

Planificación dinámica - Concepto

Despacho y ejecución en orden: una limitación.

Stall también detiene a operaciones independientes.

DIVD F0,F2,F4 (1)

ADDD F10,F0,F8 (2)

SUBD F12,F8,F14 (3)

No siempre ejecutar en el orden del programa.

Dividir la etapa ID en dos:

Emisión: decodificación y hazards estructurales.

Leer operandos: si no hay hazard, lee los operandos.

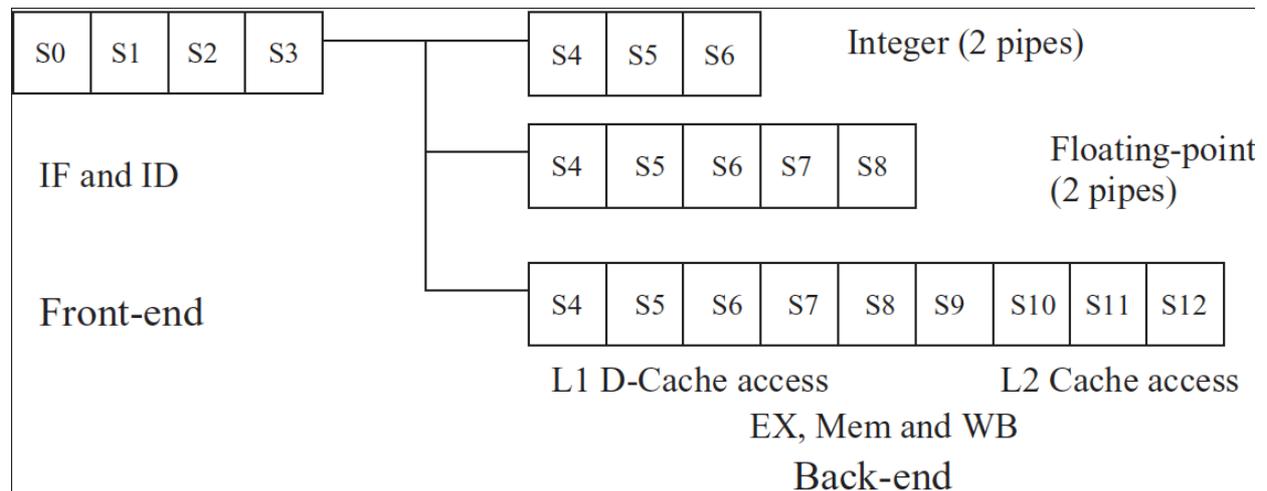
2/11/16



Guillermo Aguirre

1

Segmentación de instrucción del Alpha 21164



- Procesador super-escalar RISC de emisión “en orden”.
- Se recuperan 4 instrucciones de la I-Cache por ciclo.
- El extremo inicial lleva 4 ciclos.
- La segmentación de enteros son 7 etapas y de PF 9 etapas.
- Es lo que se llama super-pipeline

