

Arquitecturas y Organización de Computadoras I

4: Microarquitectura/Organización: Segmentación (paralelismo a nivel de instrucciones)

Rafael Ignacio Zurita

Depto. Ingeniería de Computadoras

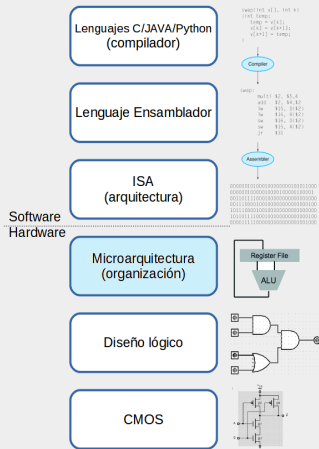
October 19, 2020

Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Temario

Microarquitectura segmentada (pipelining)



Concepto de Paralelismo a nivel de instrucciones

Mejora la productividad (no la latencia)

Se maximiza el uso de las unidades funcionales

Mejora el ciclo de reloj (frecuencia de reloj)

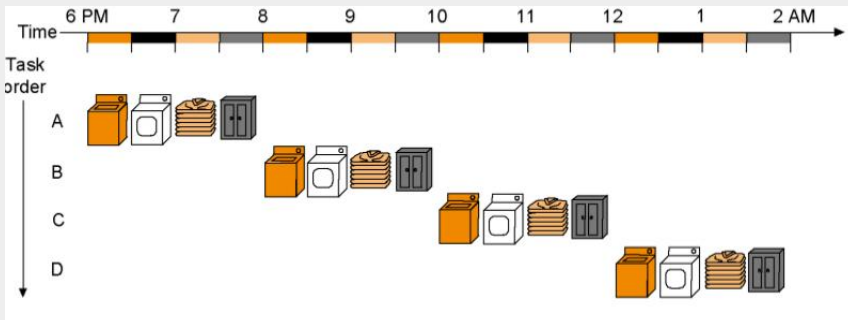
Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Implementación de la Microarquitectura

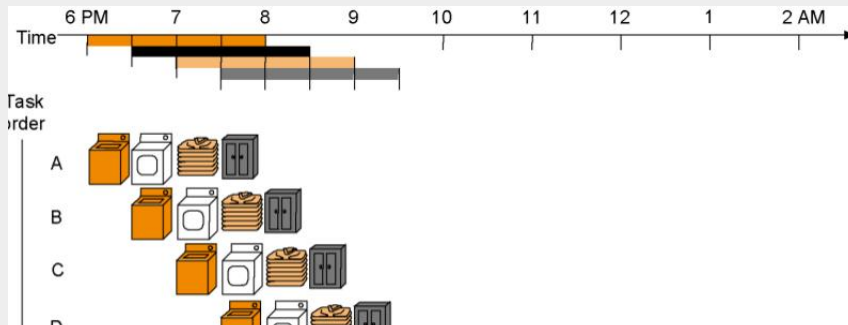
Ejemplo de la lavandería

- * Problema de trabajo
 - * Cuatro etapas de tareas
 - * Tiempos balanceados (aprox. 30 min.)
 - * Secuencia fija de pasos
 - * Tiempo total para n trabajos = $(n * 2\text{hs.})$



Ejemplo de la lavandería

- * Las unidades operan independientemente
- * Se pueden utilizar unidades al mismo tiempo
- * Tiempo promedio por lavado similar al anterior
- * Tiempo total para n cargas = aprox. ($n \times$ tiempo por etapa)



Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Implementación de la Microarquitectura

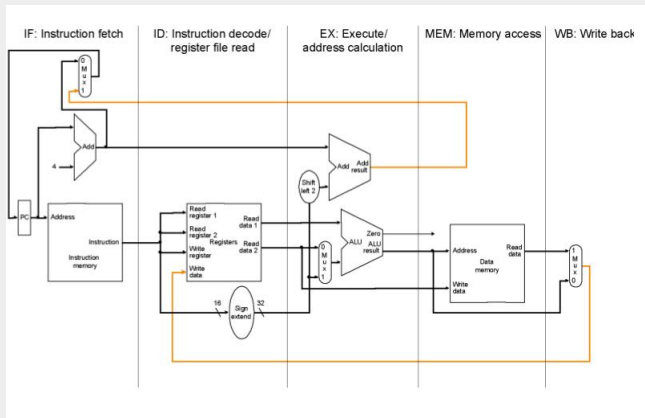
Microarquitectura segmentada

- * Uso de varias unidades al mismo
- * Compartir elementos entre diferentes instrucciones
- * **Paralelismo a nivel de instrucciones**

