به نام خدا

دانشکده مهندسی کامپیوتر

درس معماری کامپیوتر
نیم‌سال دوم ۰۴-۰۳
استاد: دکتر اسدی

تمرین سری پنجم

- پرسش‌های خود را در سامانه CW و تالار مربوط به تمرین مطرح نمایید.
- پاسخ سوالات را تایپ نمایید.
- پاسخ تمرین را به صورت یک فایل زیپ با فرمت HW1_401234567.zip آپلود کنید. فایل زیپ باید به صورتی باشد که پس از باز کردن آن بدون هیچ پوشه‌ای فایل‌های زیر با ساختار زیر قرار گرفته باشند:

```
.
|-- HW1T_401234567.pdf
|-- practical
    |-- HW1P_401234567_401234568.pdf
    |-- schematic.circ
    |-- verilog
        |-- circuit
        |   |-- main.v
        |   |-- ...
        |-- gates
        |   |-- ...
        |-- toplevel
            |-- ...
```

- در صورت عدم تطابق فایل آپلود شده با فرمت بالا، تمرین شما تصحیح **نخواهد شد**.
- پاسخ سوالات تئوری و گزارش تمرین‌های عملی باید به فرمت pdf باشد.
- هر دانشجو می‌تواند حداکثر سه تمرین را با دو روز تأخیر بدون کاهش نمره ارسال نماید.
- تمرینات عملی به صورت گروه‌های دو نفر تحویل داده شود.
- هر دو عضو گروه موظف هستند تمرینات خود را بارگذاری کنند.
- عواقب عدم تطابق بین پاسخ دو عضو گروه برعهده خودشان است.
- تحویل تمرین به صورت انگلیسی مجاز نیست. در صورت تحویل تمرین به صورت انگلیسی (حتی بخشی از تمرین) نمره تمرین موردنظر صفر در نظر گرفته می‌شود.
- در صورت مشاهده تقلب برای بار اول نمره هر دو طرف صفر می‌شود. در صورت تکرار نمره کل تمرینات صفر خواهد شد.
- استفاده از ابزارهایی مانند ChatGPT به منظور ابزار کمک آموزشی مجاز است به شرط آن که به خروجی آن اکتفا نشود.
- توجه شود که پروژه نهایی درس در گروه‌های چهار نفر تحویل گرفته می‌شود.
- سوالات با عنوان **اختیاری** نمره‌ای ندارند اما جواب دادن به آن‌ها کمک به‌سزایی در یادگیری درس می‌کند.

تمارین تئوری

۱. میخواهیم مجموعه دستورات موجود در جدول زیر را به پردازنده میپس اضافه کنیم. فرض کنید نوشتن در بانک ثبات و حافظه در لبه بالارونده، خواندن از حافظه در لبه پایینرونده و خواندن از بانک ثبات به صورت آسنکرون انجام شود. مشخص کنید چه تغییراتی باید در ساختار پردازنده اعمال کنیم و همچنین سیگنال‌های کنترلی موردنیاز برای پیاده‌سازی هر دستور را نیز مشخص کنید. (همراه با ALUOP)

جدول ۱:

Instruction	Operation
swap $rs,$rt,imm	$rs \leftrightarrow Mem[rt+imm]$
addnz $rs,$rt,imm	if $rt != 0 then $rs = rs + imm$
loadpc $rd	rd= pc+4
brsumz rs, rt, offset	$\text{if } (rs + rt == 0) \rightarrow PC = PC + 4 + offset$

۲. جدول زیر تاخیر بلوک‌های منطقی مختلف در یک مسیر داده پردازنده MIPS را نشان می‌دهد (تاخیر واحد کنترلی را صفر در نظر بگیرید). با توجه به این جدول به سوالات زیر پاسخ دهید:

ALU Ctrl	Shift-left-2	Sign-extend	D-Mem	Regs	ALU	Mux	Add	I-Mem
۵۵ps	۲۰ps	۹۰ps	۱۰۰۰ps	۲۲۰ps	۱۸۰ps	۱۰۰ps	۱۵۰ps	۵۰۰ps

(آ) زمان چرخه ساعت در این پردازنده چقدر است؟ (فرض کنید زمان چرخه ساعت برابر با حداقل مقدار ممکن خود است.)

