UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

Instruction Set Architecture (ISA)

Prof.ssa Giulia Cisotto

giulia.cisotto@units.it

Trieste, 12 marzo 2025

$$0111111100011010_2 = 7F1A_{16}$$

$$AB2_{16} = 101010110010_2$$

$-4_{10} = \mathbf{1}0000100_2$

$-7_{10} = 01001_2$

$$3.625_{10} = 11.101_2$$

1 10000010 01001100000000000000000 **= -10.375**

01101001 01101110 01100110 01101111 01110010 01101101 01100001 01110100 01101001 01100011 01100001

i	n	f	o	r	m	a	t	i	c	a

X + Y = Z

00000001000010010101000000100000

00000001000010010101000000100000

Sotto assunzione di star condividendo la stessa architettura programmativa e quindi standard di codifica:

questa sequenza di 32 bit è l'istruzione di una SOMMA.

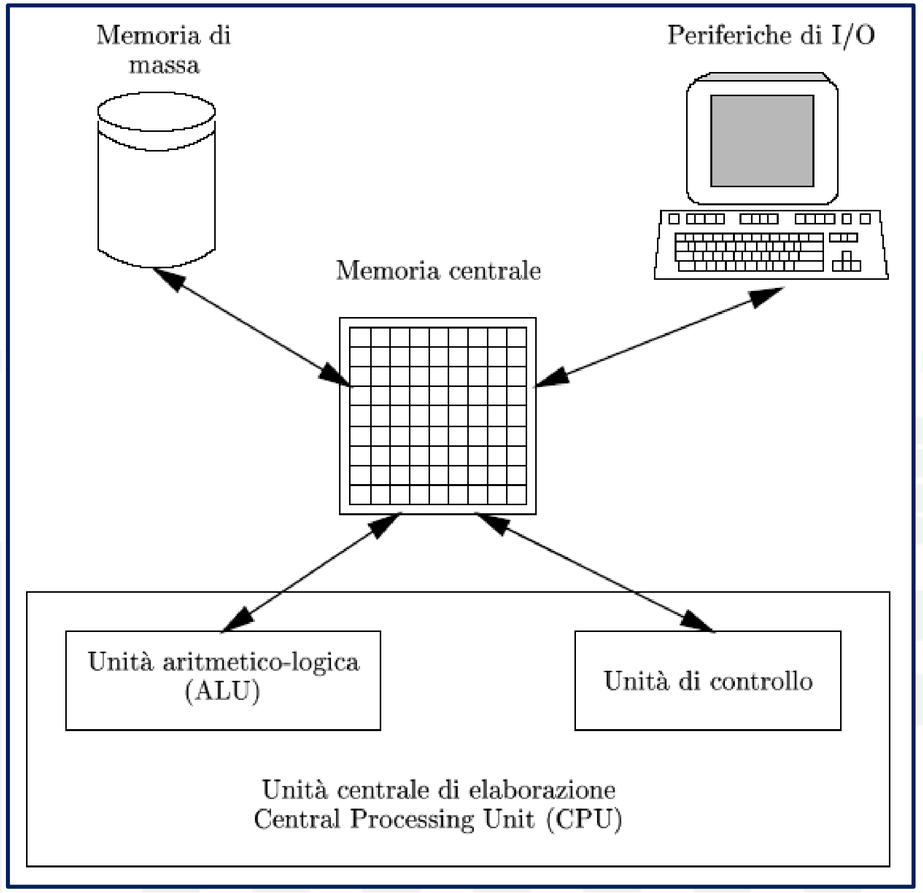
add $10, $8, $9 *Assembly*

X + Y = Z oppure Z = X + Y

Supponendo che X ed Y siano contenuti nei registri 8 e 9 e il risultato vada memorizzato (temporaneamente) nel registro 10

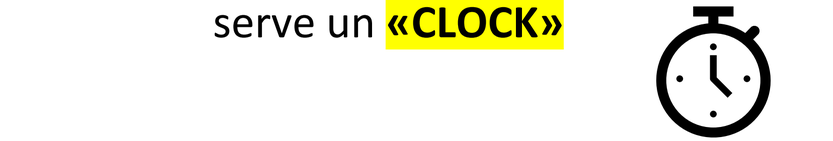
MODULO 1: Architettura degli elaboratori

John Von Neumann
Matematico ed informatico di origine ungherese che viveva e lavorava negli Stati Uniti negli anni ‘40



