Lekcija #1 – Uvod u predmet

Uvod u predmet CS120

Predmet CS120 – Organizacija računara uvodi studente da usvoje osnovne pojmove i principe iz oblasti organizacije računarskih sistema.

Cilj predmeta jeste da se student upozna sa arhitekturom računara i osnovim konceptima arhitektura najpoznatijih mikroprocesora Intel Core i7, kao i procesore sa ARM i RICS-V arhitekturama.

Studenti se upoznaju sa osnovnim pojmovima i principima iz oblasti slojevite organizacije računarskih sistema. Studenti se najpre se uvode u referentne modele savremenih računarskih sistema koja uključuje procesore, primarnu i sekundarnu memoriju, kao i ulazno/izlazne uređaje i magistrale koje povezuju ostale podsisteme.

Studenti se zatim upoznaju sa svim slojevima organizacije računara, i to: sloj digitalne logike, sloj mikroarhitekture i operativnog sistema, sloj skupa instrukcija i asemblerskog jezika, kao i sloj arhitekture paralelnih računarskih sistema, koja uključuje paralelizam na-čipu, ko-procesore, multiprocesore i multiračunare, ali i grid računare.

Nakon odslušanog kursa CS120, i uspešno ispunjenih postavljenih obaveza, student stiče sledeće kompetencije (ITE-PFT Platform Technologies) koje mu omogućavaju da:

- Opiše kako je istorijski razvoj hardvera i računarskih platformi operativnih sistema proizveo računarske sisteme koje imamo danas.
- Prepozna kako da izabere između opcija hardverskih uređaja i odgovarajućih platformi na kojima ti uređaji mogu da funkcionišu.
- Razvije blok dijagram, uključujući i međusobne veze, glavnih delova računarskog sistema i ilustrujte metode koje se koriste na računaru za skladištenje i preuzimanje podataka.

Pravila na predmetu

Student se ocenjuje u toku celog semestra. Ocenjuju se njegovi domaći zadaci, rad na projektu, testovi i aktivnost u nastavi. Na kraju u ispitnom roku, ocenjuje se i pismeni ispit.

Ocene se daju u poenima. Maksimalni broj poena je 100 (uključujući i pismeni ispit). Na pismenom ispitu student može dobiti do 30 poena, dok mu aktivnosti u toku semestra (predispitne obaveze) mogu doneti do 70 poena, po sledećoj strukturi:

- **Testovi**: 14 poena. Tokom semestra postoje 5 testa, svaki test nosi maksimalno do 2.8 poena;
- **Domaći zadaci**: 21 poen. Tokom semestra postoje 14 domaćih zadataka, svaki domaći nosi maksimalno do 1.5 poena;
- **Projektni zadatak**: 25 poena. Tokom semestra svaki student dobija jedinstveni projektni zadatak u vidu tri mini-projekta. Ukupan broj poena na sva tri mini projekta može iznositi 25 poena.
- **Zalaganje**: 10 poena.

Predmetni profesor i asistent u Beogradu:

- Predmetni profesor: Doc. dr Venezija Ilijazi
- email: venezija.ilijazi@metropolitan.ac.rs
- Predmetni asistent: Nemanja Veselinović
- email: nemanja.veselinovic@metropolitan.ac.rs

Predmetni profesor i asistent u Nišu:

- Predmetni profesor: Doc. dr Nemanja Zdravković
- email: nemanja.zdravkovic@metropolitan.ac.rs
- Predmetni asistent: Mihajlo Milovanović
- email: mihajlo.milovanovic@metropolitan.ac.rs

Rad sa online studentima:

- Nemanja Veselinović
- email: nemanja.veselinovic@metropolitan.ac.rs

Jezici, nivoi, virtuelne mašine

Digitalni računar (en. digital computer) je mašina koja može raditi posao za ljude tako što izvršava instrukcije koje su joj date. Sekvenca instrukcija koja opisuje način kako odraditi neki zadatak naziva se program. Elektronska kola u svakom računaru mogu prepoznati i direktno izvršiti samo ograničen broj jednostavnih instrukcija, tako da su svi programi konvertovani u niz ovih jednostavnih instrukcija.

Ove instrukcije su retko komplikovanije od sledećih:

- Sabiranje dva broja,
- Provera da li je neki broj jednak nuli,
- Kopiranje dela podataka iz jednog dela računarske memorije u drugi.

Zajedno, ove primitivne instrukcije formiraju jezik pomoću kojeg ljudi mogu da komuniciraju sa računarom. Ovakav jezik naziva se mašinski jezik (en. machine language).

Kada se projektuje novi računar, inženjeri treba da odluče koje će instrukcije biti uključene u mašinski jezik računara.

U opštem slučaju, cilj je da ove instrukcije budu što jednostavnije i u skladu sa namenom računara.

Budući da su mnogi mašinski jezici jednostavni, tj. sadrže mali skup jednostavnih instrukcija, oni su teški za ljude da ih koriste.

Ovakav način projektovanja doveo do strukturisanja računara kao skup apstrakcija, pri kojoj se svaka apstrakcija gradi na prethodnoj. Na ovaj način, složenost ovakvih sistema se može savladati i računari se mogu projektovati na sistematski i organizovan način.

Ovakav pristup naziva se struktuirana organizacija računara (en. structured computer organization).

Prevođenje i interpretacija

Postoji velika razlika u tome šta je ljudima jednostavno uraditi i šta računar može direktno da izvrši. Ovaj problem se može rešiti na dva načina: oba načina uključuju projektovanje novog skupa instrukcija koji je jednostavniji ljudima za razumevanje od skupa ugrađenih mašinskih instrukcija (instrukcija za mašinski jezik računara). Zajedno, ove nove instrukcije formiraju jezik, koji ćemo nazvati L1, dok skup ugrađenih mašinskih instrukcija nazivamo jezikom L0.

Dva načina koja su opisana razlikuju se na koji način računar izvršava programe napisane na jeziku L1 koristeći svoj jezik L0.

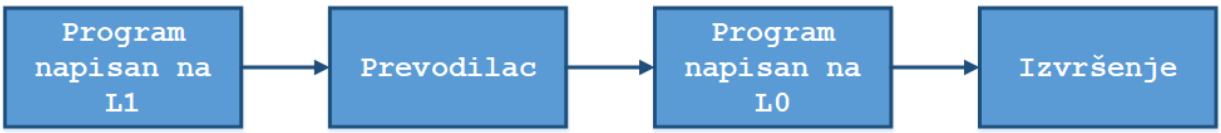
Prvi način izvršenja programa koji je napisan na jeziku L1 jeste da najpre zameni svaku instrukciju unutar tog programa ekvivalentnom sekvencom instrukcija na jeziku L0. Rezultujući program sadrži samo instrukcije na jeziku L0. Računar onda može izvršiti novi program napisan na jeziku L0 umesto stari program napisan na jeziku L1. Ova tehnika se naziva ***prevođenje*** (en. translation).

Drugi način izvršenja programa jeste napisati program na jeziku L0 koji uzima programe napisane na jeziku L1 kao ulazne podatke (en. input), analizira svaku instrukciju pojedinačno i odmah izvršava njoj ekvivalentan skup instrukcija napisan na L0.

Ova tehnika ne zahteva generisanje novog programa na jeziku L0, naziva se ***interpretacija*** (en. interpretation), dok se program koji izvršava ovaj proces naziva interpreter (en. interpreter).

U interpretaciji, nakon što se svaka instrukcija na jeziku L1 analizira i dekoduje, odmah se izvršava, tj. ne generiše se prevedeni program. Interpreter je na ovaj način preuzeo kontrolu nad računarom, jer su za njega programi napisani na jeziku L1 samo ulazni podaci.

Razlika između prevođenja i interpretacije prikazana je na slikama.



Slika-1 Prevođenje programa. [Izvor: Autor]

Slika-2 Interpretacija programa. [Izvor: Autor]

Virtuelne mašine

Umesto razmišljanja o organizaciji računara preko termina prevođenja i interpretacije, često je lakše zamisliti postojanje ***hipotetičkog računara*** ili virtuelne mašine (en. virtual machine), ***čiji je mašinski jezik L1***. Ovu mašinu nazvaćemo M1. Na sličan način mašina koja radi na jeziku L0 zvaće se M0.