(ب) برای جلوگیری از طولانی شدن زمان مسیر بحرانی در مسیر داده، واحد کنترلی چه مدت زمانی را می‌تواند برای تولید سیگنال MemWrite صرف کند؟

(ج) کدام سیگنال کنترلی بیشترین زمان را برای تولید شدن دارد؟ واحد کنترلی چه مدت زمانی را می‌تواند برای تولید این سیگنال صرف کند بدون آن که در مسیر بحرانی قرار گیرد؟

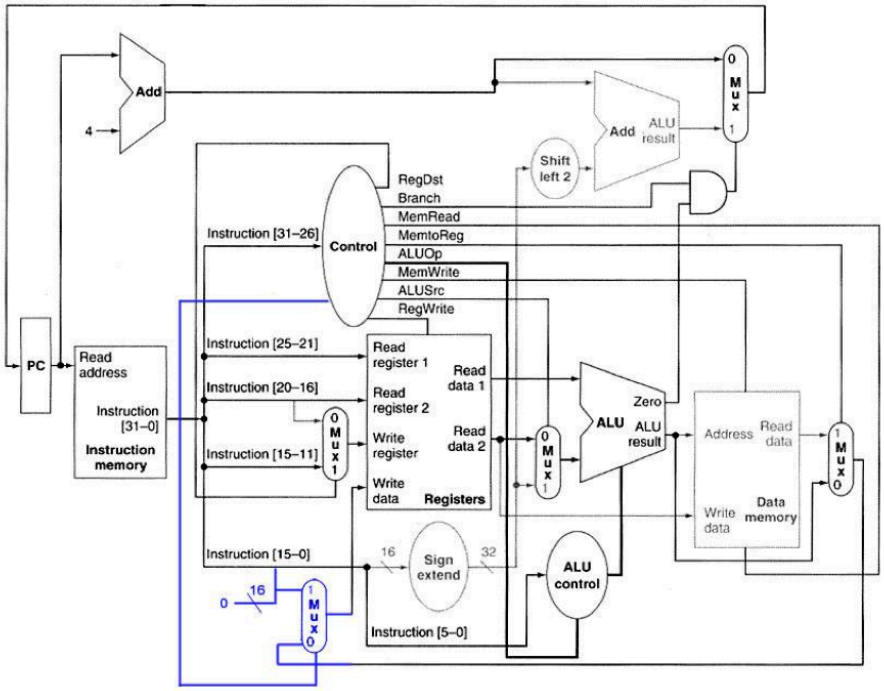
(د) تولید سریع‌تر کدام سیگنال کنترلی در مسیر داده ضروری است؟ واحد کنترلی در چه مدت زمانی باید این سیگنال را تولید کند؟

۳. فرض کنید می‌خواهیم دستور forr را به پردازنده میپسی که در اسلایدهای درس دیده‌اید اضافه کنیم. طرز کار این دستور به این صورت است که محتوی $rt را از $rs کم کرده و نتیجه را در $rs می‌نویسد. اگر این حاصل صفر نباشد به صورت نسبی به مقدار ثابتی که در دستور است پرش می‌کند.

(آ) این دستور از چه نوعی است؟

(ب) توضیح دهید چه تغییراتی در مدار میپس باید اضافه شود و سیگنال‌های کنترلی باید به چه صورت تنظیم شوند تا این دستور اجرا شود.

۴. با توجه به تغییرات به وجود آمده در مسیرداده‌ی پردازنده به سوالات زیر پاسخ دهید:



(آ) توضیح دهید که این تغییرات برای اضافه شدن چه دستوری انجام شده‌اند.

(ب) جدول زیر را با توجه دستور جدید و سیگنال کنترلی اضافه شده پر نمایید.

Instr	RegDst	ALUSrc	Mem toReg	Reg Write	Mem Read	Mem Write	Branch	ALUOp ۱	ALUOp ۲	سیگنال جدید
R-type	۱	۰	۰	۱	۰	۰	۰	۱	۰	
lw	۰	۱	۱	۱	۱	۰	۰	۰	۰	
sw	x	۱	x	۰	۰	۱	۰	۰	۰	
beq	x	۰	x	۰	۰	۰	۱	۰	۱	
دستور جدید										

۵. برای هر یک از ۸ سیگنال کنترلی (ALUOp را در نظر نگیرید) موجود در شکل سوال ۱، مشخص کنید با بروز اختلال‌های stuck-at-0 و stuck-at-1 پردازنده در اجرای چه دستوراتی به مشکل می‌خورد. (لطفا از نوشتن پاسخ طولانی اجتناب کنید. انتظار می‌رود برای هر کدام از سیگنال‌های کنترلی نهایتا ۲ خط توضیح در مجموع بنویسید.) (**اختیاری** - برای بیت پرارزش و کم‌ارزش ALUOp نیز همین کار را انجام دهید.)