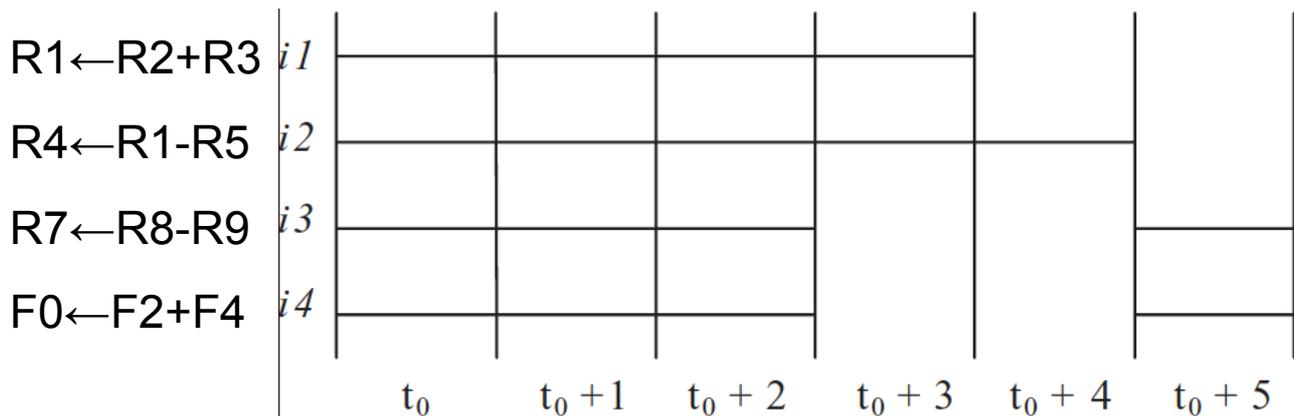
2/11/16



Guillermo Aguirre

2

Utilización del extremo inicial



Hazards de datos

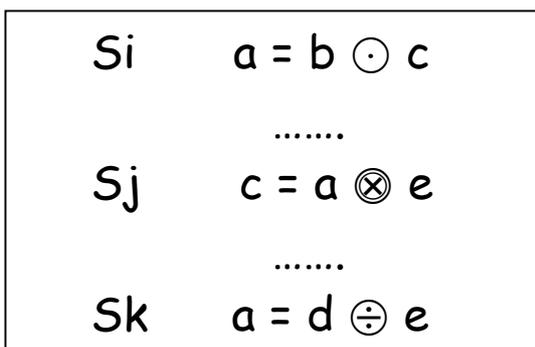
Se produce cuando por la segmentación, el orden de LECTURA de los operandos y la ESCRITURA de resultados se modifica respecto a lo especificado en el programa.

Se produce un riesgo si existe dependencia entre instrucciones que se ejecutan concurrentemente.

Dominio: operandos de la instrucción

Rango: resultado de la instrucción

Situaciones: *i* precede a *j*



$D(i) \cap D(j) \neq \phi$ Sin RIESGO

$D(i) \cap R(j) \neq \phi$ riesgo WAR

$R(i) \cap D(j) \neq \phi$ riesgo RAW

$R(i) \cap R(k) \neq \phi$ riesgo WAW



Planificación dinámica - Riesgos WAR y WAW

DIVD F0,F2,F4
ADDD F10,F0,F8
SUBD F8,F8,F14

Antidependencia (WAR)

