

Tomasulov algoritam

Hardversko odmotavanje petlji

Hw odmotavanje petlje

- * Prava snaga eliminisanja WAW i WAR hazarda kroz dinamičko preimenovanje registara najbolje se može videti na pimeru izvršenja petlje

➤ loop:	LD	F0, 0(R1)
➤	MULD	F4, F0, F2
➤	SD	F4, 0(R1)
➤	SUBI	R1, R1, #8
➤	BNEZ	R1, Loop

- * Ako je predviđavanje da će se grananje obaviti, korišćenje RS će omogućiti da se više iteracija petlje izvršava jednovremeno

- * ovo se postiže bez promene koda

- u suštini petlja se dinamički odmotava uz pomoć hw korišćenjem RS koje deluju kao dodatni registri

Usvajamo sledeće

- * množenje traje 4 clk.
- * prva load traje 8 clocks (recimo zbog keš promašaja)
- * druga load traje 4 clocks (pogodak).
- * inicijalno $R1 = 80$.
- * branch je predviđen kao taken (obavlja se).
- * posmatramo izvršenje prve dve iteracije

Loop primer

Instruction status:

Exec Write

<i>ITER</i>	Instruction		<i>j</i>	<i>k</i>	<i>Issue CompResult</i>		<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1		Load1	No		
1	MULTD	F4	F0	F2		Load2	No		
1	SD	F4	0	R1		Load3	No		
2	LD	F0	0	R1		Store1	No		
2	MULTD	F4	F0	F2		Store2	No		
2	SD	F4	0	R1		Store3	No		

Reservation Stations:

$$S1 \quad S2 \quad RS$$

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	No						SUBI	R1	R1	#8
	Mult2	No						BNEZ	R1	Loop	

Register result status

<i>Clock</i>	R1		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
0	80	<i>Fu</i>									

Loop primer Cycle 1

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	CompResult		Busy	Addr	Fu
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2		Load2	No		
1	SD	F4	0	R1		Load3	No		
2	LD	F0	0	R1		Store1	No		
2	MULTD	F4	F0	F2		Store2	No		
2	SD	F4	0	R1		Store3	No		

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:	
	Add1	No						LD	F0 0 R1
	Add2	No						MULTD	F4 F0 F2
	Add3	No						SD	F4 0 R1
	Mult1	No						SUBI	R1 R1 #8
	Mult2	No						BNEZ	R1 Loop

Register result status

Clock	R1		F0	F2	F4	F6	F8	F10	F12	...	F30
1	80	Fu	Load1								

Prva load izdata

Loop primer Cycle 2

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	CompResult		Busy	Addr	<i>Fu</i>
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2	2	Load2	No		
1	SD	F4	0	R1		Load3	No		
2	LD	F0	0	R1		Store1	No		
2	MULTD	F4	F0	F2		Store2	No		
2	SD	F4	0	R1		Store3	No		

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	Yes	Multd		R(F2)	Load1		SUBI	R1	R1	#8
	Mult2	No						BNEZ	R1	Loop	

Register result status

Clock	R1		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
2	80	<i>Fu</i>	Load1		Mult1						

MULTD izdata

Loop primer Cycle 3

Instruction status:

					<i>Exec Write</i>				
<i>ITER</i>	<i>Instruction</i>		<i>j</i>	<i>k</i>	<i>Issue</i>	<i>CompResult</i>	<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1		Load1	Yes	80
1	MULTD	F4	F0	F2	2		Load2	No	
1	SD	F4	0	R1	3		Load3	No	
2	LD	F0	0	R1			Store1	Yes	80
2	MULTD	F4	F0	F2			Store2	No	Mult1
2	SD	F4	0	R1			Store3	No	

Reservation Stations:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>	
	Add1	No						LD	F0 0 R1
	Add2	No						MULTD	F4 F0 F2
	Add3	No						SD	F4 0 R1
	Mult1	Yes	Multd					SUBI	R1 R1 #8
	Mult2	No						BNEZ	R1 Loop

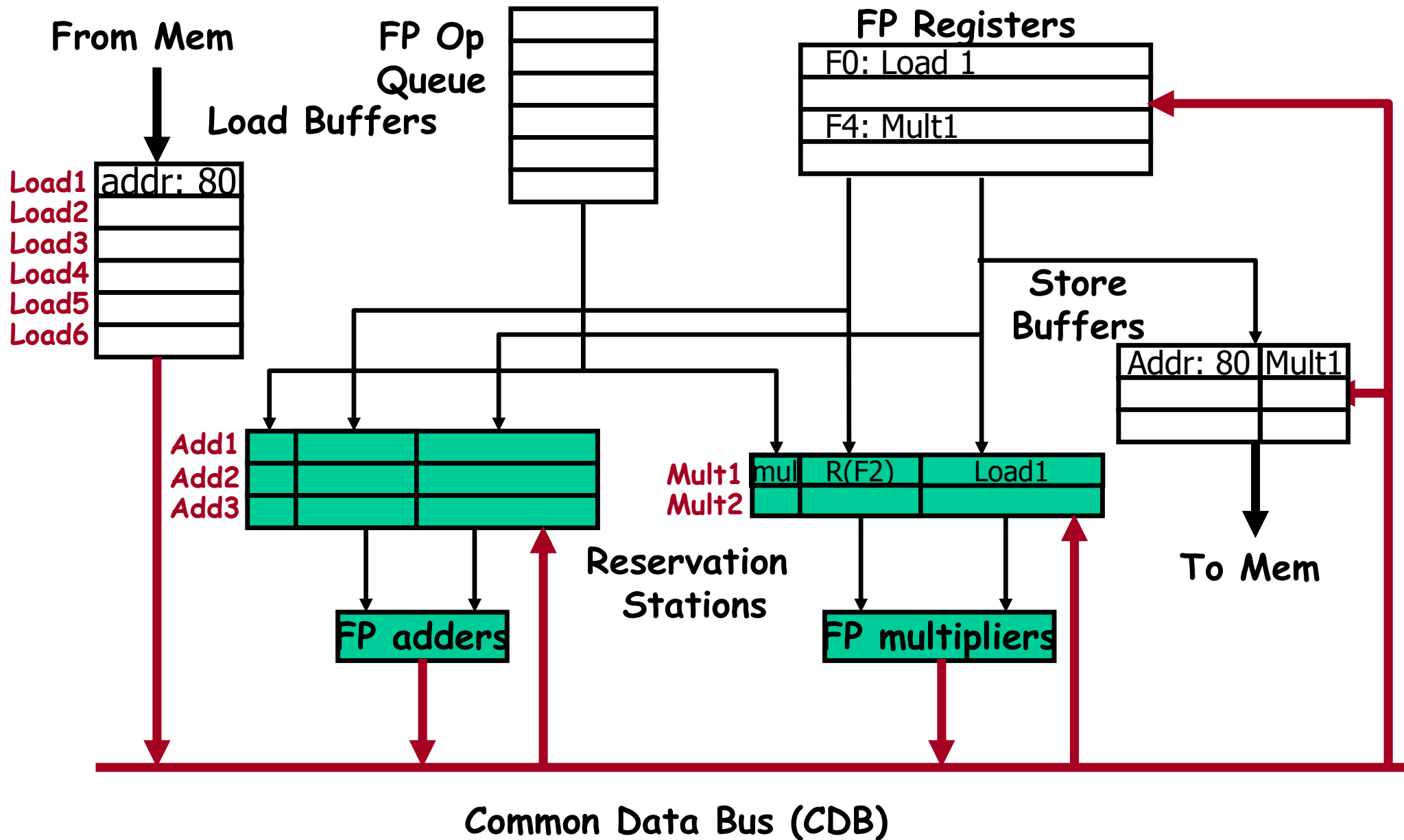
Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
3	80	<i>Fu</i>	Load1		Mult1						

