B.Tech (CSE) Subject Name: **Computer Organization** Topic: Stored program concept Total Time for Lesson: 50 mins Teacher: Shanta Phani	Year / Semester: 2nd / 1st Subject Code: **CS 303** Subtopics: Introduction to RISC architectures. RISC vs CISC architectures Department: CSE/ IT

RISC (*Reduced Instruction Set Computer)*

RISC stands for *Reduced Instruction Set Computer*. To execute each instruction, if there is separate electronic circuitry in the control unit, which produces all the necessary signals, this approach of the design of the control section of the processor is called RISC design. It is also called *hard-wired approach*.

Examples of RISC processors:

- IBM RS6000, MC88100
- DEC's Alpha 21064, 21164 and 21264 processors

Features of RISC Processors:

The standard features of RISC processors are listed below:

- RISC processors use a small and limited number of instructions.
- RISC machines mostly uses hardwired control unit.
- RISC processors consume less power and are having high performance.
- Each instruction is very simple and consistent.
- RISC processors uses simple addressing modes.
- RISC instruction is of uniform fixed length.

CISC (Complex Instruction Set Computer)

CISC stands for *Complex Instruction Set Computer*. If the control unit contains a number of micro-electronic circuitry to generate a set of control signals and each micro-circuitry is activated by a micro-code, this design approach is called CISC design.

Examples of CISC processors are:

- Intel 386, 486, Pentium, Pentium Pro, Pentium II, Pentium III
- Motorola's 68000, 68020, 68040, etc.

Features of CISC Processors:

The standard features of CISC processors are listed below:

- CISC chips have a large amount of different and complex instructions.
- CISC machines generally make use of complex addressing modes.
- Different machine programs can be executed on CISC machine.
- CISC machines uses micro-program control unit.
- CISC processors are having limited number of registers.

RISC vs CISC

	CISC	**RISC**
1.	Complexity found in hardware	Complexity on software side
2.	Memory-to-memory : load and store functionality found in a single instruction	Register-to-Register : load and store are separate instructions
3.	Less lines of code needed to provide same functionality	More instructions necessary to provide same functionality
4.	instructions not always the same size	all instructions of a uniform size
5.	instructions are difficult to decode because instructions are not uniform	instructions are easier to decode because of how they are set up-ex: opcode will always be in the same place
6.	to make use of pipelining, instructions need to be broken down to smaller components at processor level	capable of using pipelining by design