Università degli Studi di Milano
Dipartimento di Informatica
Corso di Laurea Triennale in Informatica per la Comunicazione Digitale

Architettura degli Elaboratori

Presentazione del corso

1

Il docente (me!)

- Marco Tarini
- Mi trovate … su google. Oppure:
- marco.tarini@unimi.it
- http://tarini.di.unimi.it
- Ricevimento: Martedì 14:30-17:30 (o chiedere per mail)
- Ufficio: 4to piano,
  (Dipartimento di Informatica, Via Celoria 18)

2

Lezioni e materiale didattico

- Lezioni frontali
- Svolgimento di esercizi in aula

Materiale

- Slides (pubblicate sul sito del corso)
- Testi consigliati ad integrazione delle lezioni:
  - M.Morris Mano, C. R. Kime, Reti logiche, Pearson [prima parte]
  - D.A. Patterson, J.L. Hennessy, Struttura e Progetto dei Calcolatori, Zanichelli [Seconda parte] (cap.2 e cap.4)

3

Modalità d'esame

- Prova scritta basata primariamente su esercizi

4

Obiettivo del corso

- Studio del **processore (CPU)**,
  (il componente centrale del sistema calcolatore!)
- processore = dispositivo che interpreta ed esegue le **istruzioni** di un **programma** scritto in **linguaggio binario** (utilizzando circuiti **hardware**)

- Domande chiave a cui daremo risposta:
  - Come si rappresentano **istruzioni** e **dati** in un programma binario?
  - Che relazione c'è con i programmi scritti in linguaggi come C, Go, Python?

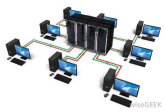
5

Calcolatori

Personal computer: utente singolo, costo limitato

Computer embedded : computer usati per scopi specifici. Prestazioni, costo e consumo energetico limitato.

Computer in dispositivi mobili: utente singolo, alimentati da batteria, connessione wireless / rete cellulare

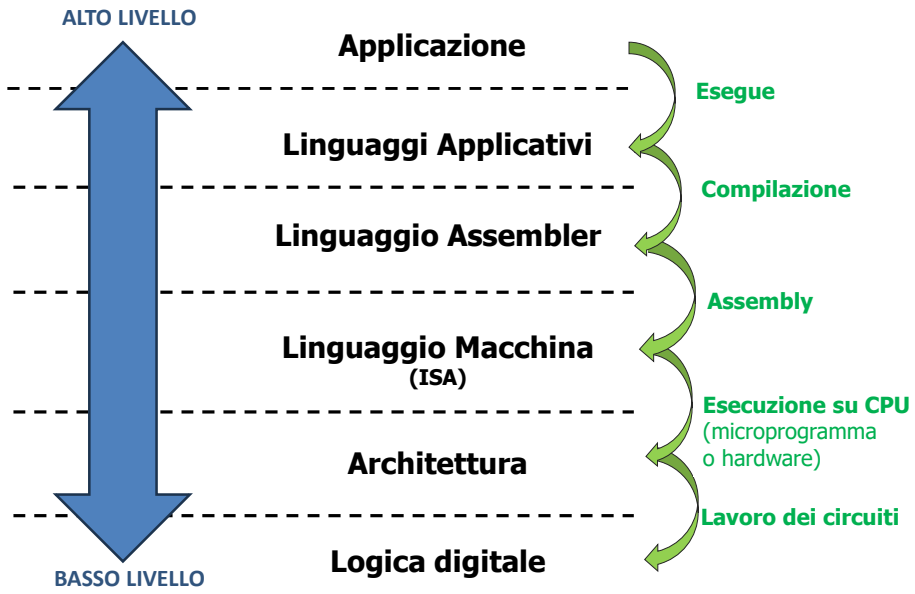


Server: computer normalmente utilizzato attraverso una rete e quindi accessibile da utenti diversi.
(spesso: prestazioni elevate)

Supercomputer: un server di prestazioni estremamente elevate, costituito da migliaia di processori, costo estremamente elevato