* Izdata je SD

* implicitno preimenovanje: u MULT1 ne figurišu imena registara

Sta to fizički znači?



Loop primer Cycle 4

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>		<i>j</i>	<i>k</i>	<i>Issue</i>	<i>CompResult</i>	<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1		Load1	Yes	80
1	MULTD	F4	F0	F2	2		Load2	No	
1	SD	F4	0	R1	3		Load3	No	
2	LD	F0	0	R1			Store1	Yes	80
2	MULTD	F4	F0	F2			Store2	No	Mult1
2	SD	F4	0	R1			Store3	No	

Reservation Stations:

S1 S2 RS

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	Yes	Multd		R(F2)	Load1		SUBI	R1	R1	#8
	Mult2	No						BNEZ	R1	Loop	

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
4	80	<i>Fu</i>	Load1		Mult1						

* izdavanje SUBI Instrukcije

Loop primer Cycle 5

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>		<i>j</i>	<i>k</i>	<i>Issue</i>	<i>CompResult</i>		<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1		Load1	Yes	80	
1	MULTD	F4	F0	F2	2		Load2	No		
1	SD	F4	0	R1	3		Load3	No		
2	LD	F0	0	R1			Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2			Store2	No		
2	SD	F4	0	R1			Store3	No		

Reservation Stations:

S1 S2 RS

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	Yes	Multd		R(F2)	Load1		SUBI	R1	R1	#8
	Mult2	No						BNEZ	R1	Loop	

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
5	72	<i>Fu</i>	Load1		Mult1						

* i, BNEZ instrukcije

Loop primer Cycle 6

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	CompResult		Busy	Addr	Fu
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2	2	Load2	Yes	72	
1	SD	F4	0	R1	3	Load3	No		
2	LD	F0	0	R1	6	Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2		Store2	No		
2	SD	F4	0	R1		Store3	No		

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:	
	Add1	No						LD	F0 0 R1
	Add2	No						MULTD	F4 F0 F2
	Add3	No						SD	F4 0 R1
	Mult1	Yes	Multd		R(F2)	Load1		SUBI	R1 R1 #8
	Mult2	No						BNEZ	R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
6	72	Fu	Load2		Mult1					

* Izdaje se sledeća LD; u F0 se nikad ne upiše rezultat Load1 sa lokacije 80

Loop Example Cycle 7

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	CompResult		Busy	Addr	<i>Fu</i>
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2	2	Load2	Yes	72	
1	SD	F4	0	R1	3	Load3	No		
2	LD	F0	0	R1	6	Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2	7	Store2	No		
2	SD	F4	0	R1		Store3	No		

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:	
	Add1	No						LD	F0 0 R1
	Add2	No						MULTD	F4 F0 F2
	Add3	No						SD	F4 0 R1
	Mult1	Yes	Multd		R(F2)	Load1		SUBI	R1 R1 #8
	Mult2	Yes	Multd		R(F2)	Load2		BNEZ	R1 Loop

Register result status

Clock	R1		F0	F2	F4	F6	F8	F10	F12	...	F30
7	72	<i>Fu</i>	Load2		Mult2						

- * Izadje se i druga MULTD; Registarski fajl potpuno izolovan od iteracije 1 (u F0 se nikad ne upiše rezultat Load1, u F4 se nikad ne upiše rezultat MUL1)

Loop Example Cycle 8

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>CompResult</i>		<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2	2	Load2	Yes	72	
1	SD	F4	0	R1	3	Load3	No		
2	LD	F0	0	R1	6	Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2	7	Store2	Yes	72	Mult2
2	SD	F4	0	R1	8	Store3	No		

Reservation Stations:

