

11. VIRTUALNI MEMORIJSKI SUSTAV

- Problemi s memorijom u jedno- i višekorisničkom sustavu
- Memorijska hijerarhija
- Fizički i logički adresni prostori
- Vremenska i prostorna lokalnost
- Denningov model memorije
- Straničenje
- Segmentacija

- Proizvođači računalskih sustava isporučuju sustave s glavnom (radnom) memorijom kapaciteta od nekoliko desetaka do stotinu i više stotina M bajtova
- Sekundarna memorija – nekoliko desetaka ili stotina (i više) G bajtova.

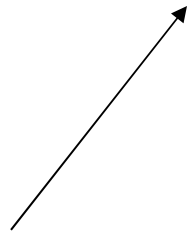
Problem: Kapacitet glavne memorije otvoreno pitanje odnosa **performansa/cijena**

Posljedica: Nesklad između memorijskih zahtjeva programa i kapaciteta stvarne, fizičke memorije

Jednokorisnički računalni sustav:

- Problem se rješavao uporabom postupka **prekrivanja** ili **preklapanja** (engl. **Overlay**):
 - programer je dijelio program na nekoliko programskih blokova (modula) i blokova podataka
 - svi programski blokovi i blokovi podataka **nisu istodobno potrebni** tijekom izvođenja programa
 - u glavnoj se memoriji nalazi programski modul koji se trenutno izvršava i njemu potreban blok podataka
 - u glavnoj se memoriji **trajno** pohranjuje onaj dio programa kojim se upravlja premještanje blokova između glavne memorije i sekundarne memorije

- svi ostali programski moduli su smješteni u **sekundarnoj memoriji** i premještaju se u glavnu memoriju upravo kad su potrebni
- moduli se premještaju u isto područje glavne memorije u kojem su bili pohranjeni prethodni moduli



PREKLAPANJE !!!

Značajka: Statičko rukovanje memorijom

Pojavom višekorisničkih računalnih sustava postupak prekrivanja ili preklapanja je **nedjelotvoran i skoro neizvodljiv**:

- zbog **istodobnog** postojanja više programa različitih korisnika u glavnoj memoriji
- zbog smanjenja raspoloživog prostora za svakog korisnika u glavnoj memoriji
- zbog prisutnosti više aktivnih programskih modula različitih korisnika – uporaba zajedničkih sustavskih modula → uvođenje dodatnih zaštitnih mehanizama pristupa pojedinim programskim modulima
- zbog toga što se podatkovne strukture mijenjaju dinamički tijekom izvođenja programa
- zbog potrebe razmještanja programskih modula tijekom izvođenja programa

RJEŠENJE: Dinamičko rukovanje memorijom

Virtualni memorijski sustav

(lat. virtus – hrabrost, snaga, vrlina – snažan, jak sposoban za djelovanje, no skriven, koji se ne pojavljuje ali se može pojaviti)

- problem kapaciteta glavne memorije rješava se upotrebom **memorijske hijerarhije**

Uporabom tog arhitektonskog koncepta ostvaruje se sljedeći cilj:

Glavna ili primarna memorija se prividno (virtualno) pojavljuje kao memorija koja ima kapacitet sekundarne memorije (npr. nekoliko desetaka ili stotina G bajtova) a brzinu ima jednaku brzini najbrže (ili skoro najbrže) memorije u memorijskoj hijerarhiji.

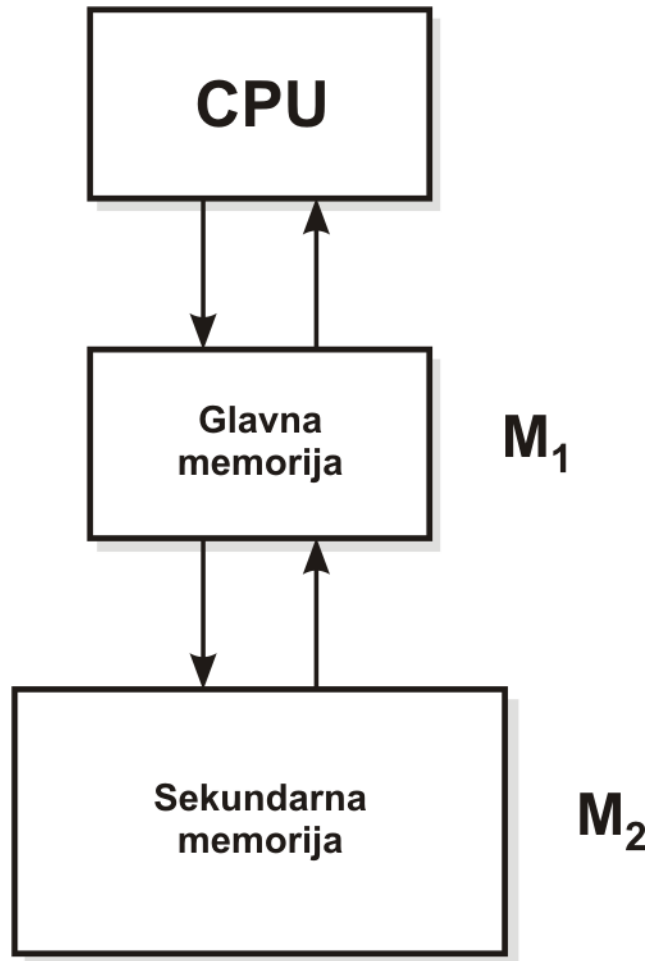
Memorijska hijerarhija:

$(M_1, M_2, M_3, \dots M_n)$

M_i je “podređena” memoriji M_{i-1}

Procesor komunicira s prvim članom hijerarhije (M_1)

Primjer:



c_i – cijena po bitu

t_{ai} – vrijeme pristupa

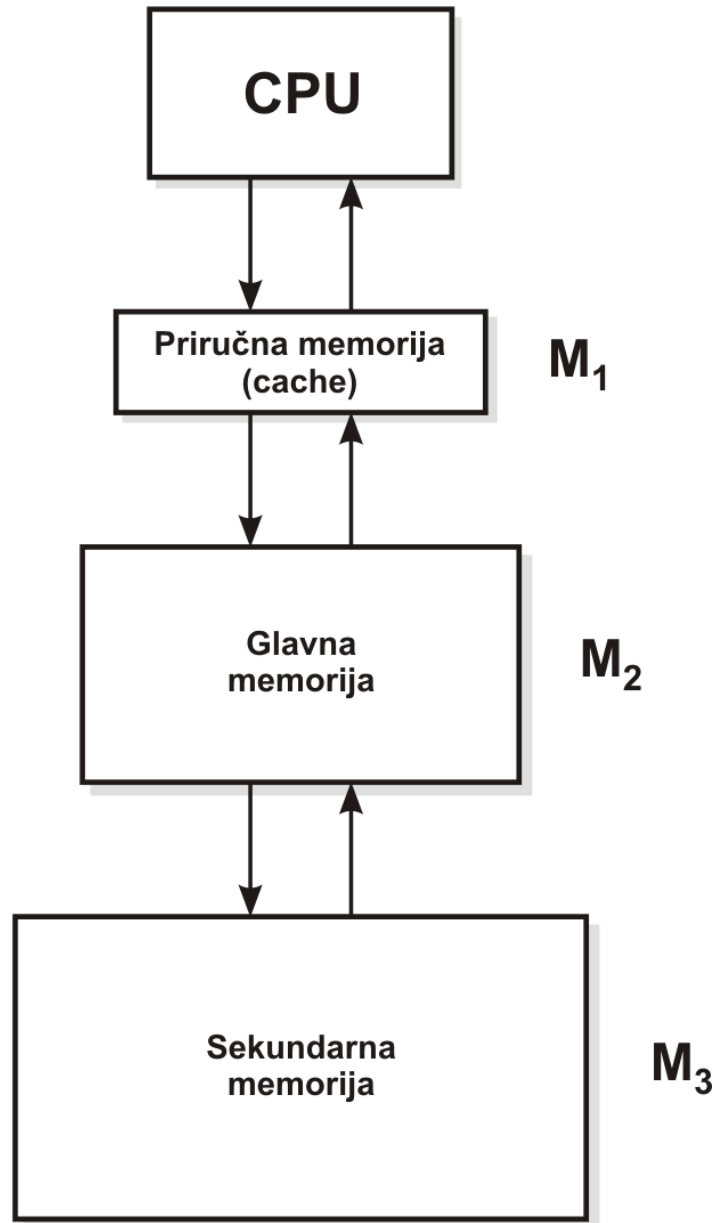
S_i – kapacitet memorije

Vrijedi: $c_i > c_{i+1}$

$t_{ai} < t_{ai+1}$

$S_i < S_{i+1}$

Primjer:



Fizički i logički adresni prostor

Skup stvarnih, fizičkih memorijskih lokacija glavne memorije oblikuje **fizičku memoriju**



memorija priključena na sabirnicu procesora,
odnosno računala

Skup adresa koje su jednoznačno dodijeljene tim memorijskim (fizičkim) lokacijama predstavlja **fizički adresni prostor**

Logički adresni prostor skup je logičkih adresa.
Adresa koje upotrebljava programer ili koju generira program ili proces (dretva) kao najmanja programska jedinica naziva se **logička adresa**.



adresa koju generira procesor

Odnos između fizičkog adresnog prostora (FAP) i logičkog adresnog prostora (LAP):

- $LAP = FAP$ /računala na bazi 8-bitnih mikroprocesora/
- $LAP < FAP$ /računala na bazi 8-bitnih mikroprocesora – memorijske banke/
- $LAP > FAP$ /16-, 32-, 64-bitni mikroprocesori/

!!!

