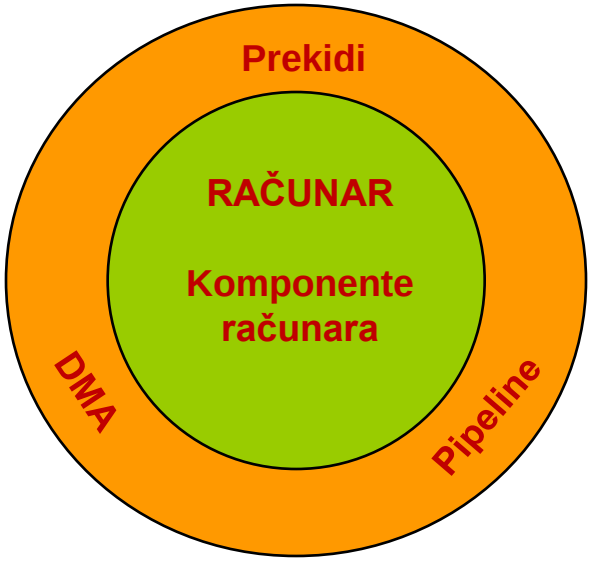
Mehanizmi za efikasniji rad računara



TEME

- ✓ *Pipeline* mehanizam
- ✓ *DMA* mehanizam
- ✓ Mehanizam prekida

Pipeline mehanizam

- Koncept je uveden sa ciljem da se ubrza proces izvršavanja programa.
- Zasniva se na konkurentnosti izvršavanja instrukcija.

Proces izvršavanja instrukcije

IF (*Instruction Fatch*) → dohvatanje instrukcije iz memorije i smeštanje u *IR*

D (*Decode*) → dekodiranje koda operacije instrukcije

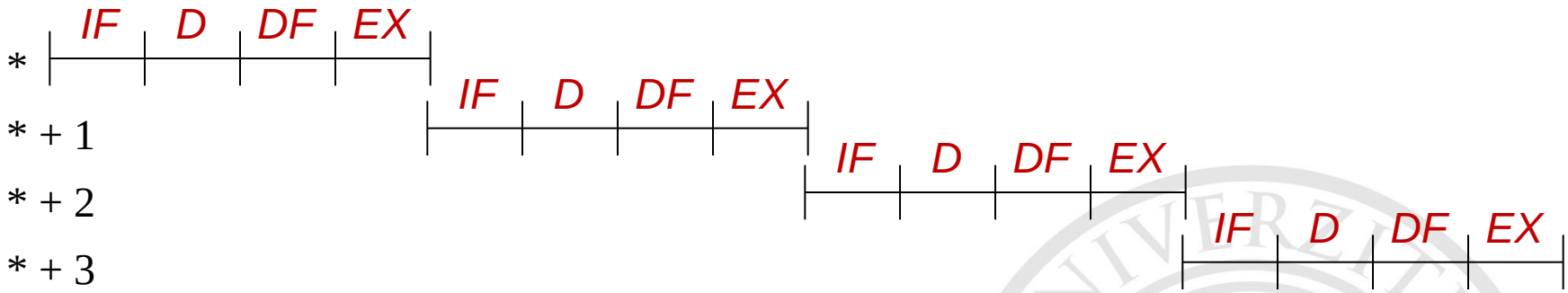
DF (*Data Fatch*) → dohvatanje operanda iz memorije i smeštanje u registar za operande

