Nolan Griffith, Ant Nosaryev
ECE-251-1

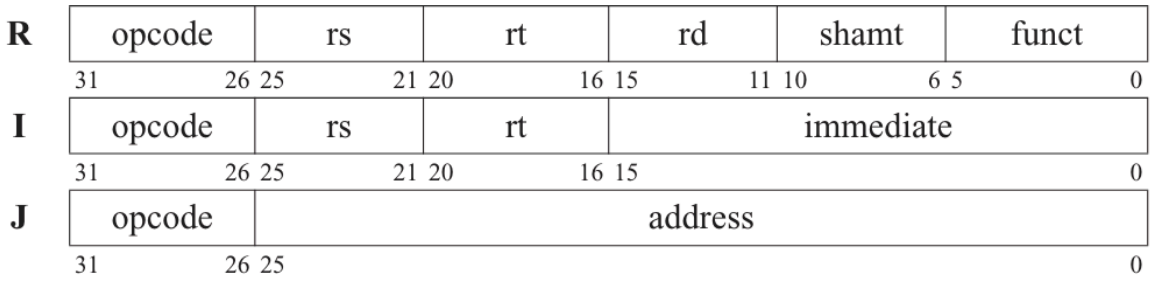
¿HablaGhjeePeeTee Reference Data

CORE INSTRUCTION SET

Name	Syntax	Type	Operation	Opcode/Funct (hex)
Add	`add`	R	R[rd] = R[rs] + R[rt]	$0/20_{(hex)}$
Add Immediate	`addi`	I	R[rt] = R[rs] + SignExtImm	$24_{(hex)}$
And	`and`	R	R[rd] = R[rs] & R[rt]	$0/24_{(hex)}$
Branch on Equal	`beq`	I	if(R[rs]==R[rt]) PC = PC+4+BranchAddr	$4_{(hex)}$
Jump	`j`	I	PC =JumpAddr	$2_{(hex)}$
Jump and Link	`jal`	I	R[15]= PC+4; PC =JumpAddr	$3_{(hex)}$
Jump Register	`jr`	R	PC = R[rs]	$0/8_{(hex)}$
Move From Hi	`mfhi`	R	R[rd] = Hi	$0/10_{(hex)}$
Move From Lo	`mflo`	R	R[rd] = Lo	$0/12_{(hex)}$
Multiply	`mult`	R	{R[Hi], R[Lo]} = R[rs] * R[rt]	$0/18_{(hex)}$
Load Word	`lw`	I	R[rt] = M[R[rs]+SignExtImm]	$23_{(hex)}$
Or	`or`	R	R[rd] = R[rs] | R[rt]	$0/25_{(hex)}$
Store Word	`sw`	I	M[R[rs]+SignExtImm] = R[rt]	$2b_{(hex)}$
Subtract	`sub`	R	R[rd] = R[rs] - R[rt]	$0/22_{(hex)}$

Note: Nor not implemented due to saturation of all possible 3-bit wide `alucontrol` values.

BASIC INSTRUCTION FORMATS

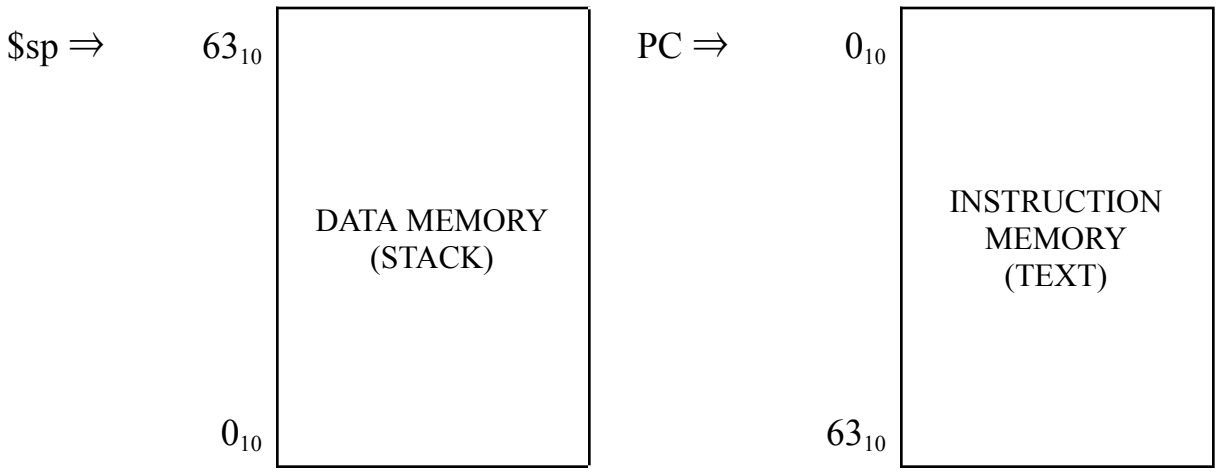


REGISTER NAME, NUMBER, USE, CALL CONVENTION

NAME	NUMBER	USE	PRESERVED ACROSS A CALL?
$zero	0	The Constant Value 0	N.A.
$at	1	Assembler Temporary	No
$v0-$v1	2-3	Values for Function Results and Expression Evaluation	No
$a0-$a3	4-7	Arguments	No
$t0-$t7	8-15	Temporaries	No
$s0-$s7	16-23	Saved Temporaries	Yes
$t8-$t9	24-25	Temporaries	No
$k0-$k1	26-27	Reserved for OS Kernel	No
$gp	28	Global Pointer	Yes
$sp	29	Stack Pointer	Yes
$fp	30	Frame Pointer	Yes
$ra	31	Return Address	No

MEMORY ALLOCATION

Note: The diagram below contains word addresses.



Basic Instruction Formats and *Register Name, Number Use, Call Convention* Tables taken from Patterson and Hennessy, Computer Organization and Design, 4th ed.