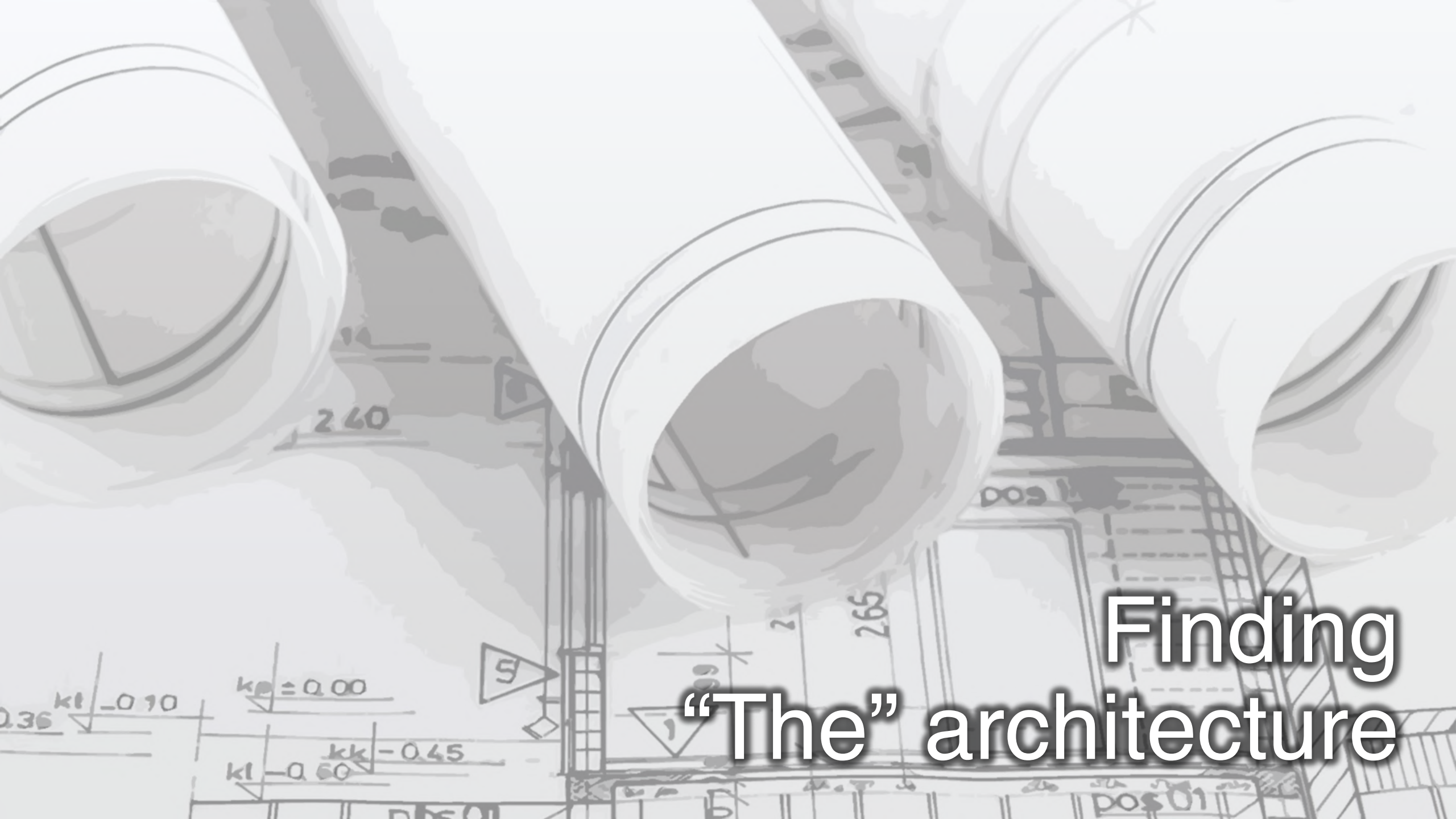
An introduction to Computer Architecture

Whitening the computer black box

Gabriele Gaetano Fronzé

INFN Istituto Nazionale di Fisica Nucleare
UNIVERSITA' DEGLI STUDI DI TORINO
ELEMENTO MODULAR CLOUD



Finding “The” architecture

The Von Neumann architecture (1945)

John Von Neumann (1903-1957)
Physicist and Mathematician

The human brain is a great machine:

- Reads and produces sentences;
- Receives information from the senses;
- Transmits information;
- Is able to store memories either forever or for a limited time;
- Learns new skills and stores them alongside memories.

The Von Neumann architecture (1945)

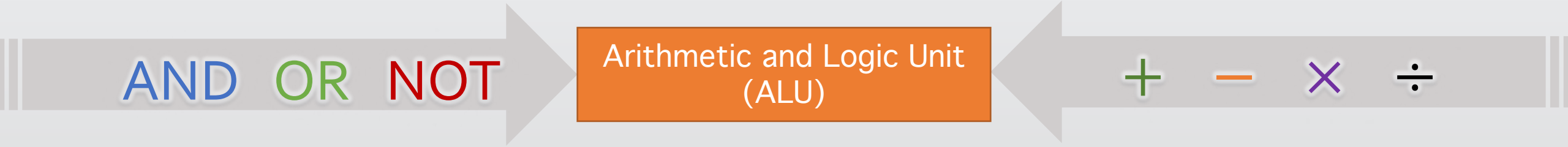
John Von Neumann (1903-1957)
Physicist and Mathematician

A calculator is a great machine:

- Reads and produces data;
- Receives data from the inputs
- Transmits data
- Is able to store data either forever or for a limited time;
- Stores instructions alongside data

The Von Neumann architecture (1945)

Reads and produces `data` → Can perform instructions



The ALU performs basic (yet generic) arithmetic and logic operations.

Grammar, language syntax, complex calculus, and any other kind of job can be reduced to basic operations.

Even basic arithmetic operations can be realised using basic logical conditions.

The Von Neumann architecture (1945)

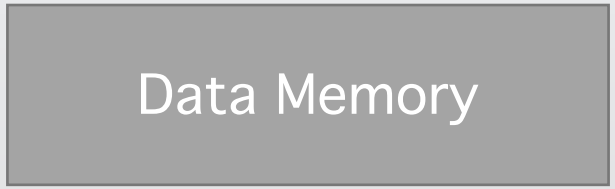
Receives information from the `inputs` → Has input channels

Transmits `data` → Has output channels

Input and output channels are used to retrieve new data and to "publish" results.

The Von Neumann architecture (1945)

Is able to store `data` [...] → Has some memory to store data



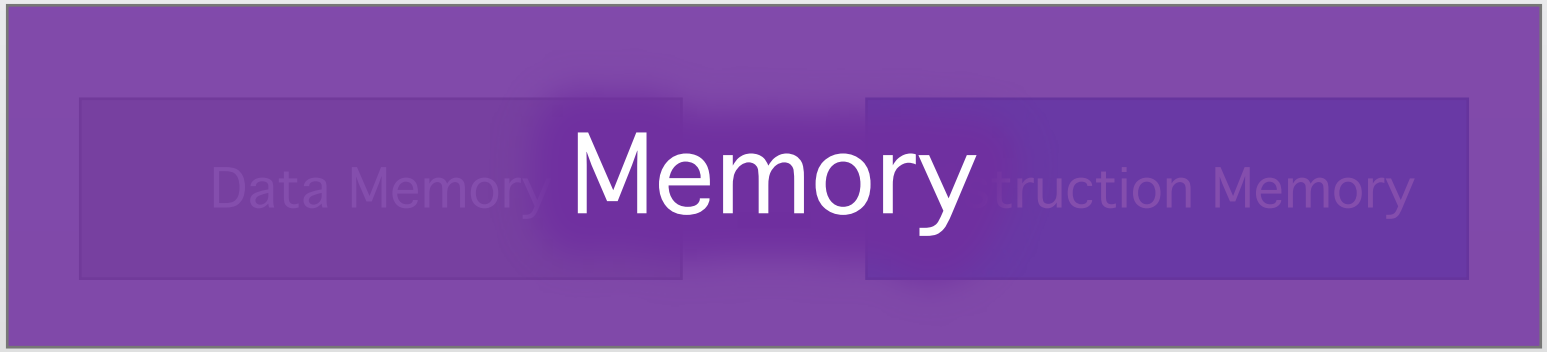
Data can be stored after computation for future usage.

