejecutan mas rápido
  - + reuso de hardware caro en múltiples ciclos
  - - el sobre-costo de secuenciar es pagado múltiples veces
- **Los mismos pasos de diseño: camino de datos & control**

Computer Architecture

Elementos de Estado Multiciclo

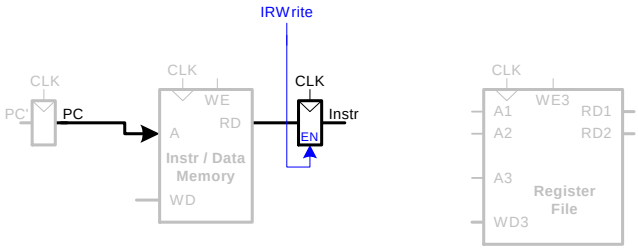
- Reemplaza la memoria de datos e instrucciones con una sola memoria unificada – es mas real



Computer Architecture

Camino de Datos Multiciclo: Instrucción Fetch

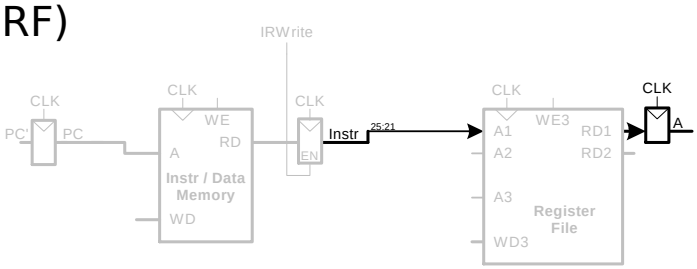
PASO 1: Instrucción Fetch



Computer Architecture

Camino de Datos Multiciclo: lw Leer R

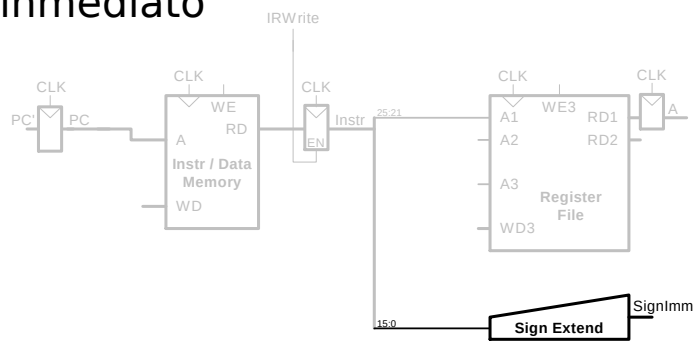
PASO 2a: Lectura de operandos fuente desde archivo de registros (RF)



