

پاسخ تمرین سری سوم

فهرست مطالب

۲	۱	سیگنال‌های کنترلی در یک پردازنده‌ی Single Cycle
۲	۲	پیش‌نیازهای داده‌ای
۲	۱.۲	وابستگی‌ها
۳	۲.۲	افزودن NOOP بدون وجود Full Forwarding
۳	۳.۲	افزودن NOOP با وجود Full Forwarding
۴	۳	نوشتن یک Pipeline Stage
۵	۴	رجیسترهای Pipeline ها
۵	۱.۴	رجیسترهای Pipeline
۶	۲.۴	اتفاقات هنگام اجرا
۶	۵	Execution در یک CPU عه Pipeline شده
۶	۶	Hazard ها
۶	۷	ارتباط بین Forwarding, Hazard و ISA Desing

۱ سیگنال‌های کنترلی در یک پردازنده‌ی Single Cycle

	RegWrite	MemRead	ALUMux	MemWrite	ALUOp	RegMux	Branch
a.	1	0	0 (Reg)	0	AND	1 (ALU)	0
b.	0	0	1 (Imm)	1	ADD	X	0

۲ پیش‌نیازهای داده‌ای

۱.۲ وابستگی‌ها

	Instruction Sequence	Dependencies
a.	I1: SW R16, -100(R6) I2: LW R4, 8(R16) I3: ADD R5, R4, R4	RAW on R4 from I2 to I3
b.	I1: OR R1, R2, R3 I2: OR R2, R1, R4 I3: OR R1, R1, R2	RAW on R1 from I1 to I2 and I3 RAW on R2 from I2 to I3 WAR on R2 from I1 to I2 WAR on R1 from I2 to I3 WAW on R1 from I1 to I3

برای مطالعه‌ی معنی وابستگی‌ها به این لینک مراجعه کنید:

https://en.wikipedia.org/wiki/Data_dependency

۲.۲ افزودن NOOP بدون وجود Full Forwarding

	Instruction Sequence	
a.	SW R16, -100(R6) LW R4, 8(R16) NOOP NOOP ADD R5, R4, R4	Delay I3 to avoid RAW hazard on R4 from I2
b.	OR R1, R2, R3 NOOP NOOP OR R2, R1, R4 NOOP NOOP OR R1, R1, R2	Delay I2 to avoid RAW hazard on R1 from I1 Delay I3 to avoid RAW hazard on R2 from I2

۳.۲ افزودن NOOP با وجود Full Forwarding

	Instruction Sequence	
a.	SW R16, -100(R6) LW R4, 8(R16) NOOP ADD R5, R4, R4	Delay I3 to avoid RAW hazard on R4 from I2 Value for R4 is forwarded from I2 now
b.	OR R1, R2, R3 OR R2, R1, R4 OR R1, R1, R2	No RAW hazard on R1 from text (forwarded) No RAW hazard on R2 from text (forwarded)

۳ نوشتن یک Pipeline Stage

	Instruction
a.	SW R16, 12(R6) LW R16, 8(R6) BEQ R5, R4, Label ; Assume R5 != R4 ADD R5, R1, R4 SLT R5, R15, R4

IF	ID	EXE	MEM	WB									
	IF	ID	EXE	MEM	WB								
		IF	ID	EXE	MEM	WB							
			**	**	IF	ID	EXE	MEM	WB				
						IF	ID	EXE	MEM	WB			

	Instruction
b.	SW R2, 0(R3) OR R1, R2, R3 BEQ R2, R0, Label ; Assume R2 == R0 OR R2, R2, R0 Label: ADD R1, R4, R3

IF	ID	EXE	MEM	WB				
	IF	ID	EXE	MEM	WB			
		IF	ID	EXE	MEM	WB		
			**	IF	ID	EXE	MEM	WB

۴ رجیسترهای Pipeline ها

۱.۴ رجیسترهای Pipeline

- برای هر دستورالعمل، رجیستر IF/ID مقدار $PC + 4$ و خود دستورالعمل را نگه می دارد.
- رجیستر ID/EX تمام سیگنال های کنترلی را برای
 - EX،
 - MEM،
 - WB،
 - $PC + 4$ ،
 - دو مقدار خوانده شده از رجیسترها،
 - ۱۶ بیت پایین دستورالعمل که sign-extend شده است و
 - قسمت های Rd و Rt از دستورالعمل (حتی برای دستورالعمل هایی که فرمت آنها از این فیلدها استفاده نمی کند).
- رجیستر EX/MEM سیگنال های کنترلی را برای مراحل
 - MEM،
 - WB،
 - $PC + 4 + \text{Offset}$ (جایی که آفست ۱۶ بیت دستورالعمل ها sign-extend شده است، حتی برای دستورالعمل هایی که فیلد آفست ندارند)،
 - نتیجه ALU و مقدار خروجی صفر آن،
 - مقداری که از ثبات دوم در مرحله ID خوانده شد (حتی برای دستورالعمل که هرگز به این مقدار نیاز ندارند) و
 - شماره رجیستر مقصد (حتی برای دستورالعمل هایی که نیازی به ثبت ندارند. برای این دستورالعمل ها شماره ثبت مقصد به سادگی یک انتخاب "تصادفی" بین Rd یا Rt است).
- رجیستر MEM/WB
 - سیگنال های کنترل WB،
 - مقدار خوانده شده از حافظه (یا مقداری تصادفی وقتی که چیزی از مموری خوانده نشده است)،
 - نتیجه ALU و
 - عدد رجیستر مقصد.

۲.۴ اتفاقات هنگام اجرا

	EXE	MEM
a.	$-100 + R6$	Write value to memory
b.	$R1 \text{ Or } R0$	Nothing

۵ Execution در یک CPU به Pipeline شده

هیچ سیگنالی در IF و ID وارد نمی‌شود، باقی سیگنال‌ها:

	EXE	MEM	WB
a.	ALUSrc = 1 ALUOp = 00 RegDst = 0	Branch = 0 MemWrite = 0 MemRead = 1	MemToReg = 0 RegWrite = 1
b.	ALUSrc = 0 ALUOp = 10 RegDst = 1	Branch = 0 MemWrite = 0 MemRead = 0	MemToReg = 1 RegWrite = 1

۶ Hazardها

۷ ارتباط بین Forwarding، Hazard و ISA Desing