At least two types of memory are needed:

1. Permanent memory for important data;
2. Volatile memory for flushable data.

The Von Neumann architecture (1945)

`Stores instructions` alongside `data` → Memory can <u>seamlessly</u> contain instructions or data

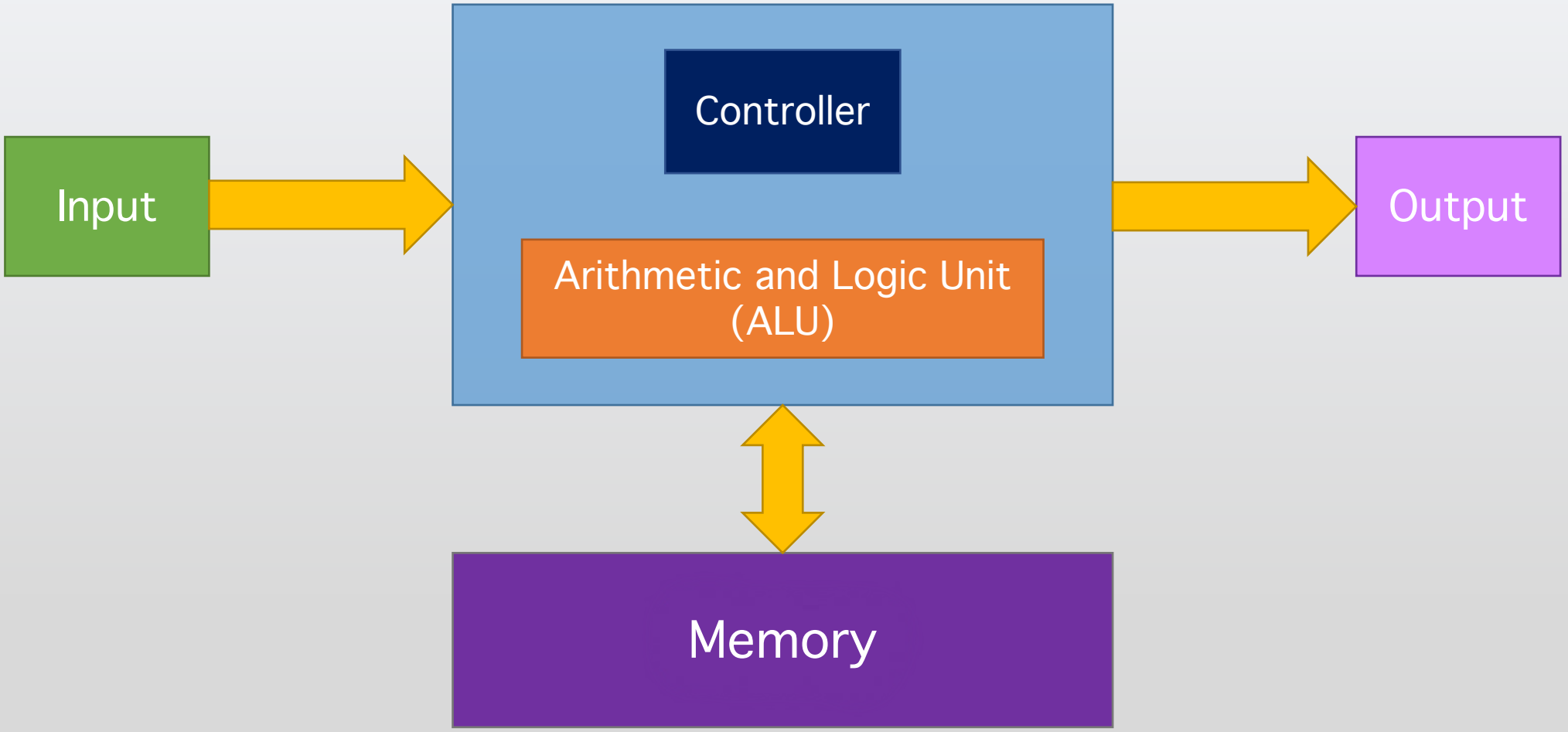


Data and instructions are stored in the same memory, hence in the same address space.

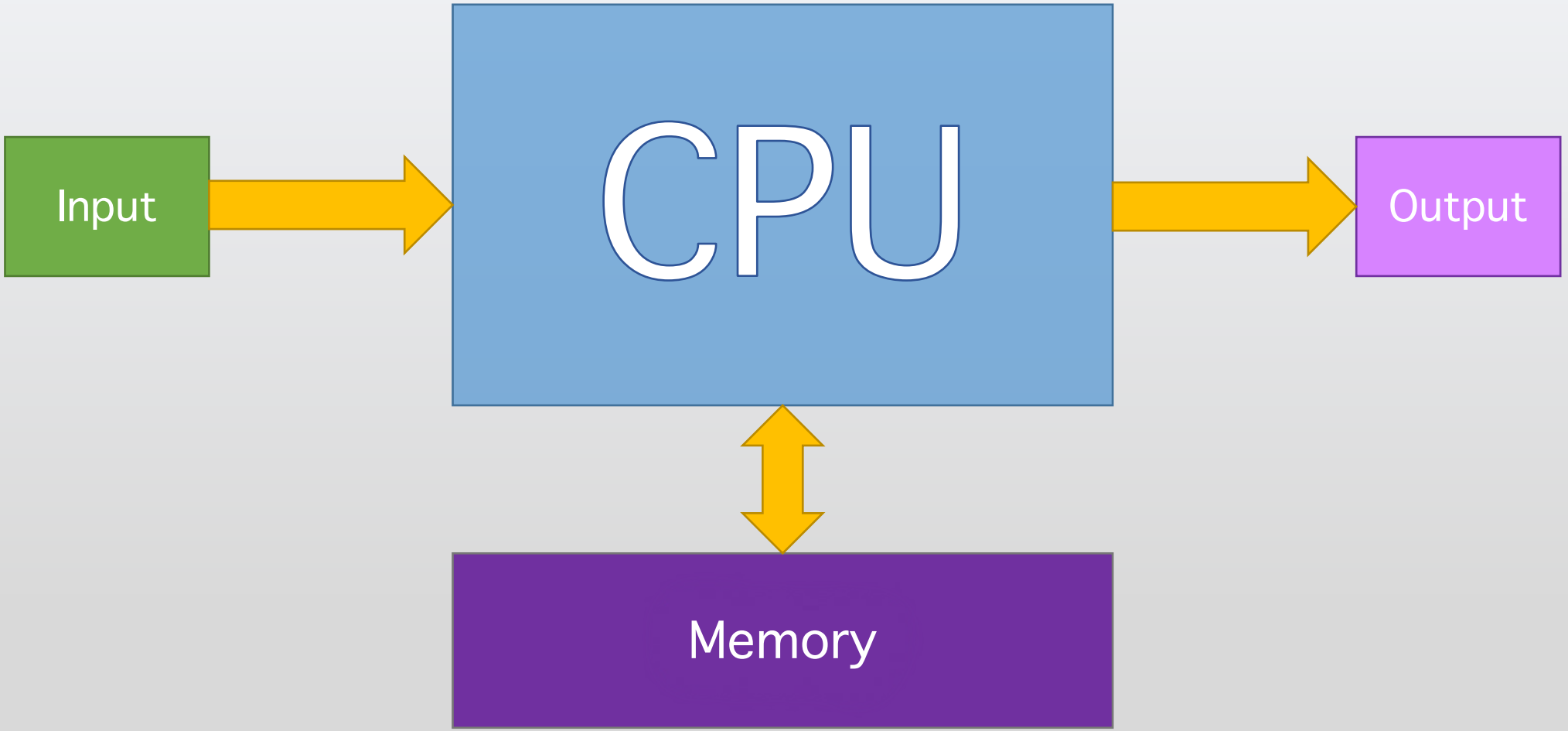
Accessing data and/or instructions is handled in the same way using the same electronics to access it.

It is possible to treat instructions as data and data as instructions.

