NAME: GABRIEL-OHANU VICTOR

MATRIC NUMBER: 17/SCI01/035

COURSE CODE: CSC310

COURSE TITLE: Computer Architecture and Organization II

1. **RISC**

**What is RISC?**  
RISC, or *Reduced Instruction Set Computer*. is a type of microprocessor architecture that utilizes a small, highly-optimized set of instructions, rather than a more specialized set of instructions often found in other types of architectures.

**History of RISC**  
The first RISC projects came from IBM, Stanford, and UC-Berkeley in the late 70s and early 80s. The IBM 801, Stanford MIPS, and Berkeley RISC 1 and 2 were all designed with a similar philosophy which has become known as RISC. Certain design features have been characteristic of most RISC processors:

* *one cycle execution time*: RISC processors have a CPI (clock per instruction) of one cycle. This is due to the optimization of each instruction on the CPU and a technique called <i.pipelining< i="">;

</i.pipelining<>

* *pipelining*: a techique that allows for simultaneous execution of parts, or stages, of instructions to more efficiently process instructions;
* *large number of registers*: the RISC design philosophy generally incorporates a larger number of registers to prevent in large amounts of interactions with memory.

**Advantages of RISC Architecture**  
 Besides performance improvement, some advantages of RISC and related design improvements are:

* A new microprocessor can be developed and tested more quickly if one of its aims is to be less complicated.
* Operating system and application programmers who use the microprocessor's instructions will find it easier to develop code with a smaller instruction set.
* The simplicity of RISC allows more freedom to choose how to use the space on a microprocessor.
* Higher-level language compilers produce more efficient code than formerly because they have always tended to use the smaller set of instructions to be found in a RISC computer.

**Disadvantages of RISC architecture**

* Mostly, the performance of the RISC processors depends on the programmer or compiler as the knowledge of the compiler plays a vital role while changing the CISC code to a RISC code
* While rearranging the CISC code to a RISC code, termed as a code expansion, will increase the size. And, the quality of this code expansion will again depend on the compiler, and also on the machine’s instruction set.
* The first level cache of the RISC processors is also a disadvantage of the RISC, in which these processors have large memory caches on the chip itself. For feeding the instructions, they require very fast memory systems.

1. **CISC**

**What is CISC?**

The term "CISC" (complex instruction set computer or computing) refers to computers designed with a full set of computer instructions that were intended to provide needed capabilities in the most efficient way. Later, it was discovered that, by reducing the full set to only the most frequently used instructions, the computer would get more work done in a shorter amount of time for most applications. Since this was called reduced instruction set computing ([RISC](https://search400.techtarget.com/definition/RISC" \t "_top)), there was now a need to have something to call full-set instruction computers - thus, the term CISC.

### **Advantages of CISC Architecture:**

* Microprogramming is easy to implement and much less expensive than hard wiring a control unit.
* It is easy to add new commands into the chip without changing the structure of the instruction set as the architecture uses general-purpose hardware to carry out commands.
* This architecture makes the efficient use of main memory since the complexity (or more capability) of instruction allows to use less number of instructions to achieve a given task.
* The compiler need not be very complicated, as the microprogram instruction sets can be written to match the constructs of high  level languages.

**Disadvantages of CISC Architecture**

* The performance of the machine slows down due to the amount of clock time taken by different instructions will be dissimilar
* Only 20% of the existing instructions is used in a typical programming event, even though there are various specialized instructions in reality which are not even used frequently.
* The conditional codes are set by the CISC instructions as a side effect of each instruction which takes time for this setting – and, as the subsequent instruction changes the condition code bits – so, the compiler has to examine the condition code bits before this happens.

1. **VLIW**

**WHAT IS VLIW?**

Very long instruction word (VLIW) describes a computer processing architecture in which a language [compiler](https://whatis.techtarget.com/definition/compiler" \t "_top) or pre-processor breaks program [instruction](https://whatis.techtarget.com/definition/instruction" \t "_top) down into basic operations that can be performed by the [processor](https://whatis.techtarget.com/definition/processor" \t "_top) in [parallel](https://whatis.techtarget.com/definition/parallel" \t "_top) (that is, at the same time). These operations are put into a very long instruction [word](https://whatis.techtarget.com/definition/word" \t "_top) which the processor can then take apart without further analysis, handing each operation to an appropriate functional unit.

VLIW is sometimes viewed as the next step beyond the reduced instruction set computing ( [RISC](https://search400.techtarget.com/definition/RISC" \t "_top) ) architecture, which also works with a limited set of relatively basic instructions and can usually execute more than one instruction at a time (a characteristic referred to as *superscalar* ). The main advantage of VLIW processors is that complexity is moved from the hardware to the software, which means that the hardware can be smaller, cheaper, and require less power to operate. The challenge is to design a compiler or pre-processor that is intelligent enough to decide how to build the very long instruction words. If dynamic pre-processing is done as the program is run, performance may be a concern.

**Advantage of VLIW Architecture**

* the number of FUs can be increased without needing additional sophisticated hardware to detect parallelism, like in superscalars.
* VLIW is less complex than the Superscalar because VLIW.
* Because VLIW is implemented at the software level, all available storage space can be used. But for superscalar, its implemented in the hardware and some space will have to be allocated for the hardware so not all available storage space can be used.

**Disadvantage of VLIW Architecture**

* Superscalar machines are able to dynamically issue multiple instructions each clock cycle from a conventional linear instruction stream while VLIW is static.