Precondiciones

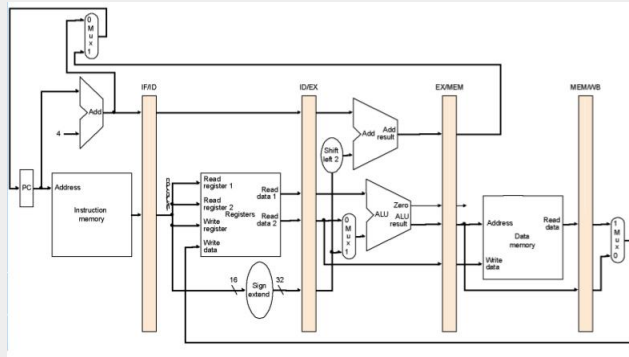
- * Diseño del conjunto de instrucciones
 - * Instrucciones de ancho fijo (idealmente)
 - * Pocos formatos de instrucciones
 - * Operandos en memoria sólo para instrucciones de carga y almacenamiento
 - * Datos alineados
- * Origen de los problemas
 - * Instrucciones con longitud variable
 - * Datos no alineados

Camino de datos de un ciclo



» Implementación de la Microarquitectura

Camino de datos segmentado



Camino de datos segmentado

- * Idea clave: aumentar la productividad (no la latencia)
- * Hardware mas complejo: registros de la microarquitectura
- * Posibilita un incremento de la frecuencia del reloj

Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * **Mejora del rendimiento**
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

Tiempo de ejecución

The top diagram illustrates sequential execution. The timeline shows three instructions: `lw $1, 100($0)`, `lw $2, 200($0)`, and `lw $3, 300($0)`. Each instruction takes 8 ns to complete. The second instruction starts only after the first has finished, and the third starts after the second. This results in a total execution time of 24 ns for three instructions.

The bottom diagram illustrates a 5-stage pipeline. The stages are: Instruction fetch, Reg, ALU, Data access, and Reg. Each stage takes 2 ns. The instructions are overlapped in time. The first instruction starts at time 0 and finishes at 10 ns. The second instruction starts at 2 ns and finishes at 8 ns. The third instruction starts at 4 ns and finishes at 6 ns. The pipeline achieves a throughput of one instruction every 2 ns, completing three instructions in 6 ns.

Estas fórmulas son especialmente útiles porque distinguen los tres factores claves que influyen en las prestaciones. Estas fórmulas se pueden utilizar para comparar dos realizaciones diferentes o para evaluar un diseño alternativo si se conoce el impacto en estos tres parámetros.

Componentes de las prestaciones	Unidades de medida
Tiempo de ejecución de CPU de un programa	Segundos por programa
Número de instrucciones	Número de instrucciones ejecutadas por el programa
Ciclos por instrucción (CPI)	Número medio de ciclos por instrucción
Tiempo de ciclo del reloj	Segundos por ciclo de reloj

[8/35]

Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * **Ejemplo con una única instrucción: lw**
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

Etapas

-
- The diagram illustrates the internal structure of a 5-stage MIPS processor. The stages are labeled as IF/ID, ID/EX, EX/MEM, MEM/WB, and WB. The components and data flow are as follows:
- Instruction fetch (IF/ID):** The PC (Program Counter) provides an address to the instruction memory. The instruction memory outputs a 32-bit instruction. The instruction is split into a 4-bit opcode and a 28-bit instruction word. The opcode is used by the ALU to calculate the next PC value.
 - Instruction decode (ID/EX):** The instruction word is split into a 16-bit register file address and a 16-bit register file data. The register file data is split into a 16-bit register file address and a 16-bit register file data. The register file address is used to access the register file. The register file data is used to access the register file.
 - Execute (EX/MEM):** The register file data is used to access the register file. The register file data is used to access the register file. The register file data is used to access the register file.
 - Memory access (MEM/WB):** The register file data is used to access the register file. The register file data is used to access the register file. The register file data is used to access the register file.
 - Write back (WB):** The register file data is used to access the register file. The register file data is used to access the register file. The register file data is used to access the register file.
- The timing diagram at the bottom shows the sequence of these stages over time. The stages are labeled 1 through 5, corresponding to the five stages of the processor.

