

Architettura degli elaboratori

Docenti:

- **Lezioni:** Federico Fontana
federico.fontana@uniud.it
- **Arduino:** Andrea Gulli
gulli.andrea@spes.uniud.it
- **Laboratorio:** Andrea Gulli e Federico Fontana.

Risorse online:

- **MS Team:**
2021-ARCHITETTURA DEGLI ELABORATORI
E LABORATORIO [id.130708] - CDS 725
- **E-learning** <https://elearning.uniud.it>:
ArcDegElaELab21_819MA0013.

Obiettivi dell'insegnamento

- Spiegare come sono fatti e come funzionano **fisicamente** i calcolatori

Obiettivi dell'insegnamento

- Spiegare come sono fatti e come funzionano **fisicamente** i calcolatori
- Illustrare le componenti base e le loro interazioni

Obiettivi dell'insegnamento

- Spiegare come sono fatti e come funzionano **fisicamente** i calcolatori
- Illustrare le componenti base e le loro interazioni
- Nella tradizionale distinzione **hardware–software**:
questo è un corso di **hardware**.

Nota sul colore del testo nelle slide:

- in **blu**: definizioni, termini **da ricordare**
- in **rosso**: questioni **molto** importanti.

Diversificazione dell'insegnamento

L'insegnamento è diversificato tra Informatica e Internet of Things, Big Data, Machine Learning (IBM).

- **Informatica**: 12 CFU = 9 CFU di lezione + 3 CFU di laboratorio.
- **IBM**: 6 CFU di lezione.

Motivazione:

- i due corsi di laurea hanno obiettivi differenti.

Organizzazione (tipica) delle lezioni

- Alcuni argomenti solo per Informatica.
Corrispondenti **titoli delle slide in rosso**.
- Il **laboratorio** è svolto solo per Informatica.
- **INF**: 6 ore di lezione (lunedì, martedì e giovedì)
+ 2 ore di laboratorio (lunedì) a settimana
IBM: 4 ore di lezione (lunedì e martedì) a settimana.
- Questa **prima settimana** 6 ore sia INF che IBM.
- Orario di **ricevimento** (meglio previa email):
giovedì 15:30 – 17:30, o quando mi trovate in studio (il venerdì se posso lavoro al CEPO - Sede di Pordenone).

Organizzazione esame

- Solo **scritto**! :(pandemie permettendo):
- INF: 14 domande in 2h45m per 48 punti da riproporzionare in trentesimi + lode.
- IBM: 10 domande in 1h40m per 30 punti da riproporzionare in trentesimi + lode.
- Tipicamente: $\approx 30\%$ domande di **teoria**, $\approx 70\%$ **esercizi** da risolvere (!).
- La serie storica (tre anni e 1/2, 5+5+2+5 esami scritti) è reperibile del sito di e-learning.
- Durante il corso vedremo esercizi simili a quelli dati all'esame, ma **non c'è** una struttura rigida dello scritto (che nel 2021 è stato più univoco).

Laboratorio - solo Informatica

- Quasi tutti i lunedì pomeriggio (non oggi prima lezione). 3 CFU: 24 ore.
- Docente di riferimento: Andrea Gulli.
- Ci sarò anch'io salvo eccezioni.
- Inizio: lunedì 4 ottobre (lezione di avvio: configurazione dell'ambiente software e primi esercizi).
- **Credo** sia necessario registrarsi per poter accedere alle postazioni. Consegnare nella stanza tra Lab1 e Lab2 la **richiesta di accesso**:
`https://www.uniud.it/it/ateneo-uniud/ateneo-uniud-organizzazione/dipartimenti/dmif/assets/moduli`

Contenuti

Corso in parte nozionistico - descrittivo:

- si descrivono le diverse parti del calcolatore e come interagiscono
- vengono presentati principi di funzionamento e concetti base
- spesso non sono dettagliati nè tecnologicamente all'avanguardia
- le nozioni vanno **comprese** in vista dell'esame, **non** memorizzate e **NON** indotte dagli esercizi:
 - capire **prima** i meccanismi alla base del funzionamento delle diverse componenti,
 - costruire **col tempo** una comprensione generale, collegare tra loro le diverse nozioni.

