Single Cycle MIPS Processor

Instructions Supported

Memory References	lw, sw
ALU	add, sub, and, or, slt, addi
Control Transfer	beq, j, jr
Subroutine Support Instructions	jal

Register File

This processor contains 32 registers each of 32 bits (word length).

Register Naming Conventions

$zero ***:*** R0 | stores constant value 0

$t0 - $t7: R8 – R15 | temporary registers

$s0 - $s7: R16 – R23 | saved registers

$t8 - $t9: R24 – R25 | more temporary registers

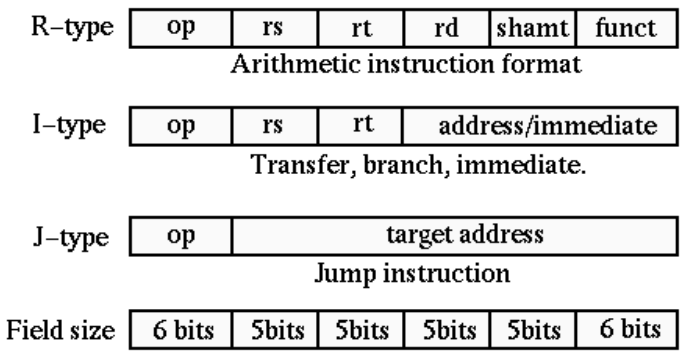
$ra ***:*** R31 | stores return address used in *"jal"*

Note: other registers are beyond the scope of this assignment

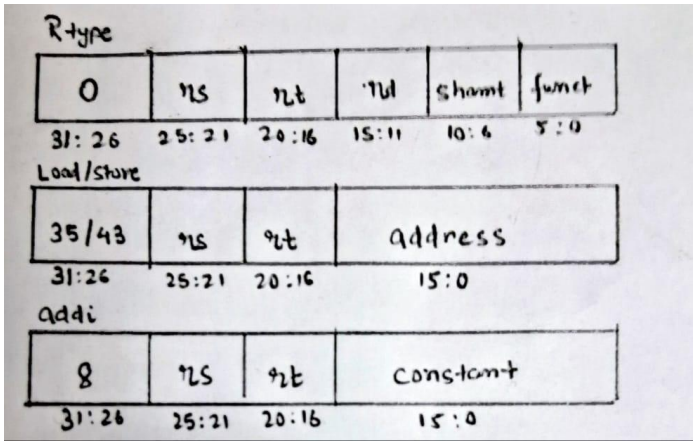
Types of Instructions in MIPS

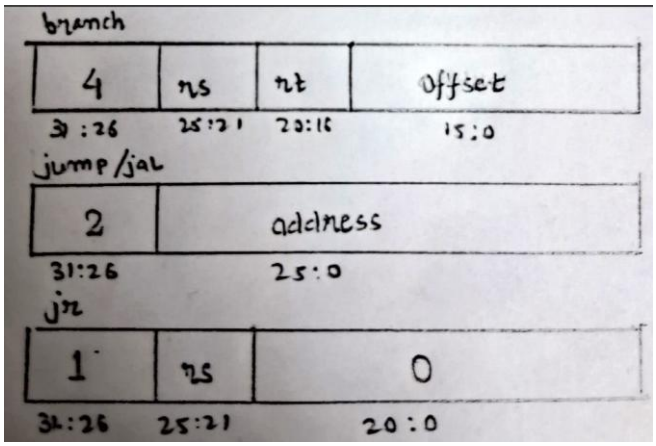
There are three types of instructions in MIPS:

R-type, I-type and J-type



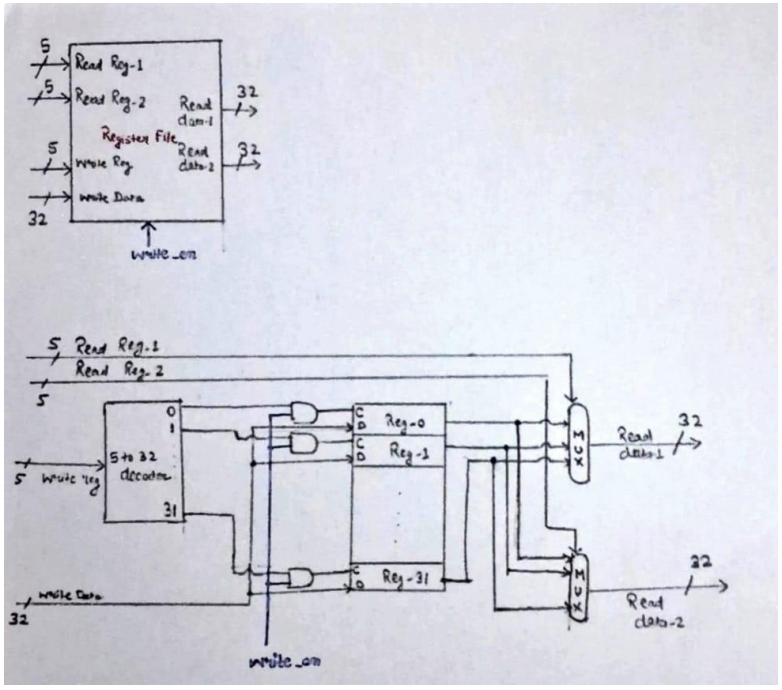
Instruction Type	Instruction	Syntax Example	Description
R-Type	add	add $t0, $t1, $t2	$t0 = $t1 + $t2
	sub	sub $t0, $t1, $t2	$t0 = $t1 - $t2
	and	and $t0, $t1, $t2	$t0 = $t1 & $t2
	or	or $t0, $t1, $t2	$t0 = $t1 | $t2
	slt	slt $t0, $t1, $t2	$t0 = ($t1 < $t2) ? 1 : 0
	jr	jr $ra	Jump to address in $ra (used for returns)
I-Type	lw	lw $t0, 32($s1)	Load word from memory
	sw	sw $t0, 16($s2)	Store word to memory
	beq	beq $s0, $s1, label	Branch if equal
	addi	addi $t0, $t1, 10	$t0 = $t1 + 10
J-Type	j	j target_label	Jump to label
	jal	jal subroutine_label	Jump and link (store return address in $ra)