S1 S2 RS

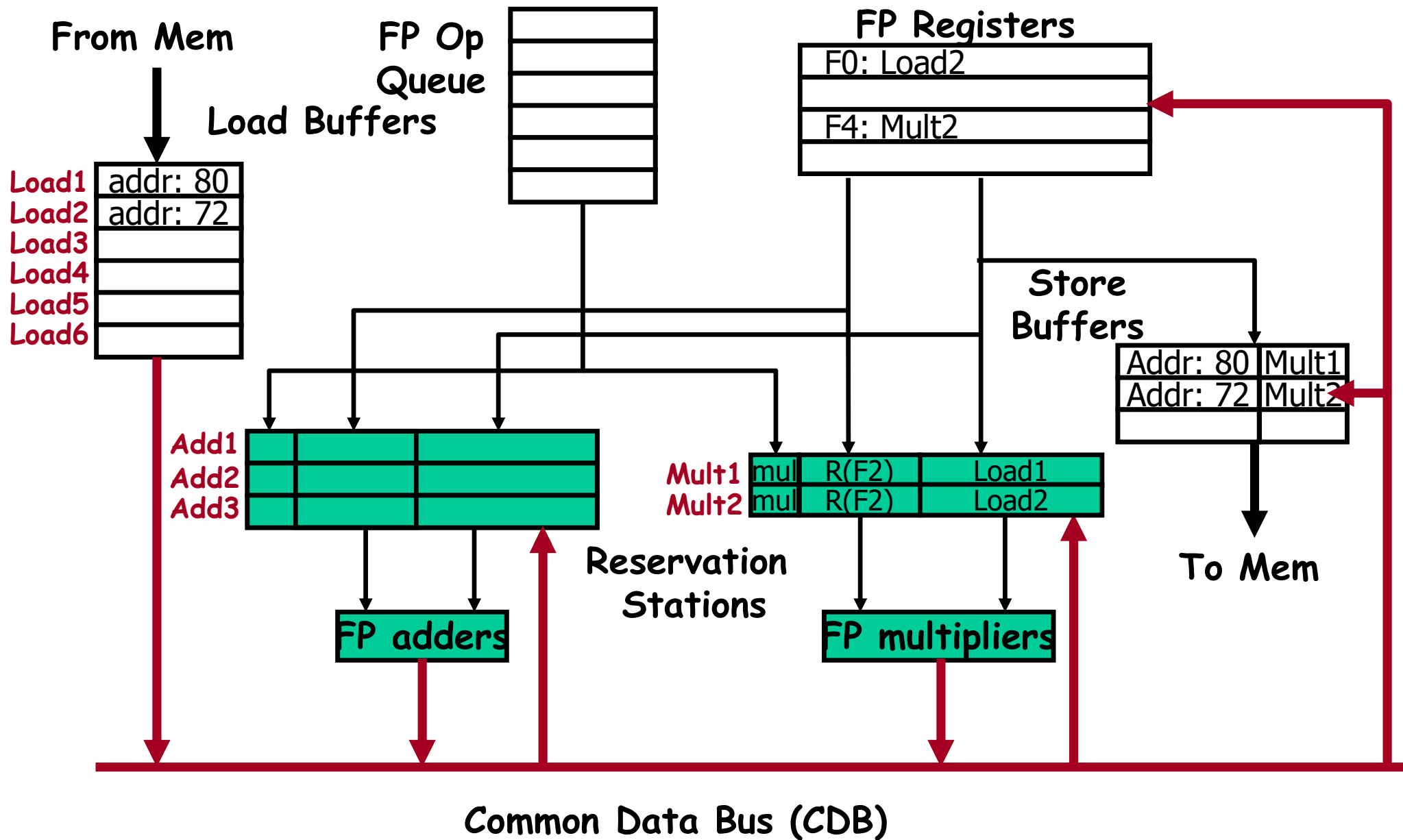
<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	Yes	Multd			R(F2)	Load1	SUBI	R1	R1	#8
	Mult2	Yes	Multd			R(F2)	Load2	BNEZ	R1	Loop	

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
8	72	<i>Fu</i>	Load2		Mult2						

* Izdata je i druga SD; prva i druga iteracija se potpuno preklapaju

Sta to fizički znači?



Loop primer Cycle 9

Instruction status:

Exec Write

<i>ITER</i>	Instruction		<i>j</i>	<i>k</i>	<i>Issue CompResult</i>			<i>Busy</i>	<i>Addr</i>	<i>Fu</i>											
1	LD	F0	0	R1	<table><tr><td>1</td><td>9</td></tr><tr><td>2</td><td></td></tr><tr><td>3</td><td></td></tr><tr><td>6</td><td></td></tr><tr><td>7</td><td></td></tr><tr><td>8</td><td></td></tr></table>	1	9	2		3		6		7		8		Load1	Yes	80	
1	9																				
2																					
3																					
6																					
7																					
8																					
1	MULTD	F4	F0	F2	2	Load2	Yes	72													
1	SD	F4	0	R1	3	Load3	No														
2	LD	F0	0	R1	6	Store1	Yes	80	Mult1												
2	MULTD	F4	F0	F2	7	Store2	Yes	72	Mult2												
2	SD	F4	0	R1	8	Store3	No														

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:				
	Add1	No						LD	F0	0	R1	
	Add2	No						MULTD	F4	F0	F2	
	Add3	No						SD	F4	0	R1	
	Mult1	Yes	Multd			R(F2)	Load1	SUBI	R1	R1	#8	←
	Mult2	Yes	Multd			R(F2)	Load2	BNEZ	R1	Loop		

Register result status

Clock	R1		F0	F2	F4	F6	F8	F10	F12	...	F30
9	72	Fu	Load2		Mult2						

- * Load1 okončana: ko čeka na rezultat?
- * izdavanje SUBI

Loop primer Cycle 10

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	<i>Issue CompResult</i>			<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2			Load2	Yes 72
1	SD	F4	0	R1	3			Load3	No
2	LD	F0	0	R1	6	10		Store1	Yes 80 Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes 72 Mult2
2	SD	F4	0	R1	8			Store3	No

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
4	Mult1	Yes	Multd	M[80]	R(F2)			SUBI	R1	R1	#8
	Mult2	Yes	Multd		R(F2)	Load2		BNEZ	R1	Loop	

Register result status

Clock	R1		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
10	64	<i>Fu</i>	Load2	Mult2							

- * Load2 okončana: ko čeka rezultat?
- * izdavanje BNEZ

Loop primer Cycle 11

Instruction status:

Exec Write

ITER	Instruction		<i>j</i>	<i>k</i>	<i>Issue CompResult</i>				<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No		
1	MULTD	F4	F0	F2	2			Load2	No		
1	SD	F4	0	R1	3			Load3	Yes	64	
2	LD	F0	0	R1	6	10	11	Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes	72	Mult2
2	SD	F4	0	R1	8			Store3	No		

Reservation Stations:

				<i>S1</i>	<i>S2</i>	<i>RS</i>				
<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>		
	Add1	No						LD	F0	0 R1
	Add2	No						MULTD	F4	F0 F2
	Add3	No						SD	F4	0 R1
3	Mult1	Yes	Multd	M[80]	R(F2)			SUBI	R1	R1 #8
4	Mult2	Yes	Multd	M[72]	R(F2)			BNEZ	R1	Loop

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
11	64	<i>Fu</i>	Load3	Mult2							

* sledeća load u sekvenci (iz treće iteracije)

Loop Example Cycle 12

Instruction status:

Exec Write

<i>ITER</i>	Instruction			<i>j</i>	<i>k</i>	<i>Issue CompResult</i>			<i>Busy</i>	<i>Addr</i>	<i>Fu</i>	
1	LD	F0	0	R1	1	9	10	Load1	No			
1	MULTD	F4	F0	F2				2	Load2			No
1	SD	F4	0	R1				3	Load3			Yes
2	LD	F0	0	R1	6	10	11	Store1	Yes	80	Mult1	
2	MULTD	F4	F0	F2	7			Store2	Yes	72	Mult2	
2	SD	F4	0	R1	8			Store3	No			

Reservation Stations:

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
2	Mult1	Yes	Multd	M[80]	R(F2)			SUBI	R1	R1	#8
3	Mult2	Yes	Multd	M[72]	R(F2)			BNEZ	R1	Loop	

Register result status

Clock	R1	<i>Fu</i>	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
12	64		Load3	Mult2							

* *zašto se ne izdaje treća MULDD?*

Loop Example Cycle 13

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2			Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes 72 Mult2
2	SD	F4	0	R1	8			Store3	No

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2
	Add3	No						SD F4 0 R1
1	Mult1	Yes	Multd	M[80]	R(F2)			SUBI R1 R1 #8
2	Mult2	Yes	Multd	M[72]	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
13	64	<i>Fu</i>	Load3	Mult2						

Mult1 i Mult2 su još aktivne. Zašto se ne izdaje treća SD?

Loop primer Cycle 14

Instruction status:

Exec Write

ITER	Instruction		<i>j</i>	<i>k</i>	<i>Issue CompResult</i>				<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No		
1	MULTD	F4	F0	F2	2	14		Load2	No		
1	SD	F4	0	R1	3			Load3	Yes	64	
2	LD	F0	0	R1	6	10	11	Store1	Yes	80	Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes	72	Mult2
2	SD	F4	0	R1	8			Store3	No		

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
0	Mult1	Yes	Multd	M[80]	R(F2)			SUBI	R1	R1	#8
1	Mult2	Yes	Multd	M[72]	R(F2)			BNEZ	R1	Loop	

Register result status

Clock	R1		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
14	64	<i>Fu</i>	Load3	Mult2							

* Mult1 okončana. ko čeka?

Loop primer Cycle 15

Instruction status:

Exec Write

ITER	Instruction		<i>j</i>	<i>k</i>	<i>Issue CompResult</i>				<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No		
1	MULTD	F4	F0	F2	2	14	15	Load2	No		
1	SD	F4	0	R1	3			Load3	Yes	64	
2	LD	F0	0	R1	6	10	11	Store1	Yes	80	[80]*R2
2	MULTD	F4	F0	F2	7	15		Store2	Yes	72	Mult2
2	SD	F4	0	R1	8			Store3	No		

Reservation Stations:

				<i>S1</i>	<i>S2</i>	<i>RS</i>				
<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>		
	Add1	No						LD	F0	0 R1
	Add2	No						MULTD	F4	F0 F2
	Add3	No						SD	F4	0 R1
	Mult1	No						SUBI	R1	R1 #8
0	Mult2	Yes	Multd	M[72]	R(F2)			BNEZ	R1	Loop

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
15	64	<i>Fu</i>	Load3	Mult2							

- * Mult1 slobodna;
- * Mult2 okončana. ko čeka?

Loop primer Cycle 16

Instruction status:

Exec Write

ITER	Instruction	<i>j</i>	<i>k</i>	Issue	Comp	Result		Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No	
1	MULTD	F4	F0	F2	2	14	15	Load2	No	
1	SD	F4	0	R1	3			Load3	Yes	64
2	LD	F0	0	R1	6	10	11	Store1	Yes	80 [80]*R2
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes	72 [72]*R2
2	SD	F4	0	R1	8			Store3	No	

Reservation Stations:

S1 S2 RS

Time	Name	Busy	Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	Code:			
	Add1	No						LD	F0	0	R1
	Add2	No						MULTD	F4	F0	F2
	Add3	No						SD	F4	0	R1
	Mult1	Yes	Multd		R(F2)	Load3		SUBI	R1	R1	#8
	Mult2	No						BNEZ	R1	Loop	

Register result status

Clock	R1		F0	F2	F4	F6	F8	F10	F12	...	F30
16	64	Fu	Load3		Mult1						

*MULTD iz treće iteracije se izdaje (FU Mult1 aktivna)

Loop primer Cycle 17

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Comp</i>	<i>Result</i>	<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 [80]*R2
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes 72 [72]*R2
2	SD	F4	0	R1	8			Store3	Yes 64 Mult1

Reservation Stations:

S1 S2 RS

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1
Mult1	Yes	Multd			R(F2)	Load3		SUBI R1 R1 #8
Mult2	No							BNEZ R1 Loop

Register result status

<i>Clock</i>	<i>R1</i>	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
17	64	<i>Fu</i>	Load3	Mult1						

Izdaje se treća SD

Loop primer Cycle 18

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Comp</i>	<i>Result</i>	<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3	18		Load3	Yes
2	LD	F0	0	R1	6	10	11	Store1	Yes
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes
2	SD	F4	0	R1	8			Store3	Yes
									64
									80
									72
									Mult1

Reservation Stations:

S1 S2 RS

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>
Add1	No							LD
Add2	No							F0
Add3	No							0
Mult1	Yes	Multd						R1
Mult2	No							F2
								F4
								0
								R1
								#8
								Loop

Register result status

<i>Clock</i>	<i>R1</i>	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
18	64	<i>Fu</i>	Load3	Mult1						

Loop primer Cycle 19

Instruction status:

Exec Write

<i>ITER</i>	<i>Instruction</i>	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Comp</i>	<i>Result</i>	<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3	18	19	Load3	Yes
2	LD	F0	0	R1	6	10	11	Store1	No
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes
2	SD	F4	0	R1	8	19		Store3	Yes

Reservation Stations:

S1 S2 RS

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>
Add1	No							LD
Add2	No							F0
Add3	No							0
Mult1	Yes	Multd		R(F2)	Load3			R1
Mult2	No							F4

Register result status

<i>Clock</i>	<i>R1</i>	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
19	64	<i>Fu</i>	Load3	Mult1						

Loop primer Cycle 20

Instruction status:

					<i>Exec Write</i>					
<i>ITER</i>	<i>Instruction</i>	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Comp</i>	<i>Result</i>		<i>Busy</i>	<i>Addr</i>	<i>Fu</i>
1	LD	F0	0	R1	1	9	10	Load1	No	
1	MULTD	F4	F0	F2	2	14	15	Load2	No	
1	SD	F4	0	R1	3	18	19	Load3	Yes	64
2	LD	F0	0	R1	6	10	11	Store1	No	
2	MULTD	F4	F0	F2	7	15	16	Store2	No	
2	SD	F4	0	R1	8	19	20	Store3	Yes	64
										Mult1

Reservation Stations:

					<i>S1 S2 RS</i>					
<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>	<i>Code:</i>		
	Add1	No						LD	F0	0 R1
	Add2	No						MULTD	F4	F0 F2
	Add3	No						SD	F4	0 R1
	Mult1	Yes	Multd		R(F2)	Load3		SUBI	R1	R1 #8
	Mult2	No						BNEZ	R1	Loop

Register result status

<i>Clock</i>	<i>R1</i>		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	<i>...</i>	<i>F30</i>
20	64	<i>Fu</i>	Load3		Mult1						

Ponovo imamo izdavanje po redosledu, izvršenje i okončanje van redosleda pribavljanja

Zašto Tomasulo može da preklapa iteracije petlje?

* Preimenovanje registara

- Različite iteracije koriste različite fizičke destinacije za registre (dinamičko odmotavanje petlje)
- imena registara zamenjena imenima RS koje generišu rezultat
- efektivno povećava veličinu registarskog fajla

Superskalarni i VLIW procesori

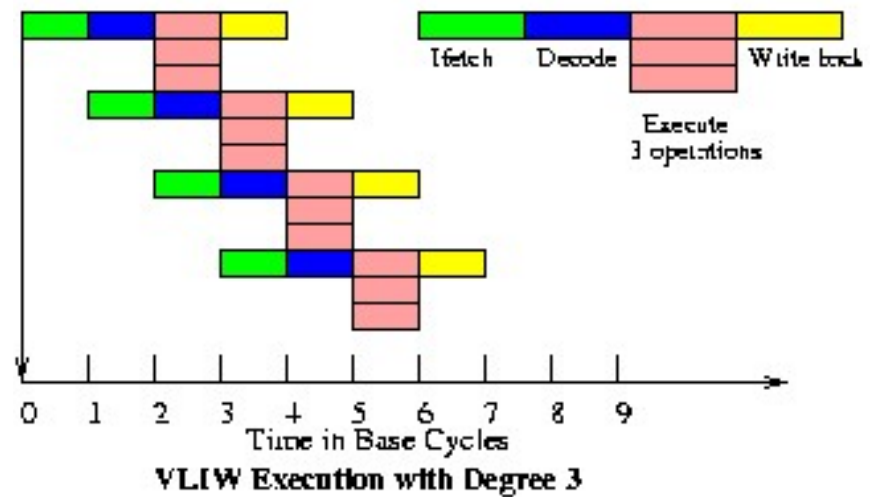
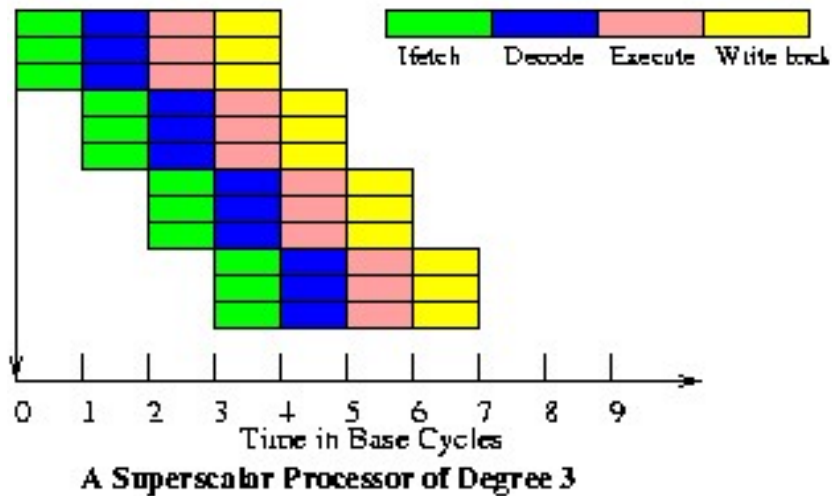
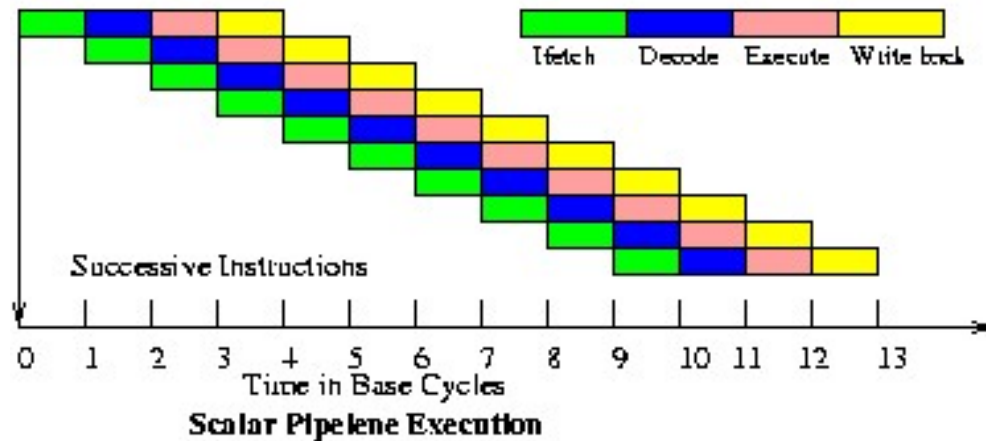
Ciljevi SS i VLIW

- * Tehnika odmotavanja petlje ima za cilj da poveća količinu raspoloživog ILP-a
- * Scoreboard i Tomasulo tehnike imaju za cilj da postignu idealni CPI od 1inst/clk
- * BPB i BTB imaju zacilj da redukuju kašnjenje uzrokovano naredbama grananja
- * CPI ne može biti < 1 ako se pribavlja i izdaje jedna instrukcija u clk ciklusu
- * Da bi se CPI dalje redukovao potrebno je pribaviti i izdati više od jedne instrukcije u jednom clk ciklusu.
 - Rad superskalarnih (SS) i VLIW (Very Long Instruction Word) procesora se zasniva na ovoj ideji

