Uvod

Algoritam

Algoritam je precizno definisan postupak za rešavanje nekog problema

Računarstvo je nauka o algoritmima:

- njihovim **formalnim osobinama**
  - tačnost, ograničenja, efikasnost, cena
- njihovim **hardverskim realizacijama**
  - projektovanje računarskih sistema
- njihovim **jezičkim realizacijama**
  - programiranje i programski jezici
- njihovim **primenama**
  - inženjerstvo, finansije, medicina, fizika, biologija, hemija...

Pojam arhitekture računara

Program je:

- **algoritam opisan jezikom računara**
- **niz instrukcija koje obavljaju određeni posao**

Računar je **mašina** koja **izvršava zadate instrukcije**

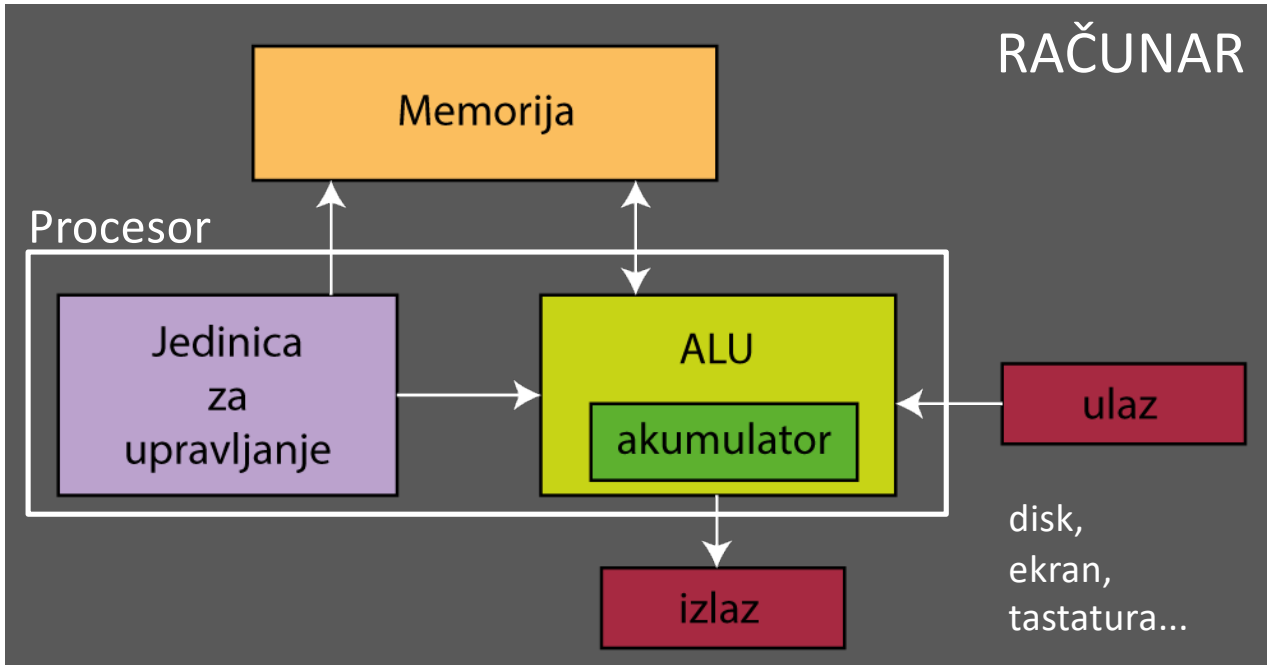
Jezik računara (mašine) je **skup** jednostavnih **instrukcija** koje mogu da izvrše elektronska kola računara (procesor)

Dizajn skupa instrukcija i **njihove implementacije** predstavlja **arhitekturu računara**

Pojam arhitekture računara

- Posebni problemi **upotrebe** (**programiranja**) i **pravljenja** (**implementacije**) **računara**
- Način programiranja zavisi od osobina skupa naredbi - **arhitektura naredbi**
- Implementacija arhitekture naredbi obuhvata **organizaciju i realizaciju računara**
- **Arhitektura računara = arhitektura naredbi + implementacija**

Arhitektura računara



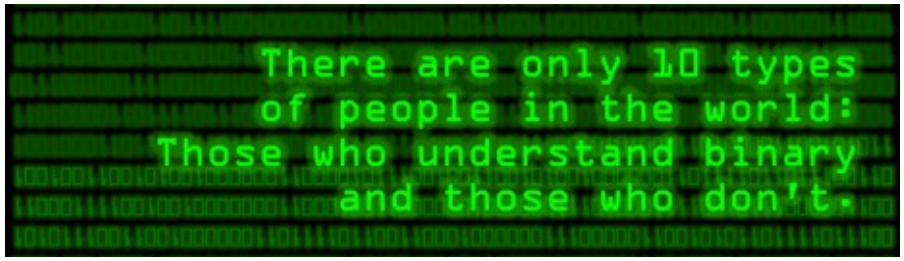
- **Uopšteni model računara – memorija** i **procesor**
- Brzi procesor/memorija naspram sporih izlazno/ulaznih uređaja
- Ključni problem je **savladavanje kompleksnosti**

