|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Instruction Set Architecture** | **Micro-architecture** | | * Interface between software and hardware. Software and compiler assumptions. Hardware promises. What the software writer needs to know and debug the system and user programs | * Implementation of ISA. The fundamental design of micro-processor. Abbreviated as uArch. It is not visible to the software.   **Advantages** | |
| **Advantages**   |  |  | | --- | --- | | * ISA changes very slowly. This offers backward compatibility. * Offers high levels of abstraction. Programmer only needs to know the semantics specified by ISA | * Micro-architecture reduces the cost by increasing reusability of hardware by implementing pipelines etc * U-arch can execute instructions in any order as long as it obeys the semantics specified by the ISA when making the instruction results visible to software Programmer have to see the order specified by the ISA |   **Disadvantages**   |  |  | | --- | --- | | * ISA changes very slowly. This makes ISA reluctant to accommodate new innovations. | Changes in microarchitecture may lead to issues like   * Power consumption * Choice of design level constraints * Speed and Cost of the hardware * Performance bottleneck by other hardware components outside the CPU | |