Advanced Computer Architectures
(High Performance Processors and Systems)

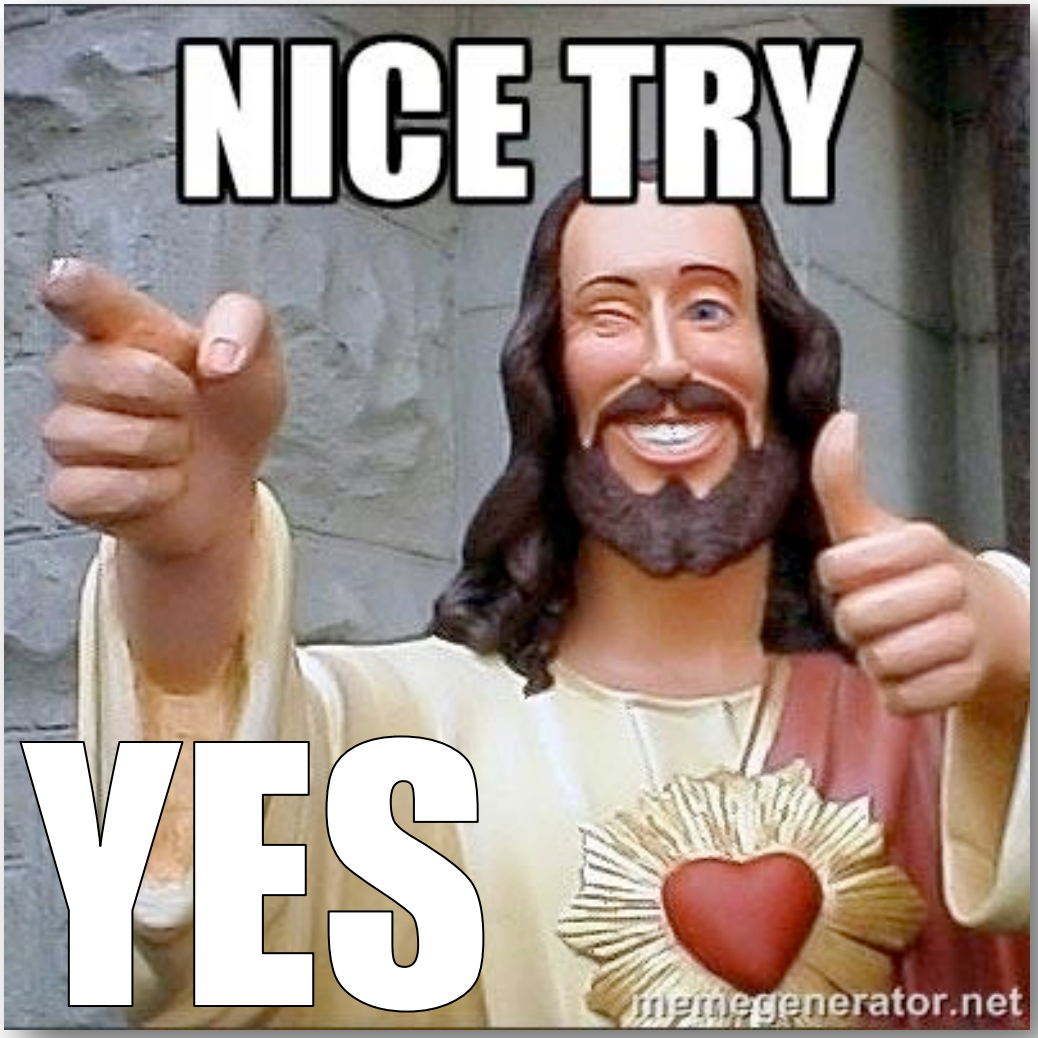
Computing Systems
… and other interesting things ;)

Politecnico di Milano

v1

Alessandro Verosimile <Alessandro.verosimile@polimi.it>
Marco D. Santambrogio <marco.santambrogio@polimi.it>

ACA through other possibilities?



ACA Project Rules

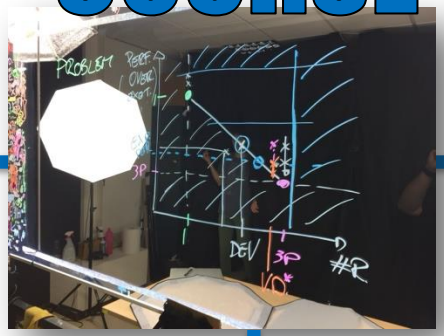
- Project presented on Monday 12.5.2025
  - Where: ACA Class (T.1.1)
  - When: @ 2pm
  - What: Two tracks will be presented during the 12.5.25 class

- Teams and projects
  - Teams up 2 members
  - Each team has to pick/select one of the two tracks (share the decision via email to the instructors) – Deadline: 18.5.2025
  - Starting: May 12, 2025 - Deadline: June 30, 2025
  - Each project will be presented to Prof. Santambrogio
    - The date will be discussed/agreed with him
    - It's an ACA **project**, **questions** wrt to all the ACA topics will be asked during the project presentation!

ACA Exams

ACA @17.2

COURSE



EXAMS A1, A2

MID-EVALUATIONS

@MAY

ACA Prj

@ 30.6

PROJECTS
7.5 - 30.6

PRJ SUBMISSION

HPPS Project Rules

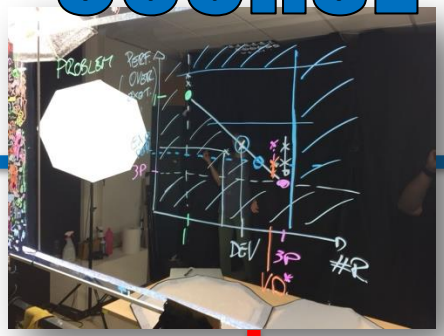
- Project presented on Friday 21.2.2025
  - Where: NECSTLab Meeting Room
  - When: @ 9am

- Teams and projects
  - No team (aka 1 project = 1 student)
  - The mapping will be done during the meeting @21.2.2025
  - Starting: Feb 21, 2025 - Deadline: June 30, 2025
  - Each project will be presented to Prof. Santambrogio
    - The date will be discussed/agreed with him
    - It's an HPPS **project**, **questions** wrt to all the HPPS topics will be asked during the project presentation!

ACA Exams

ACA @17.2

COURSE



EXAMS A1, A2

HPPS

PROJECTS 21.2- 30.6

@MAY

MID-EVALUATIONS

@ 30.6

PRJ SUBMISSION

How To ACA

Let us know how you'll complete the ACA course

https://tinyurl.com/HowToACA2025

Deadline March 7, 2025

MAIL POLICY...

MAIL POLICY...

... ALWAYS IN CC

Alessandro Verosimile

alessandro.verosimile@polimi.it

MAIL POLICY...

... OVER THE WEEKEND


MAIL POLICY…
I'M WITH THEM!!!!
FRI@6PM – MON@5AM


AND NOW...
https://flic.kr/p/8puKvo

VIDEOS: C1, C2

https://tinyurl.com/video-ACA-polimi

Different Classes of Computers…

… Different needs

Different Classes of Computers… … Different needs

- Personal Mobile Device (PMD)
  - Emphasis on energy efficiency and real-time

Different Classes of Computers… … Different needs

- Personal Mobile Device (PMD)
  - Emphasis on energy efficiency and real-time
- Desktop Computing
  - Emphasis on price-performance

Different Classes of Computers… … Different needs

- Personal Mobile Device (PMD)
  - Emphasis on energy efficiency and real-time
- Desktop Computing
  - Emphasis on price-performance
- Servers
  - Emphasis on availability, scalability, throughput

Different Classes of Computers… … Different needs

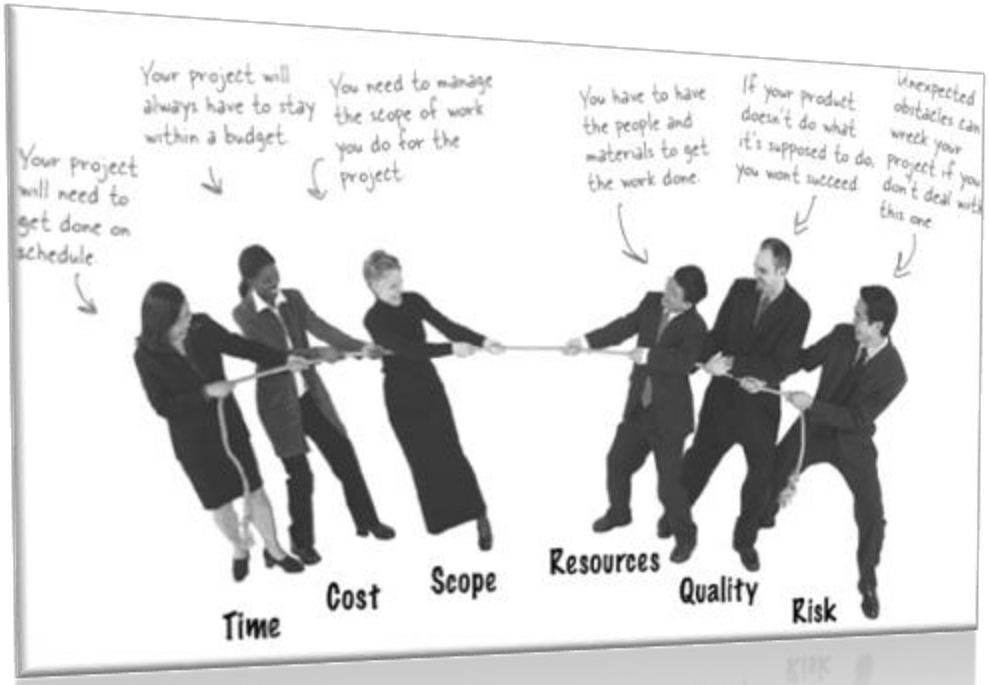
- Personal Mobile Device (PMD)
  - Emphasis on energy efficiency and real-time
- Desktop Computing
  - Emphasis on price-performance
- Servers
  - Emphasis on availability, scalability, throughput
- Clusters / Warehouse Scale Computers - Used for “SaaS”
  - Emphasis on availability and price-performance
  - Sub-class: Supercomputers, emphasis: floating-point performance and fast internal networks

Different Classes of Computers… … Different needs

- Personal Mobile Device (PMD)
  - Emphasis on energy efficiency and real-time
- Desktop Computing
  - Emphasis on price-performance
- Servers
  - Emphasis on availability, scalability, throughput
- Clusters / Warehouse Scale Computers - Used for “SaaS”
  - Emphasis on availability and price-performance
  - Sub-class: Supercomputers, emphasis: floating-point performance and fast internal networks
- Embedded Computers
  - Emphasis: price

Issues as new opportunities

- Programming has become very difficult
  - Impossible to balance all constraints manually



Issues as new opportunities

- Programming has become very difficult
  - Impossible to balance all constraints manually
- More computational horse-power than ever before
  - Cores are free

Overview of Factors Affecting Performance

- Algorithm complexity and data set
- Compiler
- Instruction set
- Available operations
- Operating system
- Clock rate
- Memory system performance
- I/O system performance and overhead

ONLINE CLASSES

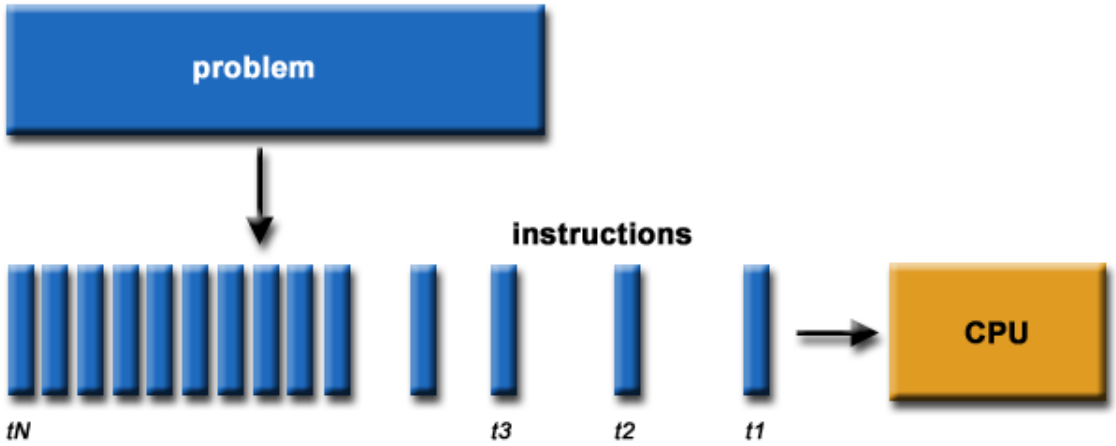
https://tinyurl.com/video-ACA-polimi

MORE ON PERFORMANCE/COSTS

VIDEOS: C1

"Traditional" Computation

- Software is written for serial computation
  - It has to be executed on a single computer having a single Central Processing Unit (CPU)
  - A problem is broken into a discrete series of instructions
    - Instructions are executed one after another
  - Only one instruction may execute at any moment in time



The Program…

…

```
k=b+c+d;
```

…

Breaking down performance

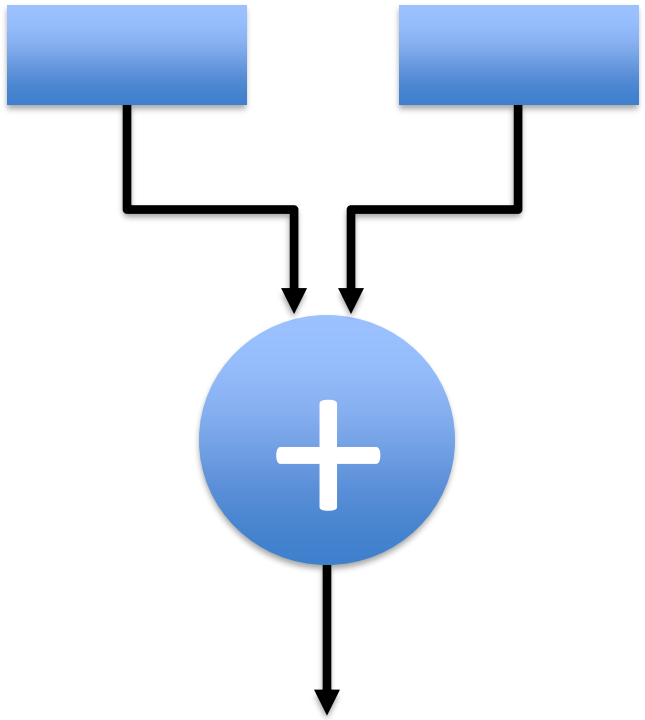
- A program is broken into instructions
  - Hardware is aware of instructions not programs
- At lower level hardware breaks instructions into clock cycles
  - Lower level state machines change state every cycle

The Program and the ISA

…

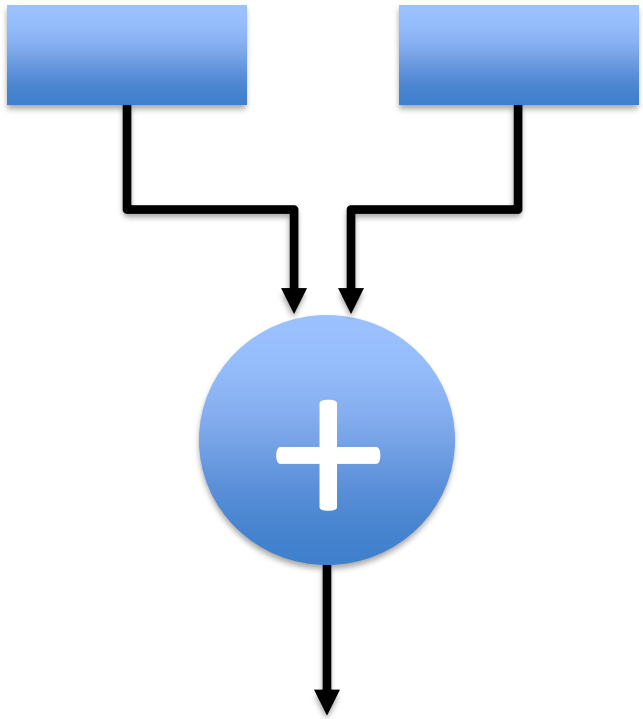
k=b+c+d;

…



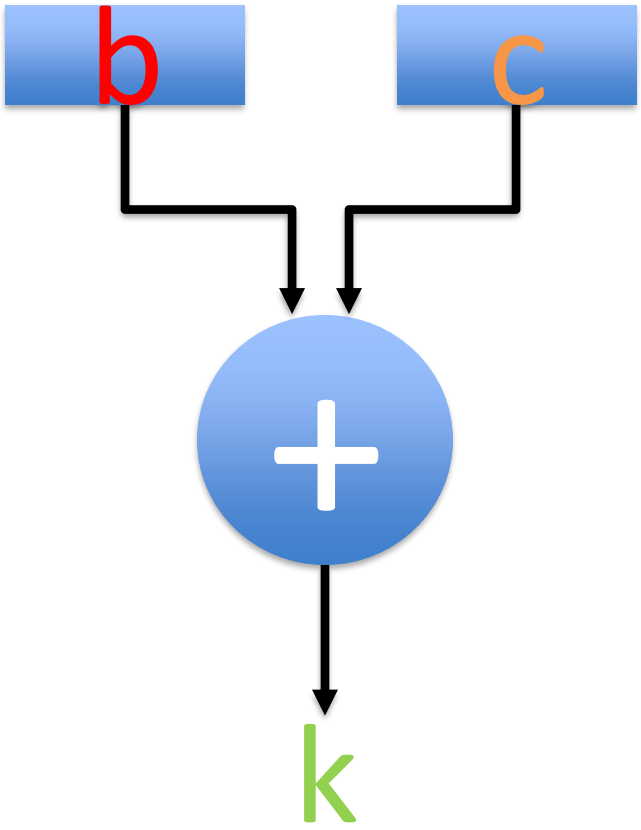
The Program and the ISA

```
…
k=b+c;
k=k+d;
…
```