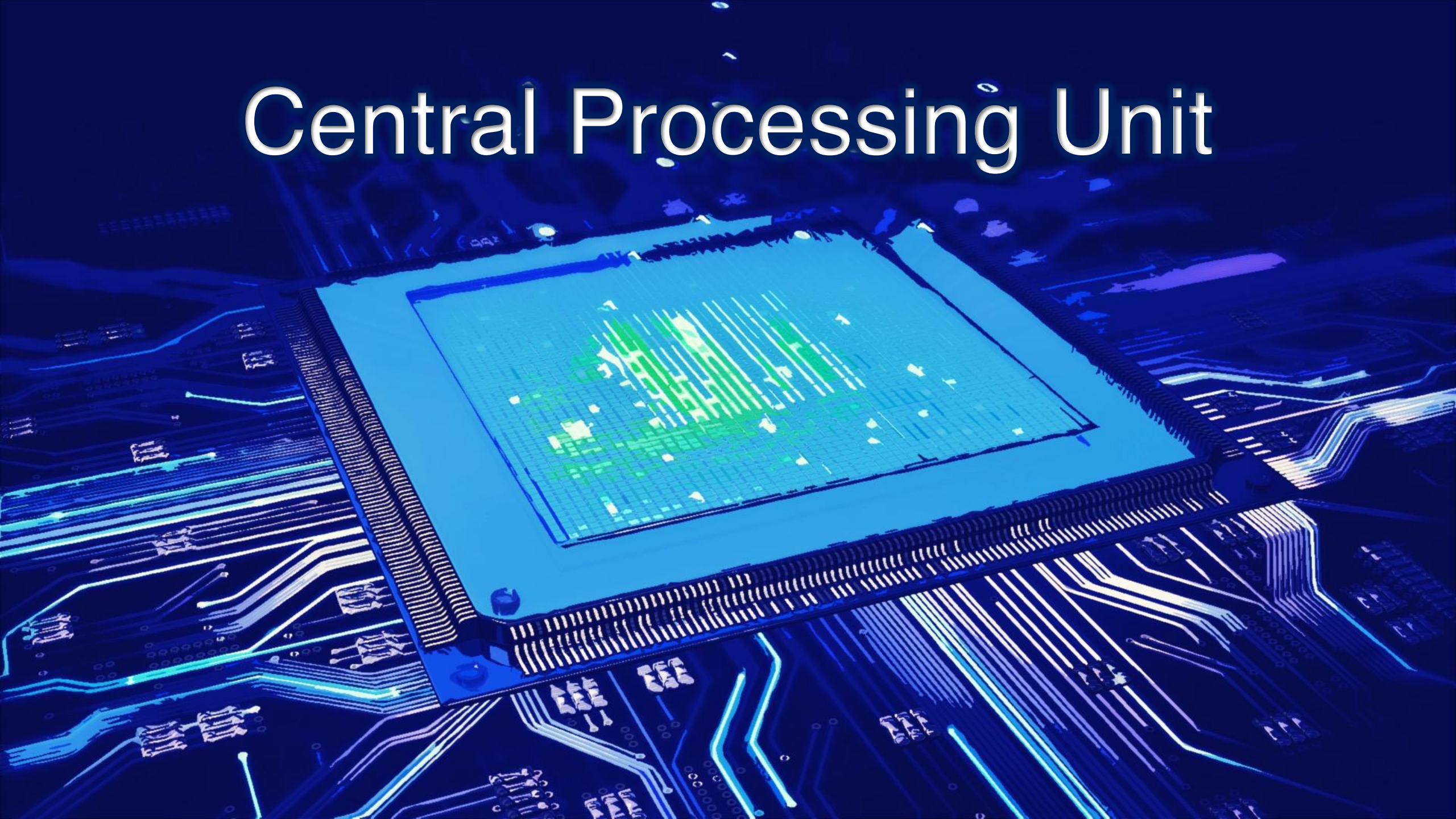
The Von Neumann architecture (1945)



The Von Neumann architecture (1945)



Central Processing Unit



Central Processing Unit

A CPU performs arithmetic, logic and data transfer operations

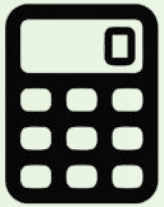
Possible instructions are defined in standard Instruction Sets Architectures (ISA) such as:

Central Processing Unit

In a ISA specification, several kinds of instructions are foreseen

Data handling

Describe how to load, store, move data and how to set to constant a values.



Arithmetic Logic

Describe how to perform bitwise comparisons, logical conditions and arithmetical operations on integer and floating point values.

Dataflow

Describe how to perform a branching (conditional, indirect or direct), how to call a sub-routine and how to return to the main flow.

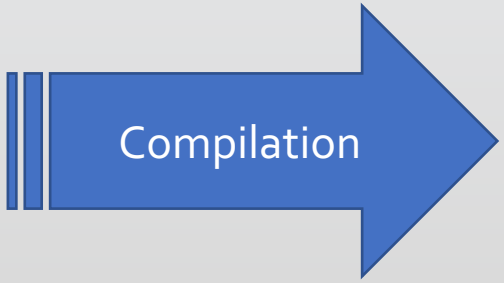
Head to agner.org for a complete list of available uops for every CPU: http://www.agner.org/optimize/instruction_tables.pdf

Central Processing Unit

By compiling code, the high level program one wrote is converted by the compiler in a binary file which contains the corresponding ISA instructions calls:

```
int sum(int num, int num2) {
     return num + num2;
}
```

Simple integer adder code



GCC 9.3 with "–m32"

```
sum(int, int):
push  ebp
mov   ebp, esp
mov   edx, DWORD PTR [ebp+8]
mov   eax, DWORD PTR [ebp+12]
add   eax, edx
pop   ebp
ret
```

Corresponding x86 compliant assembly code

Interactive compiler at https://godbolt.org

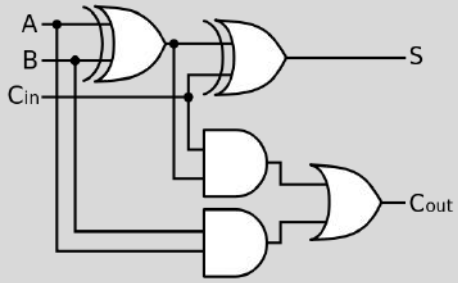
Central Processing Unit

Each CPU implements several micro-operations (uops) in form of digital electronic circuitry.

Each instruction described in the ISA is written as a call to a single or multiple micro-operations (uops).

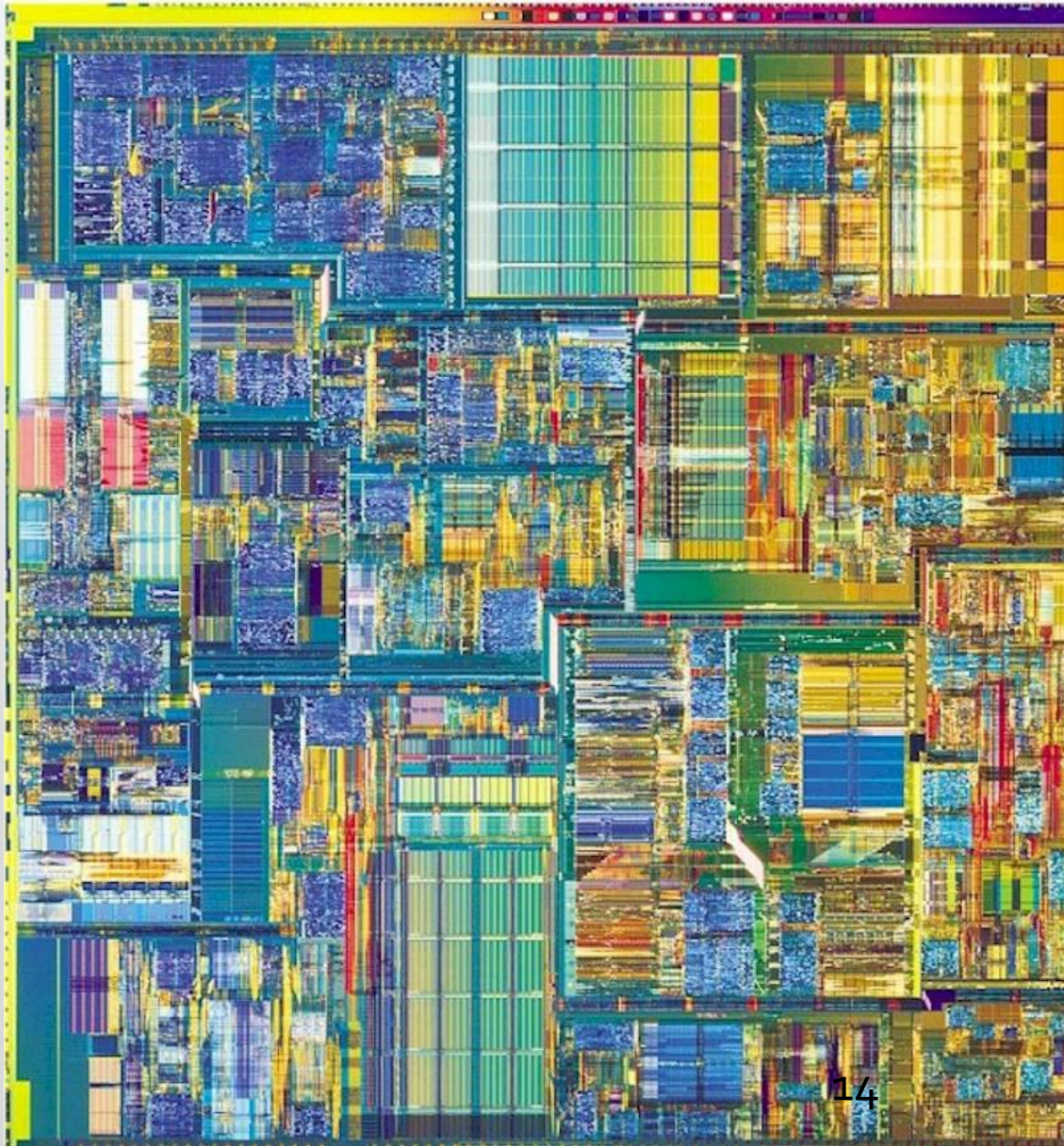
A specific CPU must support the instructions described by a supported ISA, via the implemented uops.

