

# Predavanje 1 (20. 1. 2026)

## Uvod

- Arhitektura rac. == racunalnik z vidika programerja
- Organizacija rac. == dejanska zgradba, sestavni deli, povezave (t.i. mikroarhitektura)
- Ista arhitektura -> vec organizacij

## Kaj je rac. arhitektura?

- Ta doloca *nivoje abstrakcije*:
  1. Aplikacija
  2. Algoritem
  3. Programski jezik
  4. Zbirni jezik
  5. Ukazna arhitektura
  6. Mikroarhitektura
  7. RTL (registri?)
  8. Logicna vrata
  9. Naprave (tranzistorji)
  10. Fizika

## Aplikacija

- Te zahtevajo izboljšanje arhitekture
- Le tehnologija omejuje ucinkovitost

## Zakaj se to sploh uciti?

- Poznavanje delovanja sistema -> boljsa gradnja aplikacij
  - gradnja je cenejsa, hitrejsa, ucinkovitejsa

## Razlogi za strojno racunanje

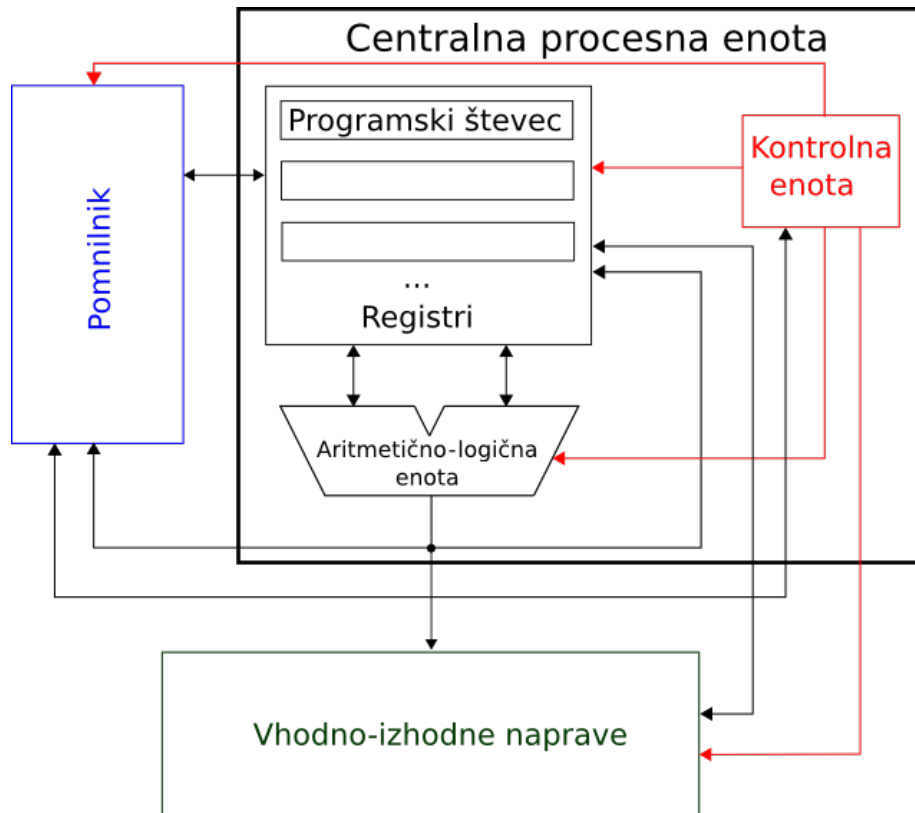
- Rocno racunanje je pocasno in nezanesljivo

## Povezava med strojnim in racnim racunanjem:

- Rocno racunanje:
  - papir -> pomnilnik
  - mozgani -> procesor
- Papir:
  - ukazi/algoritem -> navodila (ukazni pomnilnik)
  - operandi -> npr. stevila (operandni pomnilnik)
- Mozgani (2 funkciji)

- kontrolna fun. (kontrolna enota)
- izvršilna fun. (aritmetična-logična enota)

### Dejanska arhitektura



- Glavni pomnilnik shranjuje tako ukaze kot operande

### Zgodovina

#### Digitalni princip

- Digit == prst (en)
- **Omejeno** stevilo stanj

#### Analogni princip

- Fizikalna velicina zvezno predstavlja vrednosti/stevila
- Omejena natančnost, ampak **neomejeno** st. stanj
- Analognih računalnikov danes (praktično) ni več
  - Lahko pride do veliko napak (npr. zaradi kaksnega suma)
- Primer: logaritmicno računalno (t.i. Rechenschieber)

### Kako dobimo različna stanja (npr. 0 in 1)

- Lahko z napetostno, npr.:
  - od 3.5V do 5.0V je 1
  - od 1.5V do 3.0V je 0
  - (večja območja zaradi suma)
  - Vecje stevilo stanj dobimo z *vzemanjem vec bitov!!!* -> 4 biti == 16 stanj itd.
- Problem z analogno -> ce je *najmanjsi* sum, je ze **drugo stanje**

