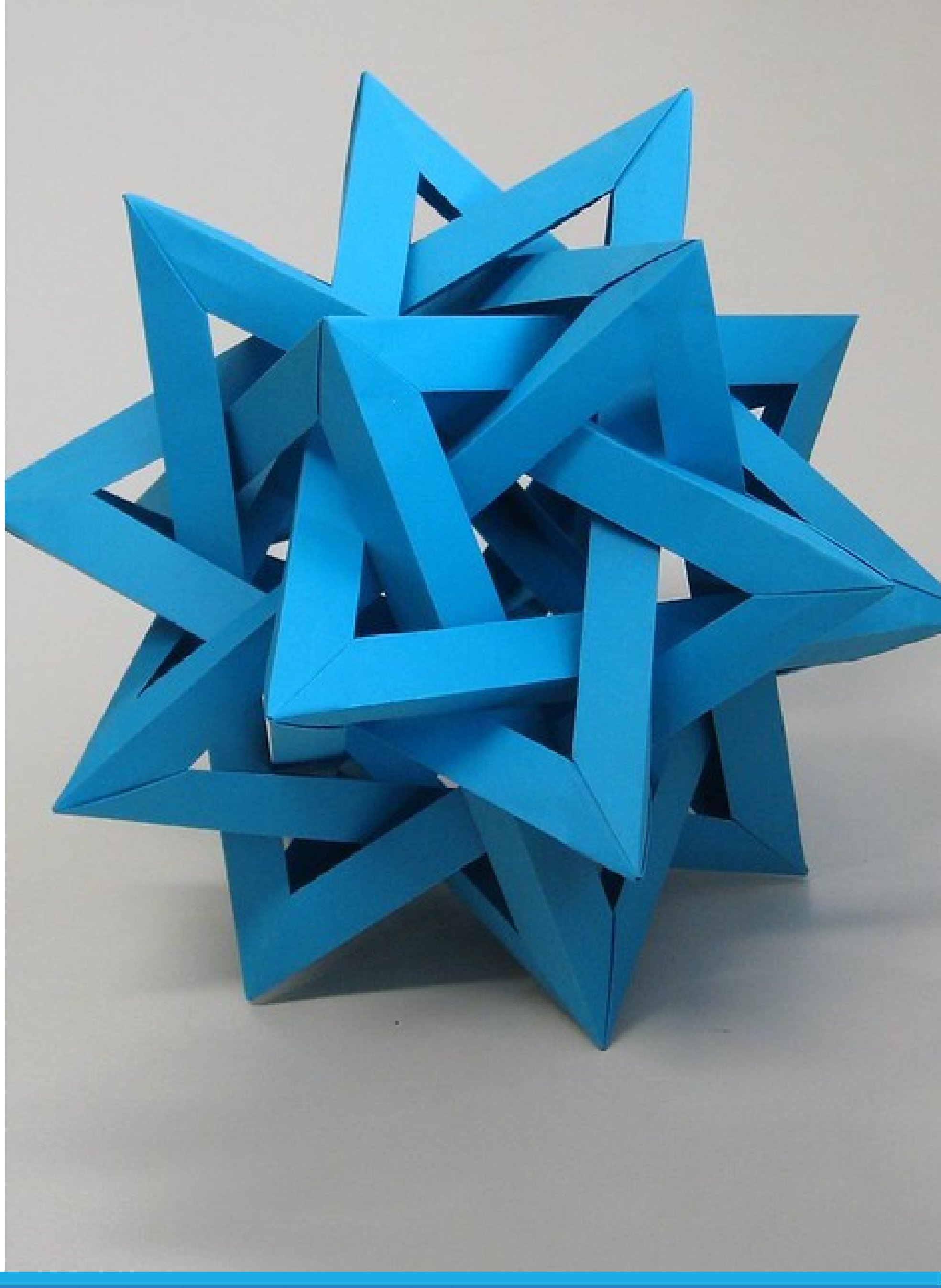
Politecnico di Torino

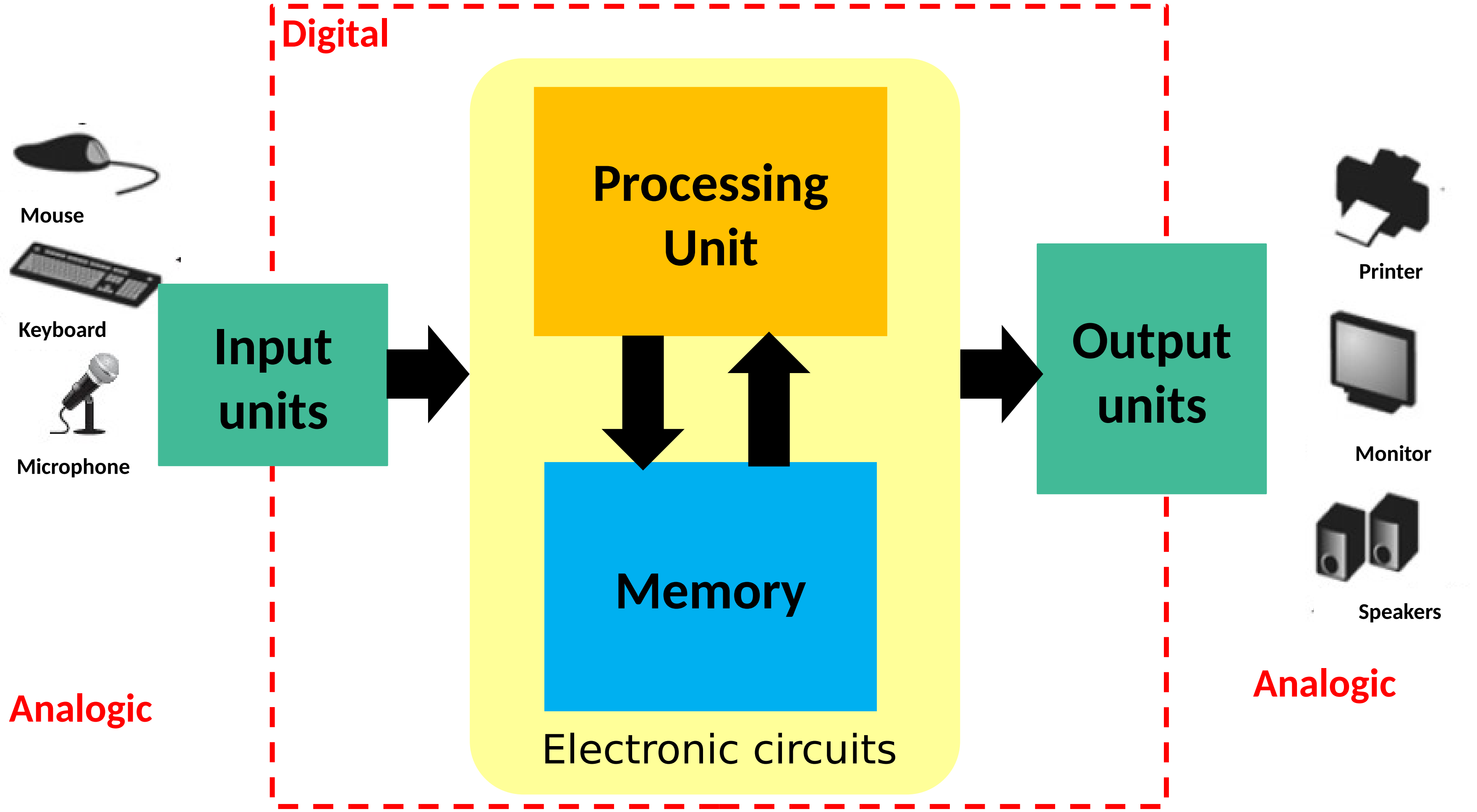
Dipartimento di Automatica e Informatica

