AT5118 – Arquitectura e Ingeniería de Computadores I

Juan Antonio Maestro

Universidad Antonio de Nebrija

Procesadores superescalares y VLIW

Curso 2011-2012

¿Se puede incrementar el ILP?

- Hasta ahora, el CPI teórico (óptimo) es 1.
- ¿Qué ocurre si se quiere incrementar el ILP, y hacer CPI < 1?
  - Es preciso lanzar *más de una instrucción* por ciclo de reloj.
- Esto genera técnicas más complejas para resolución de:
  - dependencias de datos (planificación).
  - dependencias de control (especulación).

Alternativas para incrementar el ILP

Alternativas:

- Superescalar:
  - Lanzamiento de 1- 8 instrucciones por ciclo.
  - Planificación estática (compilador).
  - Planificación dinámica (procesador).
- VLIW:
  - Lanzamiento de un número fijo de instrucciones por ciclo.
  - Sólo planificación estática (compilador).

Superescalar (basado en el DLX)

- Se pueden lanzar hasta dos instrucciones por ciclo (CPI teórico = 0.5):
  - 1ª: Load, store, salto, operación entera en ALU
  - 2ª: Operación en Punto Flotante.

Instruction type	Pipe stages							
Integer instruction	IF	ID	EX	MEM	WB			
FP instruction	IF	ID	EX	MEM	WB			
Integer instruction		IF	ID	EX	MEM	WB		
FP instruction		IF	ID	EX	MEM	WB		
Integer instruction			IF	ID	EX	MEM	WB	
FP instruction			IF	ID	EX	MEM	WB	
Integer instruction				IF	ID	EX	MEM	WB
FP instruction				IF	ID	EX	MEM	WB

Superescalar (basado en el DLX)

- Si la 1ª instrucción es un Load de Punto Flotante, se crea un riesgo que afecta a las 3 siguientes instrucciones (el resultado no se puede utilizar por la otra instrucción de ese ciclo ni por las dos del siguiente).
- Lo mismo ocurre con el riesgo de control si la 1ª instrucción es un salto.
- Implica traer y decodificar 64 bits por ciclo.
- Se duplican recursos (puertos de memoria, de registros, …).
- Unidades Funcionales segmentadas o duplicadas.

Superescalar simple: Ejemplo

```
Loop:   LD      F0,0(R1)
        ADDD    F4,F0,F2
        SD      0(R1),F4
        SUBI    R1,R1,#8
        BNEZ    R1,Loop
```

Desenrollar 5 veces y planificar:

- 12 ciclos ⇒ 2.4 ciclos por elemento
- Planificación no superescalar: 3.5 ciclos por elemento

	Integer instruction		FP instruction	Clock cycle
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
	LD	F18,-32(R1)	ADDD F12,F10,F2	5
	SD	0(R1),F4	ADDD F16,F14,F2	6
	SD	-8(R1),F8	ADDD F20,F18,F2	7
	SD	-16(R1),F12		8
	SUBI	R1,R1,#40		9
	SD	16(R1),F16		10
	BNEZ	R1,LOOP		11
	SD	8(R1),F20		12

Desventajas del Superescalar simple

Desventajas del modelo simplificado:

- CPI = 0.5 sólo alcanzable cuando el 50% son instrucciones en Punto Flotante.
- Planificación estática. No adaptable a cambios en la ejecución.

Superescalar con planificación dinámica

- Extensión del algoritmo de Tomasulo.
- Dos instrucciones por ciclo: Etapa de lanzamiento dividida en dos subciclos, a fin de evitar riesgos de datos.
- Estaciones de reserva separadas para Enteros y Punto Flotante.

Iteración	Instrucción	Lanzada	Ejecutada	Escribe	
1	LD F0,0(R1)	1	2	4	
1	ADDD F4,F0,F2	1	5	8	
1	SD 0(R1),F4	2	9		
1	SUBI R1,R1,#8	3	4	5	4 ciclos por iteración
1	BNEZ R1,LOOP	4	5		
2	LD F0,0(R1)	5	6	8	
2	ADDD F4,F0,F2	5	9	12	
2	SD 0(R1),F4	6	13		
2	SUBI R1,R1,#8	7	8	9	
2	BNEZ R1,LOOP	8	9		

VLIW

- Se lanza un conjunto de instrucciones simultáneamente (número fijo).
- El compilador realiza una organización de código para evitar dependencias.
- Las instrucciones tienen un ancho de bits elevado.
- El compilador planifica instrucciones cruzando fronteras de salto (*Trace Scheduling)*

VLIW: Ejemplo

```
LOOP:   LD F0,0(R1)
        ADDD F1,F0,F2
        SD 0(R1),F4
        SUBI R1,R1,#8
        BNEZ R1, LOOP
```