Problem: za $LAP \gg FAP$ odrediti funkciju f :

$f: LAP \rightarrow FAP$

$$f: \text{LAP} \rightarrow \text{FAP}$$

$$\text{LAP} = \{0, 1, 2, \dots, N-1\}$$

$$\text{FAP} = \{0, 1, 2, \dots, M-1\}$$

$$\text{vrijedi: } N \gg M$$

$$f: \text{LAP} \rightarrow \text{FAP} \cup \phi$$

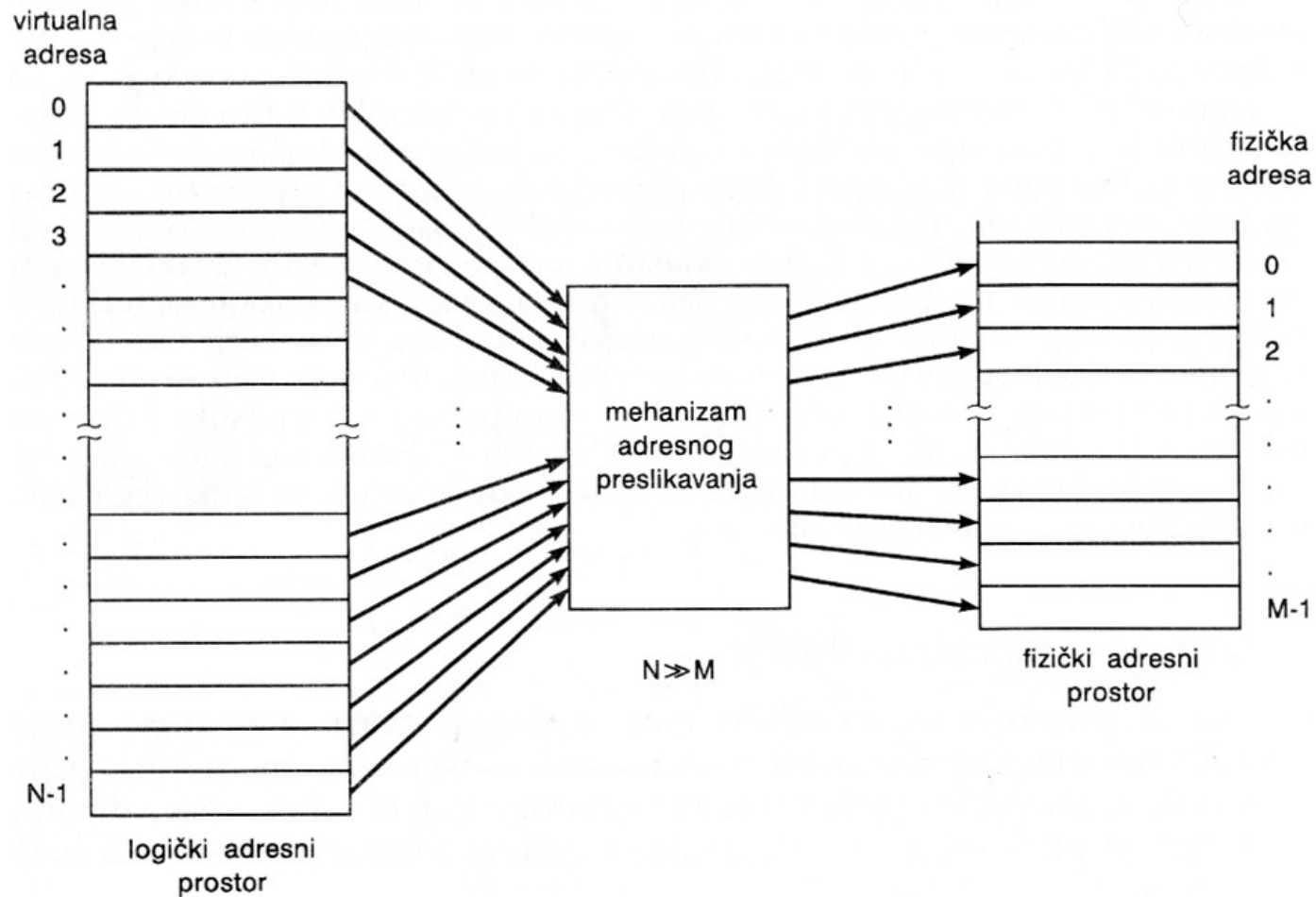
Neka je $a \in \text{LAP}$

Funkcija f je definirana kao:

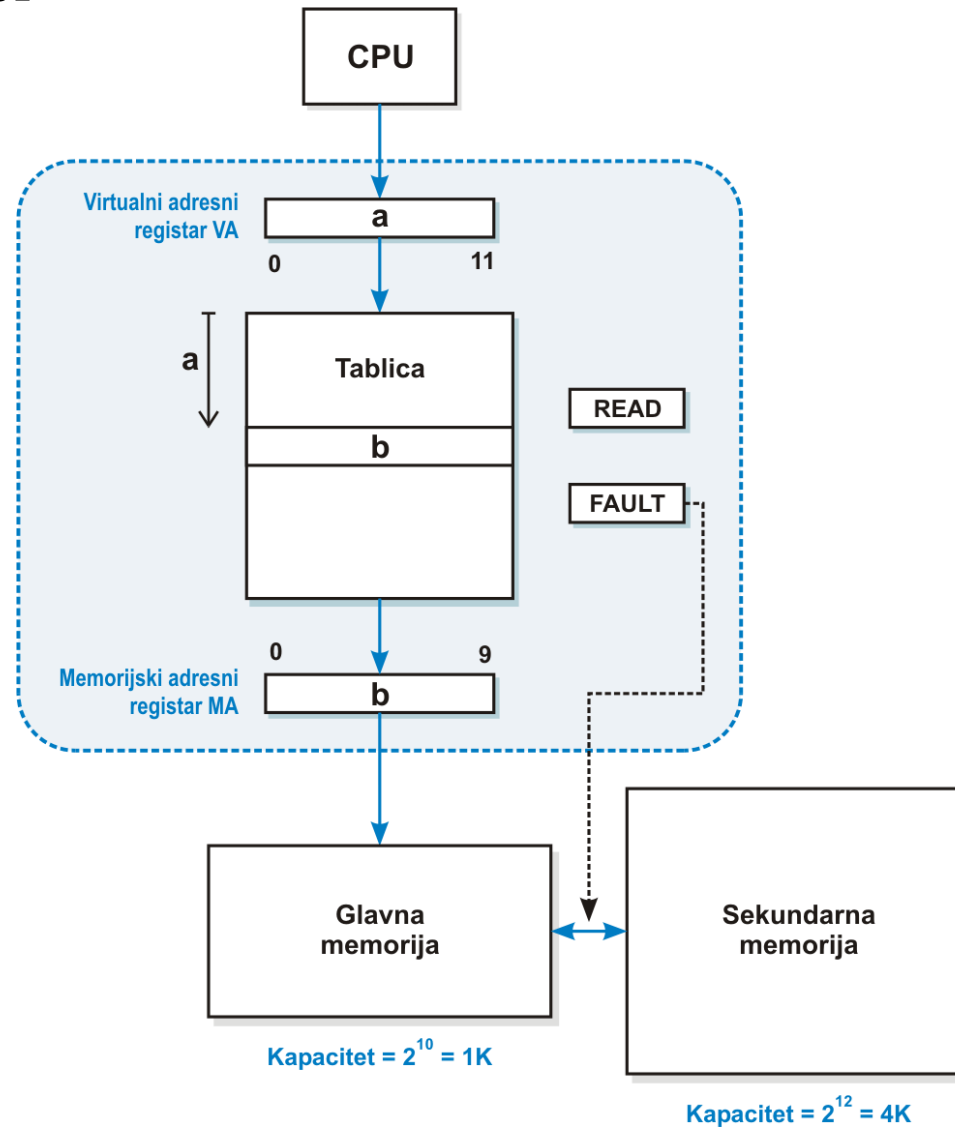
$f(a) = a'$ ako se podatak s virtualnom adresom a nalazi na adresi a' u fizičkoj memoriji ($a' \in \text{FAP}$)

$f(a) = \phi$ označava **PROMAŠAJ** (engl. Missing-item fault)

Adresno preslikavanje:

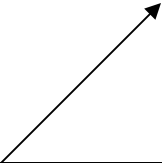


Denningov model



Denningov model ima (namjerno) ugrađenu nelogičnost:

Tablica preslikavanja ima broj elemenata jednak broju adresa u logičkom prostoru (kapacitet sekundarne memorije)?!



Broj registara potreban za izvedu tablice preslikavanja premašuje kapacitet fizičke memorije

Tablicu preslikavanja treba smanjiti!

Rješenje:

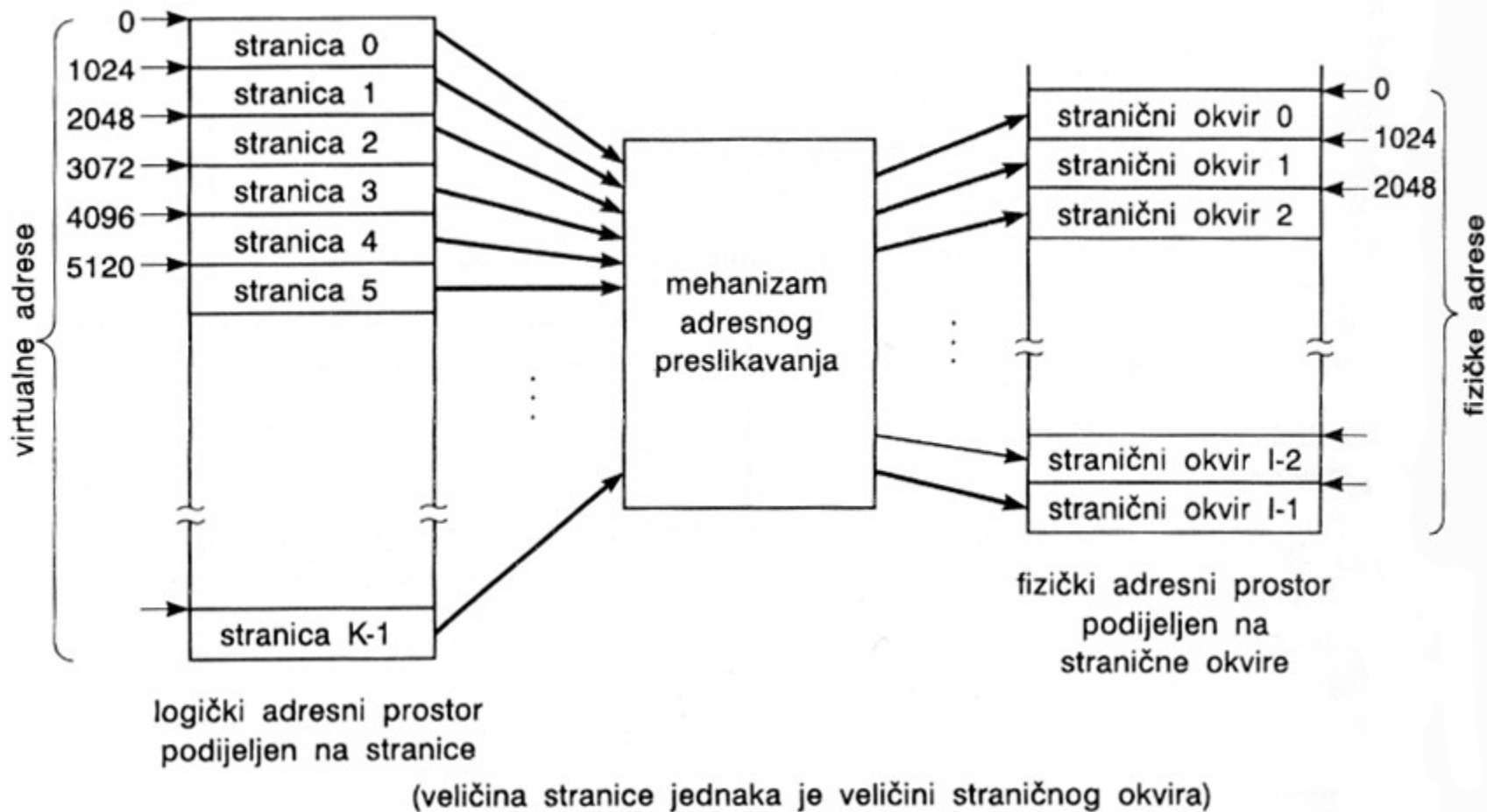
Element u tablici preslikavanja neka sadrži adresu bloka podataka umjesto adrese pojedinačno naslovljavanog podatka (u širem smislu te riječi)

Dijeljenje logičkog i fizičkog adresnog prostora na blokove!