Unit T2: Computer Architecture



Computer Architecture

Fundamental blocks of a computer



Fundamental blocks of a computer

- I/O Units
  - Interface from / to the user
  - May require conversions from different domains
    - Human: analogic, non-synchronous, non-electrical
    - Computer: digital, synchronous, electricaltrico
- Processing unit
  - Contains the the circuits to execute “instructions”
  - “Microprocessor”
- Memory
  - Stores data and programs, both permanently (storage) and during the execution of a program (e.g. to save intermediate results)

Microprocessor

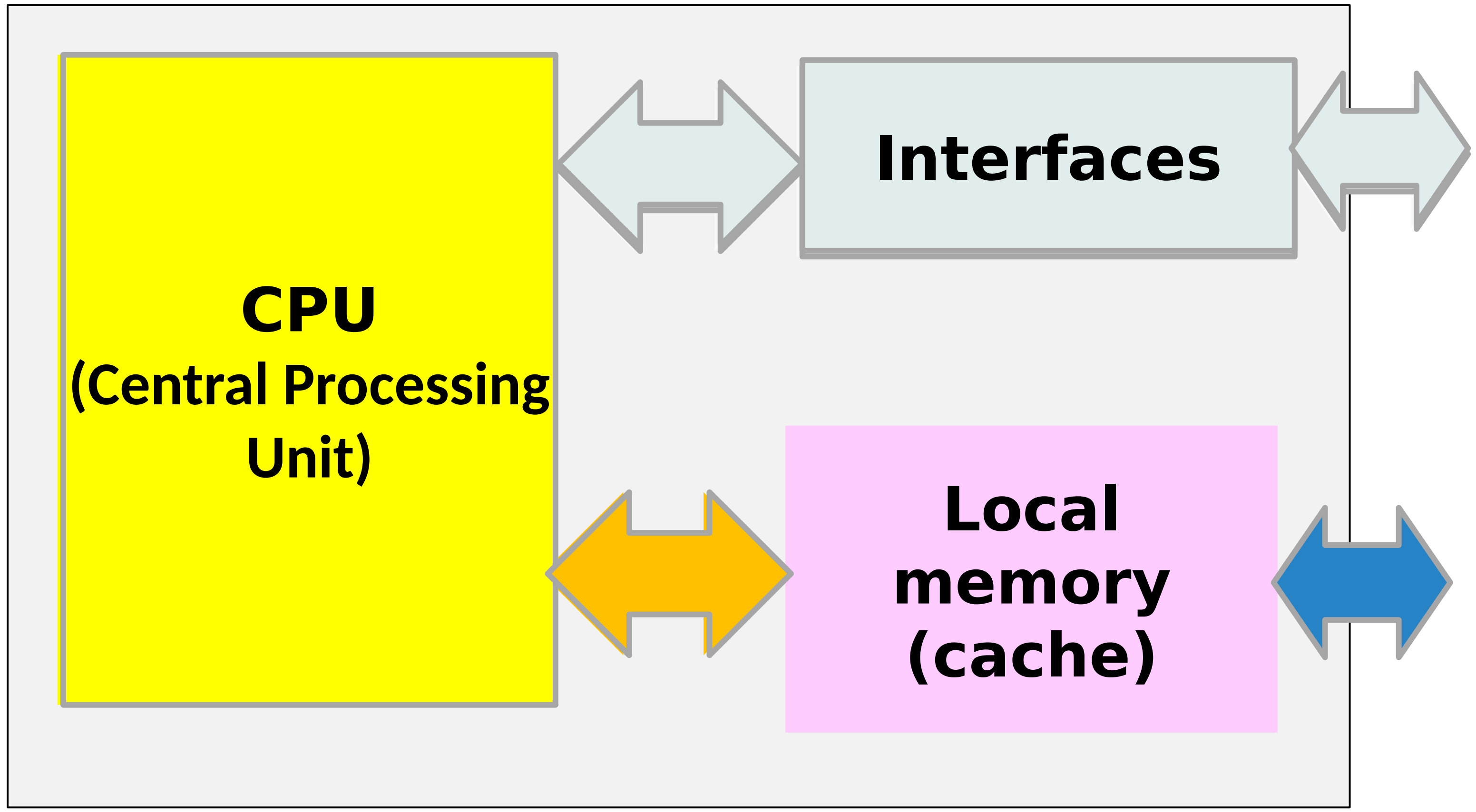
Microprocessor

- The microprocessor is responsible for the execution of all instructions

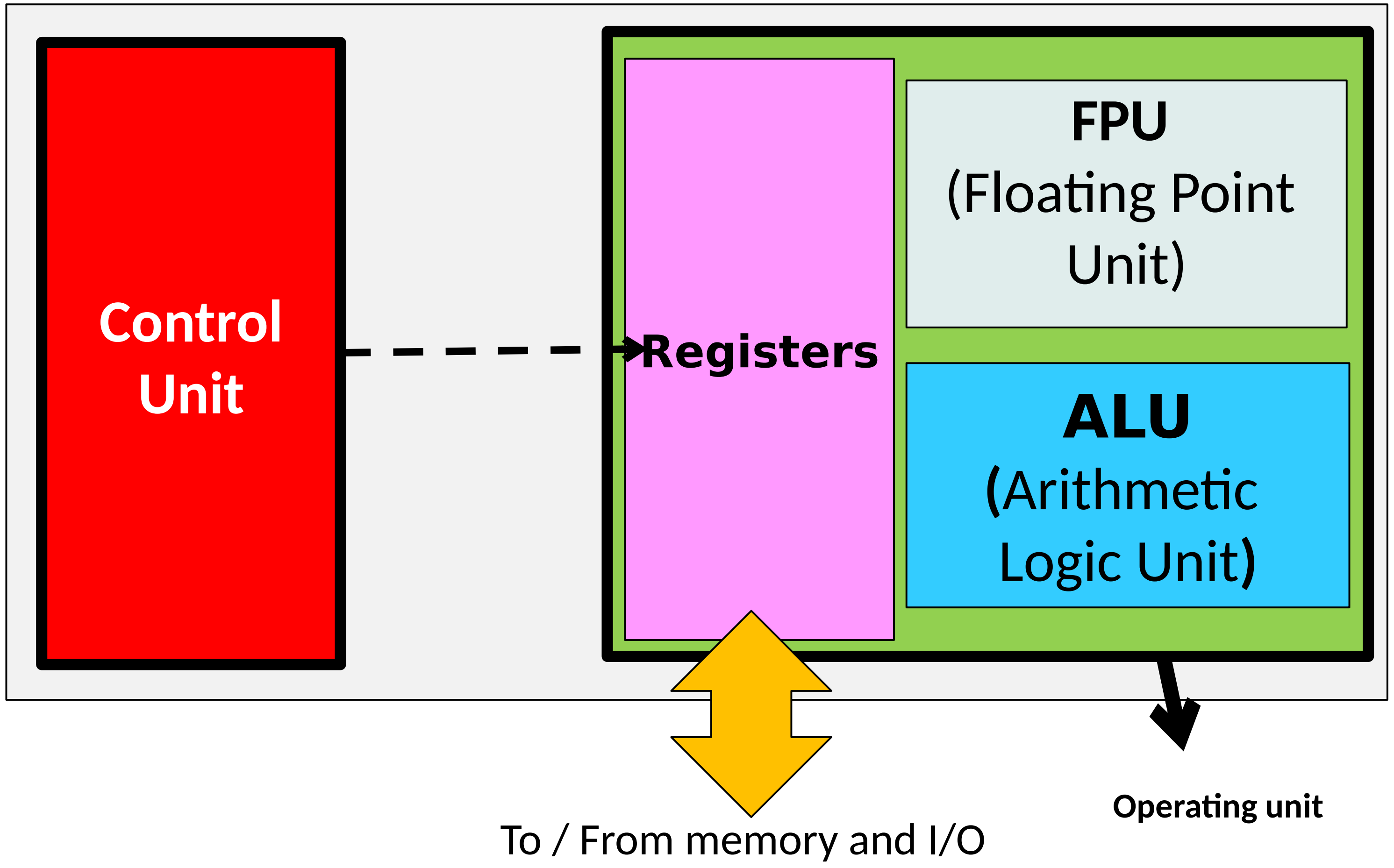
- It contains:
  - Circuits for all basic operations on integer and real (float) numbers, and all logical operations
  - Circuits to control the correct execution of the instructios (e.g. the correct sequence of instructions, possible errors, ...)
  - Interfaces to transfer data to/from memory
  - Interfaces to transfer data to/from I/O units

- Has limited memory capabilities
  - It can store the minimum amount of information required to execute the instructions

Microprocessor

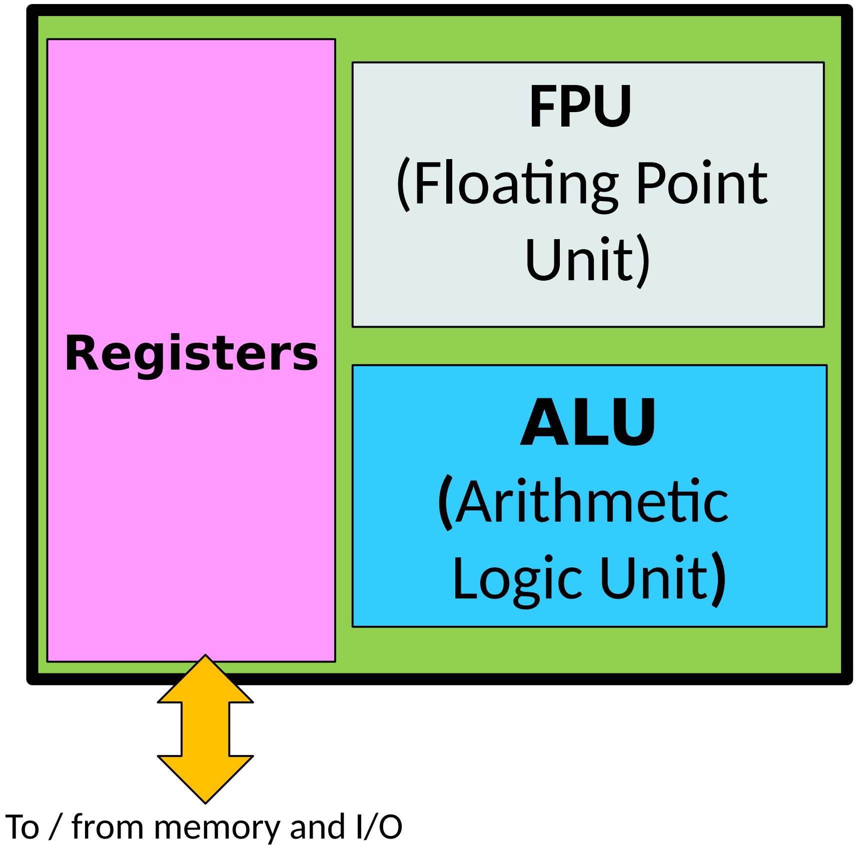


Central Processing Unit



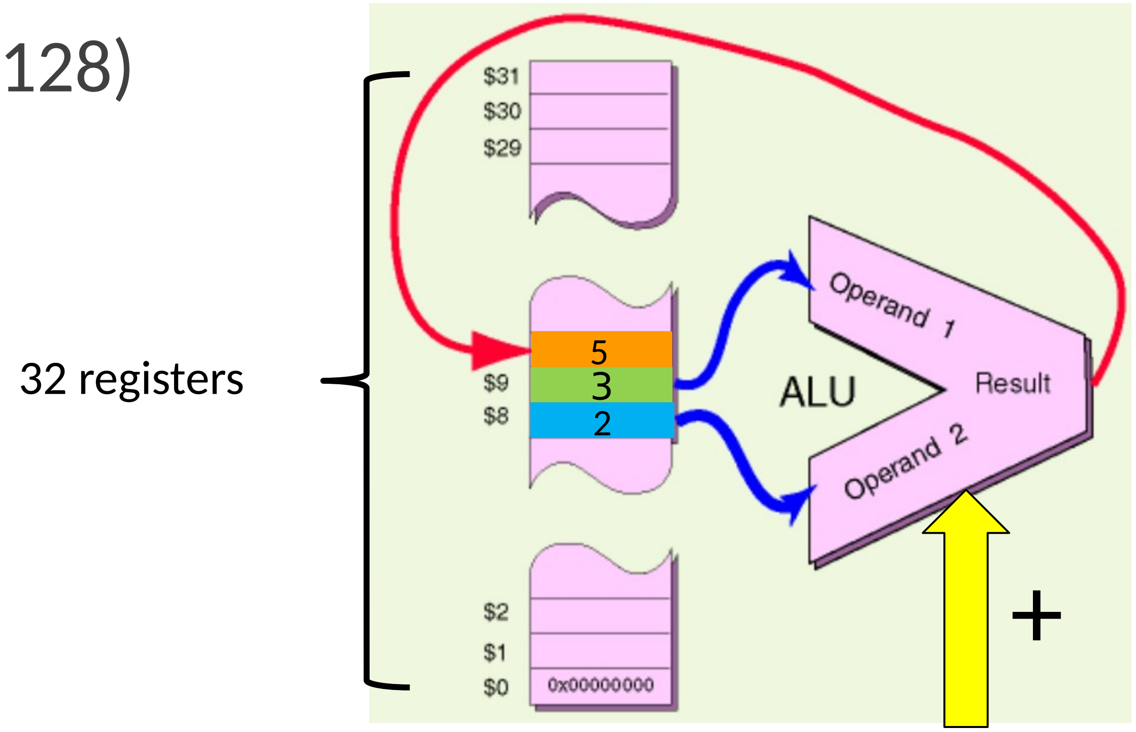
Operating Unit

- Performs the requested computations (arithmmetic, logic, ...)
- It’s composed of:
  - **ALU** (Arithmetic Logic Unit)
  - **Registers** (instruction and data)
  - **FPU** (Floating Point Unit)
  - **Flag register**