Computer Architecture

Camino de Datos Multiciclo: lw Inmediato

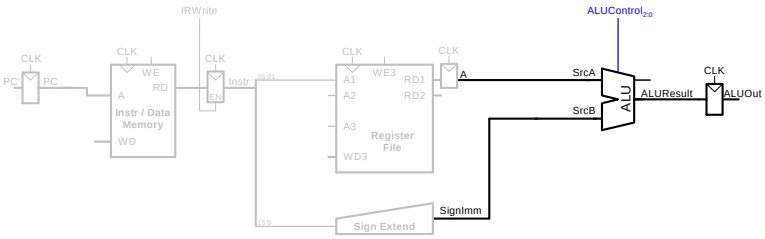
STEP 2b: Extender en signo el inmediato



Computer Architecture

Camino de Datos Multiciclo: lw dirección

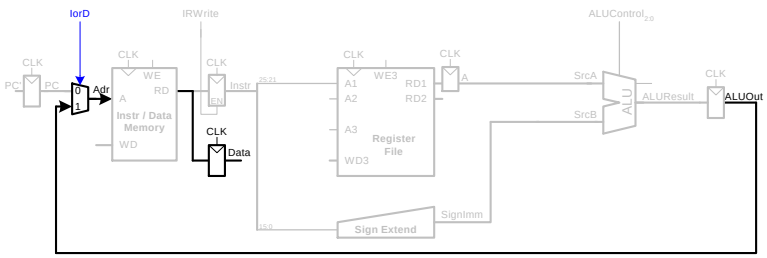
PASO 3: Calcular dirección de memoria



Computer Architecture

Camino de Datos Multiciclo: lw Leer Memoria

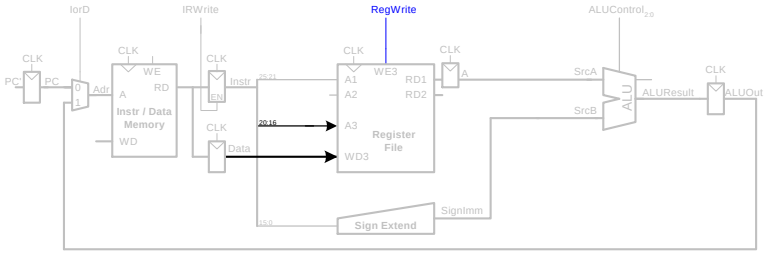
Paso 4: Leer dato desde memoria



Computer Architecture

Camino de Datos Multiciclo: lw Escribir Registro

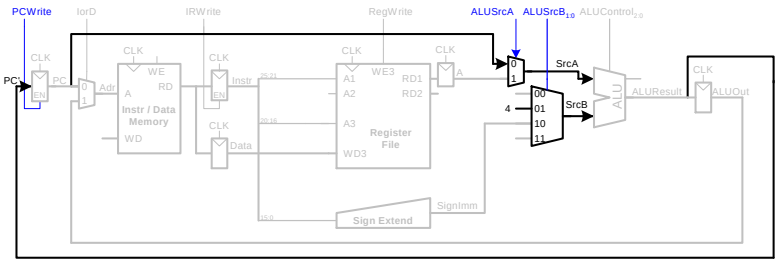
PASO 5: Escribir datos al archivo de registros



Computer Architecture

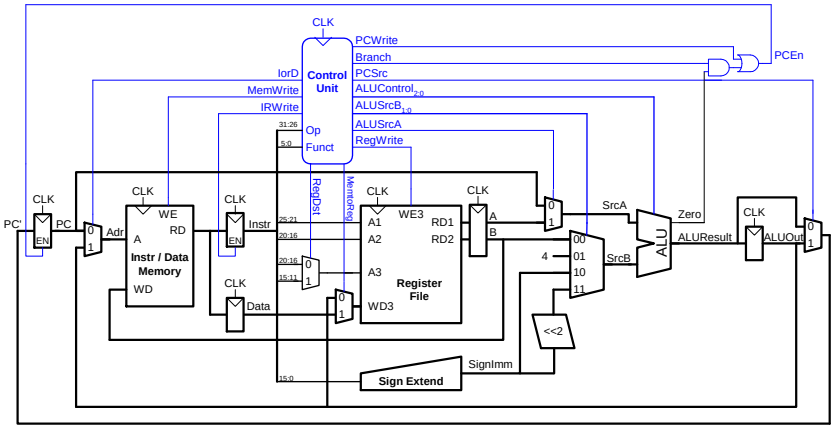
Camino de Datos Multiciclo: Incrementar PC