7

Livelli di astrazione in un elaboratore



8

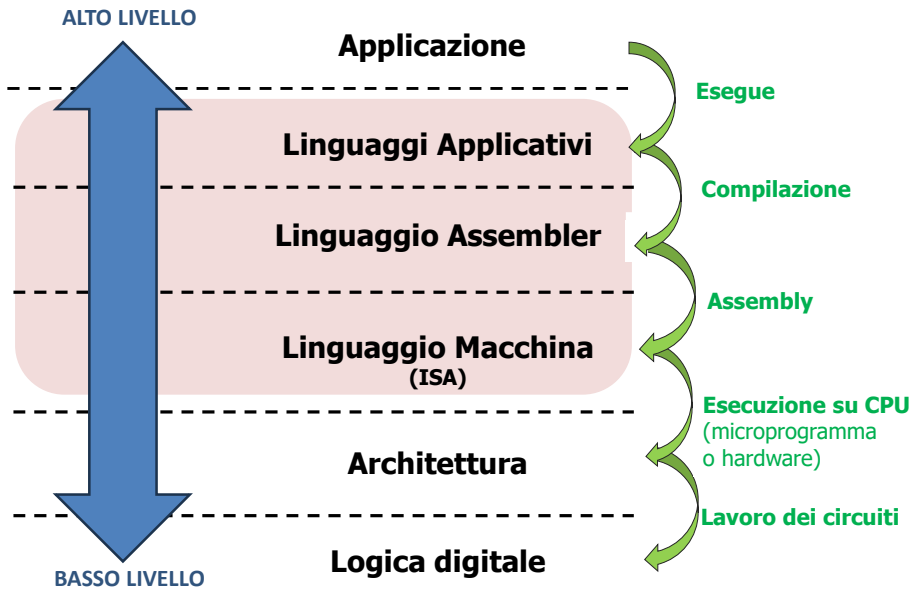
Livelli di astrazione: note

Ciascun livello consiste di:

- Un'interfaccia (verso l'alto)
  - quello che è visibile dall'esterno
  - è usata dal livello superiore
- Un'implementazione (verso il basso)
  - come lavora internamente quell livello
  - usa l'interfaccia del livello inferiore

10

Livelli di astrazione in un elaboratore



11

Compilazione e interpretazione

- Compilazione: traduzione di un programma da un linguaggio (ad alto livello) ad un altro (a più basso livello)
- Differenze:
  - Complilazione: la traduzione avviene per tutto il programma PRIMA dell'esecuzione del programma
  - Interpretazione: la traduzione avviene istruzione per istruzione DURANTE l'esecuzione
- Esempi:
  - complilatore C++ : traduce da C++ ad assembly
  - Interprete BASIC : un programma interpeta un programma scritto in BASIC, eseguendo un'istruzione dopo l'altra

12

Compilazione

Linguaggio applicativo (GO)

```
package main

import "fmt"
func main() {
   fmt.Println("hello world");
}
```

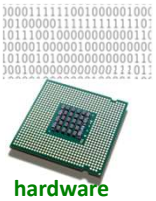
compilatore

Assembler (MIPS)

```
multi $2, $5,4
add   $2, $4,$2
lw    $15, 0($2)
lw    $16, 4($2)
sw    $16, 0($2)
sw    $15, 4($2)
jr    $31
```

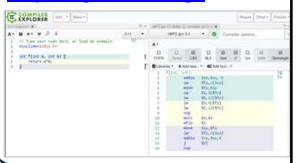
assembler

Linguaggio macchina (MIPS)



hardware

Se vuoi, prova su
https://godbolt.org/



13

Linguaggio Assembly

È la rappresentazione simbolica del linguaggio macchina di un elaboratore.

```
add $t0 $s2 $s3
Binary: 00000010010100110100000000100000
Hex: 0x02534020
```

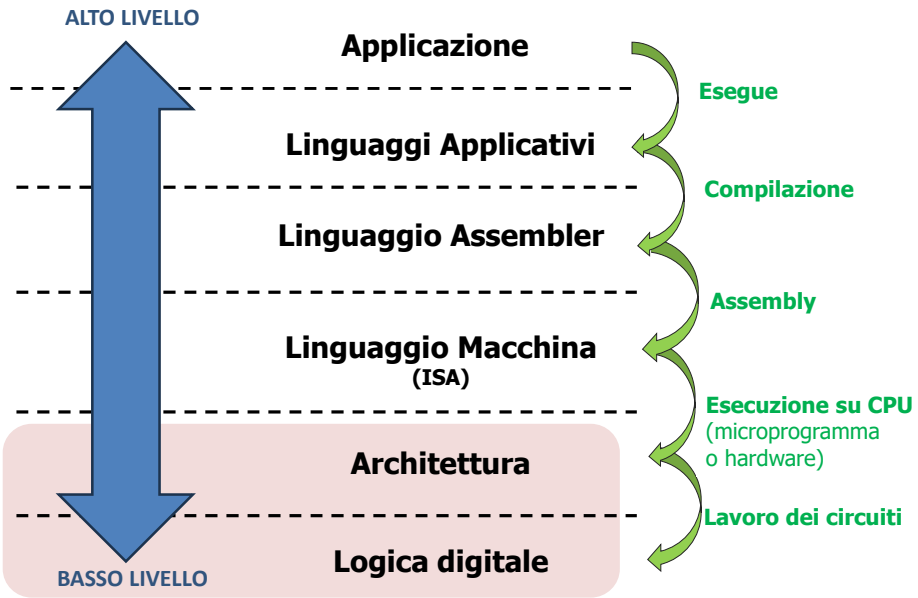
31–26	25–21	20–16	15–11	10–6	5–0
SPECIAL 000000	$s2 10010	$s3 10011	$t0 01000	0 00000	ADD 100000
6	5	5	5	5	6

