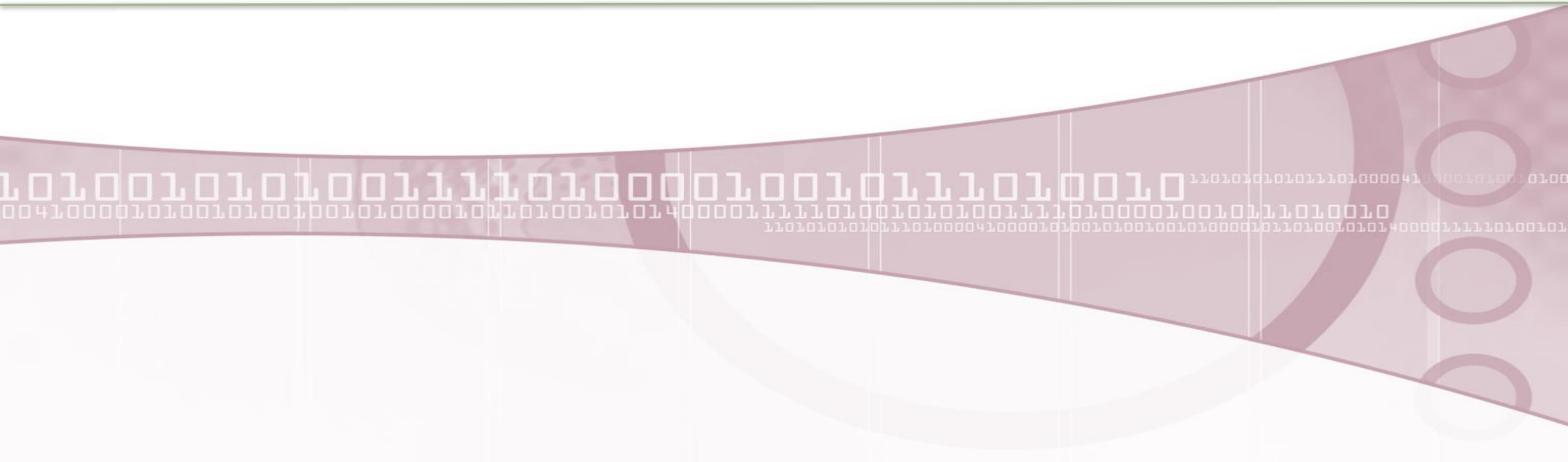
STRUKTURA I FUNKCIJE RAČUNARSKOG SISTEMA

Deo 5



SADRŽAJ

- Struktura računarskog sistema
  - Fon Nojmanova mašina
- Mašinske instrukcije
- Struktura centralnog procesora
  - Aritmetičko-logička jedinica
  - Upravljačko-kontrolna jedinica
- Memorija

STRUKTURA RAČUNARSKOG SISTEMA

Svaki računarski sistem treba da obezbedi:

- Mogućnost učitavanja ulaznih podataka
- Obradu podataka
- Čuvanje međurezultata obrade
- Čuvanje i/ili prikazivanje rezultata na nekom spoljašnjem medijumu

Njegova struktura se u najvećoj meri podudara sa fon Nojmanovom arhitekturom.

FON NOJMANOVA ARHITEKTURA (1945. GODINE)

- … dosta često da izvodi osnovne aritmetičke operacije. Stoga je razumljivo da treba da sadrži specijalizovane organe koji obavljaju te operacije... centralni aritmetički organ, CA računara.

- Centralno upravljanje - logička kontrola uređaja…mora da postoji razlika između specifičnih instrukcija koje definišu problem i služe za njegovo rešavanje i opštih, za upravljačke organe koji nadgledaju kako se izvode te specifične instrukcije...

- Bilo koji uređaj koji izvodi dugačak i komplikovan niz operacija izračunavanja mora da ima odgovarajuću memoriju za instrukcije i podatke

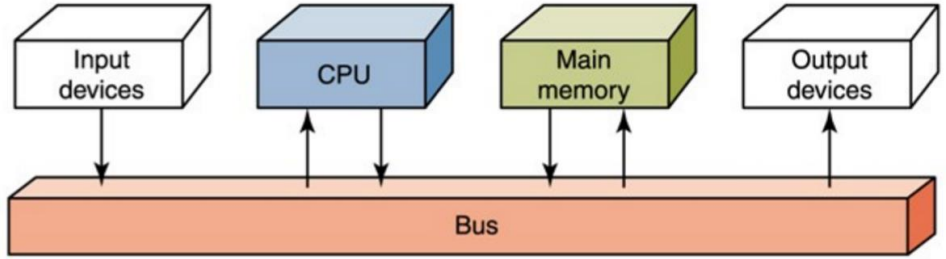
- Uređaj mora da podržava ulaz i izlaz podataka

SAVREMENI RAČUNARSKI SISTEM

Fon Nojmanova arhitektura se zasniva na tri ključna koncepta:

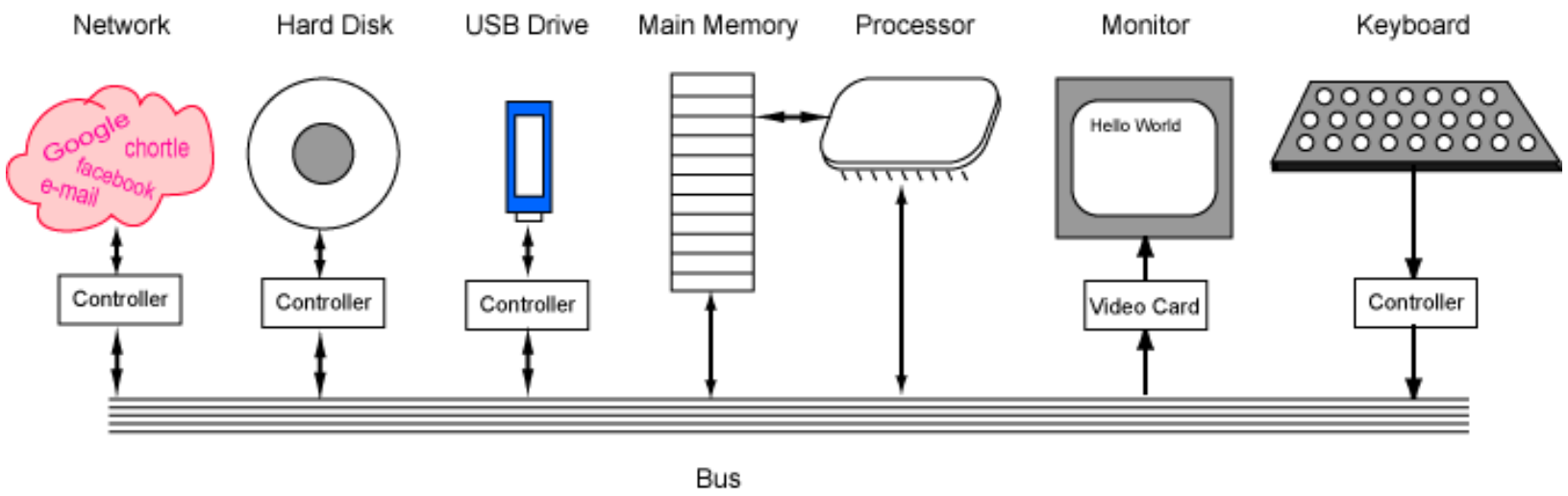
- Podaci i instrukcije se pohranjuju u jednoj memoriji za čitanje i pisanje.
- Sadržaj ove memorije se može adresirati po lokaciji, bez obzira na tip podataka koji su tamo sadržani.
- Instrukcije se (osim ako ovo nije eksplicitno izmenjeno) izvršavaju sekvencijalno – jedna za drugom.

Za razliku od Fon Nojmanove arhitekture, spoljašnji uređaji savremenih računarskih sistema mogu da komuniciraju direktno s memorijom.

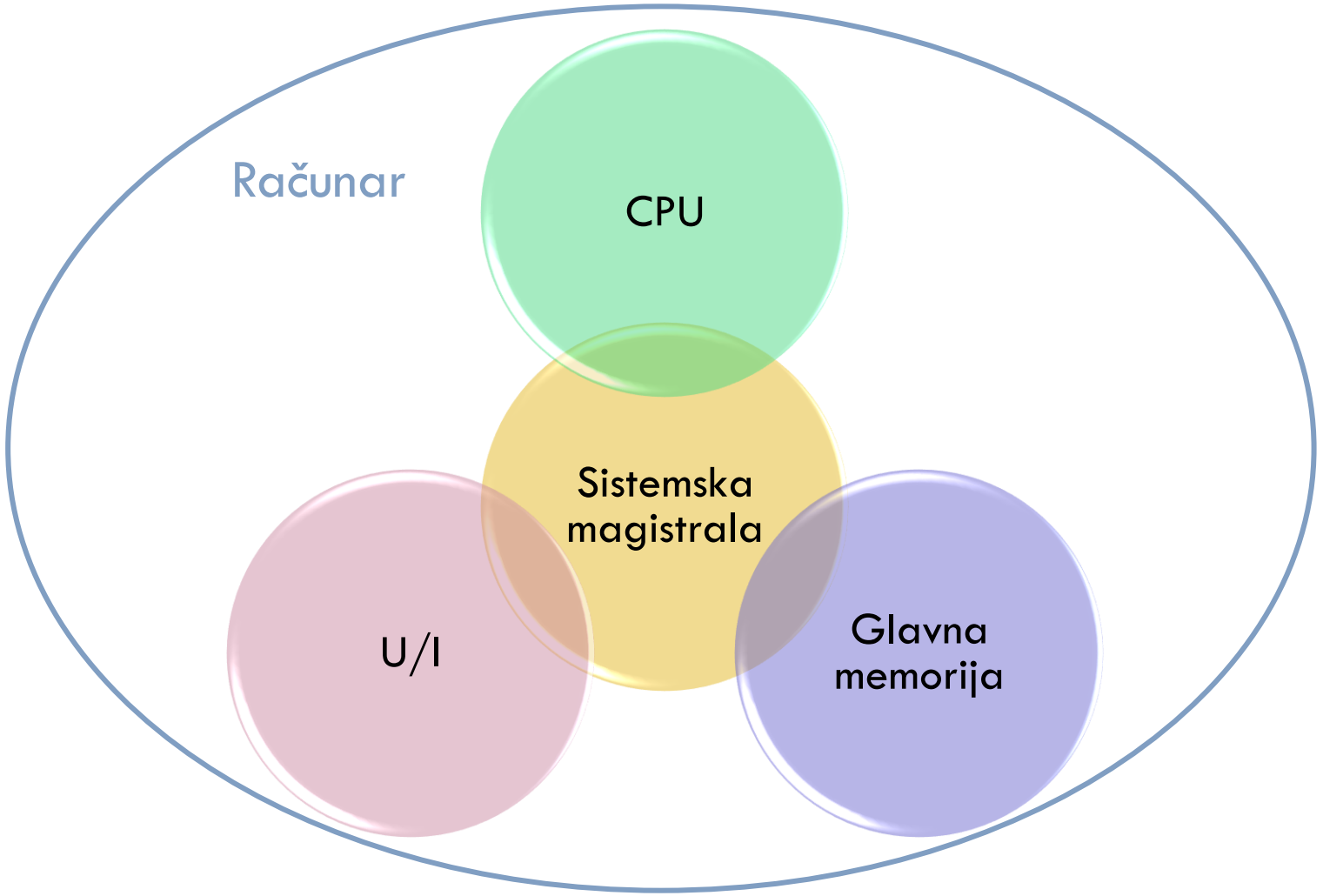


STRUKTURA RAČUNARSKOG SISTEMA

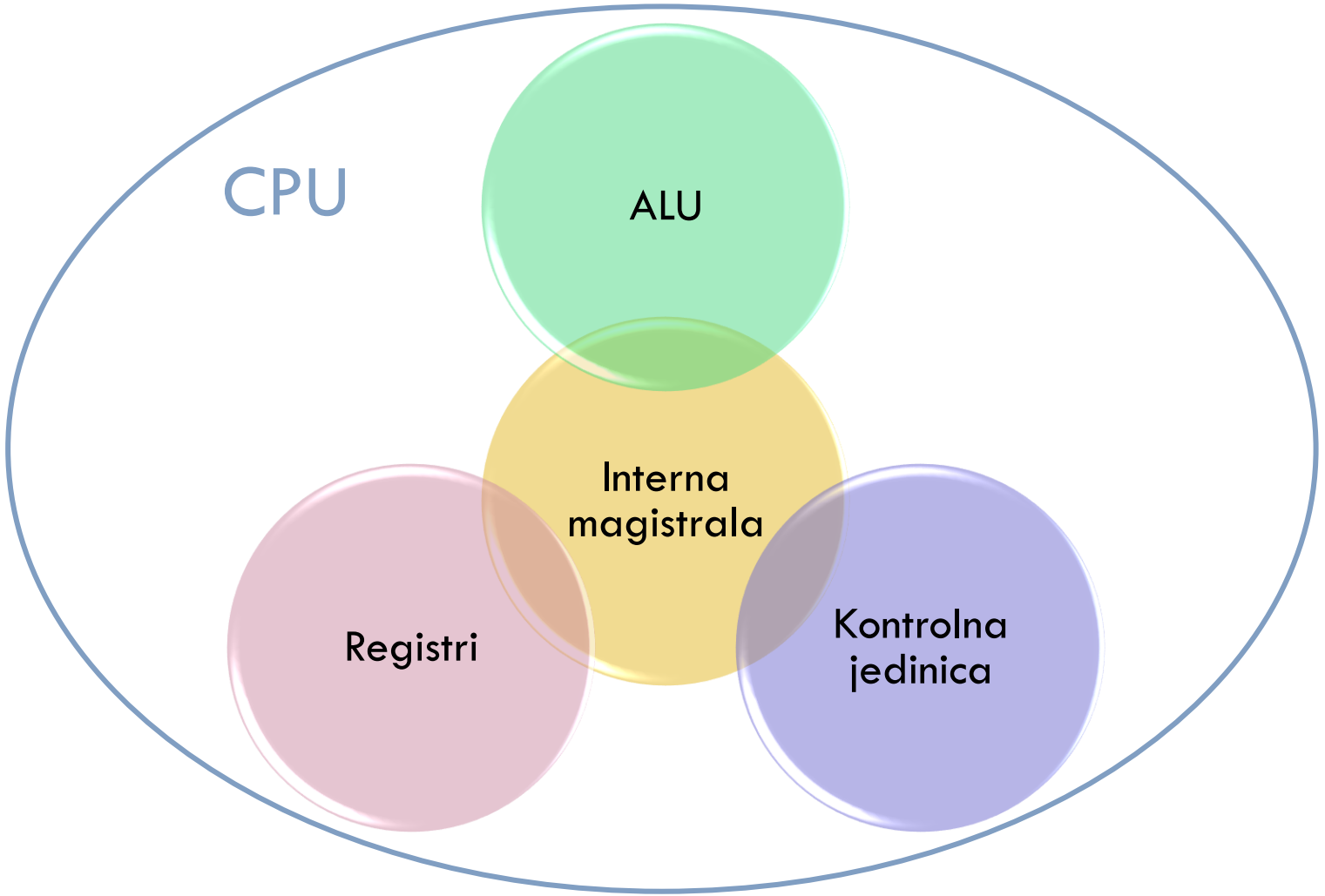
- Centralna procesorska jedinica (*CPU – Central Processing Unit*)
  - Vrši obradu podataka
  - Kontroliše operacije koji se izvode u računarskom sistemu
- Unutrašnja i spoljašnja memorija
  - Čuvaju podatke privremeno ili trajno
- Ulazno/izlazni uređaji
  - Prenose podatke između računarskog sistema i okruženja
- Magistrale
  - Koriste se za komunikaciju između procesora, ulazno/izlaznih uređaja i memorije
- Softver
  - Omogućava rad računarskog sistema i obezbeđuje interfejs prema korisnicima



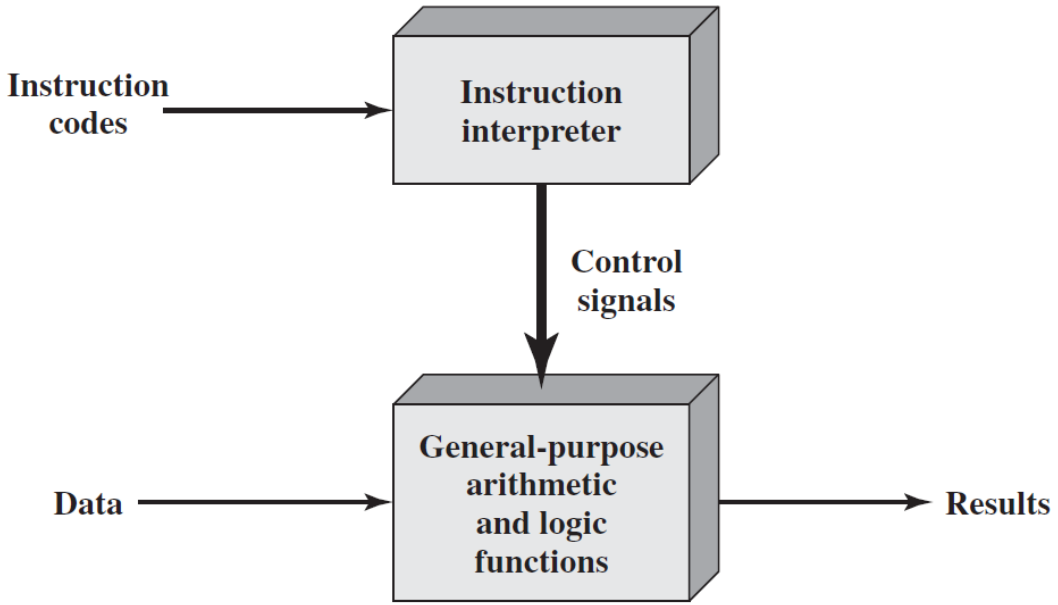
STRUKTURA RAČUNARSKOG SISTEMA NA NAJVIŠEM NIVOU



STRUKTURA CENTRALNOG PROCESORA NA NAJVIŠEM NIVOU

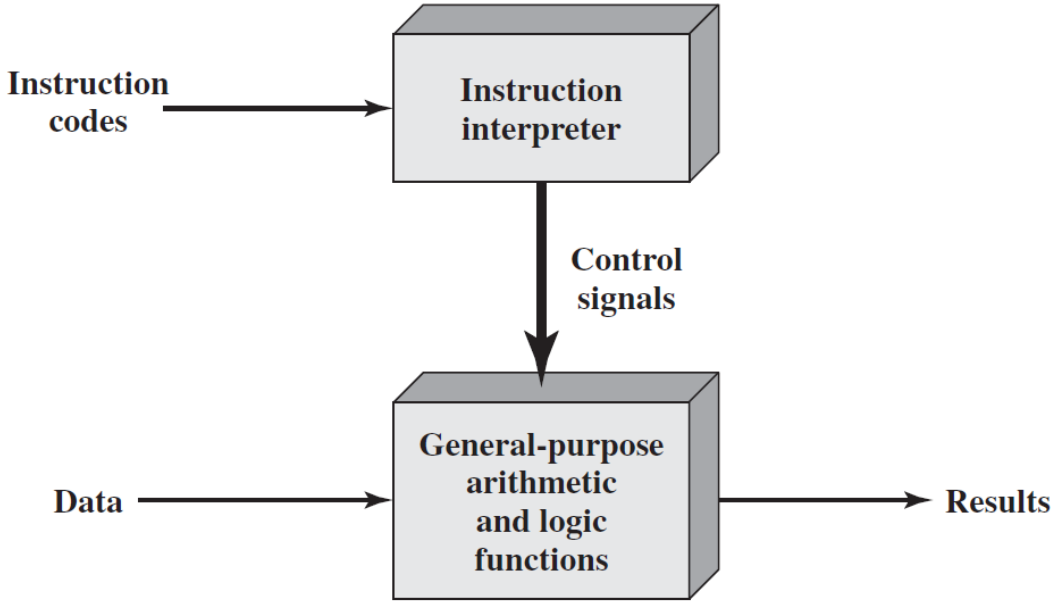


KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



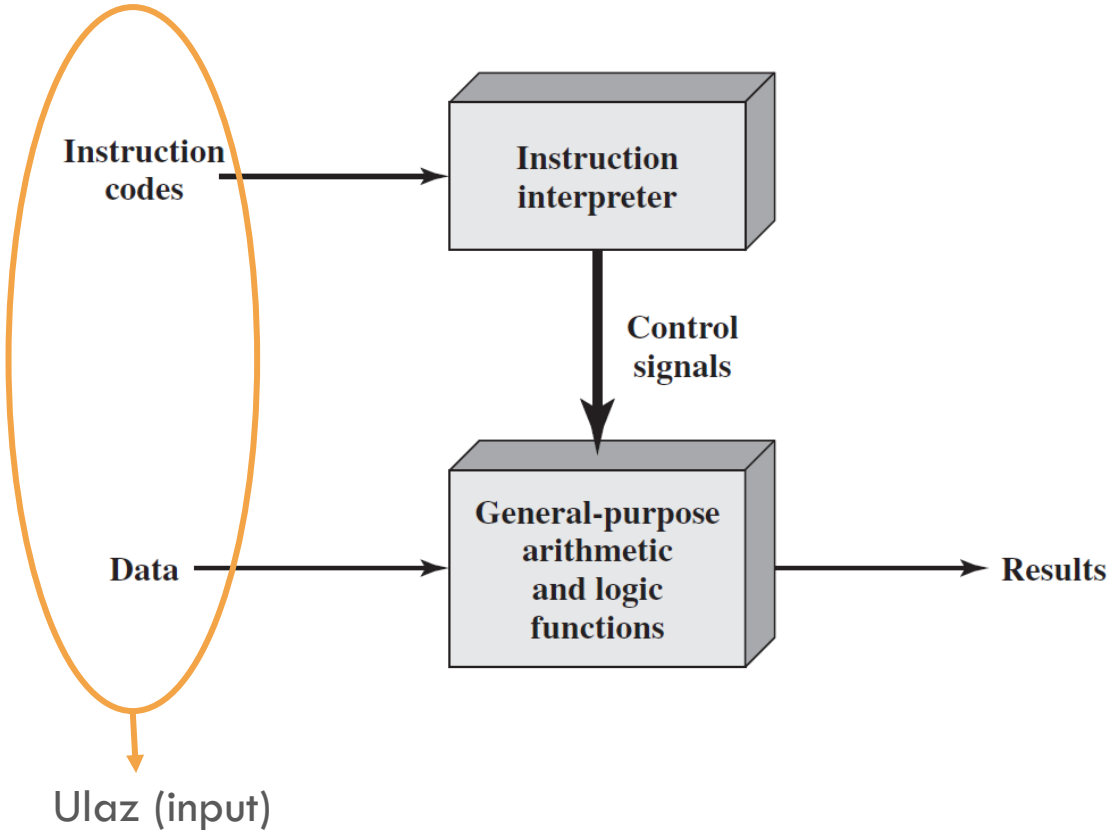
- Računarski sistem izvršava programe.
- Program je niz koraka. Na svakom koraku, pojedine aritmetičke ili logičke operacije se obavljaju nad određenim podacima.
- Operacije koje treba obaviti date su u vidu kodiranih instrukcija.
- Deo hardvera interpretira instrukcije programa i generiše kontrolne signale, što rezultira izvršavanjem ovih instrukcija.

KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



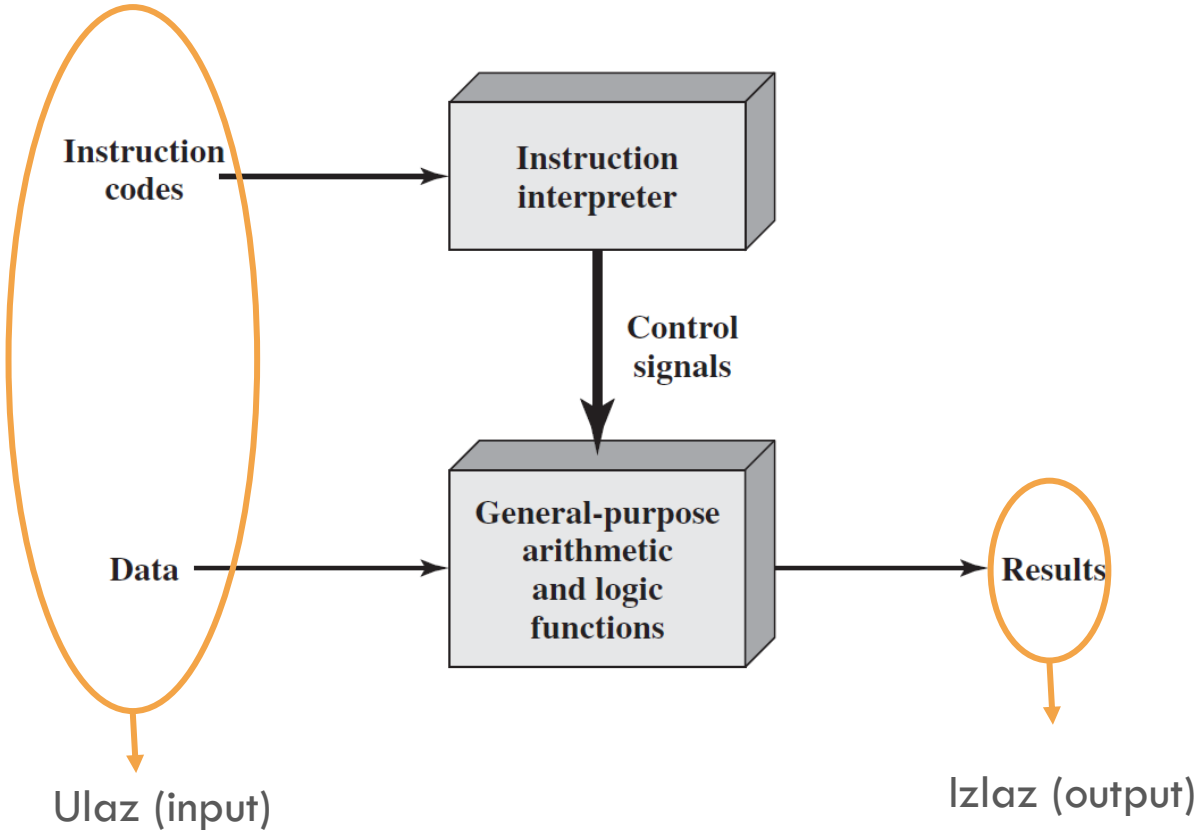
- Instrukcije i podaci moraju biti „uneti“ u sistem – ulazni modul.

KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



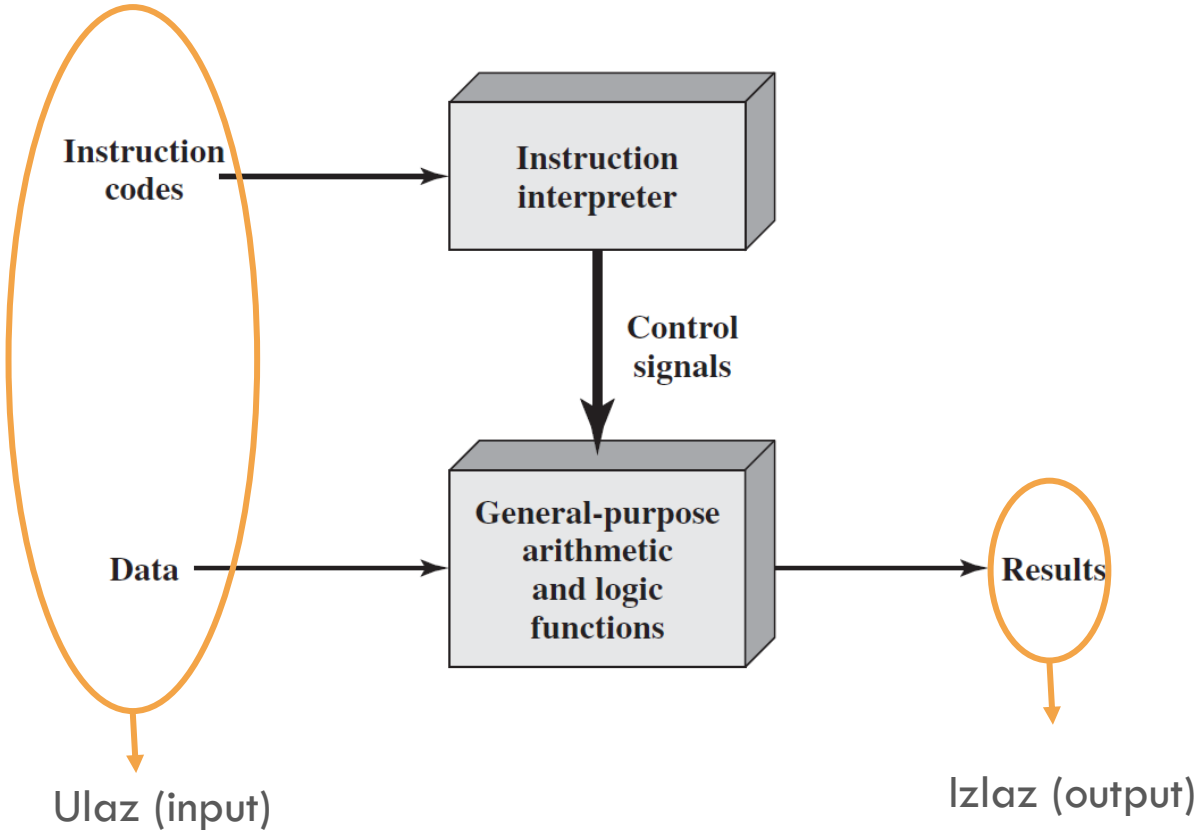
- Instrukcije i podaci moraju biti „uneti“ u sistem – ulazni modul.

KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



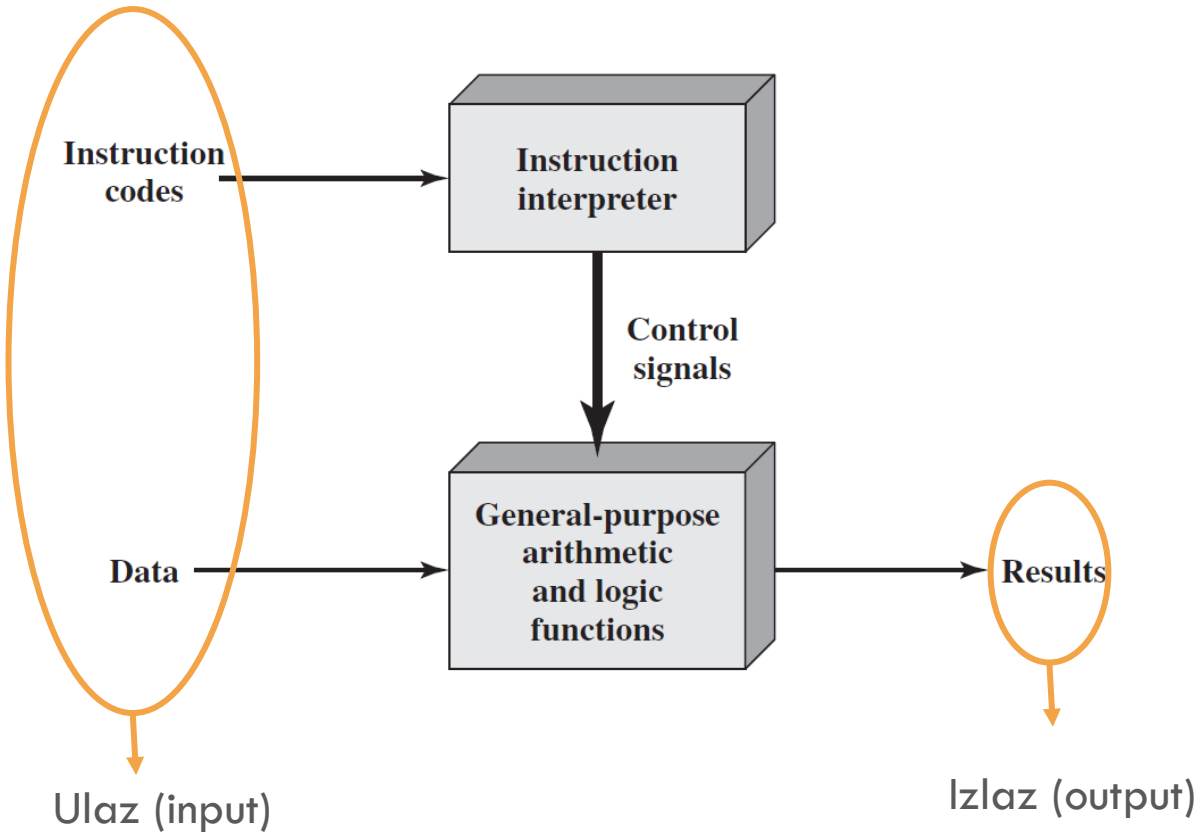
- Instrukcije i podaci moraju biti „uneti“ u sistem – ulazni modul.
- Prikazivanje rezultata – izlazni modul.

KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



- Instrukcije i podaci moraju biti „uneti“ u sistem – ulazni modul.
- Prikazivanje rezultata – izlazni modul.
- Delovi sistema za unos instrukcija i podataka, i za prikaz rezultata zajedno čine **U/I komponente** (*I/O components*).

KAKO FUNKCIONIŠE RAČUNARSKI SISTEM?



- Instrukcije i podaci moraju biti „uneti“ u sistem – ulazni modul.
- Prikazivanje rezultata – izlazni modul.
- Delovi sistema za unos instrukcija i podataka, i za prikaz rezultata zajedno čine **U/I komponente** (*I/O components*).
- Mesto gde će se pohranjivati instrukcije, podaci, međurezultati i rezultati – glavna memorija (main memory).

CPU – STRUKTURA I FUNKCIONALNOSTI

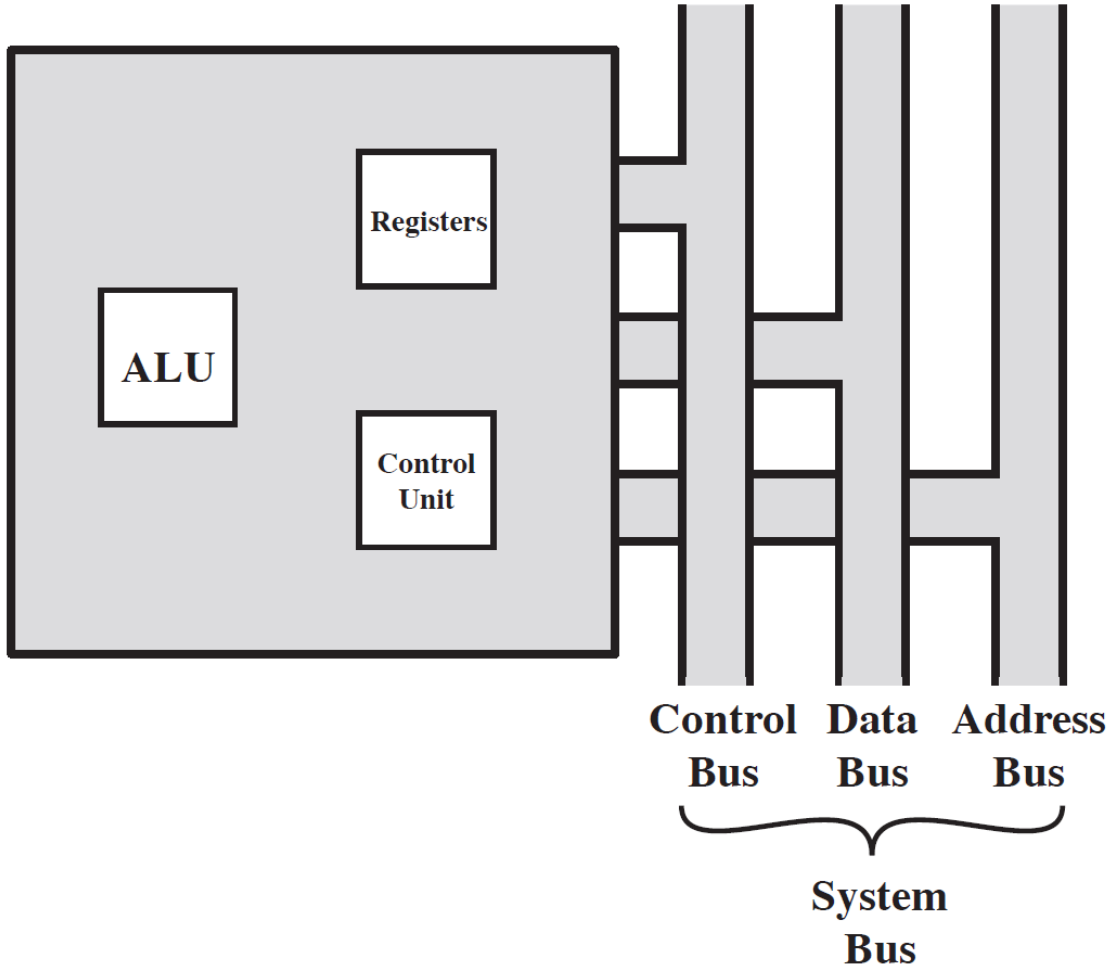
CPU mora:

- **Dohvatiti instrukcije**. Procesor čita instrukcije iz memorije (registra, keš memorije, glavne memorije)
- **Interpretirati instrukcije.** Instrukcija se dekodira da bi bilo utvrđeno koju akciju zahteva.
- **Dohvatiti podatke.** Izvršavanje instrukcije može zahtevati čitanje podataka iz memorije ili sa I/O modula.
- **Procesirati podatke.** Izvršavanje instrukcije može zahtevati neku vrstu aritmetičkih ili logičkih operacija nad podacima.
- **Upisivati podatke.** Rezultat izvršavanja može zahtevati upisivanje podataka u memoriju ili na I/O modul.

STRUKTURA CPU

Osnovne komponente CPU:

- Aritmetičko-logička jedinica ALU
- Kontrolna jedinica CU
- Registri
- Sistemska magistrala (Bus)
  - Magistrala za prenos kontrolnih signala
  - Magistrala za prenos podataka
  - Magistrala za prenos adresa



PROCESORSKI REGISTRI

- Procesorski registar je interna memorija centralnog procesora.
- Procesor tokom izvršavanja instrukcija koristi procesorske registre.
- Osnovne karakteristike ove vrste memorije su velika brzina, ali mali kapacitet.
- Procesorski registri su na vrhu memorijske hijerarhije i najbrži su način za pristup podatku u računaru.

PROCESORSKI REGISTRI - PODELA

- **Registri vidljivi korisnicima**: optimalno korišćenje ovih registara obezbeđuje minimalno referenciranje na glavnu memoriju u radu programa na mašinskom jeziku ili asembleru.

- **Kontrolni i statusni registri**: koristi ih kontrolna jedinica da kontroliše rad procesora kao i privilegovani programi operativnog sistema radi kontrole izvršavanja programa.

REGISTRI VIDLJIVI KORISNICIMA

- **Registri opšte namene** (General Purpose) – mogu sadržati operande nad kojima treba vršiti operacije (akumulator AC).
- **Registri podataka** (Data) – sadrže samo podatke i ne mogu se koristiti za određivanje adresa operanada.
- **Adresni registri** (Address)
- Registri koji sadrže adrese moraju biti barem dovoljno dugi da sadrže najveću adresu. Registri za podatke bi trebalo da mogu da sadrže vrednosti većine tipova podataka.

REGISTRI DELIMIČNO VIDLJIVI KORISNICIMA

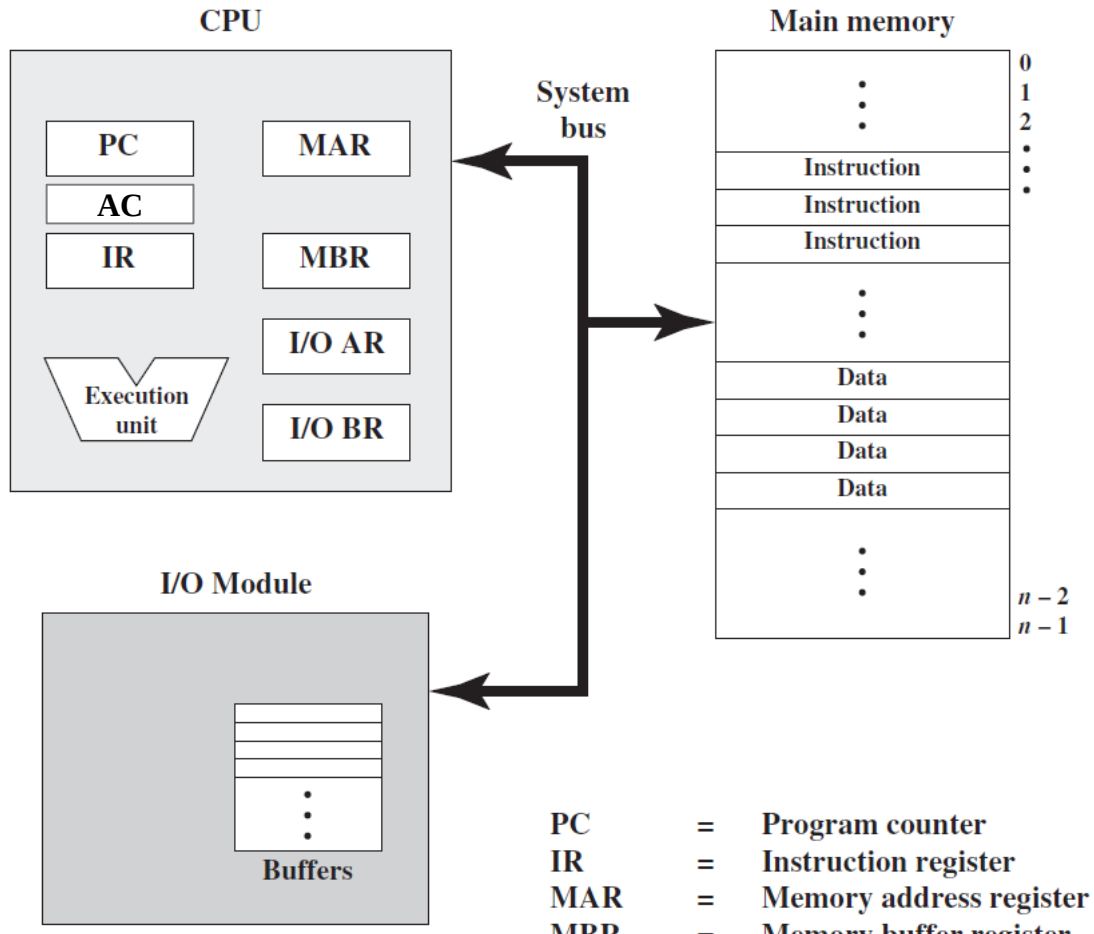
- **Registri uslovnih kodova** (*Condition Codes, Flags*) - bitovi čije vrednosti postavlja hardver procesora kao rezultat operacija.
- Na primer, aritmetička operacija može dovesti do pozitivnog, negativnog, nultog ili *overflow* rezultata. Osim samog rezultata koji se čuva u registru ili memoriji, takođe se postavljen i uslovni kod. Kod može kasnije biti testiran kao deo operacije u uslovnoj grani.