EX (*Execute*) → izvršavanje instrukcije

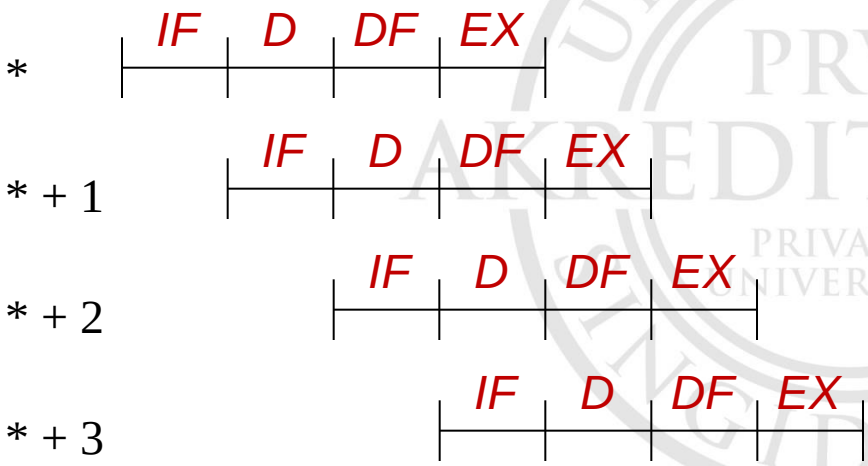
Pipeline mehanizam

Primer 1 Izvršavanje programa od 4 instrukcije smeštenog u memoriji od adrese *.

a) bez *pipeline-a*



b) sa *pipeline-om*



Pipeline mehanizam

- U primeru
  - pod a) program se izvršava u 16 taktova
  - pod b) program se izvršava u 7 taktova
  - ostvareno ubrzanje je 56,25%.
- Ubrzanje zavisi od broja stepeni (*stages*) ili dubine *pipeline*-a, odnosno broja faza u izvršavanju instrukcije.
- Ubrzanje se postiže po cenu složenijeg i skupljeg procesora, težeg za projektovanje.
- Svi današnji procesori podržavanju *pipeline* koncept.

DMA mehanizam

- Koncept *DMA* (*Direct Memory Access*) omogućava direktan prenos podataka između periferije i memorije bez učešća procesora.

U sistemu bez DMA:

- procesor mora da kopira deo po deo podataka i da ga prosleđuje od izvora do odredišta
- za to vreme, procesor nije raspoloživ ni za kakvu drugu aktivnost
- dodatno vreme se gubi zato što su periferije mnogo sporije od *OM.*

U sistemu sa DMA:

- procesor je oslobođen poslova oko prenosa podataka
- za vreme prenosa, procesor može da obavlja druge aktivnosti, ali ne može da koristi magistralu
- prenosom podataka upravlja posebna komponenta – *DMA* kontroler.