Contenuti

Aspetti concettuali. Es.:

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri
- operazioni su espressioni booleane

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri
- operazioni su espressioni booleane
- macchine a stati finiti (solo Informatica).

Aspetti **progettuali**. Es.:

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri
- operazioni su espressioni booleane
- macchine a stati finiti (solo Informatica).

Aspetti **progettuali**. Es.:

- circuiti combinatori e sequenziali
- organizzazione di processori e memorie

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri
- operazioni su espressioni booleane
- macchine a stati finiti (solo Informatica).

Aspetti **progettuali**. Es.:

- circuiti combinatori e sequenziali
- organizzazione di processori e memorie
- programmazione assembly del processore ARM (solo Informatica).

Contenuti

Aspetti **concettuali**. Es.:

- rappresentazione dei numeri
- operazioni su espressioni booleane
- macchine a stati finiti (solo Informatica).

Aspetti **progettuali**. Es.:

- circuiti combinatori e sequenziali
- organizzazione di processori e memorie
- programmazione assembly del processore ARM (solo Informatica).

Alcuni argomenti progettuali vengono **approfonditi** in laboratorio (solo Informatica).

Non imparate a memoria gli esercizi dati agli scritti e in laboratorio!

Considerazioni generali

- L'università fornisce pochi stimoli a studiare durante l'anno (no interrogazioni, compiti).
- Quest'anno va peggio: libertà di assistere da remoto alle registrazioni della lezione.
- Siate responsabili: in ottobre-dicembre studio **per capire**, in gennaio per fissare nella mente.
- Tra le valutazioni a Fontana: "Fare più esercizi!" Risultato: esiti disastrosi appena invento un esercizio nuovo. Nessun beneficio apparente.
- Ancora: "Non si esprime in modo chiaro!" Segnalatemi se sto andando troppo veloce. Segnalatemi se una slide non racchiude efficacemente il concetto affrontato.

Qualche consiglio pratico

- Seguire le lezioni è il miglior modo per confrontarsi con la materia.
- Ripassare di volta in volta gli argomenti trattati a lezione: **che cosa non ho capito?**
- Non riesco più a seguire agevolmente le lezioni: **Sto ripassando a sufficienza gli argomenti trattati?**
- Da parte mia cercherò di mettere i lucidi nel web e-learning **prima** di ogni lezione.
- Per quanto detto, **non** porrò l'accento sugli esercizi.
- La lezione simultanea in presenza & da remoto è **complessa** da tenere anche per il docente.

Ulteriori (sempre gli stessi) consigli

- **Darsi tempo per la comprensione iniziale:** le due settimane precedenti l'esame servono solo a memorizzare i concetti.
- Non perdere il contatto con ciò che viene presentato a lezione.
- Le registrazioni delle lezioni sono più brutte e noiose rispetto alle puntate di un serial TV.
- Per Informatica: seguire le lezioni di laboratorio.
- Per tutti: non lasciare slide incomprese (!)
- Sfruttate l'orario di ricevimento e, in generale, i servizi di tutorato. Nessuna domanda è stupida. **Fate domande al docente!** (anche se rispondere efficacemente dal vivo & alla chat è complicato).

Libro di testo

A. S. Tanenbaum, T. Austin *Architettura dei calcolatori - Un approccio strutturale*. Pearson, 6 Ed.

È possibile utilizzare anche le edizioni precedenti del libro di testo, solo poche parti sono state aggiornate nella nuova edizione.

- L'autore è un autorità nella didattica dell'informatica e nel campo dei sistemi operativi. Insegnante di Linus Torvalds e autore di MINIX, un precursore di Linux.
- Il corso segue abbastanza da vicino il libro di testo.
- Non così aggiornato sulle tecnologie hardware in uso oggi (non è uno scopo del corso).