7 iteraciones en 9 ciclos:

- 1.3 ciclos por iteración
- Ocupación menor del 50%

Mem ref 1	Mem ref 2	FP op	FP op	Int op/branch
LD F0,0(R1)	LD F6,-8(R1)			
LD F10,-16(R1)	LD F14,-24(R1)			
LD F18,-32(R1)	LD F22,-40(R1)	ADDD F4,F0,F2	ADDD F8,F6,F2	
LD F26,-48(R1)		ADDD F12,F10,F2	ADDD F16,F14,F2	
		ADDD F20,F18,F2	ADDD F24,F22,F2	
SD 0(R1),F4	SD -8(R1),F8	ADDD F28,F26,F2		
SD -16(R1),F12	SD -24(R1),F16			
SD -32(R1),F20	SD -40(R1),F24			SUBI R1,R1,#56
SD 8(R1),F28				BNEZ R1, LOOP

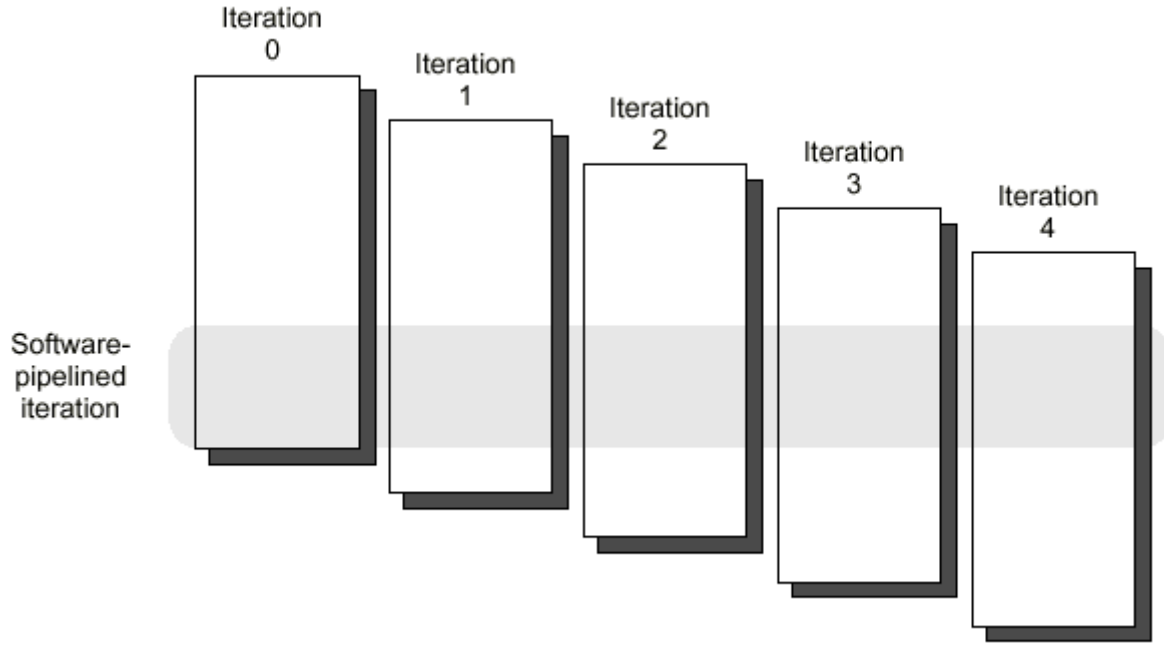
Incrementar el ILP mediante técnicas software

Mecanismos software para incrementar el paralelismo a nivel de instrucción:

- Desenrollado de bucles. (Tema anterior)
- Software *pipelining*.
- Planificación de trazas.

Software *pipelining*

- Reorganización de bucles: Cada instrucción pertenece a una iteración distinta, de tal manera que no existan dependencias de datos.