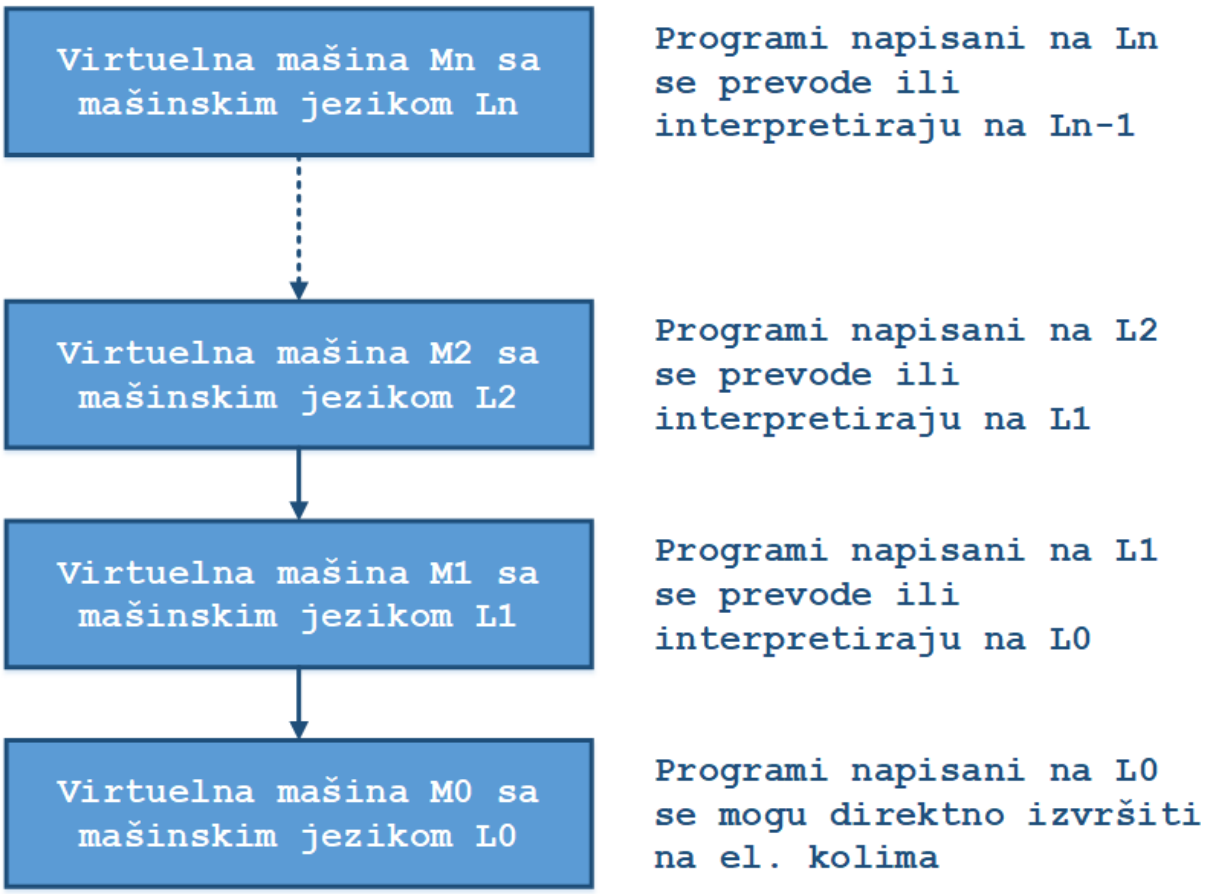
Ukoliko bi bilo moguće konstruisati takvu mašinu relativno jeftino, ne bi bilo potrebe za mašinom koja zahteva jezik L0 ili mašinu koja izvršava programe napisane na jeziku L0.

Ljudi bi jednostavno mogli da pišu svoje programe na jeziku L1. tj. za virtuelne mašine kao da zaista postoje.

Da bi prevođenje i interpretacija bila moguća, jezici L0 i L1 ne bi trebalo biti mnogo različiti. Ovo ograničenje znači da jezik L1, iako „bolji“ od L0, i dalje ne bi bio idealan za mnoge primene. Rešenje i dalje postoji – treba izmisliti još jedan skup instrukcija koji je više orjentisan ljudima (tj. ljudskom govoru) a manje orjentisan mašinama. Ovaj jezik nazvaćemo L2, uz odgovarajući virtuelnu mašinu M2.

Na ovaj način, ljudi mogu pisati programe na L2 kao da postoji virtuelna mašina sa direktnim izvršavanjem jezika L2. Ovakvi programi se mogu prevesti ili interpretirati na L1.

Formiranje celog skupa jezika, gde je svaki jezik jednostavniji za ljude od prethodnog, može ići sve dok ne dostignemo nivo jednostavnosti koji želimo. Svaki jezik koristi prethodni kao osnovu, tako da možemo gledati na računar kao na skup slojeva ili nivoa, jedan preko drugog, kao što je prikazano na slici. Nivo na dnu je najjednostavniji za računare za razumevanje, dok je nivo na vrhu najsofisticiraniji.



Slika-3 Višenivoovska mašina. [Izvor: Autor]

Postoji bitni odnos između jezika i virtuelne mašine. Svaka mašina ima svoj mašinski jezik, koji se sastoji od svih instrukcija koja ta mašina može direktno izvršiti. Na taj način *mašina definiše jezik*.

Danas je moguće napraviti mašinu koja bi koristila C ili C++ ili Java programski jezik za mašinski jezik, ali bi bila previše kompleksna i skupa za konstrukciju.

Računar sa n nivoa može se gledati kao **n** virtuelnih mašina, svaka sa svojim mašinskim jezikom. Samo programi napisani na jeziku **L0** se mogu direktno izvršiti u elektronskom kolima bez potrebe prevođenja ili interpretacije. Programi napisani na jezicima **L1**, **L2**, **Ln** moraju se ili prevesti ili interpretirati na jezik nižeg nivoa.

Osoba koja piša programe za virtuelnu mašinu nivoa n ne treba da bude svesna interpretera i prevodioca ispod svoj nivoa.

Sama struktura mašine garantuje da će se programi na neki način izvršiti.

Za programere mašine **Mn** na jeziku **Ln** nije od interesa koliko će koraka biti izvršeno u pozadini dok se program ne pokrene – program se jednostavno izvršava.

Mnogim programerima koji koriste n-nivoovske mašine od interesa je samo poslenji, gornji nivo, tj. programski jezik koji najmanje liči na mašinski jezik (koji je na dnu).

Međutim, za ljude koje zaista interesuje kako računar radi, potrebno je poznavati sve nivoe.

Takođe, ko se bavi projektovanjem novih računara ili novih nivoa, treba znati sve nivoe, a ne samo poslednji, gornji nivo.

Koncepti i tehnike konstruisanja mašine kao niza nivoa i detalja svakog nivoa glavni je predmet istraživanja na predmetu CS120.

Računari sa n nivoa

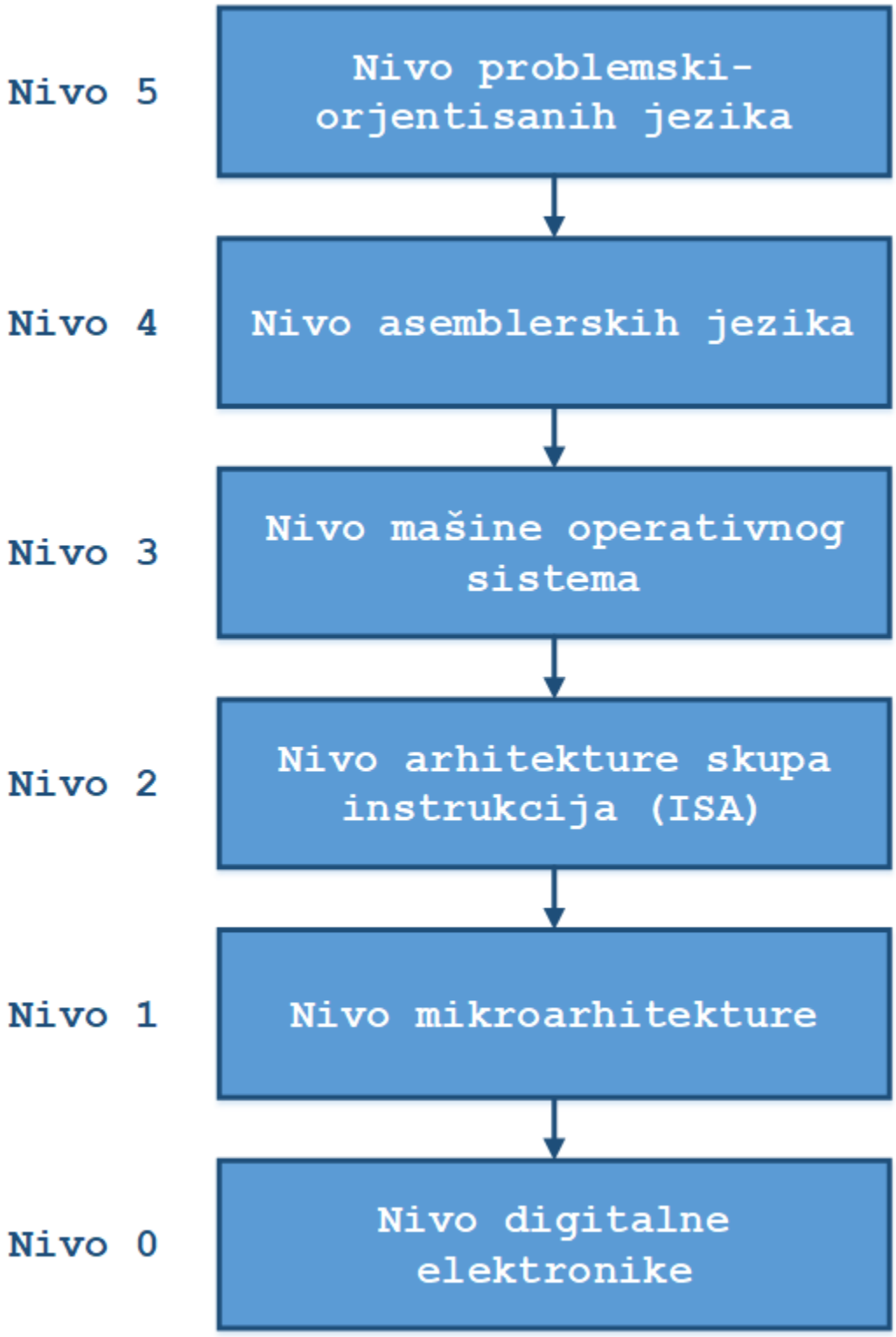
Većina savremenih računara sastoji se od dva ili više nivoa.

Na slici je prikazana jedna takva mašina, koja ima šest nivoa. Na nultom nivou nalati se pravi hardver mašine. Elektronska kola izvršavaju mašinski programski jezik napisan na nivou L1.

Čak se može reći da postoji i još jedan nivo ispod nivoa 0, koji se naziva ***nivoom uređaja*** (en. device level), na kojem projektant radi sa pojedinačnim tranzistorima, koji su najmanja jedinica projektovanja računara. Radom samih tranzistora bavi se fizika čvrstog stanja (en. solid state physics).

