



Tema 3-3. Procesadores Superescalares

Arquitectura de
Computadores
Grado en Ingeniería
Informática





Sumario

- Introducción: $CPI < 1$
- Planificación Estática
 - Desenrollamiento de bucles
- Planificación Dinámica
 - Algoritmo de Tomasulo

CPI < 1: Enviar a ejecutar varias Instrucciones/Ciclo



- 2 filosofías de procesadores que alcanzan $CPI < 1$
- Very Long Instruction Words VLIW:
número fijo de instrucciones (4-16) planificadas por el compilador y ejecutadas en paralelo
 - IA-64
- Superscalar (SS): número variable de instrucciones/ciclo (1 á 8), planificadas por: (a) el compilador, y/o (b) HW (Tomasulo y otros)
 - Procesadores de Intel, AMD, etc.
- Una nueva forma de medir prestaciones :
Instrucciones por ciclo ($IPC = CPI^{-1}$) vs. CPI



DLX32ss: Superescalar con Planificación Estática



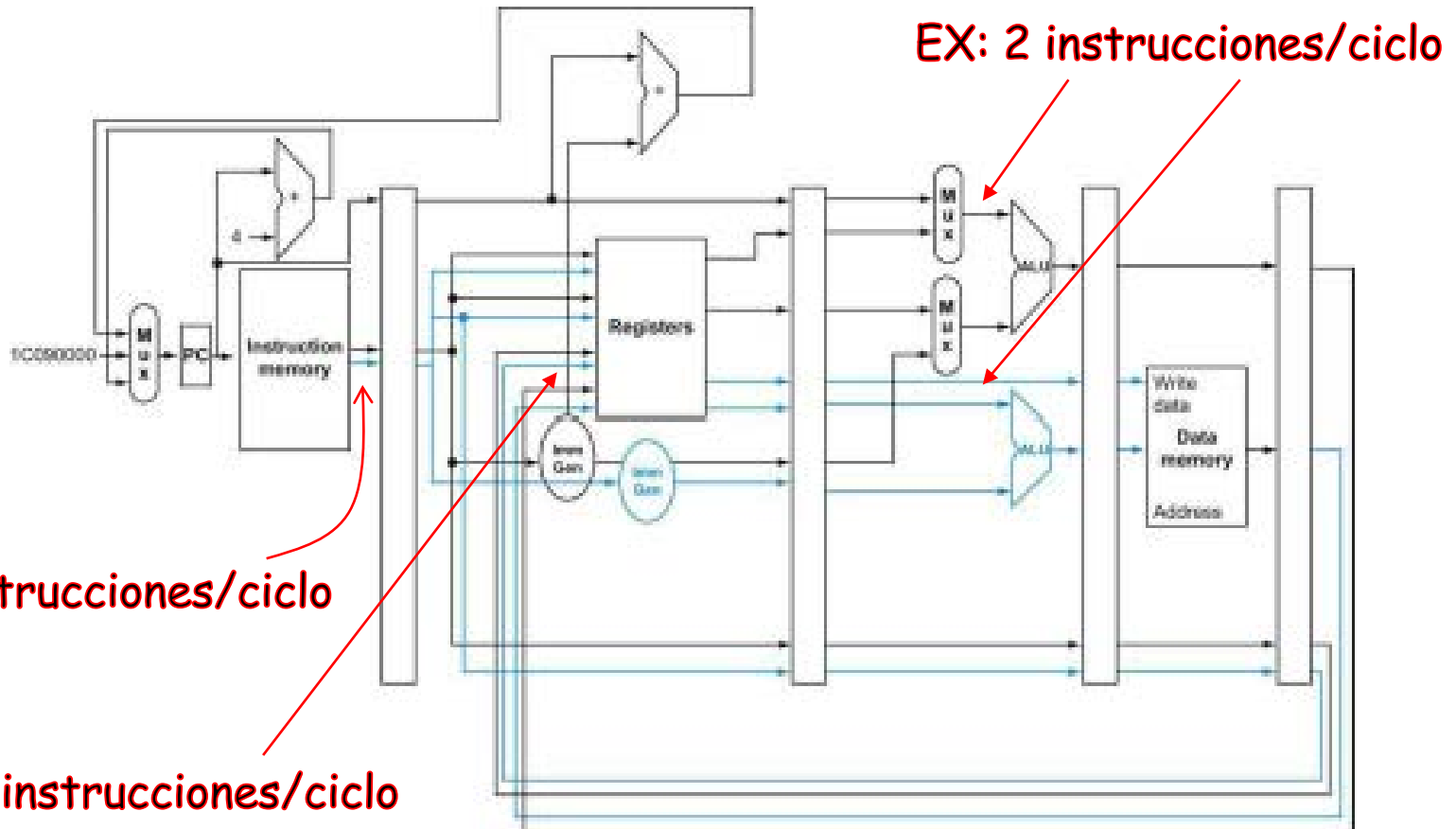
- DLX32ss Superescalar con Planificación Estática 2 Vías:
2 instrucciones de 32-bit (1 FP & 1 ENTEROS)
- IF: Búsqueda 64-bits/ciclo
 - » Emparejamiento en IF: ENTEROS izquierda, FP derecha
- EX: Envío a ejecución en orden
 - » Emparejamiento en EX: Se envía a ejecutar la 2ª instrucción si se envía la 1ª
 - » Dependencias se analizan en la etapa de envío a ejecutar (EX)
- Prestaciones de procesador pueden mejorar en factor 2X