Architettura di un elaboratore di Von Neumann

Le varie parti dell’architettura devono sincronizzare le proprie attività: serve un **«CLOCK»**



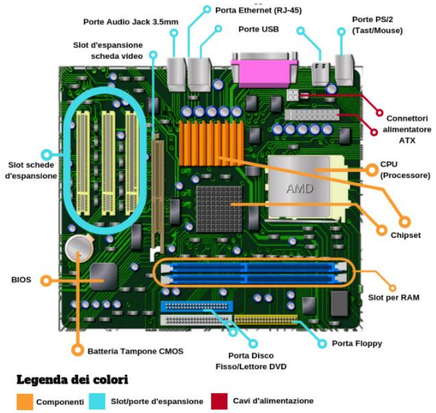
E’ un segnale periodico.

Più è veloce, più attività si possono fare nell’unità di tempo!

1GHz = 10^9 Hz

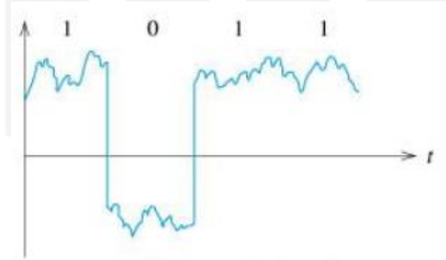
Design dell'«architettura»

ARCHITETTURA: organizzazione «logica» delle sue componenti interne e le modalità secondo le quali queste cooperano tra di loro per compiere azioni più o meno complesse.



HARDWARE: microprocessore, RAM, registri, scheda video, tastiera, mouse, monitor, ecc.

In base al design elettronico di questi elementi hardware, noi abbiamo a disposizione più o meno «memoria» e capacità di gestire sequenze più o meno lunghe di bit.



HARDWARE ⟺ **SOFTWARE**

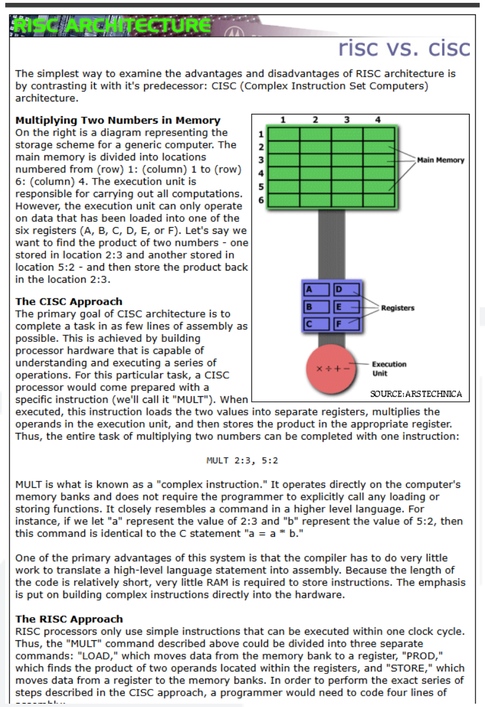
- Ogni linea elettrica di conduzione della corrente (qui integrata nella scheda madre) può «portare» un'informazione 0 oppure 1. Quindi, per avere sequenze di bit… *ho bisogno di più «linee»!*
- La memoria (RAM, registri, cache) altro non è che un circuito elettronico costituito da ***più o meno elementi***, a loro volta composti da ***transistor***, condensatori e altri elementi elettronici.

UNIVERSITÀ DEGLI STUDI DI TRIESTE

FILOSOFIE DI COSTRUZIONE DELL'ARCHITETTURA

RISC (Reduced Instruction Set Computing)
- Poche istruzioni semplici
- Struttura circuitalmente semplice
- Esecuzione veloce di una singola istruzione
- *Occorrono più istruzioni per fare cose anche semplici*
- Esempio: MIPS, ARM, PowerPC…

CISC (Complex Instruction Set Computing)
- Istruzioni complesse
- Struttura circuitalmente complicata
- Esecuzione più lenta di una singola istruzione
- *Occorrono meno istruzioni*
- Esempio: Intel x86 e tutti i derivati (Pentium ecc.)



RISC ARCHITECTURE

risc vs. cisc

The simplest way to examine the advantages and disadvantages of RISC architecture is by contrasting it with it's predecessor: CISC (Complex Instruction Set Computers) architecture.