DIVD F0,F2,F4
ADDD F10,F0,F8
SUBD F10,F8,F14

Riesgo WAW

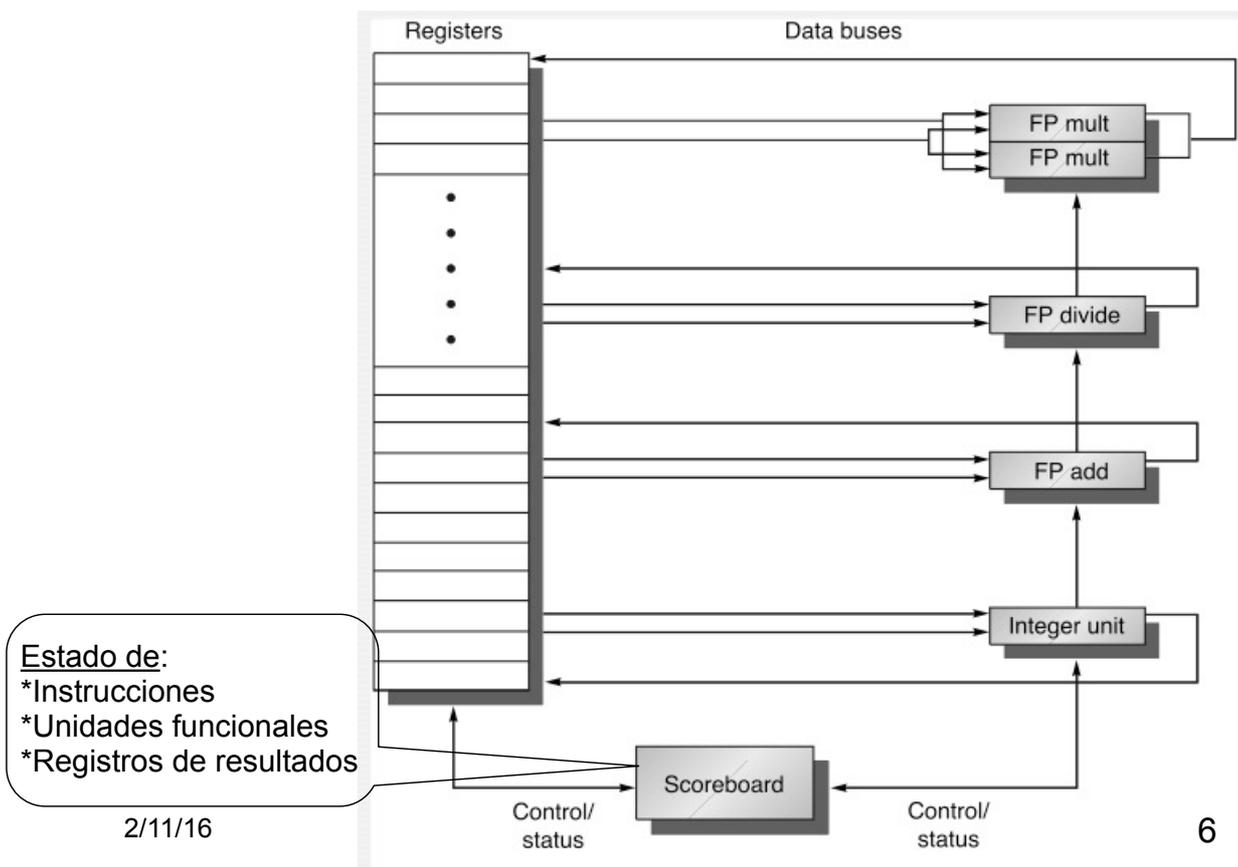
2/11/16



Guillermo Aguirre

5

MIPS usando Scoreboard



Etapas en Scoreboarding (1)

Etapas de emisión: controla que la unidad funcional esté disponible y que no haya otra instrucción que tenga el mismo registro destino. Con esto se evitan hazards estructurales y hazard WAW. Se emite la instrucción y se actualizan las estructuras de datos. Si hay WAW o hazard estructural se genera un stall.



Etapas en Scoreboarding (2)

Etapas de despacho: Controla si los operandos están disponibles.

Resuelve RAW dinámicamente.

Si es posible se leen los operandos desde los registros para comenzar la ejecución.



Etapas en Scoreboarding (3)

Etapas de ejecución : Hace la ejecución y notifica al scoreboard que se ha completado la ejecución.

Reemplaza la etapa de EX en el pipeline clásico.