$$IPC_{max}=2$$

Tipo Ins	Ciclos							
ENT	IF	ID	EX	MEM	WB			
FP	IF	ID	EX	EX	EX	WB		
ENT		IF	ID	EX	MEM	WB		
FP		IF	ID	EX	EX	EX	WB	
ENT			IF	ID	EX	MEM	WB	
FP			IF	ID	EX	EX	EX	WB

Posible debido a la existencia de 2 bancos de registros: ENT + FP

Microarquitectura generalizada de un procesador superescalar DLX32ss



Recordatorio: Desenrollamiento + Reordenamiento en el DLX32p escalar (Optimización "-O3")



LD á ADDD: 1 clk
ADDD á SD: 2 clk

```
1 Loop: LD      F0,0(R1)
2      LD      F6,-8(R1)
3      LD      F10,-16(R1)
4      LD      F14,-24(R1)
5      ADDD    F4,F0,F2
6      ADDD    F8,F6,F2
7      ADDD    F12,F10,F2
8      ADDD    F16,F14,F2
9      SD      0(R1),F4
10     SD      -8(R1),F8
11     SD      -16(R1),F12
12     SUBI    R1,R1,#32
13     BNEZ    R1,LOOP
14     SD      8(R1),F16
```

→
Transformación:
Optimizar el
emparejamiento
en DLX32ss
manteniendo las
latencias

```
Loop: LD      F0,0(R1)
      LD      F6,-8(R1)
      { LD      F10,-16(R1)
        ADDD    F4,F0,F2
      }
      { LD      F14,-24(R1)
        ADDD    F8,F6,F2
      }
      { SUBI    R1,R1,#32
        ADDD    F12,F10,F2
      }
      { SD      32(R1),F4
        ADDD    F16,F14,F2
      }
      SD      24(R1),F8
      SD      16(R1),F12
      BNEZ    R1, Loop
      SD      8(R1),F16
```

4 iteraciones: 14 clks,
3.5 clk / iteración (2.6X)

Desenrollamiento en DLX32ss Superscalar con Planificación Estática "-O5"

ORDEN DEL PROGRAMA EN LA CACHE DE INSTRUCCIONES

	INT	FP	CLK
Loop:	LD F0,0(R1)		1
	LD F6,-8(R1)		2
	LD F10,-16(R1)	ADDD F4,F0,F2	3
	LD F14,-24(R1)	ADDD F8,F6,F2	4
	SUBI R1,R1,#32	ADDD F12,F10,F2	5
	SD 32(R1),F4	ADDD F16,F14,F2	6
	SD 24(R1),F8		7
	SD 16(R1), F12		8
	BNEZ R1, Loop		9
	SD 8(R1),F16		10