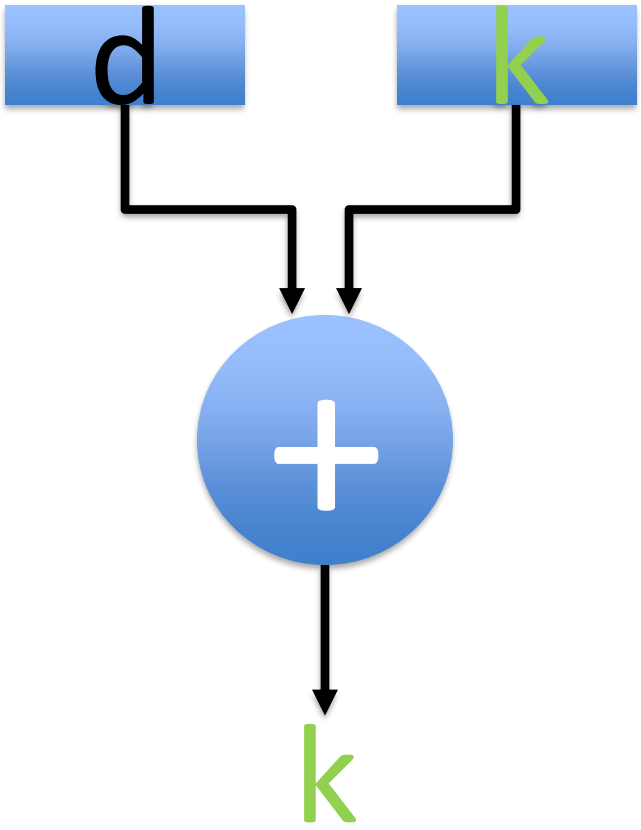
The Program and the ISA

```
…
k=b+c;
k=k+d;
…
```



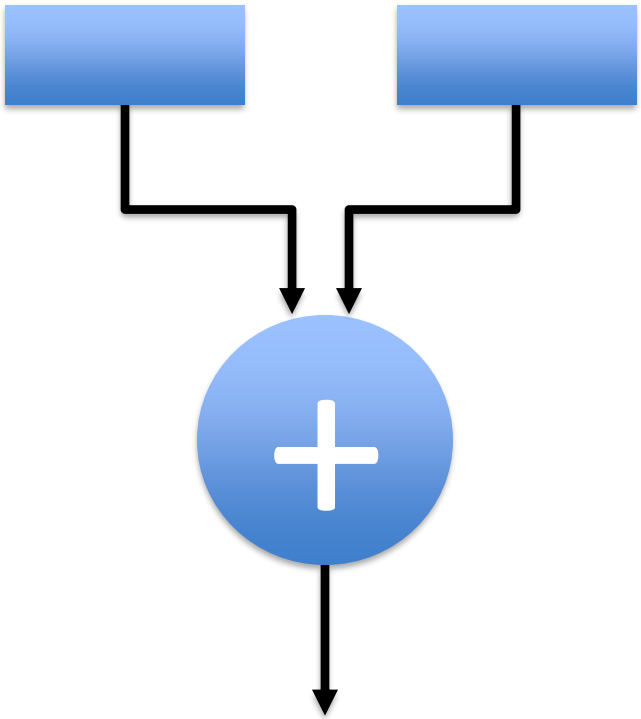
The Program and the ISA

```
…
k=b+c;
k=k+d;
…
```



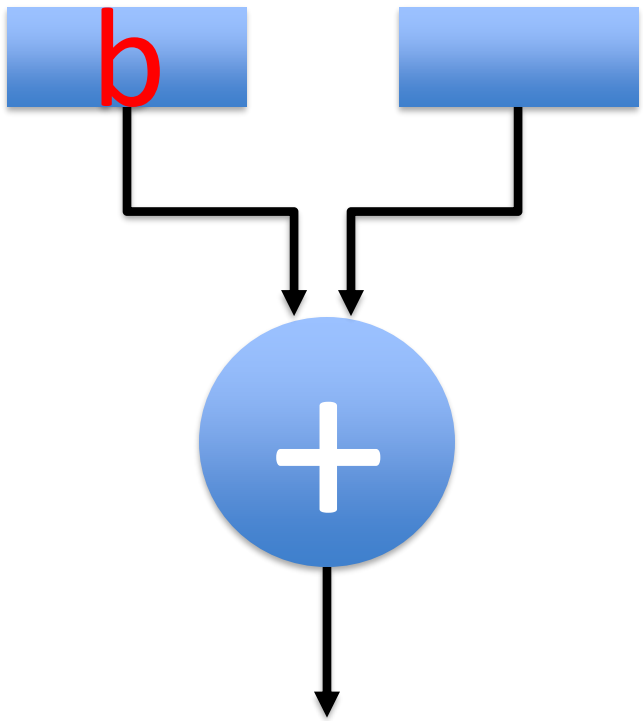
The Instructions and the ISA

… …	… … …	
0789	load	R02,4000
0790	load	R03,4004
0791	add	R01,R02,R03
0792	load	R02,4008
0793	add	R01,R01,R02
0794	store	R01,4000
… …	… … …	



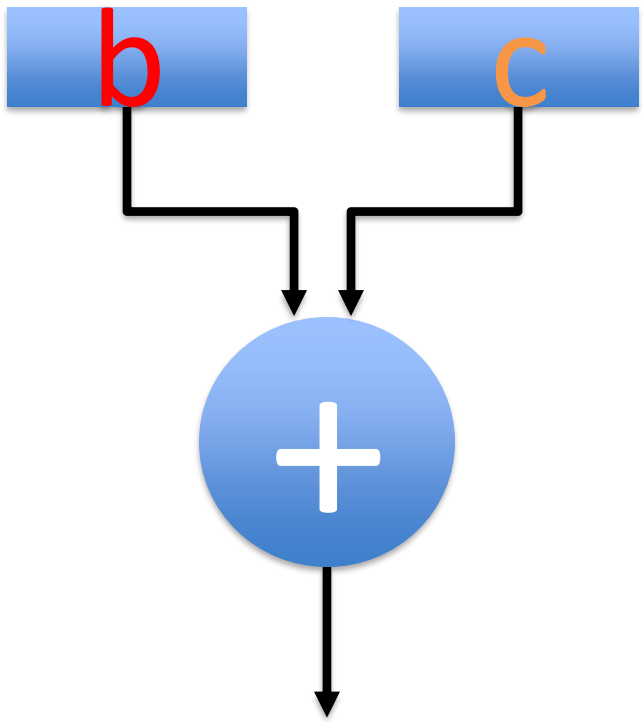
The Instructions and the ISA

```
… …   … … …
0789  load  R02,4000
0790  load  R03,4004
0791  add   R01,R02,R03
0792  load  R02,4008
0793  add   R01,R01,R02
0794  store R01,4000
… …   … … …
```



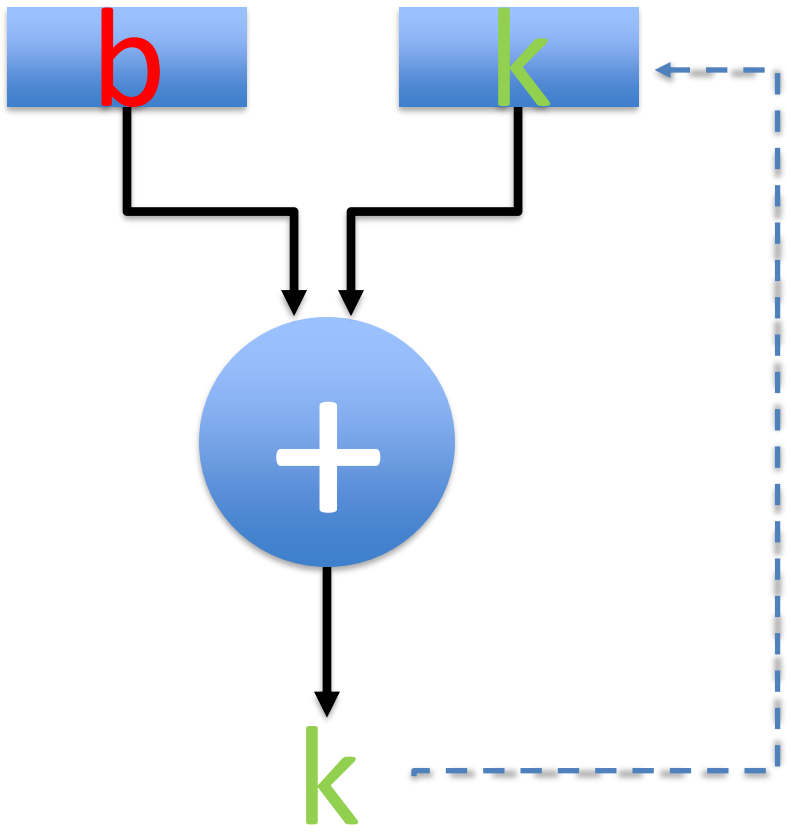
The Instructions and the ISA

… …	… … …	
0789	load	R02,4000
0790	load	R03,4004
0791	add	R01,R02,R03
0792	load	R02,4008
0793	add	R01,R01,R02
0794	store	R01,4000
… …	… … …	



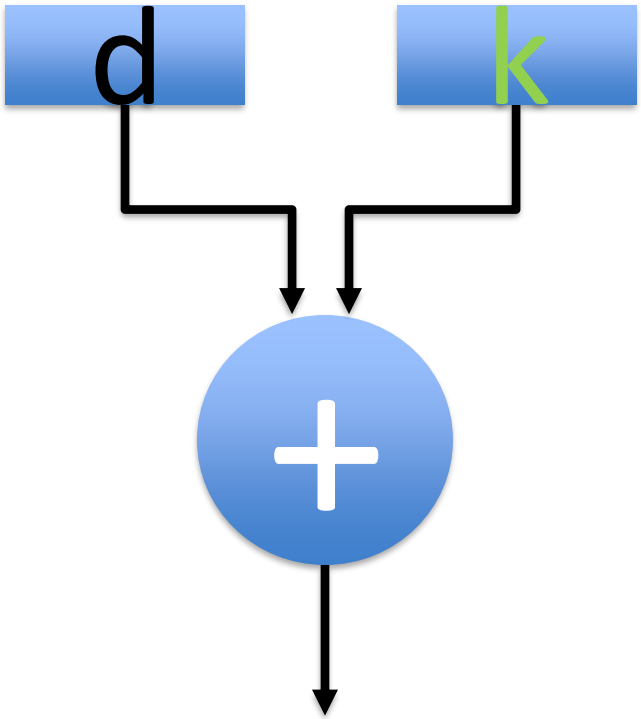
The Instructions and the ISA

… …	… … …	
0789	load	R02,4000
0790	load	R03,4004
0791	add	R01,R02,R03
0792	load	R02,4008
0793	add	R01,R01,R02
0794	store	R01,4000
… …	… … …	



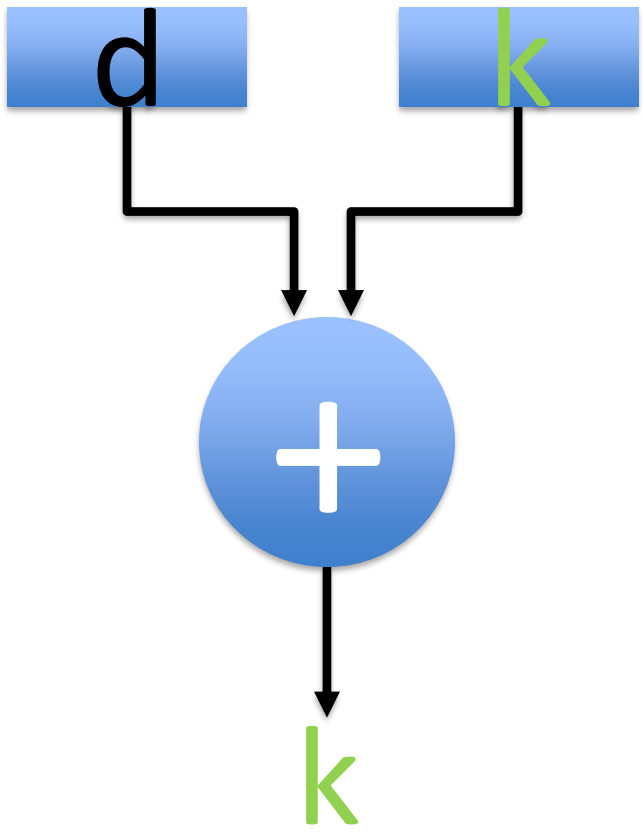
The Instructions and the ISA

… …	… … …	
0789	load	R02,4000
0790	load	R03,4004
0791	add	R01,R02,R03
0792	load	R02,4008
0793	add	R01,R01,R02
0794	store	R01,4000
… …	… … …	

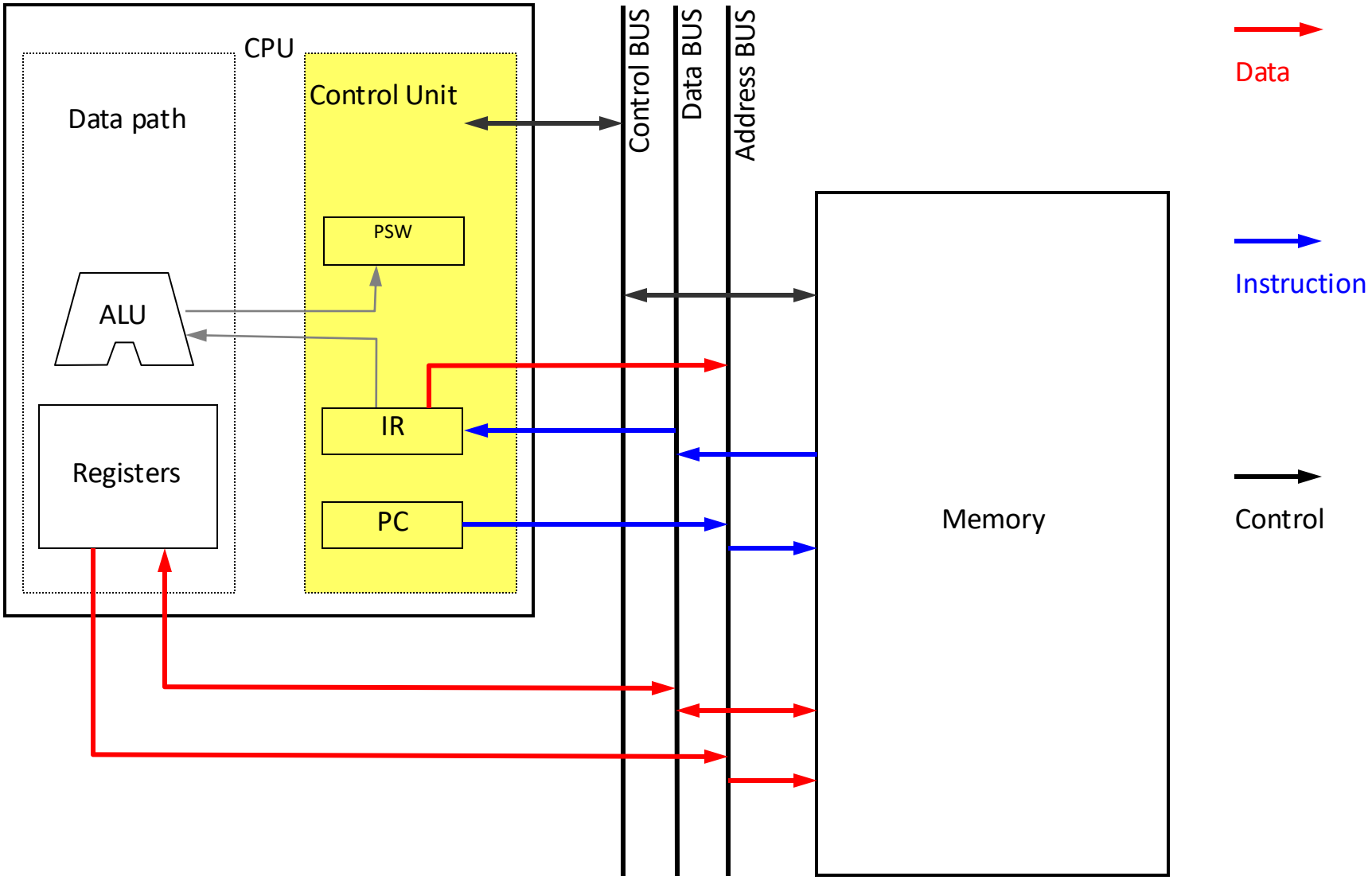


The Instructions and the ISA

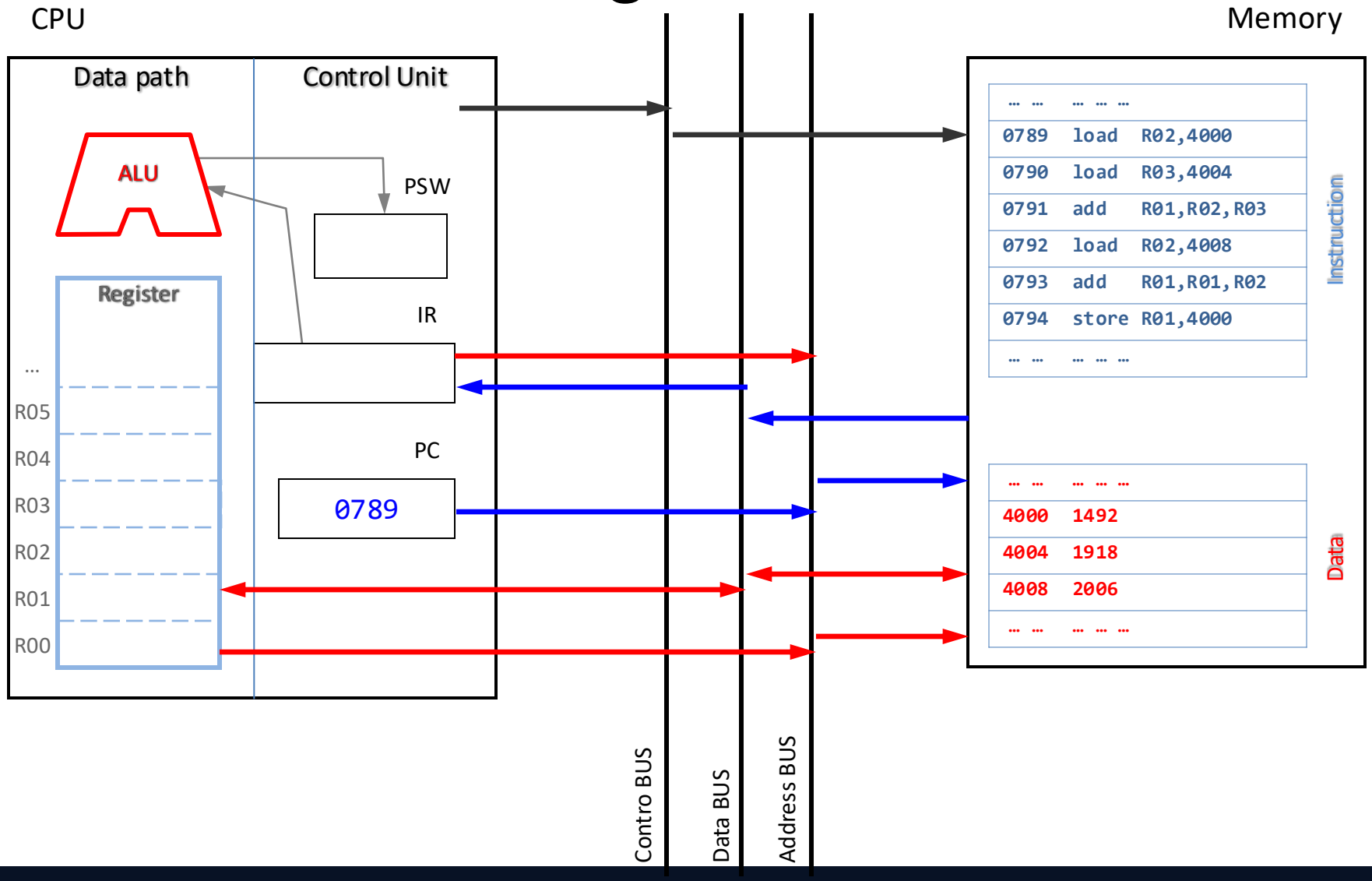
```
… …   … … …
0789  load  R02,4000
0790  load  R03,4004
0791  add   R01,R02,R03
0792  load  R02,4008
0793  add   R01,R01,R02
0794  store R01,4000
… …   … … …
```



