

Project Name: Build a Virtual CPU Emulator

Week 04: Instruction Execution

Objective: The objective for this week is to develop the instruction fetch-decode-execute cycle, a fundamental component of the CPU's operation. The focus will be on implementing an efficient mechanism for fetching instructions, decoding them to understand their purpose, and executing them using the Arithmetic Logic Unit (ALU) and registers. This will establish the core functionality required for program execution and will be tested using simple programs.

1. Implement the instruction fetching mechanism.

The instruction-fetching mechanism is a critical component of a processor's control unit, responsible for fetching instructions from memory for execution. Here's an implementation of an instruction-fetching mechanism at a high level, often used in computer architecture.

Steps of Instruction Fetching

- **Program Counter (PC):** Holds the address of the next instruction.
- **Instruction Memory Access:** Access the instruction stored at the address in the PC.
- **Instruction Register (IR):** The read instruction is stored in another special register in the CPU, called the instruction register.

Without the instruction fetching process, the CPU cannot understand the instructions of a program and cannot act on them. This is the first and very important step in program execution.

2. Decode instructions and execute them using the ALU and registers.

Decoding and executing instructions using the Arithmetic Logic Unit (ALU) and registers is a key part of a computer's Instruction Execution Cycle. Here's a breakdown of the process:

A. Fetch Stage

The CPU fetches the next instruction from memory (pointed to by the **Program Counter (PC)**) and stores it in the **Instruction Register (IR)**.

B. Decode Stage

The instruction is decoded to determine:

- The operation to be performed (e.g., addition, subtraction, logical operations).
- The source and destination registers or memory addresses.
- Whether immediate values (constants) are involved.

C. Execute Stage (Using ALU and Registers)

The execution process involves the **ALU**, **Registers**, and sometimes **Memory**:

1. Identify the Operand Sources:

- Operands are fetched from **registers** or **memory**, based on the instruction.
- Some instructions might include immediate values that are directly available in the instruction itself.

2. ALU Performs the Operation:

- The ALU performs the operation specified in the instruction (e.g., addition, subtraction, AND, OR, etc.).
- Control signals from the **Control Unit** dictate the type of operation.

3. Store the Result:

- The result of the ALU operation is stored in a destination register or written back to memory.

Example: ADD Instruction

Instruction: ADD R1, R2, R3

1. **Fetch:** The CPU fetches the instruction ADD R1, R2, R3.

2. **Decode:**

- Operation: ADD
- Operands: R2 and R3
- Destination: R1

3. **Execute:**

- Fetch values from R2 and R3 registers.
- Add value in the ALU: $\text{ValueR2} + \text{ValueR3}$.
- Store the result in R1.

4. Test with simple programs.

Testing the instruction-fetch-decode-execute cycle is essential to ensure the emulator operates correctly. Testing involves designing simple programs that use basic instructions and verifying their execution.