Loop: LD F0,0(R1)
LD F6,-8(R1)
LD F10,-16(R1)
ADDD F4,F0,F2
LD F14,-24(R1)
ADDD F8,F6,F2
SUBI R1,R1,#32
ADDD F12,F10,F2
SD 32(R1),F4
ADDD F16,F14,F2
SD 24(R1),F8
SD 16(R1),F12
BNEZ R1, Loop
SD 8(R1),F16

- Desenrollado 4 veces
- 10 ciclos, 2.5 clk/iteración (3.6X)
- Eficiencia = $14 \text{ inst} / 20 \text{ inst_max} = 70\%$, $\text{IPC} = 14 / 10 = 1.4$

Resumen de Prestaciones



- Codificación Inicial sin optimización: 9 clk/iteración
- "-O1" Reordenamiento Instrucciones: 6 clk/ iteración (1.5X)
- "-O2" Desenrollamiento Bucles: 6.8 clk/ iteración (1.3X)
- "-O3" Desenrollamiento Bucles + Reordenamiento Instrucciones: 3.5 clk/ iteración (2.6X)
- "-O4" DLX32vliw + Desenrollamiento Bucles + Reordenamiento Instrucciones: 1.3 clk/ iteración (6.9X)
- "-O5" Desenrollamiento Bucles + DLX32ss:
2.5 clk/iteración (3.6X)

Problemas con Planificación Estática que obstaculizan conseguir en DLX32ss el IPC_{max}



- Se requiere recompilar el código fuente
- Un retardo de 1 ciclo de las cargas produce una penalización en SS de 3 instrucciones
 - Mitad derecha y siguiente pareja no pueden usar el resultado de la carga → reducción de eficiencia en procesadores SS

Problemas con DLX32ss en general que obstaculizan conseguir el IPC_{max}



- Se requieren múltiples unidades funcionales FP o segmentarlas, porque si no sería un cuello de botella y no se observarían las mejoras asociadas a SS
- Riesgo Estructural y/o RAW: Al enviar a ejecutar un load/store/move sobre un registro FP + una instrucción FP
 - Añadir un puerto de escritura más a banco FP
- La diferencia en latencias de las operaciones que se ejecutan en cada vía del SS puede hacer que las excepciones sean imprecisas

Problemas con DLX32ss en general que obstaculizan conseguir el IPC_{max}



- PARA LLEGAR A MANTENER EL RITMO IPC_{max} SE REQUIEREN DE NUEVAS TÉCNICAS QUE DISMINUYAN LOS EFECTOS NEGATIVOS QUE ORIGINAN ESTOS PROBLEMAS
- Posible alternativa: Planificación dinámica de instrucciones en procesadores SS
 - Principal ventaja: no hace falta recompilar el código fuente
 - Principal desventaja: complica el diseño del procesador

Planificación DINÁMICA en Superscalares



- **Objetivo:** ejecutar instrucciones fuera de orden en procesadores superescalares (DLX32ss)
- “Arquitectura Desacoplada”: Aplica el algoritmo de Tomasulo a un procesador superescalar (por ejemplo, suponemos que es de 2 vías: FP + ENT)
 - 2 caminos independientes de ejecución: FP, ENT
 - Envío a ejecutar en orden
 - Ejecución fuera de orden
 - Terminación fuera de orden
 - Colas de desacoplo entre las dos vías

Arquitectura Desacoplada



Cola de Instrucciones
(instrucciones ENT y FP mezcladas)



Envío a ejecutar 2 instrucciones/ciclo
EN PARALELO: 1 ENT + 1 FP

**PUNTO
FLOTANTE (FP)**

Banco
Registros
FP

ER

UF FP

CDB2 (FP)

Cache
de Datos

Unidad
de cargas
y almace-
namientos

ER

UF ENT

ENTEROS (ENT)