PASO 6: Incrementar PC



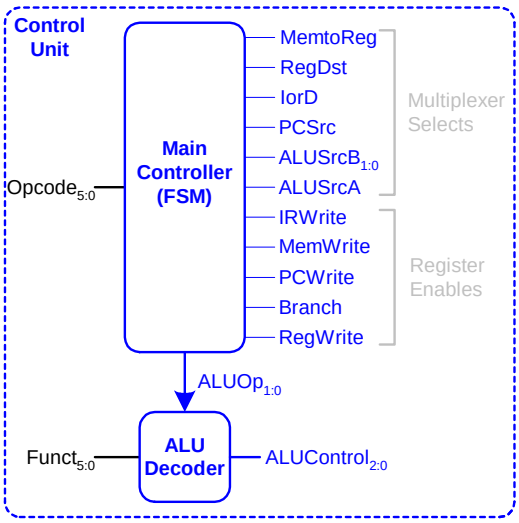
Computer Architecture

Procesador Multiciclo



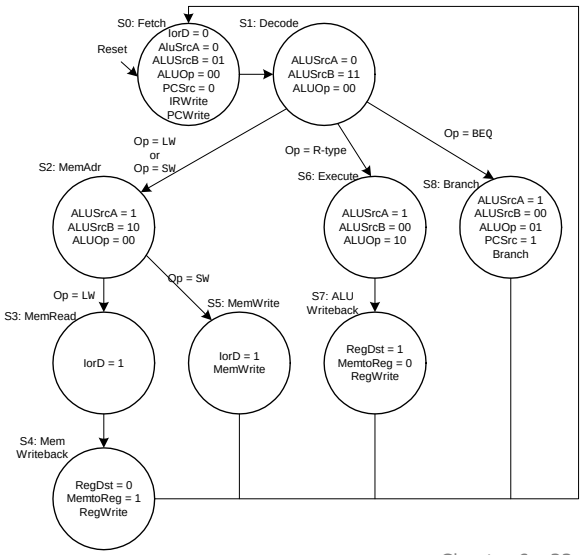
Computer Architecture

Control Multiciclo



Computer Architecture

MEF del Controlador Multiciclo



Computer Architecture

Rendimiento del Procesador Multiciclo

- Las instrucciones requieren distintos números de ciclo para su ejecución:
  - 3 ciclos: beq, j
  - 4 ciclos: Tipo R, sw, addi
  - 5 ciclos: lw
- CPI es el promedio ponderado
- El benchmark SPECINT2000:
  - 25% de lecturas de memoria (loads)
  - 10% de escritura de memoria (stores)
  - 11% de saltos de rama (branches)
  - 2% de saltos incondicionales (jumps)
  - 52% de Tipo R

CPI promedio = (0.11 + 0.02)(3) + (0.52 + 0.10)(4) + (0.25)(5) **= 4.12**

Computer Architecture

Ejemplo Rendimiento Multiciclo

Elemento	Parametro	Retardo(ps)
Registro reloj-a-Q	t_{pcq_PC}	30
Setup Registro	t_{setup}	20
Multiplexor	t_{mux}	25
ALU	t_{ALU}	200
Lectura Memoria	t_{mem}	250
Lectura Arch. Registro	t_{RFread}	150
Setup Arch. Registro	$t_{RFsetup}$	20

$$T_c = t_{pcq\_PC} + t_{mux} + \max(t_{ALU} + t_{mux}, t_{mem}) + t_{setup}$$
$$= t_{pcq\_PC} + t_{mux} + t_{mem} + t_{setup}$$
$$= [30 + 25 + 250 + 20]\ \text{ps}$$
$$= \mathbf{325\ ps}$$

Computer Architecture

Ejemplo Rendimiento Multiciclo

Programa con 100 mil millones de instrucciones

Tiempo de Ejecución = (# instrucciones) × CPI × T_c

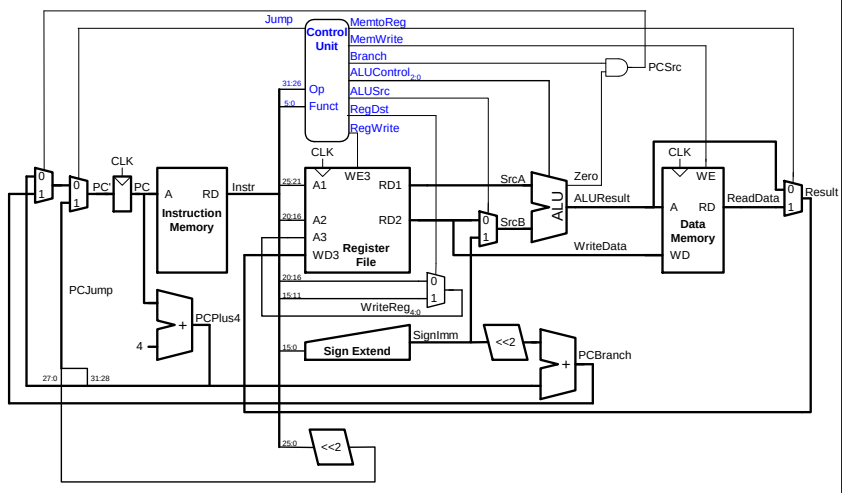
$$= (100 \times 10^9)(4.12)(325 \times 10^{-12})$$
$$= \mathbf{133.9\ segundos}$$

Este es **mas lento** que el procesador monociclo (92.5 segundos). ¿Por que?