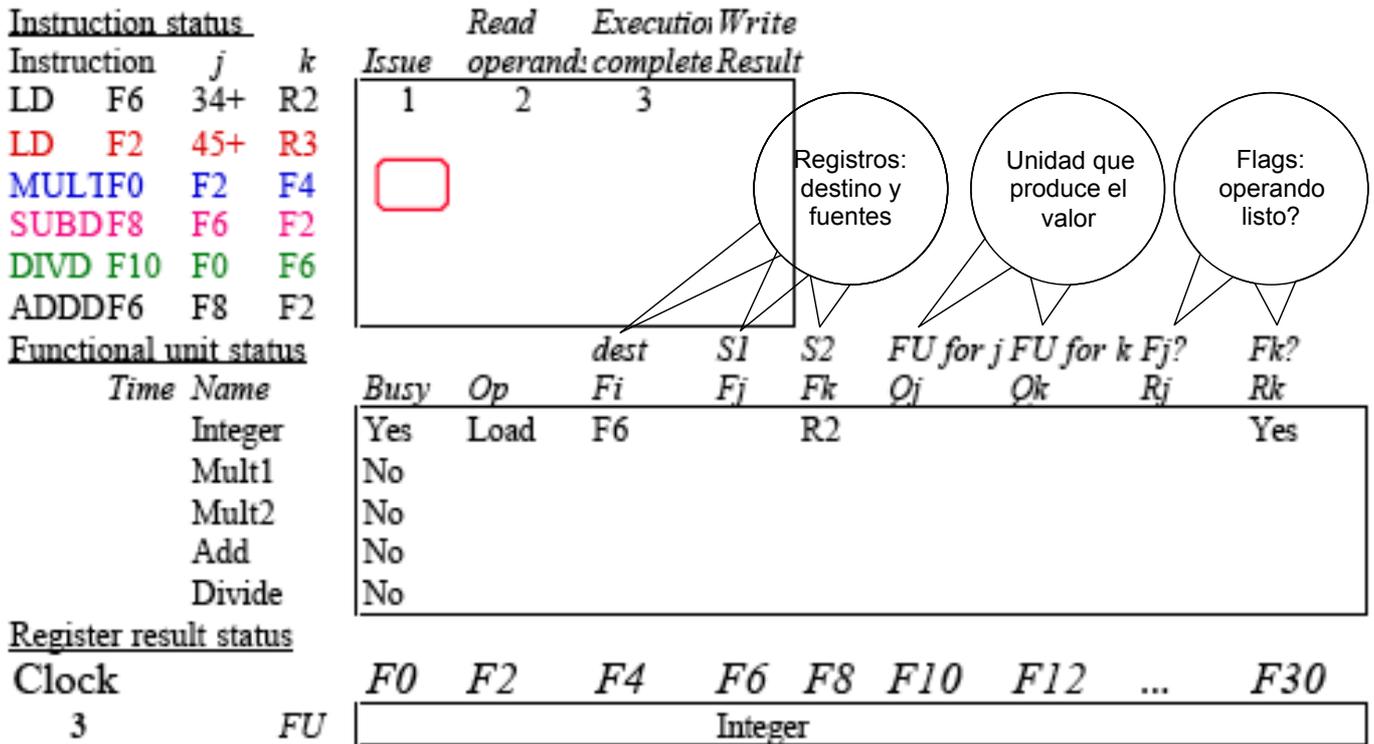


Etapas en Scoreboarding (4)

Etapas de escritura: Se chequea por hazards WAR y se hace un stall de la instrucción a punto de finalizar, si es necesario. Si el resultado se escribe en un operando fuente de alguna instrucción activa anterior, la instrucción no puede finalizar y debe esperar que la otra lea el operando.