e.g. x86 (x64) "add" requires 32bit (64bit) of full adder stages



Single bit Full Adder

Intel Pentium 4 Willamette die uphotography →



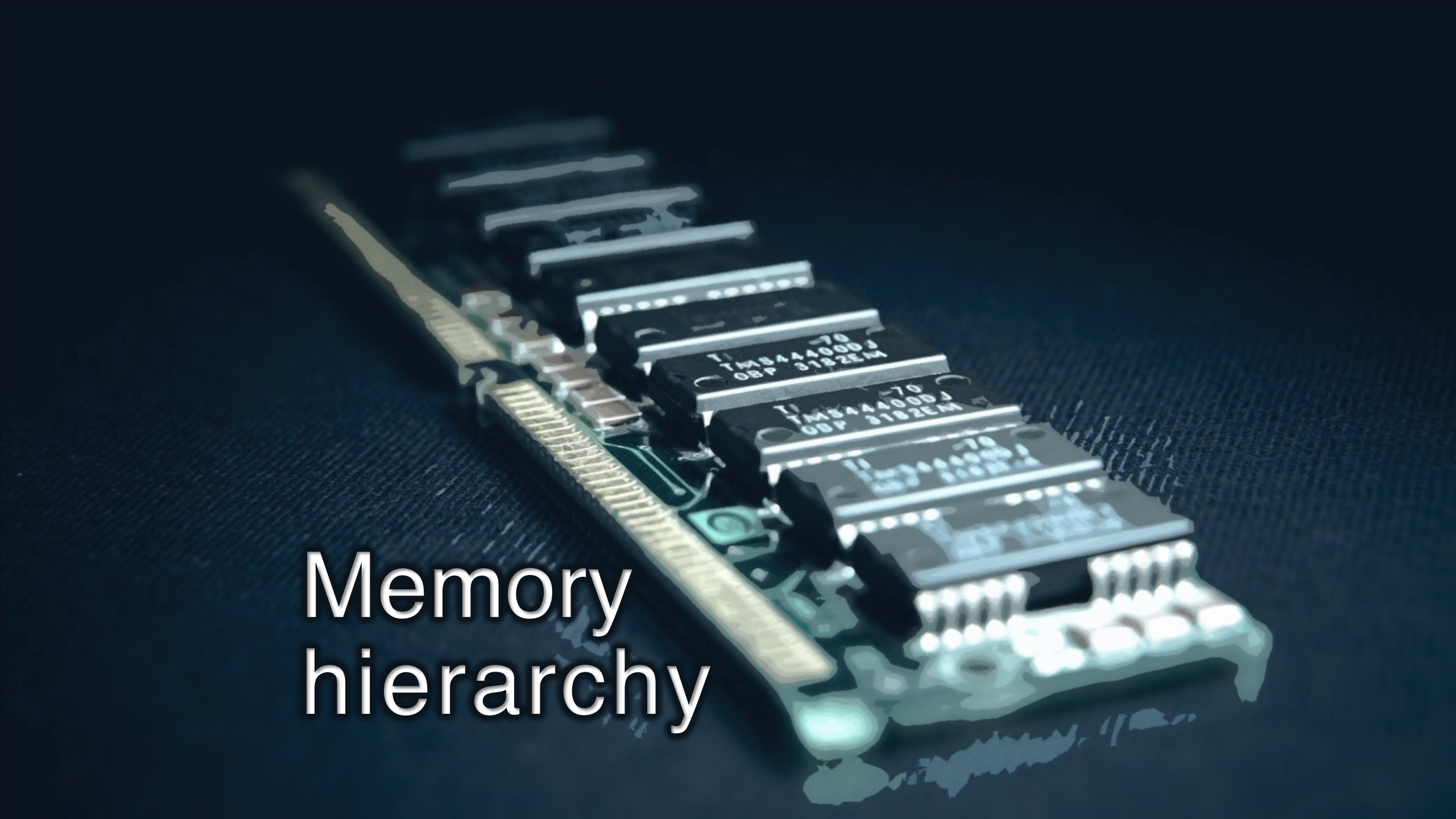
Central Processing Unit

The Instruction Cycle is made of 5 sections:

- The instruction is retrieved from memory and loaded in the instruction register(s)
- The fetched instruction is decoded and converted in a sequence of uops (micro-operations) in the executing CPU
- In case of indirect addressing, data is retrieved from memory (e.g. input value is the pointer to memory location of required data from memory
- The uops are executed by the deputed ALU (integer of floating point)
- Write in memory the results of the performed instructions

REMEMBER:

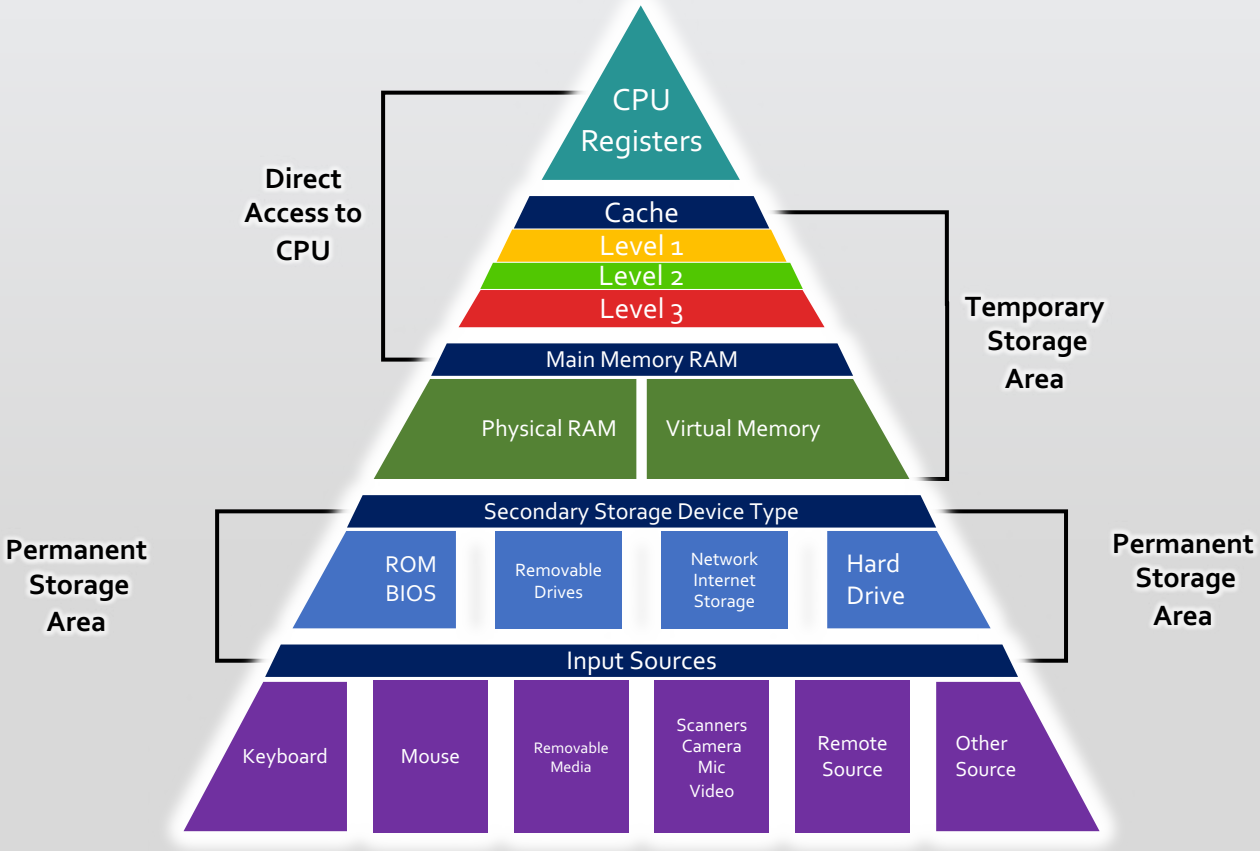
Each stage can translate in one or more uops in the CPU


Memory
hierarchy

Memory

In the Von Neumann architecture, memory contains both data and instructions.

Several memory layers with different characteristics are present in a modern computer.



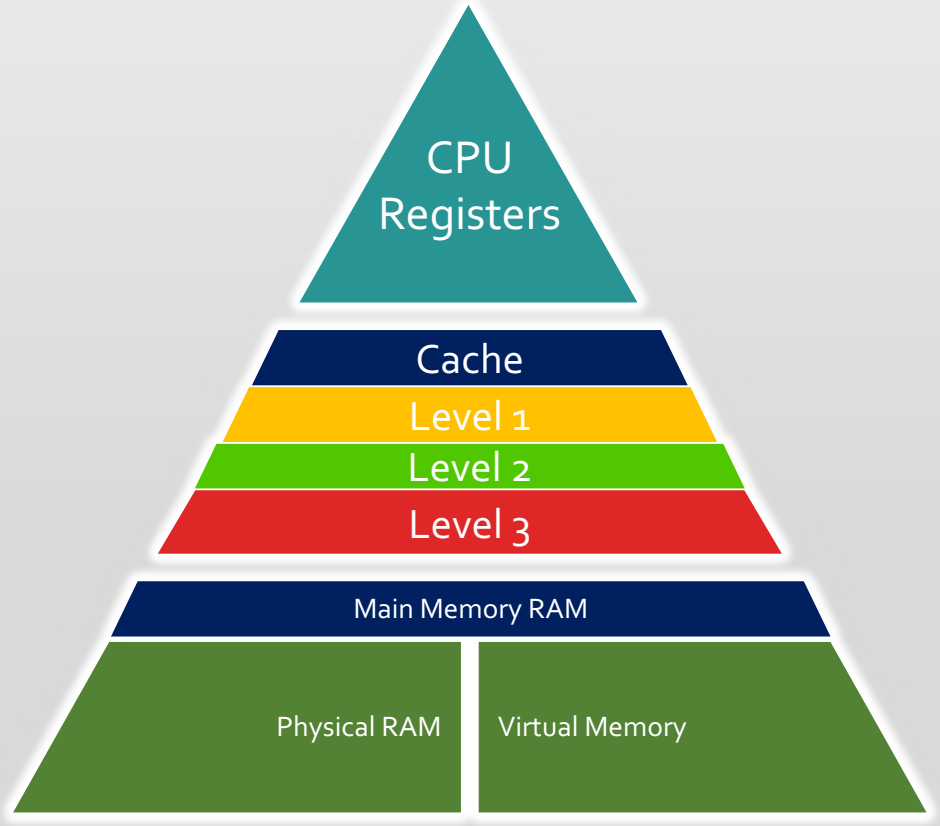
Several layers of fast memory are present inside the CPU.