MIPS instruction converter
https://www.eg.bucknell.edu/~csci320/mips_web/

- Dà alle istruzioni una forma *human-readable* e permette di usare **label** per referenziare con un nome parole di memoria che contengono istruzioni o dati.
- Programmi coinvolti:
  - **assembler**: «traduce» le istruzioni assembly (da un **file sorgente**) nelle corrispondenti istruzioni macchina in formato binario (in un **file oggetto**);
  - **linker**: combina i files oggetto e le librerie in un **file eseguibile** dove la «destinazione» di ogni label è determinata.

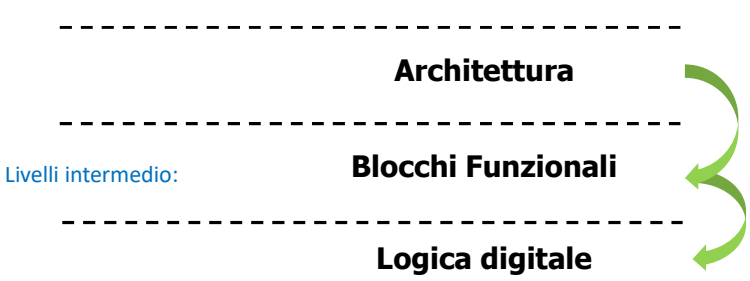
14

Livelli di astrazione in un elaboratore



18

Livelli di astrazione intermedi

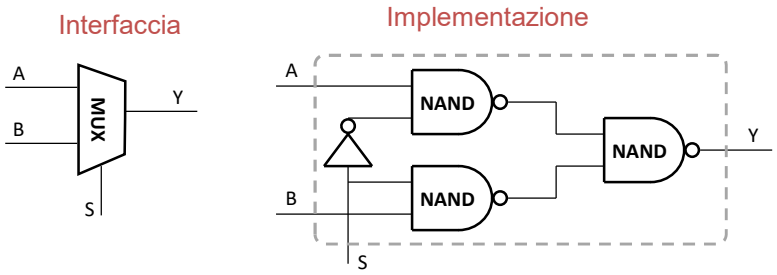


19

Livelli di astrazione intermedi (spoiler)

Esempio:

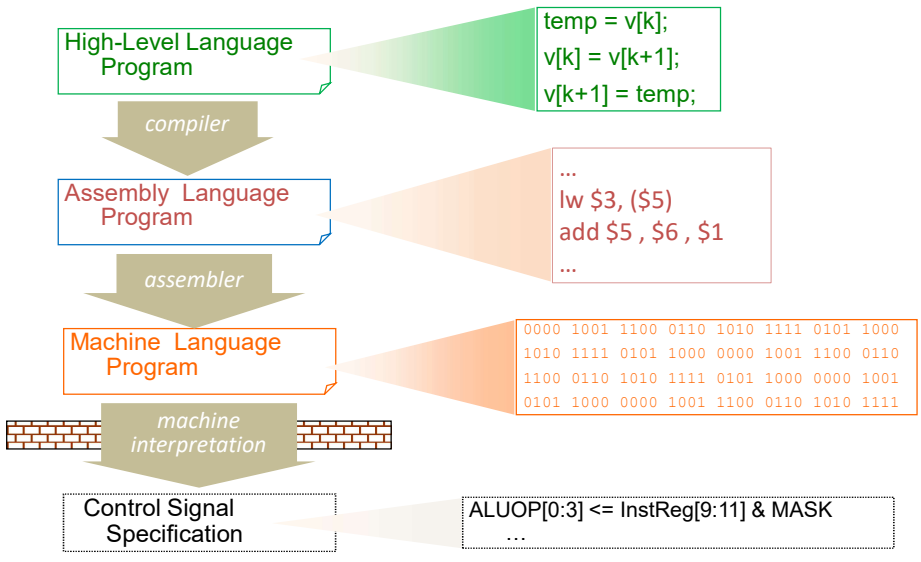
livello dei Blocchi Funzionali (vedremo)