REGISTRI SPECIJALIZOVANE NAMENE KONTROLNI I STATUSNI REGISTRI

- **Brojač instrukcija (Program Counter, PC)** – sadrži adresu instrukcije koja treba da bude dopremljena.
- **Registar instrukcija (Instruction Decoding Register, IR)** – sadrži poslednju dopremljenu instrukciju.
- **Memorijski adresni registar (Memory Address Register, MAR)** – sadrži adresu memorijske lokacije.
- **Memorijski bafer registar (Memory Buffer Register, MBR)** – sadrži podatak koji treba da bude upisan u memoriju, ili podatak koji je poslednji pročitan iz memorije.
- **Program Status Word, PSW** – registar procesora koji sadrži status programa koji se trenutno izvršava: razne indikatore (znak poslednje aritmetičke operacije, prenos, overflow), uslovne kodove, itd.

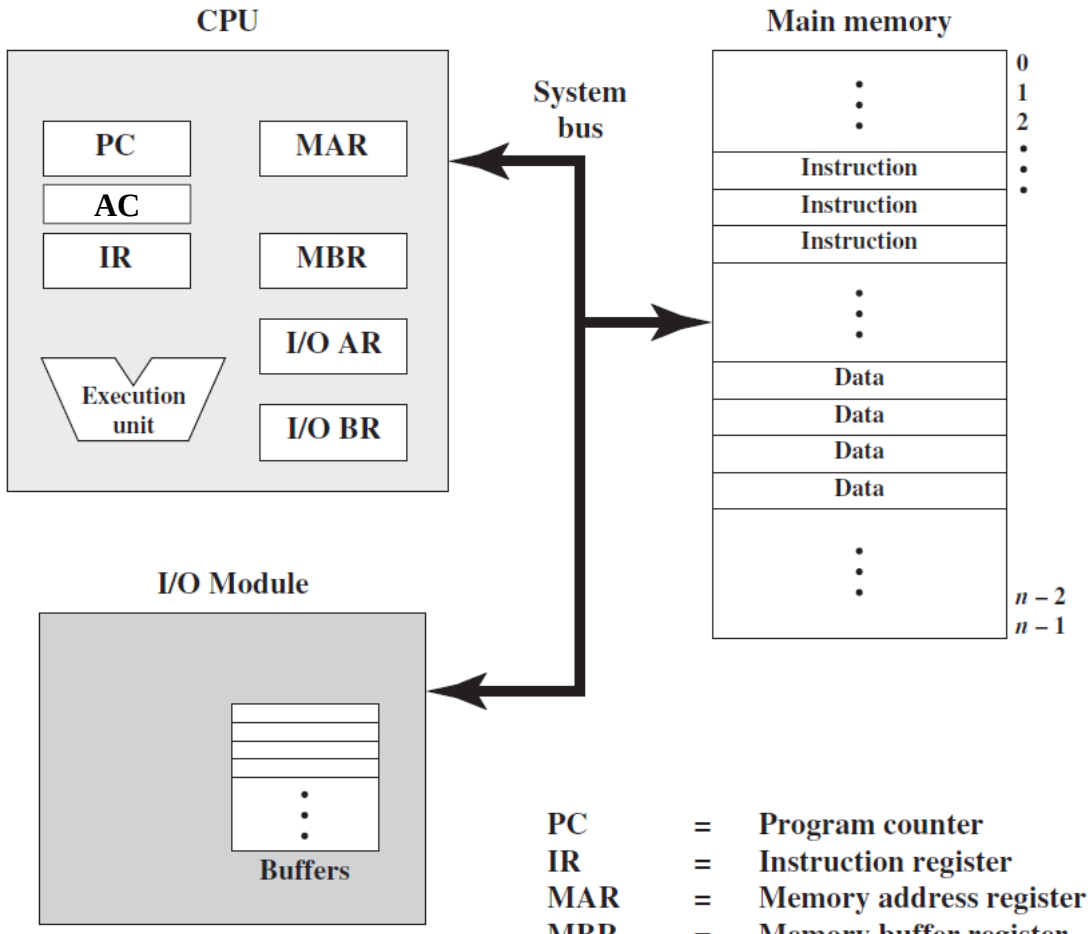
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:



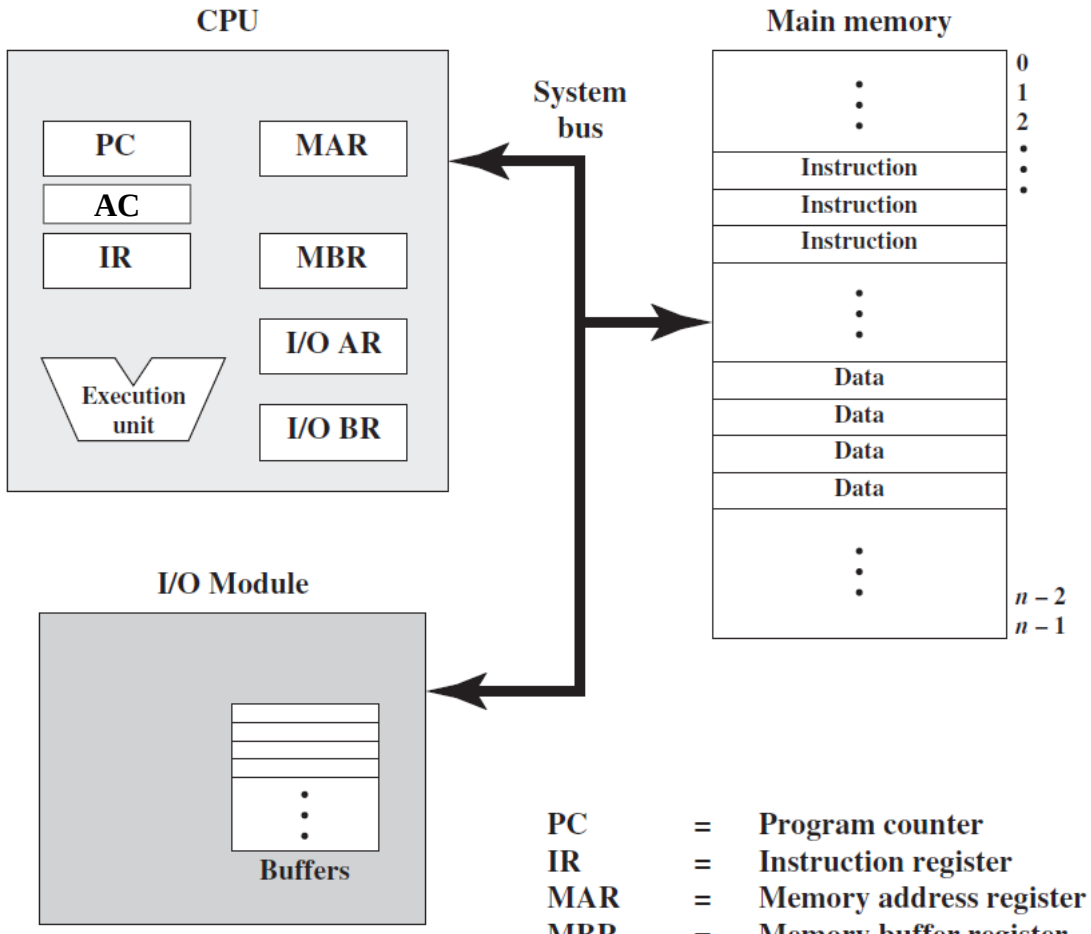
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.



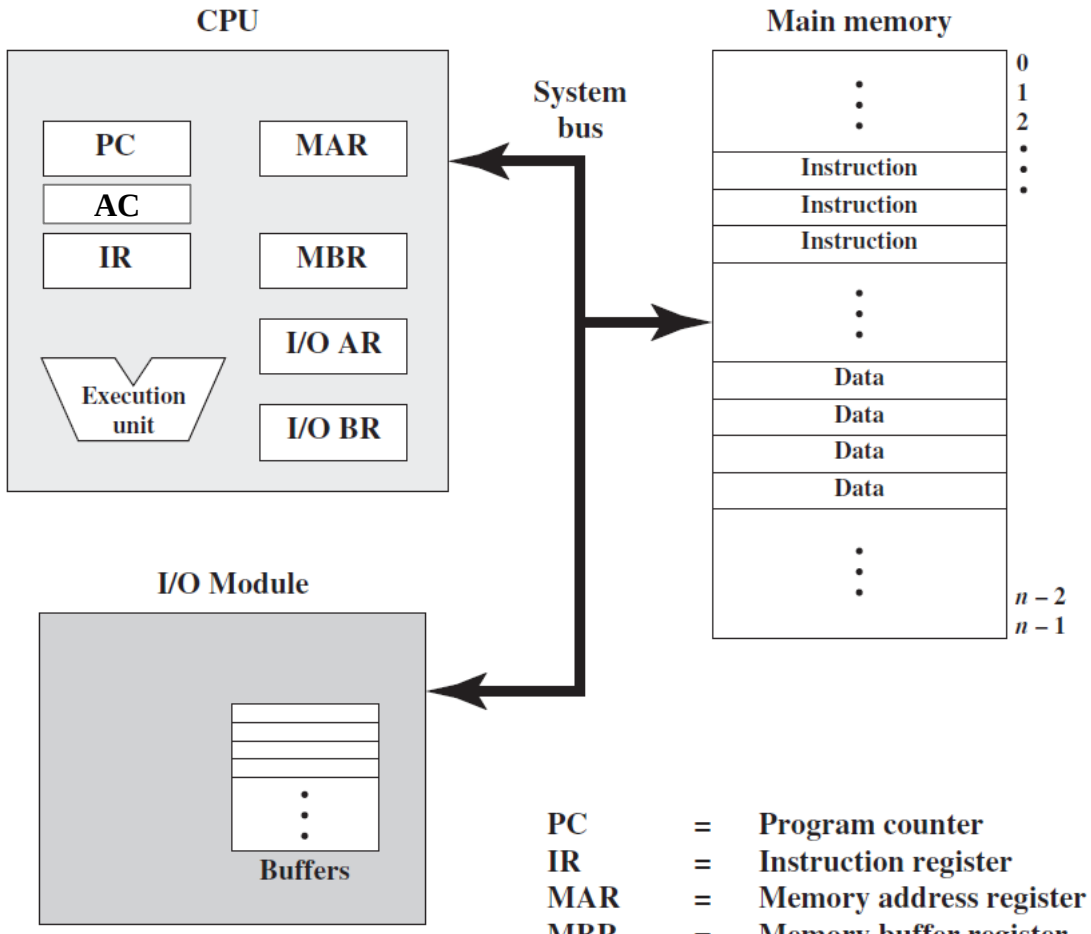
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.
  2. Memorijski bafer registar (MBR) – sadrži podatke koji su pročitani iz memorije ili koji treba da budu upisani u memoriju.



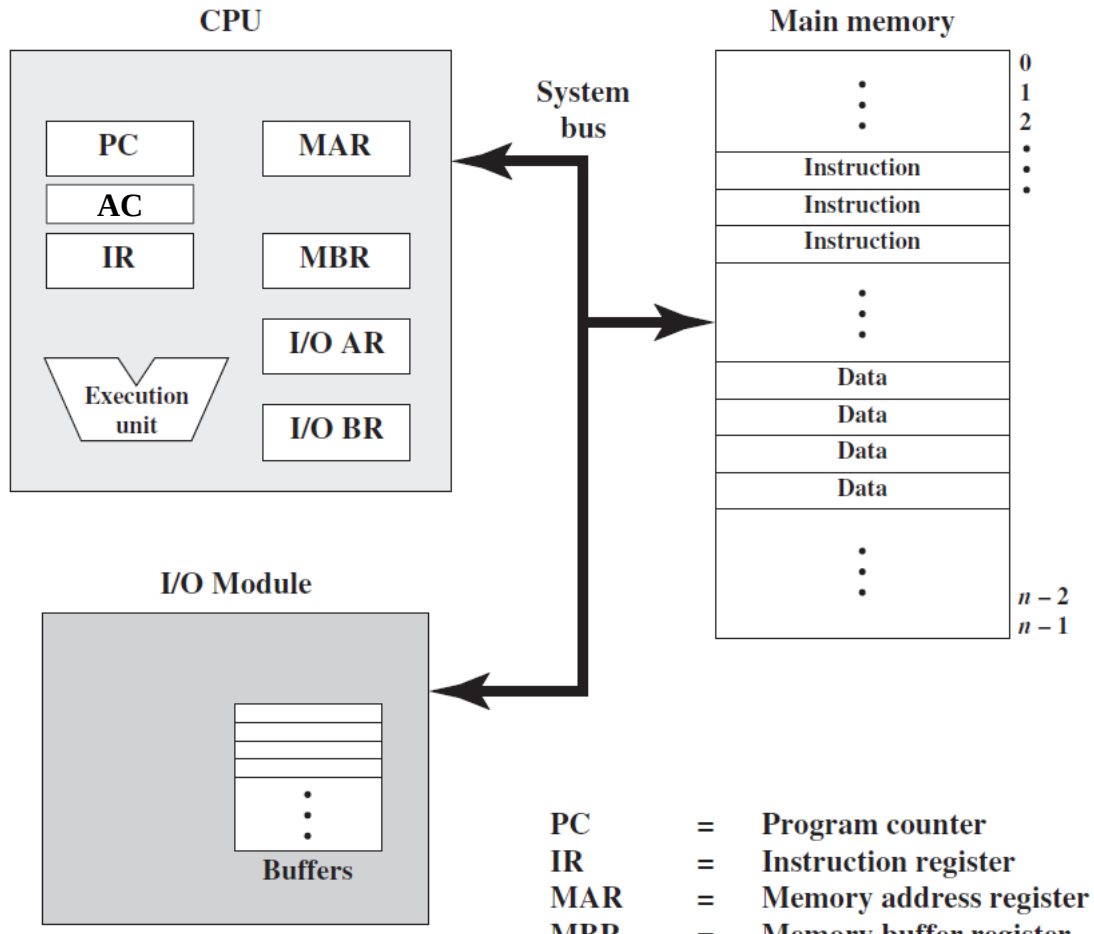
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.
  2. Memorijski bafer registar (MBR) – sadrži podatke koji su pročitani iz memorije ili koji treba da budu upisani u memoriju.
- I/O adresni registar određuje sa kog ulaznog uređaja se učitavaju podaci ili instrukcije, odnosno na koji izlazni uređaj se prosleđuju rezultati.



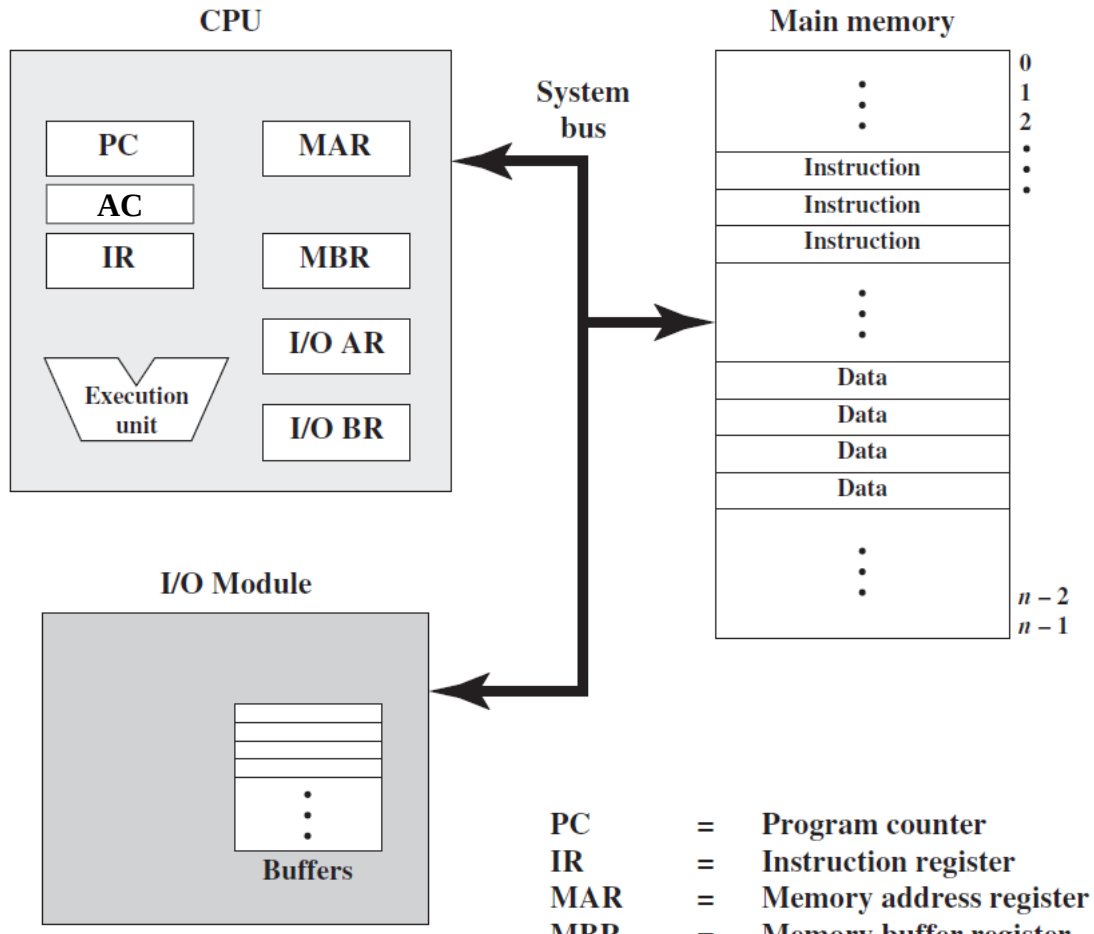
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.
  2. Memorijski bafer registar (MBR) – sadrži podatke koji su pročitani iz memorije ili koji treba da budu upisani u memoriju.
- I/O adresni registar određuje sa kog ulaznog uređaja se učitavaju podaci ili instrukcije, odnosno na koji izlazni uređaj se prosleđuju rezultati.
- I/O bafer registar sadrži podatke, instrukcije ili rezultate koji se učitavaju/prosleđuju sa ulaznog uređaja, odnosno na izlazni uređaj.



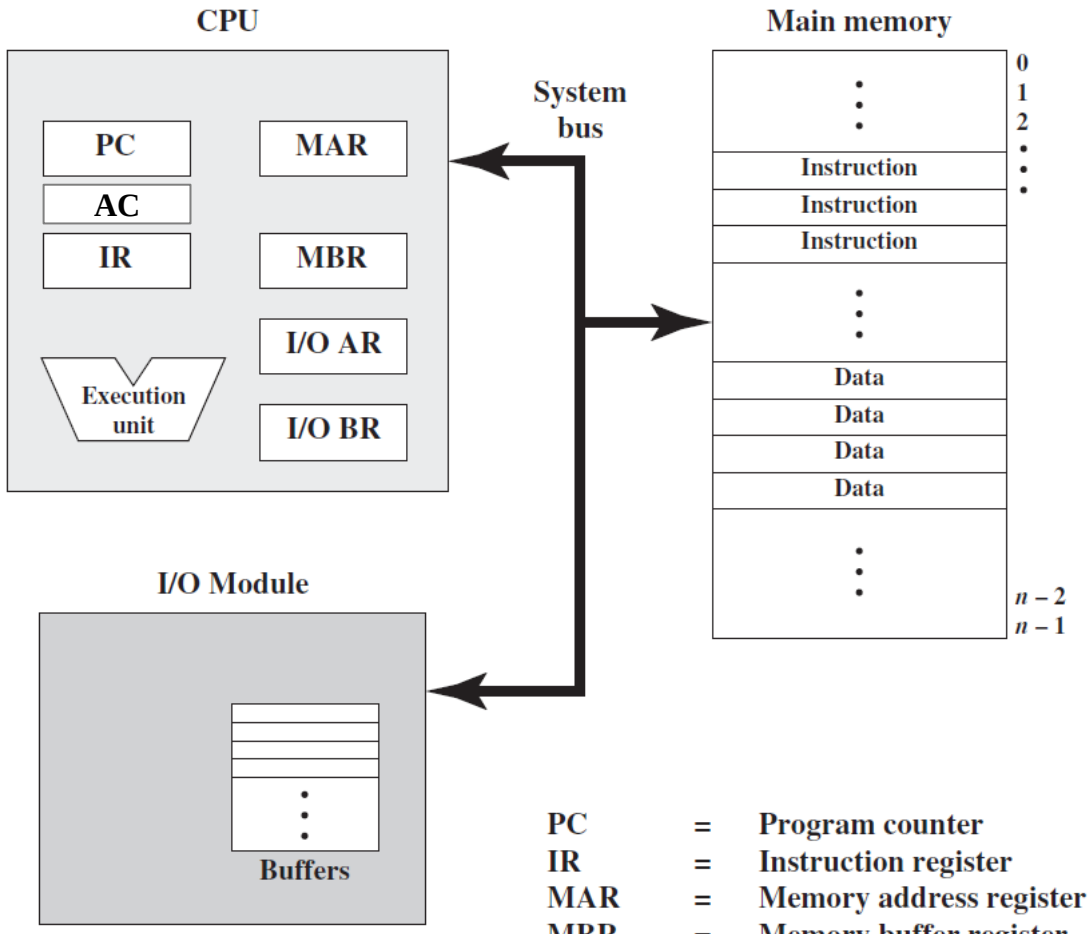
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.
  2. Memorijski bafer registar (MBR) – sadrži podatke koji su pročitani iz memorije ili koji treba da budu upisani u memoriju.
- I/O adresni registar određuje sa kog ulaznog uređaja se učitavaju podaci ili instrukcije, odnosno na koji izlazni uređaj se prosleđuju rezultati.
- I/O bafer registar sadrži podaci, instrukcije ili rezultate koji se učitavaju/prosleđuju sa ulaznog uređaja, odnosno na izlazni uređaj.
- Registar instrukcije (IR) sadrži operacioni kod instrukcije koja se trenutno izvršava.