DMA mehanizam

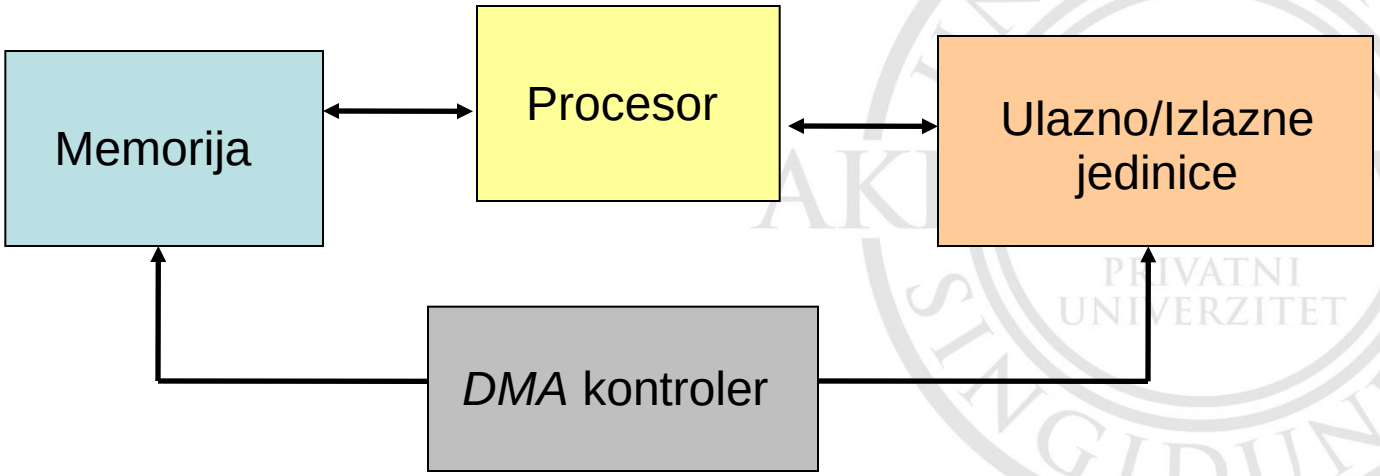
DMA ciklus

- *DMA* kontroler obaveštava procesor da ima potrebe za *DMA* prenosom.
- Procesor završava ciklus na magistrali i šalje dozvolu *DMA* kontroleru da može da obavi prenos (procesor inicira, tj. daje dozvolu za prenos).
- *DMA* kontroler obavlja prenos (generiše adrese i signale upisa/čitanja), a za to vreme procesor može da radi nešto što ne zahteva pristup magistrali.