### Diferencni stroj (Charles Babbage)

- Stroj je podoben danasnjim
- Lastnosti:
  - Aproksimira funkcije s polinomi (končne difference)
  - zaporedje fiksni operacij
  - speciliziran stroj -> ni splošen

### Analitični stroj (isto)

- Prvi računalnik (le nacrtovan)
- Sestava:
  - Mlin -> CPE
  - Pomnilnik -> komunicira z luknjanimi karticami (s tam dobi podatke)
  - Program:
    - \* Ukazne kartice (posljejo ukaze mlinu)
    - \* Operandne kartice (posljejo operande pomnilniku)

### Elektromehanski stroji

- Rele (Relay)
  - elektricno-krmiljeno stikalo
  - ce je tok -> zaprto, druge odprto
  - deluje na principu magnetne sile (indukcija?)

### Prvi delujoč računalnik

- Sestavljen **Konrad Zuse**
- Bili so Z1, Z2, Z3, Z4 (najnovejši/najnaprednejši)
- Lastnosti:
  - veliko relejev
  - 8-bitni ukazi
  - luknjan trak
  - plavajoca vejica
  - tipkovnica
  - frekvenca == 5-10 Hz

## Harvard Mark I

- IBM
- Dolg (15 m)
- elektromehanska desetiska stevna kolesa
- luknjan trak
- pomnilnik 72 x 23 desetiskih mest
- Oblika ukaza -> A1 A2 OP

## Vpliv Babbagea in vpliv modela

- Vsi ti stroji so nastali z zamisli Charlesa
- Problem je bil mehanika, ki prinasa omejitve:
  - hitrost (giblivi deli)
  - zanesljivost (zobniki, vzvodi -> zgubi natančnost)
- V tem času so začeli razvijati elektroniko, ki reši te probleme:
  - **Elektronke**
  - **Tranzistorji** (boljši)

## Prvi elektronski računalniki

- Elektronika je bistveno hitrejša od mehanike -> zato so se na to bolj usredotočili
- Prvi je bil ENIAC (programabilen, vendar ročno -> pretikanje stikal)
- Lastnosti:
  - velik
  - drag (veliko elektrike, veliko elektronk)
  - težak
- Programiranje je trajalo več dni -> Von Neumann začel razmišljati o novem modelu

## (Elektronski) računalnik s shranjenim programom

- Von Neumann predlagal EDVAC:
  - Voden od znotraj -> notri je shranjen jezik
- Prednosti:
  - Ista hitrost za dostop do ukazov in operandov
  - Izhod enega programa lahko uporabimo kot vhod drugega

## Razvoj po letu 1950

- Serijska proizvodnja -> nizja cena
- Uporaba ni več samo za numerične probleme (urejevanje tekst, igre, banke itd.)

## Razvoj tehnologije

1. Tranzistorji

## 2. Integrirani cipi

### Moorov zakon

- Pravi, da se na ~18 mesecev podvoji stevilo tranzistorjev na cipih, ampak uporabijo enako kolicino energije
- Vcasih je veljal (70., 80. leta), zdaj ne vec toliko

### Dennardovo skaliranje

- Z zmanjsevanjem dimenzij tranzistorjev ostane poraba energije na površino konstantna
  - napetost in tok se *manjšata*, a tranzistorjev je *vec*
- Tok odtekanja je manjši (manj elektrike z njih gre stran?) -> **pregrevanje**

### Vgrajeni sistemi (angl. Embedded Systems)

- Sistemi, ki so vgrajeni v nekaj in namenski za to stvar
- To so npr.:
  - telefoni, kamere
  - konzole, igrace
  - (avtonomna) vozila
  - dvigalo