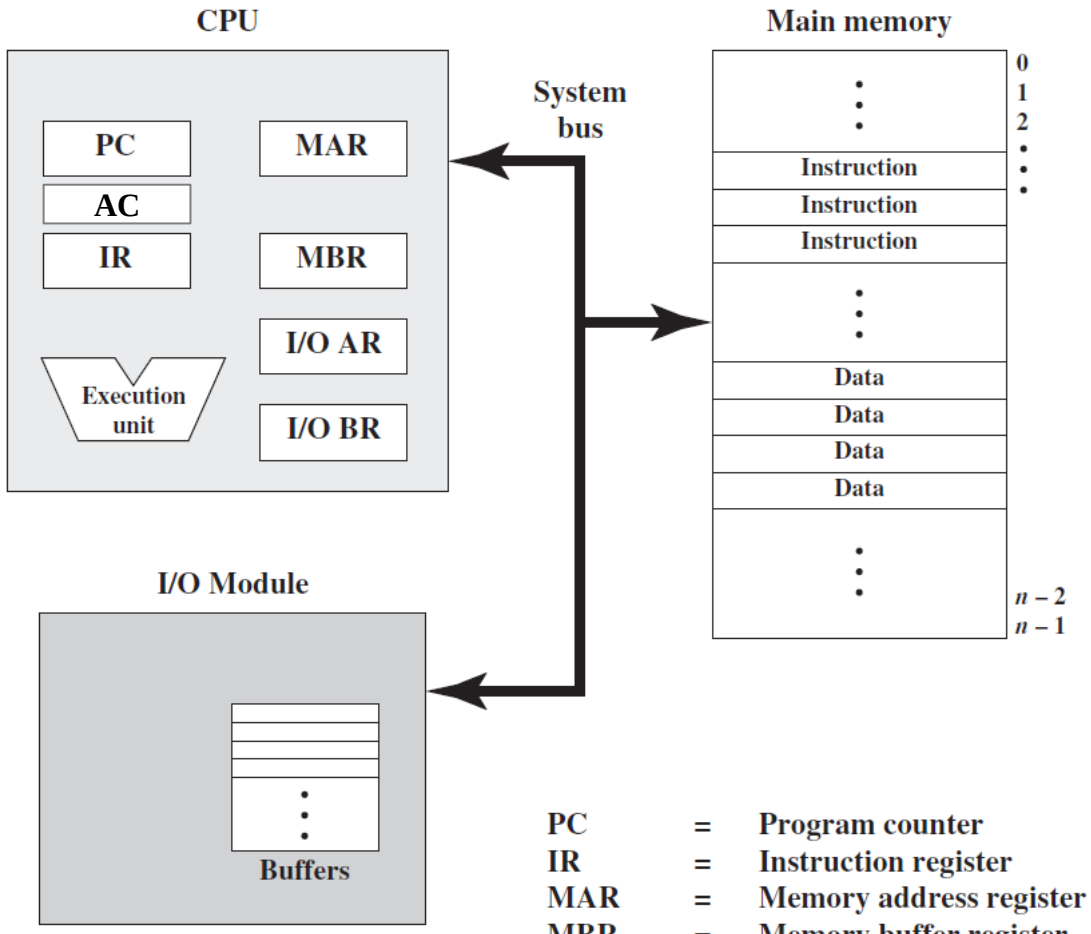
KOMPONENTE NAJVIŠEG NIVOA

- CPU razmenjuje podatke sa memorijom korišćenjem dva interna registra:
  1. Memorijski adresni registar (MAR) – sadrži sledeću adresu sa koje čitati ili u koju treba upisivati.
  2. Memorijski bafer registar (MBR) – sadrži podatke koji su pročitani iz memorije ili koji treba da budu upisani u memoriju.
- I/O adresni registar određuje sa kog ulaznog uređaja se učitavaju podaci ili instrukcije, odnosno na koji izlazni uređaj se prosleđuju rezultati.
- I/O bafer registar sadrži podaci, instrukcije ili rezultate koji se učitavaju/prosleđuju sa ulaznog uređaja, odnosno na izlazni uređaj.
- Registar instrukcije (IR) sadrži operacioni kod instrukcije koja se trenutno izvršava.
- Akumulator (AC) registar za privremeno čuvanje podataka



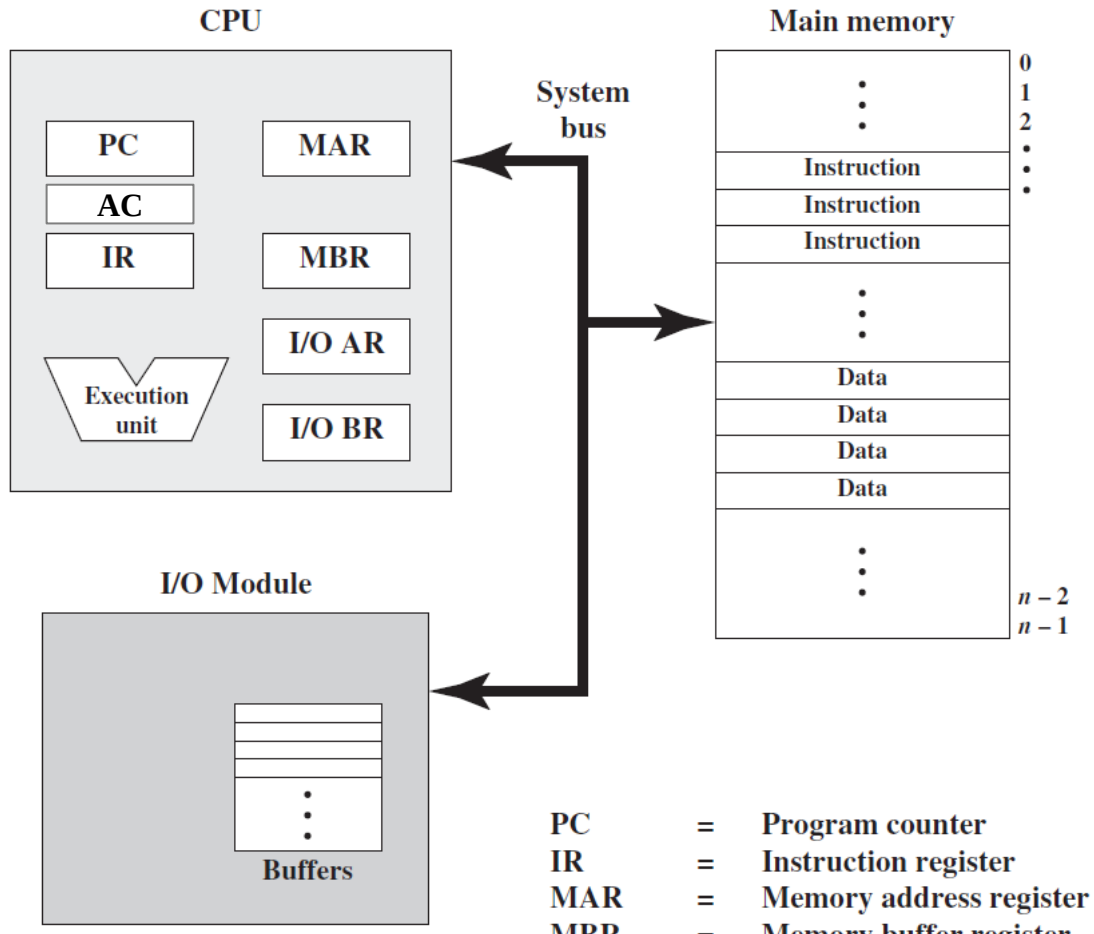
KOMPONENTE NAJVIŠEG NIVOA

- Memorijski modul se sastoji od skupa lokacija, definisanih adresama.
- Adrese su numerisane uzastopnim brojevima.
- Svaka lokacija sadrži binarni broj koji se može interpretirati kao instrukcija ili podatak.



KOMPONENTE NAJVIŠEG NIVOA

- Memorijski modul se sastoji od skupa lokacija, definisanih adresama.
- Adrese su numerisane uzastopnim brojevima.
- Svaka lokacija sadrži binarni broj koji se može interpretirati kao instrukcija ili podaci.
- I/O modul prenosi podatke sa spoljnih uređaja na CPU i memoriju, i obrnuto. Sadrži interne bafere za privremeno čuvanje ovih podataka sve dok se ne mogu poslati.

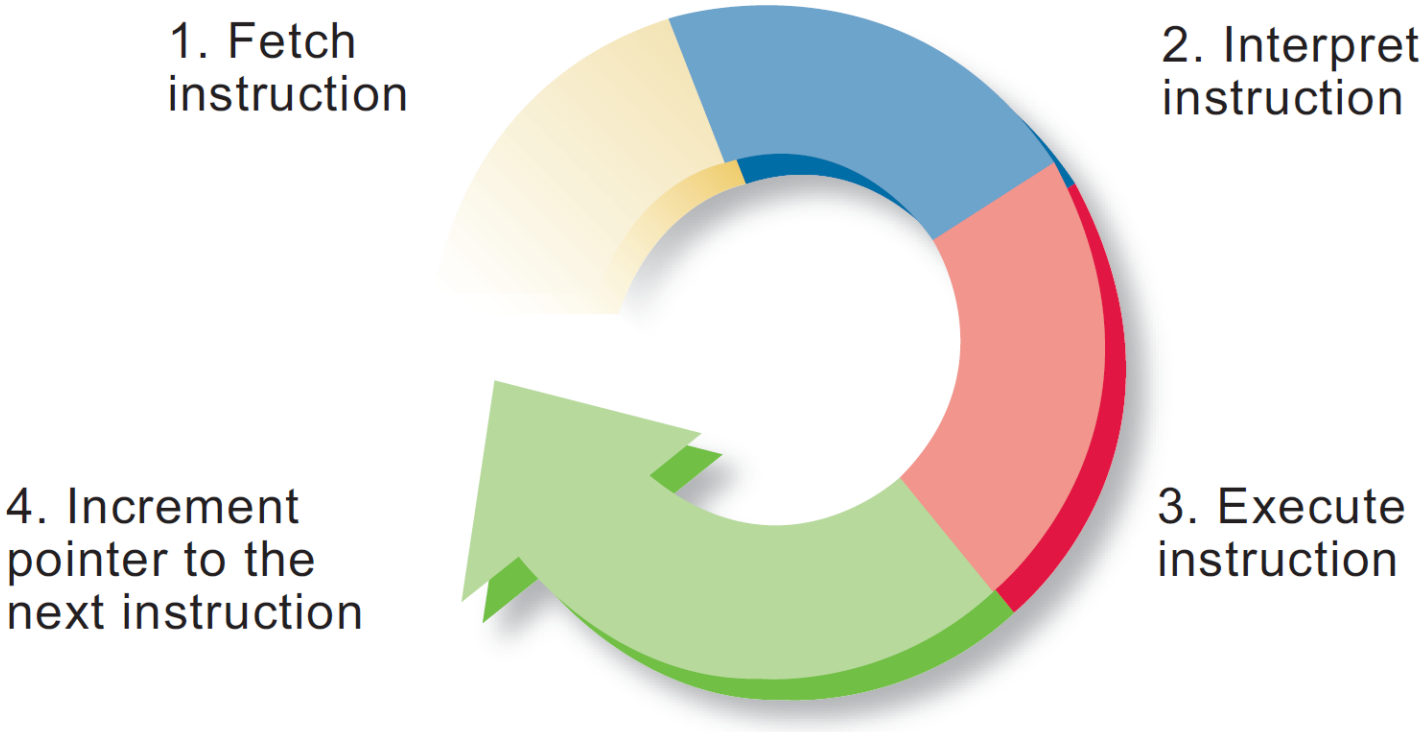


IZVRŠAVANJE PROGRAMA U RAČUNARSKOM SISTEMU

- Osnovna funkcija koju obavlja računar je izvršavanje programa koji se sastoji od skupa instrukcija pohranjenih u memoriji.
- Procesor izvršava instrukcije navedene u programu.
- U najjednostavnijem obliku, izvođenje programa se sastoji u sledećem:
  - procesor „**dohvata**“ jednu po jednu instrukciju iz memorije i
  - **izvršava** instrukciju.
- Izvršavanje instrukcije može uključiti nekoliko operacija i zavisi od prirode instrukcije.

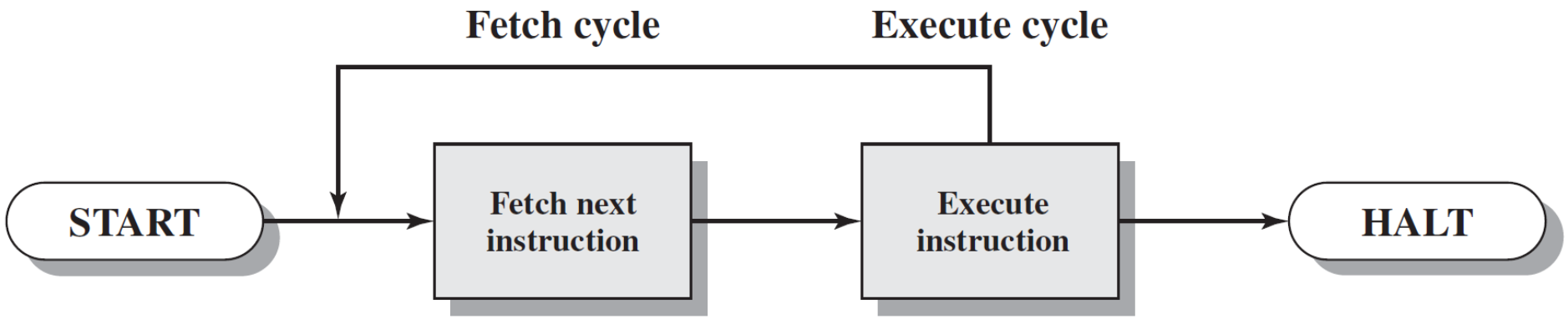
CIKLUS INSTRUKCIJE

Termin ciklus instrukcije odnosi se na proces u kojem računar izvršava samo jednu instrukciju.



CIKLUS INSTRUKCIJE

1. Procesor preuzima instrukciju iz memorije. Registar koji se zove programski brojač (PC) sadrži adresu instrukcije koja je sledeća na redu za izvršenje. Osim ako nije drugačije rečeno, procesor uvećava PC nakon svake instrukcije, tako da će po izvršenju jedne instrukcije, dohvatiti instrukciju koja se nalazi na sledećoj višoj memorijskoj adresi.
2. Preuzeta instrukcija se učitava u procesorski registar instrukcija (IR).
3. Instrukcija sadrži bitove koji određuju akciju koju procesor treba da preduzme. Procesor tumači instrukciju i obavlja potrebnu akciju.



MAŠINSKE INSTRUKCIJE

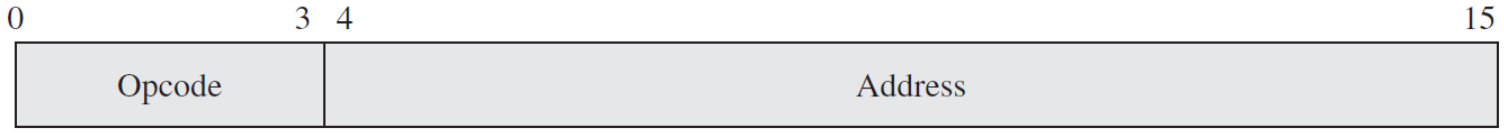
- Mašinska instrukcija – instrukcija koju izvršava procesor.
- Skup mašinskih instrukcija procesora – skup različitih mašinskih instrukcija koje jedan procesor može da izvrši.
- Struktura instrukcija je standardizovana kako bi procesor mogao da prepozna i izvrši datu instrukciju.

ELEMENTI MAŠINSKE INSTRUKCIJE

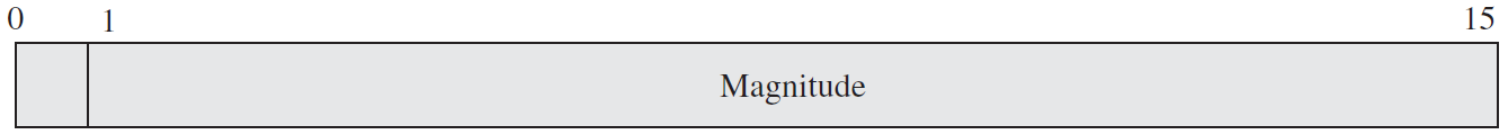
- Operacioni kod instrukcije (eng. *opcode)* – definiše operaciju koja će biti izvršena. Operacija je određena binarnim kodom – kod operacije (*opcode*)
- Referenca na ulazne operande instrukcije – može ih biti i više.
- Referenca na rezultat izvršenja instrukcije.
- Referenca na narednu instrukciju koja treba biti preneta u procesor po izvršenju tekuće instrukcije. U većini slučajeva, sledeća instrukcija koju treba preuzeti odmah prati trenutnu instrukciju. U tim slučajevima nema eksplicitne reference za sledeću naredbu. Kada je potrebna eksplicitna referenca, onda se mora dostaviti njena adresa u memoriji.

Primer instrukcije dužine 16 bitova

- Kod operacije zauzima 4 bita što daje mogućnost kodiranja $2^4 = 16$ različitih operacija.
- Primeri kodova:
  - 0001 – učitaj iz memorije u AC
  - 0010 – sačuvaj sadržaj AC u memoriji
  - 0101 – saberi sadržaj AC sa podatkom iz memorije
- Adresni deo zauzima 12 bitova što daje mogućnost adresiranja $2^{12} = 4096$ različitih reči u memoriji kojima se može direktno pristupiti.



(a) Instruction format

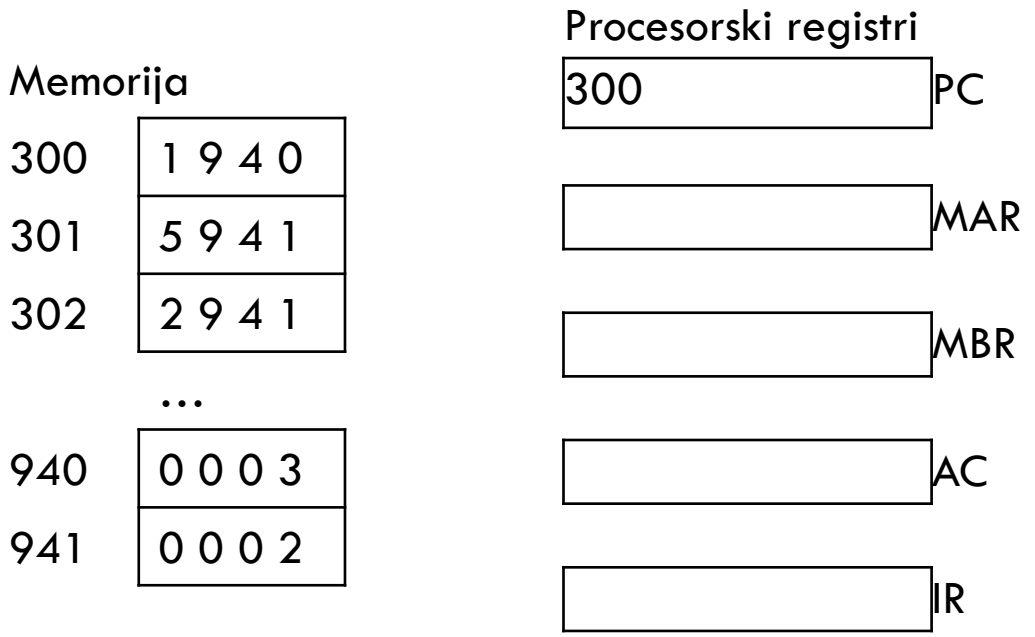


(b) Integer format

PRIMER - NASTAVAK

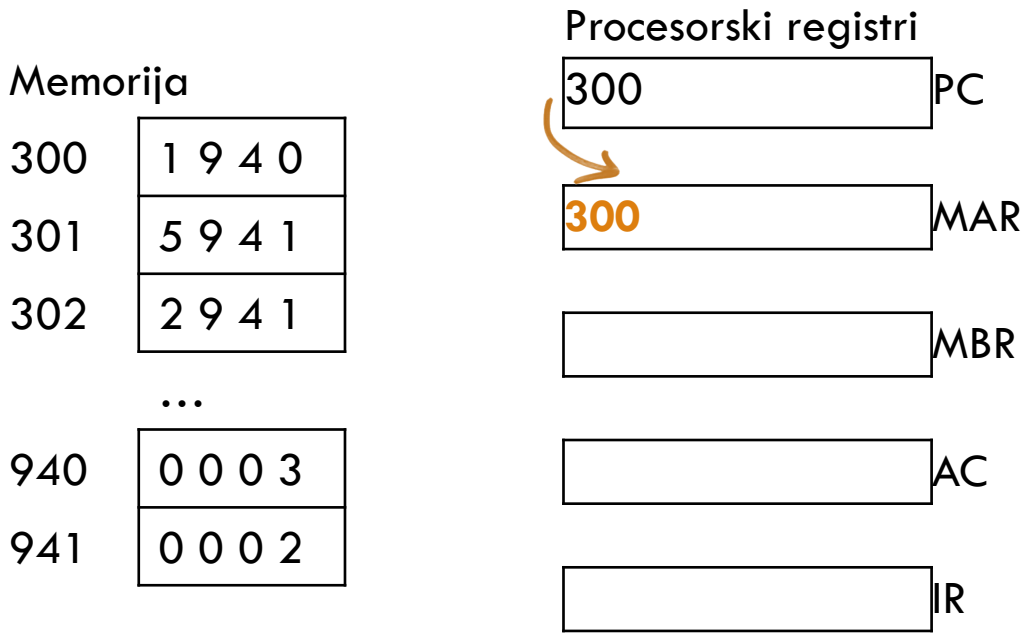
Program koji sabira sadržaj memorijske reči koja se nalazi na adresi 940, i sadržaj memorijske reči na adresi 941, i rezultat smešta na adresu 941

- Brojač instrukcija PC = 300 (adresa prve instrukcije).