Tokom ovog kursa, najniži nivo kojim ćemo se baviti jeste nivo digitalnih kola (en. gates), čiji su gradivni blokovi digitalna kola. Iako se prave od analognih komponenti (tranzistora), digitalna kola se mogu modelovati kao digitalni uređaji. Svako kolo ima jedan ili više digitalnih ulaza (0 ili 1) i računa izlaz iz kola kao jednostavna logička funkcija, ***ILI*** (en. OR) ili ***I*** (en. AND). Svako digitalno kolo sastoji se od nekolicine tranzistora. Takođe, ova kola mogu poslužiti za skladištenje jednog bita memorije (skladištenje 0 ili 1).

Ove jednobitne memorije se mogu grupisati grupe od npr. 16, 32 ili 64 da formiraju registar (en. register). Svaki registar može skladištiti binarni broj do neke vrednosti. Digitalna kola se takođe mogu kombinovati da formiraju i samo jezgro računanja u računaru.



Slika-1 Računarska mašina sa šest nivoa. [Izvor: Autor]

Nivo mikroarhitekture i nivo arhitekture skupa instrukcija

Sledeći nivo jeste nivo mikroarhitekture (en. microarchitecture level). Na ovom nivou imamo skup registara (npr. skup od 8 ili od 32 registra) koji formiraju lokalnu memoriju ili komplikovanije kolo koje se naziva aritmetičko-logička jedinica (en. Arithmetic Logic Unit, ***ALU***), koja služi za jednostavne aritmetičke operacije.

Registri su povezani na ALU formiraju ***putanju podataka*** (en. data path), preko koje podaci prolaze. Osnovna uloga putanje podataka jeste da izabere neki (ili nekoliko) registara, da ALU izvrši neku operaciju nad vrednostima u njima (npr. da ih sabere) i da rezultat vrati nazad u neki registar.

U nekim mašinama operacije putanje podataka su kontrolisane od strane ***mikroprograma*** (en. microprogram). U drugim mašinama, sam hardver kontroliše putanju podataka. Ukoliko softver kontroliše putanju podataka, mikroprogram je interpreter za instrukcije na nivou 2. Ovakav interpreter bi ***prihvatio*** (en. fetch) instrukcije, ***video*** koje su instrukcije u pitanju (en. examine) i konačno bi ***izvršio*** te instrukcije (en, execute) jednu po jednu.

Primer:

Za instrukciju sabiranja ADD, instrukcija bi se prihvatila, operandi pronašli i ubacili u registre, sabiranje bi izvršila ALU jedinica, i rezultat bi se skladištio gde već treba.

Za mašinu čiju putanju podataka kontroliše hardver, isto bi se desilo, samo bez eksplicitnog programa koji kontroliše interpretaciju instrukcija 2. nivoa.

Drugi nivo naziva se nivo arhitekture skupa instrukcija (en. Instruction Set Architecture, ***ISA***).

Svaki proizvođač računarske opreme ovog tipa objavljuje i uputstvo za ISA sloj, bez nižih slojeva.

Kada proizvođač opisuje skup instrukcija računara, zapravo se opisuju instrukcije koje se izvršavaju interpretiranjem mikroprograma ili samog hardvera.

Hibridni nivo višenivoovske mašine

Sledeći nivo je često nazvan hibridnim. Većina instrukcija u ovom sloju postoje i na ISA sloju. Takođe postoji i novi skup instrukcija, drugačija organizacija memorije, mogućnost da se dva ili više programa izvršavaju u isto vreme ili ***konkurentno*** (en. concurrent), kao i niz drugih osobina.

Na ovom nivou postoji viša varijacija nego na nivoima 1 ili 2. Istorijski, ovaj nivo se zvao nivo operativnog sistema.

Neke instrukcije na nivou 3 se interpretiraju od strane operativnog sistema, a neke direktno od strane mikroprograma ili hardvera.

Zbog toga se ovaj nivo naziva hibridnim. Do kraja kursa, ovaj nivo nazivaćemo nivo mašine operativnog sistema (en. operating system machine level).

Postoji bitna razlika između nivoa 3 i 4. Najniža tri nivoa nisu za „svakodnevne“ programere. Zapravo, ovo su nivoi koji se primarno koriste za izvršenje interpretera i prevodioca za podršku viših nivoa.

Ovi interpreteri i prevodioci pišu ljudi koji se nazivaju ***sistemski programeri*** (en. system programmers), čija je specijalizacija projektovanja i implementacija novih virtuelnih mašina.

Još jedna promena u ovom nivou jeste način na koji su viši nivoi podržani. Nivoi 2 i 3 su uvek interpretirani. Nivoi od nivoa 4 pa naviše su uglavnom, ali ne uvek, podržani i kroz prevođenje.

Još jedna razlika između nivoa 1, 2 i 3 sa nivoima 4, 5 i viših jeste priroda programskih jezika za ove nivoe.

Mašinski jezici za nivoe 1 i 2 i 3 su numerički.

Progrmi u njemu se sastoje od dugog niza brojeva, koji su laki za razumevanje mašina, ali teški za razumevanje od strane ljudi. Od 4. nivoa, programski jezici zaista sadrže reči ili skraćenice koji imaju neki značaj za ljude koji pišu programe na ovim jezicima.

Nivo asemblerskih jezika

Nivo 4, tj. nivo asemblerskih jezika (en. assembly language level), predstavlja simboličku formu nekih od jezika na nižem nivou.

Ovaj nivo pruža mogućnost da ljudi pišu programe na nivoima 1, 2 i 3 u formi koja nije tako neprijatna kao mašinski jezik tih virtuelnih mašina.

Programi u asemblerskom jeziku se najpre prevode na nivo 1, 2 ili 3, pa se onda interpretiraju na pravoj (ili virtuelnoj) mašini. Program koji vrši ovo prevođenje naziva se asembler (en. assembler).

Mašinski kod:

```
B8 22 11 00 FF
01 CA
31 F6
53
8B 5C 24 04
8D 34 48
39 C3
72 EB
C3
```

Ekvivalentan kod napisan u asemblerskom jeziku:

```
foo:
movl $0xFF001122, %eax
addl %ecx, %edx
xorl %esi, %esi
pushl %ebx
movl 4(%esp), %ebx
leal (%eax, %ecx, 2), %esi
cmpl %eax, %ebx
jnae foo
retl
```

Tok instrukcija (en. instruction stream):

```
B8 22 11 00 FF 01 CA 31 F6 53 8B 5C 24 04 8D 34 48 39 C3 72 EB C3
```

Nivo problemski-orjentisanih jezika

Peti nivo najčešće se sastoji od programskih jezika koji služe za rešavanje konkretnih problema.

Ovi programski jezici se nazivaju ***viši programski jezici*** (en. high-level languages), i to su jezici kao što su C, C++, Java, Python, JavaScript i sl.

Programi koji su napisani u ovim jezicima često su prevedeni na nivo 3 ili nivo 4 koristeći prevodilac koji se naziva kompajler (en. compiler), ali postoje i interpretirani jezici.

Programi koji se napišu u Javi se recimo najpre prevode na jezik sličan ISA jezicima nazvan ***Java bajt-kod*** (en. Java byte code), koji se kasnije interpretira.

U nekim slučajevim, nivo 5 se sastoji od interpretera za specifičnu aplikaciju ili domen, kao što je simbolična matematika. Onda takav jezik pruža podatke i operacije za rešavanje problema u ovom domenu kroz termine koji ljudi iz tog domena mogu da razumeju sa lakoćom.

Kompajler naspram interpretera (video):

https://www.youtube.com/embed/e4ax90XmUBc

Arhitektura računara

Bitno je zapamtiti da su računari projektovani da poseduju više nivoa, tako da se svaki gradi na prethodnom.

Svaki nivo prestavlja posebnu apstrakciju, sa svojim skupom objekata i operacija.

Analizirajući ovako projektovane računare, možemo se fokusirati samo na nivo od interesa, i na taj način smanjujemo nivo kompleksnosti cele mašine.

Skup tipova podataka, operacija i osobina svakog nivoa naziva se arhitektura.

Arhitektura ima za cilj da samo oni aspekti iz tog nivoa budu vidljivi korisniku tog nivoa. Svojstva koje programer vidi, kao što su količina raspoložive memorije, jeste deo arhitekture. Aspekti implementacije, kao što je tehnologija koja se koristi za implementaciju te memorije, nije deo arhitekture.

Deo računarske nauke koja se bavi projektovanjem koji su delovi računarskog sistema vidljivi programerima naziva se arhitektura računara (en. computer architecture).

Arhitektura računara naspram organizacije računara (video):

https://www.youtube.com/embed/Ol8D69VKX2k

Generacije i tipovi računara

U ovoj sekciji objašnjene su najpre generacije računara.

Od prvobitnih mehaničkih računarskih mašina iz 17. veka do poslednje generacije računara koji staju na ljudski dlan, evolucija računarskih sistema je doživela najveću ekspanziju.

Zatim, urađena je podela računara po tipovima.