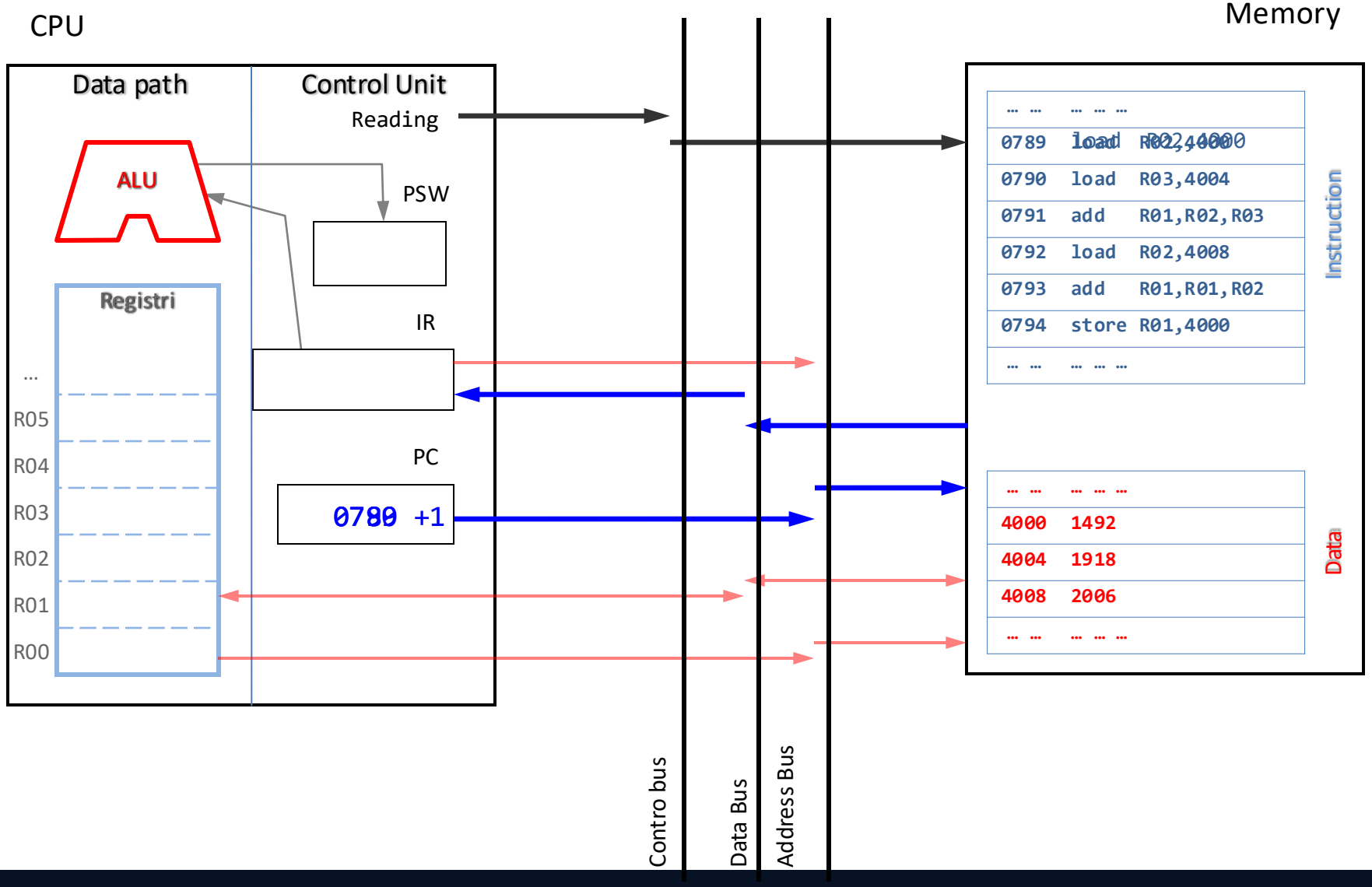
Computing Infrastructure



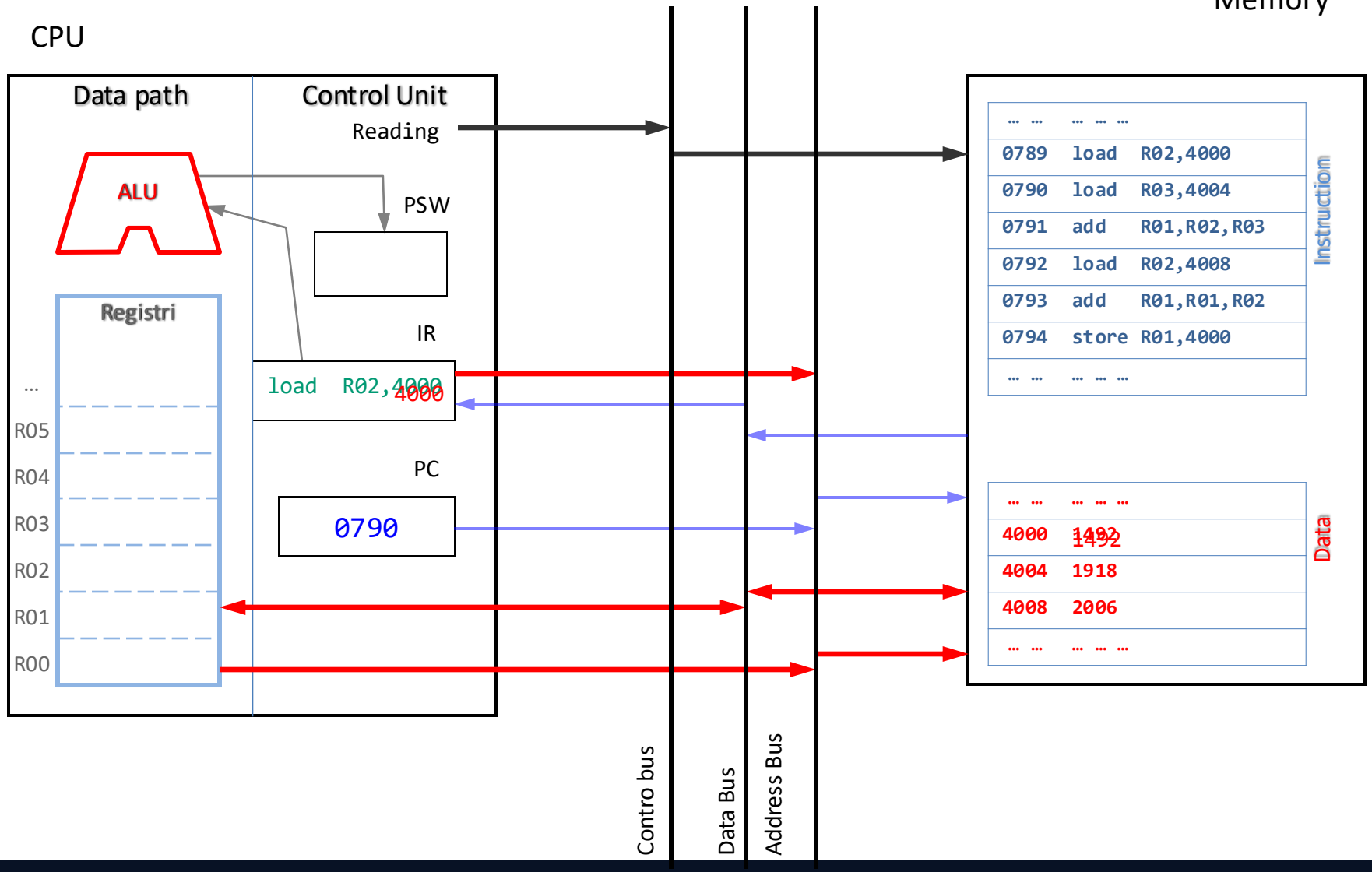
Starting scenario



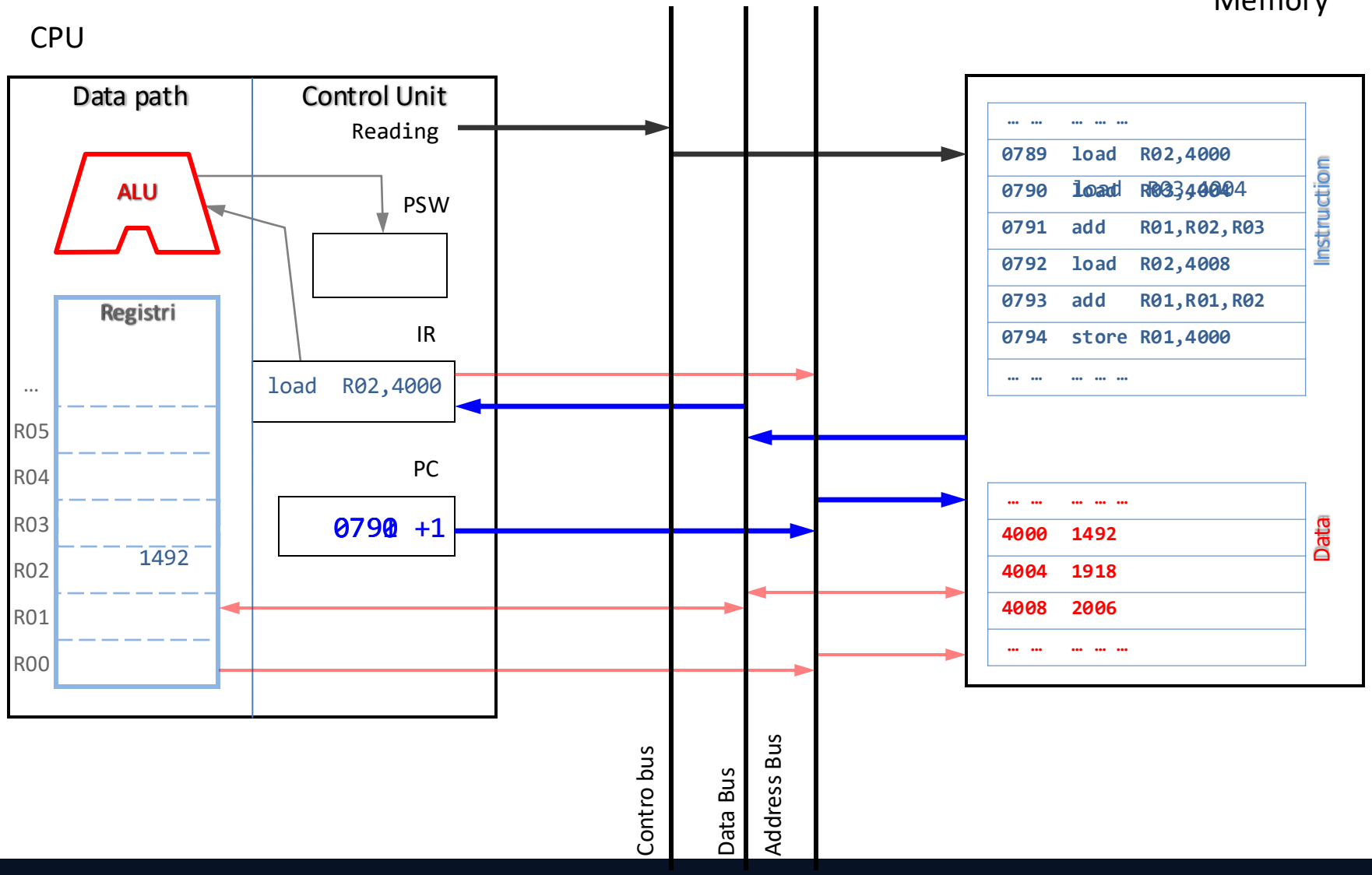
Read Instruction 0789



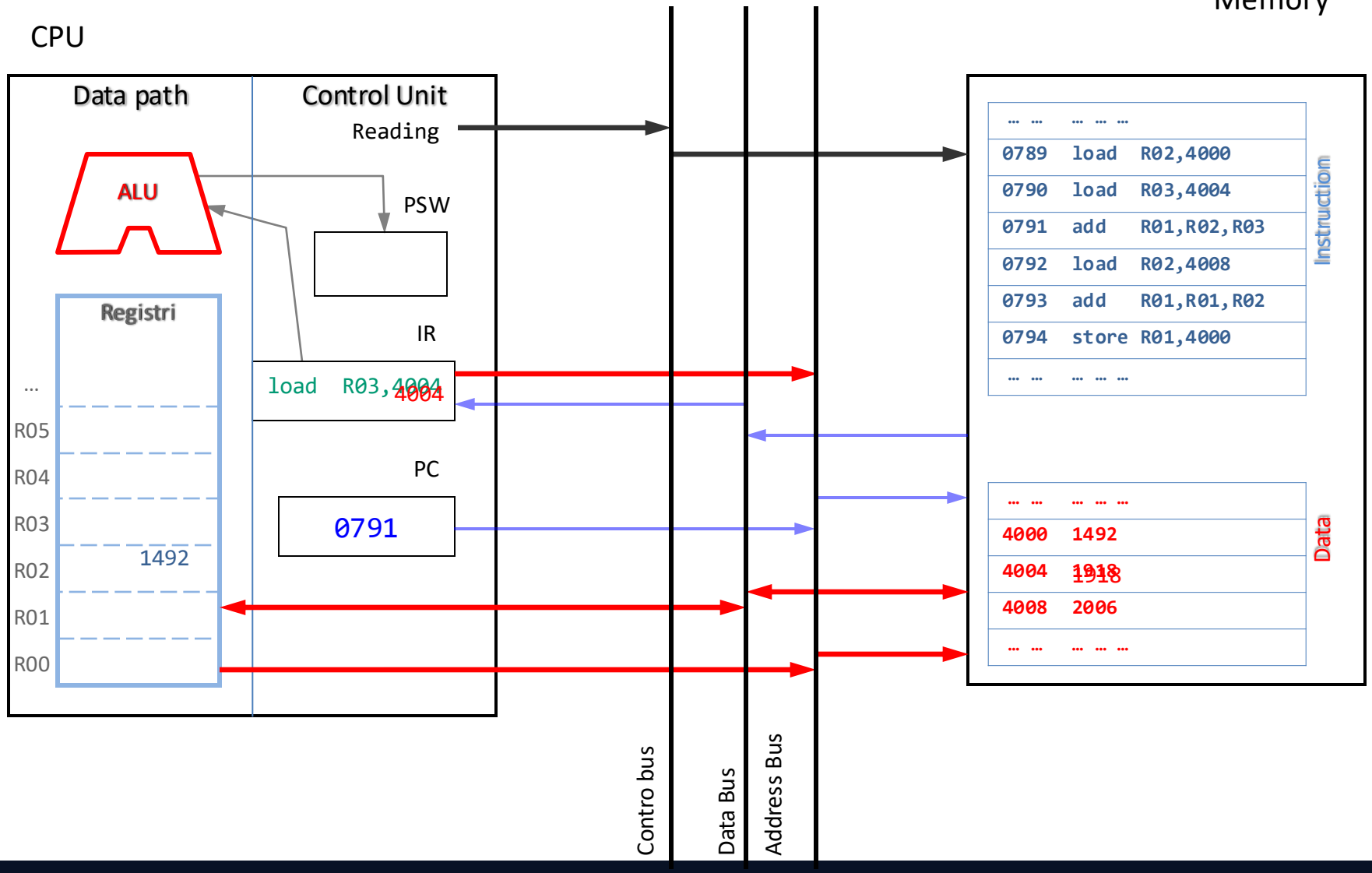
Exe Instruction 0789



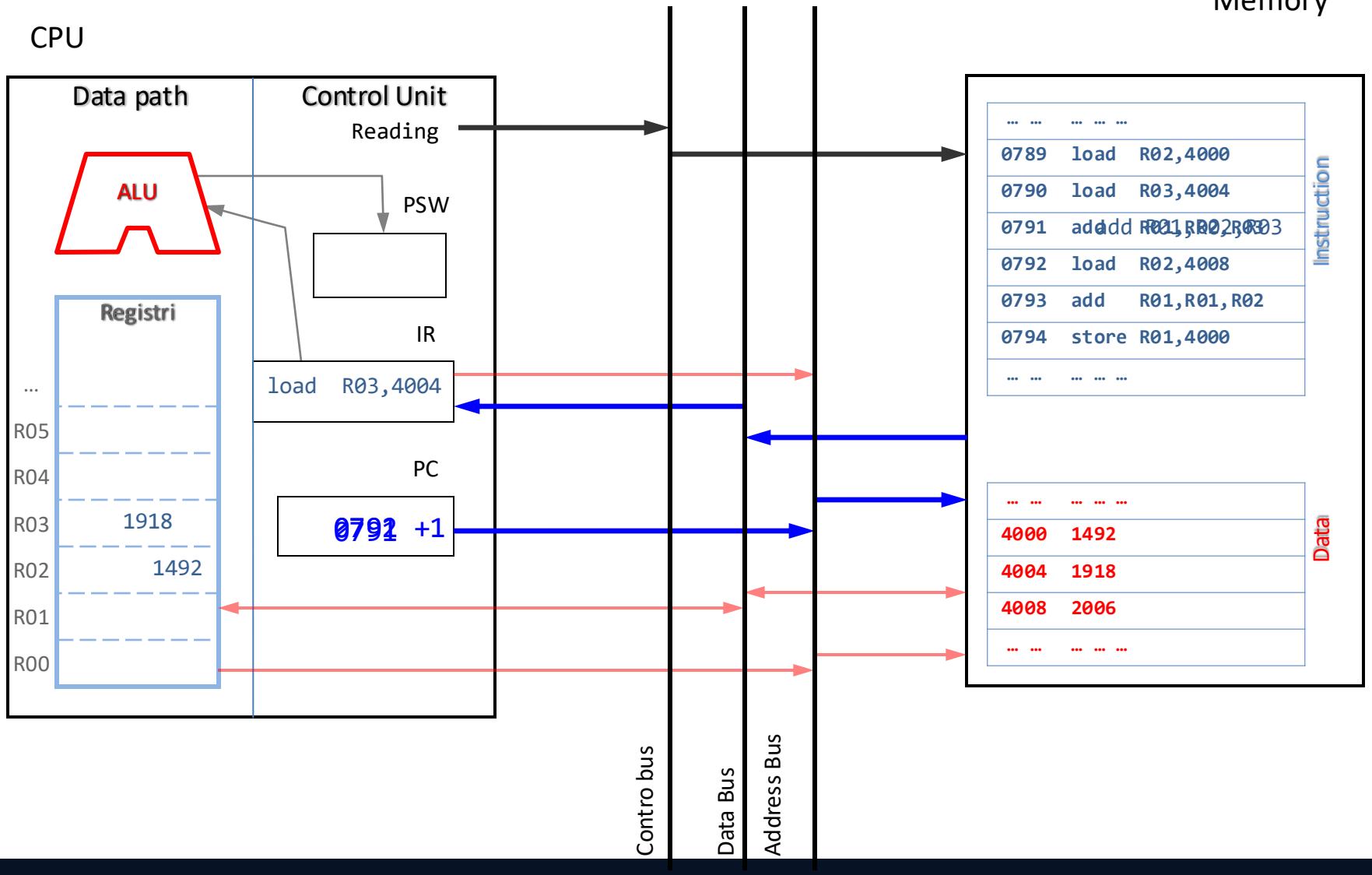
Read instruction 0790



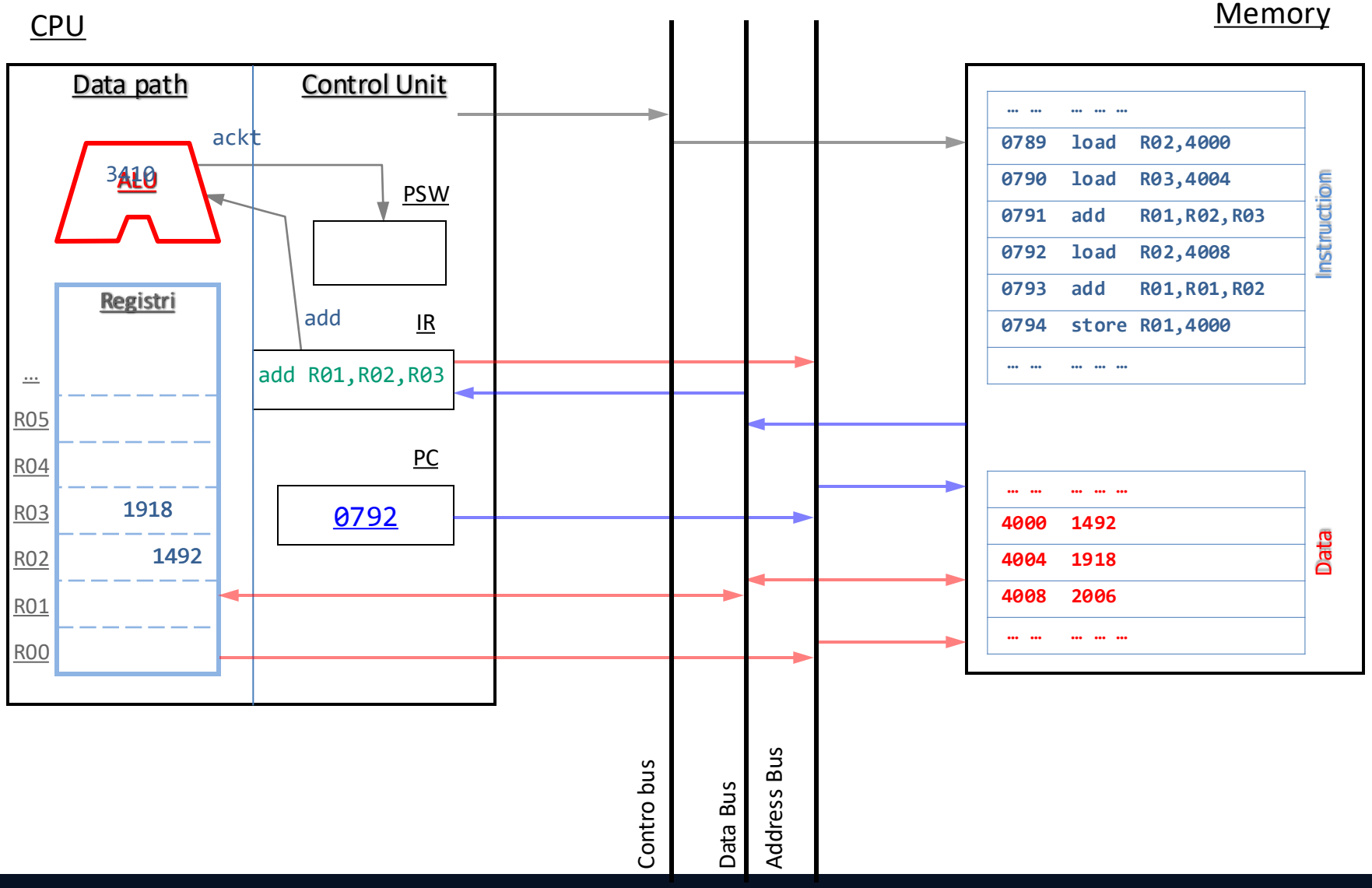
Exe Instruction 0790



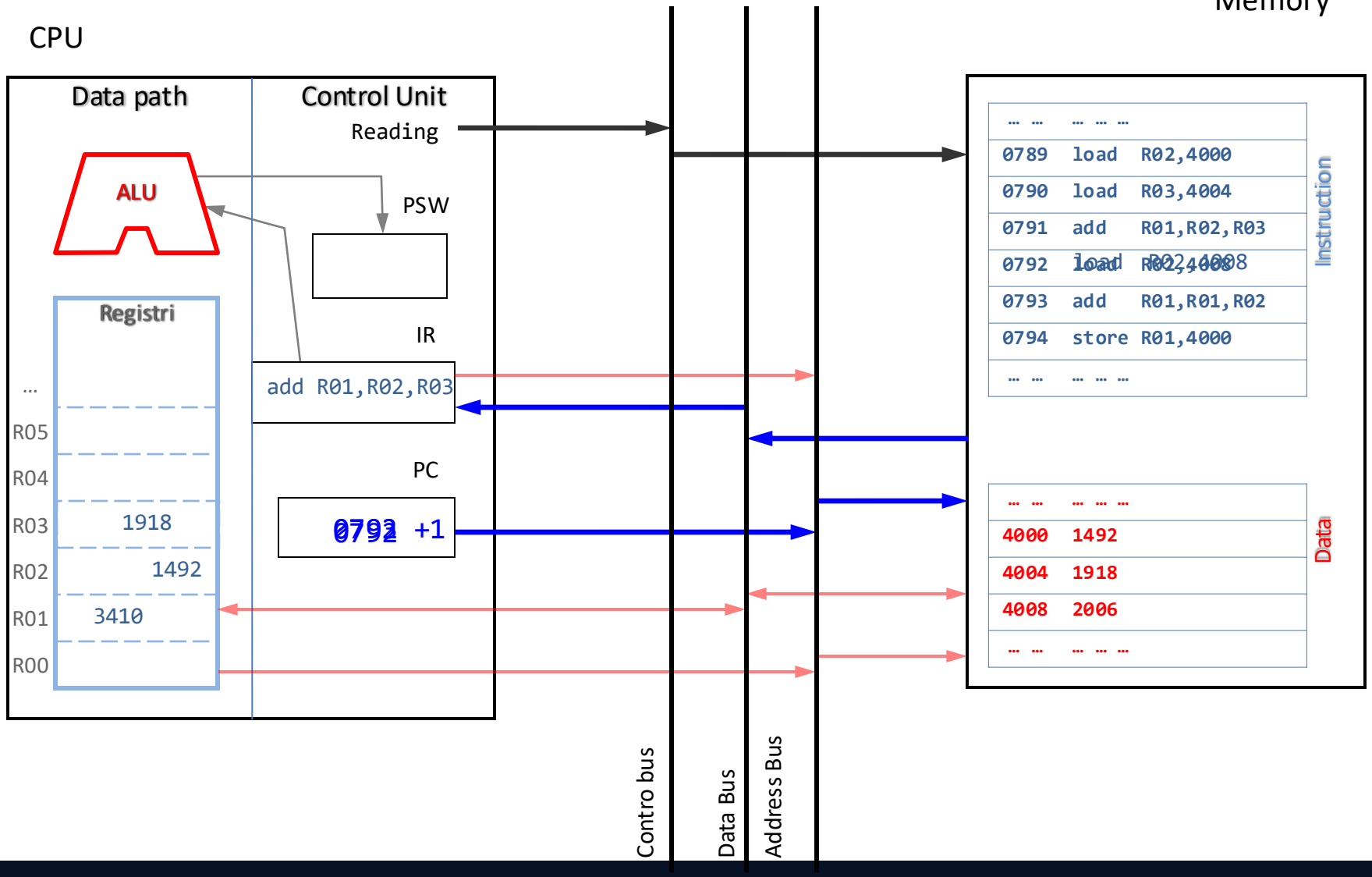
Read Instruction 0791



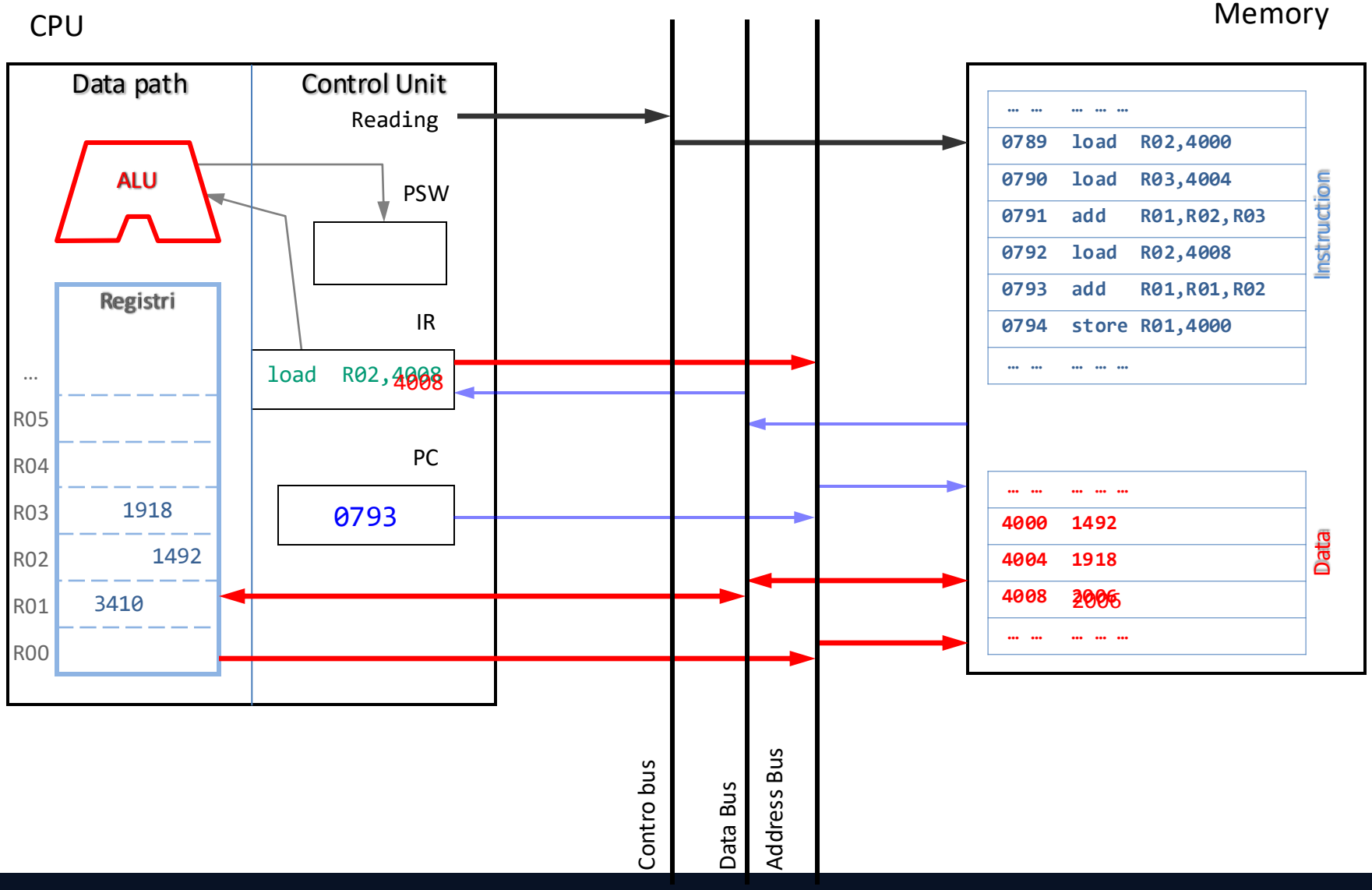