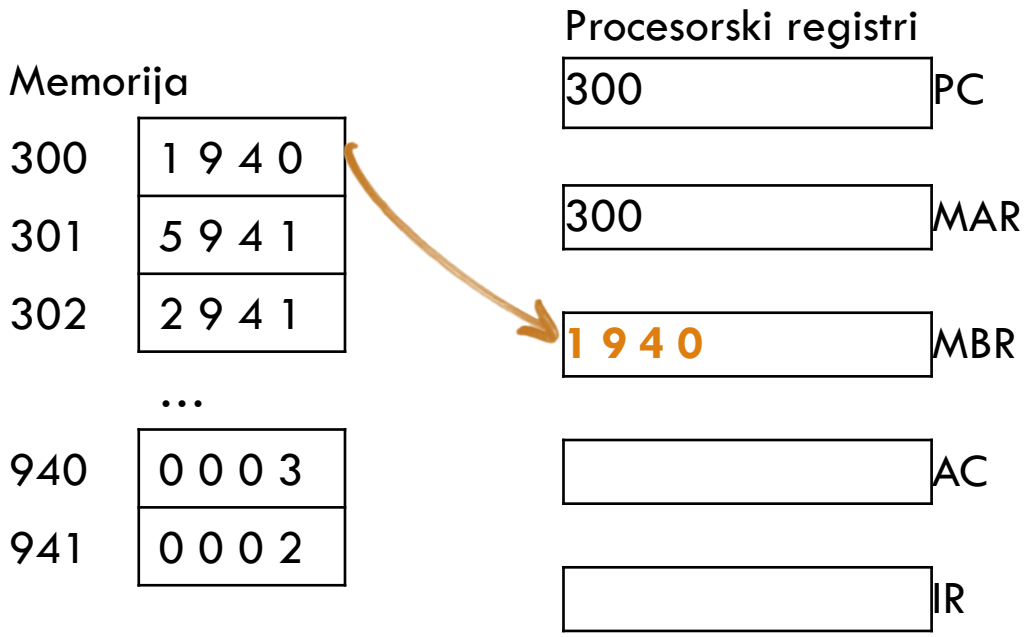
PRIMER - NASTAVAK

- Brojač instrukcija PC = 300 (adresa prve instrukcije).
- Adresa instrukcije se iz brojača instrukcija prenosi u memorijski adresni registar (MAR).



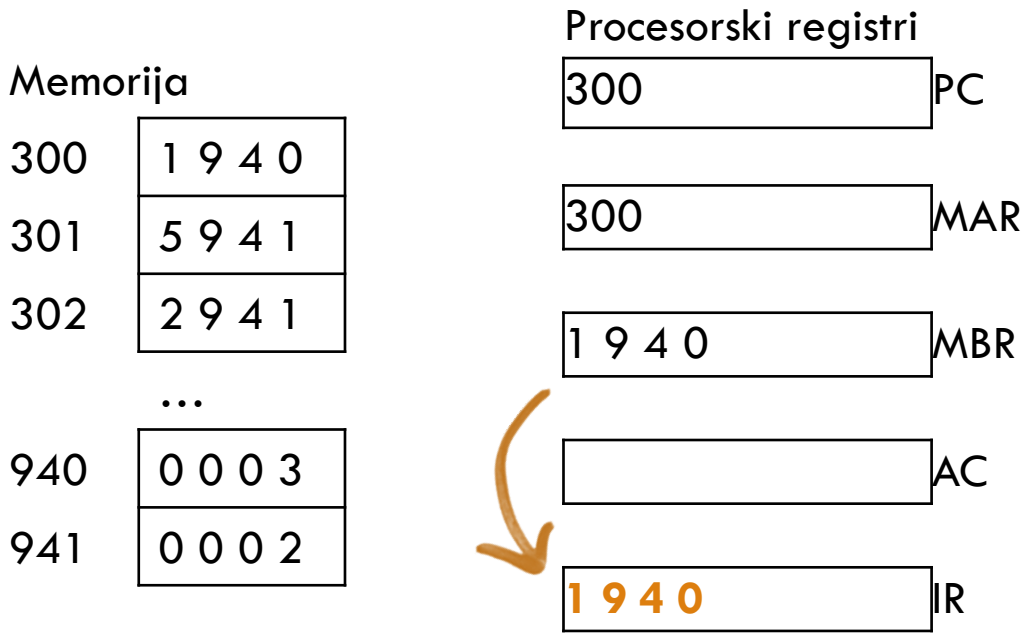
PRIMER - NASTAVAK

- Vrši se prepisivanje instrukcije #1940 sa adrese 300 u memorijski bafer registar (MBR).



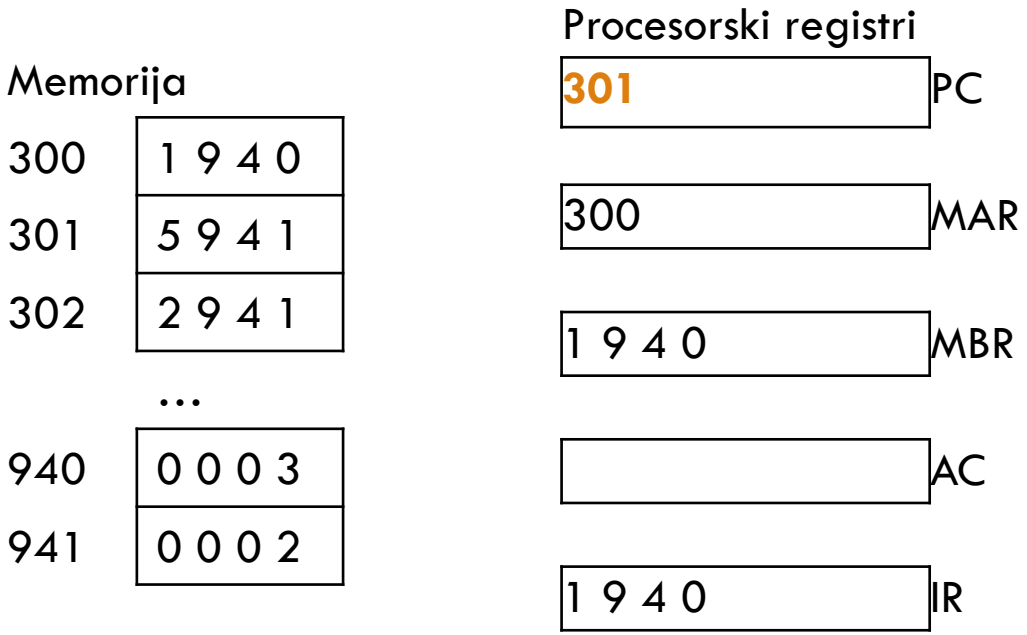
PRIMER - NASTAVAK

- Iz MBR instrukcija se prepisuje u registar instrukcija IR.



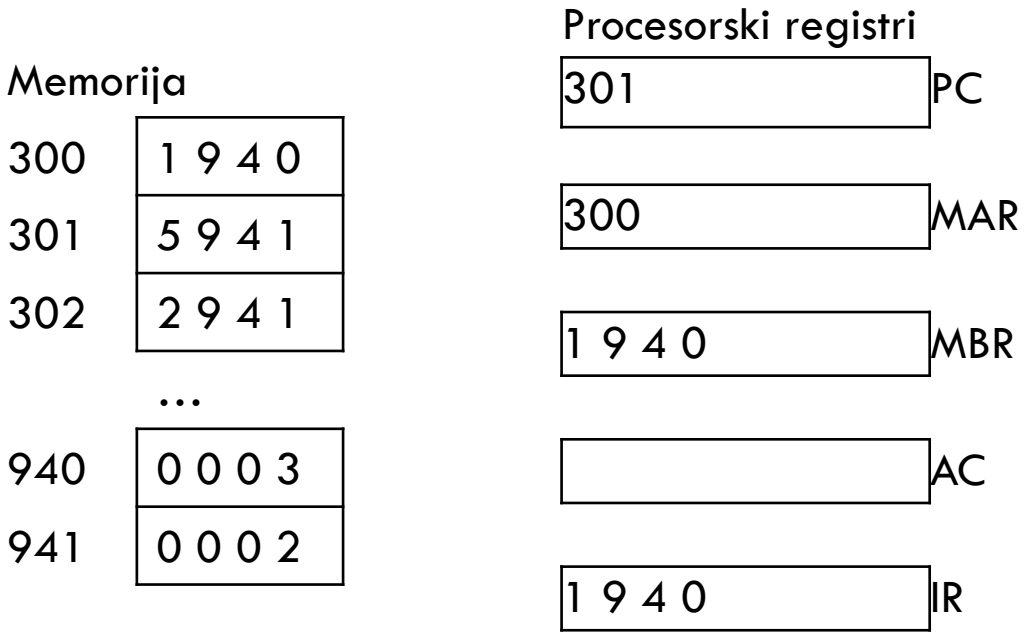
PRIMER - NASTAVAK

- PC se uvećava za 1.



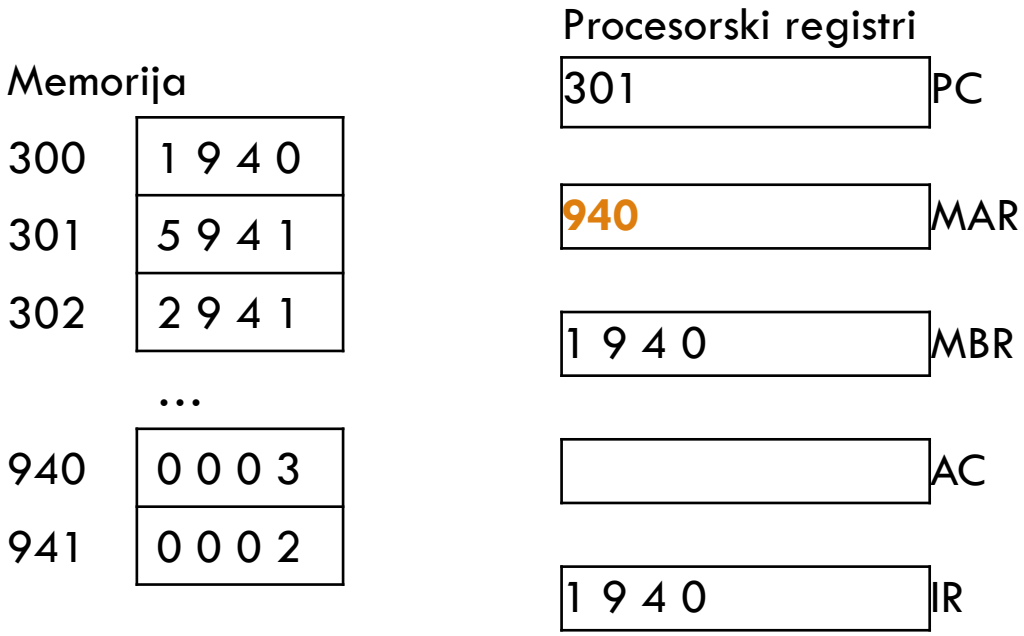
PRIMER - NASTAVAK

- Tumačenje instrukcije #1940: svaka heksadekadna cifra 4 bita.
- Prva četiri bita #1 su operacioni kod, koji odgovara operaciji učitaj u akumulator iz memorije.
- Adresa sa koje učitavamo sadržana je u preostala 3 bita: #940.



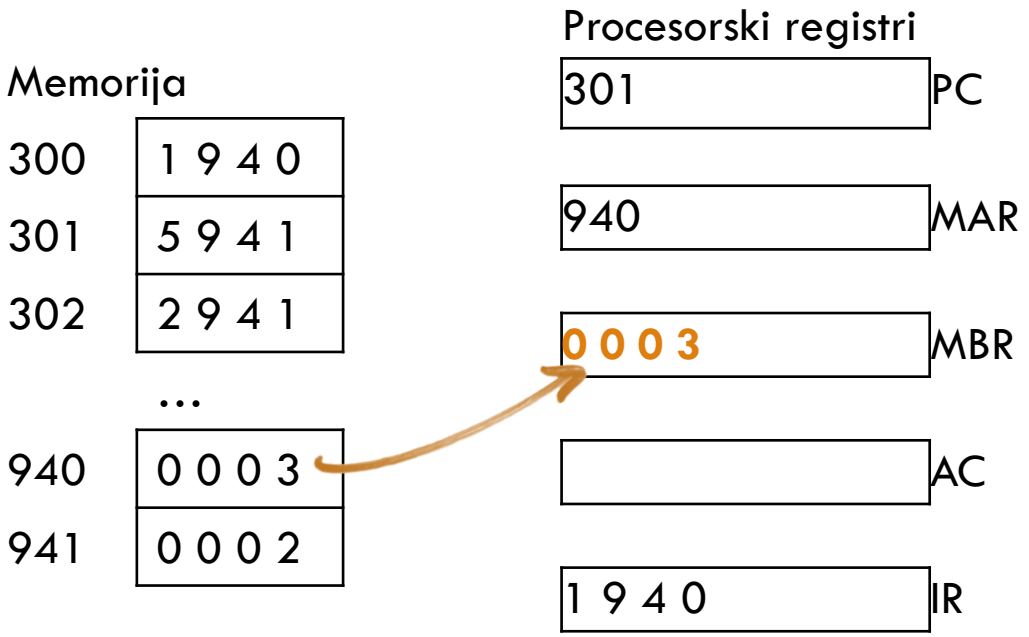
PRIMER - NASTAVAK

- Izvršavanje instrukcije #1940:
- Adresa iz instrukcije se prenosi u memorijski adresni registar (MAR).



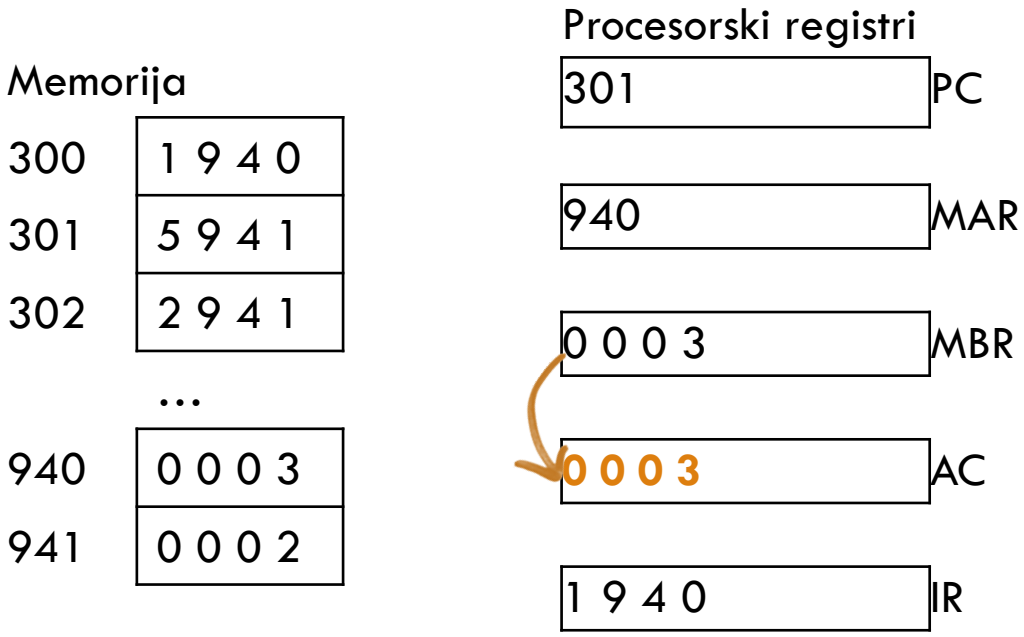
PRIMER - NASTAVAK

- Izvršavanje instrukcije #1940:
- Adresa iz instrukcije se prenosi u memorijski adresni registar (MAR).
- Čita se sadržaj adrese 940 u memoriji i prepisuje u MBR



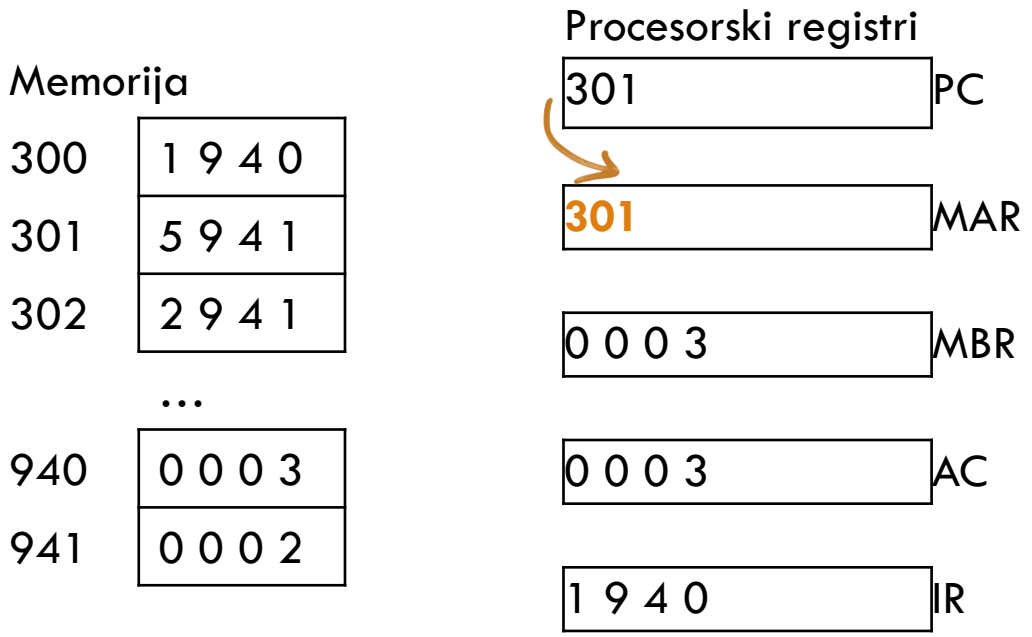
PRIMER - NASTAVAK

- Izvršavanje instrukcije #1940:
- Adresa iz instrukcije se prenosi u memorijski adresni registar (MAR).
- Čita se sadržaj adrese 940 u memoriji i prepisuje u MBR
- Prvi sabirak se dalje iz MBR smešta u AC



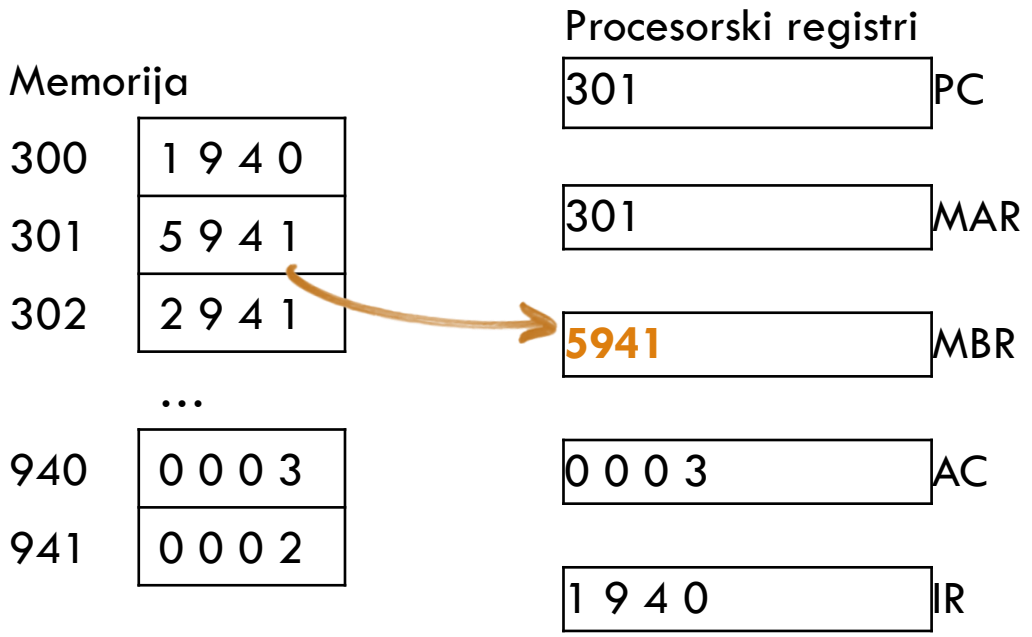
PRIMER - NASTAVAK

- Brojač instrukcija PC = 301 (adresa prve instrukcije).
- Adresa instrukcije se iz brojača instrukcija prenosi u memorijski adresni registar (MAR).