- *DMA* kontroler obaveštava procesor da je prenos završen i da je magistrala slobodna.

DMA mehanizam

Primena *DMA* prenosa

- kada je potrebno preneti veliku količinu podatka, pa bi transfer preko procesora znatno usporio prenos
- kada je periferija relativno brza (primer takve periferije je hard disk)



Mehanizam prekida

- Koncept kojim se postiže da procesor ne troši vreme čekajući na spoljašnje događaje.

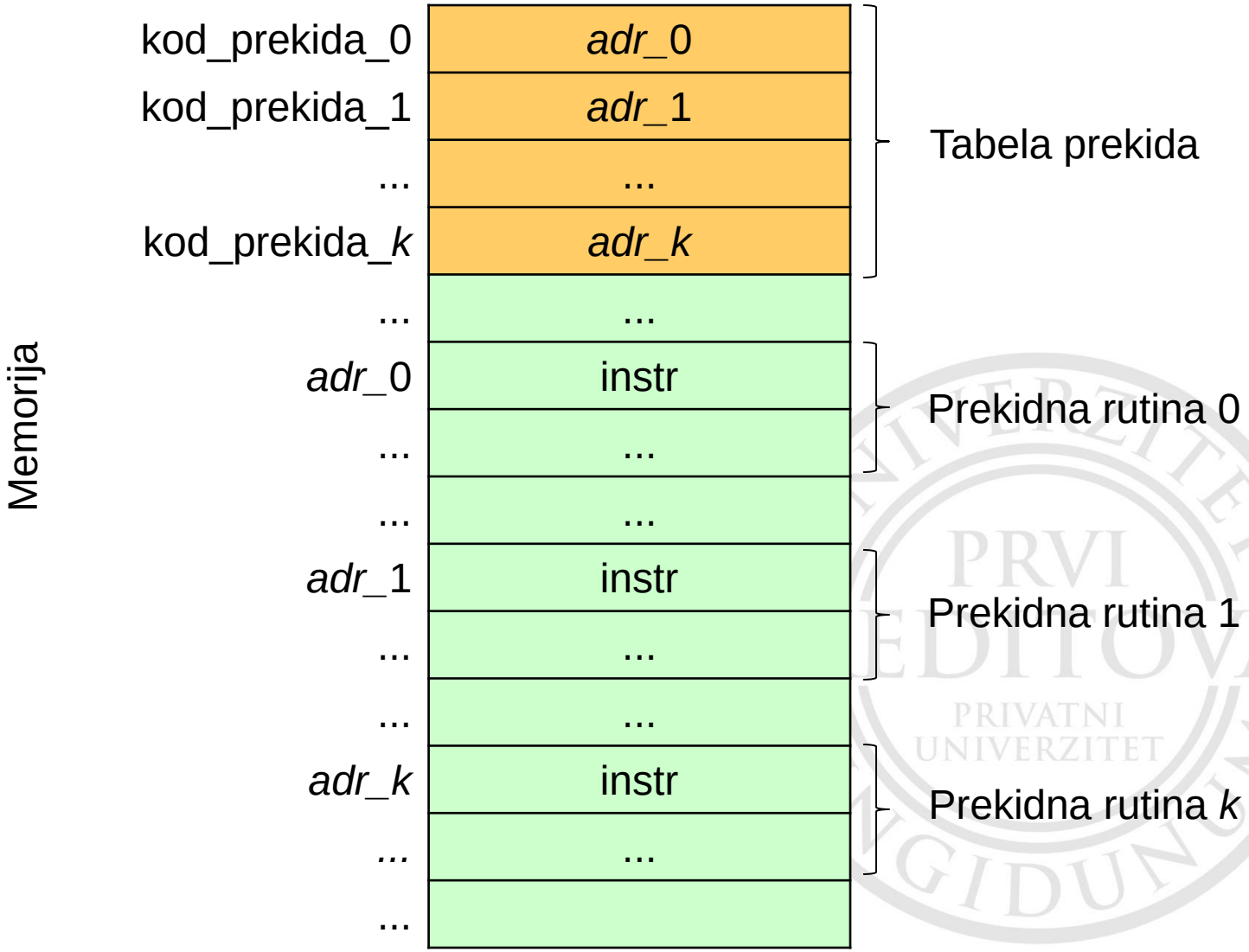
U sistemu bez mehanizma prekida:

- ako procesor treba da odštampa veći sadržaj, morao bi u delovima da ga šalje štampaču i da čeka da štampač odštampa prispeli deo pre nego što mu pošalje sledeći deo
- za to vreme, procesor ne bi ništa radio.

U sistemu sa mehanizmom prekida:

- procesor pošalje deo podataka štampaču, a zatim obavlja svoje aktivnosti
- kada štampač završi štampanje prispelog dela, šalje zahtev za prekidom procesoru, kako bi mu ovaj poslao naredni deo sadržaja.

Mehanizam prekida



Mehanizam prekida

Opsluživanje prekida

- Periferija šalje procesoru zahtev za prekidom koji sadrži njen kôd prekida (identifikator konkretne periferije).
- Po prijemu zahteva, procesor završava tekuću instrukciju, prekida izvršavanje programa i pristupa opsluživanju periferije.
- Svaki procesor može da opsluži određeni skup prekida; za svaki prekid postoji odgovarajuća prekidna rutina u memoriji; na osnovu dobijenog kôda prekida, procesor iz tabele prekida čita adresu prekidne rutine i izvršava je; u rutini se nalaze instrukcije kojima se opslužuje periferija.
- Po završetku prekidne rutine, procesor se vraća na izvršavanje prekinutog programa.

Mehanizam prekida

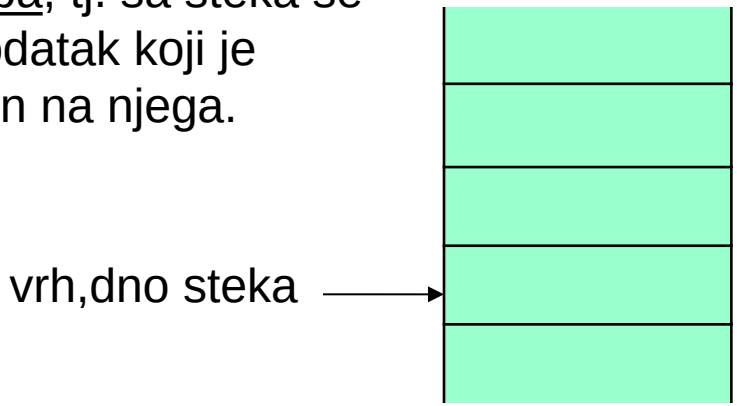
Kontekst procesora

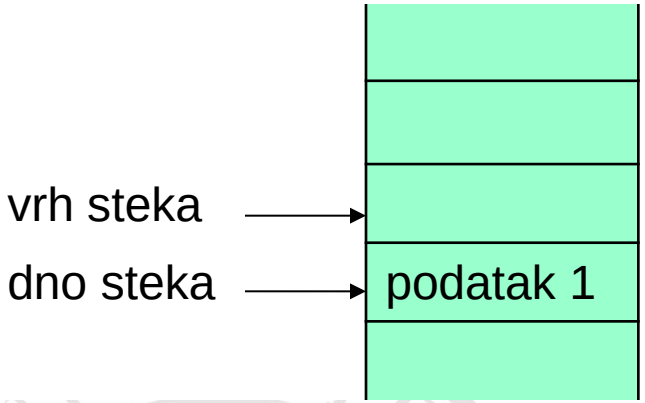
- Da bi mehanizam prekida mogao uspešno da funkcioniše, neophodno je obezbediti da se pri povratku iz prekidne rutine procesor nađe u potpuno istim uslovima u kojima je bio kada je počeo opsluživanje prekida; ti uslovi se nazivaju kontekst procesora.
- Pe nego što pređe na izvršavanje odgovarajuće prekidne rutine, procesor mora da sačuva tekući kontekst koga čine sadržaji pojedinih registara procesora (PC obavezno).
- Kontekst procesora se mora sačuvati zato što prekidna rutina koristi iste procesorske registre kao i prekinuti program, a obično menja njihov sadržaj.
- Kontekst procesora se obično čuva na steku.

Mehanizam prekida

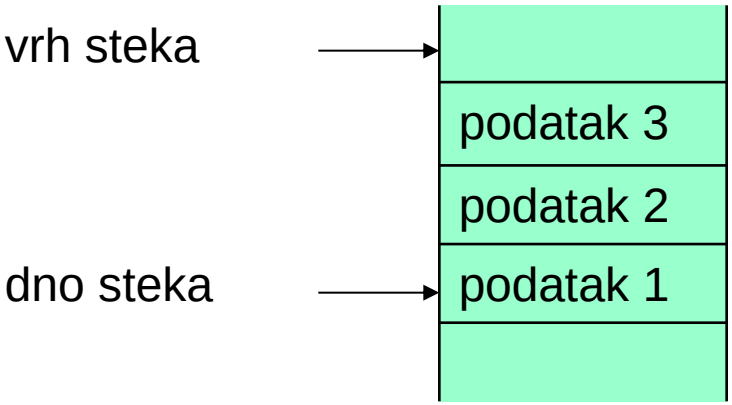
Stek podržava LIFO (*last in, first out*) disciplinu pristupa, tj. sa steka se najpre uzima podatak koji je poslednji stavljen na njega.