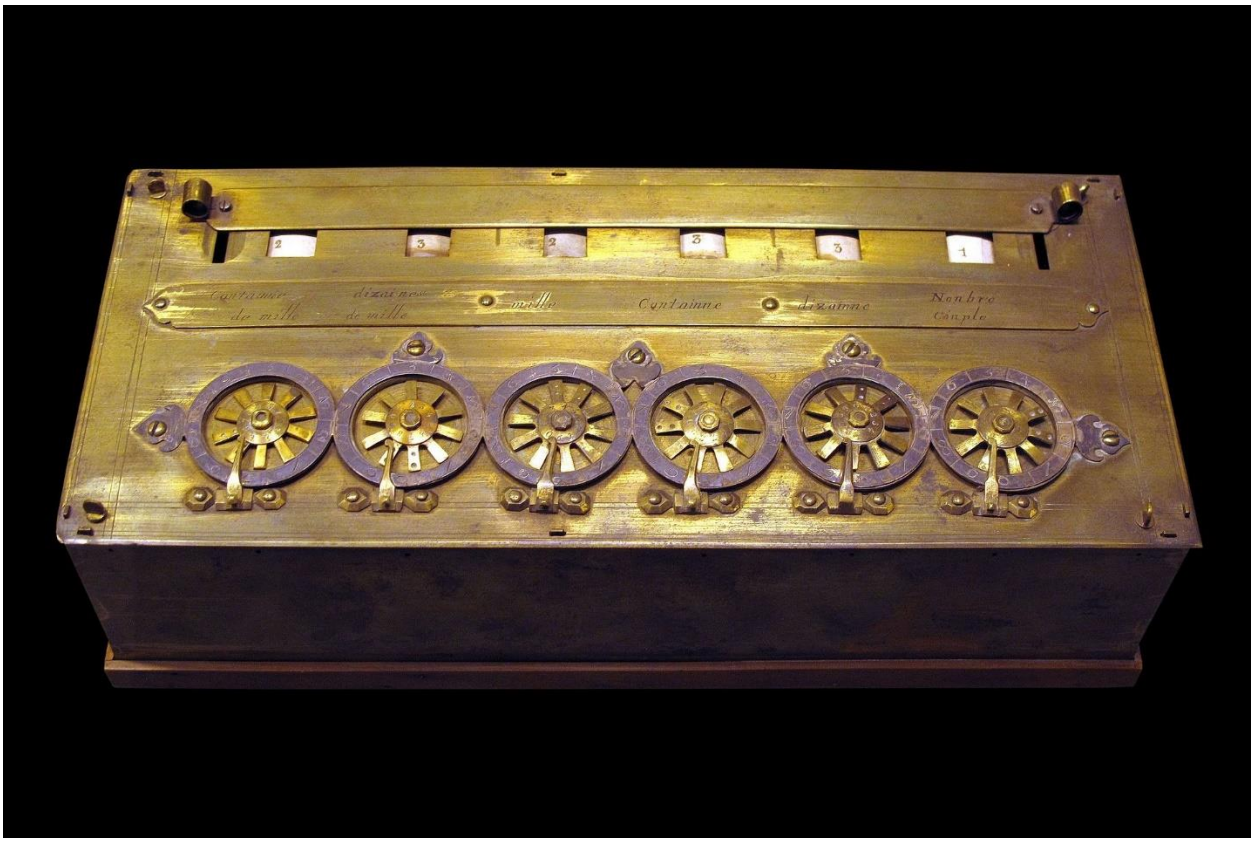
Generacije računara:

https://www.youtube.com/embed/FLst_k_eWkE

Nulta i prva generacija računara

Nulta generacija (1642-1945)

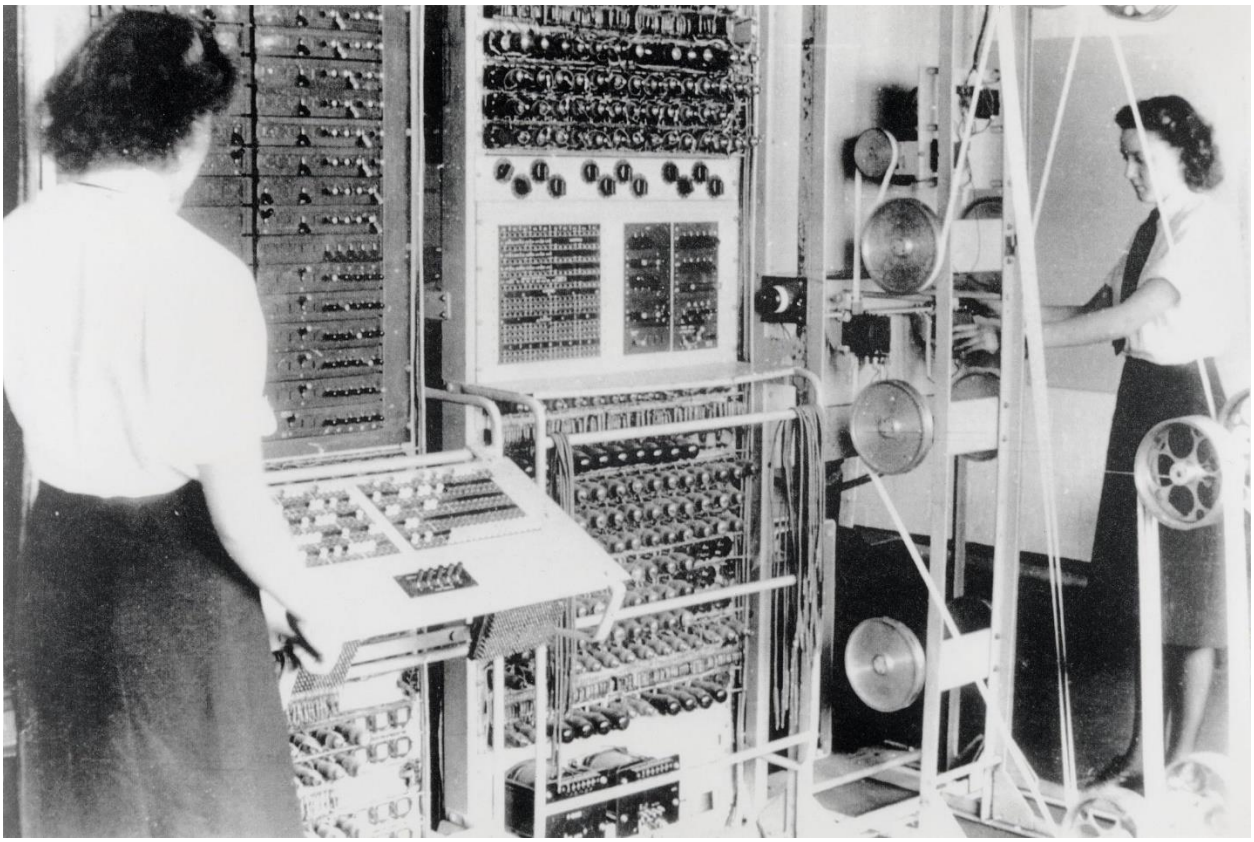
Prva osoba koja je napravila mašinu za računanje bio je francuski naučni Paskal, po kome je i programski jezik Pascal dobio naziv. Ovaj uređaj, koji je mogao da radi operacije sabiranja i oduzimanja, radio je pomoću niza zupčanika i ručne poluge. Na osnovu ove mašine Lajbnic (Gottfried Wilhelm von Leibniz) je napravio mehaničku mašinu koje je radila i operacije množenja i deljenja.



Slika-1 Paskalov kalkulator. Izvor: [https://en.wikipedia.org/wiki/Pascal%27s_calculator]

Prva generacija (1945-1955) – Vakuumske cevi

Elektronski računarski sistemi počeli su sa razvojem u periodu Drugog Svetskog rata. ENIGMA mašina enkripcije poruka koju su koristile sile osovine bila je osnova za razvoj prvog elektronskog računara COLOSSUS, koji je imao zadatak da dekriptuje poruke. COLOSSUS računar je projektovao matematičar Alan Turing.
Prvi zvanični računarski sistem ENIAC (Electronic Numerical Integrator And Computer) razvijen je 1945. godine, sastojao se od 18000 vakuumskih cevi i 1500 releja, bio je težak 30 tona i trošio je 140 kilovata električne energije. Ovaj računar imao je 20 registara kod kojih je svaki mogao da sadrži desetocifreni broj.



Slika-2 COLOSSUS Mark 2 računar. [Izvor: https://en.wikipedia.org/wiki/Colossus_computer]

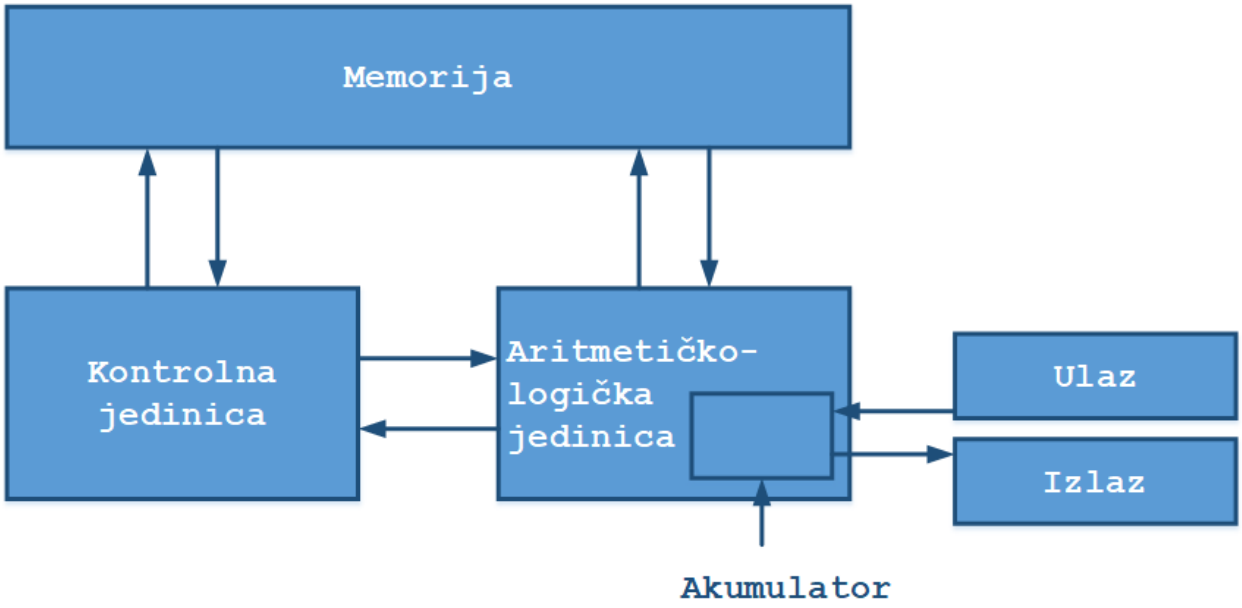
von Neumann mašina

Nakon nekoliko sličnih mašina (EDVAC i EDSAC i ISA mašina), arhitektura računarskih mašina razvijala se po principima koje je postavio Džon von Njuman (John von Neumann), tako da je arhitektura računara nazvana ***von Njuman mašina*** (en.von Neumann machine).