Libro di testo

- Molto discorsivo. Gli argomenti tecnici e quantitativi vengono “aggirati”.
- Ordine di presentazione degli argomenti un po’ controintuitivo:
 - capitolo 2: descrizione superficiale del calcolatore, riassunto di tutti gli argomenti;
 - capitoli successivi: descrizione dettagliata, spesso di aspetti che non approfondiremo (es.: dettagli sulla tecnologia dei chip Intel).

Slide del corso

Slide:

- presentazione delle lezioni
- più sintetiche del libro
- alcuni dettagli non sono presenti sul testo
- esercizi svolti
- testi d'esame (da circa metà novembre)
- comunicazioni dei docenti.

È **molto** utile, sostanzialmente **sufficiente** seguire la presentazione delle slide a lezione, integrando quando necessario con appunti.

Occorre iscriversi all'insegnamento specifico nel portale dell'e-learning!

Azzeramento dei concetti

Architettura di calcolo: macchina capace di eseguire una sequenza di istruzioni semplici (**istruzioni macchina**).

Programma: nel nostro corso, **sequenza** di istruzioni macchina eseguibili anche più volte.

I programmi, e i **dati** su cui il programma lavora risiedono nella **memoria principale** del calcolatore.

La memoria principale tipicamente si svuota quando l'architettura è spenta per inattività. In tal caso i programmi e i dati sono memorizzati in strutture chiamate *file*, custoditi nella **memoria di massa**.

Processore

Il cuore dell'architettura è il **processore** o **central processing unit (CPU)**, la componente che fisicamente esegue le istruzioni contenendo le risorse per eseguire calcoli e controllarne la corretta esecuzione.

Processore e memoria principale vengono realizzati adoperando particolari circuiti chiamati **circuiti integrati**, fisicamente contenuti in uno o più **chip**. Un chip ha la dimensione di pochi centimetri e può contenere miliardi di *transistor*, alla base della realizzazione del circuito. Un solo chip è sufficiente per realizzare un processore o una memoria.

Memoria

La memoria del calcolatore è, abbiamo visto, di **due tipi**:

- principale (**memoria RAM**, parte integrante dell'architettura): contiene i dati durante la loro elaborazione da parte del processore. É tipicamente realizzata dentro un chip
- di massa (**memoria ROM**, disco rigido): contiene i dati non in elaborazione, in modo permanente. I *file* sono etichettati in modo da permettere un accesso strutturato, non ambiguo ai dati. Non è necessariamente integrata in un chip a bordo dell'architettura.

Capacità di memoria

La memoria si realizza in modo economico giustapponendo elementi di **un bit**, ciascuno in grado di assumere **2 configurazioni** stabili.

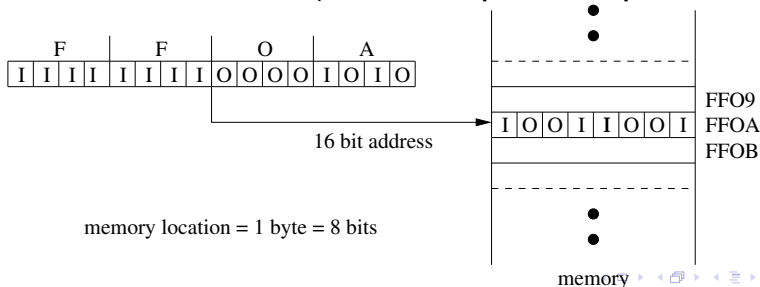
Il **byte** è una giustapposizione di **8 bit**. Il byte quindi può assumere **$2^8 = 256$** configurazioni distinte. Es.: memorizzare un numero naturale tra 0 e 255, oppure 256 codifiche di carattere (a, b, c, ...), eccetera.

