# Verilog

RISC → MIPS 32

## MIPS 32

32 registers of 32 bit => R0 to R31

R0 => 0

32 bit register program counter.

No flag

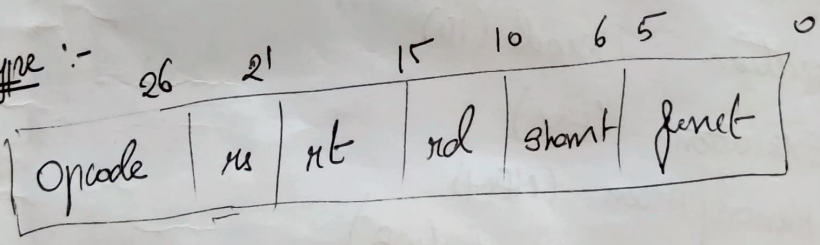
Load & Store instructions ; 32 bit address only.

Load & Store ;

LW , SW

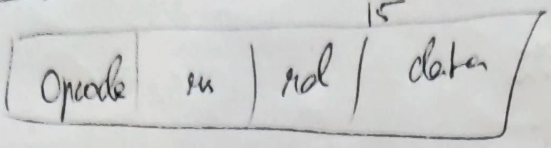
ALU ;

R Type :-




| | |
|---|---|
| ADD | 000000 |
| SUB | 000001 |
| AND | 000010 |
| OR | 000011 |
| SLT | 000100 |
| MUL | 000101 |
| HLT | 111111 |

I-Type :- 16 bits data




| | |
|---|---|
| LW | 001000 |
| SW | 001001 |
| ADDI | 001010 |
| SUBI | 001011 |
| SLTI | 001100 |
| BNEQZ | 001101 |
| BEQZ. | 001110 |

Prefetch all registers and usage only be determined when decoded.

Instruction cycle

1) Instruction Fetch (IF)
2) Instruction Decode (ID)
3) Execution (EX)
4) Memory Access (MEM)
5) Register Write back (WB)

1) IF :-

IR ← Mem[PC]

instruction register.

NPC ← PC+1;

ID :-

A ← Reg[rs];

B ← Reg[rt];

Imm ← (IR15)16 ## IR15...0

Imm1 ← (IR25)6 ## IR25...0 (not used here).

IR
A
B
Imm

Ex:

Memory reference

ALUout ← A + Imm;

Register - Register

ALUout ← A func B

Register - immediate

ALUout ← A func imm

Branch.

ALUout ← NPC + Imm;

Cond ← (A op 0);

MEM

load

PC ← NPC;

LMD ← Mem[ALUout];

store;

PC ← NPC;

Mem[ALout] ← B;

Branch

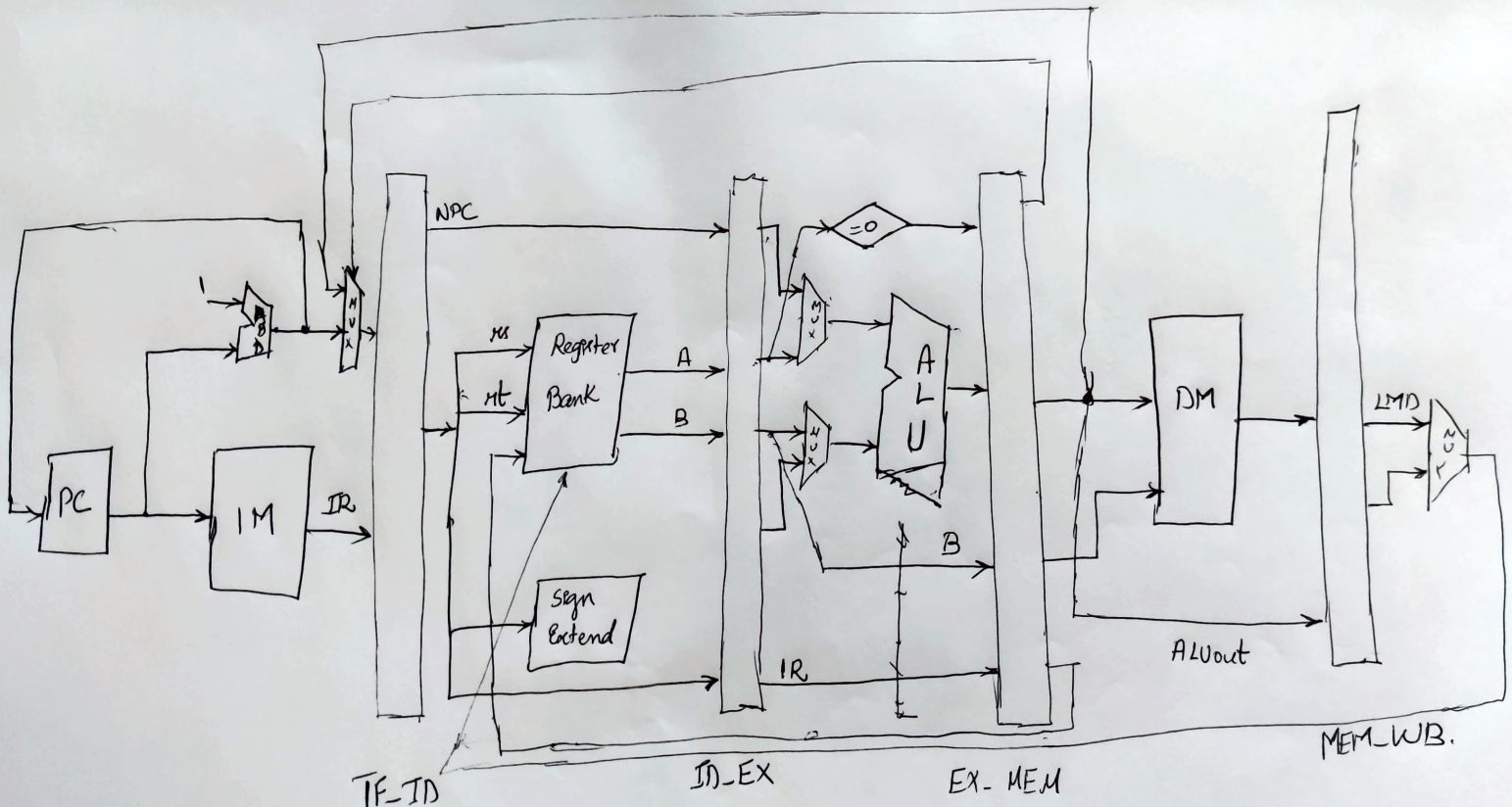
if(cond) PC ← ALUout;

else PC ← NPC;

Other

PC ← NPC;

# MIPS 32


NPC
=0
PC
IM
IR
Register Bank
rs
rt
A
B
sign Extend
ALU
DM
LMD
ALUout
IR
B
IF_ID
ID_EX
EX_MEM
MEM_WB.