In the name of god

Computer Architecture

Instructor: Dr. Mohammad Reza Moosavi

Teacher Assistants:

Siavash Jan Ahmadian

Amir Mohammad Tavakkoli

Reyhane Yazdani

Zahra Jamali

Ali Asnaashari

Zahra Mohammadpour

Project

- Any fraud shall be recognized and the person(s) will be punished and won't get any of this project's grade.

- Wish you the best of luck.

برای پروژه درس معماری، هدف پیاده سازی سیستم mips است که برای این امر باید از یک زبان همه منظوره نظیر Java یا Python و نه یک زبان شبیه سازی سخت افزار یا همان HDL کمک گرفت. برای سادگی کار، صرف پیاده سازی به صورت single cycle کافیست اما باید امکان اجرای دستوراتی که در عکس پایین بیان شده اند وجود داشته باشد.

Instruction	Example	Meaning	Comments
add	add $s1,$s2,$s3	$s1 = $s2 + $s3	Three register operands
subtract	sub $s1,$s2,$s3	$s1 = $s2 – $s3	Three register operands
add immediate	addi $s1,$s2,20	$s1 = $s2 + 20	Used to add constants
load word	lw $s1,20($s2)	$s1 = Memory[$s2 + 20]	Word from memory to register
store word	sw $s1,20($s2)	Memory[$s2 + 20] = $s1	Word from register to memory
load half	lh $s1,20($s2)	$s1 = Memory[$s2 + 20]	Halfword memory to register
load half unsigned	lhu $s1,20($s2)	$s1 = Memory[$s2 + 20]	Halfword memory to register
store half	sh $s1,20($s2)	Memory[$s2 + 20] = $s1	Halfword register to memory
load byte	lb $s1,20($s2)	$s1 = Memory[$s2 + 20]	Byte from memory to register
load byte unsigned	lbu $s1,20($s2)	$s1 = Memory[$s2 + 20]	Byte from memory to register
store byte	sb $s1,20($s2)	Memory[$s2 + 20] = $s1	Byte from register to memory
and	and $s1,$s2,$s3	$s1 = $s2 & $s3	Three reg. operands; bit-by-bit AND
or	or $s1,$s2,$s3	$s1 = $s2 | $s3	Three reg. operands; bit-by-bit OR
nor	nor $s1,$s2,$s3	$s1 = ~ ($s2 | $s3)	Three reg. operands; bit-by-bit NOR
and immediate	andi $s1,$s2,20	$s1 = $s2 & 20	Bit-by-bit AND reg with constant
or immediate	ori $s1,$s2,20	$s1 = $s2 | 20	Bit-by-bit OR reg with constant
shift left logical	sll $s1,$s2,10	$s1 = $s2 << 10	Shift left by constant
shift right logical	srl $s1,$s2,10	$s1 = $s2 >> 10	Shift right by constant
branch on equal	beq $s1,$s2,25	if ($s1 == $s2) go to PC + 4 + 100	Equal test; PC-relative branch
branch on not equal	bne $s1,$s2,25	if ($s1!= $s2) go to PC + 4 + 100	Not equal test; PC-relative
set on less than	slt $s1,$s2,$s3	if ($s2 < $s3) $s1 = 1; else $s1 = 0	Compare less than; for beq, bne
set on less than unsigned	sltu $s1,$s2,$s3	if ($s2 < $s3) $s1 = 1; else $s1 = 0	Compare less than unsigned
set less than immediate	slti $s1,$s2,20	if ($s2 < 20) $s1 = 1; else $s1 = 0	Compare less than constant
jump	j 2500	go to 10000	Jump to target address
jump register	jr $ra	go to $ra	For switch, procedure return
jump and link	jal 2500	$ra = PC + 4; go to 10000	For procedure call

هیچ اجباری برای استفاده از از کتابخانه خاصی برای پیاده سازی این سیستم نیست اما دو کتابخانه برای دو زبان Java و Python پیشنهاد میشوند که کتابخانه هایی با قابلیت gate level به عنوان بستر شبیه سازی سخت افزار هستند:

1- https://github.com/turingkomplete/GraphBasedMinimalistCircuitSimulator برای Java
که ویدیویی برای توضیح این کتابخانه در لینک زیر موجود است:
https://drive.google.com/drive/folders/1SoTbt4_f9CCDB9ZdNrecaRlFk5lUa2XM?usp=sharing

2- https://github.com/Atelier-Developers/Logical-Circuit-Simulator برای زبان Python