The volatile memory is the RAM, which loses all the data at power off. RAM is mounted not far from the CPU.

Permanent memory supports are the slowest ones and can be placed beyond local or wide area connections.

Memory

CPU has direct access only to a lower part of the memory layers



Needed data and instructions are transferred from slower and bigger memories to the faster and smaller ones.

A CPU always operates on data and instructions which have been transferred from RAM to the lower cache levels.

Single instructions and operands are loaded from lower cache levels to internal registers which can be used in the Instructions Cycle.

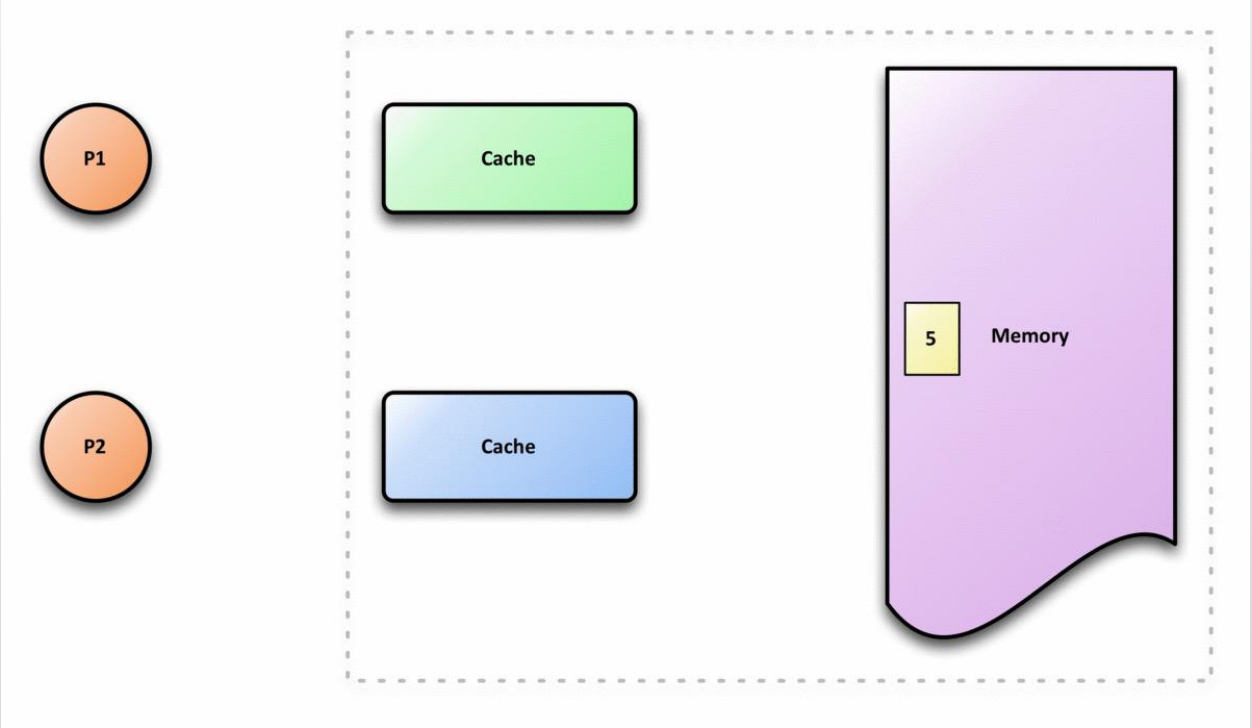
Memory

Memory coherence must be handled

Cache is a local copy of higher level shared memory data.

The results from CPU computations are written from registers to the lower caches and propagated all the way up to the system memory (RAM).

The final state of the RAM copy depends on the order of the single computations.



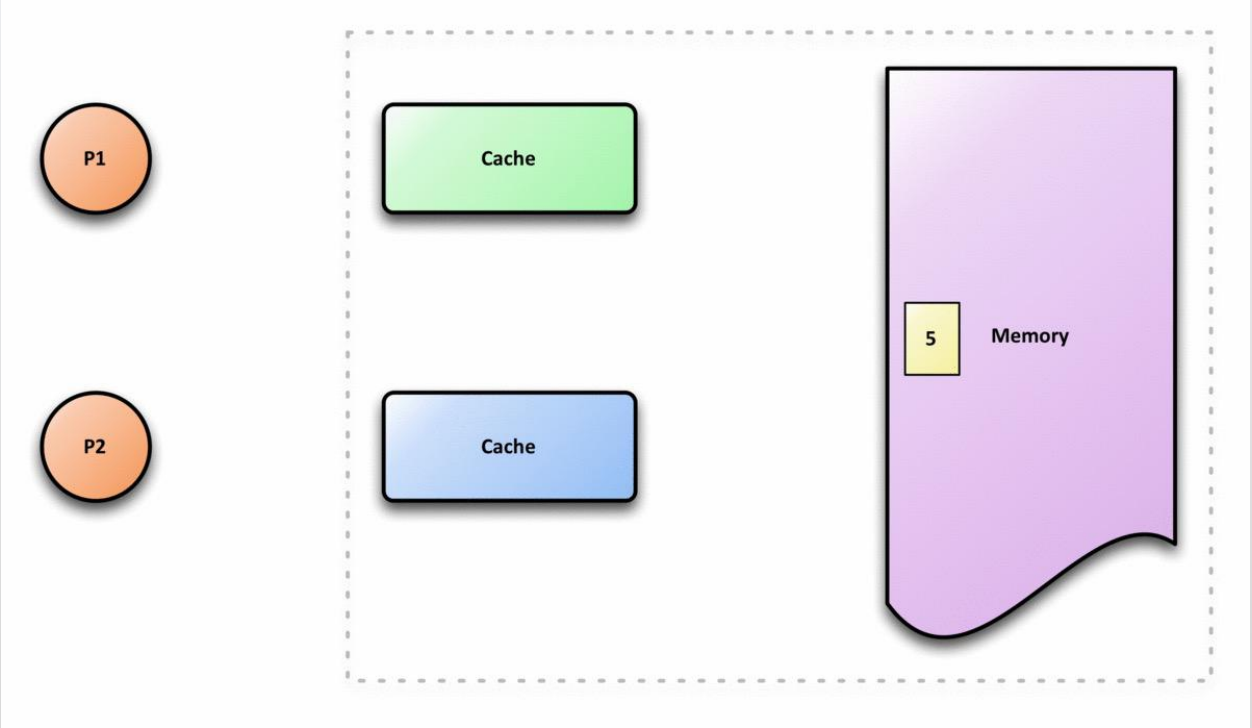
Non-coherent memory hierarchy

Memory

Memory coherence must be handled

Several cache implementations contain some sort of procedure to guarantee cache and memory coherence.

The most adopted solution is the write-invalidate method.
Whenever a write happens on data some cache has a copy of, the the given cache reads the new value at the following memory access.