- No todos los pasos son del mismo largo
- Sobrecosto por secuenciar en cada paso (t_{pcq} + t_{setup}= 50 ps)

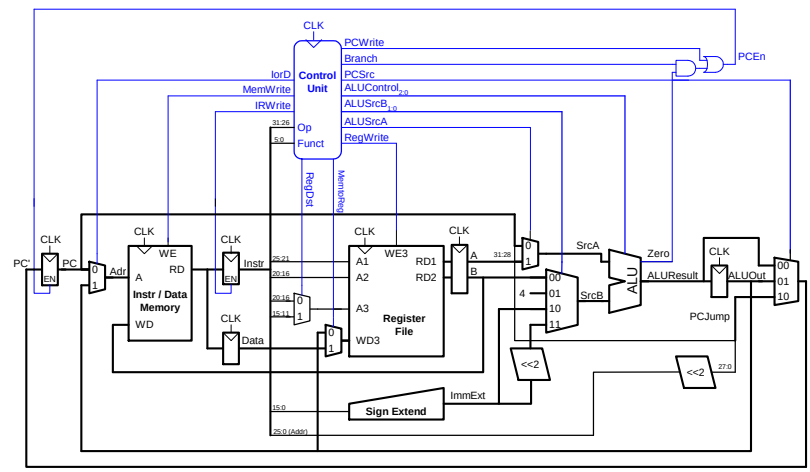
Computer Architecture

Repaso: Procesador Mono-ciclo



Computer Architecture

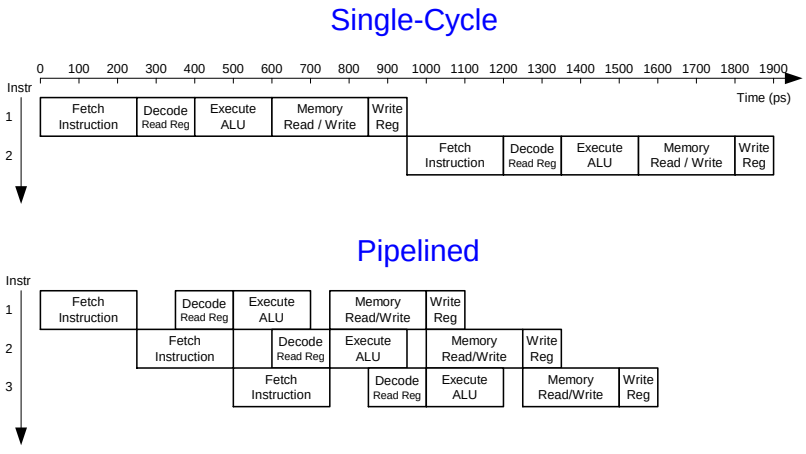
Repaso: Procesador Multiciclo



Procesador MIPS con Pipeline

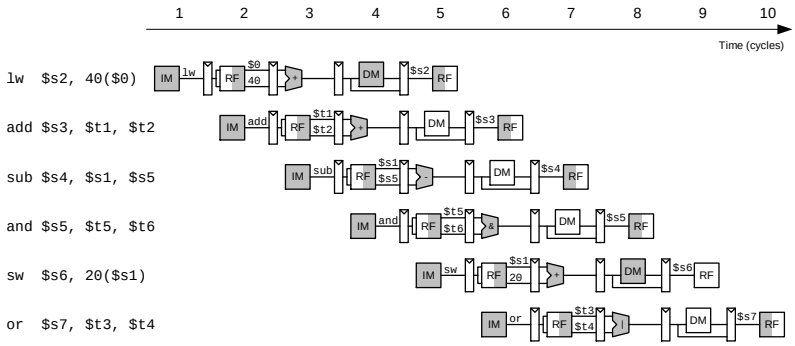
- Paralelismo temporal
- Divida procesador monociclo en 5 etapas:
  - Fetch
  - Decodificar (Decode)
  - Ejecutar (Execute)
  - Memoria (Memory)
  - Escritura en Registro (Writeback)
- Agregue registros de pipeline entre etapas

