BASIC MIPS IMPLEMENTATION

1. **Instruction fetch cycle (IF):**

IR = Mem[PC];

NPC = PC + 4; Operation:

Send out the PC and fetch the instruction from memory into the instruction register (IR). Increment the PC by 4 to address the next sequential instruction.

IR - holds instruction that will be needed on subsequent clock cycles Register NPC - holds next sequential PC.

2. **Instruction decode/register fetch cycle (ID):**

A = Regs[rs]; B = Regs[rt];

Imm = sign-extended immediate field of IR; Operation:

Decode instruction and access register file to read the registers (rs and rt -register specifiers).

Outputs of general purpose registers are read into 2 temporary registers (A and B) for use in later clock cycles.

Lower 16 bits of IR are sign extended and stored into the temporary register Imm, for use in the next cycle.

3. **Execution/effective address cycle (EX):**

- ALU operates on the operands prepared in the prior cycle, performing one of four functions depending on the MIPS instruction type.

i. Memory reference: ALUOutput = A + Imm;

ii. Register-Register ALU instruction:

ALUOutput = A func B; Operation:

ALU performs the operation specified by the function code on the value in register A and in register B.

a) Result is placed in temporary register ALU Output

b) Register-Immediate ALU instruction:

ALU Output = A op Imm; Operation:

ALU performs operation specified by the opcode on the value in register A and register Imm.

Result is placed in temporary register ALU Output.

iv)Branch:

ALUOutput = NPC + (Imm << 2); Cond = (A == 0)

Operation:

- ALU adds NPC to sign-extended immediate value in Imm, which is shifted left by 2 bits to create a word offset, to compute address of branch target.
- Register A, which has been read in the prior cycle, is checked to determine whether branch is taken.

a) Considering only one form of branch (BEQZ), the comparison is against 0.

4. Memory access/branch completion cycle (MEM):

* PC is updated for all instructions: PC = NPC; i. Memory reference: LMD = Mem[ALUOutput] or

Mem[ALUOutput] = B;

Operation:

a) Access memory if needed.

b) Instruction is load-data returns from memory and is placed in LMD (load memory data)

c) Instruction is store-data from the B register is written into memory

ii.Branch:

if (cond) PC = ALU Output

Operation: If the instruction branches, PC is replaced with the branch destination address in register ALU Output.

5.Write-back cycle (WB):

* Register-Register ALU instruction: Regs[rd] = ALUOutput;
* Register-Immediate ALU instruction: Regs[rt] = ALUOutput;
* Load instruction:

  Regs[rt] = LMD;

  Operation: Write the result into register file, depending on the effective opcode.