Registers

- Local memory used to temporarly store data (e.g. partial results) or instructions
  Transfers between memory and processors involve registers
- Very small number (8…128)



Flag register

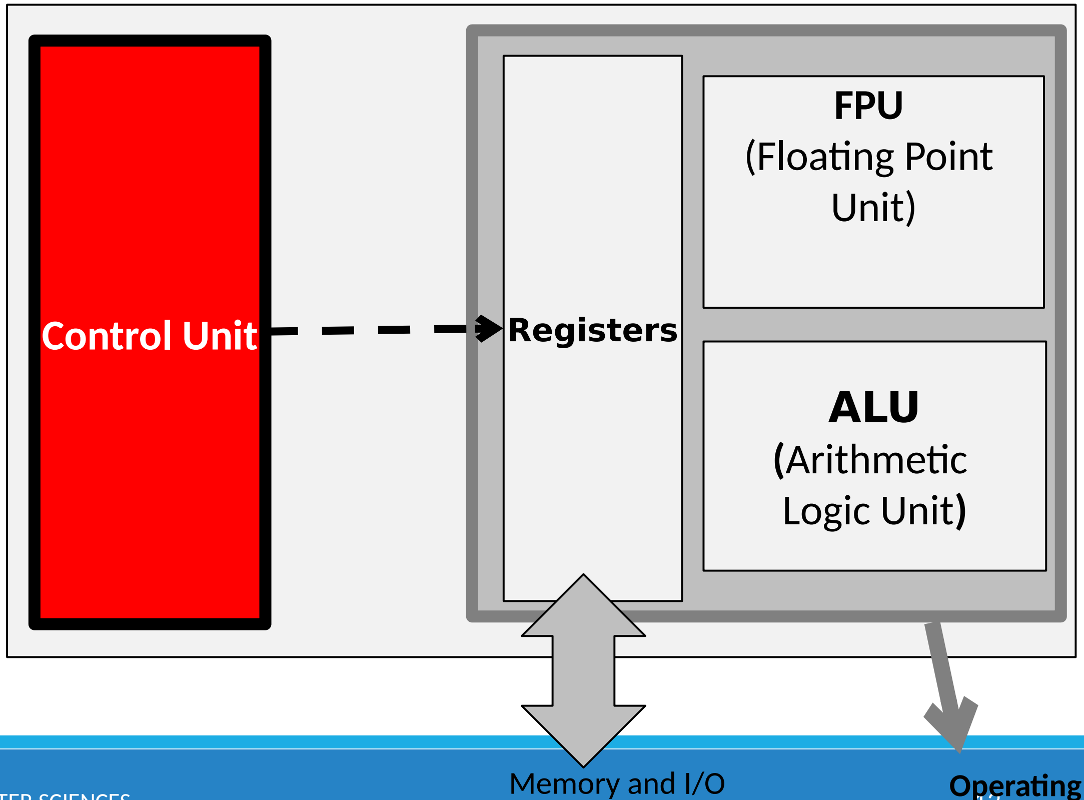
- Special register whose bits are used to represent
  - Special statuses of different computing units
  - Informations on the last executed operation
- They are also used to implement some conditional instructions
- Some examples
  - Zero: Set if the result of the operation was 0.
  - Sign: Used to represent the sign of the last operation
  - Overflow: set to represent that overflow occurred in the previous operation

ALU and FPU

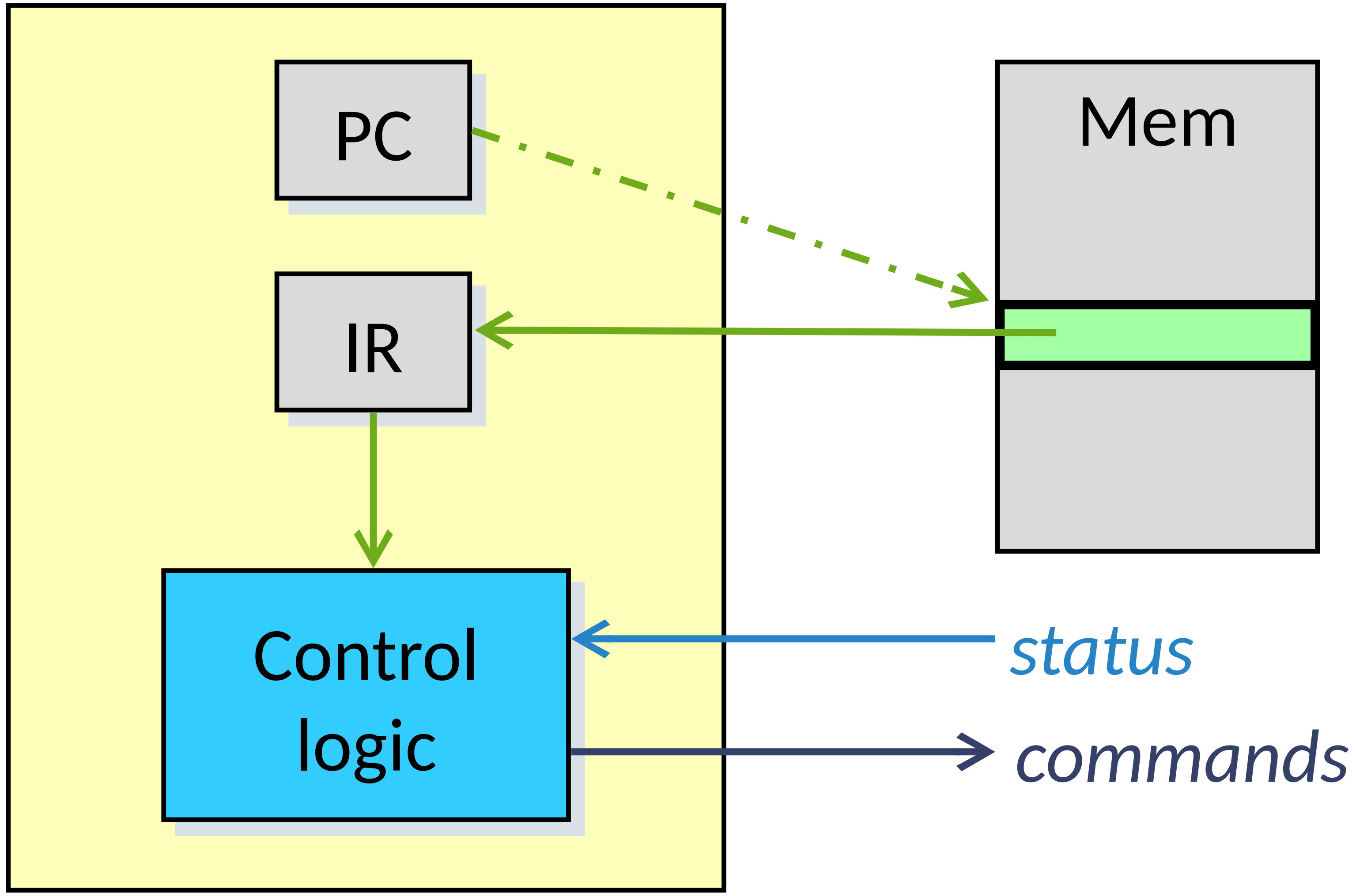
- **ALU**
  - Performs all computations on integer numbers
- **FPU**:
  - Performs all computations on floating point numbers (typically, these require much more time than the integer computations)