Multiplying Two Numbers in Memory
On the right is a diagram representing the storage scheme for a generic computer. The main memory is divided into locations numbered from (row) 1: (column) 1 to (row) 6: (column) 4. The execution unit is responsible for carrying out all computations. However, the execution unit can only operate on data that has been loaded into one of the six registers (A, B, C, D, E, or F). Let's say we want to find the product of two numbers - one stored in location 2:3 and another stored in location 5:2 - and then store the product back in the location 2:3.

Main Memory
Registers
Execution Unit
SOURCE: ARSTECHNICA

The CISC Approach
The primary goal of CISC architecture is to complete a task in as few lines of assembly as possible. This is achieved by building processor hardware that is capable of understanding and executing a series of operations. For this particular task, a CISC processor would come prepared with a specific instruction (we'll call it "MULT"). When executed, this instruction loads the two values into separate registers, multiplies the operands in the execution unit, and then stores the product in the appropriate register. Thus, the entire task of multiplying two numbers can be completed with one instruction:

```
MULT 2:3, 5:2
```

MULT is what is known as a "complex instruction." It operates directly on the computer's memory banks and does not require the programmer to explicitly call any loading or storing functions. It closely resembles a command in a higher level language. For instance, if we let "a" represent the value of 2:3 and "b" represent the value of 5:2, then this command is identical to the C statement "a = a * b."

One of the primary advantages of this system is that the compiler has to do very little work to translate a high-level language statement into assembly. Because the length of the code is relatively short, very little RAM is required to store instructions. The emphasis is put on building complex instructions directly into the hardware.

The RISC Approach
RISC processors only use simple instructions that can be executed within one clock cycle. Thus, the "MULT" command described above could be divided into three separate commands: "LOAD," which moves data from the memory bank to a register, "PROD," which finds the product of two operands located within the registers, and "STORE," which moves data from a register to the memory banks. In order to perform the exact series of steps described in the CISC approach, a programmer would need to code four lines of

QtSpim

Nel seguito del corso: MIPS (come esempio!)
- SPIM = simulatore sw di MIPS
- versioni per diversi sistemi (Windows, Mac, Linux)

[Architetture Risc vs cisc]

REGOLE FONDAMENTALI DELL'ARCHITETTURA MIPS32 (RISC)

- 32 registri da 32 bit
- Istruzioni da 32 bit
- Manipolazioni di dati solo sui registri
- Trasferimento di dati tra memoria e registri
- Alterazione del flusso di controllo ("salti")

00000001000010010101000000100000

Problema architetturale:

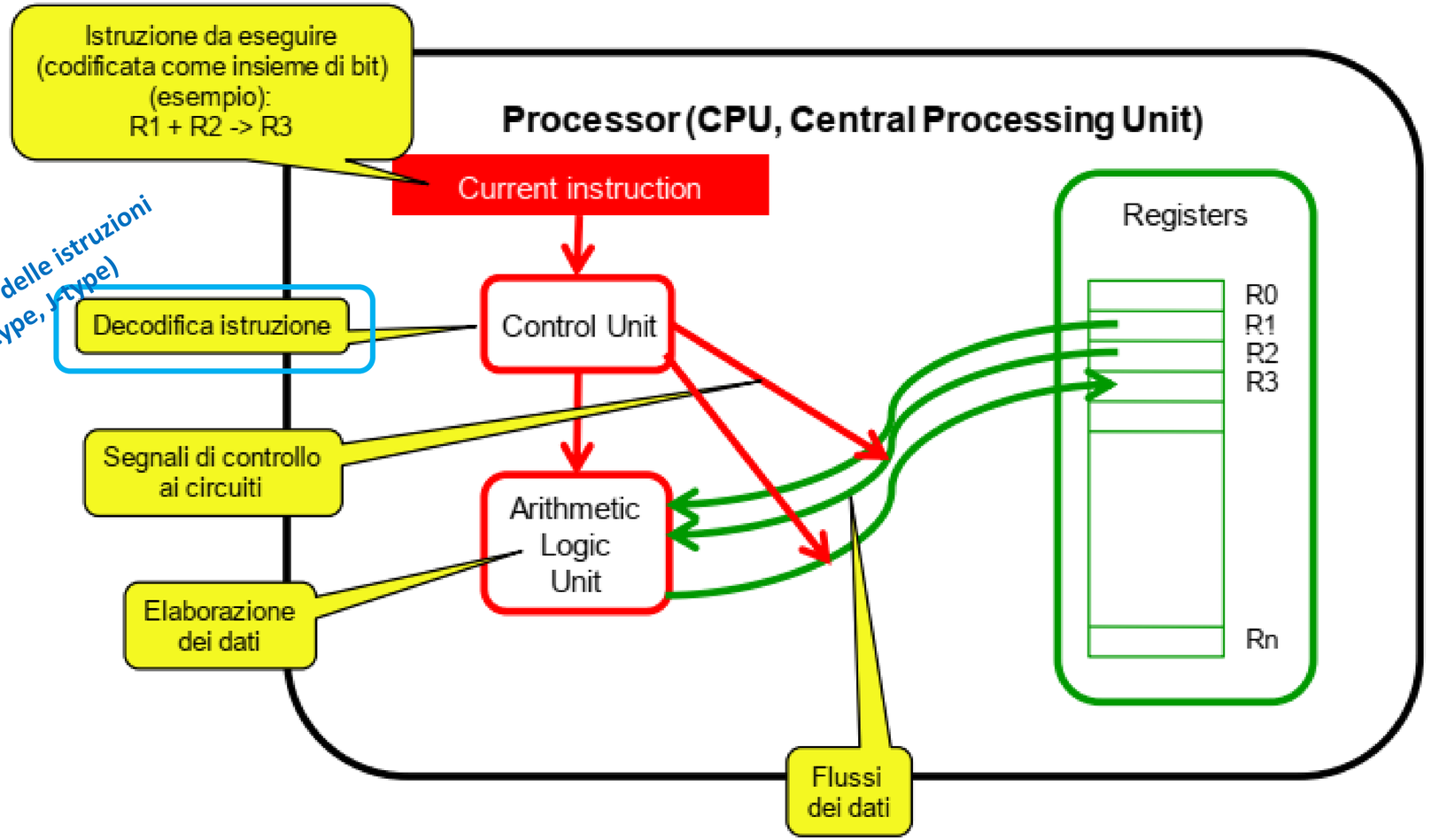
- *come esprimere tutti i tipi di operazioni in 32 bit??* …mantenendo il più possibile omogenea la struttura delle istruzioni