Blokovi = stranice /ako su čvrste duljine/

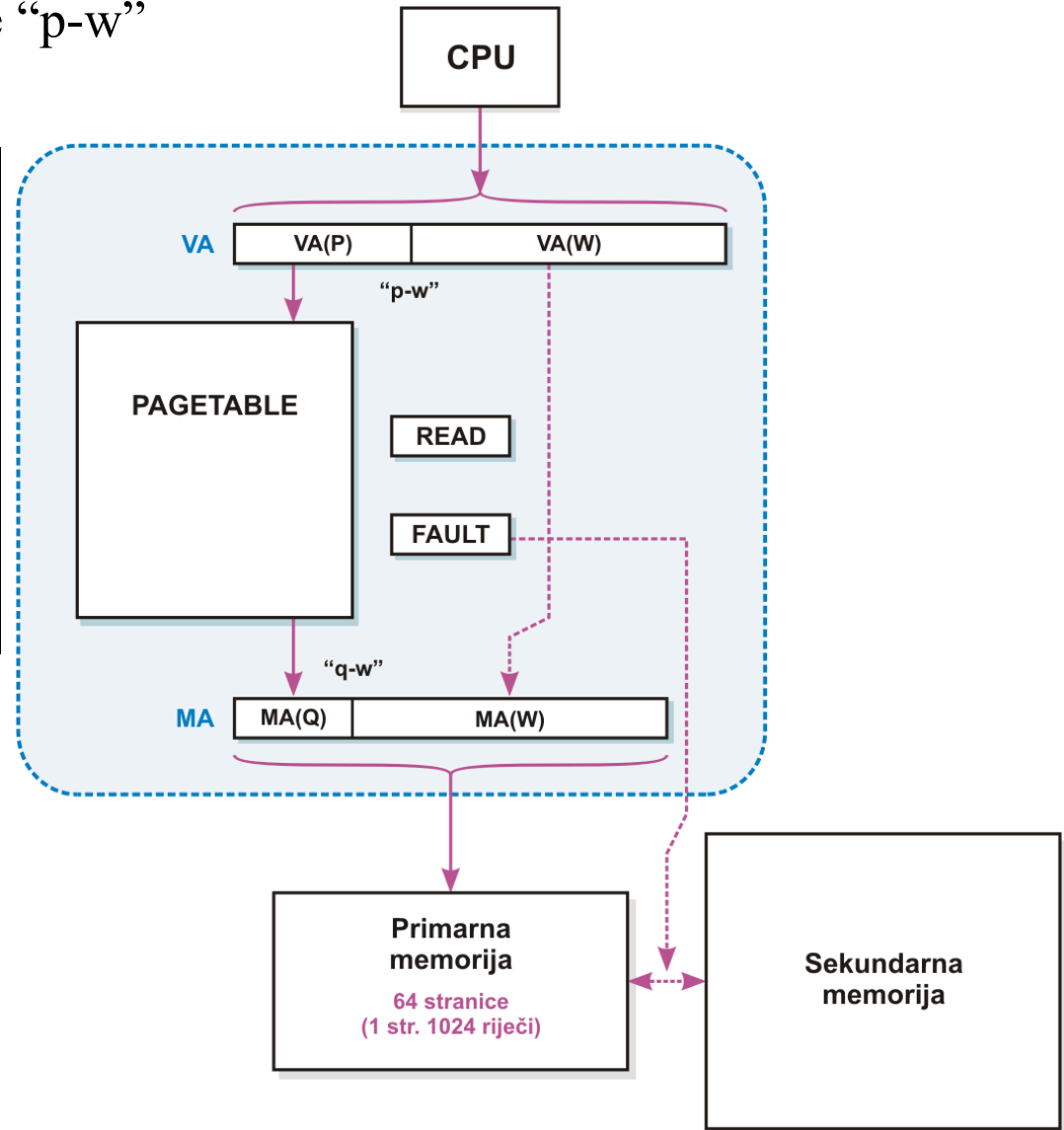
Blokovi = segmenti /ako su promjenjive duljine/

Straničenje (engl. Paging)

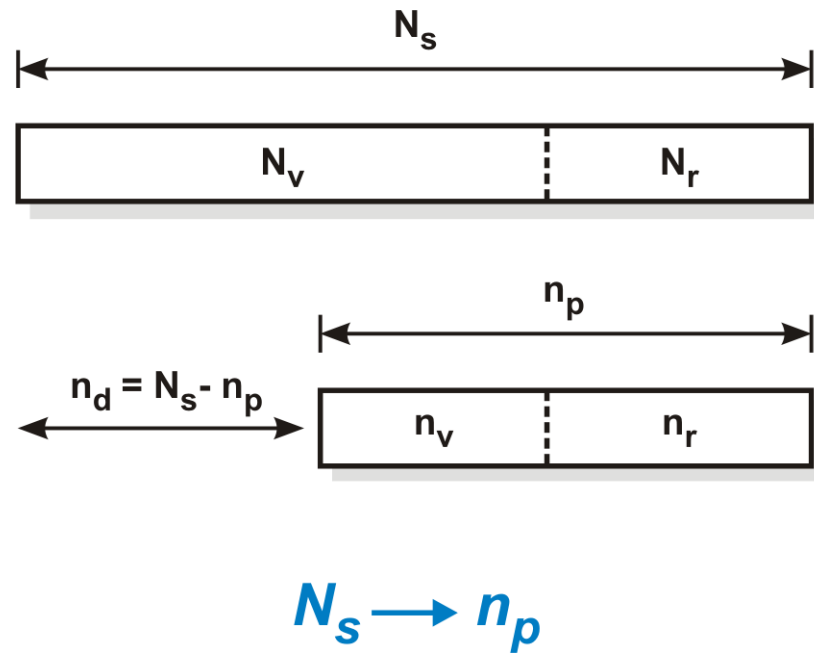


Translacija zadane virtuale adrese “p-w”
u fizičku adresu “q-w”:

```
VA ← “p-w”, FAULT ← 0  
READ ← 1  
MA(q) ← PAGETABLE(VA(p)),  
      MA(w) ← VA(w)  
IF (MA(q) = 0) THEN  
    FAULT ← 1,  
END
```



Funkcija translacije adrese:



Omjer pogotka H (engl. Hit ratio)

H – vjerojatnost da se logička adresa koja je generirana od procesora odnosi na informaciju koja se nalazi u glavnoj memoriji (M_1)

M_1, N_1 - broj pozivanja

M_2, N_2 - broj pozivanja

$$H = N_1 / (N_1 + N_2)$$

Omjer promašaja (engl. Miss ratio)

$$\text{Miss_ratio} = 1 - H$$

Performansa memorijskog sustava zavisi od:

- statističkih svojstava pozivanja
/ redoslijed i frekvencija pojavljivanja logičkih adresa/
- veličine bloka (stranice) i kapaciteta glavne memorije
- strategije zamjene stranica i tehnike adresnog preslikavanja

Lokalnost

Vremenska lokalnost – očituje se u tomu što će program u bliskoj budućnosti naslovljavati (referencirati) one programske i podatkovne objekte koje je naslovljavao i u bližoj prošlosti.