La **capacità** della memoria si misura in byte, o nei suoi multipli:

- **KB** (kilobyte = $2^{10} = 1024 \sim 1.000 = 10^3$ byte)
- **MB** (megabyte = $2^{20} \sim 1.000.000 = 10^6$ byte)
- **GB** (gigabyte = $2^{30} \sim 1.000.000.000 = 10^9$ byte)
-

Etichettatura di locazioni di memoria

Perchè è importante definire KB, MB, GB ancora come potenze di 2? L'accesso ottimale alla memoria si ottiene ancora oggi indirizzando ogni singolo byte. In un'architettura a bit quindi l'indirizzamento più efficiente è quello che **adoopera n bit per indirizzare 2^n byte**. Es.: indirizzamento a 16 bit della **locazione** di indirizzo FFOA (vedremo perchè questa sigla)



Periferiche. Bus. Computer

Oltre che con la memoria, il processore riceve e trasmette dati da e verso **periferiche**, interfacce per la comunicazione non indispensabili all'architettura ma che permettono di trarne vantaggio concreto: schermo, tastiera, stampante, scheda di rete, scanner, lettore CD, scheda audio, . . .

Processore, memoria e periferiche sono collegati tra loro attraverso dei **bus**. L'insieme di tutte queste componenti forma il moderno **computer**.

Sistema operativo e applicazioni

Il **sistema operativo** è un particolare programma, il quale permette un razionale utilizzo complessivo del processore, della memoria e delle periferiche. Il sistema operativo è permanentemente in esecuzione e distribuisce le risorse del computer a tutti gli altri programmi.

Tutti i **programmi applicativi** sono sotto il controllo del sistema operativo durante la loro esecuzione, di modo che non utilizzino le risorse in maniera globalmente inefficiente. Es.: un web browser e un word processor sono eseguibili in contemporanea in un computer grazie alla supervisione della loro esecuzione da parte del sistema operativo.

Complessità delle attuali architetture

- Il computer è diventato troppo complesso per poterlo conoscere completamente.
- Un **sistema di calcolo** (computer + programmi + dati) è strutturato in maniera tale da permettere di interagire con parti di esso conoscendo solo alcune funzionalità, ignorando come queste sono realizzate e come sfruttano il sistema.
- Lo studio e la progettazione di sistemi di calcolo oggi sono affrontati a una molteplicità di livelli.

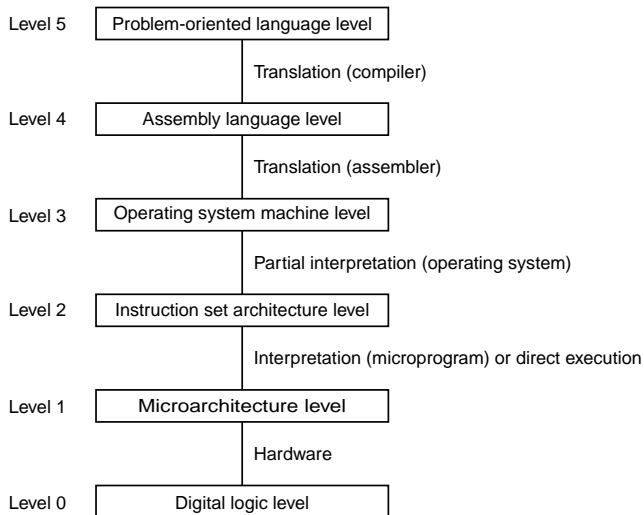
Macchine virtuali

Concetto fondamentale nell'informatica moderna, necessario per gestire l'enorme complessità di un sistema di calcolo.

Il computer viene visto come una stratificazione di **livelli**. Ciascuno strato fornisce delle funzionalità di calcolo e quindi, **virtualmente**, definisce una particolare **macchina**.

Questo paradigma permette di progettare programmi che, appoggiandosi a una **macchina virtuale** sottostante ben specificata, forniscono una serie di funzionalità ulteriori. In tal modo essi formano un livello superiore, a sua volta utilizzabile per strutturare nuove funzionalità.

Esempio di schema a livelli



Livello hardware

- I livelli 0 e (talvolta) 1 formano il livello **hardware**, cioè il **livello fisico** dell'architettura di calcolo.
- La virtualizzazione **non** crea risorse più potenti di quelle messe a disposizione dall'hardware, limitandosi a facilitarne l'uso (es.: processori virtuali, memoria virtuale, ...).
- Conoscere il livello hardware è quindi necessario per comprendere le risorse di cui un programma (il **software**) in ultima analisi dispone. La comprensione è necessaria per
 - valutare, scegliere o gestire l'hardware,
 - conoscere i fattori che determinano le prestazioni e i limiti invalicabili di un determinato computer,
 - gestire i malfunzionamenti.