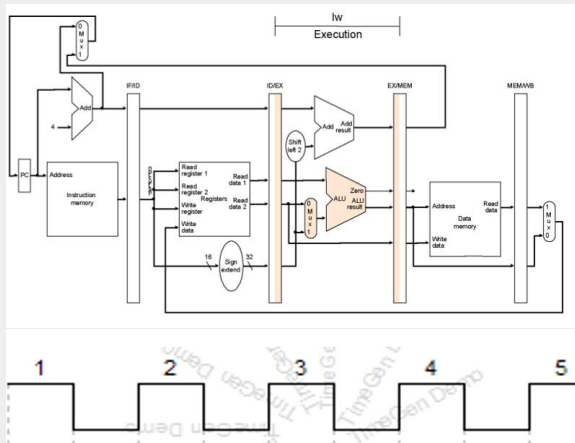
Etapas

-
- The diagram illustrates the internal structure of a 5-stage MIPS processor. The stages are:
- IF (Instruction Decode):** The PC (Program Counter) provides the address to the Instruction memory. A 4-bit adder calculates the next PC value. The Instruction memory outputs the instruction to the IF/ID register.
 - ID (Instruction Decode):** The instruction is decoded, and the register file is accessed. The register file has Read register 1, Read register 2, and Write register. The register file outputs the register data to the ID/EX register.
 - EX (Execute):** The ALU (Arithmetic Logic Unit) performs operations on the register data. The ALU has a Zero flag and a Shift left 2 operation. The ALU outputs the ALU result to the EX/MEM register.
 - MEM (Memory Access):** The ALU result is used as the address for the Data memory. The Data memory outputs the Read data to the MEM/WB register.
 - WB (Write Back):** The Read data is written back to the register file. The 16-bit multiplexer selects the data to be written back to the register file.
- The timing diagram at the bottom shows the clock signal for 5 cycles, with stages 1, 2, 3, 4, and 5 labeled above the clock pulses.

» Ejemplo con una única instrucción: lw

Etapas

- * Calculo de la dirección efectiva



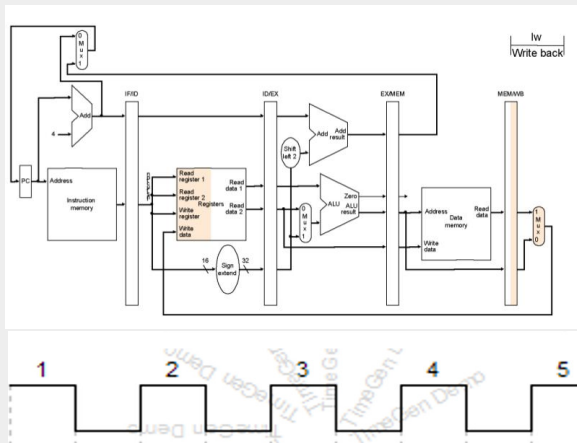
Etapas

-
- The diagram illustrates a 5-stage processor architecture. The stages are:
- IF/ID (Instruction Fetch/Decode):** The PC provides the address for Instruction memory. The 4-Mux selects between the PC and the ALU result to update the PC.
 - ID/EX (Instruction Decode/Execute):** The Register File provides Read data 1, Read data 2, and Write data. The Sign-extend unit takes a 16-bit sign-extend and produces a 32-bit result.
 - EX/MEM (Execute/Memory Access):** The ALU takes two 32-bit inputs and produces a 32-bit result. The Zero ALU result unit takes the ALU result and produces a 1-bit zero flag.
 - MEM/WB (Memory Access/Write Back):** The Data memory takes an address and produces Read data and Write data. The 1-Mux selects between the Register File and the Data memory to write back to the Register File.
- The timing diagram below shows the progression of data through the stages over five clock cycles:
- 1:** Instruction is fetched from memory.
 - 2:** Instruction is decoded, and register indices are used to read data from the Register File.
 - 3:** The ALU performs the operation on the register data, and the zero flag is calculated.
 - 4:** Data is written back to the Register File.
 - 5:** The PC is updated with the next instruction address.