Osnovna arhitektura ovakve mašine prikazana je na slici.

Von Neumann mašina sadrži pet delova: memoriju, ALU jedinicu, kontrolnu jedinicu, i ulazno-izlazni podsistem.

U savremenim računarskim sistemima ALU i kontrolna jedina su kombinovani u jedno integrisano kolo nazvano centralni procesor (en. central processing unit, ***CPU***).

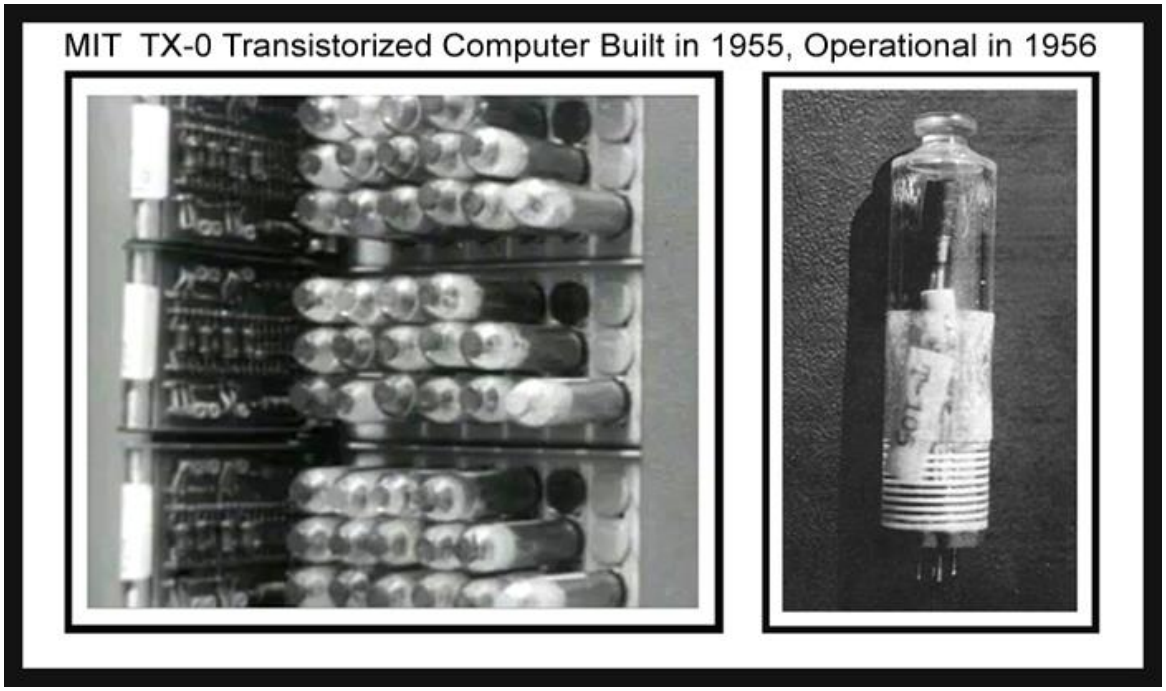


Slika-3 Dijagram originalne von Neumann mašine. [Izvor: Autor]

Druga generacija – Tranzistori (1955-1965)

Osnovne gradivne blokove računarskih sistema u ovoj generaciji činili su tranzistori. Prvi računar koji je koristio tranzistore umesto vakuumeskih cevi bio je ***TX-0 računar*** (en. Transistorized eXperimental computer 0), razvijen od strane MIT Linkoln laboratorije.

U ovoj generaciji imamo i pojavu bržih mašina, nazvanih superračunarima, čiji je pionir bio Sejmor Krej (Seymour Cray), sa računarima 6600, 7600, i Cray-1.



Slika-4 TX-0 računar. [Izvor: https://en.wikipedia.org/wiki/TX-0]

Treća generacija – integrisana kola (1965-1980)

Pronalaskom silicijumskih integrisanih kola od strane naučnika Džeka Kilbija i Robera Nojsa 1958. godine, postalo je moguće smestiti veći broj tranzistora na jedno integrisano kolo (en. integrated circuit) ili čip (en. chip).

U ovoj generaciji kompanija IBM vodila je proizvodnju računara, sa računarom System/360, čiji je najveći doprinos bio da su računari iz ove familije imali isti assemblerski jezik, kao i uvođenje multiprogramiranja, tj. mogućnost da više programa budu u memoriji računara odjednom.

Slika-5 IBM System/360 računar. Izvor: [https://en.wikipedia.org/wiki/IBM_System/360]

Četvrta i peta generacija računara

Četvrta generacija – VLSI (1980 - ?)

Četvrta generacija uvodi integraciju velikih razmera (en. very large scale integration, ***VLSI***), tj. mogućnost da najpre desetina hiljada, pa onda stotina hiljada, i konačno milione tranzistora budu na jednom činu. Od 1980. godine cene proizvodnje čipova su drastično opale, omogućivši razvoj ***ličnih računara*** (en. personal computer, ***PC***).

PC računari koristili su se u značajno druge svrhe u odnosu na velike korporacijske računare. Koristili su se prvenstveno za obradu teksta, tabela, i za zabavu. Prvi personalni računari prodavaili su se kao kit-ovi, odnosni dolazili us rasklopljeni. Najveći pomak u ovoj generaciji bili su čipovi kompanije Intel najpre sa procesorom 8080, i kasnije sa 8086, čija se arhitektura zadržala (x86 arhitektura) zadržala i do danas.

https://en.wikipedia.org/wiki/IBM_Personal_Computer

Peta generacija – računari sa niskom potrošnjom i „nevidljivi“ računari

Još početkom 80tih godina prošlog veka ulagalo se u tzv. računare pete generacije, koji bi bili bazirani na veštačkoj inteligenciji, sa Japanom na čelu. Od ovih planove se ubrzo odustalo, ali peta generacija računara se pojavila u drugom obliku – računari su smanjili dimenzije.

Krajem 80tih godina prošlog veka razvijen je prvi tablet računar GridPad kompanije GridSystems. Prvi PDA računar (en. Personal Digital Assistent) pojavio se 1993. godine sa Apple Newton računarom.
Ovi koncepti doveli su naravno do pojave pametnih telefona koji se danas svakodnevno koriste. Prvi pametni telefon, nazvan Simon, ravijen je 1994. godine od strane IBM-a u kolaboraciji sa japasnskim Mitsubišijem.

https://en.wikipedia.org/wiki/GRiDPad

Murov zakon

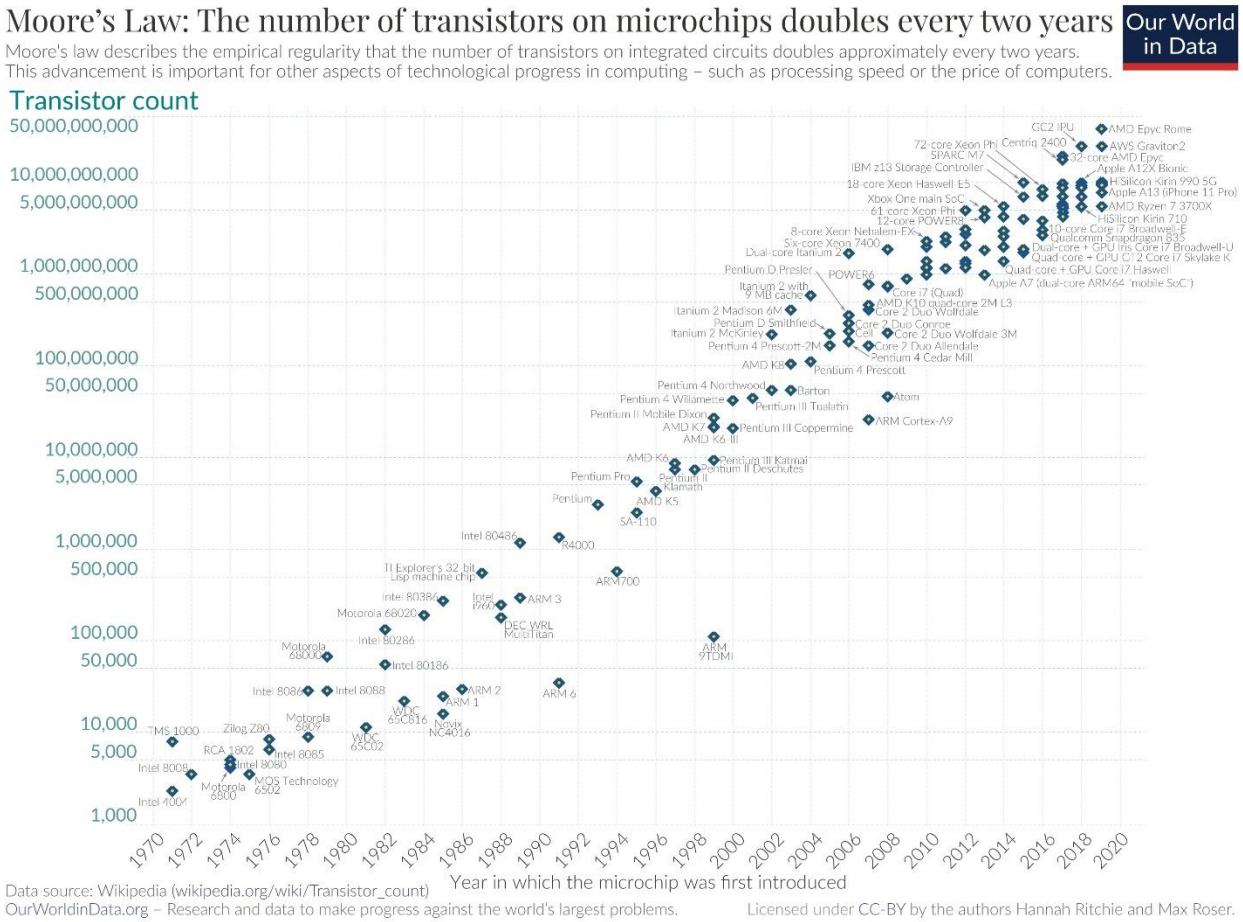
Gordon Mur (Gordon Moore), jedan od su-osnivača Intel-a, primetio je da se nova generacija procesorskih čipova objavi na svake tri godine, sa približno dvostrukim brojem tranzistora na čipu.