Coherent memory hierarchy


Performance race

Central Processing Unit

Transistors density

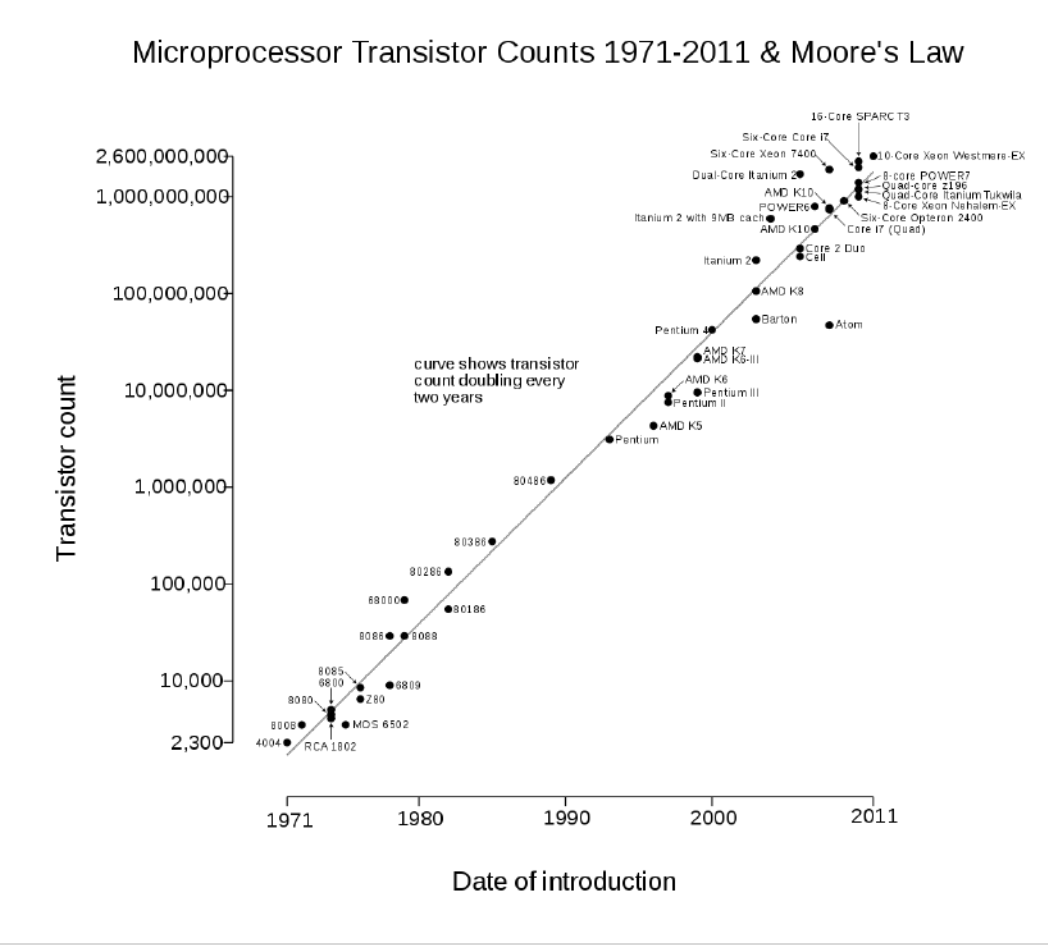
Old CPUs where able to perform rather simple uops due to a limited number of transistors.

Hardware producers wanted to reduce transistors size to pack more complex logical gates (hence uops) in the same package.

Moore's (phenomenological) law observes that:

"The transistor density of an integrated circuit doubles every two years"

This law is starting to get broken…



Central Processing Unit

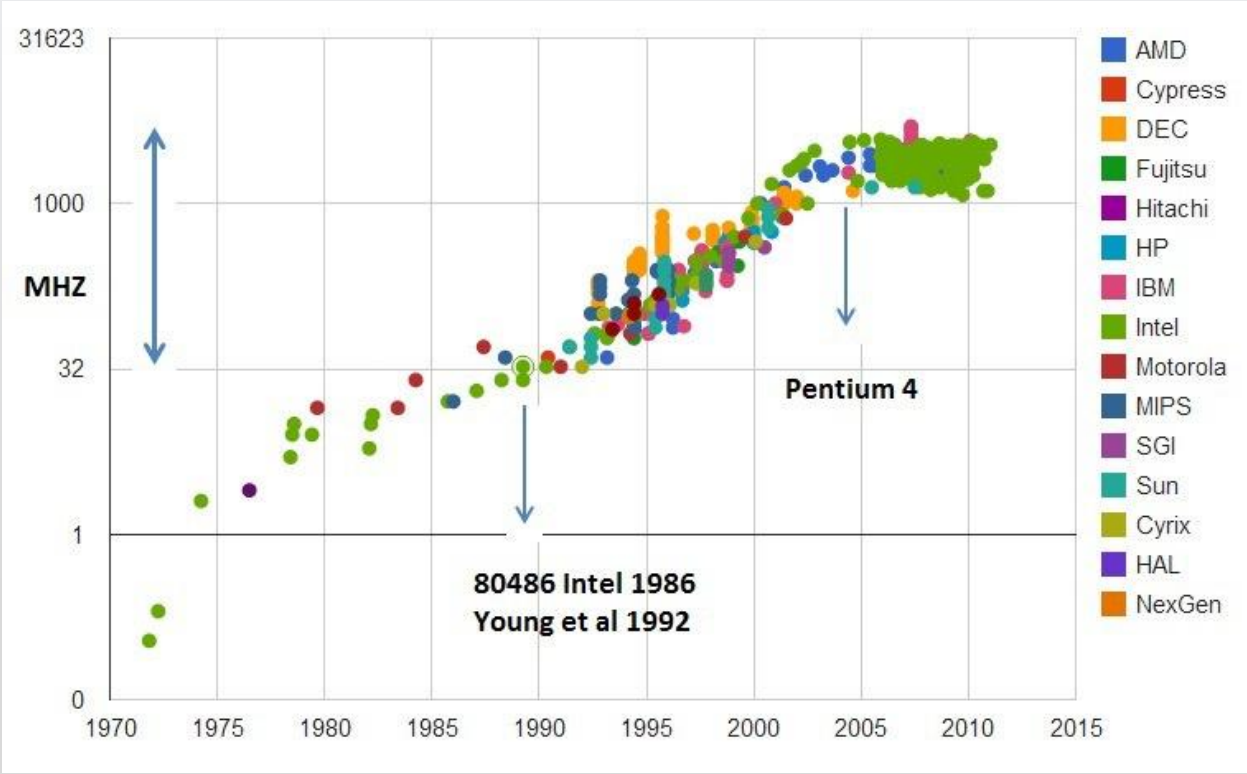
Clock frequency

The most advertised characteristic of a CPU is the clock frequency.

The clock frequency is the frequency at which several sequential uops will be executed.

The reduction of transistors size allows for an increase of CPU operating frequency.

Clock frequency is a performance indicator within a CPU family.

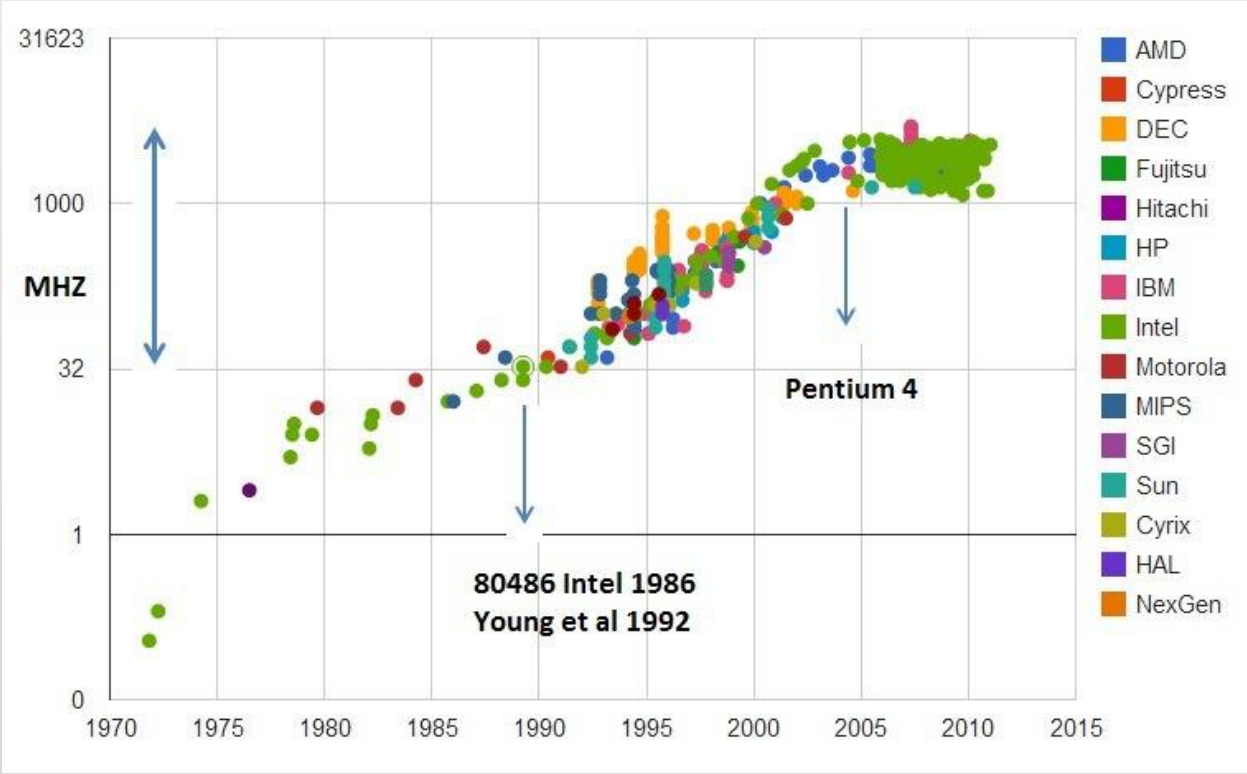


Central Processing Unit

Clock frequency

After Intel's Pentium 4 Prescott single core architecture, the CPU frequency reached the maximum value (3.8 GHz w/o overclock, 115W).

The emitted thermal power density of such CPU was higher than that of a nuclear reactor vessel (1W/mm²)!