Exe Instruction 0791



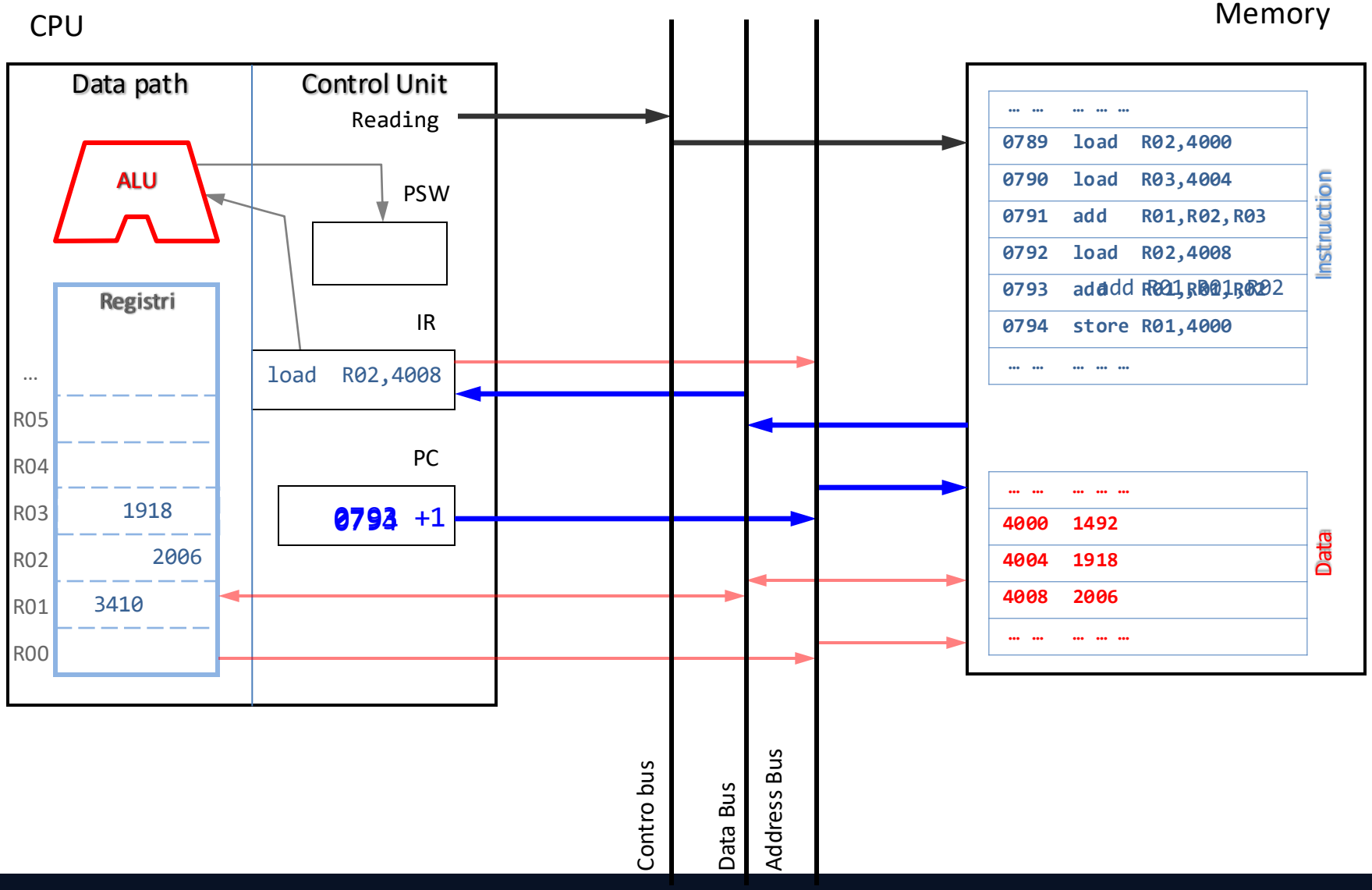
Read Instruction 0792



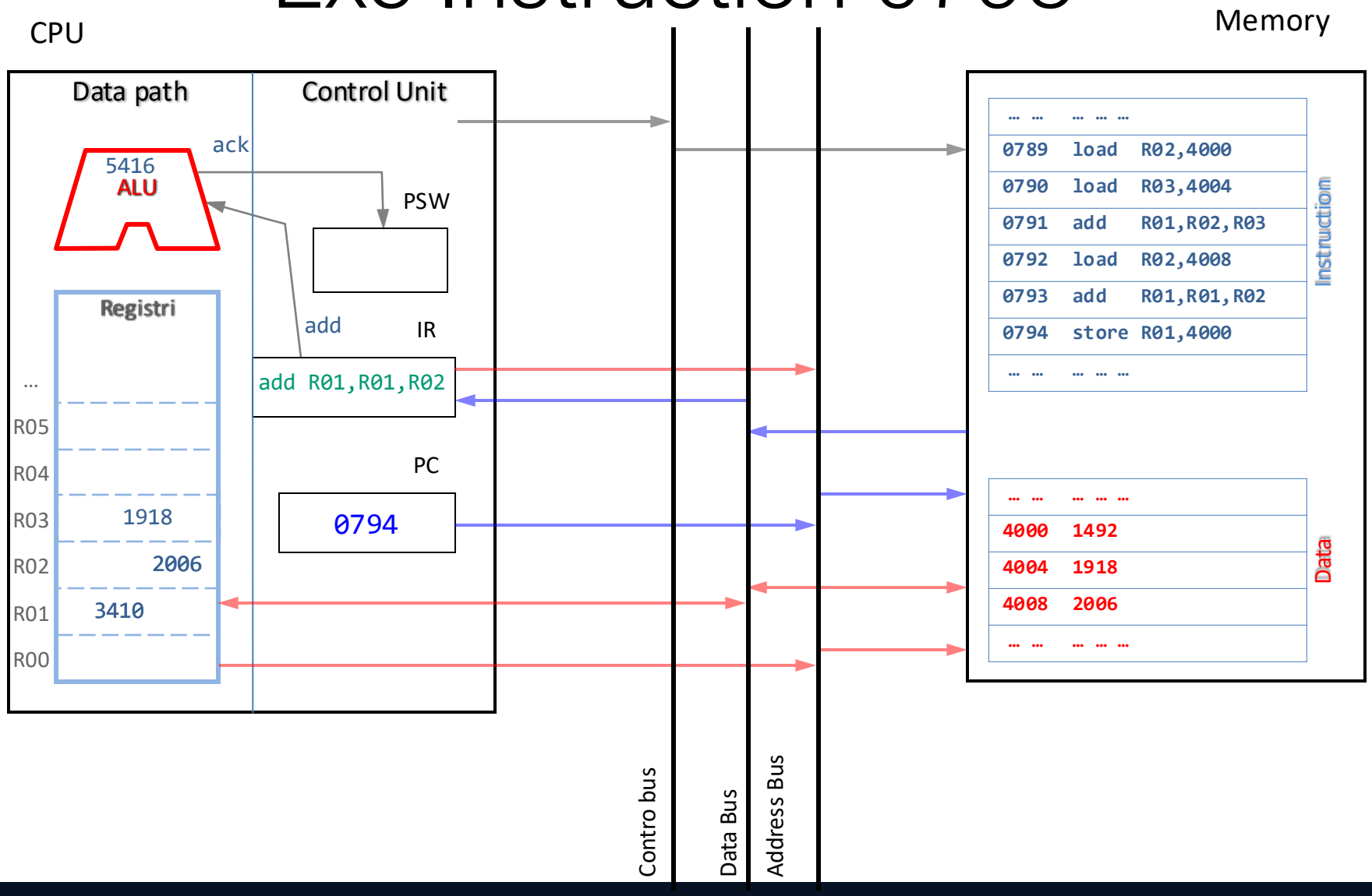
Exe Instruction 0792



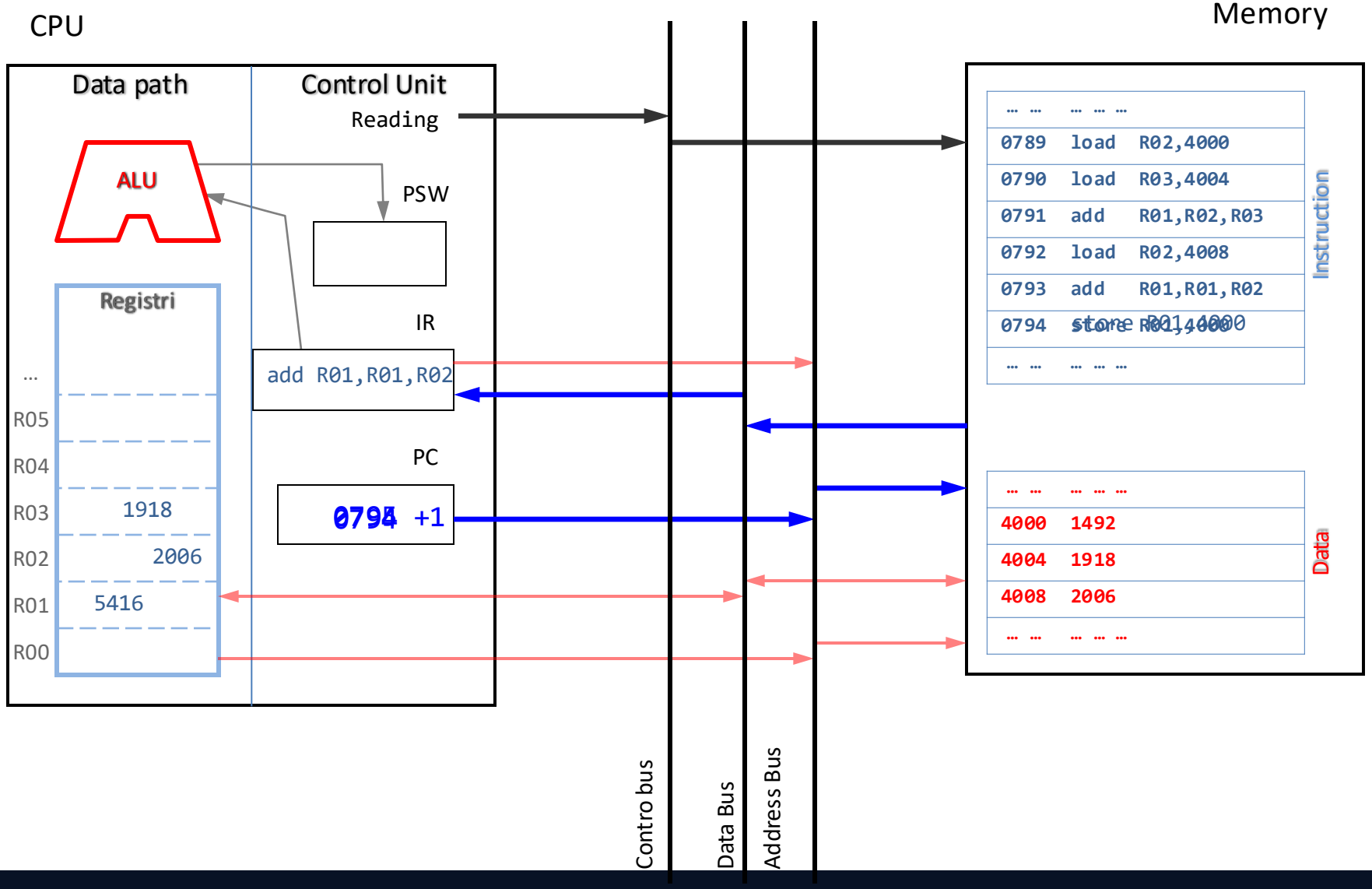
Read Instruction 0793



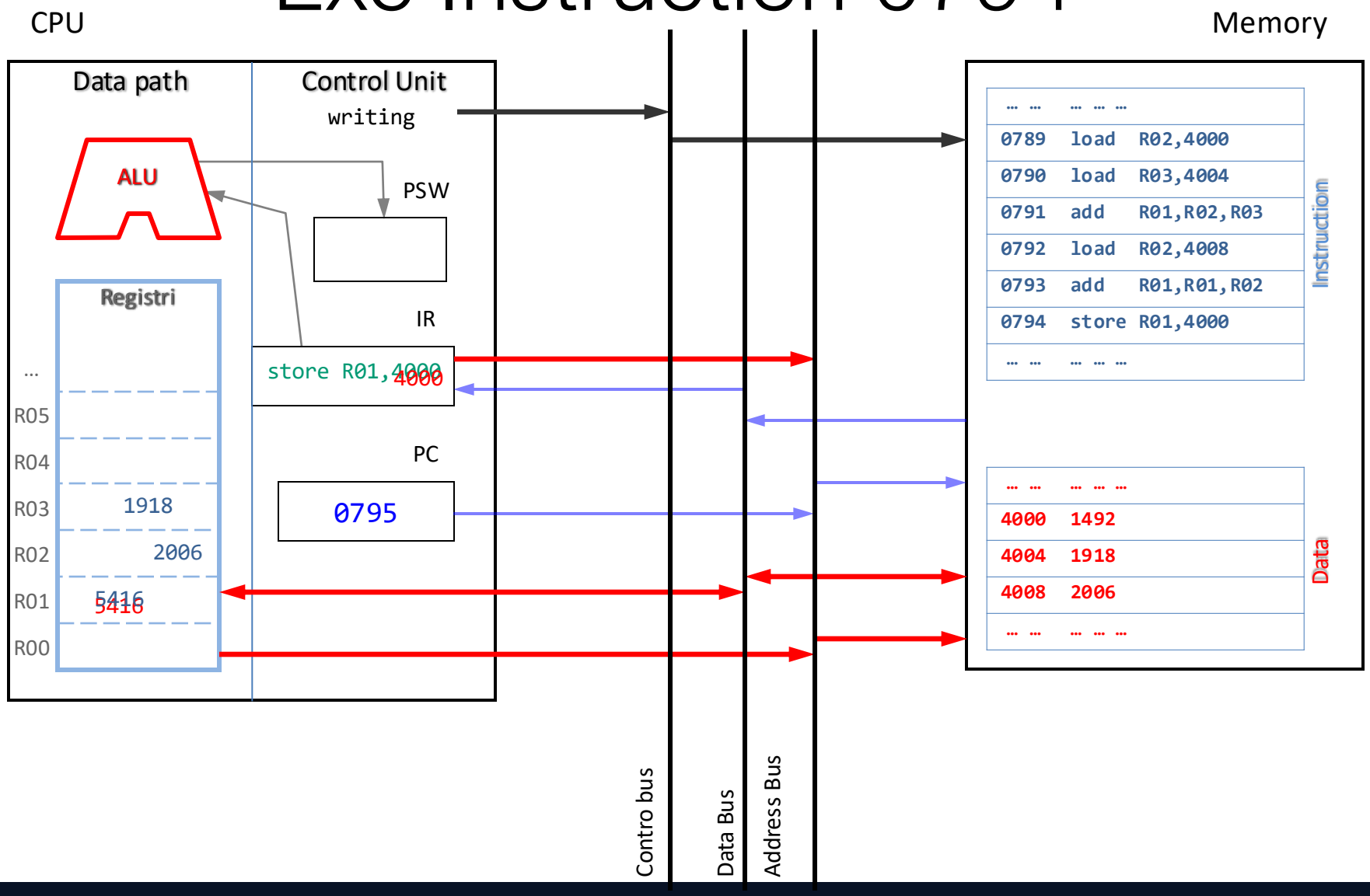
Exe Instruction 0793



Read Instruction 0794



Exe Instruction 0794



Execution of MIPS Instructions

ALU Instructions: `op $x,$y,$z`

Instr. Fetch &. PC Increm.	Read of Source Regs. `$y` and `$z`	ALU OP (`$y op $z`)	Write Back of Destinat. Reg. `$x`