Control Unit

- The CU controls the operations that are executed. Based on the instructions of a program and the status of all units, it decides which operation to execute and send the corresponding commands to the different units



Control Unit



Control Unit

- **PC** (Program Counter)
  Register that contains the address in memory of the next instruction to execute

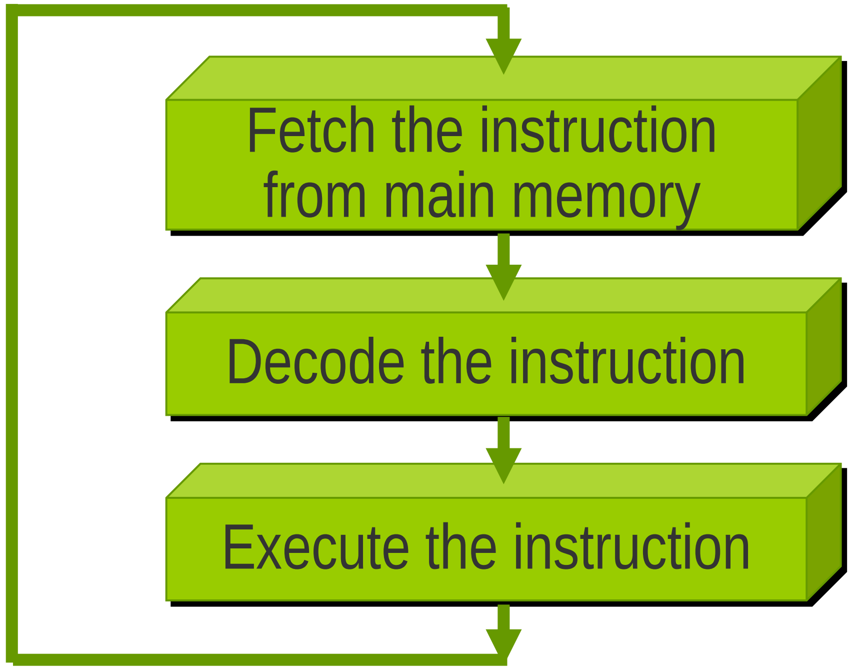
- **IR** (Instruction Register)
  Register that contains the code of the operation to execute

- Control logic
  It interprets the code in the IR to decide which commands to send to the different units

Execution of an instruction

- **Three phases:**

FETCH	**IR ← M [PC]**
	PC ← PC + 1
DECODE	orders ← **decode(IR)**
EXECUTE	activate required blocks



IR ← M [PC]: fetch from memory the intruction from the address contained in the PC

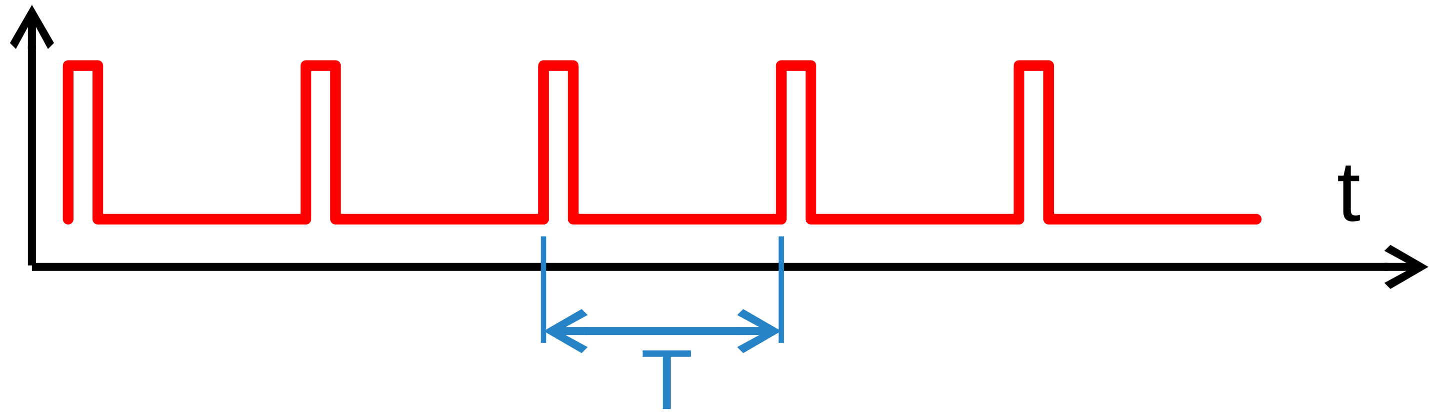
PC ← PC + 1: increase the PC (it will contain the address of the next instruction)

Elementary operations

Category	Operations
Arithmetic operations (ALU)	+, - , * , /, remainder (integers)
Artihmetic operations (FPU)	+, - , * , /, remainder (floats)
Logic operations (ALU)	and, or, xor, not
Comparisons (ALU)	<,>,=, <=, >=, !=
CPU + Memory	Data transfer from/to memory
CPU + I/O unit (+ Memory)	Read/write from/to a device

Clock

- A computer contains a synchronization component (clock) that acts as a temporal reference for all other elements
- T = clock period
  - measured in seconds (s)
- f = clock frequency (= 1/T)
  - measured in Hz (1/s)



Clock and instructions

- A machine cycle is the interval required to execute an elementary operation and is a multiple of the clock period.
- Executing an instruction requires a number of machine cycles that depend on the instruction
  - For example

