In the name of Allah

Overview Presentation

Zohre Soorani
Mahdi Haghverdi
Hussein Husseini
Hosna Rajaei

Isfahan University

November 14, 2023

Content

Computer Abstraction and Technology

Computer Abstraction and Technology

Instructions: Language of the computer

Operations of the Computer Hardware

Figure: Arithmatic Instructions in MIPS

Category	Instruction	Example	Meaning	Comments
Arithmetic	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

Operations of the Computer Hardware (Cont'd)

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 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
load upper immed	`lui $s1, 20`	`$s1 = 20 * 2`^{16}	Loads constant in upper 16 bits

Table: Data Transfer Instructions in MIPS

Operations of the Computer Hardware (Cont'd)

Figure: Logical Instructions in MIPS

Logical	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

Operations of the Computer Hardware (Cont'd)

Figure: Conditional Branch Instructions in MIPS

Conditional branch	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
	set less than immediate unsigned	`sltiu $s1,$s2,20`	if ($s2 < 20) $s1 = 1; else $s1 = 0	Compare less than constant unsigned

Operations of the Computer Hardware (Cont'd)

Figure: Unconditional Jump Instructions in MIPS

Unconditional jump	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