Computer Organization - Basic Computer Instructions

Computer organization describes how different parts of a computer system are arranged and connected to work together.

- Focuses on how hardware components interact to perform tasks.
- Includes the basic set of instructions that the computer can execute.
- Helps understand how programs are processed at a low level.
- Forms the foundation for efficient system operation.

Basic Computer Instructions

Basic computer instructions are commands that tell a computer how to perform specific tasks. These instructions are typically divided into three categories:

1. Data Transfer Instructions
2. Arithmetic and Logic Instructions
3. Control Instructions

1. Data Transfer Instructions

Data transfer instructions are used to move data from one location to another within a computer system. This includes transferring information between memory, CPU registers, and other storage locations.

Common Data Transfer Instructions are:

- **Load:** Copies data from memory into a CPU register for processing.
- **Store:** Transfers data from a CPU register back to memory.
- **Move:** Transfers data from one register to another within the CPU.

2. Arithmetic and Logic Instructions

These instructions are used to perform mathematical and logical operations. They enable computers to handle calculations and make decisions based on certain conditions.

Common Arithmetic Instructions are:

- **Add**: Adds two numbers.
- **Subtract**: Subtracts one number from another.
- **Multiply**: Multiplies two numbers.
- **Divide**: Divides one number by another.

Common Logic Instructions are:

- **AND**: Compares two bits and returns 1 if both are 1; otherwise, returns 0.
- **OR**: Compares two bits and returns 1 if at least one is 1.
- **NOT**: Inverts a bit (1 becomes 0, and 0 becomes 1).
- **XOR (Exclusive OR)**: Returns 1 if the bits are different, 0 if they are the same.

3. Control Instructions

Control instructions determine the flow of execution in a program. They guide the computer in deciding which instruction should be executed next, enabling decision-making, looping, and function execution.

Common Control Instructions are:

- **Jump (JMP):** Transfers program execution directly to a specific instruction elsewhere in the code.
- **Conditional Branch:** Transfers execution to another instruction only when a specified condition is satisfied, such as Branch if Zero (BZ) or Branch if Not Zero (BNZ).
- **Call:** Transfers control to a subroutine, which is a group of instructions designed to perform a specific task.
- **Return:** Returns control back to the main program after the subroutine finishes execution.

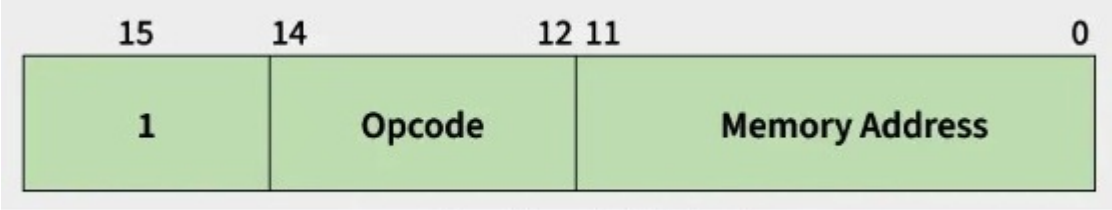
Instruction Set of a Basic Computer :

A basic computer uses a 16-bit Instruction Register (IR). An instruction can be classified as a memory reference, register reference, or input/output instruction based on specific bits in the instruction format.

1. Memory Reference Instructions

These instructions use a **memory address as an operand**, while the other operand is always the **accumulator**.

- The instruction consists of a 12-bit address, a 3-bit opcode (not equal to 111), and a 1-bit addressing mode for direct or indirect addressing.
- These instructions operate on data stored in memory.

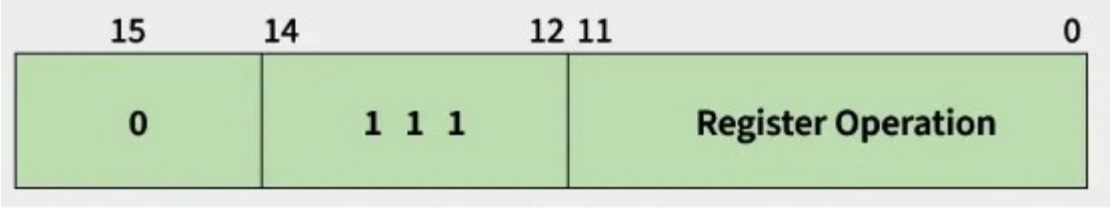


Memory Reference Instruction Format

2. Register Reference Instructions

Register reference instructions perform operations directly on CPU registers rather than memory locations.

- IR(14–12) is equal to 111 and this value distinguishes register reference instructions from memory reference instructions.
  IR(15) is equal to 0 and this value differentiates them from input/output instructions.
- The remaining 12 bits are used to specify the register operation to be performed.

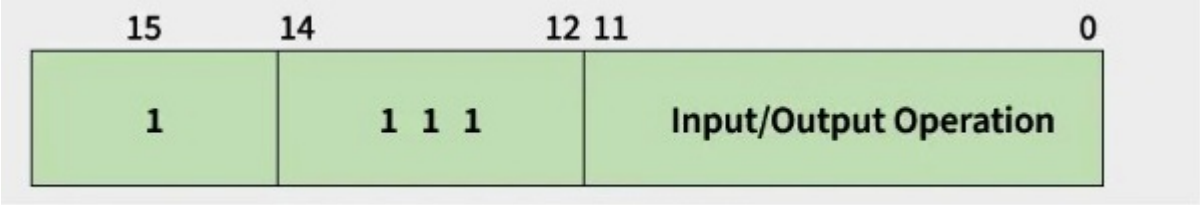


Register Reference Instruction Format

3. Input/Output Instructions

Input/output instructions are used for communication between the computer and external devices.

- IR(14–12) is equal to 111 and this distinguishes input output instructions from memory reference instructions.
- IR(15) is equal to 1 and this distinguishes them from register reference instructions.
- The remaining 12 bits are used to specify the input output operation.



Input/Output Instruction Format

Example of ADD Instruction Execution

Assume that memory address **0001** contains the value **5** and the **Accumulator (AC)** currently holds the value **10**. When the **ADD** instruction is executed, the CPU fetches the value from memory, adds it to the contents of the accumulator, and stores the result back in the accumulator.

$$AC \leftarrow AC + M[0001]$$

Result: AC = 10 + 5 = 15

Uses of Basic Computer Instructions

- **Data Manipulation:** Basic computer instructions are used to move and process data within the computer system, including transferring data between memory and the CPU and performing arithmetic and logical operations.
- **Control Flow:** These instructions control the sequence of program execution by enabling conditional branching, looping, and function calls.
- **Input and Output Operations:** Basic instructions allow data to be transferred between the computer system and external devices such as keyboards, displays, printers, and storage devices.

- **Program Execution:** They support program execution by fetching, decoding, and executing instructions and managing the movement of data required by programs.
- **Operating System Support:** Basic computer instructions provide low-level functionality that the operating system uses to implement tasks such as process control, interrupt handling, and resource management.

Issues of Basic Computer Instructions

- **Complexity:** Basic computer instructions can be difficult to understand, especially for beginners, which makes programming more challenging.
- **Limited Functionality:** These instructions support only simple operations. Complex tasks often require writing additional instructions, increasing program length and effort.
- **Compatibility:** Instruction sets may vary across different computer systems. This variation can require programmers to write system-specific code, which is time-consuming and inefficient.
- **Security:** Basic computer instructions can be vulnerable to security issues such as buffer overflows and code injection attacks.
- **Maintenance:** Programs written using basic instructions can be difficult to maintain as systems grow more complex and codebases become larger, requiring significant time and resources.