Programma del corso

- aritmetica booleana

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo
- programmazione assembly

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo
- programmazione assembly
- periferiche e interruzione della CPU

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo
- programmazione assembly
- periferiche e interruzione della CPU
- bus, comunicazione con la CPU

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo
- programmazione assembly
- periferiche e interruzione della CPU
- bus, comunicazione con la CPU
- organizzazione della memoria

Programma del corso

- aritmetica booleana
- porte logiche, circuiti logici
- memoria, circuiti sequenziali, macchine a stati
- processore, unità logico-aritmetica, controllo
- programmazione assembly
- periferiche e interruzione della CPU
- bus, comunicazione con la CPU
- organizzazione della memoria
- (sistemi multiprocessore, calcolatori paralleli).

Storia dei sistemi di calcolo

- 100 a.C.? Meccanismo di Anticitera



Storia dei sistemi di calcolo

- 1642 Pascal
Calcolatrice meccanica per somma e sottrazione



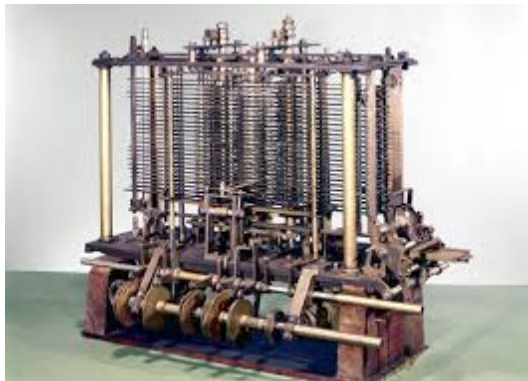
Storia dei sistemi di calcolo

- 1670 Leibniz