Ejemplo ScoreBoard - Ciclo 3



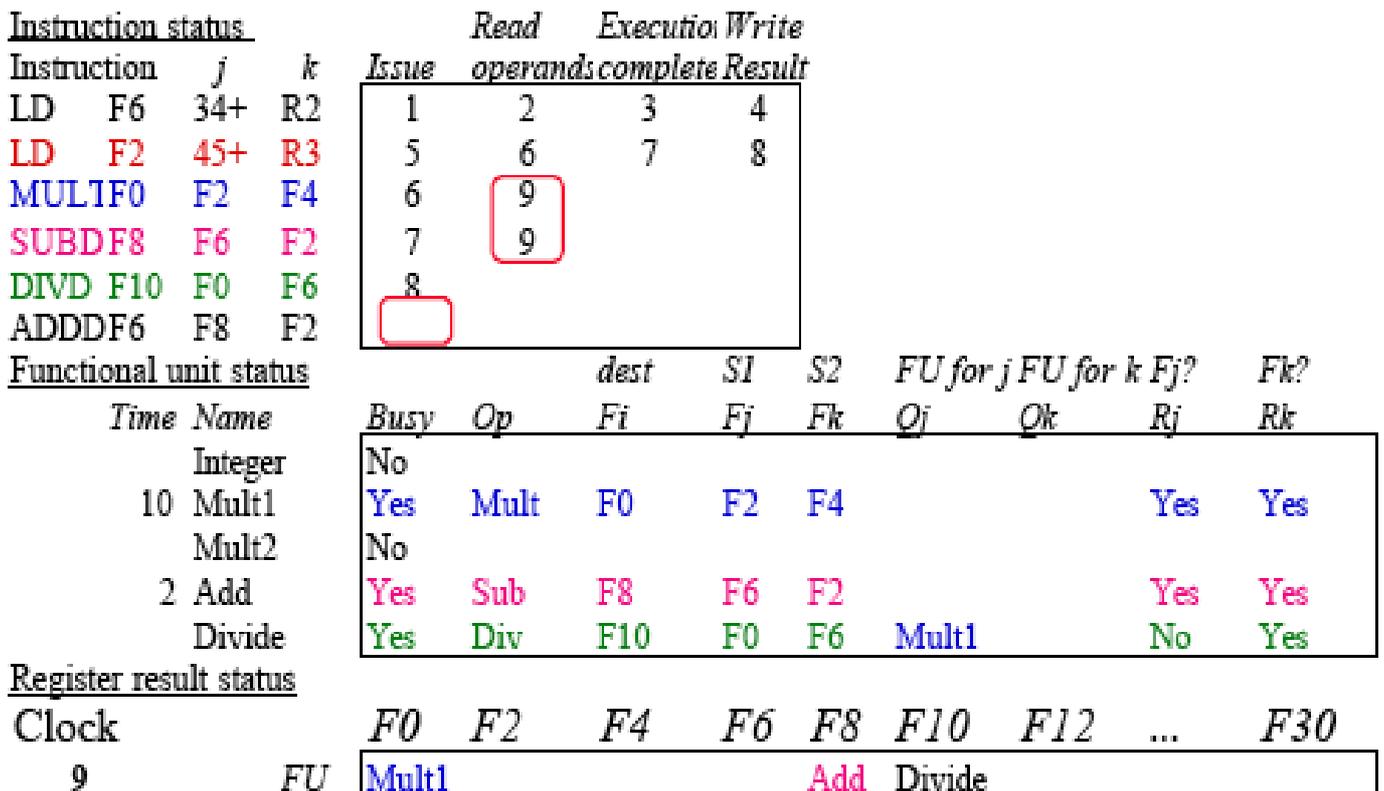
2/11/16



Guillermo Aguirre

11

Ejemplo ScoreBoard - Ciclo 9



2/11/16



Guillermo Aguirre

12

Ejemplo ScoreBoard - Ciclo 17

<u>Instruction status</u>				<i>Read</i>	<i>Execution</i>	<i>Write</i>					
Instruction	<i>j</i>	<i>k</i>	<i>R2</i>	<i>Issue</i>	<i>operands</i>	<i>complete</i>	<i>Result</i>				
LD	F6	34+	R2	1	2	3	4				
LD	F2	45+	R3	5	6	7	8				
MULIF0	F2	F4		6	9						
SUBDF8	F6	F2		7	9	11	12				
DIVDF10	F0	F6		8							
ADDDF6	F8	F2		13	14	16					

La suma toma 2 ciclos

<u>Functional unit status</u>		<i>dest</i>	<i>S1</i>	<i>S2</i>	<i>FU for j</i>	<i>FU for k</i>	<i>Fj?</i>	<i>Fk?</i>
<i>Time</i>	<i>Name</i>	<i>Fi</i>	<i>Fj</i>	<i>Fk</i>	<i>Qj</i>	<i>Qk</i>	<i>Rj</i>	<i>Rk</i>
	Integer	No						
2	Mult1	Yes	Mult	F0	F2	F4	Yes	Yes
	Mult2	No						
	Add	Yes	Add	F6	F8	F2	Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1	No