Programiranje == davanje uputstva računaru
tj. opis algoritma na jeziku razumljivom računaru

Strukturirana organizacija računara

Memorija

- Sastavljena od niza **lokacija**
  - **vrednost** – prosti tip
  - **adresa** – numerička oznaka, broj ili labela – simbolička oznaka
- **Univerzalnost**, višeznačna interpretacija
  - kodiranje naredbi i drugih tipova na opseg celobrojnih brojeva



Procesor

- **Logička kola** ili **sklopovi** – izvršavaju operacije, realizuju funkcije
- **Registri** – posebna memorija unutar procesora, označeni su posebnom oznakom
- **Naredba = oznaka operacije + operandi**
- Operandi su ili vrednosti prostih tipova ili adrese lokacija

Skup naredbi

- **Aritmetičke, relacione, logičke, upravljačke** i **naredbe prebacivanja** (između registara i memorijskih lokacija)
- **Aritmetičke**: dovoljno celobrojno + i –
- **Relacione**: a ? b može pomoću a - b

==	razlika 0
!=	razlika različita od 0
<	razlika negativna
<=	razlika negativna ili 0
>	razlika pozitivna
>=	razlika pozitivna ili 0

- **Logičke:** I, ILI, NE, ekskluzivno ILI (EILI, XOR)
- **Upravljačke**: bezuslovna i uslovna izmena redosleda izvršavanja – realizacija alternativnog (selekcija) i repetativnog toka (iteracija) izvršavanja naredbi

Adresiranje

- **Omogućava pristup operandima**
- **Ulazni i izlazni operandi** (c = a + b, a = a + b)
  - **neposredni** (engl. *immediate*) – konstante
  - **direktni** (engl. *direct*) – promenljive prostih tipova
  - **registarski** (engl. *register*) – promenljive prostih tipova
  - **indirektni** (engl. *register indirect*) – pokazivačke prom.
  - **indeksni** (engl. *indexed*) – promenljive složenih tipova

Primer algoritma - NZD

Euklidov algoritam:

a = n*NZD

b = m*NZD

|a-b| =|n-m|*NZD

Programski kod:

```
int a = 12;
int b = 10;
while (a != b) {
    if (a > b)
        a = a – b;
    else
        b = b – a;
}
```

Proceduralni programski jezici

- **Prosti tipovi podataka** – celi, realni, znakovni, logički
- **Prosti** (osnovni) **tipovi** – osnova za **složene tipove**
- **Operacije** – opisuju obradu podataka, odnosno rukovanje prostim tipovima
- **Promenljive** – daju opštost obradi podataka, imaju ime, vrednost i tip, operacija dodele
- **Promenljive**, **vrednosti prostih tipova** i **operacije** su **osnovni elementi imperativnih jezika**

Mašinske i asemblerske naredbe

- **Mašinska naredba = kod naredbe**
  + kodovi operanada
- Mašinski format naredbe
- **Asemblerska naredba – simbolička oznaka mašinske naredbe**
- **Mašinski** i **asemblerski jezik**
- **Asembler**: asemblerski program u mašinski
- **Prevodilac** (kompajler): izvorni program u asemblerski (mašinski)

Matematička osnova modela računara

Postoji više ekvivalentnih matematičkih formalizacija algoritama (tj. sistema izračunavanja):

- **Tjuringova mašina** (Alan Turing)
- **Lambda račun** (Alonzo Church)
- **Rekurzivne funkcije** (Kurt Gödel)
- **Registarske mašine** (Post-Tjuring mašine – Wang, Minsky, Lambek i još niz naučnika)