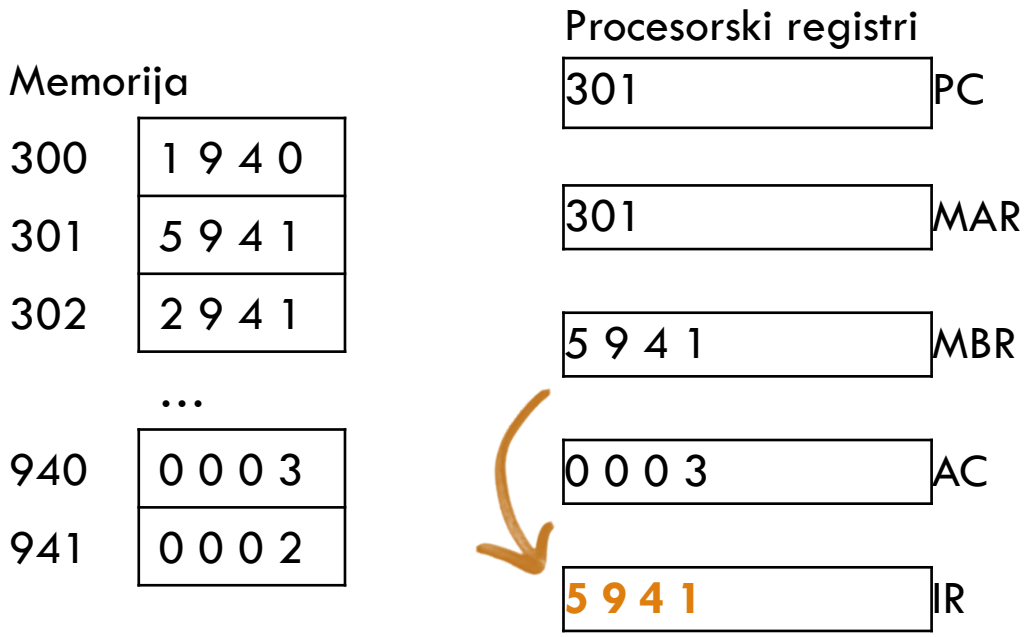
PRIMER - NASTAVAK

- Vrši se prepisivanje instrukcije #5941 sa adrese 301 u memorijski bafer registar (MBR).



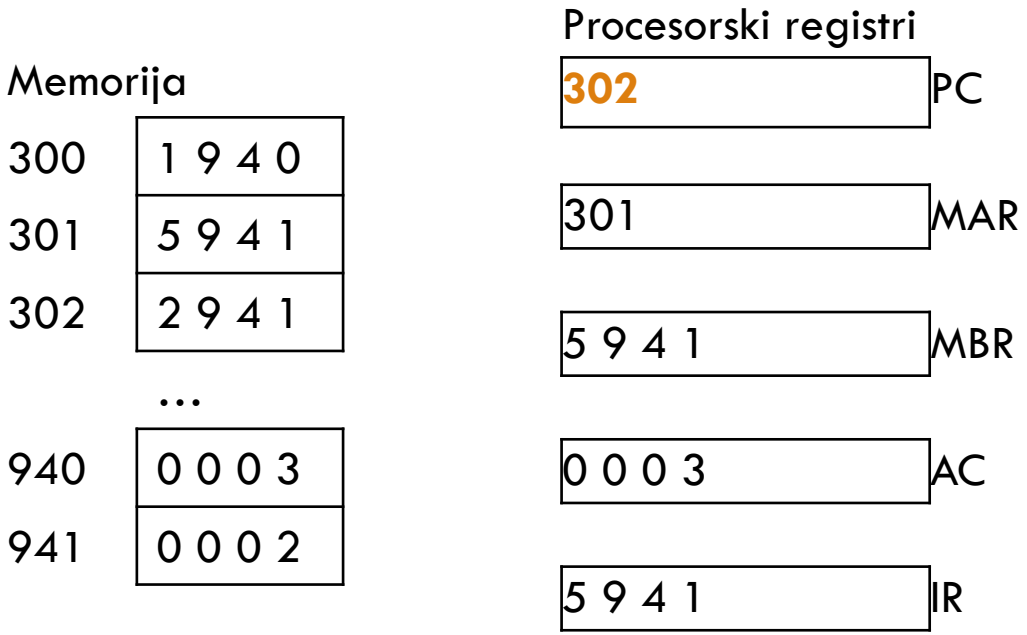
PRIMER - NASTAVAK

- Iz MBR instrukcija se prepisuje u registar instrukcija IR.



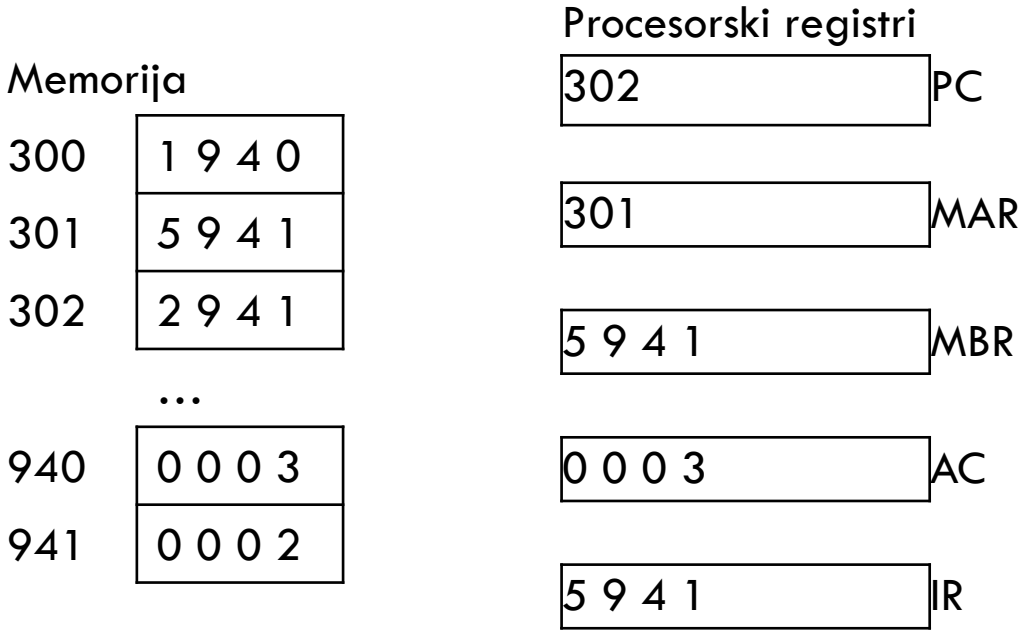
PRIMER - NASTAVAK

- PC se uvećava za 1.



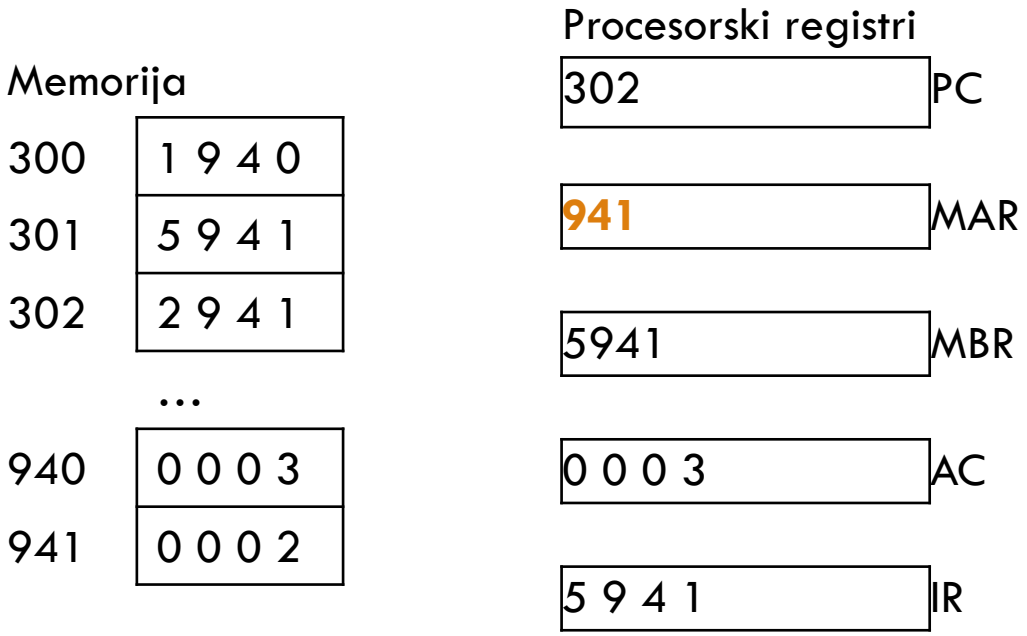
PRIMER - NASTAVAK

- Tumačenje instrukcije #5941: svaka heksadekadna cifra 4 bita.
- Prva četiri bita #5 su operacioni kod, koji odgovara operaciji 0101 – saberi sadržaj AC sa podatkom iz memorije
- Adresa sa koje učitavamo podatak sadržana je u preostala 3 bita: #941.



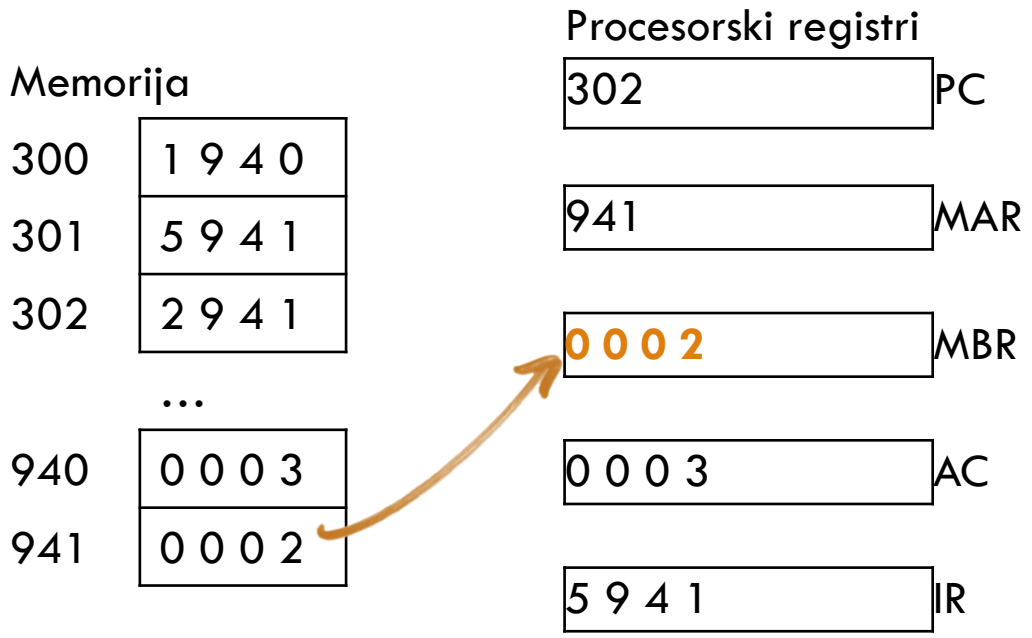
PRIMER - NASTAVAK

- Adresa #941 iz instrukcije se prenosi u memorijski adresni registar (MAR).



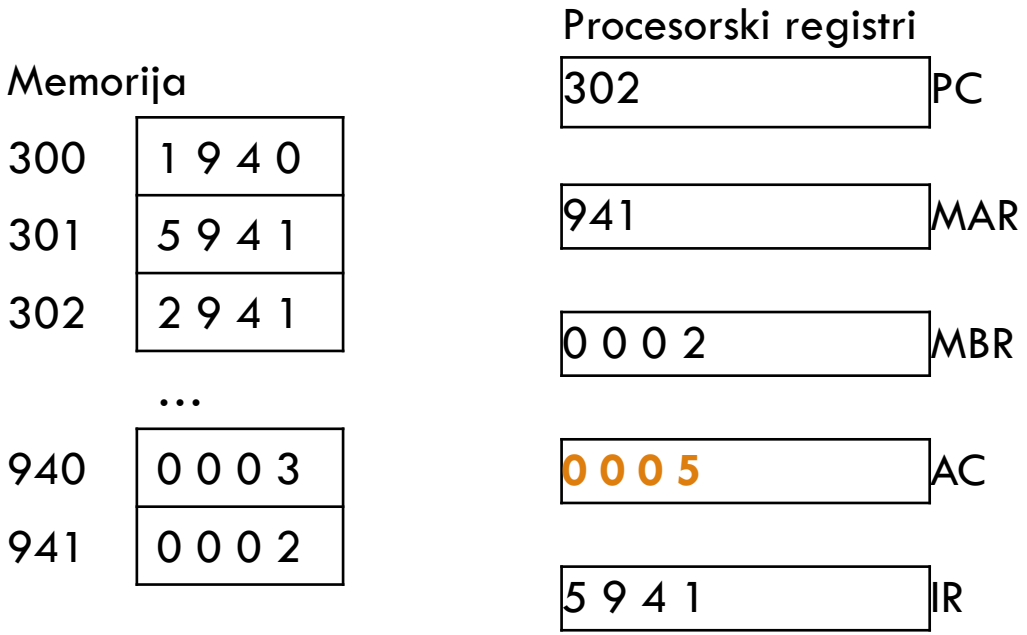
PRIMER - NASTAVAK

- Čita se sadržaj adrese 941 u memoriji i prepisuje u MBR



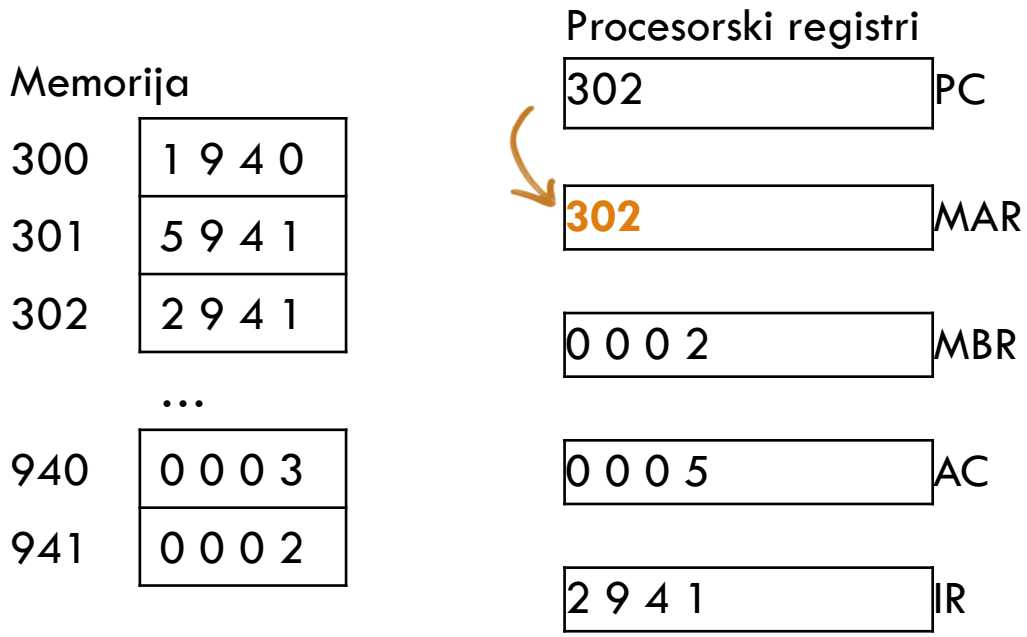
PRIMER - NASTAVAK

- Sadržaj akumulatora i podatka sa adrese 941 se sabiraju i rezultat se smešta u AC.



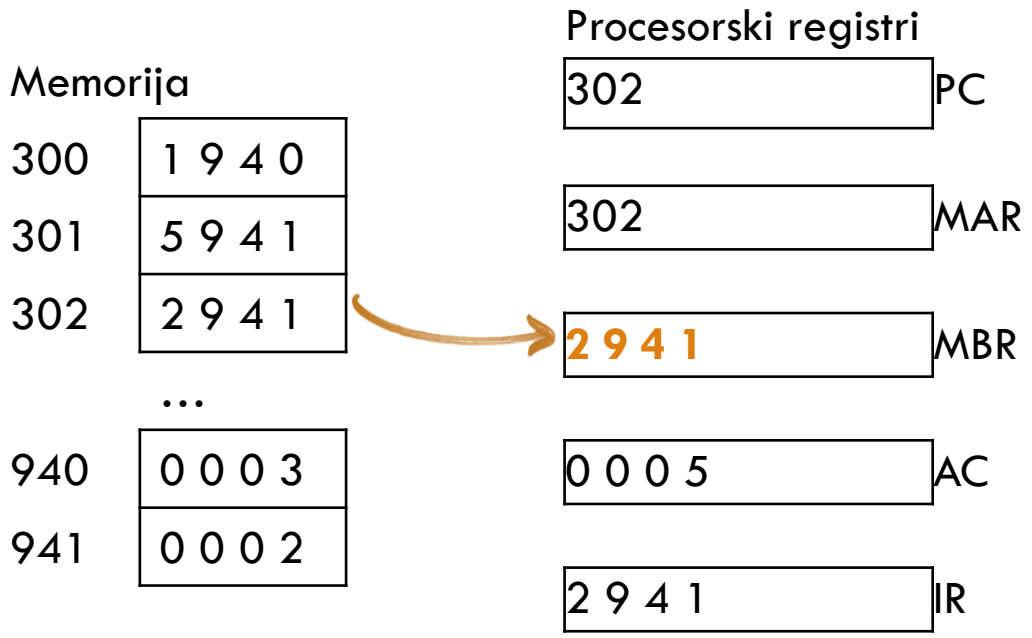
PRIMER - NASTAVAK

- Adresa instrukcije se iz brojača instrukcija prenosi u memorijski adresni registar (MAR).



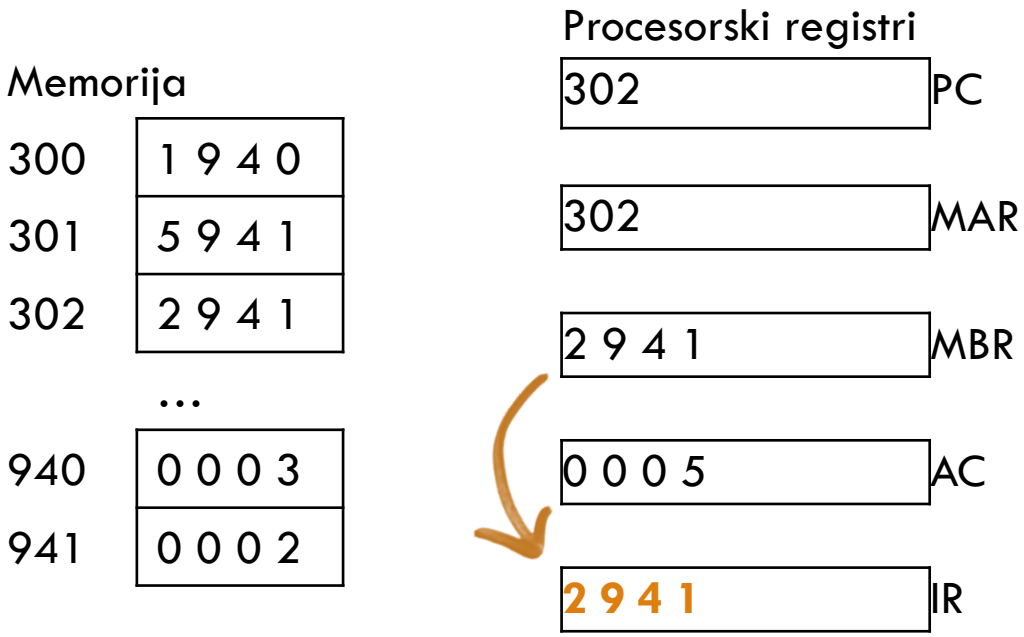
PRIMER - NASTAVAK

- Vrši se prepisivanje instrukcije #2941 sa adrese 302 u memorijski bafer registar (MBR).



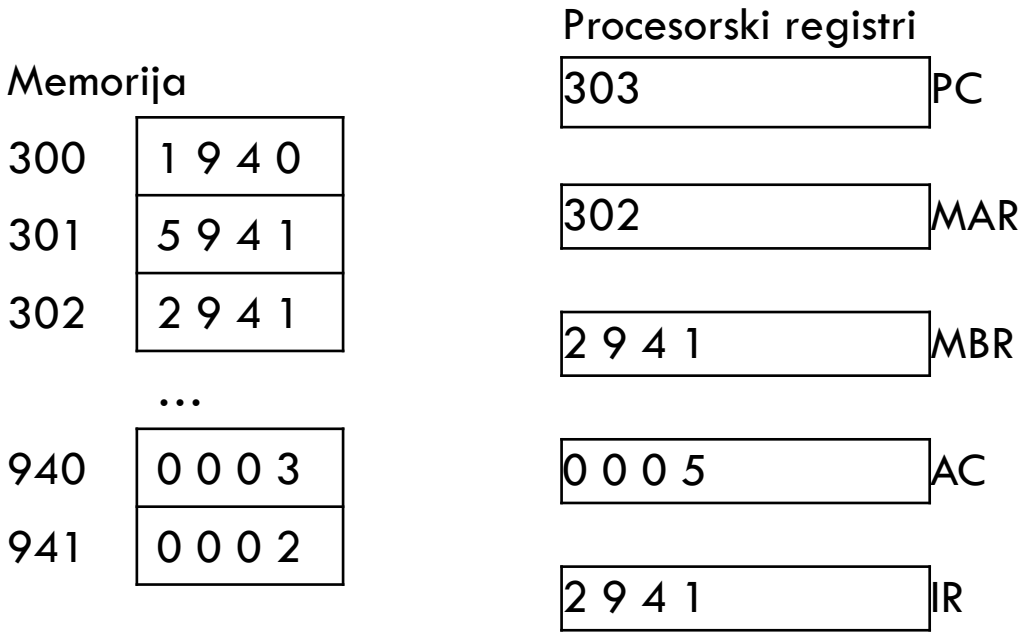
PRIMER - NASTAVAK

- Iz MBR instrukcija se prepisuje u registar instrukcija IR.