Monociclo vs. Pipeline



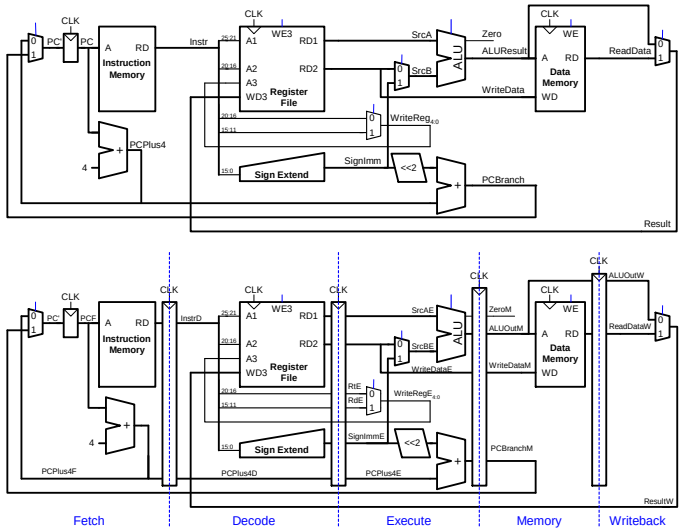
Computer Architecture

Abstracción del Procesador con Pipeline



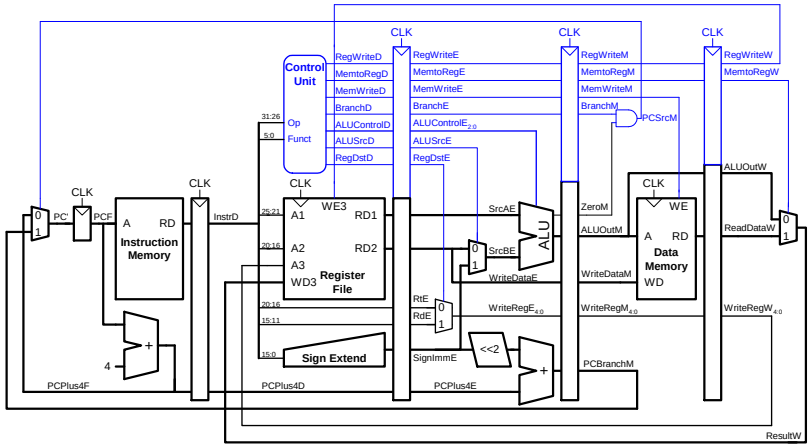
Computer Architecture

Camino de Datos Mono-ciclo & con pipeline



Computer Architecture

Control del Procesador con Pipeline



- **Es igual a la unidad de control del procesador mono ciclo**
- **El Control es retardado de acuerdo a cada etapa del pipeline**

Computer Architecture

Predicción de Rama

- Adivina si un salto de rama sera tomado
  - Los saltos hacia atrás son preferentemente tomados (ciclos)
  - Considere historia para mejorar acierto
- ¿Que pasa si la CPU se equivoca al predecir salto?
- Una buena predicción reduce la fracción de saltos que se requiere eliminar

Comparación Rendimiento Procesadores

Procesador	Tiempo de Ejecución (segundos)	Mejora (Speedup) (Monociclo es la base)
Monociclo	92.5	1
Multiciclo	133	0.70
Con Pipeline	63	1.47

Micro arquitectura

- Pipeline Profundo/Deep Pipelining
- Predicción de Rama
- Procesadores superescalares
- Procesadores Fuera de Orden
- Renombre de registros
- SIMD
- Multihebras/Multithreading
- Multiprocesadores

Pipeline Profundo

- Usualmente con 10-20 etapas
- El numero de etapas esta limitado por:
  - Riesgos de Pipeline
  - Sobrecosto por secuenciar
  - Potencia eléctrica
  - Costo