Church-Turing-ova teza

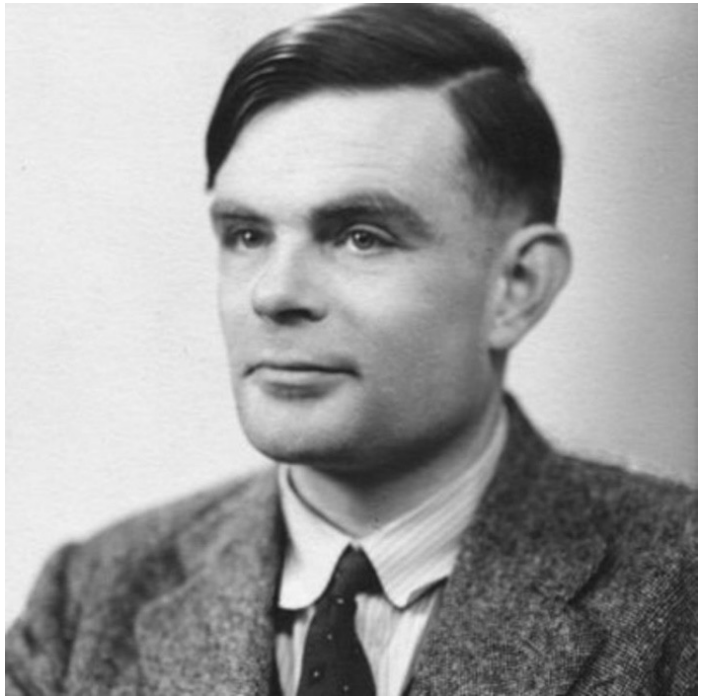
Motivacija: David Hilbert (Entscheidungsproblem 1928.) i Kurt Gödel (teorema o nekompletnosti 1931.), problem zaustavljanja (engl. halting problem)

Problem (funkcija nad prirodnim brojevima**) je algoritamski rešiv** (ignorišući ograničenja resursa**) ako se može rešiti na Tjuringovoj mašini**. Algoritmom se može smatrati svaki niz instrukcija koji se može realizovati na Tjuringovoj mašini.

Tjuringova mašina radi nad konačnim skupom simbola (elemenata) koji se mogu poređati u niz. Dakle, Tjuringova mašina je prebrojiv skup. To znači da je **skup svih algoritama prebrojiv**.

Naravno **skup svih problema odlučivanja je neprebrojiv**, što znači da **postoje problemi za koje se ne mogu definisati algoritmi**

Tjuringova mašina



Alan Turing (1912–1954)

Tjuringova mašina

Tjuringova mašina je **matematički model izračunavanja** koji definiše **apstraktnu mašinu** koja, u skladu sa tabelom operacija, **manipuliše simbolima** koji se nalaze **na traci**. Sastoji se od:

- **beskonačne trake** koja je podeljena na diskretne ćelije; sadržaj svake ćelije može biti specijalni blanko znak ili jedan ili više drugih simbola
- **glave** koja se nalazi nad tačno jednom ćelijom; glava može čitati ili upisivati simbole u svaku ćeliju, kao i pomerati se tačno jednu ćeliju levo ili desno u svakom koraku
- **registra stanja** koji pamti stanje Tjuringove mašine, u svakom trenutku tačno jedno od konačno mnogo mogućih
- **tabele operacija** koja, na osnovu stanja i trenutno pročitanog simbola, kaže mašini da ili obriše ili upiše simbol, pomeri glavu levo ili desno i pređe u novo stanje ili ostane u istom stanju

Približni model Tjuringove mašine

Originalni rad:

Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungs problem". *Proc. London Mathematical Society*. 2 (pub. 1937). **42**: 230–265.

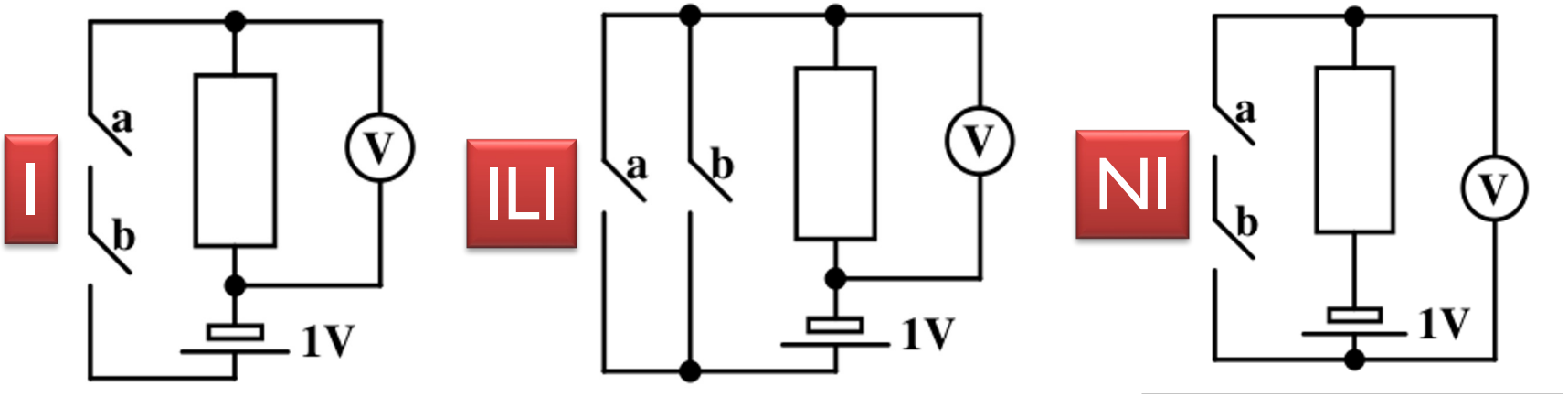
Fizička osnova modela računara

- **Dovoljno** je **samo predstavljanje celih brojeva**
- Binarni brojni sistem – dva nivoa signala
- Podudarnost cifara binarnog brojnog sistema i logičkih vrednosti omogućava **opisivanje aritmetičkih operacija logičkim funkcijama**