- 20 -

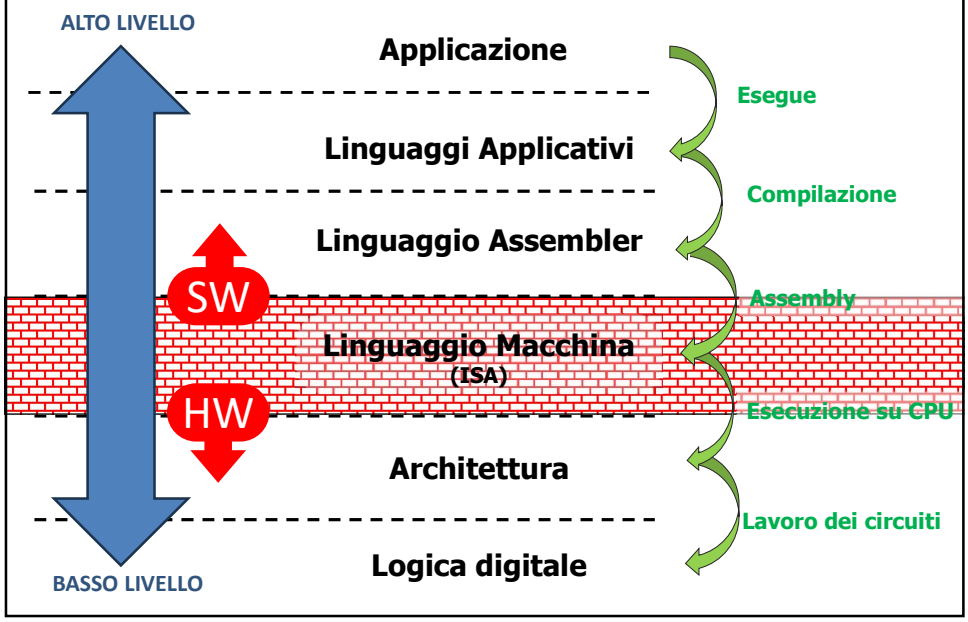
20

Il codice da un livello all'altro



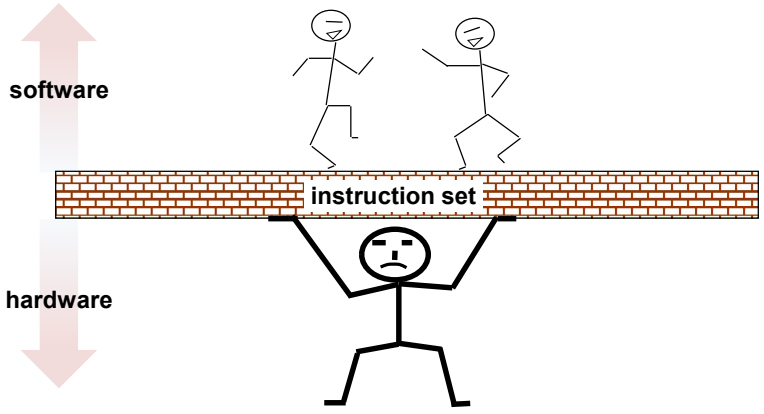
23

Livelli di astrazione in un elaboratore



24

Il set di istruzioni: un'interfaccia critica



25

ISA: Instruction Set Architecture

- Il livello visto dal programmatore assembly o dal compilatore.
- Comprende:
  - **Instruction Set** (quali operazioni possono essere eseguite?)
  - **Instruction Format** (come devono essere scritte queste istruzioni? cioè la loro *sintassi*)
  - **Data Storage** (dove sono posizionati i dati?)
  - **Addressing Mode** (come si accede ai dati?)
  - **Exceptions** (come vengono gestiti i casi eccezionali?)

26

Il set di istruzioni: un'interfaccia critica

- È un difficile compromesso fra:
  - massimizzare le prestazioni
  - massimizzare la semplicità di uso
  - minimizzare i costi di produzione
  - minimizzare i tempi di progettazione
- Definisce la sintassi e la semantica del linguaggio

27

Alcuni Istruction Set popolari oggi

RISC-V
MIPS
ARM
} approccio RISC

X86
X64
} approccio CISC

per i curiosi: cercare sulla Wikipedia…

28

Istruction set: CISC o RISC?