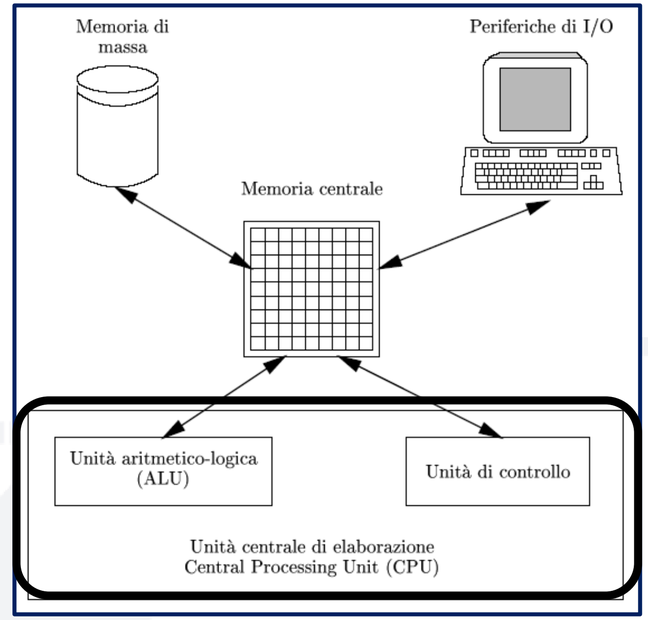
Tre formati delle istruzioni: R-type, I-type, J-type

Attenzione: i tre formati non coincidono esattamente con le tre tipologie di operazioni

ISTRUZIONE → CODIFICA → ESECUZIONE (SU MIPS32)