» Ejemplo con una única instrucción: lw

Etapas

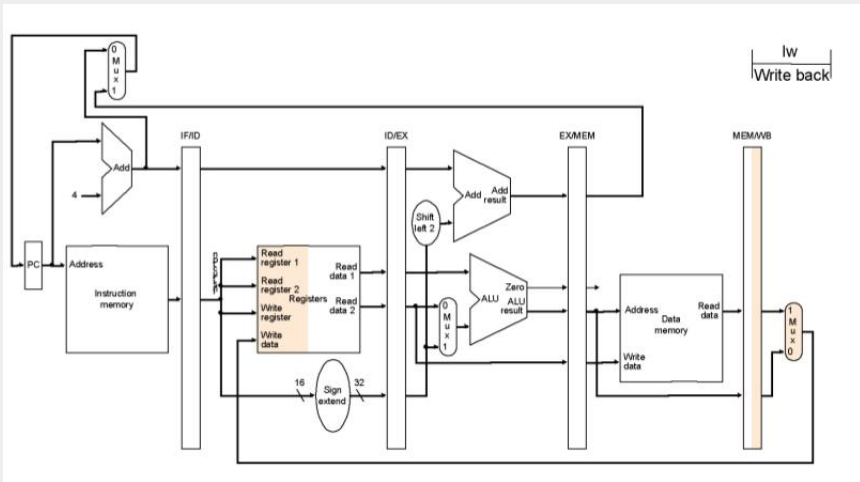
- * Escritura del dato en el archivo de registros



» Ejemplo con una única instrucción: lw

Etapas

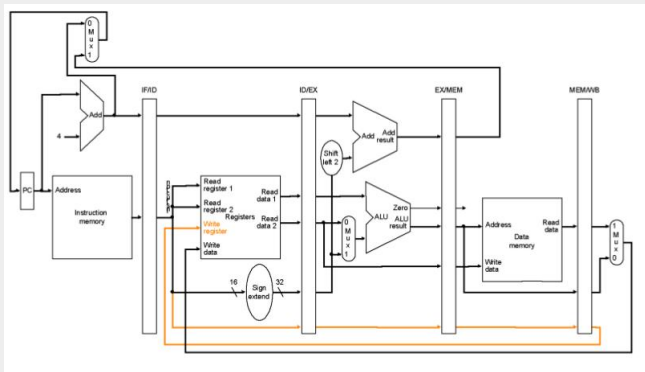
* ¿Cómo pueden llegar las señales del registro destino?



» Ejemplo con una única instrucción: lw

Etapas

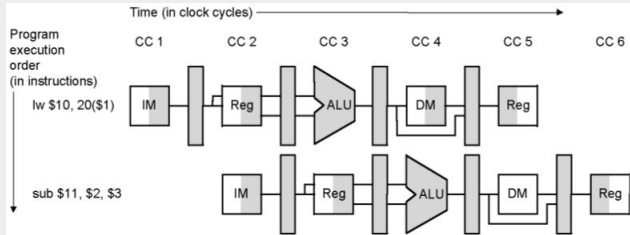
- * ¿Cómo pueden llegar las señales del registro destino?
- * Utilizando registros de la microarquitectura entre las etapas
- * Las señales son copiadas entre las etapas



» Implementación de la Microarquitectura

Representación visual del camino segmentado

- * Orientado a los recursos utilizados
- * Diagrama simplificado



♦ Example

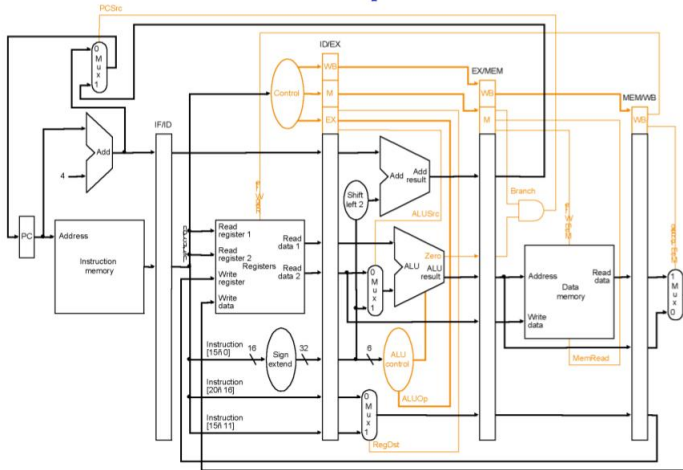
- Lw \$10, 20 (\$1)
- Sub \$11, \$2, \$3

Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * **Unidad de Control: generación de señales**
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Implementación de la Microarquitectura

Unidad de control: generación de las señales



Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Implementación de la Microarquitectura

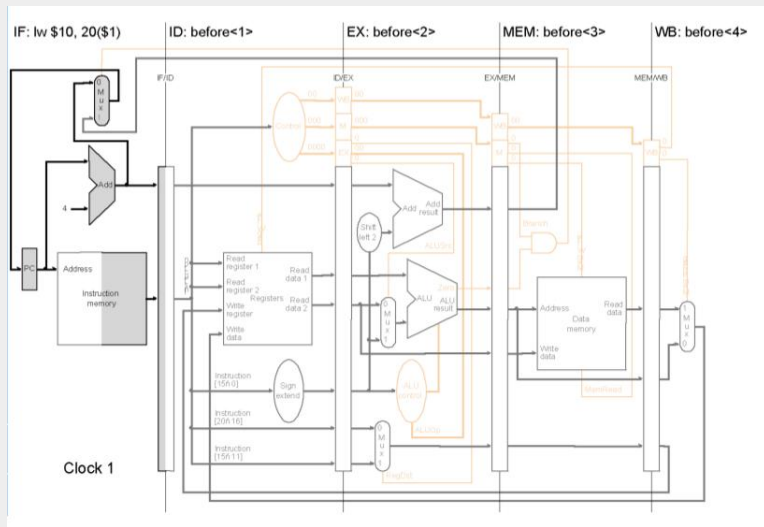
Ejemplo de una secuencia de instrucciones

- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9

» Ejemplo de una secuencia de instrucciones

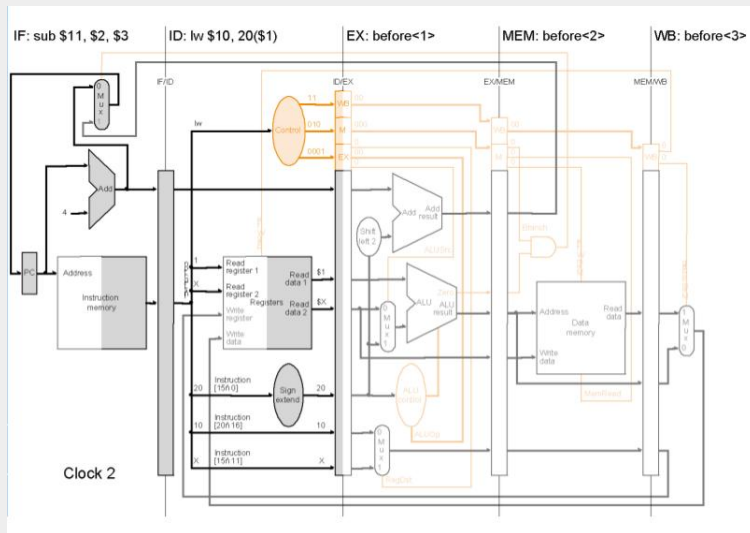
Segmentación (Pipelining)

- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



Segmentación (Pipelining)

- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



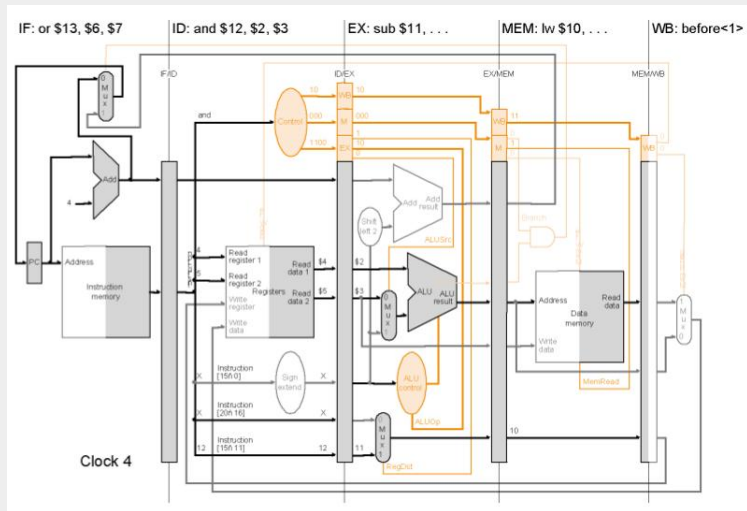
Segmentación (Pipelining)