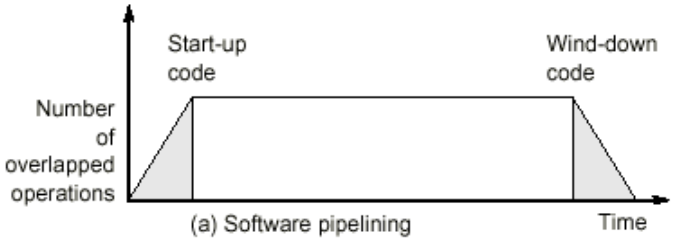


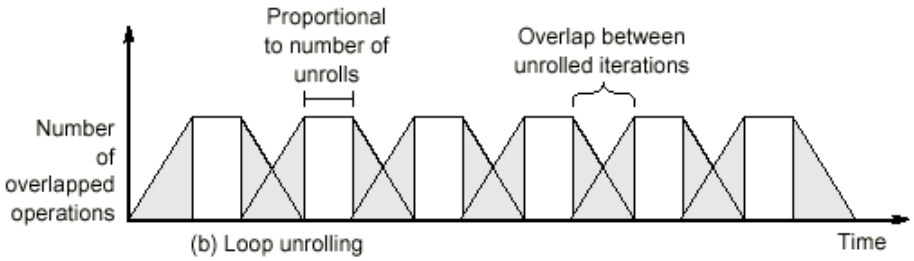
Software *pipelining*

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

```
1   SD    0(R1),F4 ; Stores M[i]
2   ADDD  F4,F0,F2 ; Adds to M[i-1]
3   LD    F0,-16(R1); Loads M[i-2]
4   SUBI  R1,R1,#8
5   BNEZ  R1,LOOP
```

Es un desenrollado de bucles simbólico, con la ventaja de que el código es más corto y el rendimiento más alto



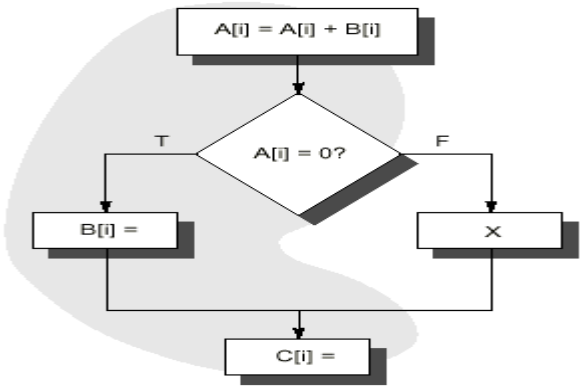


Planificación de trazas

Mecanismo por el que se definen los bloques de código con mayor probabilidad de ejecución a través de saltos:

- Selección de trazas: Se escogen los bloques lineales más probables.
- Compactación de trazas: Se optimiza el código de esos bloques.

Es preciso un mecanismo de anulación, en el caso de que la ejecución no sea la prevista.