Load Instructions: `lw $x,offset($y)`

Instr. Fetch & PC Increm.	Read of Base Reg. `$y`	ALU Op. (`$y+offset`)	Read Mem. `M($y+offset)`	Write Back of Destinat. Reg. `$x`

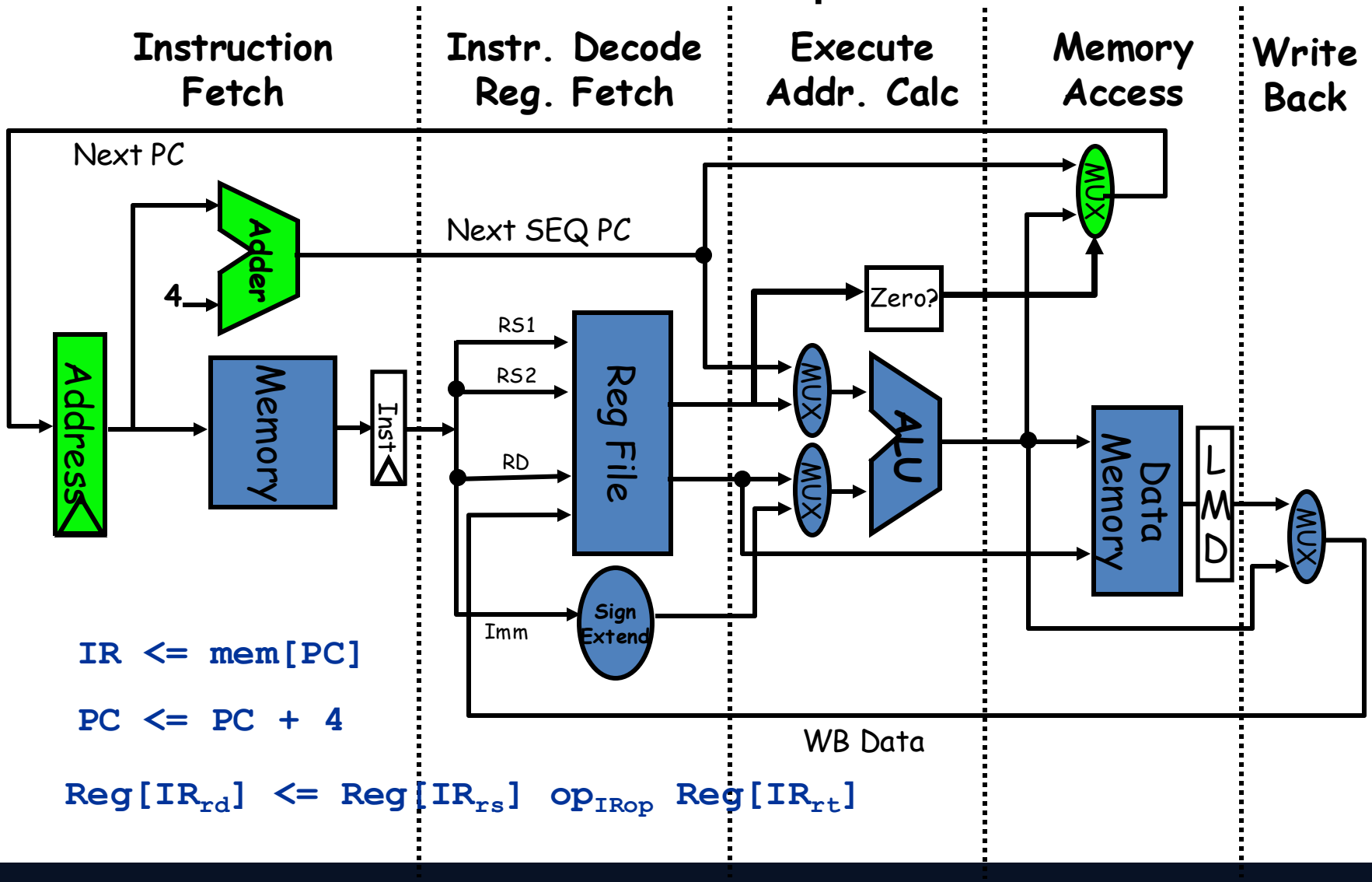
Store Instructions: `sw $x,offset($y)`

Instr. Fetch & PC Increm.	Read of Base Reg. `$y` & Source `$x`	ALU Op. (`$y+offset`)	Write Mem. `M($y+offset)`

Conditional Branch: `beq $x,$y,offset`

Instr. Fetch & PC Increm.	Read of Source Regs. `$x` and `$y`	ALU Op. (`$x-$y`) & (`PC+4+offset`)	Write PC

