به نام خدا

دانشکده مهندسی کامپیوتر

معماری کامپیوتر

نیم‌سال دوم ۹۹-۰۰

استاد: دکتر اسدی

تمرین سری هفتم

- پاسخ تمرین‌های تئوری را به صورت فایل تایپ شده در فرمت PDF در قسمت مربوطه در سامانه CW بارگزاری نمایید.
- پرسش‌های خود را می‌توانید در فروم ایجاد شده در سایت درس مطرح کنید.
- هر دانشجو می‌تواند حداکثر دو تمرین را با دو روز تاخیر بدون کاهش نمره ارسال نماید.

سوال ۱ – در دستورات میپس زیر مخاطرات[۱] ممکن و نوع آن ها را مشخص کنید. (فرض کنید مقدار اولیه تمامی ثبات‌ها ۰ است.)

```
Addi $s1, $s1, 4
Add $s2, $s2, $s1
Sub $s3, $s2, $s3
Beq $s2, $s3, P2
Add $s4, $s1, $s0
Sub $s5, $s3, $s2
P2: add $s4, $s5, $s2
```

سوال ۲– کد زیر روی پردازنده‌ای در حال اجراست که فقط از روش insert bubble برای کنترل کردن مخاطرات استفاده می‌کند. حداقل تعداد حباب‌های[۲] مورد نیاز برای جلوگیری از هرگونه مخاطره را با توضیح کامل به دست آورید.

Default values: $s0 = 0, $s1 = 4, Mem[0x0000000i] = (convert(i to decimal)) MOD 9
#of bubbles for control hazard = 3
# of bubbles for data hazard = 2

```
Add  $s2, $s1, $s0
Lw   $s3, $s2, $s0
Add  $s4,$s3, $s0
Beq  $s4, $s2, L2
Sub  $s3, $s2, $s3
L2:
add  $s2, $s1, $s0
```

سوال ۳– فرض کنید یک ماشین خط لوله[۳] ۵ مرحله‌ای داریم. این ماشین از delay slots برای برخورد با وابستگی‌های کنترلی استفاده می‌کند. فرض کنید که اهداف branch و jump در مرحله اجرا (execution) آماده می‌شوند.

الف) برای این پردازنده باید از چند delay slot استفاده کنیم؟

ب) دستورات زیر را در نظر بگیرید. شما می‌تواند دستورات را جابه‌جا کنید تا nopها کمینه شود. چه دستوراتی که باید در delay slot(ها) قرار بگیرند؟ کدهای جدید را بازنویسی کنید:

۱)

```
ADD R5 <- R4, R3
OR R3 <- R1, R2
SUB R7 <- R5, R6
J X
DELAY SLOTS
LW R10 <- (R7)
ADD R6 <- R1, R2
X:
```

[۱] Hazard

[۲] Bubble

[۳] Pipelined

۲)

```
ADD R5 <- R4, R3
OR R3 <- R1, R2
SUB R7 <- R5, R6
BEQ R5 <- R7, X
DELAY SLOTS
LW R10 <- (R7)
ADD R6 <- R1, R2
X:
```

سوال ٤ – فرض کنید یک پردازنده غیر خط لوله‌ای داریم که زمان اجرای هر کلاک در آن ۱۰ نانوثانیه و CPI آن بطور متوسط ۱,۴ باشد. میزان تسریع[۴] را در هر یک از حالات زیر حساب کنید.

الف) اگر این پردازنده را به یک پردازنده خط لوله ای با ۵ مرحله تقسیم کنیم، بهترین تسریع ممکن را بدست آورید.

ب) اگر میزان تاخیر هر مرحله از این معماری خط لوله‌ای مطابق ۱, ۱,۵, ۴, ۳ و ۰,۵ نانوثانیه باشد. بهترین تسریع ممکن را نسبت به پردازنده اصلی بدست آورید.

ج) در ادامه بخش قبل اگر در ۲۰ درصد از مواقع به اندازه ۱ کلاک و در ۵ درصد مواقع به اندازه ۲ کلاک stall داشته باشیم، CPI جدید چیست؟ مقدار تسریع نسبت به پردازنده اولیه را نیز بدست آورید.

سوال ۵– می‌خواهیم قطعه کد زیر را بر روی دو پردازنده‌ی تک سیکلی و خط لوله‌ای اجرا کنیم، در این صورت خواسته‌های زیر را بدست آورید.

الف) زمان اجرای پردازنده‌ی تک سیکلی با فرض اینکه زمان اجرای هر کلاک این پردازنده ۴ نانوثانیه باشد.

ب) زمان اجرای پردازنده‌ی خط لوله‌ای با فرض اینکه زمان اجرای هر کلاک ۱,۵ نانوثانیه و تاخیر هر ثبات‌های میانی ۰,۲۵ نانوثانیه باشد. (تعداد حباب‌ها برای مخاطره‌ی کنترلی را ۱ و برای سایر مخاطرات را ۲ در نظر بگیرید.)

ج) میزان تسریع پردازنده‌ی خط لوله‌ای به پردازنده‌ی تک سیکلی[۵] را محاسبه کنید.

```
li $t0, 58
li $t1, 1
li $t2, 0
L1: beq $t2, $t0, L2
  addi $t2, $t2, 1
  subi $t3, $t2, 3
  add $t3, $t3, $t1
  j L1
```

[۴] Speed Up

[۵] Single Cycle

L2:

سوال ٦- در این سوال تاثیرات شیوه‌های مختلف Branch-Prediction را بررسی می‌کنیم. فرض کنید که توزیع دستوراتی که داریم به صورت زیر باشد:

R-Type	BEQ	JUMP	LW	SW
40%	25%	5%	25%	5%

همچنین امکان انتخاب بین سه سیستم Branch-Prediction مختلف را داریم که هر کدام دقت متفاوتی دارد. منظور از دقت، دفعاتی است که پیش‌بینی انجام Branch به درستی صورت می‌گیرد. دقت آن‌ها به صورت جدول زیر است:

Always-Taken	Always-Not-Taken	2bit
45%	55%	85%

توجه کنید که هر کدام از این شیوه‌ها جداگانه است و در یکبار اجرای برنامه تنها از یکی از این شیوه‌ها استفاده می‌کنیم.