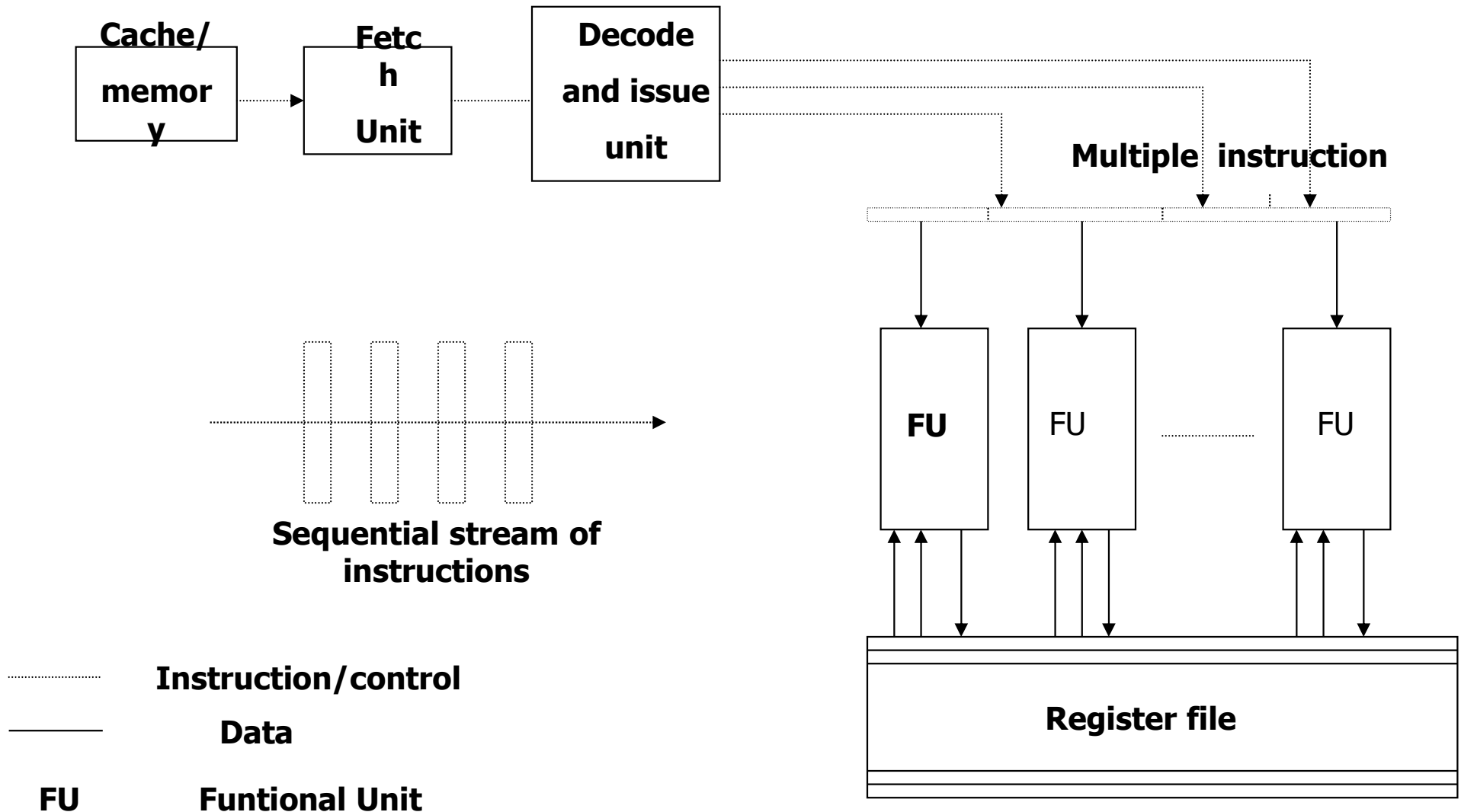
SS i VLIW

- * SS procesori mogu da izdaju različit broj instrukcija po clk ciklusu
 - Kod tipičnog SS procesora hw može da izda od 1 do 8 instrukcija u jednom clk ciklusu (u zavisnosti od raspoloživog ILP-a)
 - SS procesori mogu da koriste statičko planiranje izvršenja instrukcija (uz pomoć kompajlera) ili dinamičko zasnovano na Sc tehnici i Tomasulovom algoritmu
 - IBM PowerPC, Sun UltraSparc, DEC Alpha, HP 8000, MIPS 10000, AMD K5
- * VLIW (zovu se još i EPIC – Explicitly Parallel Instruction Computer) izdaje fiksni broj instrukcija koje su formatirane kao jedna velika instrukcija ili kao fiksni instrukcioni paket. (paralelne instrukcije su grupisane u blokove)
 - VLIW su po definiciji sa statičkim planiranjem izvršenja instrukcija
 - Intel Itanium