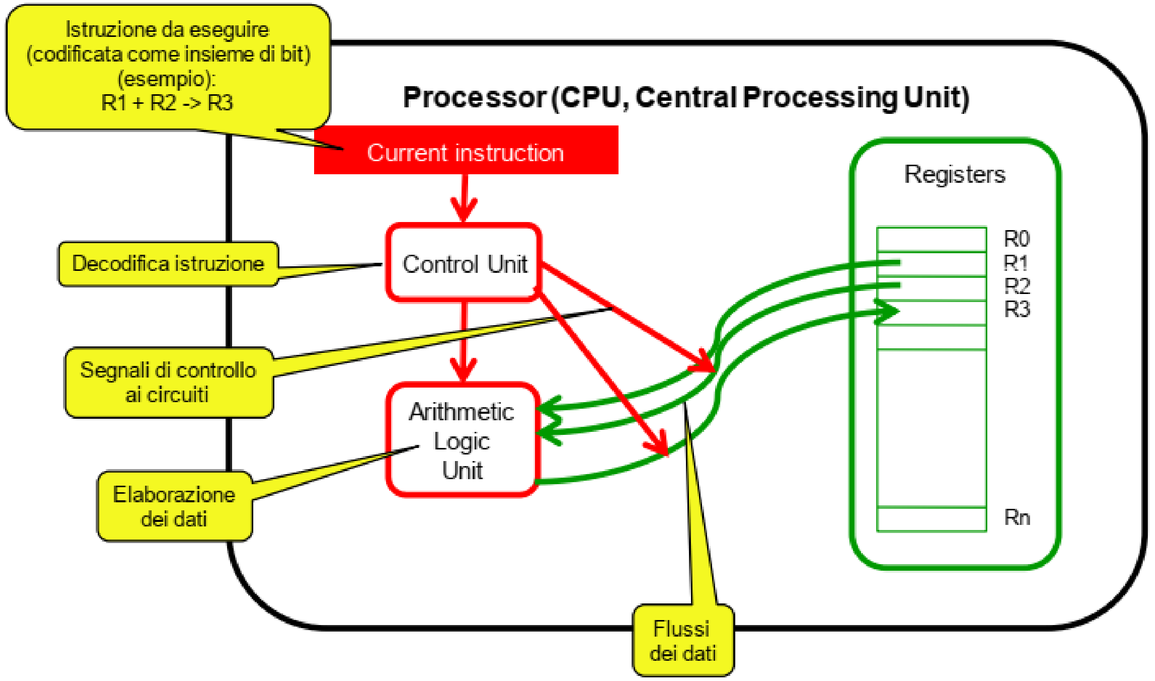
add $10, $8, $9 → 000000 01000 01001 01010 00000 01000 00000



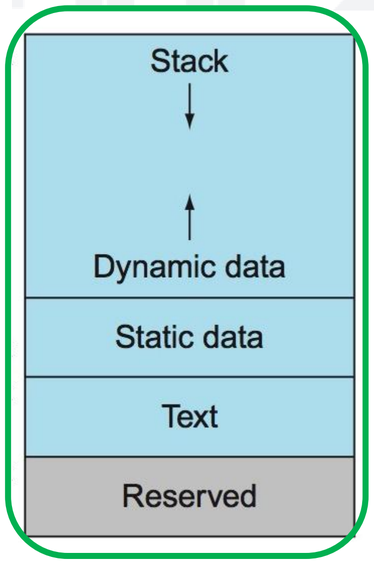


UNIVERSITÀ DEGLI STUDI DI TRIESTE

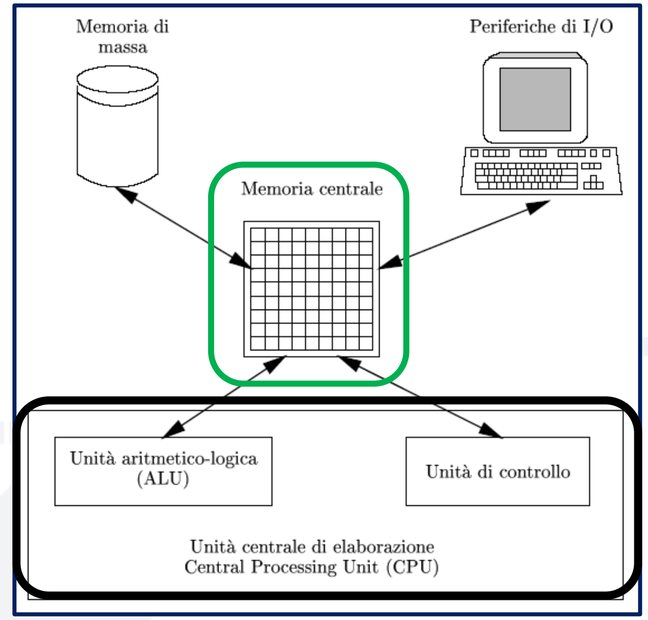
ISTRUZIONE → CODIFICA → ESECUZIONE (SU MIPS32)



Indirizzi codificati
a 32 bit



heap



*Nota: **8 cifre HEX**, ovvero 32bit!*

UNIVERSITÀ DEGLI STUDI DI TRIESTE

Materiale per la lezione

- Appendix A (p.50 per opcode, p.24 per registri)

Prossima lezione: 13 ***marzo, h.9:00****, aula 4C*