MIPS Data path




BUT REMEMBER...
https://flic.kr/p/8puKvo

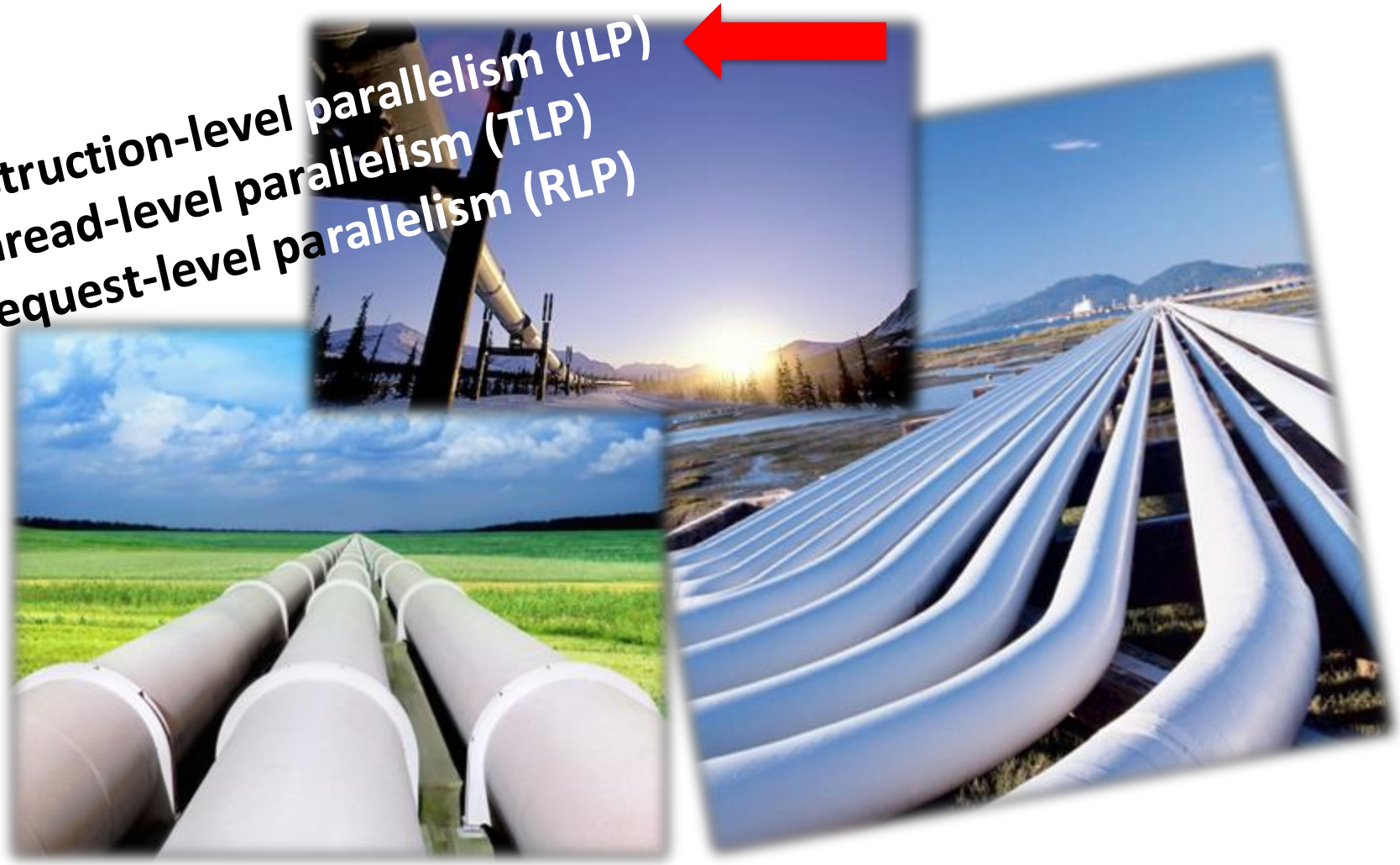
But the world is "parallel"

- Events are happening simultaneously
  - Many complex, interrelated events happening at the same time, yet within a sequence:

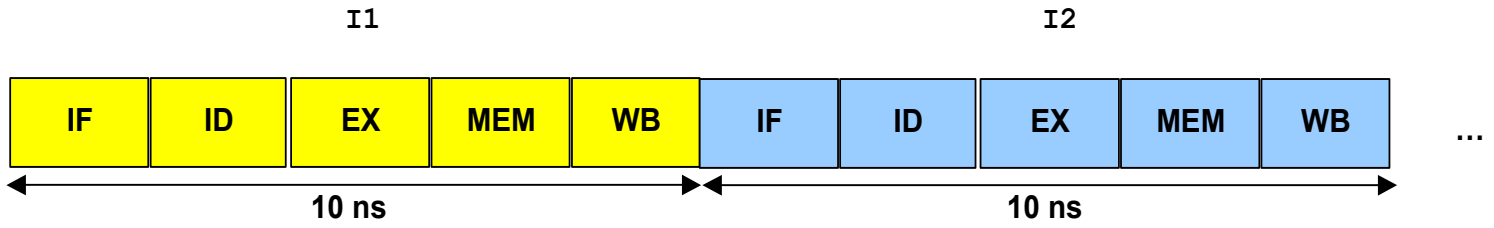
- Some examples:
  - Galaxy formation
  - Planetary movement
  - Tectonic plate drift
  - Rush hour traffic
  - Automobile assembly line
  - Building a space shuttle
  - Ordering a hamburger at the drive through

Parallelism? Which kind?

Parallelism? Which kind?



Sequential vs. Pipelining Execution



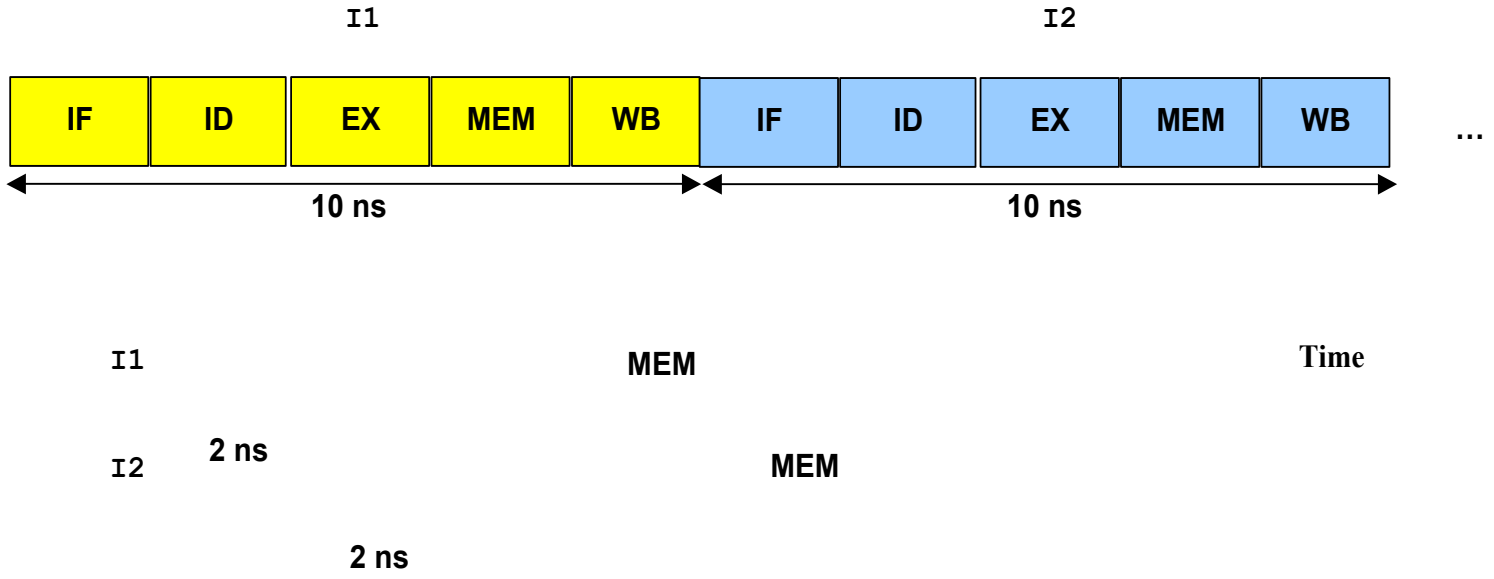
I1

MEM

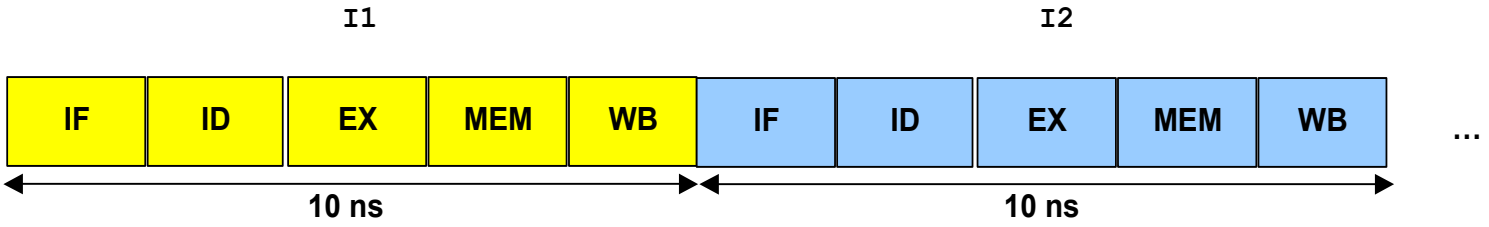
Time

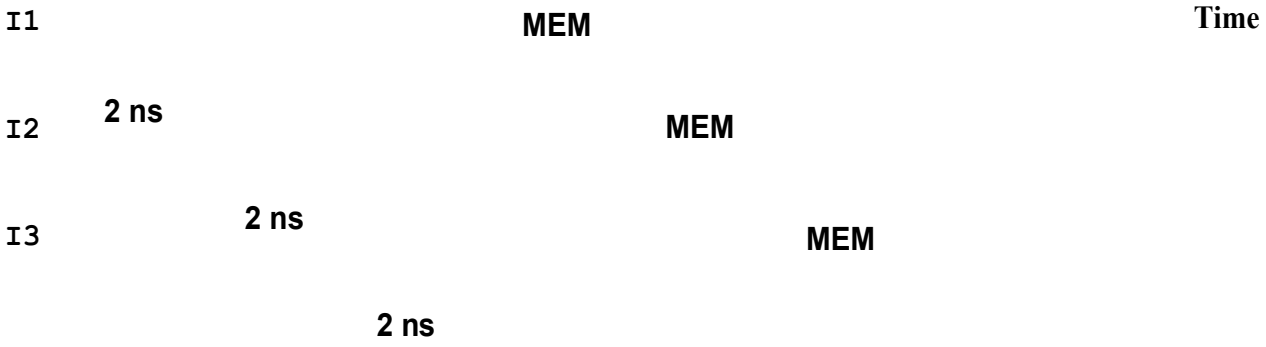
2 ns

Sequential vs. Pipelining Execution

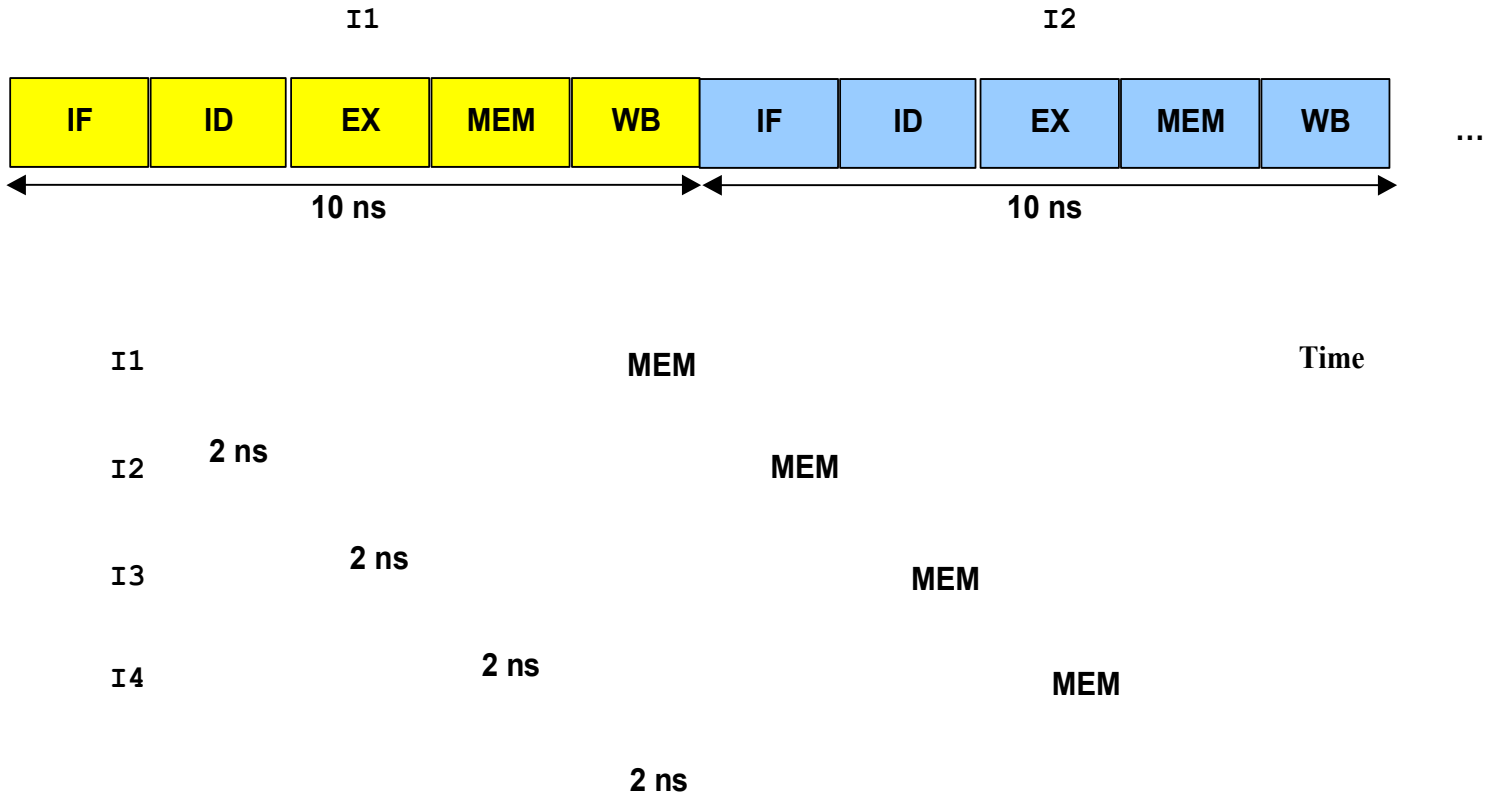


Sequential vs. Pipelining Execution

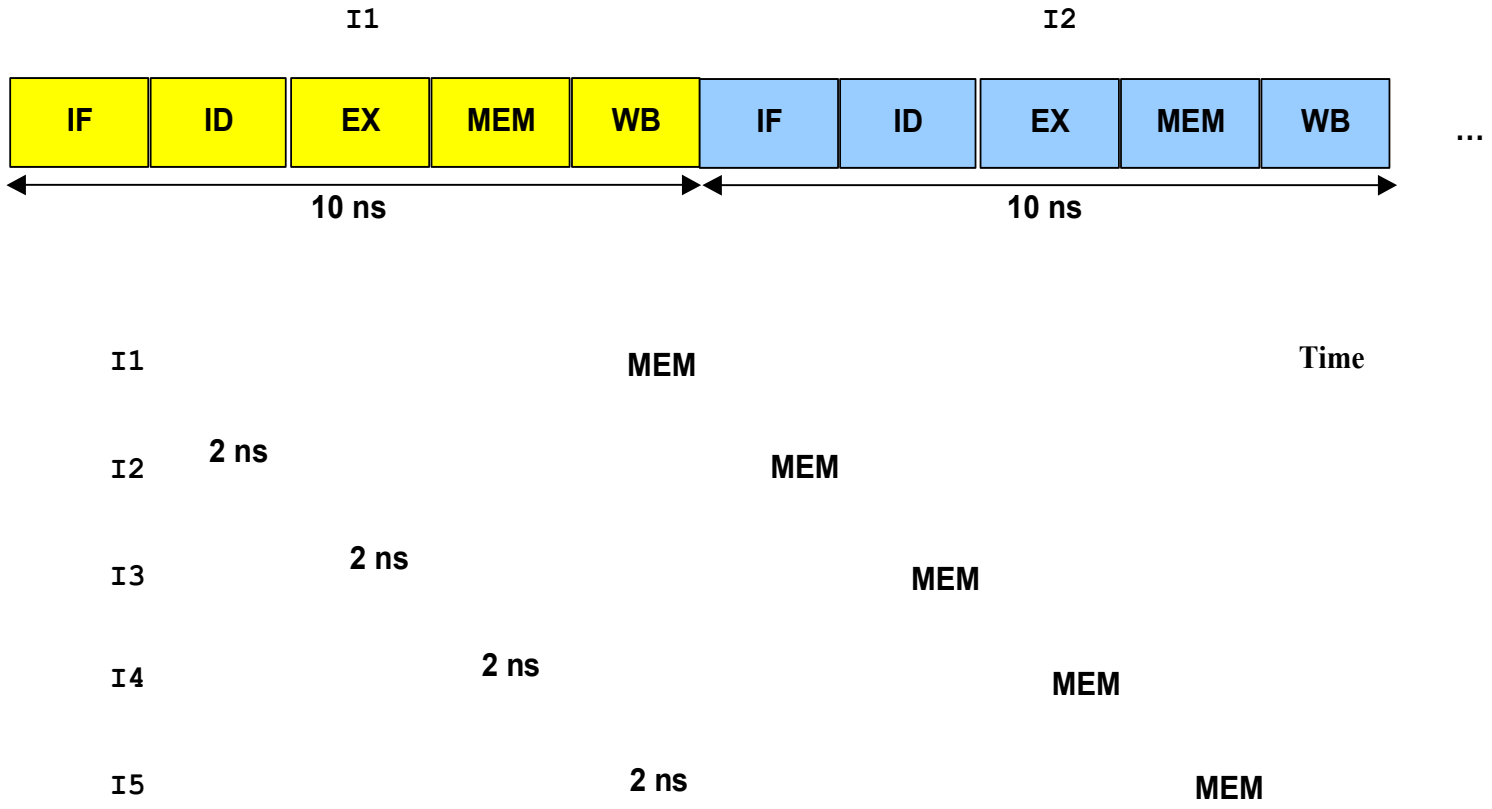




Sequential vs. Pipelining Execution



Sequential vs. Pipelining Execution



ONLINE CLASSES

https://tinyurl.com/video-ACA-polimi

MORE ON PIPELINING, CONFLICTS/HAZARDS

VIDEOS: C2

Parallelism? Which kind?