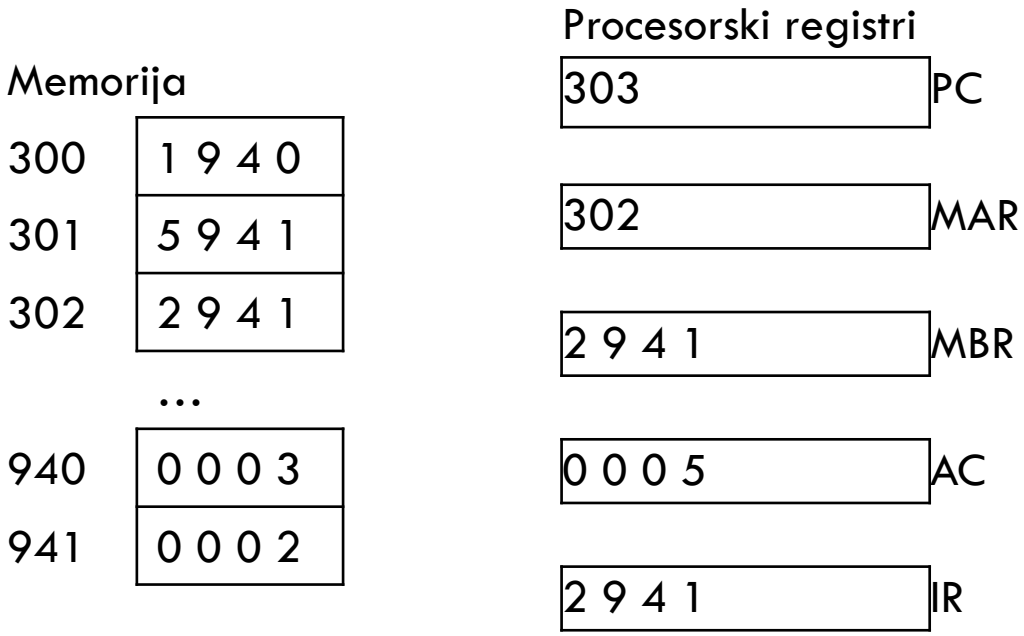
PRIMER - NASTAVAK

- PC se uvećava za 1.



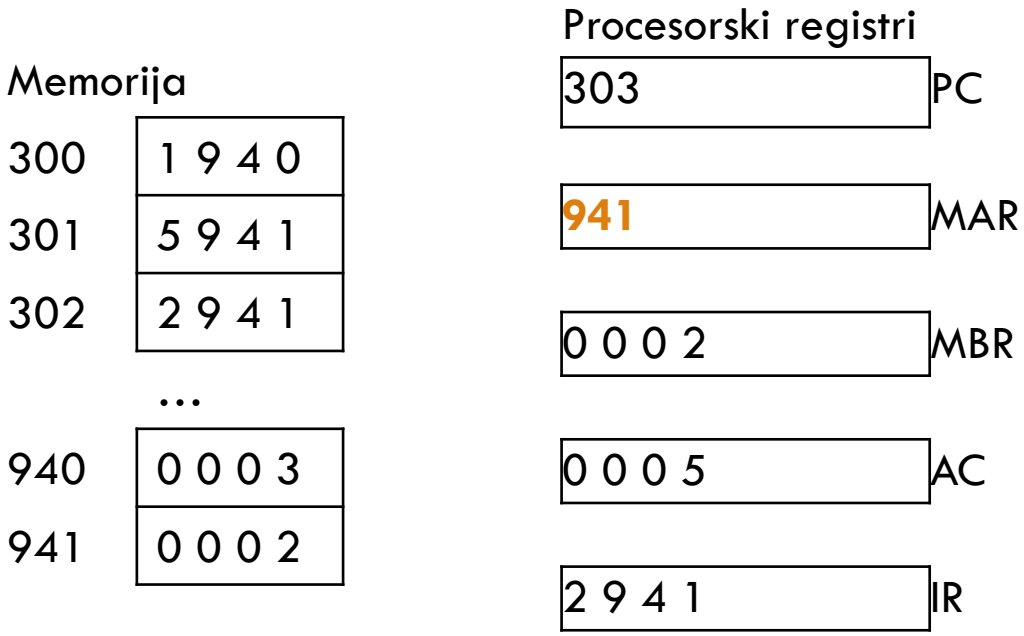
PRIMER - NASTAVAK

- Tumačenje instrukcije #2941: svaka heksadekadna cifra 4 bita.
- Prva četiri bita #2 su operacioni kod, koji odgovara operaciji 0010 – sačuvaj sadržaj AC u memoriji na adresi 941



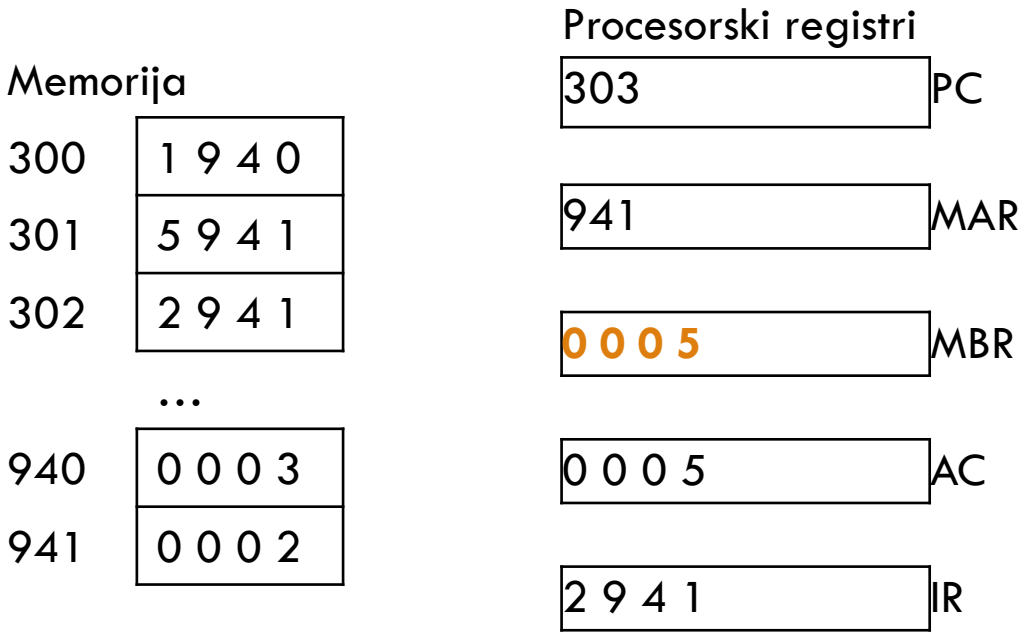
PRIMER - NASTAVAK

- Izvršavanje instrukcije #2941:
- Adresa iz instrukcije se prenosi u memorijski adresni registar (MAR).



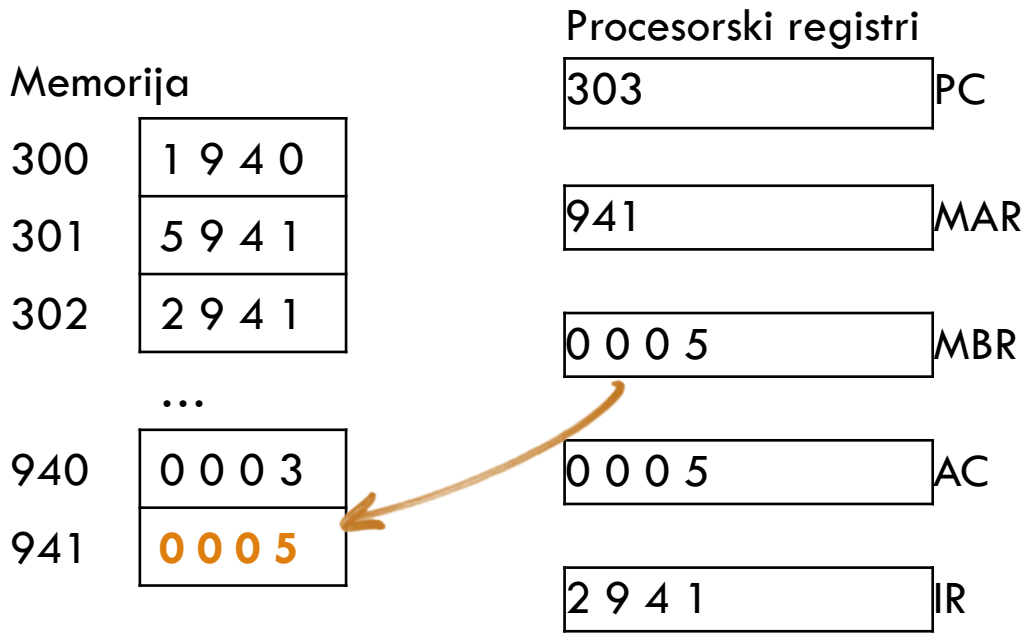
PRIMER - NASTAVAK

- Sadržaj akumulatora se prenosi u MBR.



PRIMER - NASTAVAK

- Sadržaj memorijskog bafer registra (rezultat sabiranja) se upisuje na adresu 941.
- U ovom primeru su bila potrebna tri ciklusa instrukcija da bi sadržaj memorijske lokacije 940 bio dodat na sadržaj memorijske lokacije 941.



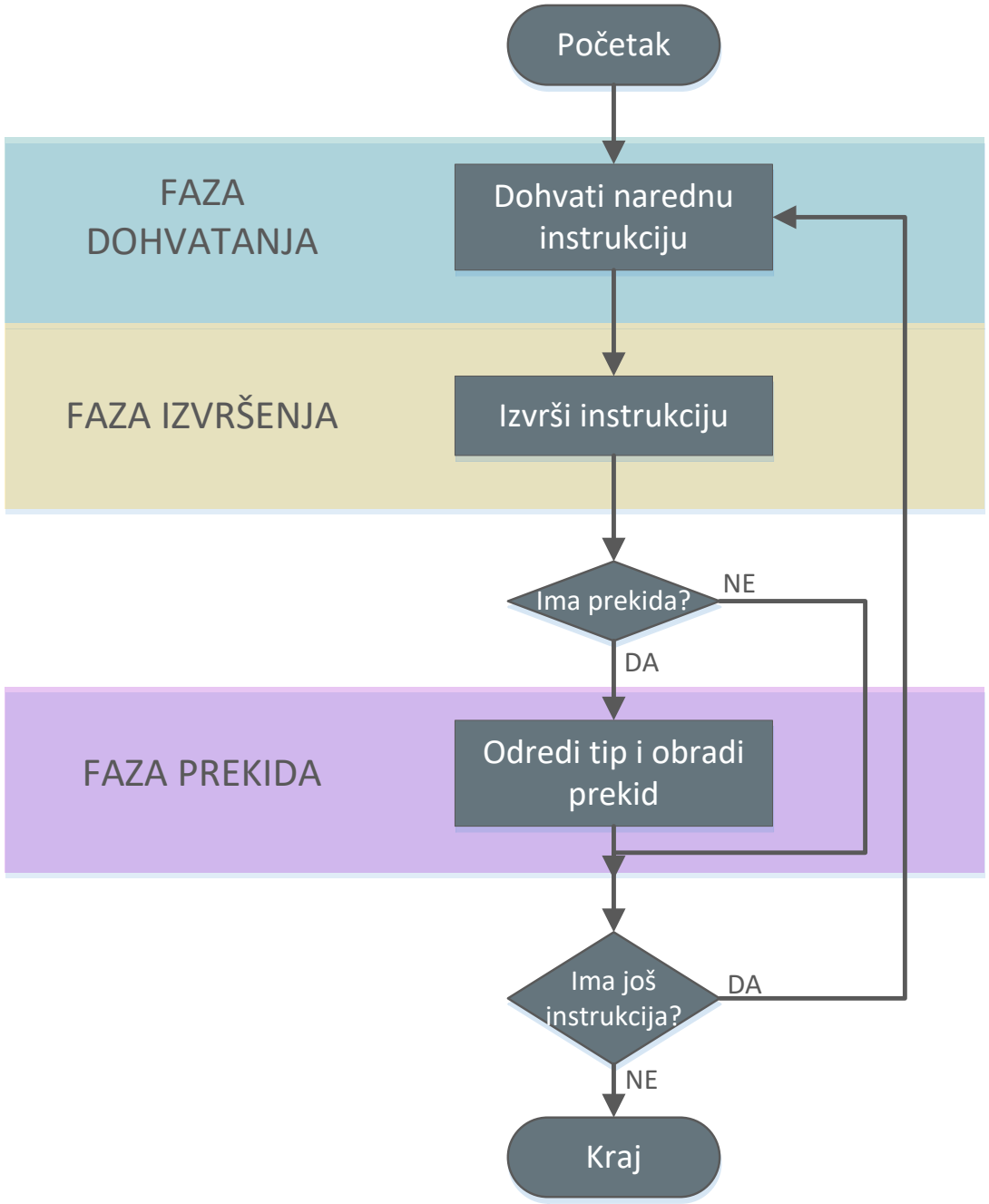
REPREZENTACIJA INSTRUKCIJA

- Instrukcije se u računaru predstavljaju nizom bitova.
- Uvode se mnemoničke oznake (skraćenice) za operacione kodove.

  - ADD saberi
  - SUB oduzmi
  - MUL pomnoži
  - DIV podeli
  - LOAD učitaj iz memorije
  - STOR sačuvaj u memoriju

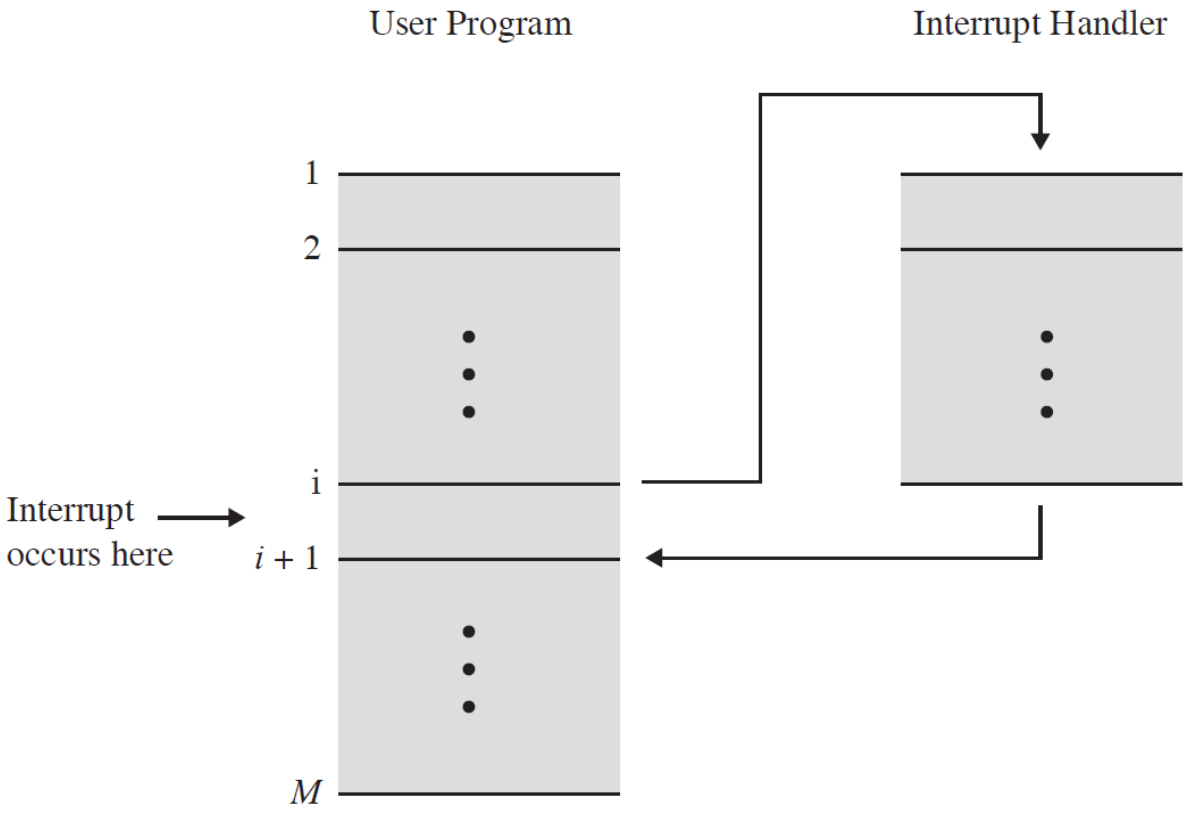
PREKID INSTRUKCIONOG CIKLUSA

- Svi računari pružaju mehanizam pomoću kog drugi moduli (I / O, memorija) mogu prekinuti normalan rad procesora.
- Upotreba prekida omogućava da procesor za vreme dok traje „spora“ ulazno/izlazna operacija na spoljašnjim uređajima bude angažovan na izvršavanju drugih instrukcija.
- Sistem prekida je mehanizam koji omogućava efikasniji rad računara.



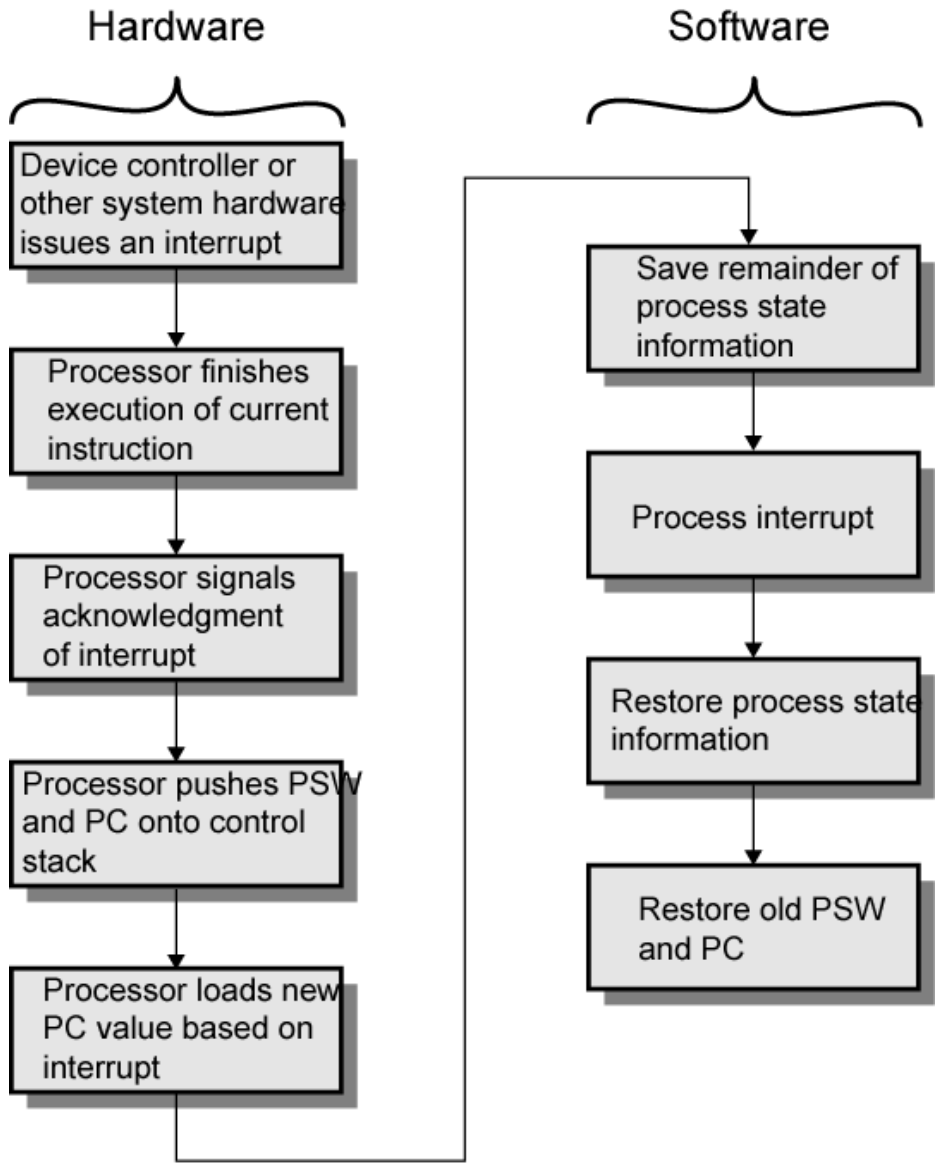
PREKID INSTRUKCIONOG CIKLUSA

- Sa stanovišta korisničkog programa: prekid normalne sekvence izvršenja.
- Kada se obrada prekida završi, izvršenje se nastavlja.
- Korisnički program ne mora sadržati nikakav poseban kod za tretiranje prekida; procesor i operativni sistem su odgovorni za obustavljanje korisničkog programa, a zatim ga ponovo nastavljaju u istoj tački.



