Computers are a key part of our everyday lives, from the machines we use for work to the smartphones and smartwatches we rely on.

- All computers, no matter their size, are based around a set of rules stating how software and hardware join together and interact to make them work. This is what is known as computer architecture
- Computer architecture determines how a computer's components exchange electronic signals to enable input, processing, and output.
- Computer architecture is defined as the end-to-end structure of a computer system that determines how its components interact with each other in helping execute the machine's purpose (i.e., processing data).

The two commonly used computer architectures are:

1. THE VON NEUMANN ARCHITECTURE

Mathematician John von Neumann and his colleagues proposed the von Neumann architecture in 1945, which stated that a computer consists of: a processor with an arithmetic and logic unit (ALU) and a control unit; a memory unit that can communicate directly with the processor using connections called buses; connections for input/output devices; and a secondary storage for saving and backing up data.

The central computation concept of this architecture is that instructions and data are both loaded into the same memory unit, which is the main memory of the computer and consists of a set of addressable locations. The processor can then access the instructions and data required for the execution of a computer program using dedicated connections called buses – an address bus which is used to identify the addressed location and a data bus which is used to transfer the contents to and from a location.

THE PROS AND CONS OF THE VON NEUMANN ARCHITECTURE

Computers as physical objects have changed dramatically in the 76 years since the von Neumann architecture was proposed. Supercomputers in the 1940s took up a whole room but had very basic functionality, compared to a modern smartwatch which is small in size but has dramatically higher performance. However, at their core, computers have changed very little and almost all of those created between then and now have been run on virtually the same von Neumann architecture.

There are a number of reasons why the von Neumann architecture has proven to be so successful. It is relatively easy to implement in hardware, and von Neumann machines are deterministic and introspectable. They can be described mathematically and every step of their computing process is understood. You can also rely on them to always generate the same output on one set of inputs.

The biggest challenge with von Neumann machines is that they can be difficult to code. This has led to the growth of computer programming, which takes real-world problems and explains them to von Neumann machines.

When a software program is being written, an algorithm is reduced to the formal instructions that a von Neumann machine can follow. However, the challenge is that not all algorithms and problems are easy to reduce, leaving unsolved problems.

2. HARVARD ARCHITECTURE

Another popular computer architecture, though less so than the von Neumann architecture, is Harvard architecture.

The Harvard architecture keeps instructions and data in separate memories, and the processor accesses these memories using separate buses. The processor is connected to the ‘instructions memory’ using a dedicated set of address and data buses, and is connected to the ‘data memory’ using a different set of address and data buses. This architecture is used extensively in embedded computing systems such as digital signal processing (DSP) systems, and many microcontroller devices use a Harvard-like architecture.

conclusion with quick differences.

- **von Neumann Architecture:** This is the traditional architecture that is still widely used today. In von Neumann Architecture, the CPU, memory, and input/output devices are all interconnected through a shared bus. It has a sequential execution model, with instructions and data stored in the same memory. While it is simple and easy to implement, it can sometimes suffer from performance limitations due to the bottleneck created by the shared bus.

- **Harvard Architecture:** This architecture separates the instruction memory and data memory into two separate units, allowing simultaneous access to both. This can potentially improve performance, especially in cases where the CPU needs to fetch instructions and access data simultaneously. However, it can be more complex to implement and may require additional hardware.

Ultimately, the choice of architecture depends on the specific requirements of the system and the trade-offs between simplicity and performance. Different architectures may be more suitable for different applications, and advancements in technology continue to evolve architectures to meet growing demands.