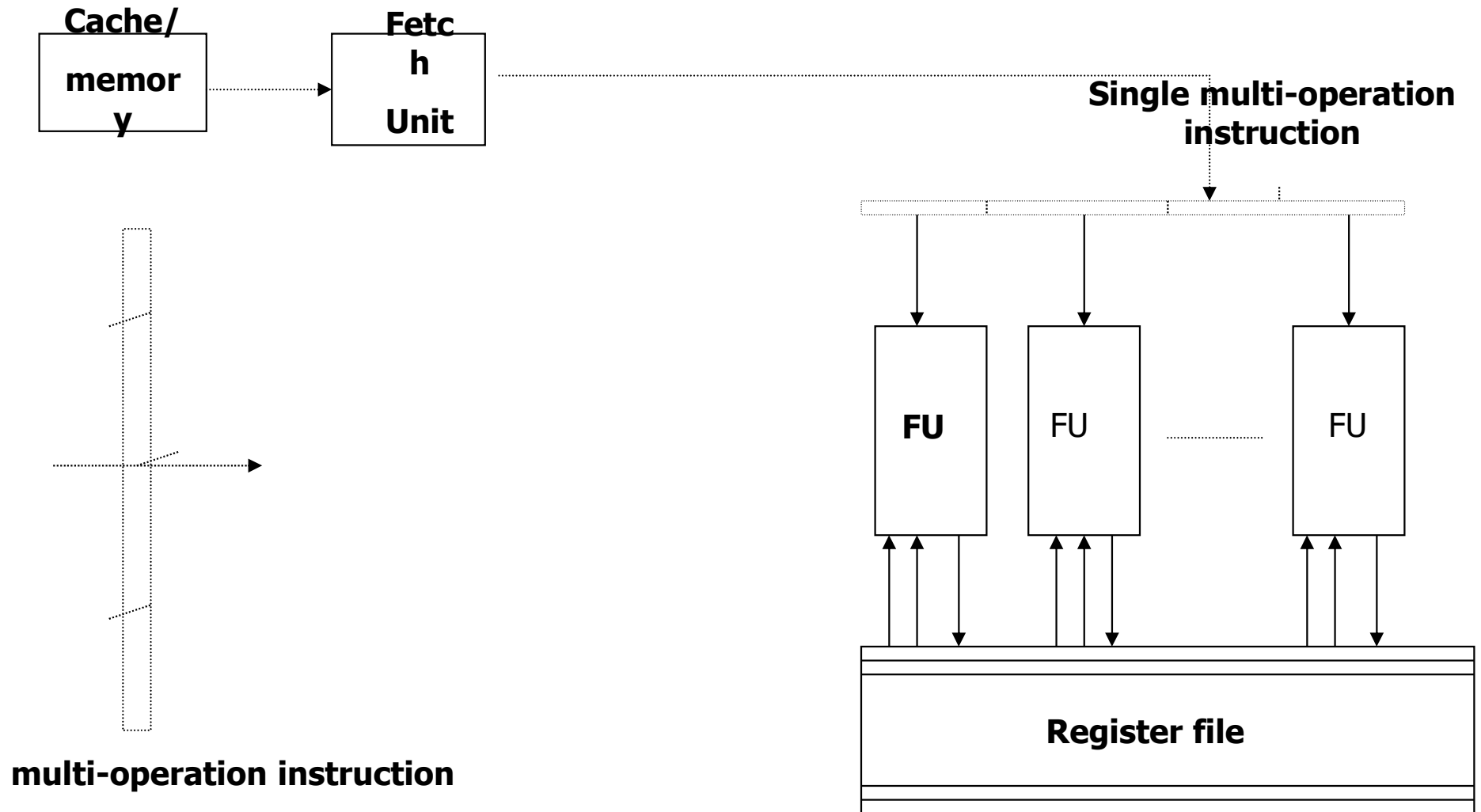
Superskalarni i VLIW procesori



Superskalarni procesori



VLIW procesori



SS primer

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

Instrukcija koja generiše rezultat	Instrukcija koja koristi rezultat	Latentnost u Clk
FP ALU operacija	druga FP ALU operacija	3
FP ALU operacija	Store double (SD)	2
Load double (LD)	FP ALU operacija	1
Load double (LD)	Store double (SD)	0

Latencije FP operacija

SS primer

- * Instrukcije koje se mogu jednovremeno izdavati moraju biti nezavisne
- * U toku jednog clk ciklusa samo jedno obraćanje memoriji je moguće
 - Ako neka instrukcija u nizu zavisi od neke prethodne ili ne zadovoljava kriterijum za izdavanje, samo instrukcije koje joj prethode biće izdate
- * **PRIMER: izdavanje dve instrukcije**
 - **Prva:** Jedna load/store/branch/integer-ALU op.
 - **Druga:** Jedna floating-point op.
 - Uvek je prva instrukcija u paru Integer instrukcija
 - Druga instrukcija se može izdati samo ako se prva može izdati
 - Izdavanje Integer instrukcije paralelno sa FP operacijom je mnogo manje zahtevno od izdavanja dve proizvoljne instrukcije – **koriste različite funkcionalne jedinice i različite skupove registara** (problem može da nastupi ako su u pitanju load i store u FP registe)

SS primer

- * Latentnost load je 1 clk, pa se kod SS procesora rezultat load ne može koristiti u istom i narednom clk ciklusu (tj. u naredne 3 instrukcije)
- * Kašnjenje za branch takodje može biti 3 instrukcije, jer branch mora biti prva u paru
- * Za odabrani primer potrebno je 5 puta odmotati petlju da bi se izbeglo zaustavljanje protočnog sistema kod SS procesora (kod skalarnog - 4 puta)

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

FIGURE 4.26 Superscalar pipeline in operation.

Odmotavanje petlje koje minimizira zastoje kod skalarnog procesora

1	Loop:	LD	F0 , 0 (R1)	LD to ADDD: 1 Cycle
2		LD	F6 , -8 (R1)	ADDD to SD: 2 Cycles
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	
12		SUBI	R1 , R1 , #32	
11		SD	16 (R1) , F12	
13		BNEZ	R1 , LOOP	
14		SD	8 (R1) , F16	

14 clk ciklusa, ili 3.5 po iteraciji

Superskalarno izvršenje

	<i>Integer instruction</i>	<i>FP instruction</i>	<i>Clock cycle</i>
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		10
	SD 16(R1),F16		9
	BNEZ R1,LOOP		11
	SD 8(R1),F20		12

* 12 clk, ili 2.4 clk po iteraciji

Primeri nekih SS procesora

* PowerPC 604

- six independent execution units:
 - Branch execution unit
 - Load/Store unit
 - 3 Integer units
 - Floating-point unit
- in-order issue
- register renaming