Una distinzione fondamentale è fra processori RISC vs. CISC: due diversi paradigmi di progettazione di un instruction set

- CISC (Complete Instruction Set Computer) - molte istruzioni anche complesse
- RISC (Reduced Instruction Set Computer) – le istruzioni sono poche e semplici. La semplicità si traduce in prestazioni più elevate di ogni istruzione, ma anche nella necessità di eseguire più istruzioni per fare le stesse cose

Primi progetti di ricerca:

- Il progetto Berkeley RISC inizia nel 1980 sotto la direzione di David Patterson
- John L. Hennessy inizia un progetto simile chiamato MIPS alla Stanford University nel 1981. Nel 1985 viene rilasciato il primo prodotto dalla società MIPS Technologies

29

Standard Risc V - https://riscv.org/

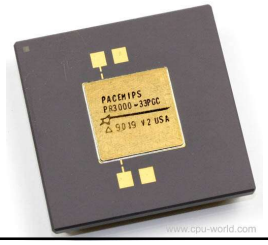
- Progetto per la definizione di una architettura delle istruzioni “aperta”
- può essere utilizzata senza dover pagare una licenza
- progetto avviato a Berkley nel 2010

“RISC-V enables the community to share technical investment, contribute to the strategic future, create more rapidly, enjoy unprecedented design freedom, and substantially reduce the cost of innovation.”

30

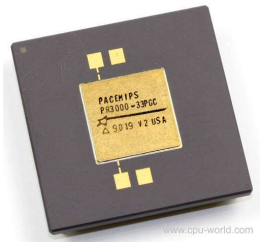
Nostra scelta dell’Instruction Set:

MIPS



31

MIPS



- In questo laboratorio lavoreremo con **MIPS**
- MIPS: Multiprocessor without Interlocked Pipeline Stages → un'Instruction Set Architecture (ISA) di tipo RISC
- Nasce a metà anni '80 come architettura *general purpose;*

- Inizialmente è un progetto accademico (Stanford), poco dopo diventa commerciale
- Oggi è superata, impiegata tuttalpiù nell'ambito dei *sistemi embedded,* ma è adatto alla presentazione dei concetti che sottendono anche gli istruction set più moderni

32

Perché il processore MIPS

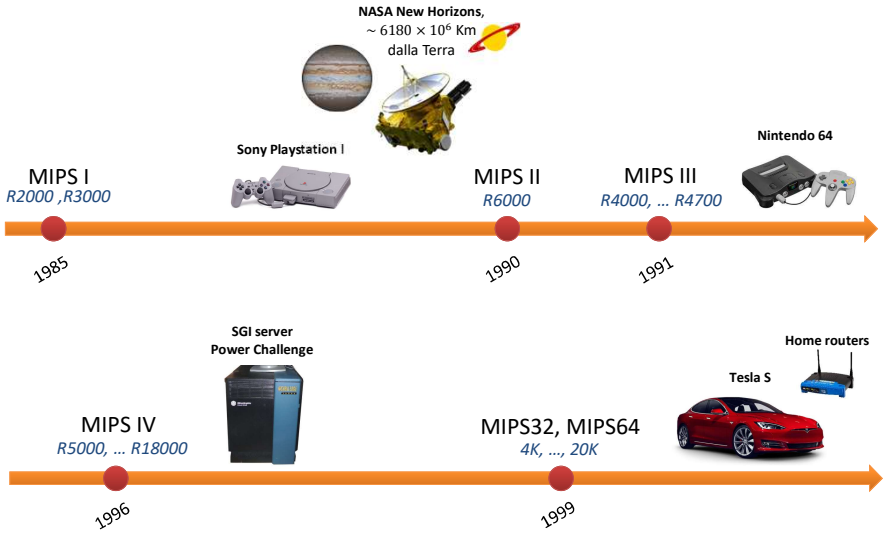
Ampiamente usato per scopi didattici

Patterson e Hennessy ricevono ACM Turing Award nel 2017

"*For pioneering a systematic, quantitative approach to the design and evaluation of computer architectures with enduring impact on thmicroprocessor industry.*"

33

MIPS: passato e presente



34

Programma del corso

Parte 1: Elementi di base

- Rappresentazione dell'informazione
  - Numeri naturali, interi, frazionari.
  - Loro manipolazione
  - Caratteri e stringhe.
- Circuiti logici
  - Algebra di Bool
  - Circuiti combinatori
  - Circuiti sequenziali

Parte 2: Architettura di un elaboratore

- Modello di Von Neumann di un calcolatore
- Il processore (**MIPS**)
  - Istruzioni Assembly e in linguaggio macchina
  - Programmazione in Assembly MIPS
  - Esempio semplice di struttura HW

Programma dettagliato sul sito del corso

37