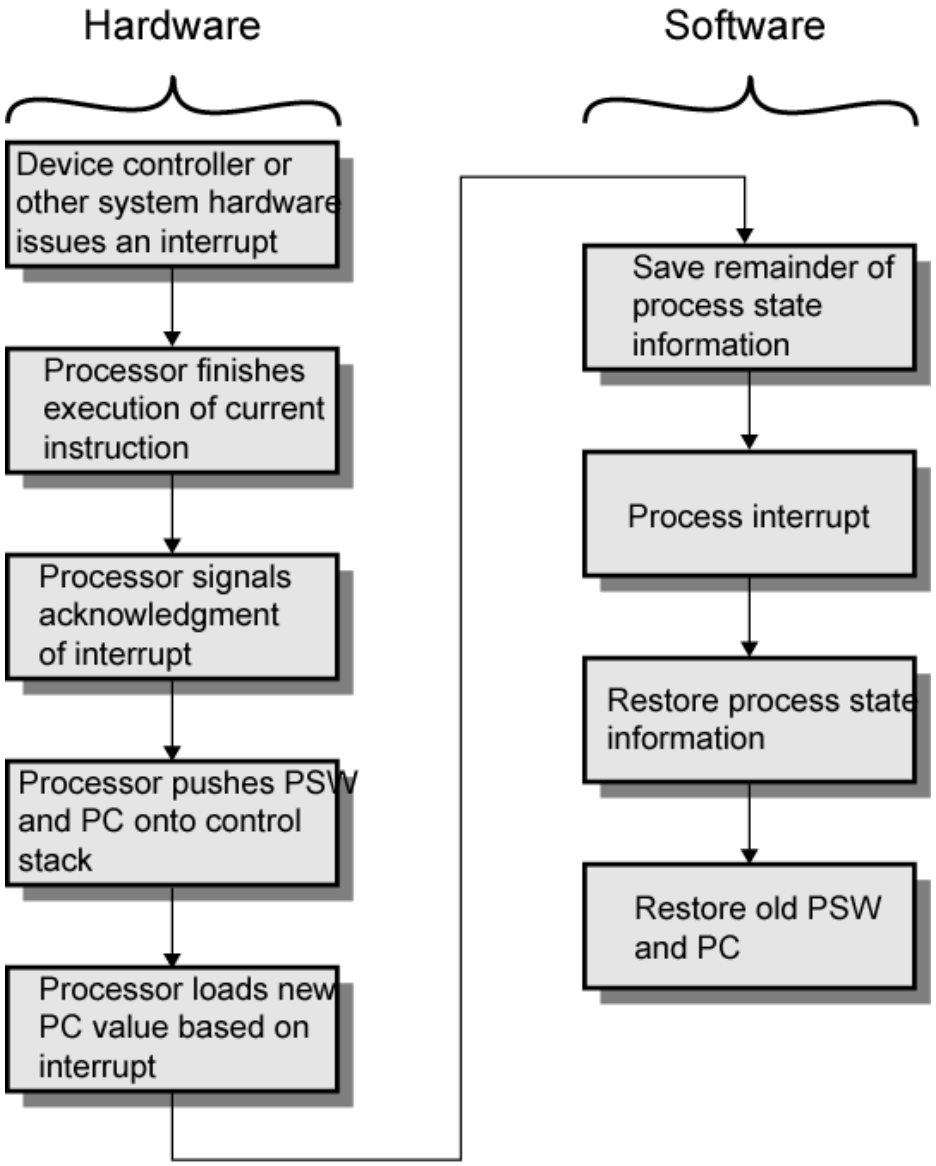
PROCESIRANJE PREKIDA

- Uređaj generiše signal za prekid.
- Procesor završava trenutnu instrukciju.
- Testira da li ima prekida, utvrđuje da ima i šalje uređaju signal da je zahtev za prekid primećen.
- Procesor se priprema da prebaci kontrolu na zadatak zbog koga je prekid zatražen.



PROCESIRANJE PREKIDA

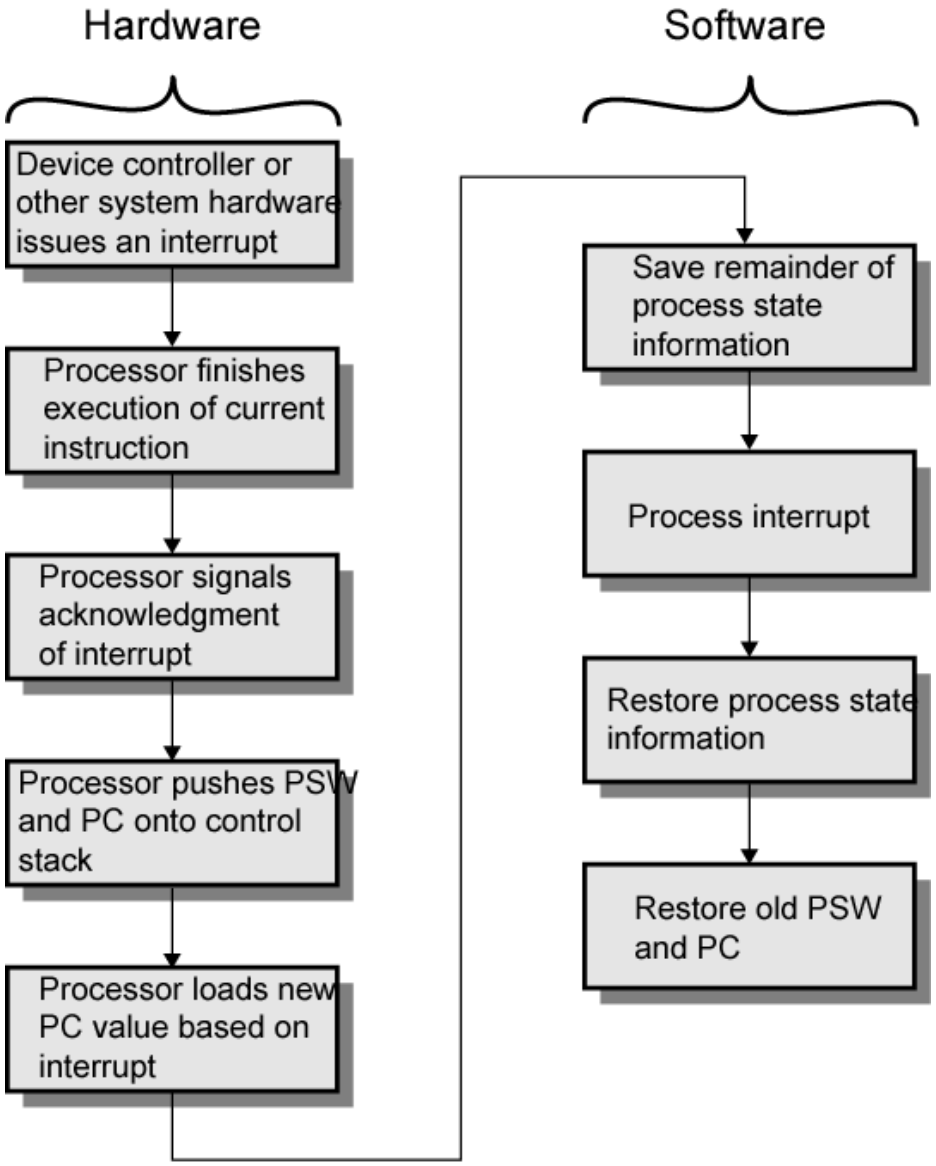
- Minimum informacija koje cpu mora da sačuva o trenutnom programu pre nego što počne da obrađuje prekid:
  - Status procesora - PSW
  - Lokacija sledeće instrukcije (PC) koju treba izvršiti
- U PC se učitava početna lokacija za program kojim će biti obrađen prekid.



PROCESIRANJE PREKIDA

- Kada se završi procesiranje prekida sačuvane vrednosti se vraćaju u registre.

- Informacije sačuvane u PSW i PC se isčitavaju i program nastavlja od tačke u kojoj je zaustavljen.



BROJ ADRESA U INSTRUKCIJI

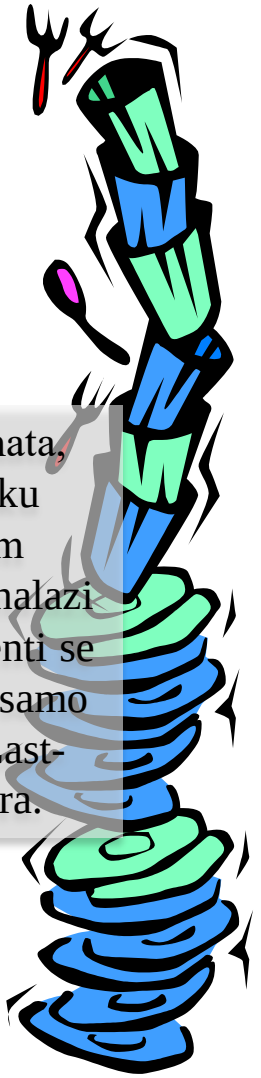
- Nula-adresne instrukcije
- Jedno-adresne instrukcije
- Dvo-adresne instrukcije
- Tro-adresne instrukcije

NULA-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.



NULA-ADRESNE INSTRUKCIJE

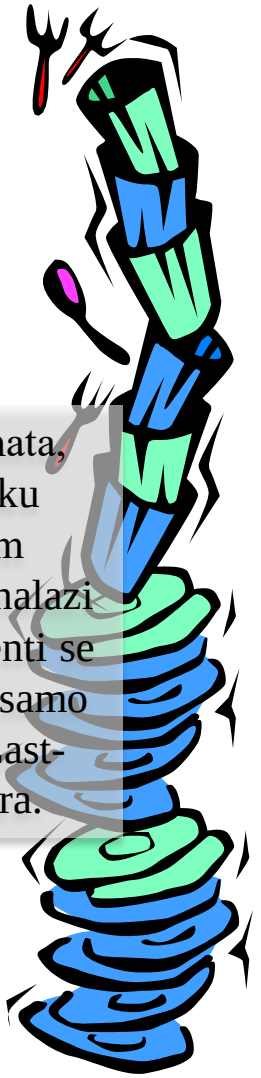
Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

PUSH A

A

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.

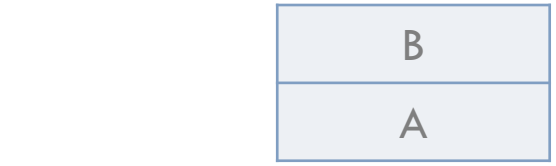


NULA-ADRESNE INSTRUKCIJE

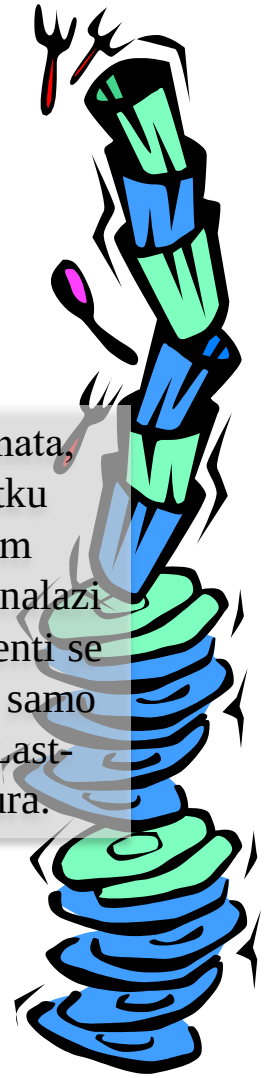
Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

PUSH A
PUSH B



Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.

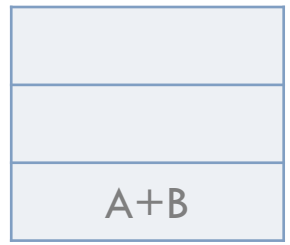


NULA-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

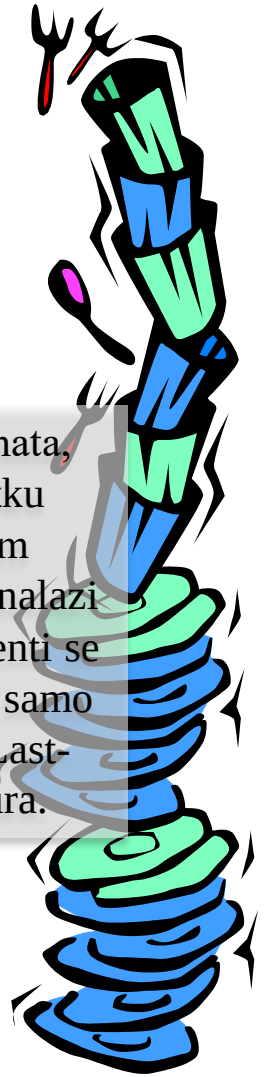
- Nula-adresne instrukcije se mogu primeniti u stek mašinama

PUSH A
PUSH B
ADD



POP B, POP A,
PUSH A+B

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.



NULA-ADRESNE INSTRUKCIJE

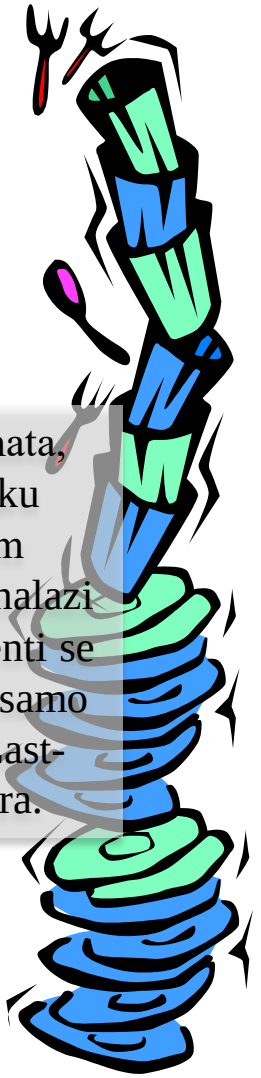
Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

```
PUSH A
PUSH B
ADD
PUSH C
```

C
A+B

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.



NULA-ADRESNE INSTRUKCIJE

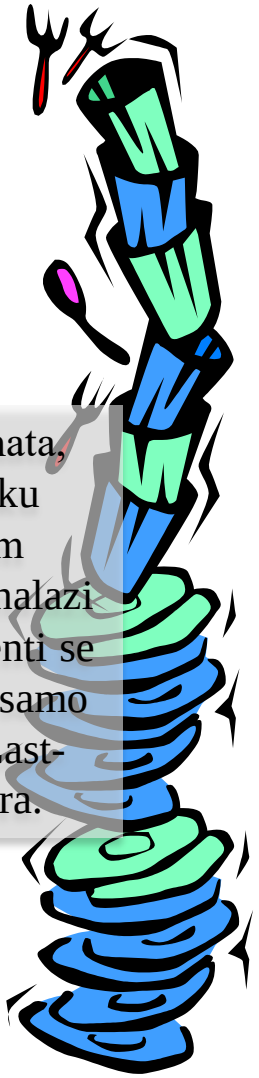
Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

```
PUSH A
PUSH B
ADD
PUSH C
PUSH D
```

D
C
A+B

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.

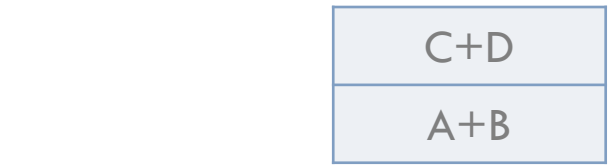


NULA-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

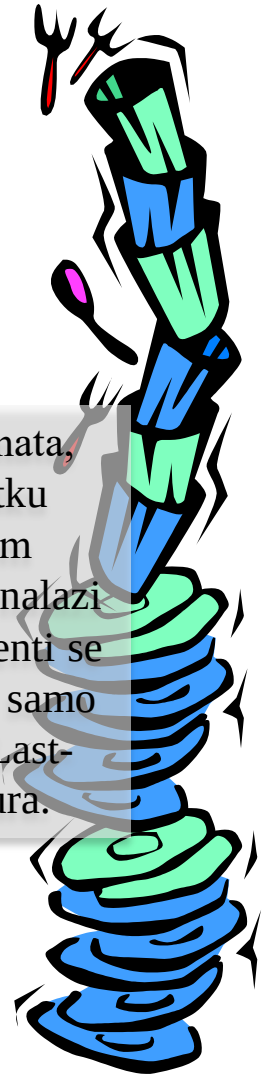
- Nula-adresne instrukcije se mogu primeniti u stek mašinama

```
PUSH A
PUSH B
ADD
PUSH C
PUSH D
ADD
```



POP D, POP C,
PUSH C+D

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.

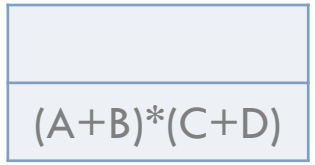


NULA-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

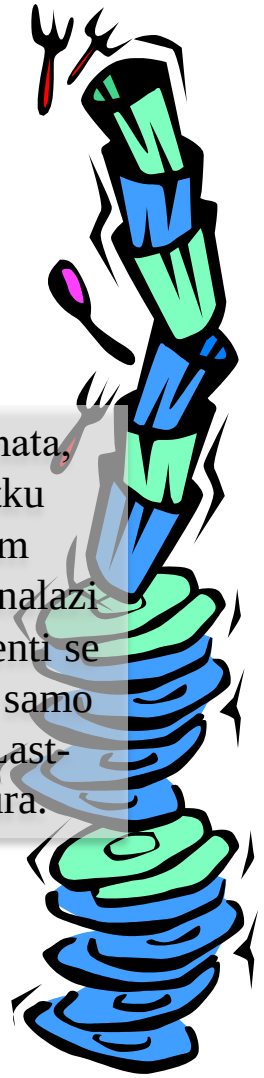
- Nula-adresne instrukcije se mogu primeniti u stek mašinama

```
PUSH A
PUSH B
ADD
PUSH C
PUSH D
ADD
MUL
```



POP C+D, POP A+B,
PUSH (A+B)*(C+D)

Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.

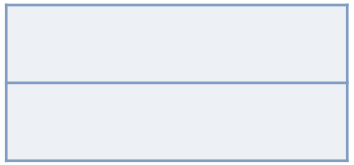


NULA-ADRESNE INSTRUKCIJE

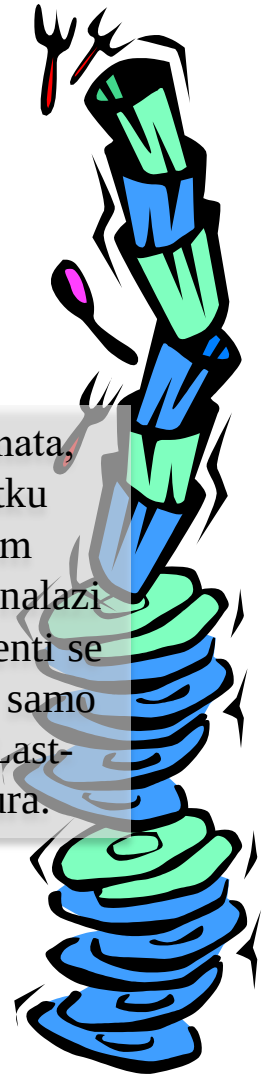
Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Nula-adresne instrukcije se mogu primeniti u stek mašinama

```
PUSH A
PUSH B
ADD
PUSH C
PUSH D
ADD
MUL
POP X
```



Stek je uređeni skup elemenata, takav da se u jednom trenutku može pristupiti samo jednom elementu i to onom koji se nalazi na vrhu steka (TOP). Elementi se mogu dodavati ili uklanjati samo sa vrha steka. Zato je stek Last-In, First Out (LIFO) struktura.



JEDNO-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Jedno-adresne instrukcije podrazumevaju implicitnu upotrebu akumulatora (AC)

```
LOAD A      // AC ← M [A]
ADD B       // AC ← AC + M [B]
STORE T     // M [T] ← AC
LOAD C      // AC ← M [C]
ADD D       // AC ← AC + M [D]
MUL T       // AC ← AC * M [T]
STORE X     // M [X] ← AC
```

Akumulator je procesorski registar opšte namene koji se koristi za privremeno čuvanje rezultata izračunavanja.

DVO-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Dvo-adresne instrukcije podrazumevaju upotrebu dva registra, pri čemu se rezultat operacije upisuje u registar u kome je bio jedan od operanada.

```
MOV R1, A        // R1 ← M [A]
ADD R1, B        // R1 ← R1 + M [B]
MOV R2, C        // R2 ← M [C]
ADD R2, D        // R2 ← R2 + M [D]
MUL R1, R2       // R1 ← R1 * R2
MOV X, R1        // M [X] ← R1
```

MOV instrukcija premešta operande u i iz memorije i procesorskih registara.

TRO-ADRESNE INSTRUKCIJE

Izračunavanje vrednosti izraza: X = (A + B) * (C + D)

- Tro-adresne instrukcije takođe podrazumevaju upotrebu dva registra, ali samo za čuvanje rezultata operacija.

```
ADD R1, A, B        // R1 ← M[A] + M[B]
ADD R2, C, D        // R2 ← M[C] + M[D]
MUL X, R1, R2       // M[X] ← R1 * R2
```

- Postojanje instrukcija sa tri adrese znatno olakšava izvršavanje zahtevane operacije.

- Programi su kompaktni, ali same instrukcije mogu da postanu glomazne zbog zahteva za čuvanjem tri operanda.