* Power PC 620

- provides in addition to the 604 out-of-order issue

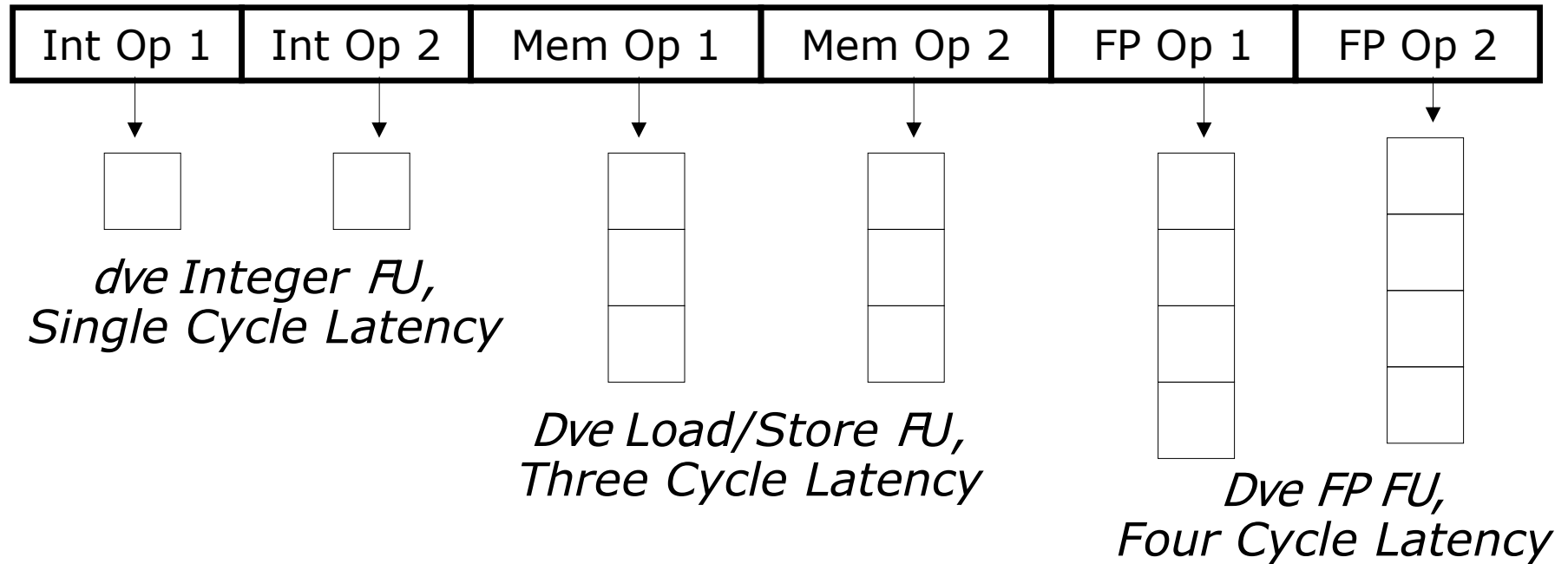
* Pentium

- three independent execution units:
 - 2 Integer units
 - Floating point unit
- in-order issue

VLIW

- * Upravljanje se ostvaruje veoma dugim instrukcijama
 - Instrukcija sadrži upravljačko (control) polje za svaku od funkcionalnih jedinica
- * Dužina instrukcije zavisi od
 - Broja funkcionalnih jedinica (5-30 FU)
 - Dužine polja control za svaku FU je 16-32 bita.
 - Dužina instrukcije od 256 do 1024 bita
- * Planiranje izvršenja instrukcija (tj. Formiranje duge instrukcije) se obavlja softverski (kompajler)
- * Manja hardverska kompleksnost se može iskoristiti
 - Da se poveća učestanost kojom se taktuje procesor (clock rate)
 - Poveća nivo paralelizma kroz ugradnju većeg broja FU
- * Mane:
 - u proseku samo jedan broj upravljačkih (control) polja se realno koristi (prazna polja se popunjavaju NOP operacijama).
 - Složeni kompajleri
 - Kompajler mora da vodi računa o hardverskim detaljima
 - Broj FU, latentnost FU, period iniciranja FU,..
 - Keš promašaji: kompajler mora da uzme u obzir najgori mogući slučaj
 - Zavisnost kompajlera od hardvera sprečava korišćenje istog kompajlera za familiju VLIW procesora

VLIW: Very Long Instruction Word



- * Više operacija u jednom instrukcionom paketu
- * Svako polje u instrukcionom paketu je za fiksnu FU
- * Latencije FU su konstantne
- * Između instrukcija koje se nalaze u jednom paketu nema nikakvih zavisnosti

VLIW primer

- * Neka instrukcioni paket ima 5 instrukcija
 - 2 FP, 2 Memory, 1 branch/integer
- * Kompajler detektuje sve hazrde
 - po definiciji sve instrukcije koje kompajler postavi u jednu veliku instrukciju (paket) su nezavisne, tj. mogu se paralelno izvršavati
- * Svi slotovi u instrukcionom paketu ne moraju biti uvek popunjeni

Odmotavanje petlje i VLIW

<i>Memory reference 1</i>	<i>Memory reference 2</i>	<i>FP operation 1</i>	<i>FP op. 2</i>	<i>Int. op/ branch</i>	<i>Clock</i>
LD F0, 0 (R1)	LD F6, -8 (R1)				1
LD F10, -16 (R1)	LD F14, -24 (R1)				2
LD F18, -32 (R1)	LD F22, -40 (R1)	ADDD F4, F0, F2	ADDD F8, F6, F2		3
LD F26, -48 (R1)		ADDD F12, F10, F2	ADDD F16, F14, F2		4
		ADDD F20, F18, F2	ADDD F24, F22, F2		5
SD 0 (R1), F4	SD -8 (R1), F8	ADDD F28, F26, F2			6
SD -16 (R1), F12	SD -24 (R1), F16				7
SD -32 (R1), F20	SD -40 (R1), F24			SUBI R1, R1, #48	8
SD 0 (R1), F28				BNEZ R1, LOOP	9

- petlja odmotana 7 puta da bi se izbegli zastoji
- 9 clk, ili 1.3 clk po iteraciji
- 2.5 operacije po clk, 50% efikasnosti

Napomena: Potrebno je više registara kod VLIW nego kod SS (15 naspram 6 kod SS)

Prednosti VLIW

Kompajler priprema fiksne instrukcione pakete koji sadrže više operacija, tj. pravi plan izvršenja

- Zavisnosti detektuje kompajler i koristi ih da planira izvršenje instrukcija
- Funkcionalne jedinice dodeljuje kompajler na osnovu pozicije u instrukcionom paketu
- Kompajler generiše kod koji nema nikakvih hazarda tako da nema potrebe za hardverom za detekciju zavisnosti ili planiranjem izvršenja

Nedostaci VLIW

Kompatibilnost koda

- VLIW code se ne može korektno izvršavati na mašini sa različitim brojem funkcionalnih jedinica i/ili različitim latencijama FU.

Gustina koda

- Niska iskorišćenost slotova unutar dugačke instrukcije (uglavnom nop-ovi)