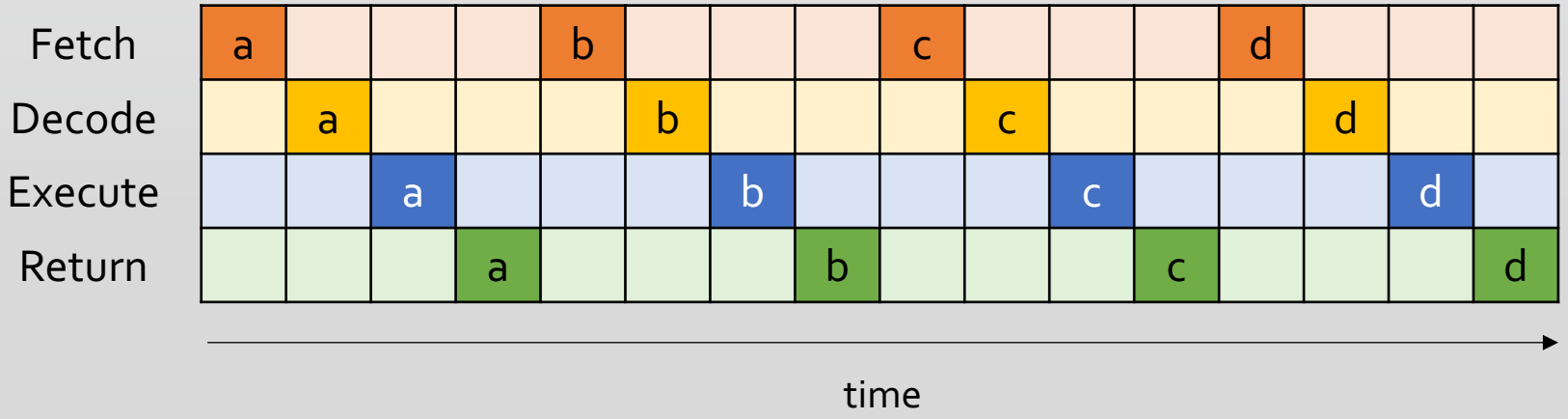


Central Processing Unit

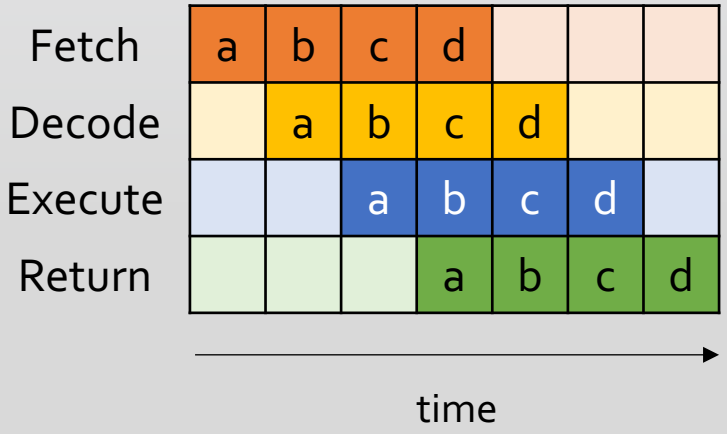
Pipelining

Typically the Instructions Cycle parts are executed by different hardware parts inside a CPU.
At least one element of each kind (fetch, decode, follow, execute and return) can be executed at the same time by a CPU.

This process allows to constantly feed the CPU ALU with uops to execute with a sensible reduction of dead time.



Simple execution of four tasks



Pipelined execution

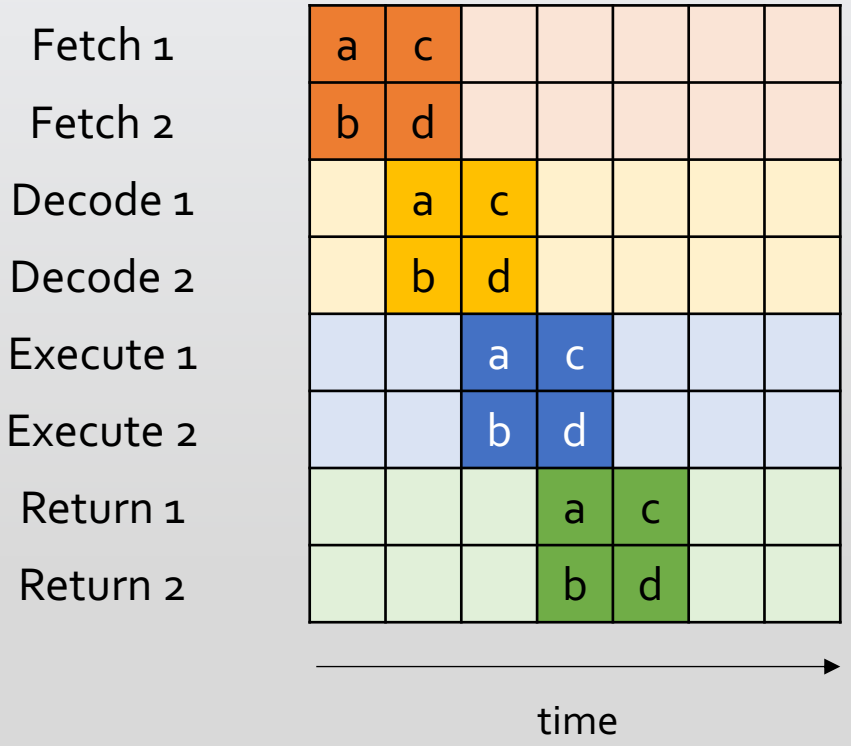
Central Processing Unit

Superscalar design

Alongside instructions pipelining superscalar design was introduced.

CPUs uops are organized in the form of ports. Each clock cycle all the ports of a CPU can be used at the same time.

Adding several copies of the same pipeline phase allows to perform the same instruction several times on different data within the same clock cycle.



Pipelined and superscalar execution

Central Processing Unit

- Pipeline execution of instruction cycles
- Replicate many times the instruction cycle stages

How to provide more power?

Central Processing Unit

- Pipeline execution of instruction cycles
- Replicate many times the instruction cycle stages

How to provide more power?

Quite literally…

Central Processing Unit

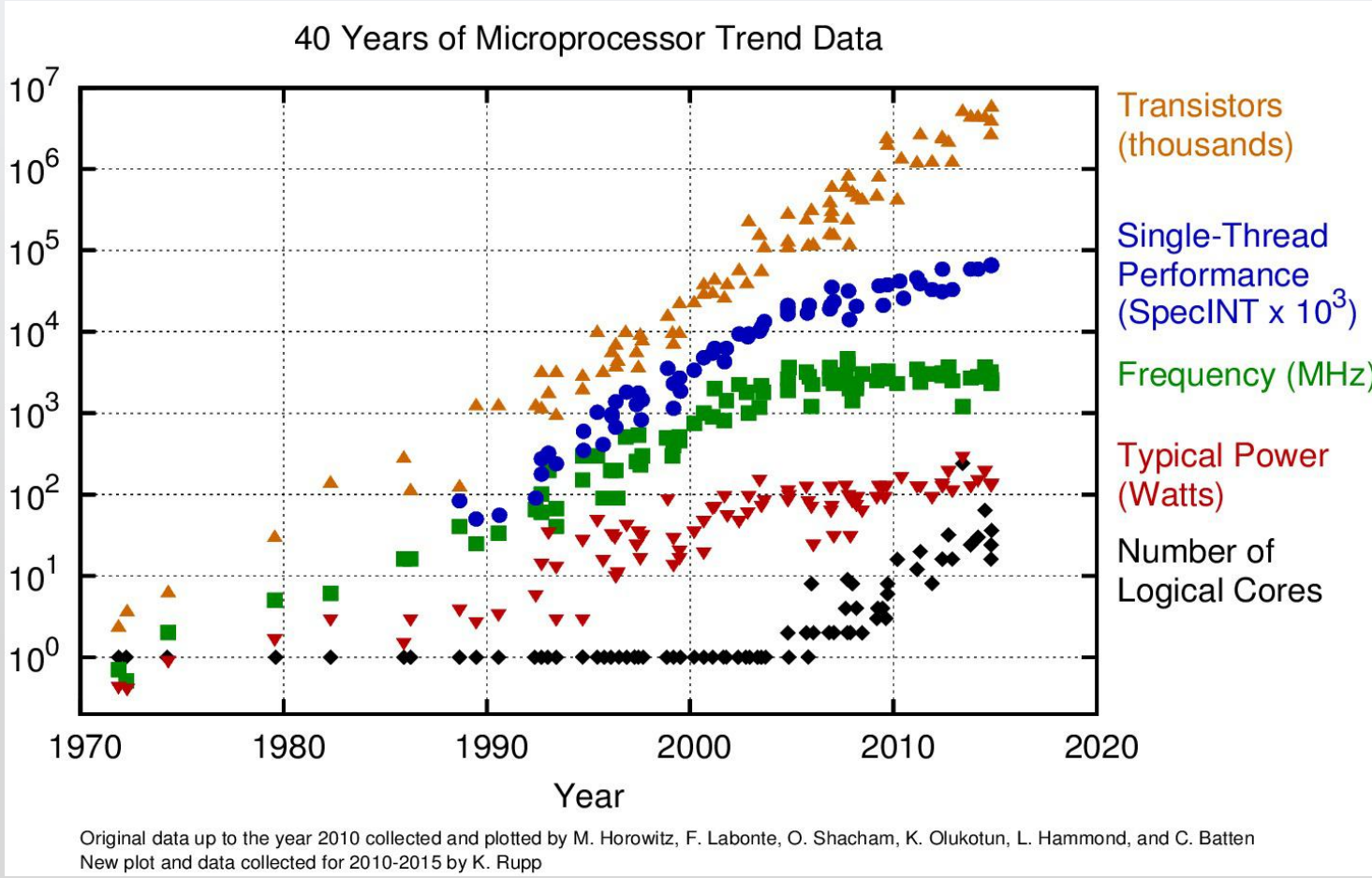
Global view

Moore's law was the leading trend up to the Pentium 4 limit in 2005.

