Batch: E2	**Roll No.: 16010123325**
Experiment / assignment / tutorial No._____	

TITLE: Study of RISC and CISC Architecture

AIM: Understanding RISC and CISC Architecture

Expected OUTCOME of Experiment: (Mentions the CO/CO's attained)

Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.
2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.
3. Dr. M. Usha, T. S. Srikanth, "Computer System Architecture and Organization", First Edition, Wiley-India.

Pre Lab/ Prior Concepts:

Reduced Set Instruction Set Architecture (RISC)

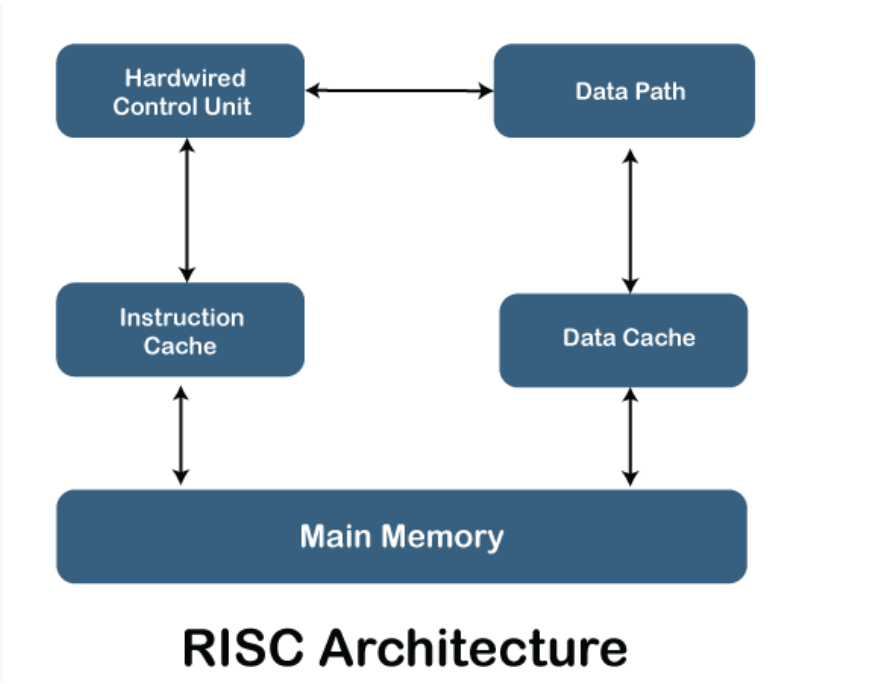
The main idea behind is to make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating and storing operations just like a load command will load data, store command will store the data.

Complex Instruction Set Architecture (CISC)

The main idea is that a single instruction will do all loading, evaluating and storing operations just like a multiplication command will do stuff like loading data, evaluating and storing it, hence it's complex.Both approaches try to increase the CPU performance

RISC Architecture

1. Diagram of RISC Architecture:



RISC Architecture

2. Brief Explanation of each component

- **Hardwired Control Unit**:
  - The control unit directs the operations of the processor by generating control signals. In a hardwired control unit, the signals are generated using combinational logic circuits, making it faster but less flexible than a microprogrammed control unit.
- **Data Path**:
  - This component is responsible for moving and processing data within the CPU. It consists of elements like registers, arithmetic logic units (ALUs), and buses, allowing data to flow from one part of the processor to another.