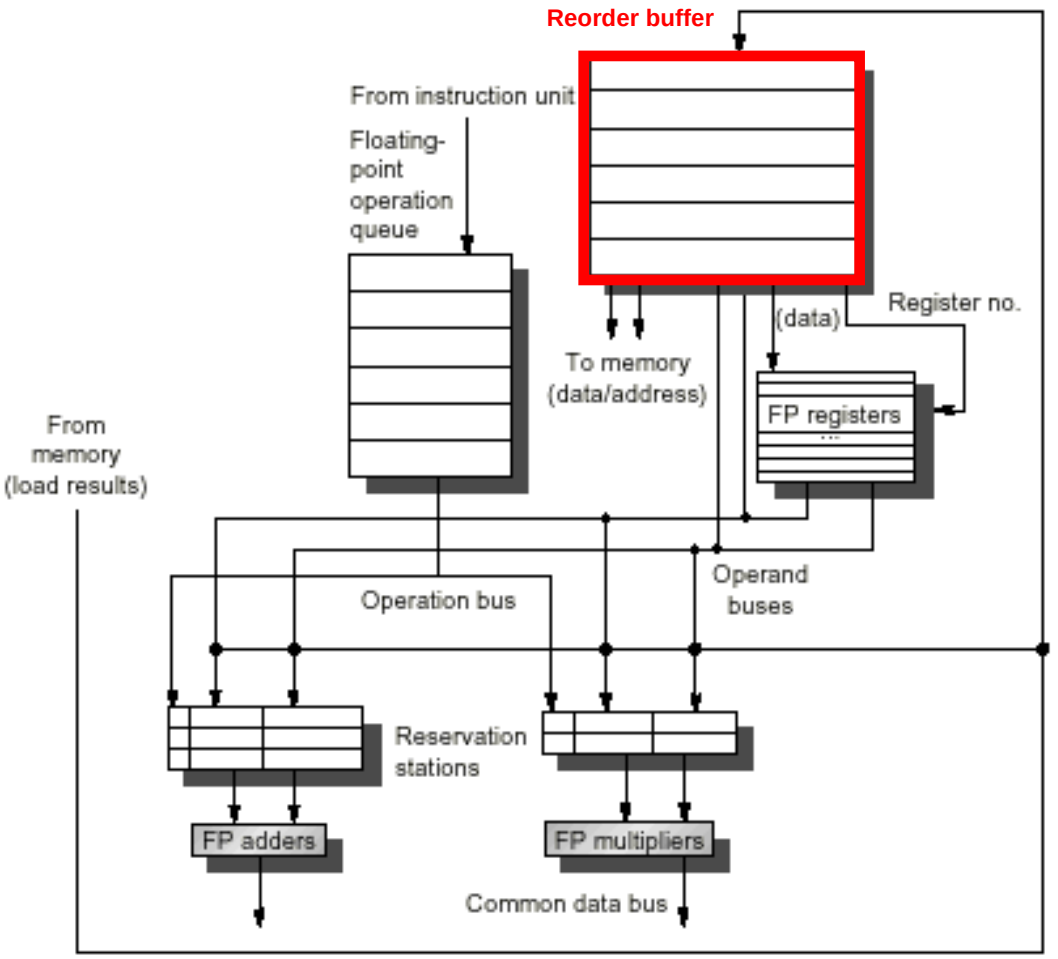


Planificación dinámica con especulación

- Lanzamiento de instrucciones más allá de saltos, antes de que se sepa si el salto se va a tomar o no: Especulación.
- Es preciso un mecanismo de anulación de instrucciones (si la predicción de salto fue errónea) que deshaga las operaciones realizadas.
- Una instrucción deja de ser especulada cuando el salto del que depende se resuelve (*Commit*).
- Las instrucciones se ejecutan fuera de orden, pero se resuelven en orden.

Planificación dinámica con especulación

Las instrucciones no resueltas se almacenan en el *Buffer de reordenamiento*. Proporciona los operandos hasta que se ejecuta el *commit* de la instrucción.



Algoritmo de Tomasulo especulativo

- Issue: Lanza la instrucción si hay una ER libre, y queda alguna entrada en el Buffer de reordenamiento. Envía los operandos a la ER si están en los registros o en el Buffer.
- Ejecución: Chequea los riesgos RAW.
- Escritura de resultados: Cuando se genera el resultado, mandarlo por el CDB al Buffer y a cada ER que lo necesite.
- *Commit*: Cuando el salto del que depende la instrucción se resuelve, se elimina dicha instrucción del Buffer de Reordenamiento.

Ejemplo de planificación dinámica y especulación

```
LD      F6,34(R2)
LD      F2,45(R3)
MULTD   F0,F2,F4
SUBD    F8,F6,F2
DIVD    F10,F0,F6
ADDD    F6,F8,F2
```

Reservation stations

	Estado	Opera	Vj	Vk	Qj	Qk	Destino
ADD1	Libre						
ADD2	Libre						
ADD3	Libre						
MUL1	Libre	MULT	M(45+(R3))	F4			#3
MUL2	Ocup.	DIV		M(34+(R2))	#3		#5

Reorder buffer

	Estado	Instrucción	Estado	Destino	Valor
1	libre	LD F6,34(R2	Commit	F6	M(34+(R2))
2	libre	LD F2,45(R3)	Commit	F2	M(45+R3))
3	Ocupada	MULTD F0,F2,F4	Escri. Res	F0	#2x(F4)
4	Ocupada	SUB F8,F6,F2	Escri. Res.	F8	#1 - #2
5	Ocupada	DIVD F10,F0,F6	Ejecuta	F10	
6	Ocupada	ADDD F6,F8,F2	Escri. Res.	F6	#4 + #2

FP register status

	F0	F2	F4	F6	F8	F10	F12	...
Buffer Nº	3			6	4	5		
Ocupado	si	no	no	si	si	si	no	

Ejemplo de planificación dinámica y especulación

```
Loop:  LD      F0,0(R1)
       MULTD   F4,F0,F2
       SD      0(R1),F4
       SUBI    R1,R1,#8
       BNEZ    R1,Loop
```

Reservation stations

	Estado	Opera	Vj	Vk	Qj	Qk	Destino
MUL1	Libre	MULT	M(0+(R1))	F2			#2
MUL2	Libre	MULT	M(0+(R1))	F2			#3

Reorder buffer

	Estado	Instrucción	Estado	Destino	Valor
1	libre	LD F0,0(R1)	Commit	F0	M(0+(R1))
2	libre	MULTD F4,F0,F2	Commit	F4	F0 x F2
3	Ocupada	SD 0(R1),F4	Escri. Res	0+R1	#2
4	Ocupada	SUBI R1,R1,#8	Escri. Res.	R1	R1-8
5	Ocupada	BNEZ R1, loop	Escri. Res		
6	Ocupada	LD F0,0(R1)	Escri. Res.	F0	M(#4)
7	Ocupada	MULTD F4,F0,F2	Escri. Res	F4	#6xF2
8	Ocupada	SD 0(R1),F4	Escri. Res	0+R1	#7
9	Ocupada	SUBI R1,R1,#8	Escri. Res	R1	#4 - #8
10	Ocupada	BNEZ R1,Loop	Escri. Res.		

Especuladas

FP register status

	F0	F2	F4	F6	F8	F10	F12	...
Buffer Nº	6		7					
Ocupado	si	no	si	no	no	no	no	