Calcolatrice meccanica per prodotto e divisione



Storia dei sistemi di calcolo

- 1834 Babbage
Calcolatrice meccanica programmabile



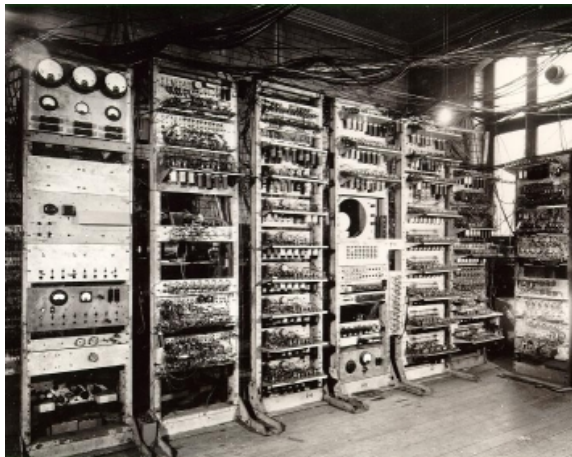
Storia dei sistemi di calcolo

- 1930 Zuse
Macchina calcolatrice a relè



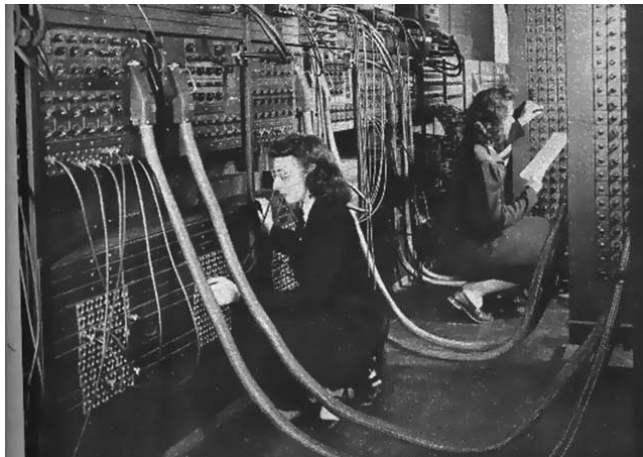
Storia dei sistemi di calcolo

- 1944 Aiken: Mark I
Macchina programmabile a relè



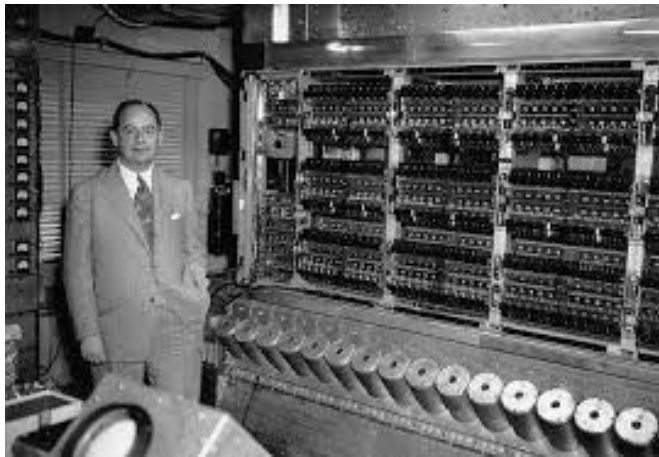
Storia dei sistemi di calcolo

- 1946 Eckert & Mauchly: ENIAC
Primo calcolatore a triodi (valvole)



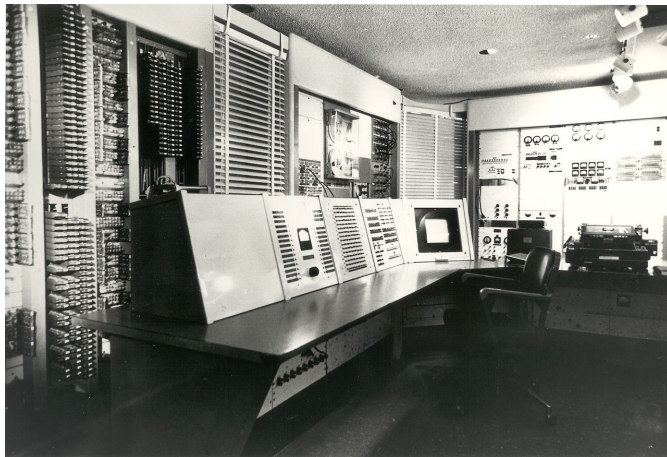
Storia dei sistemi di calcolo

- 1951 Von Neumann: IAS
Prima architettura simile agli attuali computer



Storia dei sistemi di calcolo

- 1956 TX-0
Primo calcolatore interamente a transistor



Storia dei sistemi di calcolo

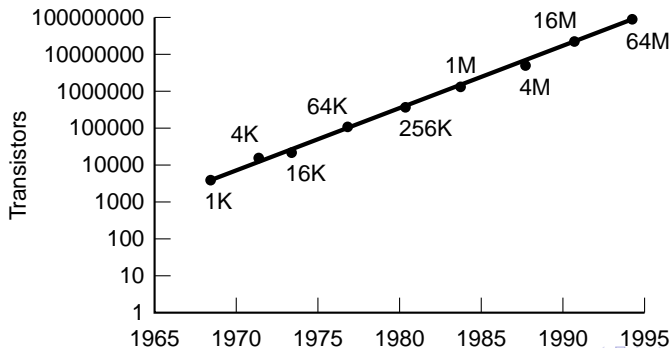
- 1954 IBM — Primi calcolatori a transistor
- 1958 — Calcolatori a circuiti integrati.



Progressi dell'integrazione

Legge di Moore (Gordon Moore, 1965): il numero dei transistor all'interno di un chip (circuito integrato) di memoria quadruplica ogni tre anni.

Dopo 18 mesi? Raddoppia, **non perchè $2 + 2 = 4$** , bensì perchè **$2 \cdot 2 = 4$!**



Numero di transistor per processore

Moore's Law