a	b	a+b	Prenos
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

a EILI b a I b

Fizička osnova modela računara



NI (NAND) : $a \circ b \equiv \sim(a \& b)$

Univerzalno logičko kolo:

I: $a \& b == (a \circ b) \circ (a \circ b)$

ILI: $a \mid b == (a \circ a) \circ (b \circ b)$

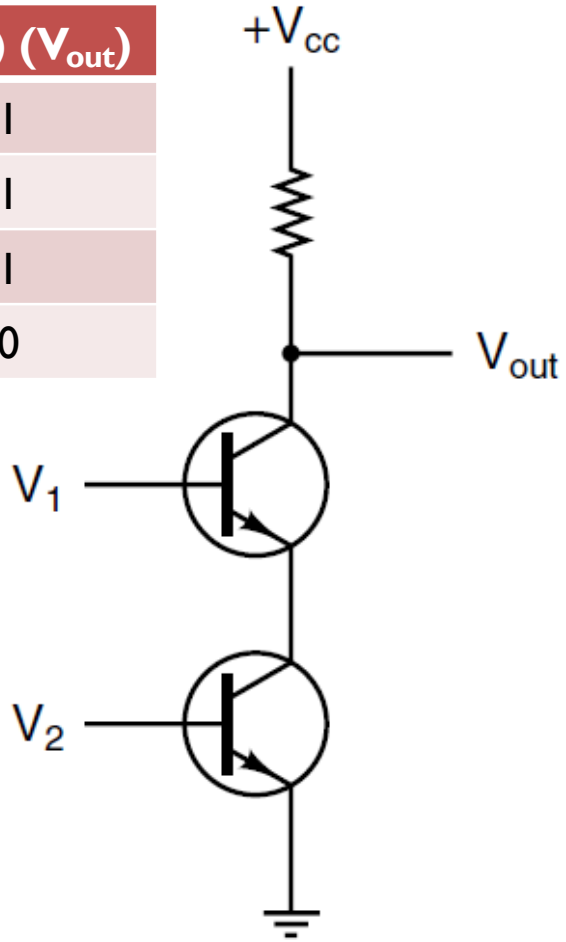
EILI (XOR): $a \wedge b == ((a \circ a) \circ b) \circ (a \circ (b \circ b))$

ENILI (IFF): $a \equiv b == (a \circ b) \circ ((a \circ a) \circ (b \circ b))$

Fizička osnova modela računara

- **Tranzistor = fizička realizacija prekidača**
  - ulaz
  - izlaz
  - upravljački izvod

a (V_1)	b (V_2)	f(a, b) (V_{out})
0	0	I
0	I	I
I	0	I
I	I	0



- **Upravljačka logička funkcija** (prekidačka funkcija)

  f(a) == a

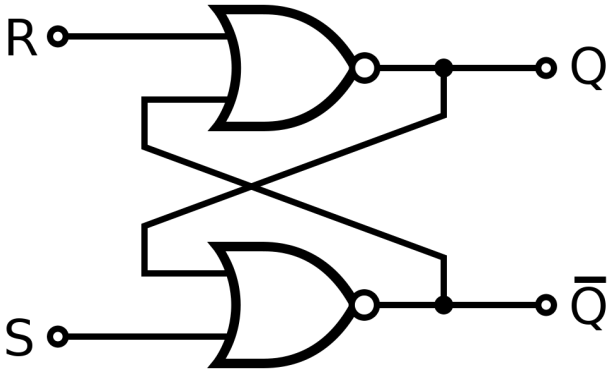
  od vrednosti argumenata zavisi stanje

  f(a, b) == a○b

Fizička osnova modela računara

- **Kombinaciona** i **sekvencijalna kola**
- **Funkcije** od **kombinacionih**, **izlaz** zavisi od **ulaza**
- **Memorija** od **sekvencijalnih**, **izlaz** zavisi od **ulaza** i **stanja**
- **Lokacija** = **niz bitova** (ćelija)
  - bit = flip-flop kolo, memoriše jednu logičku vrednost
- Lokacije organizovane u **bajtove** i **reči**

S	R	Q_{next}	Akcija
0	0	Q	Drži stanje
0	1	0	Reset
1	0	1	Set
1	1	X	Nije dozvoljeno



Osnovna logička kola