Instruction Set Architecture

It defines the set of instructions that a computer's central processing unit (CPU) can execute. It includes the specific operations a CPU can perform, the operands for each operation, the addressing modes, and the encoding of instructions.

Types of Instructions

1. Arithmetic Instructions: These instructions are used to perform arithmetic operations like addition, subtraction, multiplication, and division. They typically operate on data in registers or memory.
2. Logical Instructions: Logical instructions perform logical operations like AND, OR, XOR, and NOT on binary data. They are often used for tasks such as bit manipulation and boolean logic.
3. Data Transfer Instructions: These instructions move data between registers and memory. They include operations like load (moving data from memory to a register) and store (moving data from a register to memory).
4. Control Instructions: Control instructions manage the flow of a program. They include operations like branching (jumping to another part of the program), conditional branching (making decisions based on certain conditions), and function call/return instructions.
5. Comparison Instructions: Comparison instructions are used to compare two values and set flags or registers based on the result. They are often used in conditional branching.

Addressing Modes

Addressing modes define how the operands for instructions are specified in machine code. An addressing mode specifies how the CPU should interpret or calculate the memory address from which data should be fetched or to which it should be stored.

Types of Addressing Modes

1. Immediate Addressing: In this mode, the operand is a constant value or immediate data embedded directly within the instruction. For example, `MOV R1, #5` moves the value 5 into register R1.
2. Direct Addressing: Also known as absolute addressing, this mode uses a memory address directly. For example, `LOAD R4, 0x1000` loads the value from memory address 0x1000 into register R4.
3. Indirect Addressing: In this mode, a memory address is specified, but the data at that address is not accessed directly. Instead, the CPU uses the specified address to access another address or a register containing the actual memory address. This is often used for pointers. For example, `LOAD R5, [R6]` loads the value from the memory address pointed to by the value in register R6 into register R5.
4. Indexed Addressing: This mode involves adding an offset to a base address to calculate the effective address. It's often used in array access and data structures. For example, `LOAD R7, [R8 + #4]` loads the value from the memory address at R8 + 4 into register R7.