-
- The diagram illustrates the internal state of a MIPS processor at Clock 3. The stages and their current activities are as follows:
- IF (Instruction Fetch):** The PC is 4. The instruction memory is providing the instruction 'sub \$11, \$2, \$3' to the Register File.
 - ID (Instruction Decode):** The Register File is reading data from registers \$2 and \$3. The ALU control is receiving the operation code 'sub' and the register indices '11', '2', and '3'. The ALU is performing the subtraction of \$2 from \$3, resulting in 10.
 - EX (Execute):** The ALU result (10) is being shifted left by 2 bits to produce the branch target address (40). The ALU control is also receiving the branch target address (40) and the branch predictor output (not taken).
 - MEM (Memory Access):** The Data Memory is being accessed for the first time. The address 40 is being used to calculate the effective address for the branch instruction.
 - WB (Write Back):** The register file is being updated with the ALU result (10) into register \$11.
- The diagram also shows the control signals for the Branch predictor, which is predicting 'not taken'. The ALU control is also receiving the branch target address (40) and the branch predictor output (not taken). The ALU is performing the subtraction of \$2 from \$3, resulting in 10. The ALU control is also receiving the branch target address (40) and the branch predictor output (not taken).

» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

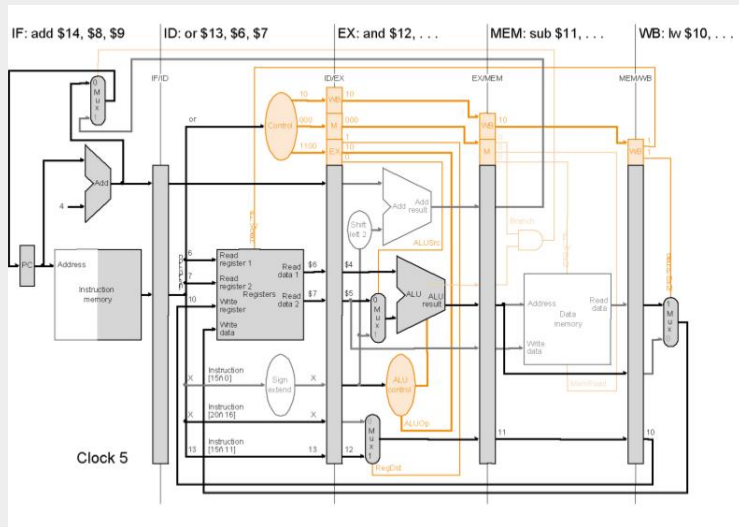
- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

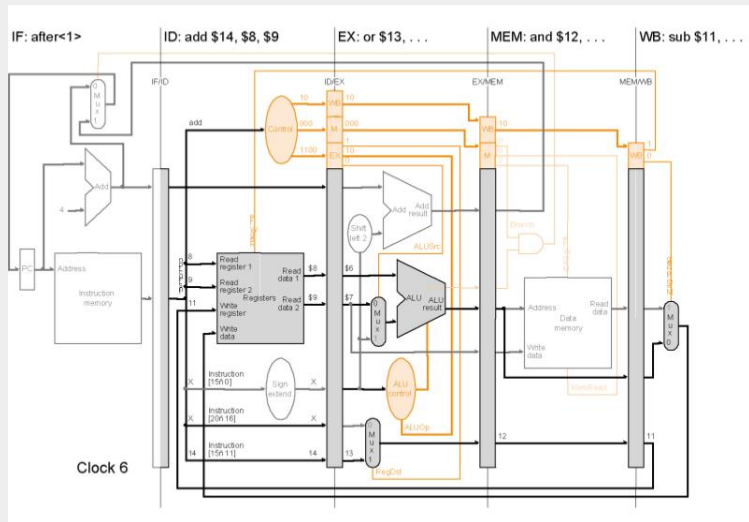
- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