### Razvoj programiranja

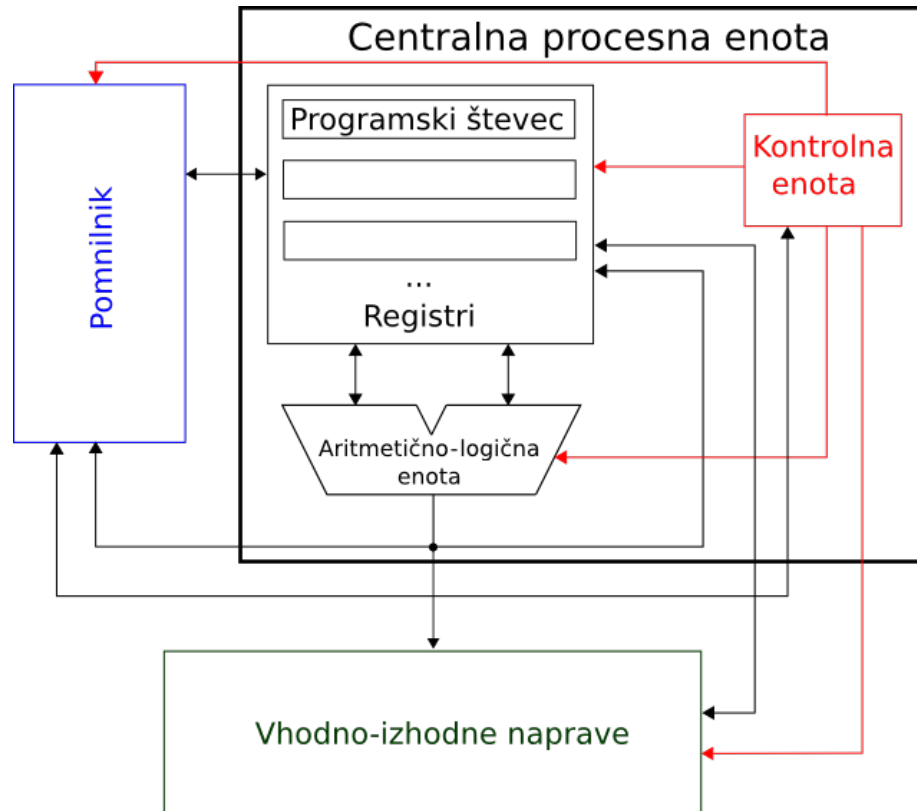
- Nekdanjih orodij ni bilo (pisali 0 in 1)
- Nalagali program iz zunanjega v glavni pomnilnik (v 50. letih)
  - bootstrap-anje ali bootloader omogoča zagon dejanskega OS
- Simbolni zapis
  - Zbirni jezik (Assembly language)
  - Zbirnik (Assembler) -> pretvarja zbirnik jezik v strojni (0 in 1)
- V 60ih se pojavijo visji programski jeziki:
  - prvi: FORTRAN, ALGOL, COBOL, Lisp, ...
  - kasneje: C, Pascal, C++, Java, Python, ...
  - Primerjava:
    - \* zbirni jezik -> hitreje izvaja, vendar počasneje pise
- IBM -> IBM 360 ISA == prvi prenosljiv ukazni nabor

### Racunalska zgradba

#### Von Neumannov racunalniski model

- Sestavljen iz:

- centralna procesna enota (CPE)
- glavni pomnilnik (GP)
- vhodno/izhodni sistem
- Program ima shranjen v GP
- CPE jemlje ukaze iz GP in jih izvršuje zaporedoma
- (Slika od prej)



- Povezava med deli se imenuje most, bridge ali chipset

### Glavni deli modela

1. CPE oz. procesor
  - mikroprocesor
  - vodi dogajanje v rac.
  - jemlje ukaze iz GP in jih izvršuje
  - tri sestavni deli
  - **kontrolna enota**:
    - prevzema ukaze in operacije
    - aktivira operacije
  - **aritmetično-logična enota** (ALE) -> izvršuje ukaze

- **registri** zacasno shranjujejo podatke
2. **Glavni pomnilnik**
- v njem shranjeni ukazi in operandi
  - sestavljajo ga pomnilniske besede (vsaka s svojim naslovom)
  - tehnologija DRAM
3. **Vhodno/izhodno sistem** (angl. I/O)
- prenos informacij iz zunanosti
  - fizicno najvidnejši del racunalnika (*periferne* naprave)
  - prevarjajo informacijo iz CPE v cloveku-razumljivo oz. za drugo napravo
  - (tudi zunanji pomnilniki)

## Ukaz

- Shranjen v eni ali vec pomnilniskih besedah
- Vsebuje:
  - **Operacijsko kodo** (katera operacija)
  - **Informacija o operandih** (reg., naslovi, konstante, ...)
- Primer:
  - Imamo 8-bitno pomnilnisko besedo, ukaz pa je 32-bitni, rabimo **4 besede**
  - Te besede morajo biti *sosednje*
- 2 koraka pri izvrsevanju ukaza:
  1. **Prevzem iz pomnilnika (fetch)**
    - ukazi strojnega jezika ali strojni ukazi
    - bere s tam, na katerega kaze programski stevec (PC)
  2. **Izvrsevanje ukaza**
    - ukaz vsebuje operacijo in operande
    - CPE (obicajno ALE) izvrsi ukaz
    - PC nastavi na naslednji ukaz ( $PC = PC + (\text{dolžino ukaza} / \text{dolžina pom. besede})$ )
      - \* Posebnost pri skocnih ukazih (PC drugi naslov kot naslednjega)

## Prekinitve

- Izjeme, ki ustavijo obicajno delovanje (prejsna dva koraka)
  - To so **prekinitve** (interrupt) in **pasti** (trap)
- Ko pride do tega, skocimo na prvi ukaz **prekinitvenega servisnega programa** (PSP)
  - Pred tem se shrani vrednost PC (da se lahko vrnemo)

## Glavni pomnilnik

- Shranjuje ukaze in operande
- Je pasiven
- Za zmogljivost je vazna hitrost prenos informacij med CPE in GP
  - Ozko grlo Von Neumannovega rac.
  - Par resitev
  - Harvardska arh. (locen pomnilnik za ukaze in operande)
  - Obicjana / Princetonska arh. (skupen pomnilnik)
  - Zdaj se uporablja drugacna:
    - \* Skupen pomnilnik ampak locena **predpomnilnika** (za shranjevanje najpogostejših ukazov in operandov)

## Pomnilniska beseda