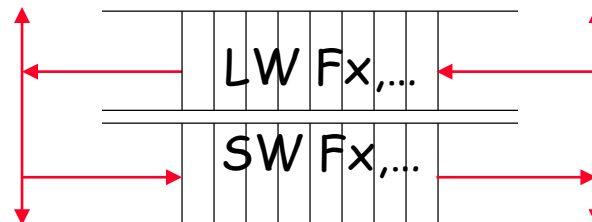
Banco
Registros
ENT

ER

UF Saltos

CDB1 (ENT)

ER: Estaciones
de reserva
UF: Unidad
funcional



Colas de Desacoplo: ENT -> FP, FP -> ENT



Planificación DINAMICA en Superscalares

- **PROBLEMA-1**: Las instrucciones ENT "cargas FP (LW)" y los "moves ENT-→FP (MOVI2FP F2, R0)" podrían causar dependencias entre las partes ENT (fuente) y FP (destino) durante el ciclo en que se envían en paralelo en la etapa IS una instrucción ENT y otra FP
- **Solución**
 - Arquitectura desacoplada con **2 buses** de datos común (CDB): 1 FP + 1 ENT
 - AÑADIR una **Cola para Cargas**: los datos cargados desde la cache de datos son leídos desde FP usando esta cola en el orden que son buscados a memoria
 - AÑADIR una **Cola para Almacenamientos de Resultados FP**: los datos que se van a guardar en la cache de datos son previamente alojados en esta cola desde FP
 - Las cargas consultan la *Cola de Almacenamiento* para evitar RAW (un almacenamiento previo coincide en dirección que una carga posterior)
 - Los almacenamientos consultan la *Cola para Cargas* para evitar WAR y WAW (WAR: una carga previa podría usar el resultado de un almacenamiento posterior)



Detalle de las Colas de Desacoplo

LD	F4,32(R2)
ADDD	F6,F4,F4

Estaciones Reserva Cargas/Almacenamientos:

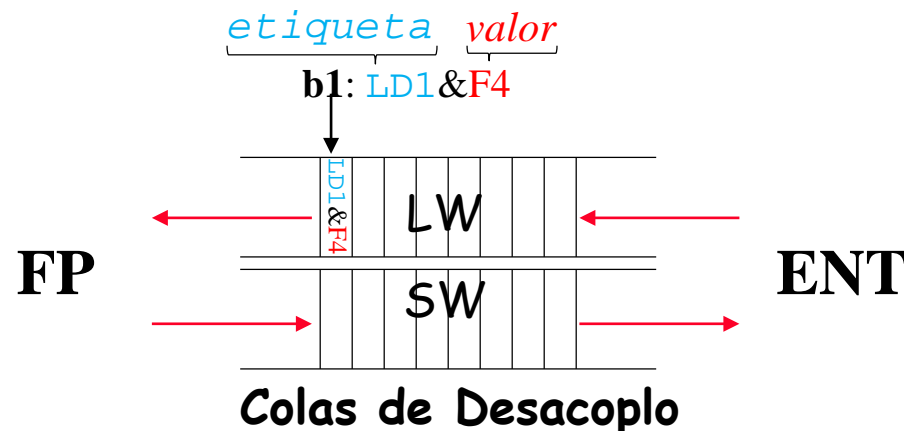
Nombre	Ocupa	Op	dirección	Cola LW
LD1	Si	LD	32+R2	b1

Cuando se envía a ejecutar, se reserva una posición de la cola ENT→FP

Estaciones Reserva FP:

Nombre	Ocupa	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
Add1	Si	ADDD			LD1	LD1

Cuando se recibe el dato dependiente, proviene de la estación de reserva LD1 reservada en la etapa issue y que se indica en la cola ENT→FP



Detalle de las Colas de Desacoplo



ADDD	F6,F4,F4
SD	F6,32(R2)