Po ovom trendu broj tranzistora na čipu se povećavao kontansnom brzinom, i Mur je prdvideo da će ovaj rast ići decenijama unapred. Ovo zapažanje postalo je poznato kao ***Murov zakon*** (en. Moore's Law).

Danas se za Murov zakon navodi da se broj tranzistora na čipu procesora udvostručuje na svakih 18 meseci, odnosno oko 60% godišnje.

https://www.youtube.com/embed/EzyJxAP6AQo

Slika sa desne strane prikazuje Murov zakon.



Slika-6 Murov zakon. [Izvor: https://en.wikipedia.org/wiki/Moore%27s_law]

Tipovi računara

Jednokratni računari

Najjednostavniji računari predstavlji bi tzv. jednokratne računare, tj. računare koji se mogu iskorisiti jednom.

Osnovni primer ovakvih računara jesu npr. rođendanske čestitke koje sviraju melodiju kada se otvore.

U ove računare spadaju i računari sa ***RFID*** (en. Radio Frequency Identification) čipovima. RFID čipovi se mogu proizvesti za nekoliko centi po komadu, ne koriste baterije i tanji su od 0.5mm, sadrže primotredajnik i jedinstveni 128-bitni broj.

Kada eksterna antena pošalje impuls, mogu se dovoljna napajati da pošalju svoj broj nazad do antene. RFID čipovi se koriste umesto bar kodova, na platnim karticama, implantima za kućne ljubimce i mnogim drugim primenama.

RFID video objašnjenje:

https://www.youtube.com/embed/DcloKaobTGI

Mikrokontroleri

Računari koji su ***ugrađeni*** (en. embedded) u druge uređaje i koji se ne prodaju kao računari nazivaju se ***mikrokontrolerima*** (en. microcontrollers).

Neke od primera mikrokontrolera jesu sledeći:

- Uslužni uređaji (satovi, uređaji bele tehnike, sistemi alarma)
- Komunikacioni uređaji (bežični telefoni, mobilni telefoni, faks, pejdžer)
- Računarske periferije (štampači, skeneri, modemi, optički uređaji)
- Uređaji za zabavu (video rekorder, DVD plejeri, muzički sistemi, mp3 plejeri, set-top boxovi)
- Uređaji za rad sa slikama (TV uređaji, digitalne kamere i kamkoderi, fotokopir aparati)
- Medicinski uređaji (skeneri, magnetne rezonance, digitalni toplomeri)
- Sistemi oružja (različite rakete, torpedo)
- Igračke

Jedan savrememen automobil može imati 50 mikrokontrolera koji izvršavaju podsisteme kao što su kočnice, injekcija goriva, radio, farovi, GPS sistem i mnoge druge primene. Avion može imati preko 200 mikrokontrolera. Za razliku od jednokratnih računara, mikrokontroleri jesu mali, ali kompletni računarski sistemi. Svaki mikrokontroler ima procesor, memoriju, i neku mogućnosti I/O podsistema. Mogu biti opšteg tipa i konkretnog tipa. Mikrokontroleri se, iako su kompletni računari, razlikuju od PC računara u bitnim svojstvima, kao što su cena, rad u realnom-vremenu i u raznim vidovima ograničenja.

https://www.youtube.com/embed/uz0jTNSQ37Y

Računari za zabavu i Mobilni računari

Računari za zabavu kao što su konzole prestavaljaju specijalizovani računarske sisteme sa specijalizovanim grafičkim mogućnostima, ali slabijim centralnim procesorima i ograničenom mogućnosti proširivanja.

Iako konzole često nemaju jednake performanse kao PC računari visokih performansi iz istog perioda prozvodnje, one ne zaostaju mnogo. Zapravo, nekad imaju i bolje performanse u specifičnim zadacima.

https://www.youtube.com/embed/PwZEBE66an0

Mobilni računari imaju dodatno ograničenje u pogledu potrošnje električne energije jer se napajaju preko baterija. Ovo predstavlja ogromno ograničenje pri projektovanju ovakvih računara, jer uređaji kao što su pametni telefoni i tableti moraju pružati visoke performanse u pogledu 3D grafike, obradu audio i video sadržaja visoke rezolucije, a pritom treba trošiti što manje energije.

https://www.youtube.com/embed/NKfW8ijmRQ4

Personalni računari

Termin personalni računar je često sinonim za termin računar. Ovakvi računari uključuju desktop i laptop/notebook modele. Ove računare karekteriše prvenstevno namena koja nije striktno definisana, već su ovo računari opšte namene. Sve računare takođe karakteriše povezivanje pojedinačnih delova na posebnu štampanu ploču, nazvana matična ploča (en. mainboard, motherboard). PC računari i dalje prate von Neumann arhitekturu.

Notebook/laptop računari su po arhitekturi takođe PC računari, samo u manjim dimenzijama, sa mogućnosti napajanja preko baterija.

https://www.youtube.com/embed/VfvGS7qJr4M

Serveri i računarski klasteri

Serveri su računari koji se mogu smatrati PC računarima jako visokoih persormansi, ali sa specifičnom namenom, a to je da obrađuju mnogo veći broj zahteva u odnosu na PC računare. U pogledu arhitekture, server sa jednim procesorom je samo PC računar sa mnogo većim performansama, dok serveri sa više fizički odvojenim procesorima mogu imati slične, ali ne identične arhitekture.

Savremeni mobilni računari, računari za zabavu, personalni računari i serveri, pokreću i operativni sistem, koji pokreće ostale ostale programe na računarima.

https://www.youtube.com/embed/UjCDWCeHCzY

Zvog kontinualnog poboljšanja u odnosu cene i performansi servera, poslednje decenije veliki broj servera se povezuju u klastere (en. cluster).

Klasteri predstavljaju veliki broj servera koji su međusobno povezani brzim računarskim mrežama i na kojima se izvršava specijalizovan softver koji omogućava da ovi računari paralelno obrađuju isti zadatak. Klasteri se često skladište u specijalizovanim zgradama nazvani data centrima, koji često sadrže i preko 10000 mašina.

https://www.youtube.com/embed/7rooWbLe1iI

Mainframe računari i superračunari

Mainframe računari predstavljaju računari koji su bili veličine soba ili spratova, i koji su su koristili od 60tih godina prošlog veka. Ovakvi računari nisu bili po performansama jači od serverskih računara, ali su imali mnogo veće I/O mogućnosti. Sa pojavnom data centara i klaud računarstva (en. cloud computing), dolazimo do ere nove iteracije mainframe računarstva.

https://www.youtube.com/embed/_a6us8kaq0g

Posebna kategorija računara, još jačih od mainframe računara jesu superračunari. Ovi računari sadrže izuzetno jake centralne procesore, i opremljeni su izuzetno brzom memorijom (kako operativnom tako i spoljašnjom) velikog kapaciteta. Prvenstveno se koriste za naučna i inženjerska izračunavanja. Slično i sa mainframe računarima, zbog popularnosti data centara, pravi superračunara ima sve manje.

https://www.youtube.com/embed/9M99STmu-vI

Intel 8086 i x86 arhitektura

Uvod u x86 familiju procesora

Godine 1976. Intel je poizveo 16-bitni CPU na jednom čipu.

Procesor 8086 je bio naslednik 8080, iako nije bio u potpunosti kompatibilan. Sve čipove koje je intel proizveo u budućem periodu nasledile su arhitekturu procesora 8086. Procesor 8088, iako kasnije proizveden, je bio sporiji, ali i jeftiniji, te je IBM izabrao upravo 8088 za svoje PC računare.

Procesor 8086 je bio pravi 16-bitni CPU na jednom čipu. 8086 je bio projektovan da bude sličan prethodnom modelu 8080, ali nije bio u potpunosti kompatibilan sa 8080. Nakon 8086, Intel je proizveo 8088, koji je imao istu arhitekturu kao i 8086 i pokrenuo je iste programe, ali je imao 8-bitnu magistralu umesto 16-bitne magistrale, što ga je činilo sporijim ali i jeftinijim modelom od 8086.

Kada je IBM izabrao 8088 kao CPU za originalni IBM PC, ovaj čip je brzo postao standard za personalne računare. Što se tiče performansi, ni 8088 ni 8086 nisu mogli da adresiraju više od 1 megabajta memorije. Do ranih 1980-ih ovo je postalo ozbiljan problem, pa je Intel projektovao 80286, koji je ostao kompatibilan sa 8086.

Osnovni skup instrukcija je bio u suštini isti kao kod 8086 i 8088, ali organizacija memorije je bila sasvim drugačija, upravo zbog zahteva kompatibilnosti sa starijim čipovima.

Sledeći logičan korak bio je pravi 32-bitni CPU na jednom čipu, 80386, predstavljen 1985. Kao i 80286, i ovaj je bio manje-više kompatibilan sa svim prethodnicima sve do 8080. Ova kompatibilnost unazad bila je posebna pogodnost za korisnike zbog starijeg softvera koji je mogao da se pokreće bez problema.

Četiri godine kasnije izašao je 80486. To je u suštini bila brža verzija 80386 koji je takođe imao jedinicu sa pokretnim zarezom i 8 kilobajta keš memorije na čipu. Keš memorija se koristi za čuvanje najčešće korišćenih memorijskih reči unutar ili blizu CPU-a, kako bi se izbegli (spori) pristupi glavnoj memoriji.

80486 takođe je imao ugrađenu podršku za više procesora, omogućavajući proizvođačima da grade sisteme koji sadrže više procesora koji dele zajedničku memoriju.