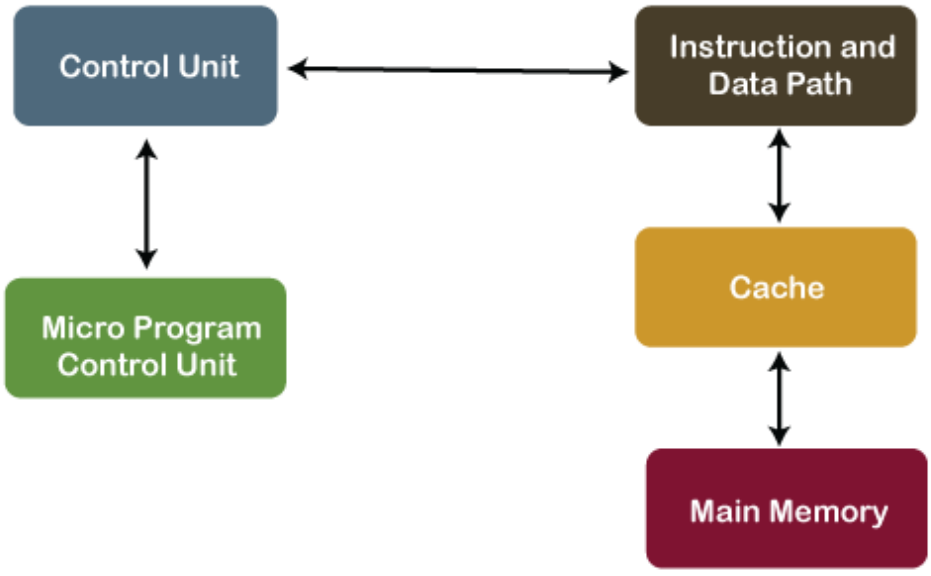
- **Instruction Cache**:
  - The instruction cache temporarily stores instructions fetched from the main memory. This helps to speed up the execution of programs by reducing the time spent fetching instructions from the main memory.
- **Data Cache**:
  - Similar to the instruction cache, the data cache stores frequently accessed data to reduce the latency of memory access. Instead of retrieving data directly from slower main memory, the CPU accesses it from this faster memory.
- **Main Memory**:
- This is the primary storage for instructions and data. It is slower than caches but provides a larger storage capacity. The CPU fetches both data and instructions from the main memory when not available in the caches.

3. RISC Processor Instruction Set Examples with explanation (Any 2)

- **LOAD R1, 1000:-** This instruction loads the data from memory address `1000` into register `R1`.
- **ADD R1, R2, R3:-** This instruction adds the contents of register `R2` and register `R3`, then stores the result in register `R1`.

CISC Architecture

1. Diagram of CISC Architecture:



CISC Architecture

2. Brief Explanation of each component

The **Control Unit** manages the execution of complex instructions by generating control signals to coordinate the processor's operations.

The **Micro Program Control Unit** breaks down complex instructions into micro-operations for sequential execution.

The **Instruction and Data Path** handles the movement and processing of data, including arithmetic and logical operations.

The **Cache** stores frequently accessed data and instructions to speed up processing.

The **Main Memory** holds all program data and instructions, accessed when not available in the cache.

3. CISC Processor Instruction Set Examples with explanation (Any 2)

- **MULT AX, BX :-** This instruction multiplies the values in the registers `AX` and `BX` and stores the result in `AX`.

- **MOV AX, [1000] :-** The **MOV** instruction copies data from one location to another. In this example, it moves the data from memory location `1000` into the register `AX`.

Post Lab Descriptive Questions

Write a tabular comparative analysis of RISC v/s CISC

RISC vs. CISC

CISC	RISC
Emphasis on hardware	Emphasis on software
Multiple instruction sizes and formats	Instructions of same set with few formats
Less registers	Uses more registers
More addressing modes	Fewer addressing modes
Extensive use of microprogramming	Complexity in compiler
Instructions take a varying amount of cycle time	Instructions take one cycle time
Pipelining is difficult	Pipelining is easy

Conclusion: In conclusion, **RISC** focuses on simple, fast instructions that execute in one cycle, making it efficient for pipelining and easier to implement. In contrast, **CISC** uses complex instructions that reduce code size but take multiple cycles to execute, making it better suited for tasks requiring fewer lines of code at the cost of execution speed.

Date: _____ **Signature of faculty in-charge**