۶. **اختیاری** - به سوالات زیر پاسخ دهید.

(آ) می‌خواهیم دستور R-type زیر را به معماری پردازنده میپس خود اضافه کنیم. با توجه به کارکرد این دستور، سیگنال‌های کنترلی را برای آن مشخص کنید.

lwd $rd, $rt($rs) // $rd = Mem[$rs + $rt]

(ب) اجرای برنامه زیر، با توجه به تاخیرهای داده شده چه مدت زمانی به طول می‌انجامد؟ (پردازنده میپس ما تک چرخه است)

```
// Memory: 5 ns, ALU: 4 ns, Register file: 3 ns
addi $sp, $sp, -4
sub $v0, $a0, $a1
or $s0, $s1, $s2
lw $t0, 4($sp)
sw $v0, 8($sp)
```

(ج) اگر ALU را ۲۵ درصد سریعتر کنیم، بهبود زمان اجرای برنامه بخش قبل چقدر خواهد بود؟

تمارین عملی

۱. در این تمرین بنا داریم طراحی پردازنده single cycle تمرین قبل را کامل کنیم. هدف نهایی در این تمرین، اجرای قطعه کد زیر است.

```
int main() {
    int xs[3] = {0, 1, 0};
    for (int i = 0; i < 10; i++)
        xs[(i + 2)%3] = xs[(i + 1)%3] + xs[i%3];
}
```

این قطعه کد در نهایت به کد اسمبلی زیر تبدیل می‌شود.

```
   addi    sp,sp,-32
   sw      s8,28(sp)
   add     s8,sp,zero
   sw      zero,12(s8)
   addi    v0,zero,1
   sw      v0,16(s8)
   sw      zero,20(s8)
   sw      zero,8(s8)
   j       $L2
   nop
   nop
   $L3:
   lw      v0,8(s8)
   nop
   addi    v1,v0,1
   addi    v0,zero,3
   div     zero,v1,v0
   nop
   mfhi    v0
   sll     v0,v0,0x2
   addi    v1,s8,8
   add     v0,v1,v0
   lw      v1,4(v0)
   lw      a0,8(s8)
   addi    v0,zero,3
   div     zero,a0,v0
   nop
   mfhi    v0
   sll     v0,v0,0x2
   addi    a0,s8,8
   add     v0,a0,v0
   lw      v0,4(v0)
   lw      a0,8(s8)
   nop
   addi    a1,a0,2
   addi    a0,zero,3
   div     zero,a1,a0
   nop
   mfhi    a0
   add     v1,v1,v0
   sll     v0,a0,0x2
   addi    a0,s8,8
   add     v0,a0,v0
   sw      v1,4(v0)
```

```
    lw      v0,8(s8)
    nop
    addi    v0,v0,1
    sw      v0,8(s8)
    $L2:
    lw      v0,8(s8)
    nop
    slti    v0,v0,10
    bnez    v0,$L3
    nop
    add     sp,s8,zero
    lw      s8,28(sp)
    addi    sp,sp,32
    jr      ra
    nop
    nop
    jr      ra
    nop
```

لیست دستوراتی که لازم است علاوه بر دستورات تمرین قبل اضافه کنید، به صورت زیر است:

```
sw $1, n($2)
lw $1, n($2)
j label
nop
div $1, $2, $3
mfhi $1
sll $1, $2, immediate
slti $1, $2, immediate
bnez $1, label
jr $1
```

همانطور که مشاهده می‌کنید در این پیاده‌سازی لازم است از دستور div پشتیبانی کنید. اما در پیاده‌سازی ALU این دستور بیش از یک کلاک طول می‌کشد. بنابراین به صورت پیش‌فرض تمام دستورات در یک کلاک انجام می‌شوند و دستور div در چند کلاک انجام می‌شود. شما باید به نحوی این مسئله را حل کنید، به طوری که تا قبل از تمام شدن کار ALU برای دستور div دستور بعدی شروع به کار نکند. همچنین شما باید یک سیگنال InstDone به پیاده‌سازی خود اضافه کنید تا فایل داوری نیز از تمام شدن دستور کنونی مطلع بشود. تمام شدن به این مفهوم است که وقتی InstDone ۱ باشد و سپس کلاک از ۰ به ۱ تغییر کند، مقدار ثبات ها با توجه به انجام این دستور بروزرسانی شده است.

سیگنال‌های ورودی و خروجی به صورت زیر خواهد بود:

Inputs:

- clk
- rst
- Jin (32 bit)
- Jen

Outputs:

- R1...R32 (32 bit)
- Jout (32 bit)
- InstDone