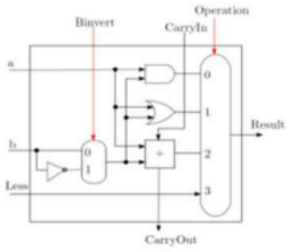


Different Components in Processors

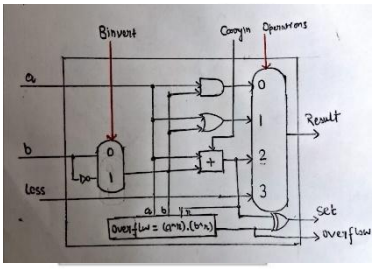
1. Register File



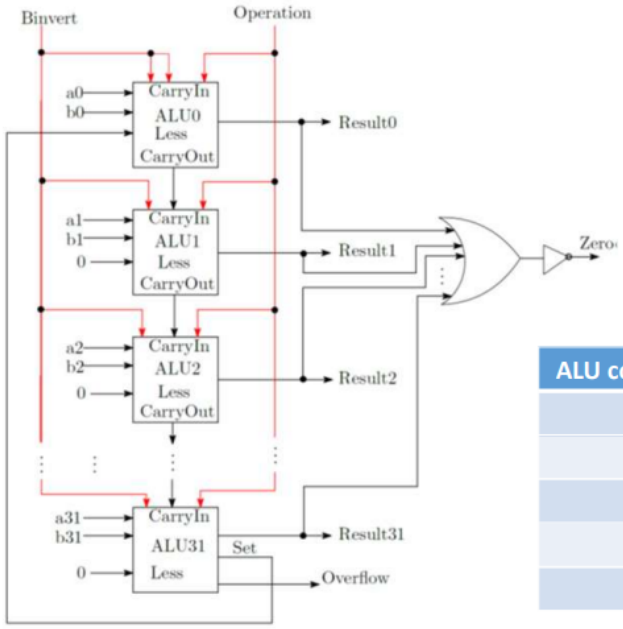
2. ALU

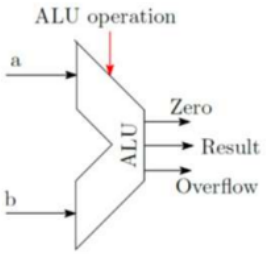


ALU 0 – ALU 30



ALU 31



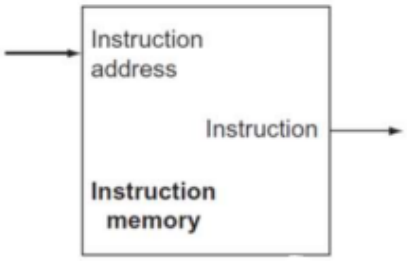


ALU control lines	Function
000	and
001	or
010	add
110	subtract
111	Set on less than

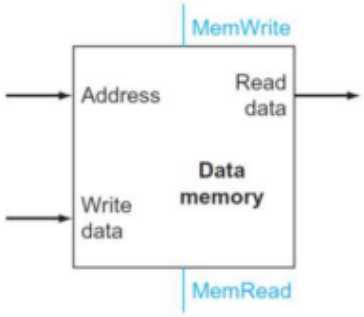
ALU Control

Opcode	ALUOp	Operation	funct	ALU Function	ALU Control
lw	00	load word	XXXXXXX	add	010
sw	00	store word	XXXXXXX	add	010
beq	01	branch equal	XXXXXXX	subtract	110
addi	00	add immediate	XXXXXXX	add	010
R-type	10	add	100000	add	010
		subtract	100010	subtract	110
		AND	100100	AND	000
		OR	100101	OR	001
		set-on-less-than	101010	set-on-less-than	111

3. Memory



a. Instruction memory



a. Data memory unit

Control Signals

Instruction	Opcode	JumpR	RegDataJ	RegDst	RegDstJ	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
R-type	000000	0	0	1	0	0	0	0	0	10	0	0	1
lw	100011	0	0	0	0	0	0	1	1	00	0	1	1
sw	101011	0	0	0	0	0	0	0	0	00	1	1	0
addi	001000	0	0	0	0	0	0	0	0	00	0	1	1
j	000010	0	0	0	0	1	0	0	0	00	0	0	0
jal	000011	0	1	0	1	1	0	0	0	00	0	0	1
jr	000001	1	0	0	0	0	0	0	0	00	0	0	0
beq	000100	0	0	0	0	0	1	0	0	01	0	0	0
halt	111111	0	0	0	0	0	0	0	0	00	0	0	0

Note: Some of the values were don't cares but for circuit simplicity we kept it to be 0.

Data Path