After the implementation and maximisation of pipelining (P4 Prescott had 31 stages) and superscalar design (Pentium Pro Family) progress reached horizontal asymptotes.

Starting in May 2005 with AMD Athlon X2 and Pentium D dual core CPUs were introduced.

Since then the core count rarely stopped growing…

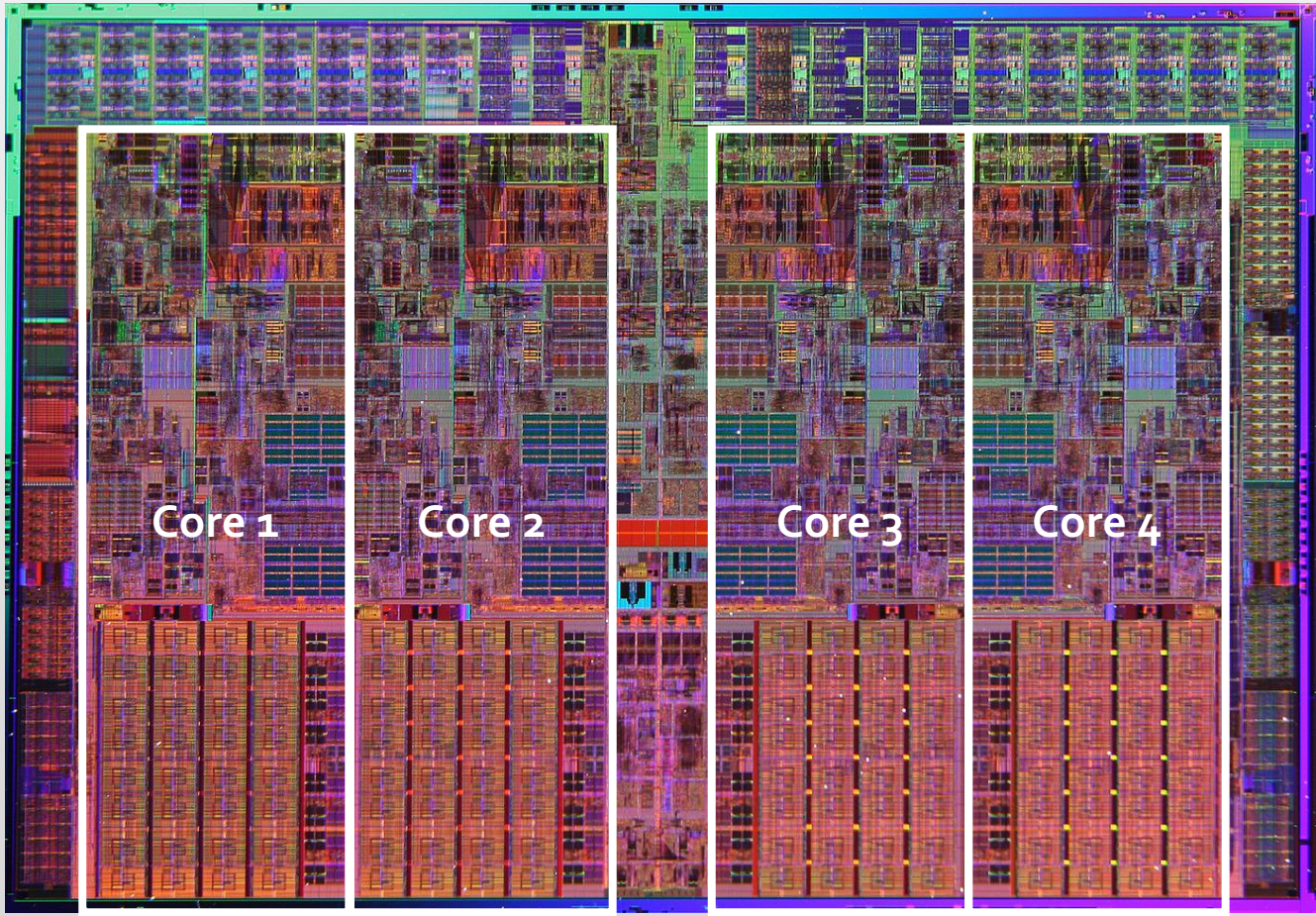


Central Processing Unit

Multi core CPUs are the current standard.

Each core of a multi-core CPU is an item which is almost equivalent to a single-core CPU.
A core contains all the execution units needed for it to execute the whole Instruction Cycle.

It is able to perform own operations independently from the state of the other cores.



Intel Nehalem (2008) quad core die lithography (WikiChip.org)

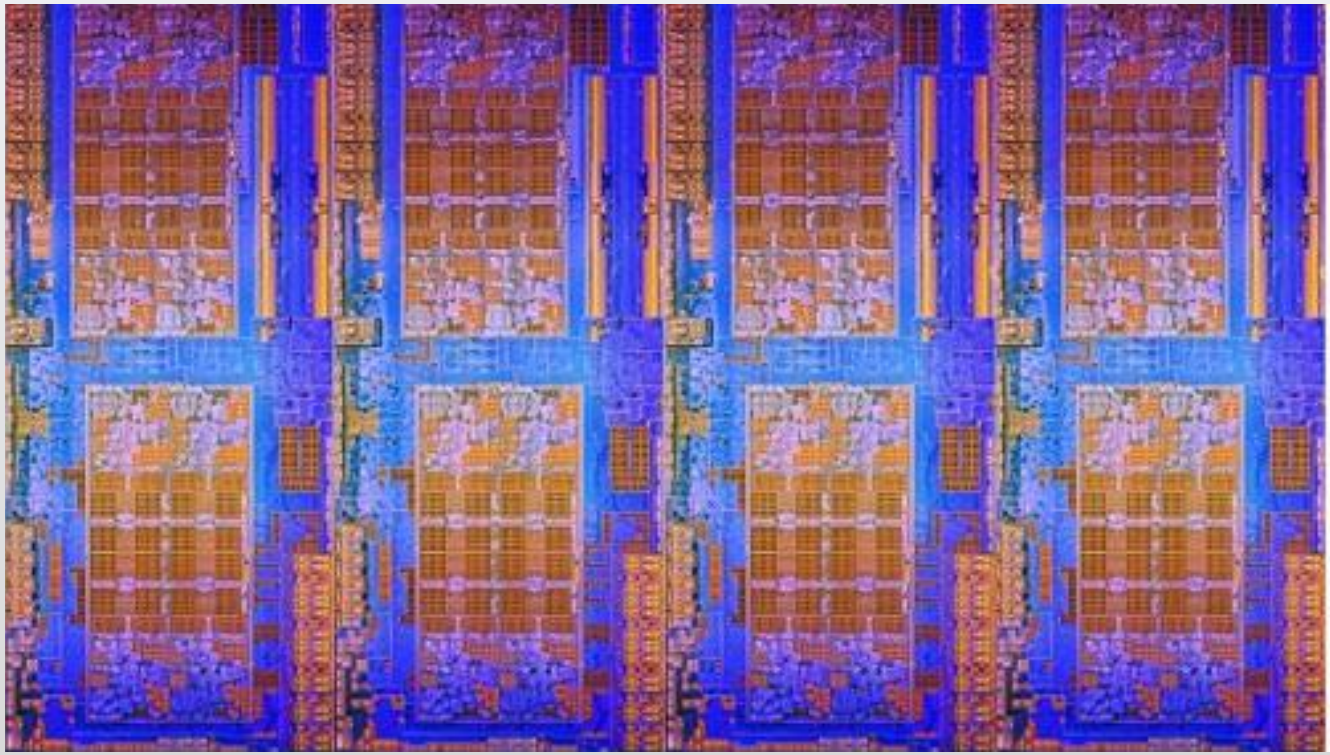
Central Processing Unit

Each core can process its own flow of instructions.

This translates in the possibility to run different software on each core or to run the same software on different data using the core multiplicity to increase performance.

The Single Instruction Multiple Data (SIMD) computing model will be addressed in this course using:

OpenMP
Enabling HPC since 1997



AMD Epyc 1 (2018) 32 cores lithography