الف) مقدار CPI اضافه شده در اثر اشتباه در پیش‌بینی Branch را برای هر یک از استراتژی‌های Branch-Prediction به طور جداگانه مشخص کنید. فرض کنید نتیجه واقعی Branch در مرحله EX مشخص می‌شود.

ب) با فرض استفاده از سیستم 2-bit از طریق تکنیک خاصی نیمی از دستورات Branch را تبدیل به یک دستور از نوع دیگر می‌کنیم. با فرض این که دقت 2bit همچنان ثابت بماند، میزان تسریع را مشخص نمایید.

ج) با فرض استفاده از سیستم 2-bit، از طریق تکنیک خاصی نیمی از دستورات Branch را تبدیل به دو دستور نوع دیگر می‌کنیم. با فرض این که دقت 2bit همچنان ثابت بماند، میزان تسریع را مشخص نمایید.

د) بعضی از Branch ها هستند که به راحتی قابل پیش‌بینی هستند. فرض کنید ۸۰ درصد Branch ها از این نوع باشند و همواره بتوانیم به درستی رفتارشان را پیش‌بینی کنیم. با فرض استفاده از روش 2bit دقت آن در ۲۰ درصد باقی‌مانده Branch ها باید چه قدر باشد که در نهایت برای دقت کل به عدد ۸۵ درصد صورت سوال برسیم.

توجه: برای حل هیچ یک از قسمت‌های این سوال نیازی به دانستن نحوه کارکرد 2bit predictor ندارید.

سوال عملی)

هدف از این تمرین طراحی یک پردازنده‌ی Single Cycle با کنار هم قرار دادن واحدهای طراحی شده در تمرین‌های قبلی است.

این پردازنده قابلیت انجام سه نوع دستور زیر را دارد:

Type	I[31..26]	I[25..21]	I[20..16]	I[15..11]	I[10..6]	I[5..0]
R-Type	opcode	$rs	$rt	$rd	shamt	funct
I-Type	opcode	$rs	$rt	imm		
J-Type	opcode	address				

دستورات R-type :

این پردازنده قابلیت پشتیبانی از دستورات R-type زیر را دارد:

funct	Instruction	Explanation
32	add $rd, $rs, $rt	Add
34	sub $rd, $rs, $rt	Subtract
00	sll $rd, $rt, shamt	Shift Right Logical
02	srl $rd, $rt, shamt	Shift Left Logical
36	nand $rd, $rs, $rt	NAND
42	slt $rd, $rs, $rt	Set on Less Than
44	min $rd, $rs, $rt	Minimum

توجه: مقدار Opcode برای تمام دستورات R-type صفر است و این دستورات توسط funct از یکدیگر تمایز داده می‌شوند.

دستوارت I-type:

این پردازنده قابلیت پشتیبانی از دستورات I-type زیر را دارد:

Opcode	Instruction	Explanation
04	beq $rs, $rt, imm	branch on equal
05	bne $rs, $rt, imm	branch on not equal
08	addi $rt, $rs, imm	Add

09	subi $rt, $rs, imm	Subtract
10	slti $rt, $rs, imm	Set on Less Than
12	nandi $rt, $rs, imm	NAND
14	mini $rt, $rs, imm	Minimum
34	lw $rt, imm($rs)	Load Word
43	sw $rt, imm($rs)	Save Word

دستورات J-type:

Opcode	Instruction	Explanation
02	j address	jump

همان‌طور که در ابتدا نیز گفته شد، برای طراحی این پردازنده می‌توانید از واحد محاسبات، بانک ثبات و واحد پرشی که در تمارین قبلی طراحی کرده‌اید استفاده کنید اما برای تکمیل این پردازنده علاوه برای این واحدها نیاز به طراحی واحد کنترلی و واحد حافظه نیز دارید.

واحد حافظه:

واحد حافظه شامل ۶۴ ثبات ۳۲ بیتی است. دستورات از بالای حافظه و داده‌ها در پایین این حافظه پیش اجرا برنامه نوشته می‌شوند. در هر کلاک دستوری که ثبات PC به آن اشاره می‌کند از این حافظه خوانده می‌شود و این دستور توسط پردازنده اجرا می‌شود. می توانید برای طراحی این واحد از زبان Verilog استفاده کنید.

واحد کنترلی:

این واحد وظیفه‌ی تولید سیگنال‌های مورد نیاز برای اجرای دستورات را دارد. سیگنال‌های تولید شده توسط این واحد باید مطابق سیگنال‌ها تدریس شده در لکچر ۶ام باشند.

در گزارشی شیوه‌ی پیاده‌سازی قسمت‌های مختلف پردازنده‌ی خود را توضیح دهید علاوه بر توضیحات بخش‌های مختلف پردازنده با تهیه‌ی waveform از صحت اجرای تمامی دستورات اطمینان حاصل کنید.

درنهایت برنامه‌ای در حافظه‌ی خود بنویسید که با استفاده از حلقه جمله‌ی ۲۰ ام دنباله‌ی فیبوناچی را محاسبه کند، تصاویر waveform مربوط به اجرای این برنامه را نیز در گزارش خود بیاورید.