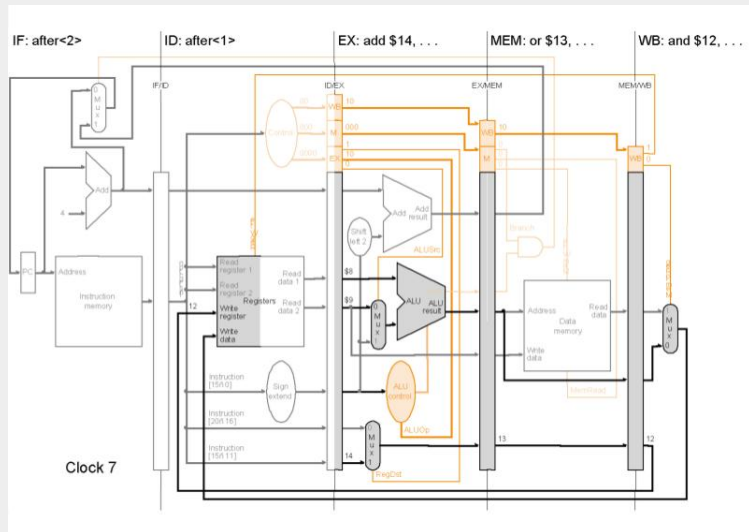
- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

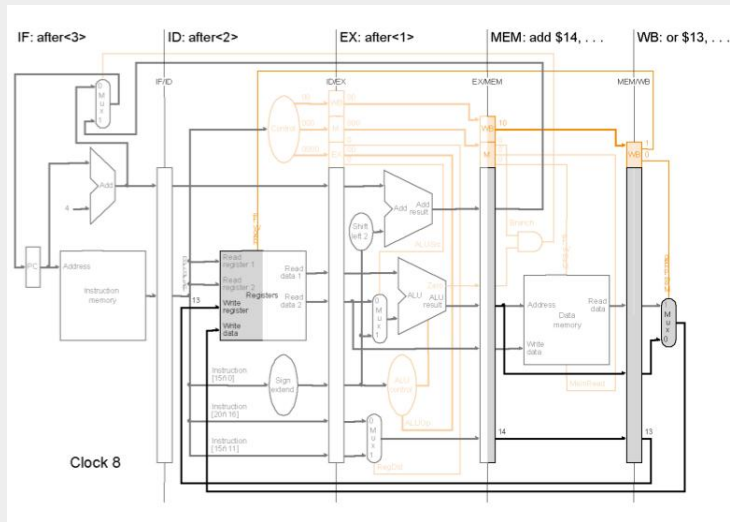
- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

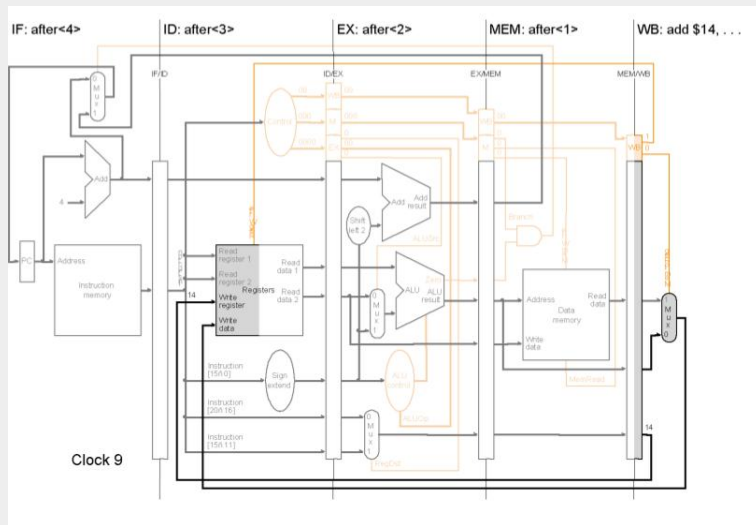
- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



» Ejemplo de una secuencia de instrucciones

Segmentación (Pipelining)

- * lw \$10, 20 (\$1)
- * sub \$11, \$2, \$3
- * and \$12, \$4, \$5
- * or \$13, \$6, \$7
- * add \$14, \$8, \$9



Segmentación (Pipelining)

- * Ejemplo de segmentación y paralelismo en una lavandería
- * Camino de datos segmentado (microarquitectura MIPS)
- * Mejora del rendimiento
- * Ejemplo con una única instrucción: lw
- * Unidad de Control: generación de señales
- * Ejemplo de una secuencia de instrucciones
- * Problemas que surgen con la segmentación (HAZARDS)