Estaciones Reserva Cargas/Almacenamientos: $S1$ RS

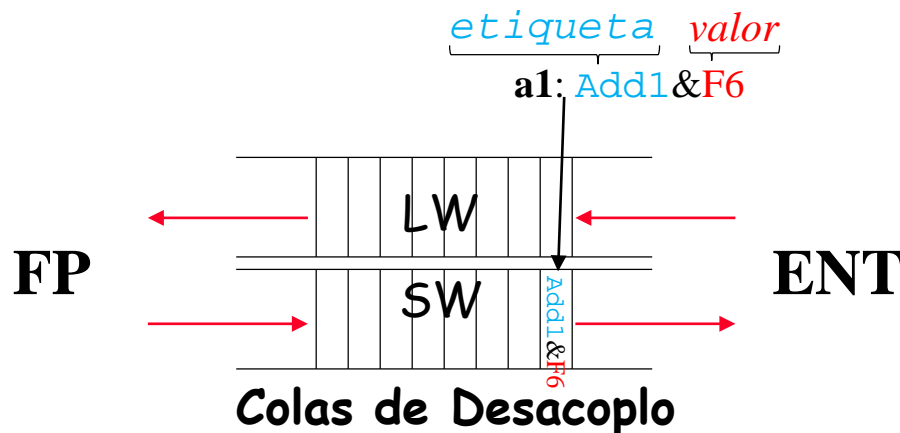
Nombre	Ocupa	Op	dirección	Vj	Qj
SD1	Si	SD	32+R2		Add1

Cuando se recibe el dato dependiente, proviene de la estación de reserva Add1 reservada en la etapa issue y que se indica en la cola FP→ENT

Estaciones Reserva FP:

Nombre	Ocupa	Op	Vj	Vk	Qj	Qk	Cola	SW
Add1	Si	ADDD	F4	F4				a1

Cuando se envía a ejecutar, se reserva una posición de la cola FP→ENT



EJEMPLO-1: Planificación Dinámica en DLX32ss

"sin" Predicción de Saltos (sin ROB)



Latencias

LD= 2 ciclos (Dir + Acceso)

ADDD= 3 ciclos

ENT= 1 ciclos

Saltos no emparejables

No saltos retardados

SIN Especulación de Saltos
debido a la inseguridad de
actualizar el estado del
procesador correctamente

Iteración	Instrucción	Empieza		
		IS	EX	WB
1	LD F0 0+ R1	1	2	4
1	ADDD F4 F0 F2	1	5	8
1	SD F4 0+ R1	2	9	
1	SUBI R1 R1 #8	3	4	5
1	BNEZ R1 LOOP	4	6	
2	LD F0 0+ R1	7	8	10
2	ADDD F4 F0 F2	7	11	14
2	SD F4 0+ R1	8	15	
2	SUBI R1 R1 #8	9	10	11
2	BNEZ R1 LOOP	10	12	


Envío a ejecutar (IS): 5 ciclos/iteración (10 ciclos/2 iteraciones)

Ocupación unidades funcionales (SD, LD, ENT, FP)= 7.5
ciclos/iteración (15 ciclos / 2 iteraciones): realmente UF ocupadas 6
ciclos: iter-1 termina en clk=9, iter-2ª termina en clk=15 →
diferencia= 6 clks

Ritmo IS > Ritmo EX: PROBABLE Saturación de la Cola de Instrucciones

Resumen de Prestaciones



Tipo de Planificación	ID de Planificación	Descripción de Planificación	Ciclos /ite-ración	Speed-Up
Estática	-	Inicial, sin optimización	9,0	1,0X
Estática	-O1	Reordenamiento Instrucciones	6,0	1,5X
Estática	-O2	Desenrrollamiento Bucles	6,8	1,3X
Estática	-O3	Desenrrollamiento Bucles + Reordenamiento	3,5	2,6X
Estática	-O4	DLX32vliw + Desenrrollamiento Bucles + Reordenamiento Instrucciones	1,3	6,9X
Estática	-O5	Desenrrollamiento Bucles + DLX32ss	2,5	3,6X
 Dinámica	-	Tomasulo + DLX32ss	6,0	1,5X