Parallel programming

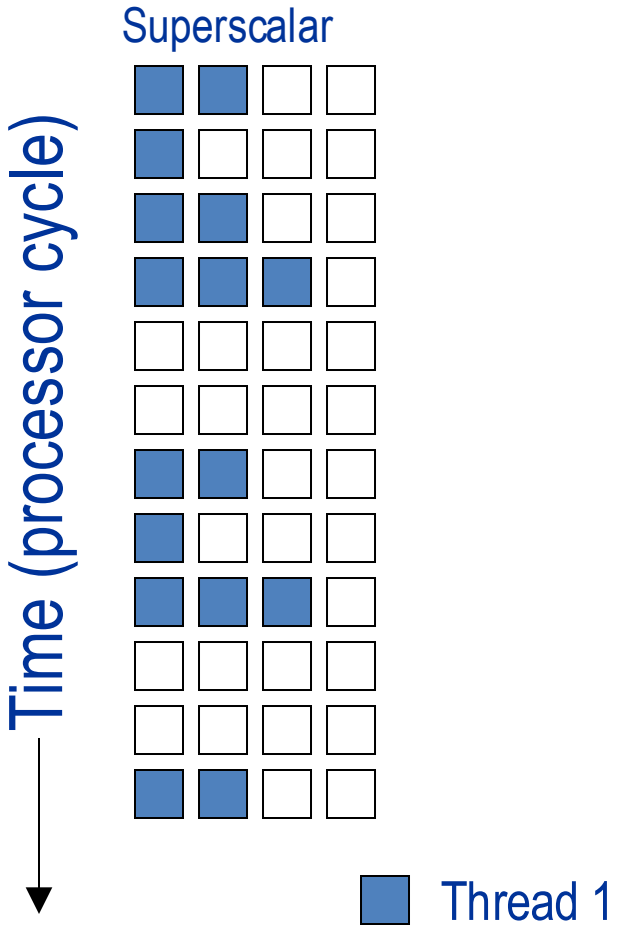
- Explicit parallelism implies structuring the applications into concurrent and communicating tasks
- Operating systems offer support for different types of tasks. The most important and frequent are:
  - processes
  - threads
- The operating systems implement multitasking differently based on the characteristics of the processor:
  - single core
  - single core with multithreading support

Process/Thread

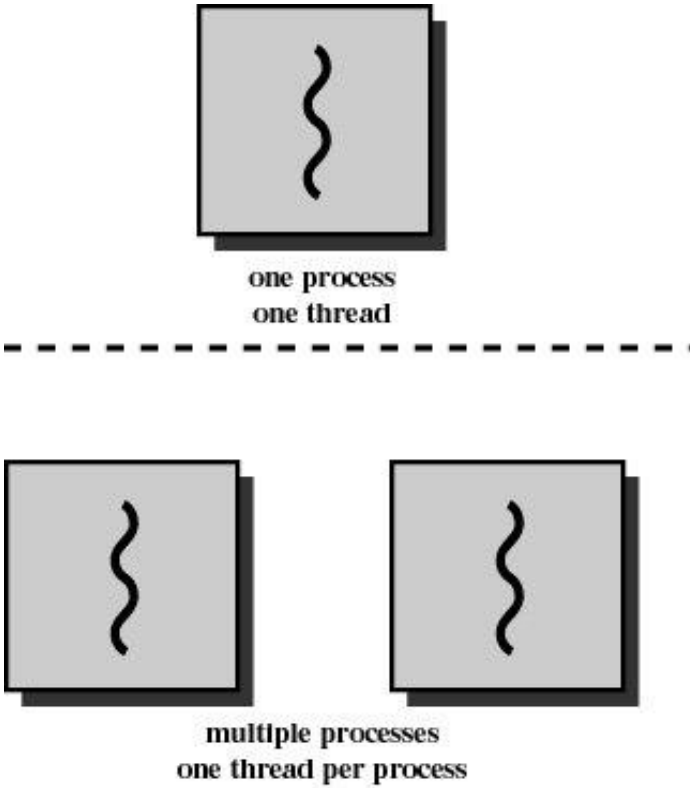
one process
one thread

= instruction trace

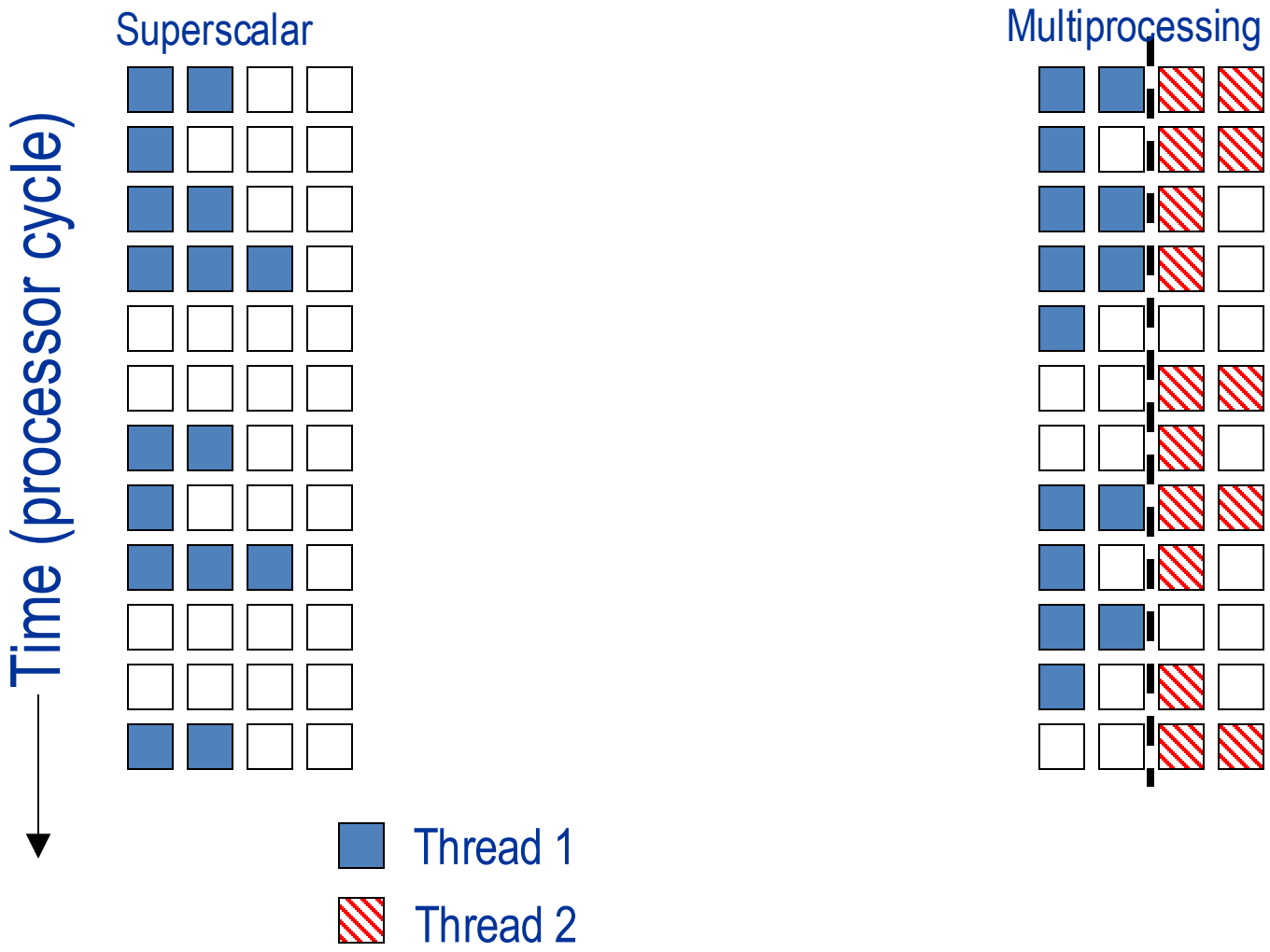
Multithreaded Categories



Multiplicity of processes



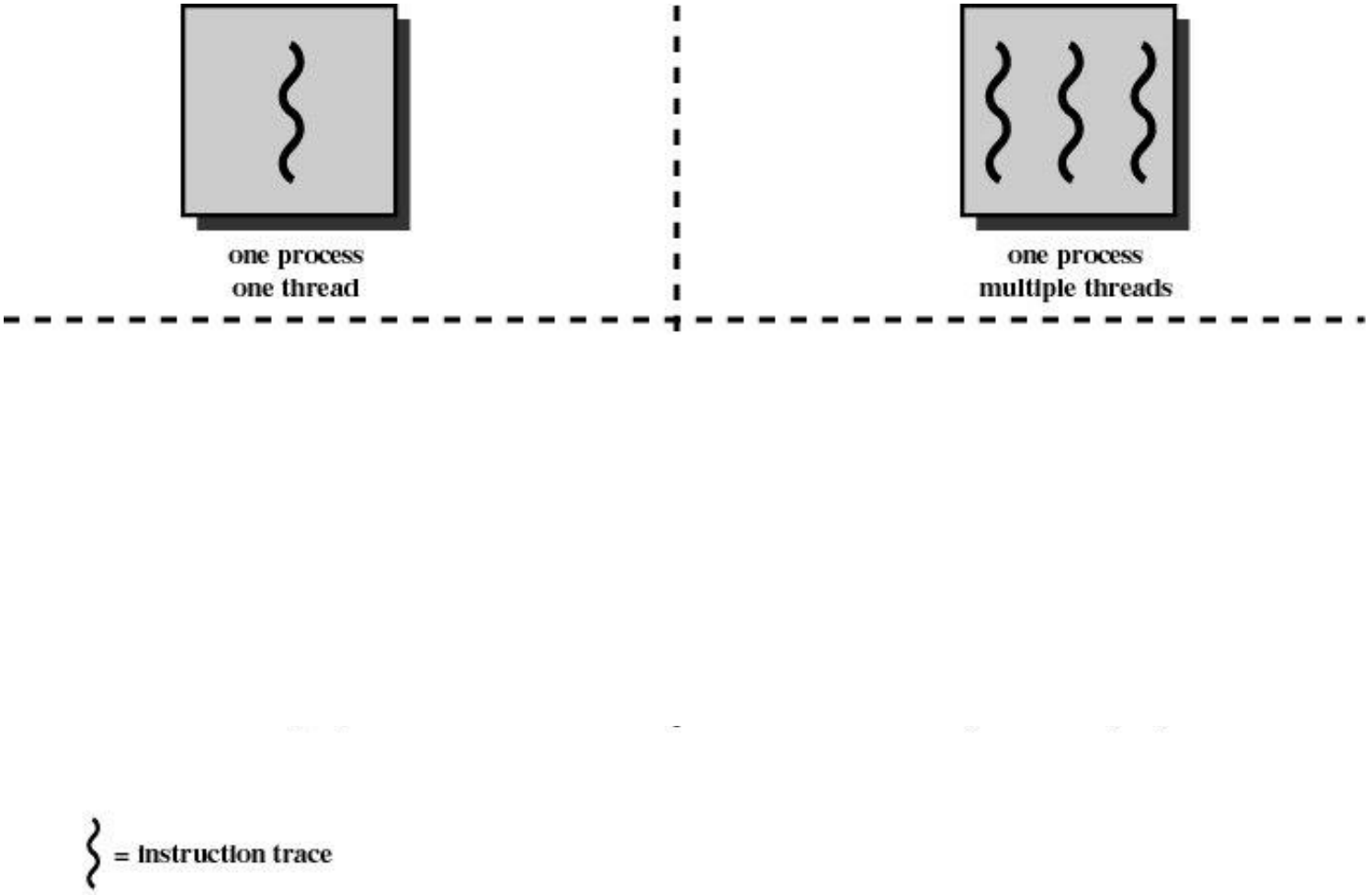
Multithreaded Categories



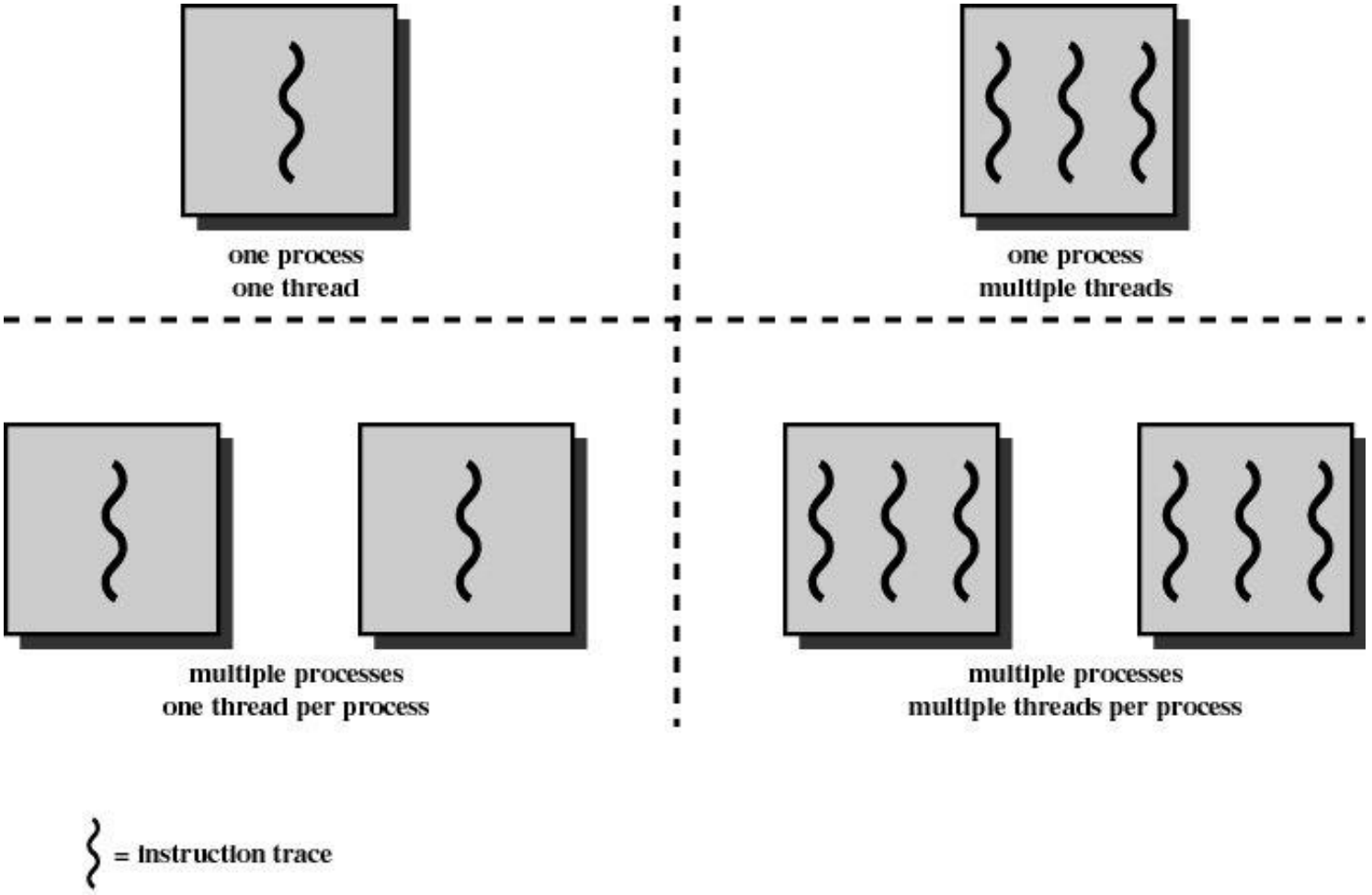
Parallel programming

- Explicit parallelism implies structuring the applications into concurrent and communicating tasks
- Operating systems offer support for different types of tasks. The most important and frequent are:
  - processes
  - threads
- The operating systems implement multitasking differently based on the characteristics of the processor:
  - single core
  - single core with multithreading support
  - multicore

Multiplicity of threads



Multiplicity of processes/threads



Thread-level parallelism (TLP)

- Fine grained multithreading
  - Switches from one thread to the other at each instruction – the execution of more threads is interleaved (often the switching is performed taking turns, skipping one thread if there is a stall)
  - The CPU must be able to change thread at every clock cycle. It is necessary to duplicated the hardware resources.

Multithreaded Categories

Fine grained multithreading

- Advantage is it can hide both short and long stalls, since instructions from other threads executed when one thread stalls

- Disadvantage is it slows down execution of individual threads, since a thread ready to execute without stalls will be delayed by instructions from other threads

Thread-level parallelism (TLP)

- Fine grained multithreading
  - Switches from one thread to the other at each instruction – the execution of more threads is interleaved (often the switching is performed taking turns, skipping one thread if there is a stall)
  - The CPU must be able to change thread at every clock cycle. It is necessary to duplicated the hardware resources.
- Coarse grained multithreading
  - switching from one thread to another occurs only when there are long stalls – e.g., for a miss on the second level cache.
  - Two threads share many system resources (e.g., architectural registers), the switching from one thread to the next requires different clock cycles to save the context.