» Problemas de la segmentación (HAZARDS)

y sus soluciones o alternativas

- * Estructurales: dos o más instrucciones intentando utilizar una misma unidad
- * Datos: dependencias

Formas de tratarlos

- * Compilador
 - * Reordenar instrucciones
 - * Agregar detenciones (instrucciones nop)
- * Unidad de control
 - * Detectar el problema (hazard)
 - * Realizar una o más detenciones (stall), o
 - * Reenvío de señales entre etapas (Forwarding)

» Problemas de la segmentación (HAZARDS)

y sus soluciones o alternativas

Ejemplo de una dependencia de datos (data hazard)

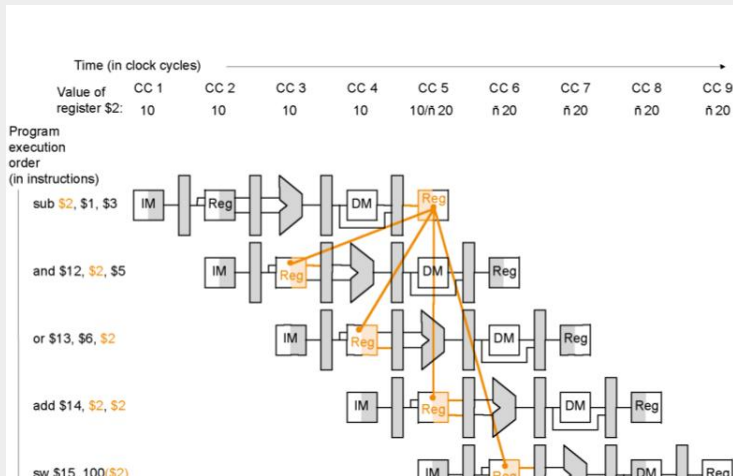
Secuencia de instrucciones

SUB	\$2,	\$1,	\$3
AND	\$12,	\$2,	\$5
OR	\$13,	\$6,	\$2
ADD	\$14,	\$2,	\$2
SW	\$15,	100	(\$2)

y sus soluciones o alternativas

Ejemplo de una dependencia de datos (data hazard)

Visión del problema en el camino de datos



» Problemas de la segmentación (HAZARDS)

y sus soluciones o alternativas

Ejemplo de una dependencia de datos (data hazard)

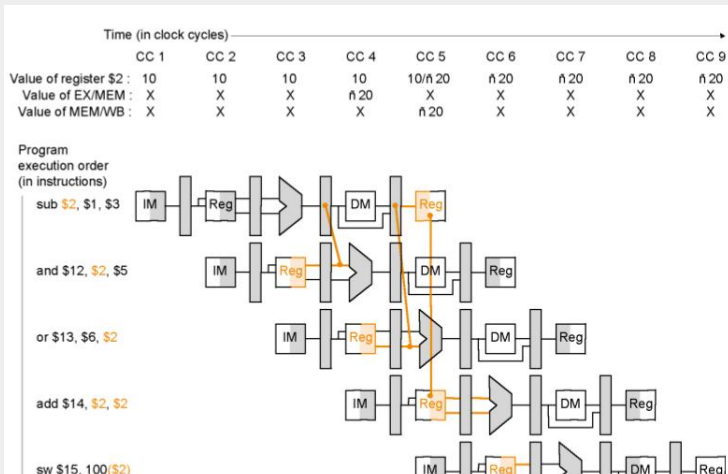
Solución 1: Resuelta por el compilador puede demorar algunas instrucciones

SUB	\$2,	\$1,	\$3
NOP			
NOP			
AND	\$12,	\$2,	\$5
OR	\$13,	\$6,	\$2
ADD	\$14,	\$2,	\$2
SW	\$15,	100	(\$2)

y sus soluciones o alternativas

Ejemplo de una dependencia de datos (data hazard)

Solución 1: Unidad de control mas compleja (forwarding)



» Hazards: Cuando no puede evitarse la detención

- * Delay slot: Se necesita una solución arquitectural (en MIPS lo soluciona el compilador)
- * Dependencias inevitables (el dato no está disponible en ninguna etapa)

» Consejos y preguntas

* ¿Preguntas?

» Bibliografia

Libros

- * David. Patterson John L. Hennessy (1995), ORGANIZACIÓN Y DISEÑO DE COMPUTADORES La interfaz hardware/software, McGraw-Hill (8 copias en biblioteca).