Instruction type	Cycles
ALU	1
FPU	15
MEM	5
I/O	100

Memory

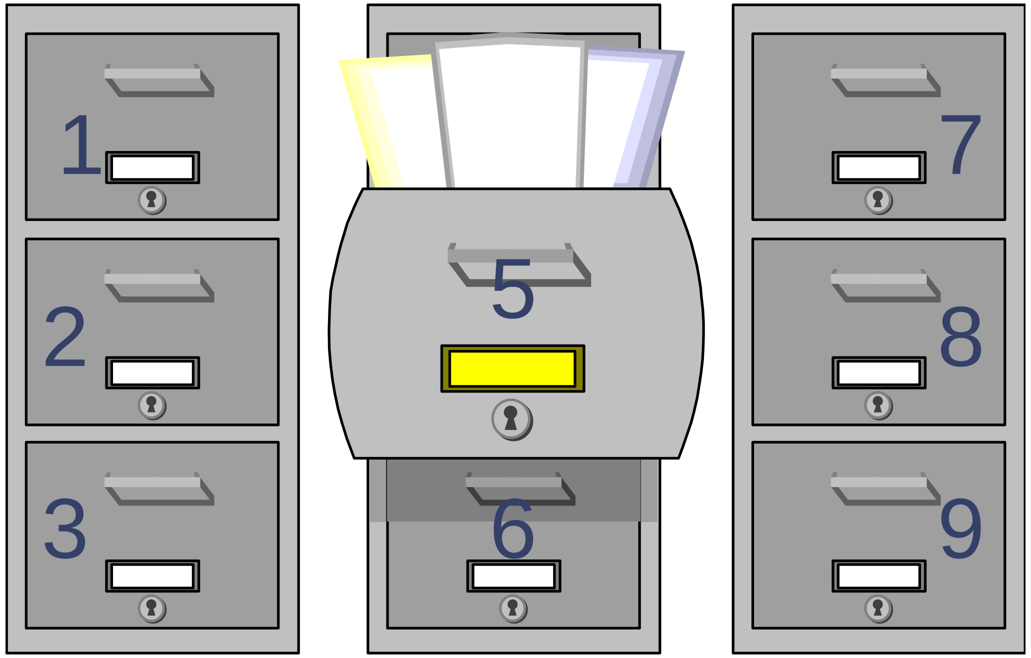
Memory

- Contains data and instructions
  - To store data and programs (mass memory)
  - For efficiency reasons

- Characteristics:
  - Addressing
  - Parallelism
  - Sequential / random access
  - Hierarchy

Addressing

- Memory is organized in cells (smalles addressable unit). Each cell has an associated unique address.



Parallelism

- Each cell contains a fixed quantity of bits (word):
  - Same for all cells (of a given memory unit)
  - Accessed by a single instruction
  - Multiple of a byte

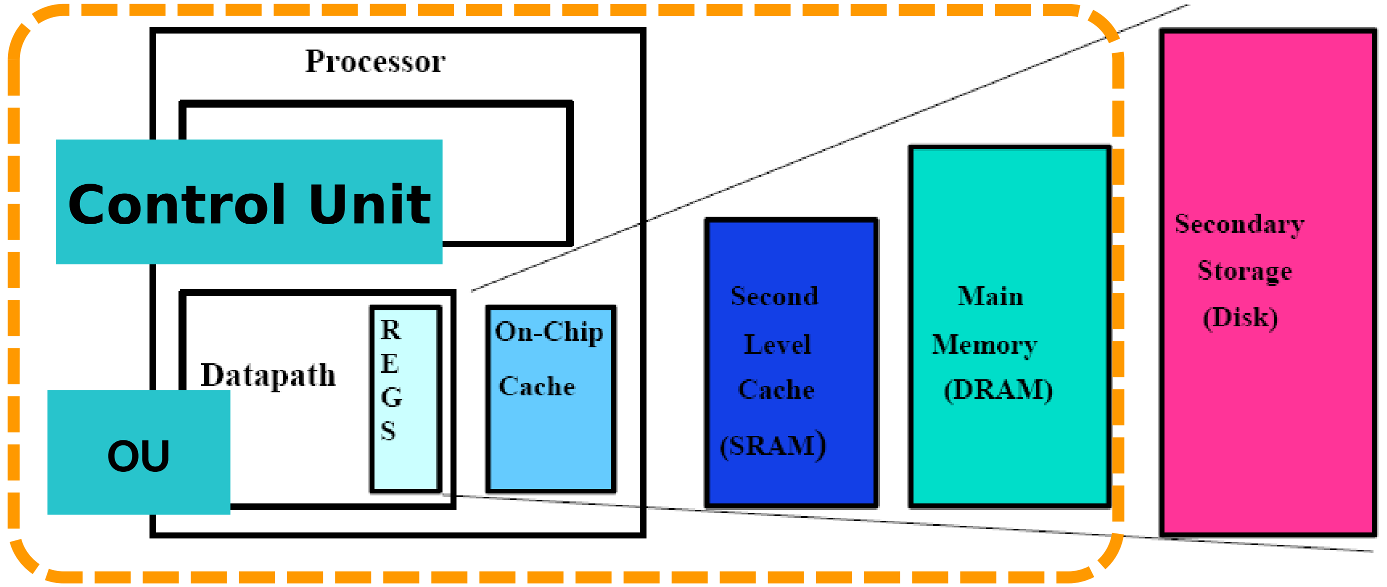
Memory hierarchy

- Ideally, memory should be
  - Large
  - Cheap
  - Fast
  - Non volatile

- The ideal memory does not exist
  - Fast memories are expensive and volatile
  - Non volatile memories can be much cheaper but are much slower
- Solution: memory hierarchy

Memory hierarchy

- Memory organized in levels
  - Faster and expensive (and volatile), but low capacity memories closer to the CPU
  - Cheaper but slower memories more distant from CPU



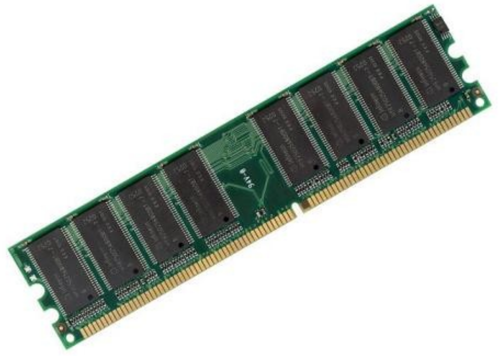
Memory hierarchy

- Registers
  - Fastest memory, inside the processor
  - Temporary storage for data and instructions
  - Very few

- Cache
  - Fast, small memory between CPU and main memory
  - Used to speed up execution: it stores copies of those data which have a high probability of being needed in the close future, to avoid accessing the (slower) main memory
  - If data is not foundin the cache, it’s loaded from the main memory (or a second, larger but slower cache)

Memory hierarchy

- Main memory: RAM (Random Access Memory)
  - Random access
    - Each cell can be accessed in the same time span, regardless of its position
    - Allows both reading (requires providing the address) and writing (requires providing the address and the datum to store)
  - Volatile
    - Data in RAM is not preserved when the computer is switched off
    - The RAM is used to temporarily store the instructions and data of a program



Memory hierarchy

- Main memory: RAM (Random Access Memory)
  - Two types of RAM:
    - SRAM: Static Random Access Memory

      Can contain information as long as power is provided

      Faster but more expensive than DRAM

      Used for cache memory
    - DRAM: Dynamic Random Access Memory
    - Slower but cheaper
    - Requires costant refresh to avoid losing data

Memory hierarchy

- Mass memory
  - Main memory is expensive and voltaile, so it’s not suited to store large amount of data indefinitely
  - Mass memory is slower, but also significantly cheaper and larger, and non-volatile
  - It’s used to store programs and all data that can be used by the computer
    - The processor cannot used directly mass memory to perform computations
    - The execution of a program requires that the program is copied from mass memory to main memory (loading)