Multithreaded Categories

Coarse grained multithreading

- Advantage: in normal conditions the single thread is not slowed down
  - Relieves need to have very fast thread-switching
  - Doesn't slow down thread, since instructions from other threads issued only when the thread encounters a costly stall

- Disadvantage: for short stalls it does not reduce the throughput loss – the CPU starts the execution of instructions that belonged to a single thread, when there is one stall it is necessary to empty the pipeline before starting the new thread

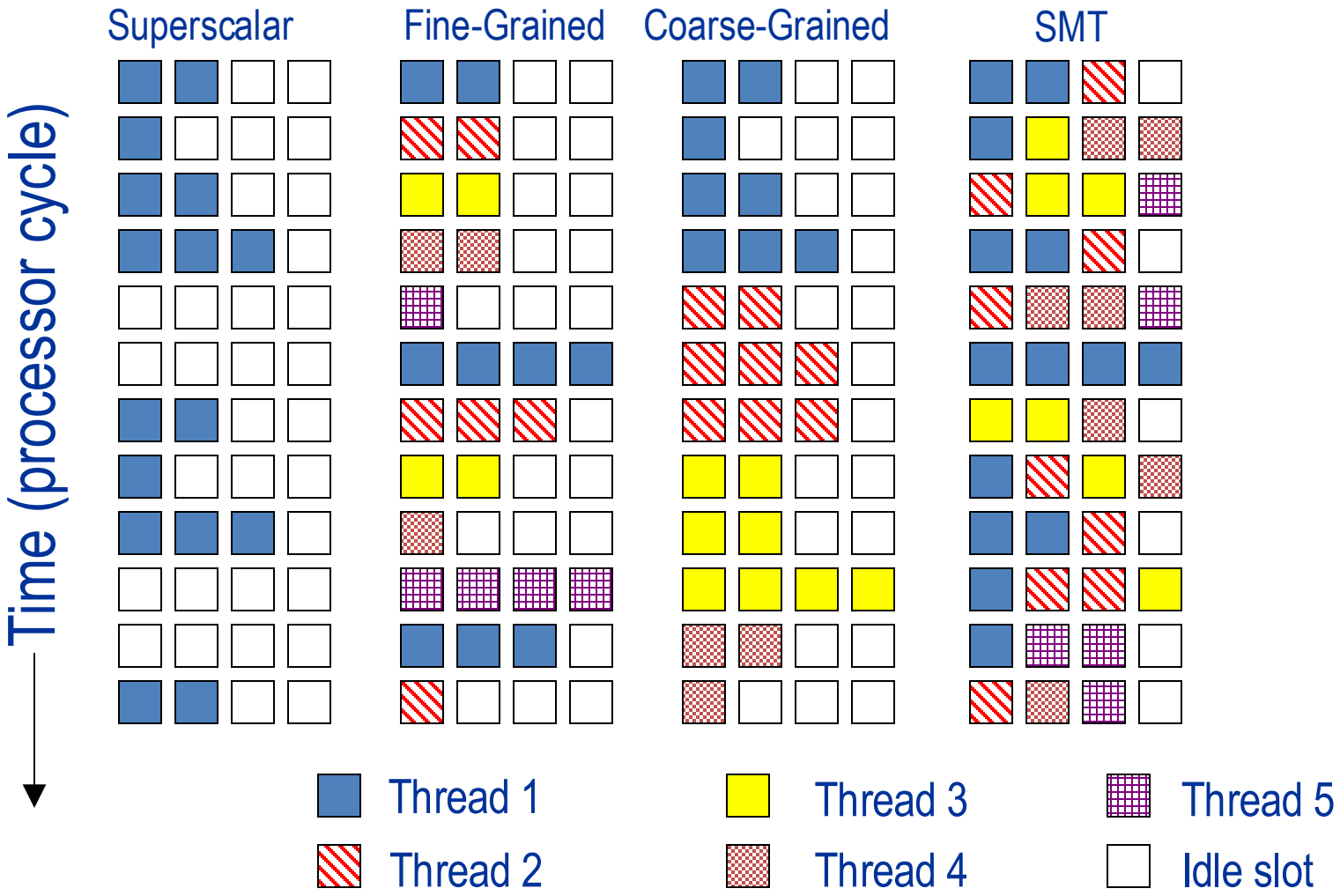
Do both ILP and TLP?

- TLP and ILP exploit two different kinds of parallel structure in a program
- Could a processor oriented at ILP to exploit TLP?
  - functional units are often idle in data path designed for ILP because of either stalls or dependences in the code
- Could the TLP be used as a source of independent instructions that might keep the processor busy during stalls?
- Could TLP be used to employ the functional units that would otherwise lie idle when insufficient ILP exists?

Simultaneous Multithreading (SMT)

- The system can be dynamically adapted to the environment, allowing (if possible) the execution of instructions from each thread, and allowing that the instructions of a single thread used all functional units if the other thread incurs in a long latency event.
- More threads use the issues possibilities of the CPU at each cycle; ideally, the exploitation of the issues availabilities is limited only by the unbalance between resources requests and availabilities.

Multithreaded Categories



What now?

- Difficult to increase performance and clock frequency of the single core
- Deep pipeline:
  - Heat dissipation problems
  - Speed light transmission problems in wires
  - Difficulties in design and verification
  - Requirement of very large design groups
- Many new applications are multi-threaded

Parallel programming

- Explicit parallelism implies structuring the applications into concurrent and communicating tasks
- Operating systems offer support for different types of tasks. The most important and frequent are:
  - processes
  - threads
- The operating systems implement multitasking differently based on the characteristics of the processor:
  - single core
  - single core with multithreading support
  - multicore

Flynn Taxonomy (1966)

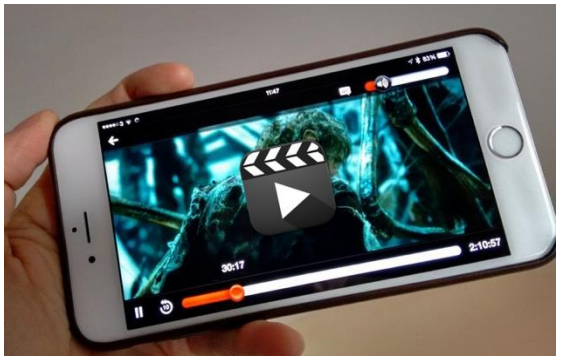
- SISD - Single Instruction Single Data
  - Uniprocessor systems
- MISD - Multiple Instruction Single Data
  - No practical configuration and no commercial systems
- SIMD - Single Instruction Multiple Data
  - Simple programming model, low overhead, flexibility, custom integrated circuits
- MIMD - Multiple Instruction Multiple Data
  - Scalable, fault tolerant, *off-the-shelf* micros

Workloads are dynamic

- On a mobile phone:
  - Phone calls
  - Short message service
  - Web browsing
  - Audio/video playing
  - Gaming

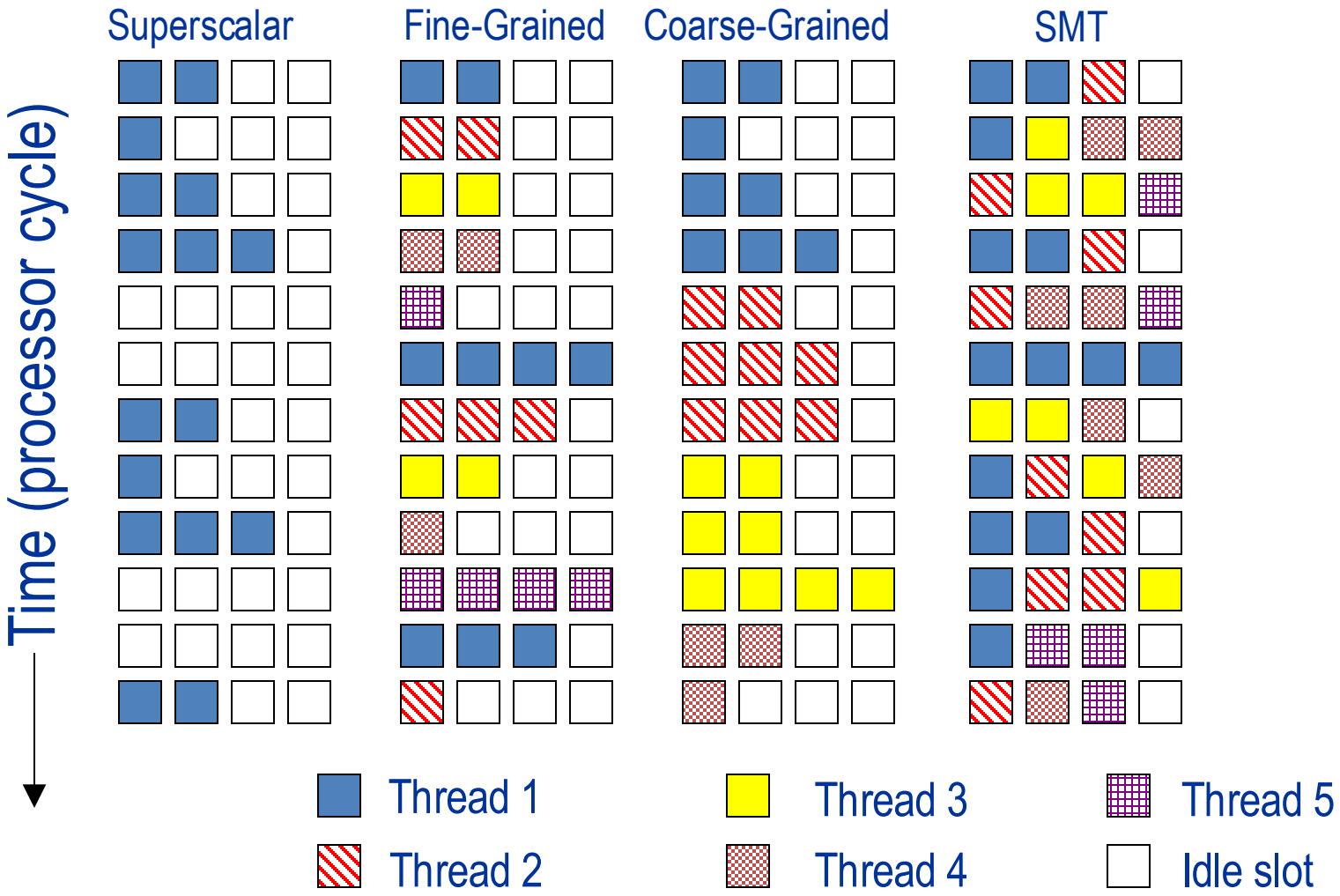
Generally short execution times, low amount of processed data, actually no QoS requirements or not-challenging ones

Generally considerable amount of data to be processed with specific throughputs to be fulfilled, high demanding elaborations




BUT BEFORE...
https://flic.kr/p/8puKvo

Multithreaded Categories



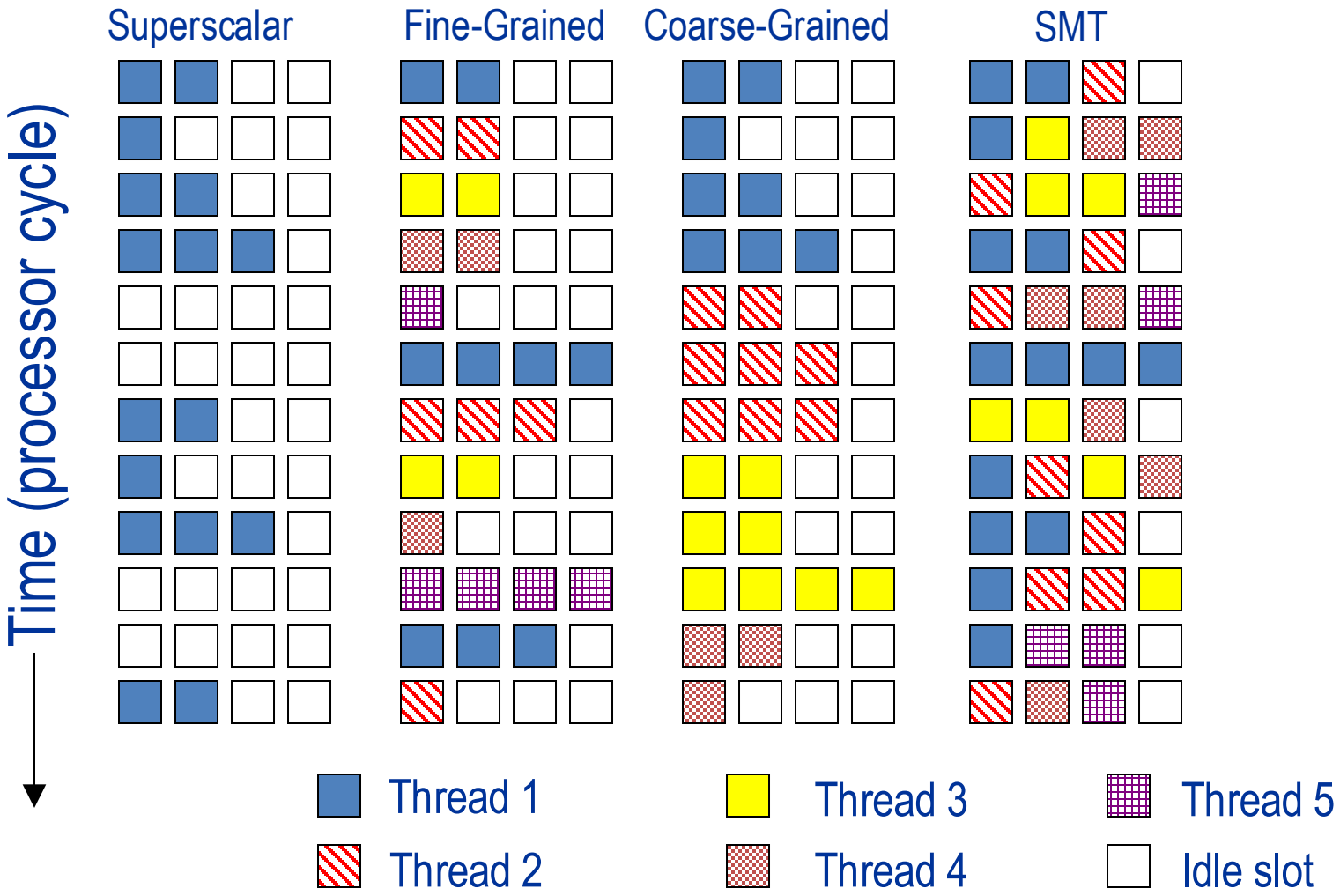
WAS IT CRISTAL CLEAR?

WAS IT CRISTAL CLEAR?

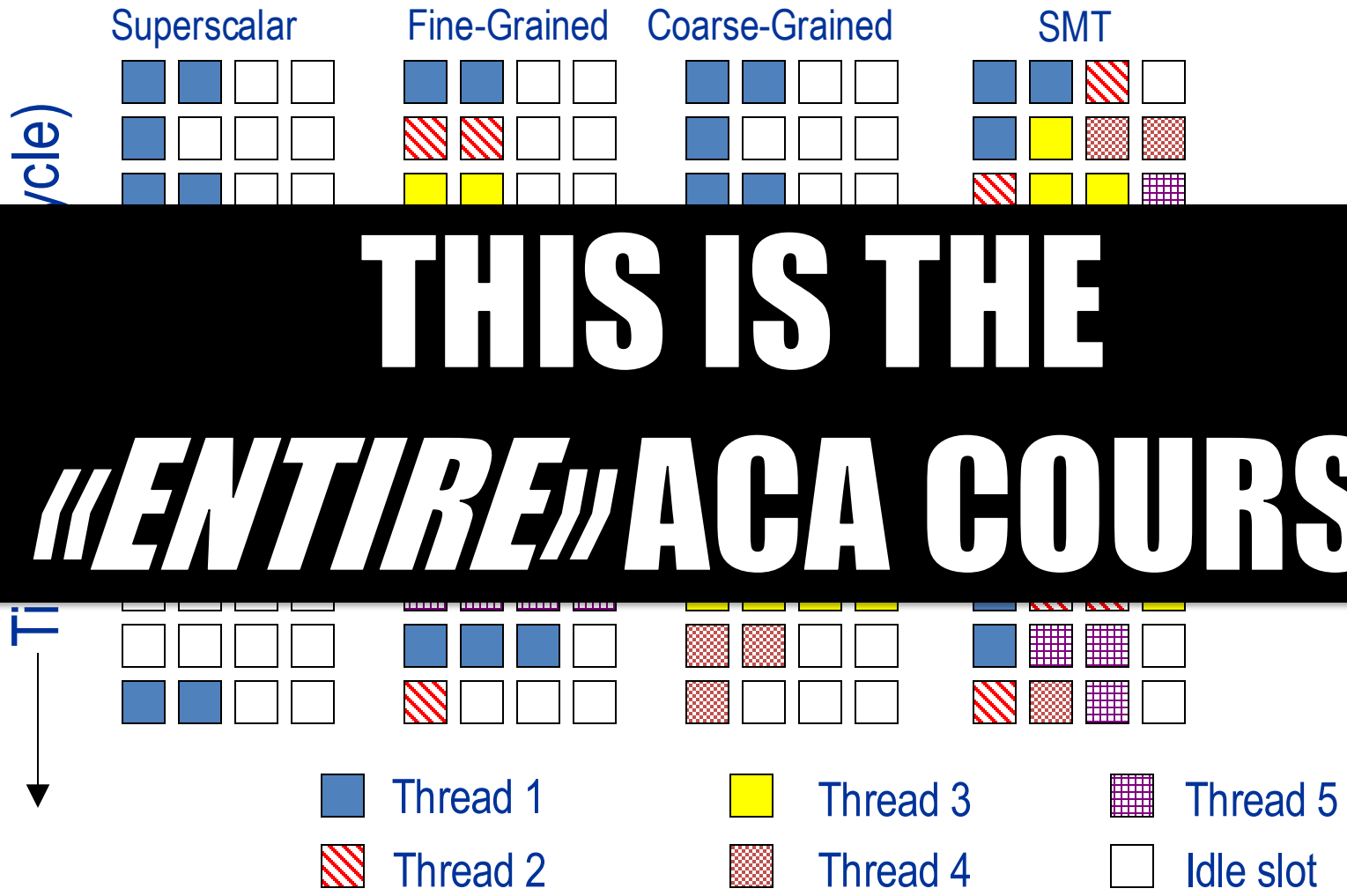
WAS IT CRISTAL CLEAR?

F**** NOT AS TRASH TALKING
BUT AS, TECH TERM

Multithreaded Categories



Multithreaded Categories



THIS IS THE "ENTIRE" ACA COURSE

QUESTIONS?

Alessandro Verosimile <Alessandro.verosimile@polimi.it>
Marco D. Santambrogio <marco.santambrogio@polimi.it>