Intel Pentium procesori

Sledeća generacija Intel-ovih čipova dobila je ime Pentium (gr. πέντε). Intel je dodao poseban skup instrukcija ***MMX*** (en. MultiMedia eXstension). Ove instrukcije su imale za cilj da ubrzaju potrebna izračunavanja za obradu audio i video zapisa, izbegavajući potrebu za posebnim ko-procesorom.

Novi čip zvao se Pentium Pro. Uprkos maloj promeni imena od svog prethodnika, ovaj procesor je predstavljao veliki napredak. Pentium Pro je imao veoma drugačiju unutrašnju organizaciju i mogao je da se izvrši do pet instrukcija odjednom.

Još jedna inovacija pronađena u Pentium Pro čipu bila je keš memorija na dva nivoa. Sam procesorski čip je imao 8 kilobajta brze memorije koja se obično koristi uputstva i još 8 kilobajta brze memorije za čuvanje često korišćenih podataka, kao i dodtanih 256 kilobajta keš memorije drugog nivoa, koja se nije nalazila na samom čipu. Iako je Pentium Pro imao veliku keš memoriju, nedostajale su MMX instrukcije. Kada se tehnologija dovoljno poboljšala da dobije i MMX instrukcije i keš memorije na jednom čipu, kombinovani proizvod je objavljen kao Pentium II.

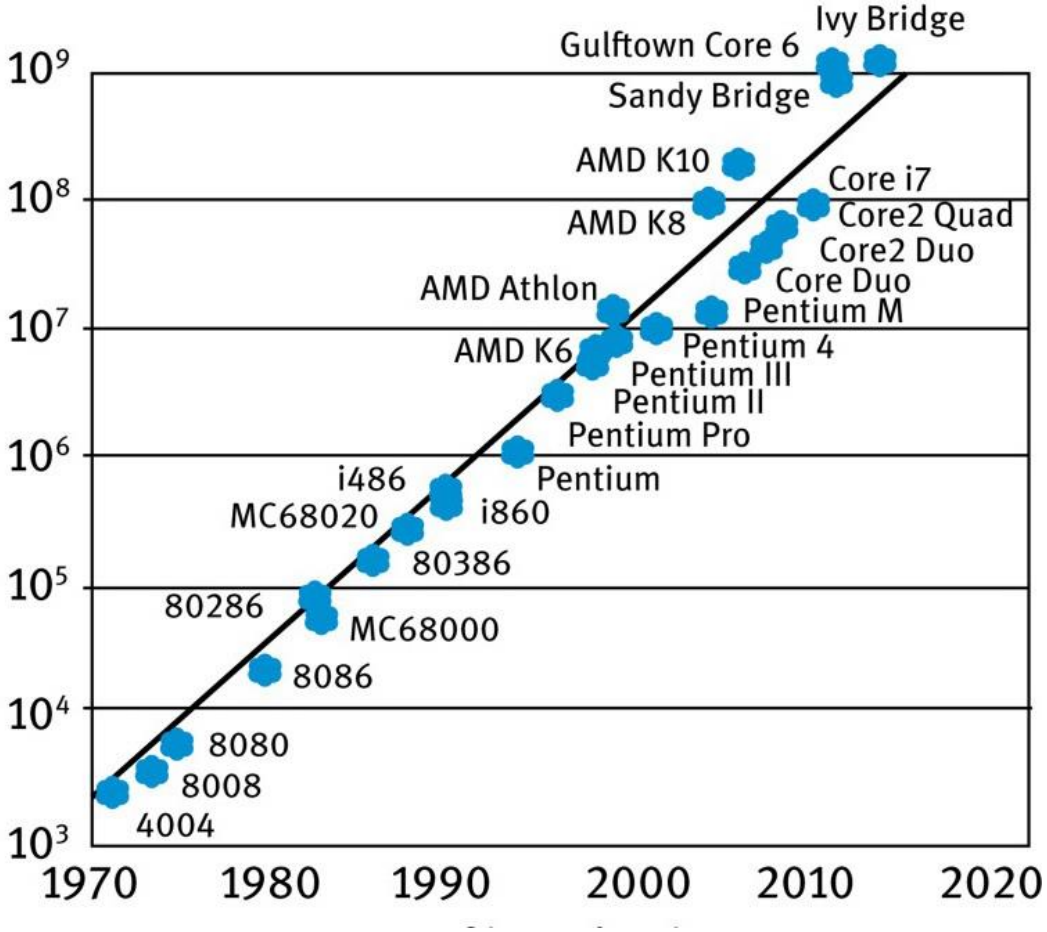
Procesor sa još više multimedijalnih instrukcija, nazvanih ***SSE*** (en. Streaming SIMD Exensions), dodate za poboljšani ras sa 3D grafikom. Novi čip je nazvan Pentium III, ali interno je u suštini bio Pentium II.

Sledeći Pentium, objavljen u novembru 2000. godine, bio je zasnovan na drugačijoj internoj arhitekturi, ali je imao isti skup instrukcija kao i raniji Pentium čipovi. Intel je prešao sa rimskih brojeva na arapske brojeve i nazvao ga Pentium 4. Pentium 4 je bio brži od svih svojih prethodnika. Verzija na 3,06 GHz je takođe predstavila novo svojstvo ovih procesora – hyperthreading. Ovo svojstvo je omogućilo programima da podele svoj rad u dve kontrolne niti koje bi Pentium 4 mogao izvršavati paralelno, ubrzavajući izvršenje. Pored toga, još jedan niz SSE instrukcija je dodat da bi se ubrzala obrada audio i video podataka.

Intel je 2006. promenio naziv brenda iz Pentium u Core i objavio je čip sa dva jezgra, nazvan Core 2 Duo.

Core serija je nastavila da se razvija, a danas je u ponudi već 13. generacija čipova Core i3, i5, i7 i i9, za računare niskih, srednjih i visokih performansi.

Evolucija Intel procesora



Slika-1 Murov zakon i Intel procesori po generacijama. [Izvor: https://www.ncbi.nlm.nih.gov/books/NBK321721/figure/oin_tutorial.F3/]

Evolucija Intel procesora (video):

https://www.youtube.com/embed/8MuaFp7zbUs

Ostale arhitekture

ARM arhitektura

Početkom 80-ih, britanska kompanija Acorn Computer, ispunjena uspehom njihovog 8-bitnog BBC Micro personalnog računara, počeli da rade na drugoj mašini sa nadom da će se takmičiti sa nedavno izdatim IBM PC-jem. BBC Micro je bio zasnovan na 8-bitnom 6502 procesoru, a Stiv Furber i njegove kolege u Acorn-u su smatrali da 6502 nema snagu da se takmiči sa IBM PC-jem sa 16-bitnim 8086 procesorom. Počeli su da razmatraju opcije na tržištu, i odlučili da su previše ograničeni.

Inspirisan Berkli RISC projektom, u kojem je mali tim dizajnirao a izuzetno brz procesor (što je na kraju dovelo do SPARC arhitekture), oni odlučili da naprave sopstveni CPU za projekat. Nazvali su svoj dizajn Acorn RISC mašina (ili ***ARM***, koja će kasnije biti prekrštena u Advanced RISC mašina kada se ARM na kraju odvoji od Acorn). Projektovanje je završeno 1985 godine, i uključuje 32-bitne instrukcije i podatke, i 26-bitni adresni prostor.

Prva ***ARM*** arhitektura (nazvana ARM2) pojavila se u Acorn Archimedes personalnom računaru. Archimedes je bio veoma brza i jeftina mašina za svoj dan, koja je radila i do 2 MIPS (milioni instrukcija u sekundi) i koštala je samo 899 britanskih funti na lansiranju.

Ovaj računar je postalo veoma popularan u Velikoj Britaniji, Irskoj, Australiji i Novom Zelandu, posebno u školama.

Na osnovu uspeha Archimedes, Apple je prišao Acorn-u da razvije ARM procesor za njihov predstojeći Apple Newton projekat, prvi tzv. palmtop računar. Da bi se bolje fokusirao na projekat, ARM arhitektonski tim je ostavio Acorn i oformio novu kompaniju pod nazivom Advanced ***RISC Machines*** (***ARM***).

Njihov novi procesor se zvao ARM 610, koji je pokretao Apple Newton, objavljen 1993. godine. Za razliku od originalnog ARM dizajna, ovaj novi ARM procesor posedovao je ugrađenu keš memoriju od 4 KB koja je značajno poboljšala performanse.

ARM arhitektura

Za razliku od mnogih računarskih kompanija, ARM ne proizvodi mikroprocesore. Umesto toga, ARM projektuje arhitekturu i alate i biblioteke za programere zasnovane na ARM-u, i licencira ih projektantima sistema i proizvođačima čipova.

ARM arhitektura je danas vodeća arhitektura za mobilne platforme, kao što su pametni telefoni i tablet uređaji, ali i novije generacije Apple procesora, počevši od M1, koristi ARM arhitekturu u svojim prenosivim laptop računarima.

https://www.youtube.com/embed/KGHdDVLnKJM

Pokazne vežbe

RADIX brojni sistemi

U svakodnevnoj primeni, decimalni brojevi se izražavaju u vidu skupa decimalnih cifara i decimalne tačke ili decimalnog zareza.

Decimalni brojevi su deo RADIX brojnog sistema čija je osnova 10 sa ciframa 0-9:

$$number = \sum_{i=-k}^{n} d_i \times 10^i$$

U opštem RADIX sistemu, broj number se može predstaviti kao suma proizvoda cifara d (sa nulom) i baze sistema b na stepenu koji ide od -k do n:

$$number = \sum_{i=-k}^{n} d_i \times b^i$$

Broj number u brojnom sistemu r može se napisati na sledeći način:

$$(number)_r$$

Primeri brojnih sistema

Najpopularniji brojni sistemi su:

Decimalni brojni sistem se sastoji od cifra- 0,1,2,3,4,5,6,7,8,9 i baze 10.