Memory hierarchy

- Flash
  - USB drives, SD cards, SSDs are based on electronic devices but, contrary to RAM, are non volatile

- Hard disks are mechanical drives that employ magnetic materials to store data
  - Reading and writing involves rotating a disk
  - Being mechanical, HDDs are much slower than flash drives, but are also significantly cheaper

Memory comparison

Faster
More expensive

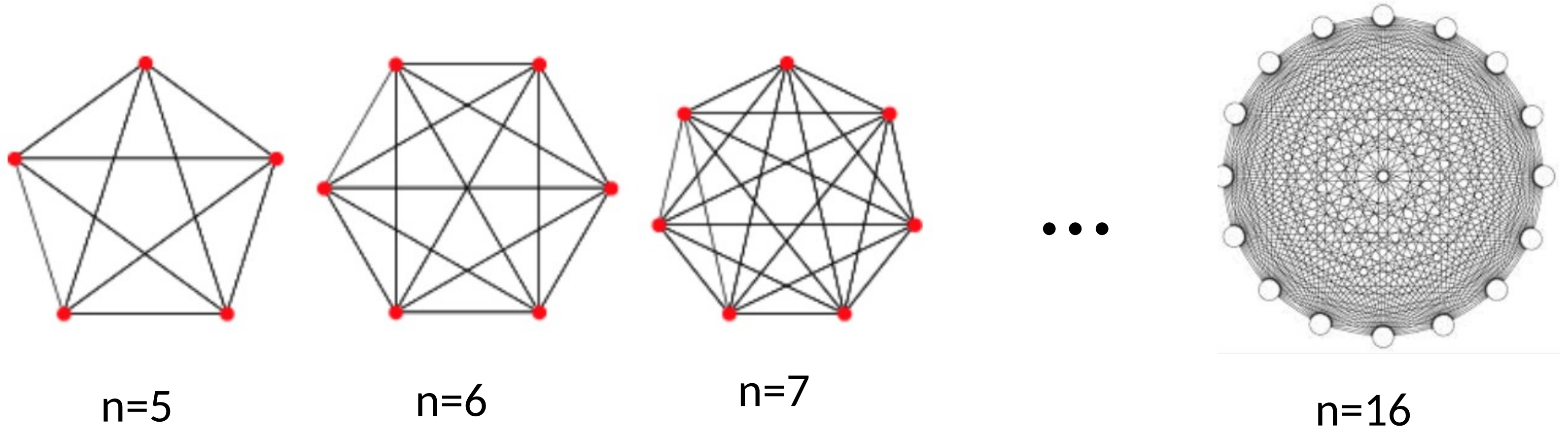
	SRAM (cache)	DRAM	SSD	HDD
Capacity	MB (1-32+)	GB (4-128+)	TB (0.5-4)	TB (1-20)
Speed	around 1TB/s	around 10GB/s	up to 7GB/s	100MB / s
Latency	10ns	50-150 ns	~0.1ms	10ms
Cost	10^{-5} \$/byte	10^{-9} \$/byte	10^{-10} \$/byte	10^{-11} \$/byte
Persistence	Volatile	Volatile	Non volatile	Non volatile

Larger capacity
Slower

Bus

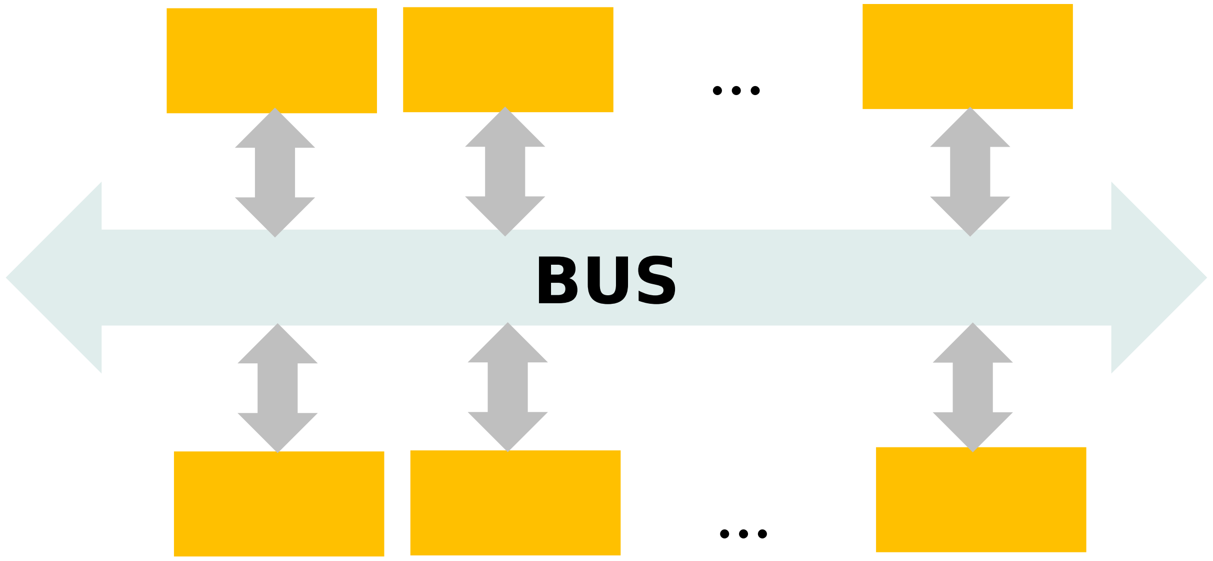
Bus

- How to connect different devices (memories, I/O devices, …)?
- Point to point connections are not practical
  - The number of requried connection grows quadratically



Bus

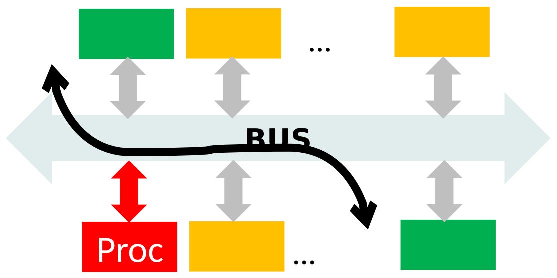
- We use a single line to connect different components



Bus

- Pros:
  - Lower production costs
  - Scalability (easy to add a new device)
  - Easier to standardize
    - Easier to define a single set of rules for all devices

- Cons:
  - Slower
    - Only one device can access the bus at a time
  - The processor must control all bus operations