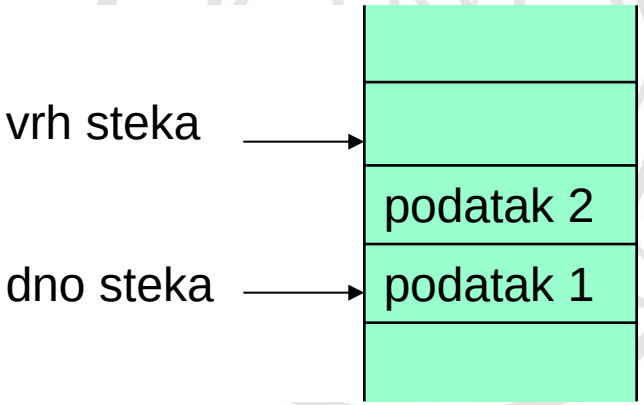
Upis podatka u stek





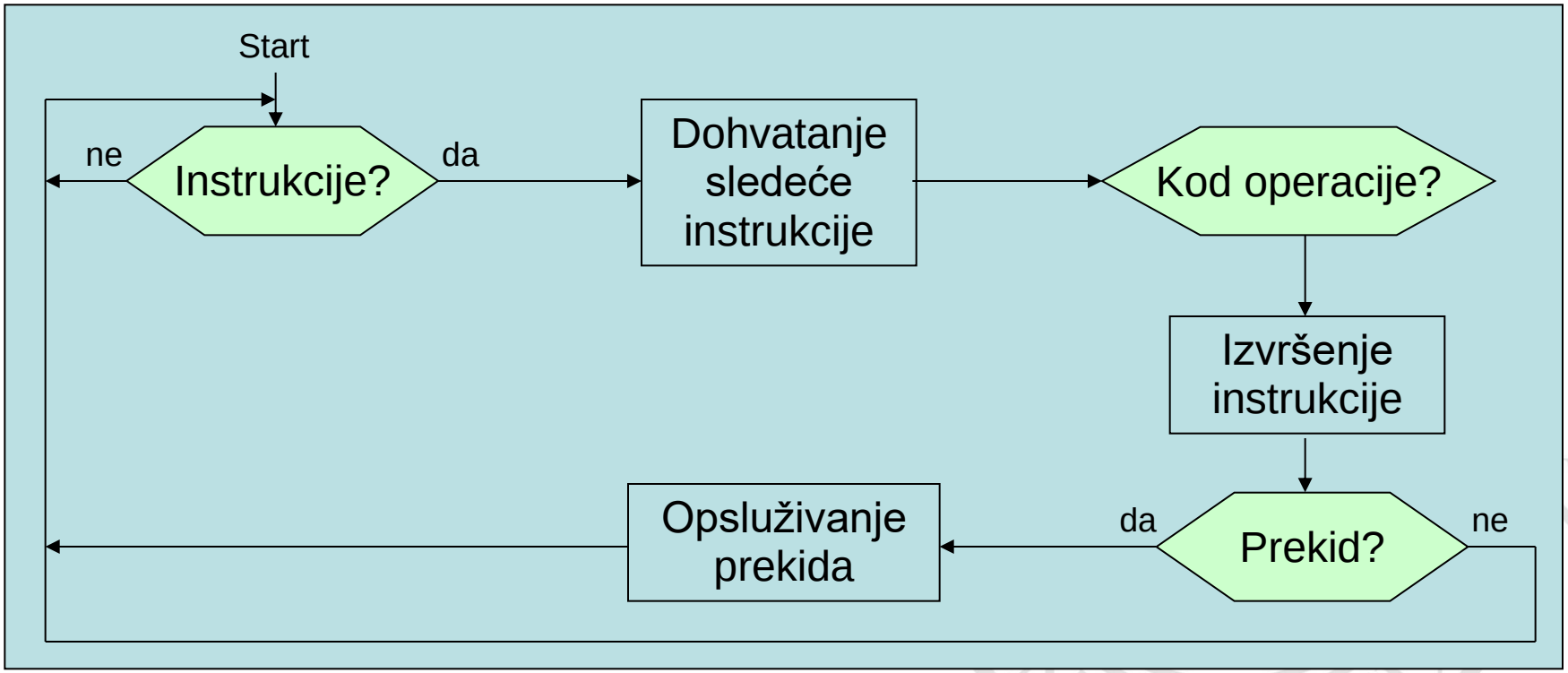
Čitanje podatka sa steka





Mehanizam prekida

Instrukcijski ciklus



Mehanizam prekida

Postoje dve vrste prekida:

- spoljašnji ili eksterni prekidi - prekidi koji dolaze od periferija
- unutrašnji prekidi - prekidi koji su posledica izvršavanja instrukcije prekida, ili posledica neke neregularnosti u izvršavanju tekuće instrukcije

- Prekidima se pridružuju prioriteti koji ukazuju na njihovu važnost.
- Prednost u opsluživanju imaju prekidi višeg prioriteta.
- Interni prekidi su višeg prioriteta od eksternih.

Šta smo naučili?