Binarni brojni sistem se sastoji od cifra- 0,1 i baze 2.

Oktalni brojni sistem se sastoji od cifra - 0,1,2,3,4,5,6,7 i baze 8

Heksadecimalni brojni sistem se sastoji od cifra - 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F i baze 16.

Brojni sistemi u računarima

Savremeni računarski sistemi koriste binarne (***BIN***), oktalne (***OCT***), dekadne (***DEC***) i heksadekadne (***HEX***) brojne sisteme za skladištenje i prikaz podataka.

Logička kola u računarima "pamte" naelektrisanje ili promenu napona čije vrednosti odgovaraju binarnom brojnom sistemu, dok se za prikaz informacija koriste i ostali sistemi.

Pored navedenih sistema, računari često koduju (en. encode) podatke koristeći ***binarno-kodovovane decimalne brojeve*** (en. Binary-Coded Decimal, ***BCD***), kod kojih se svaka cifra decimalnog brojnog sistema posebno konvertuje u binarni oblik, najčešće sa 4 ili sa 8 binarnih cifara.

Ukoliko se BCD broj koduje sa četiri binarne cifre, onda je u pitanju tzv. ***pakovani BCD broj*** (en. packed BCD number).

Ukoliko se BCD broj koduje sa osam binarnih cifara, onda je u putanju tzv. ***raspakovani BCD broj*** (en. unpacked BCD number)

Konverzija dekadnih brojeva u binarni broj

Dekadni broj se konvertuje u binarni broj tako što se dekadni broj iterativno deli brojem dva, pamteći ostatke od deljenja, sve dok je moguće.

Zatim, ostatke od deljenja poređati od poslednjeg ka prvom, i dobijeni broj je binarna reprezentacija dekadnog broja.

Primer #1 (7 minuta):

Konvertovati broj $(2023)_{10}$ u binarni.

Rešenje: $(2023)_{10} = (11111100111)_2$

```
-                ↙(ostatak)
-----------------------------------
2023:2 = 1011|1 (poslednja cifra)
1011:2 = 505 |1
505 :2 = 252 |1
252 :2 = 126 |0
126 :2 = 63  |0
63  :2 = 31  |1
31  :2 = 15  |1
15  :2 = 7   |1
7   :2 = 3   |1
3   :2 = 1   |1
1   :2 = 0   |1 (prva cifra)
```

Primer #2 (7 minuta):

Konvertovati broj $(2502)_{10}$ u binarni broj.

Rešenje:

$(2502)_{10} = (100111000110)_2$

Primer #3 (6 minuta):

Konvertovani broj $(1522)_{10}$ u binarni broj.

Rešenje:

$(1522)_{10} = (10111110010)_2$

Konverzija u oktalne i heksadecimalne brojeve

Zanimljivost:

Svaki broj jednog brojnog sistema je moguće konvertovati u broj drugog brojnog sistema iterativnim deljenjem osnove i beleženjem ostatka od deljenja.

Međutim, jednom konvertovani broj u binarni sistem se može jednostavnije konvertovati dalje u oktalni i heksadecimalni sistem.

Konverzija u oktalne brojeve

Kada imamo broj u binarnom brojnom sistemu, moguće je jednostavno konvertovati broj u oktalni brojni sistem tako što grupišemo grupe od po **tri binarne cifre** (počevši od bita najmanje težine - LSB bita). Svaka grupa od tri binarne cifre predstavljaće jednu cifru u OCT brojnom sistemu.

Primer #4 (7 minuta):

Konvertovati broj $(2023)_{10}$ u oktalni broj.

Rešenje:

Budući da već imamo binarnu reprezantaciju broja, potrebno je samo grupisati brojeve i pronaći bianrnu vrednost svake grupe:

```
|011|111|100|111|
  3   7   4   7
```

$(2023)_{10} = (11111100111)_{2} = (3747)_{8}$

Konverzija u heksadecimalne brojeve

Kada imamo broj u binarnom brojnom sistemu, moguće je jednostavno konvertovati broj u heksadecimalni brojni sistem tako što grupišemo grupe od **po četiri binarne cifre** (počevši od bita najmanje težine - LSB bita). Svaka grupa od četiri binarne cifre predstavljaće jednu cifru u HEX brojnom sistemu.

Primer #5 (7 minuta):

Konvertovati broj $(2023)_{10}$ u heksadecimalni broj.

Rešenje:

Budući da već imamo binarnu reprezantaciju broja, potrebno je samo grupisati brojeve i pronaći bianrnu vrednost svake grupe:

```
|0111|1110|0111|
  7    E    7
```

$(2023)_{10} = (11111100111)_2 = (7E7)_{16}$

Konverzija u BCD brojeve

Konverzija u BCD brojeve je jednostavnija od klasične konverzije u binarne brojeve jer je potrebno konvertovani cifru po cifru. BCD kodovanje uvek koduje decimalne brojeve.

Primer #8 (6 minuta):

Konvertovati broj 2023 u pakovani i raspakovani BCD format.

Rešenje:

```
Pakovani BCD: 0010 0000 0010 0011
----------------
2 - 0010
0 - 0000
2 - 0010
3 - 0011
----------------
Raspakovani BCD: 00000010 00000000 00000010 00000011
----------------
2 - 00000010
0 - 00000000
2 - 00000010
3 - 00000011
----------------
```

Kod pakovanih BCD brojeva svaka cifra koduje se sa pola bajta, odnosno sa četiri bita, dok se kod raspakovanih BCD brojeva svaka cifra koduje sa jednim bajtom, odnosno sa osam bitova.

Koristeći sajt RapidTables moguće je lako proveriti bilo koju konverziju brojnih sistema:

https://www.rapidtables.com/convert/number/hex-dec-bin-converter.html

Projektni zadatak iz predmeta CS120

Na predmetu CS120 - Organizacija računara svaki student samostalno radi projektni zadatak (PZ), koji se sastoji od tri mini-projekta.

Mini-projekti brane se u petoj, desetoj i petnaestoj nedelji semestra!

Svaki student dužan je da tok izrade projekta izveštava svake nedelje u obliku kratkih izveštaja (do jednog pasusa) preko LAMS lekcija u okviru dodatnih aktivnosti interaktivnih lekcija.

O formatu mini-projekata svi studenti (tradicionalni i Internet) biće obavešteni mejlom od strane predmetnog profesora/asistenta.

Zadaci za samostalni rad

Dodatni primeri za samostalno vežbanje

(20 minuta) Samostalno konvertovati sledeće brojeve:

$(2048)_{10} = (X)_2$
$(100101110111)_2 = (X)_8$
$(110100110010)_2 = (X)_{16}$
$(5FC)_{16} = (X)_8$
$(5FC)_{16} = (X)_2$
$(ABC)_{16} = (X)_{10}$
$(911)_{16} = (X)_{10}$
$(802)_{10} = (X)_8$

(20 minuta) Koristeći opšti izraz za RADIX brojne sisteme, konvertovati sledeće brojeve:

$(20.25)_{10} = (X)_2$
$(103.5)_{10} = (X)_2$
$(506.125)_{10} = (X)_2$
$(704)_{10} = (X)_6$
$(1563)_{10} = (X)_5$

Python zadaci za samostalni rad (40 minuta) :

Napisati rezurzivnu funkciju koja konvertuje uneti pozitivan ceo dekadni broj u binarni broj. Ne koristiti ugrađene ili eksterne funkcije konvertovanja.

Napisati funkciju koja konvertuje uneti binarni broj u oktalni i heksadecimalni broj. Ne koristiti ugrađene ili eksterne funkcije konvertovanja.

Domaći zadatak

Domaći zadatak #1

Domaći zadatak #1, zadatak #1

Uraditi sledeće konverzije:

$$(\text{broj\_indeksa})_{10} = (X)_2$$
$$(\text{broj\_indeksa})_{10} = (X)_5$$
$$(\text{broj\_indeksa})_{10} = (X)_8$$
$$(\text{broj\_indeksa})_{10} = (X)_{16}$$

Domaći zadatak #1, zadatak #2

Uraditi sledeće konverzije:

$$(\text{broj\_indeksa})_{16} = (X)_2$$
$$(\text{broj\_indeksa})_{16} = (X)_7$$
$$(\text{broj\_indeksa})_{16} = (X)_8$$
$$(\text{broj\_indeksa})_{16} = (X)_{10}$$

U oba zadatka broj_indeksa predstavlja Vaš broj indeksa.

Konverzije uraditi ručno (na papiru) prateći postupke konverzije prema pokaznim vežbama. Screenshot-ove urađenih zadataka poslati prema uputstvu za predaju domaćih zadataka.

Predaja domaćeg zadataka

Tradicionalni studenti:

Domaći zadatak treba dostaviti najkasnije 7 dana nakon vežbi, za 100% poena. Nakon toga poeni se umanjuju za 50%.

Domaći zadatak poslati predmetnim asistentima, sa predmetnim profesorima u CC.

Internet studenti treba poslati domaće zadatke najkasnije do 10 dana pred izlazak na ispit predmetnom asistentu zaduženog za internet studente.

Napomena:

Svaki domaći zadatak treba da bude napisan prema dokumentu za predaju domaćih zadataka koji je dat na kraju interaktivne lekcije.

Zaključak

U lekciji #1 najpre je bilo reči o pravilima vezanih za predmet i predispitnim obavezama studenata.

Zatim, uvedeni su pojmovi jezika, nivoa i mašina u kontekstu računarskih sistema.

Pokazana je struktuirana višenivoovska organizacija savremenih računarskih sistema uz osvrt na svaki nivo.

Objašnjene su generacije računarskih sistema, od primitivnih računskih mašina do računara pete generacije.

Konačno, data je kratka istorija Intel x86 arhitekture kao osnove savremenih računarskih sistema, uz spominjanje i alternativnih arhitektura kao što je ARM.

Literatura:

1. A. Tanenbaum, Structured Computer Organization, Chapter 01, pp. 26 – 54, Appendix A, pp. 669 – 681