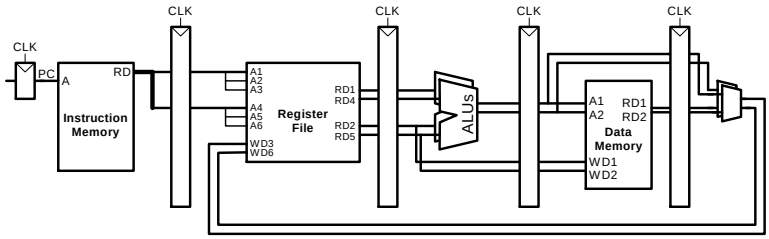
Predicción de Rama

- Procesador con pipeline ideal: CPI = 1
- Mala predicción de salto de rama incrementa CPI
- **Predicción de salto de rama estático:**
  - Chequea dirección del salto (hacia adelante o hacia atrás)
  - Si es hacia atrás, la predicción es considerada
  - Sino, la predicción no es tomada
- **Predicción de salto de rama dinámico:**
  - Guarde la historia de los últimos (varios cientos) saltos de rama en un *branch target buffer*, con los siguientes datos:
    - Destino del salto de rama

Computer Architecture

Superescalar

- Múltiples copias del camino de datos ejecutan múltiples instrucciones a la vez
- Las dependencias hace que sea astuta la decisión de procesar múltiples instrucciones a la vez



Computer Architecture

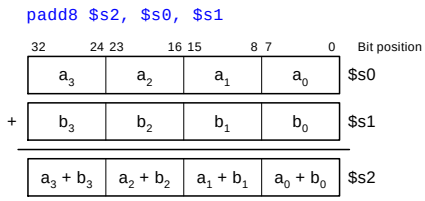
Procesador Fuera de Orden

- Mira hacia adelante a través de múltiples instrucciones
- Procesa tantas instrucciones como sea posible a la vez
- Procesa aquellas instrucciones fuera de orden (siempre y cuando no tengan dependencias)
- **Dependencias:**
  - **RAW** (read after write): una instrucción escribe, luego la instrucción lee un registro
  - **WAR** (write after read): una instrucción lee, luego la instrucción escribe en un registro
  - **WAW** (write after write): una instrucción escribe, luego la instrucción escribe en un registro

Computer Architecture

SIMD

- Single Instruction Multiple Data (SIMD)
  - una sola instrucción actúa sobre múltiples piezas de datos a la vez
  - La aplicación típica: computación gráfica
  - Realiza operaciones aritméticas cortas (a esto también se le llama *aritmética empaquetada*)
- Por ejemplo, sume cuatro elementos de 8-bit



Computer Architecture

Técnicas de Arquitecturas

- **Multihebras/Multithreading**
  - Procesador de palabras: una hebra controla el tipeo, otra el chequeo ortografico, otra la impresión
- **Multiprocesadores**
  - Múltiples procesadores (núcleos/cores) en un único chip

Computer Architecture

Hebras: Definiciones

- **Proceso:** programa que se ejecuta en un computador
  - Múltiples procesos pueden ser ejecutados a la vez: e.g., navegar en la Web, reproducir una canción, escribir un informe
- **Hebra/Thread:** parte de un programa
  - Cada proceso tiene múltiples hebras: e.g., un procesador de palabras podría tener hebras para controlar el tipeo, el chequeo ortográfico, la impresión

Computer Architecture

Hebras en un Procesador Convencional

- Una hebra se ejecuta solo una vez
- Cuando una hebra se detiene (por ejemplo, esperando memoria):
  - Se almacena el estado arquitectural de la hebra
  - El estado arquitectura de la hebra en espera se carga en el procesador y esta vuelve a ser ejecutada
  - A esto se le llama **cambio de contexto /context switching**
- Al usuario le pareciera que todas las hebras se estuvieran ejecutando simultáneamente

Computer Architecture

Multiprocesadores

- Hay múltiples procesadores (núcleos/cores) que posee un metodo de comunicación entre ellos
- Tipos:
  - **Homogéneos**: múltiples núcleos con memoria compartida
  - **Heterogéneos:** núcleos separados para diferentes tareas (por ejemplo, DSP y CPU en un teléfono celular)
  - **Clusters:** cada núcleo tiene su propio sistema de memoria

Computer Architecture

Otros recursos

- Patterson & Hennessy's: *Computer Architecture: A Quantitative Approach*
- Conferencias:
  - www.cs.wisc.edu/~arch/www/
  - ISCA (International Symposium on Computer Architecture)
  - HPCA (International Symposium on High Performance Computer Architecture)