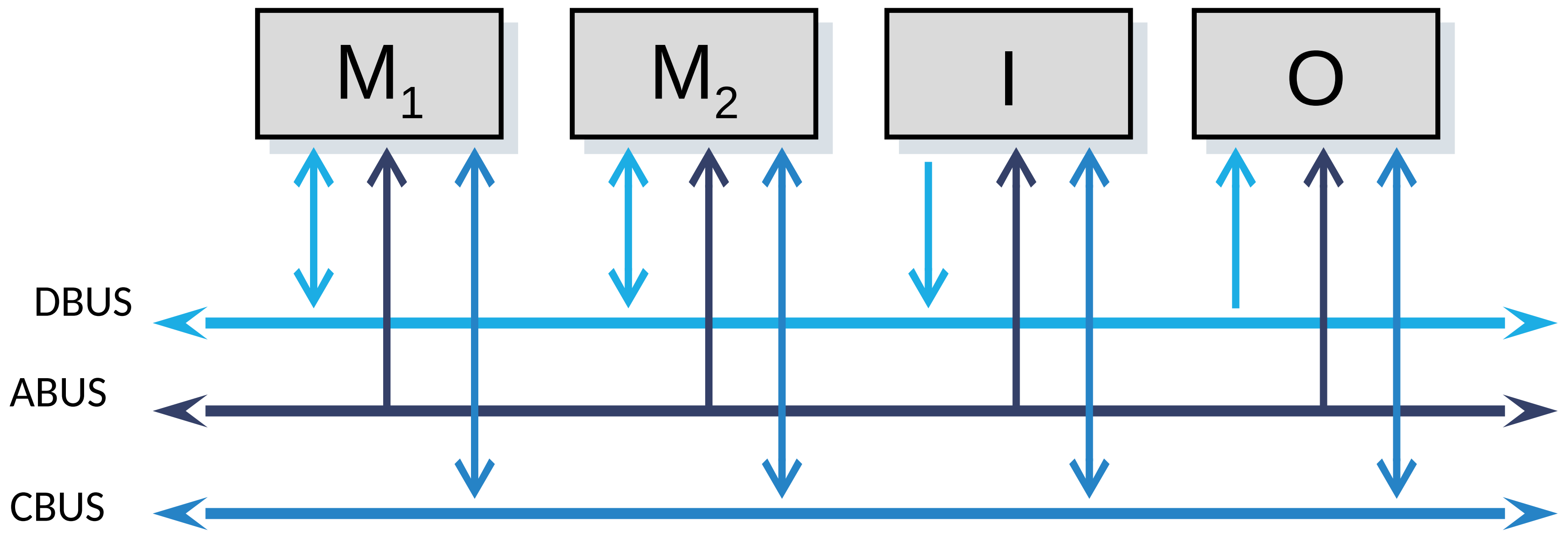


Bus

- Can transfer a single datum at a time
- Frequency: number of data trasfers in each second
- Width: number of bits of a single datum
- If the bus is too small, it can become a bottleneck

Bus

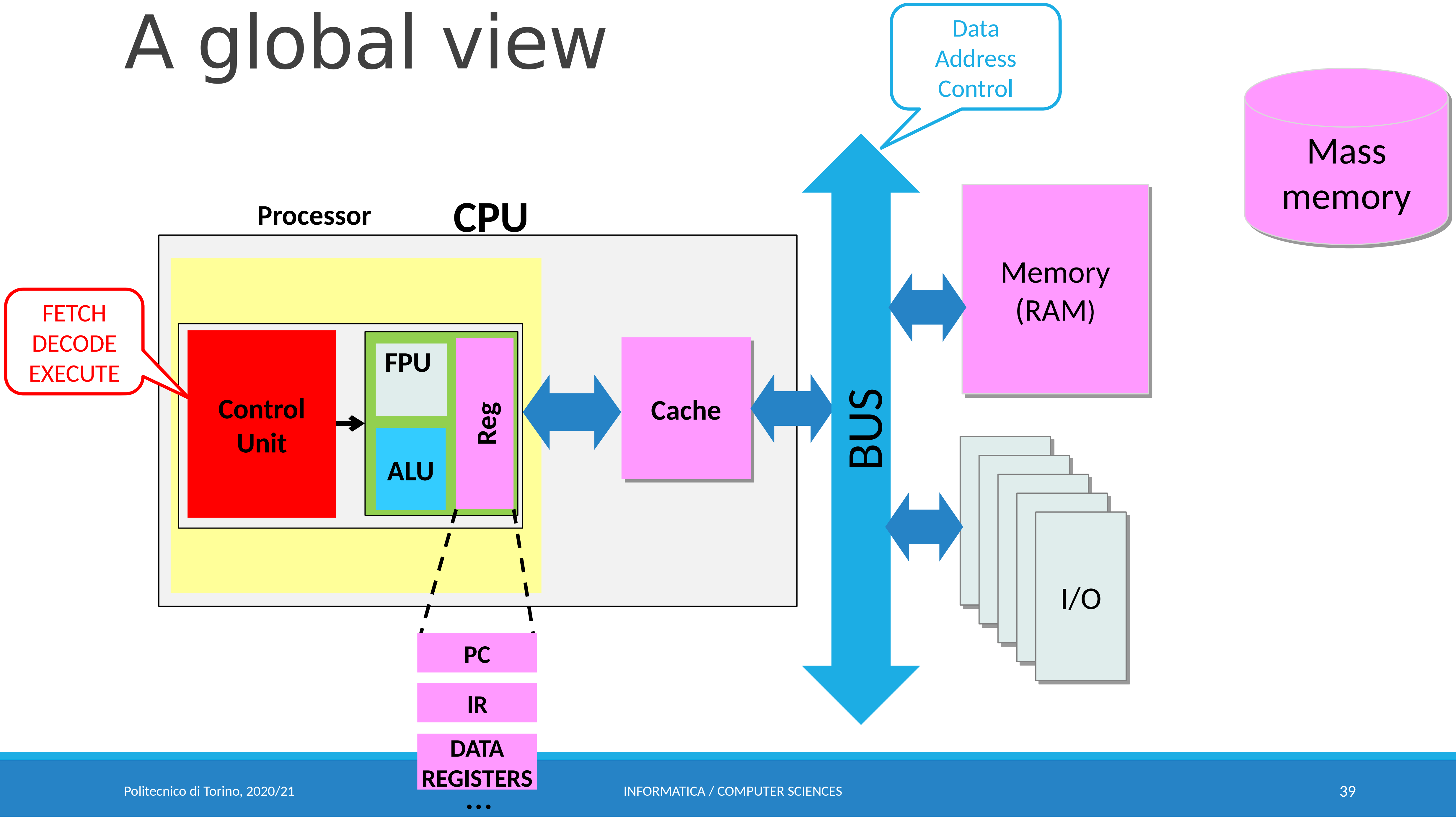
- A bus is divided in three sub-components:
  - Data bus (DBus)
  - Address bus (ABus)
  - Control bus (CBus)



Bus

- The ABus transport the address information. It determines the maximum number of addressable cells
- The DBus transfer data. Its width is related to the dimension of a memory cell
- The maximum size of a memory that can be accessd by a bus is
- **$2^{|Abus|}$ x |Dbus| bit**
- Examble: 20 bit Abus, 16 bit Dbus
  - max mem = 2^{20} x 2 byte = 2 MB

A global view


Data
Address
Control
Mass memory
Processor
CPU
Memory (RAM)
FETCH
DECODE
EXECUTE
Control Unit
FPU
ALU
Reg
Cache
BUS
I/O
PC
IR
DATA REGISTERS
…