<u>Register result status</u>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
Clock	17									

2/11/16



Guillermo Aguirre

13

Ejemplo ScoreBoard - Ciclo 62

<u>Instruction status</u>				<i>Read</i>	<i>Execution</i>	<i>Write</i>					
Instruction	<i>j</i>	<i>k</i>	<i>R2</i>	<i>Issue</i>	<i>operands</i>	<i>complete</i>	<i>Result</i>				
LD	F6	34+	R2	1	2	3	4				
LD	F2	45+	R3	5	6	7	8				
MULIF0	F2	F4		6	9	19	20				
SUBDF8	F6	F2		7	9	11	12				
DIVDF10	F0	F6		8	21	61	62				
ADDDF6	F8	F2		13	14	16	22				

La multiplicación toma 10 ciclos

La división toma 40 ciclos

<u>Functional unit status</u>		<i>dest</i>	<i>S1</i>	<i>S2</i>	<i>FU for j</i>	<i>FU for k</i>	<i>Fj?</i>	<i>Fk?</i>
<i>Time</i>	<i>Name</i>	<i>Fi</i>	<i>Fj</i>	<i>Fk</i>	<i>Qj</i>	<i>Qk</i>	<i>Rj</i>	<i>Rk</i>
	Integer	No						
	Mult1	No						
	Mult2	No						
	Add	No						
0	Divide	No						

<u>Register result status</u>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
Clock	62									

2/11/16



Guillermo Aguirre

14

Costo y beneficio

Mejora del desempeño entre 1.7 y 2.5

El scoreboard no requiere muchos más circuitos que las unidades comunes.

Requiere muchos buses, \cong 4 veces más.

De interés al considerar :

Despacho de múltiples instrucciones por ciclo
Especulación



Factores limitantes

Cantidad de paralelismo disponible entre las instrucciones.

Número de entradas en el ScoreBoard.

Número y tipo de unidades funcionales.

Presencia de antidependencias y dependencias de salida



Algoritmo de Scoreboard (1)

Emisión o Issue

Esperar que	Tareas
Unidad Funcional[FU] no esté ocupada y Result[D] esté en blanco	$Busy[FU] \leftarrow yes; Op[FU] \leftarrow op;$ $Fi[FU] \leftarrow D \quad Fj[FU] \leftarrow S1;$ $Fk[FU] \leftarrow S2; Qj \leftarrow Result[S1];$ $Qk \leftarrow Result[S2]; Rj \leftarrow not \ Qj;$ $Rk \leftarrow not \ Qk; result[D] \leftarrow FU$

16/09/2013



Guillermo Aguirre

17

Algoritmo de Scoreboard (2)

Lectura de registros y ejecución

Esperar que	Tareas
Rj y Rk sean verdaderos	$Rj \leftarrow no; Rk \leftarrow no;$ $Qj \leftarrow 0; Qk \leftarrow 0$
La FU finalice	

16/09/2013



Guillermo Aguirre

18

Algoritmo de Scoreboard (3)

Escritura de resultados

Esperar que	Tareas
$\forall f(($ $F_j[f] \neq F_i[FU] \vee R_j[f] = \text{no}$ $) \wedge ($ $F_k[f] \neq F_i[FU] \vee R_k[f] = \text{no}$ $)$ $)$	$\forall f(\text{ Si } Q_j[f] = \text{FU} \text{ then } R_j[f] \leftarrow \text{yes};$ $\forall f(\text{ Si } Q_k[f] = \text{FU} \text{ then } R_k[f] \leftarrow \text{yes};$ $\text{Result}[F_i[FU]] \leftarrow 0; \text{Busy}[FU] \leftarrow \text{No}$