Prostorna lokalnost – se manifestira u tomu što će program naslovljavati u skoroj budućnosti one programske i podatkovne objekte koji imaju **adrese bliske** onima koje su upotrebljavane u bližoj prošlosti.

Lokalnost – izražena **radnim skupom** $WS(t, h)$ (engl. **Working set**).

$WS(t, h)$ predstavlja skup memorijskih lokacija ili blokova koji su u vremenu t referencirani u posljednjih h pozivanja:

$$WS(t, h) = \{i \in N \mid i \in r_{k-h}, r_{k-h+1}, \dots, r_k\}$$



skup stranica /blokova/ $N = \{1, 2, \dots, n\}$ koji čine neki program

R – slijed naslovljavanja stranica:

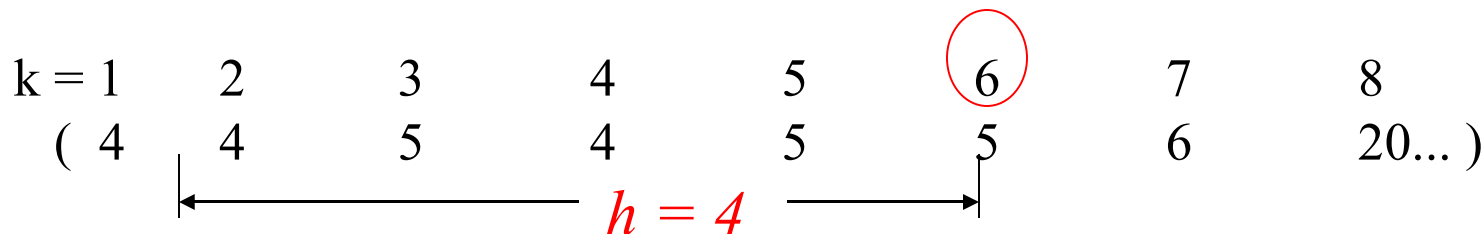
$$R = (r_1, r_2, \dots, r_k, \dots)$$

$WS(k, h)$ – skup stranica koje se javljaju u “oknu” naslovljavanja veličine h (gledajući unatrag u slijedu naslovljavanja stranica).

Primjer:

Neka je $R = (4, 4, 5, 4, 5, 5, 6, 20, 20, \dots)$ slijed naslovljavanja stranica

$$WS(6, 4) = \{4, 5\}$$



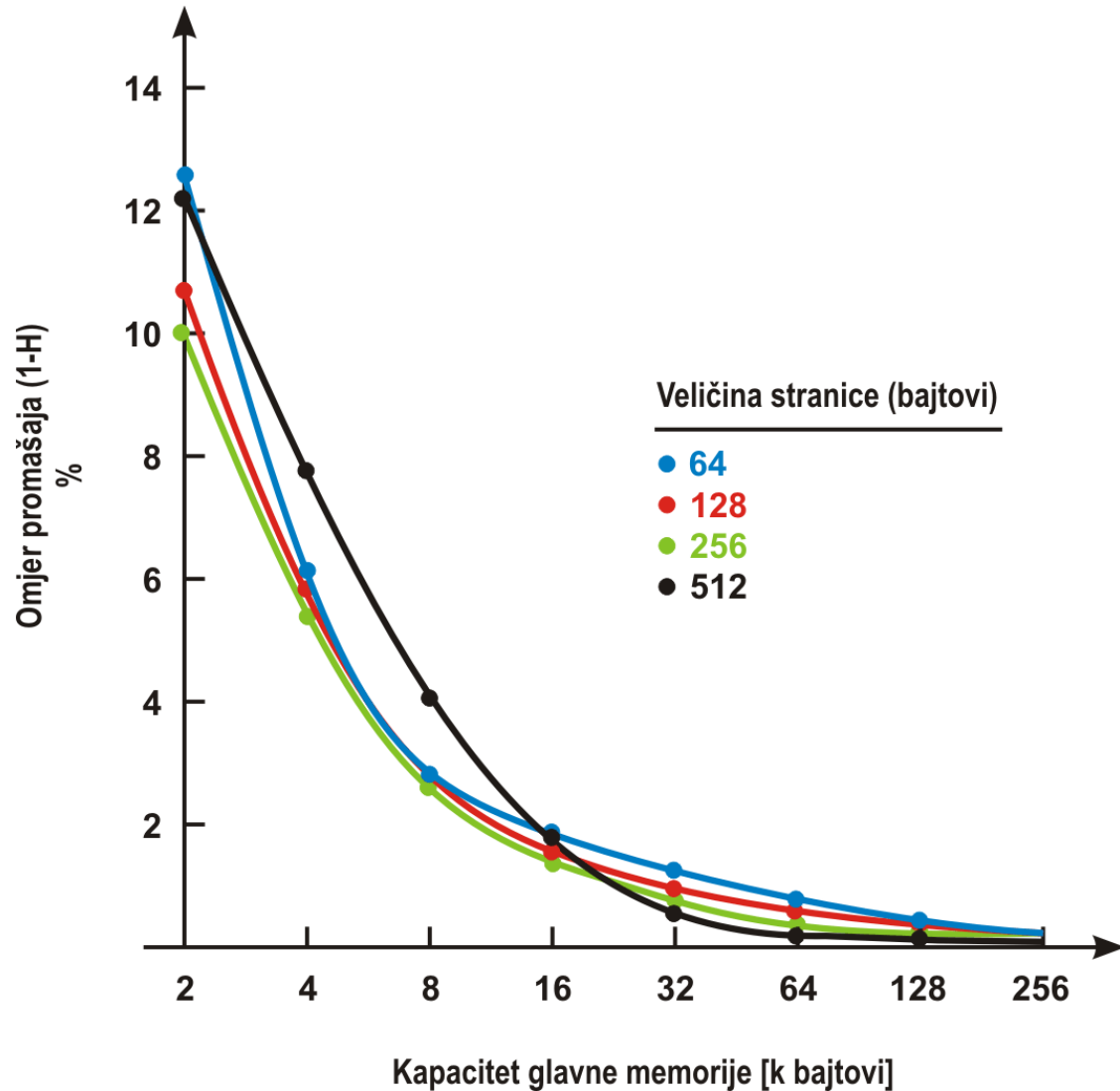
Primjer:

Način smještanja
stranica:

- potpuno
asocijativno
preslikavanje

Način zamjene
stranica:

- LRU algoritam
zamjene stranica

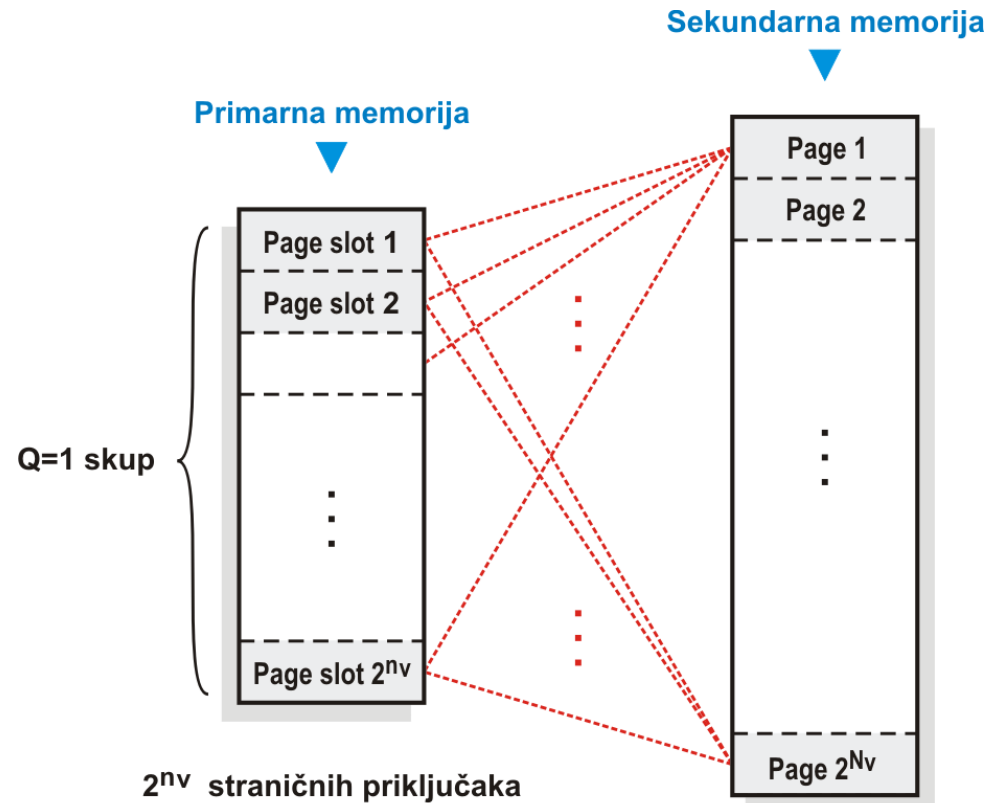


Način smještanja stranica određuje kako se stranice iz sekundarne memorije preslikavaju u stranice primarne memorije:

- potpuno asocijativno preslikavanje
- izravno (direktno) preslikavanje
- skupno asocijativno preslikavanje

Potpuno asocijativno
preslikavanje:

-stranica iz sekundarne
memorije može se se smjestiti
na bilo koji slobodni stranični
priključak



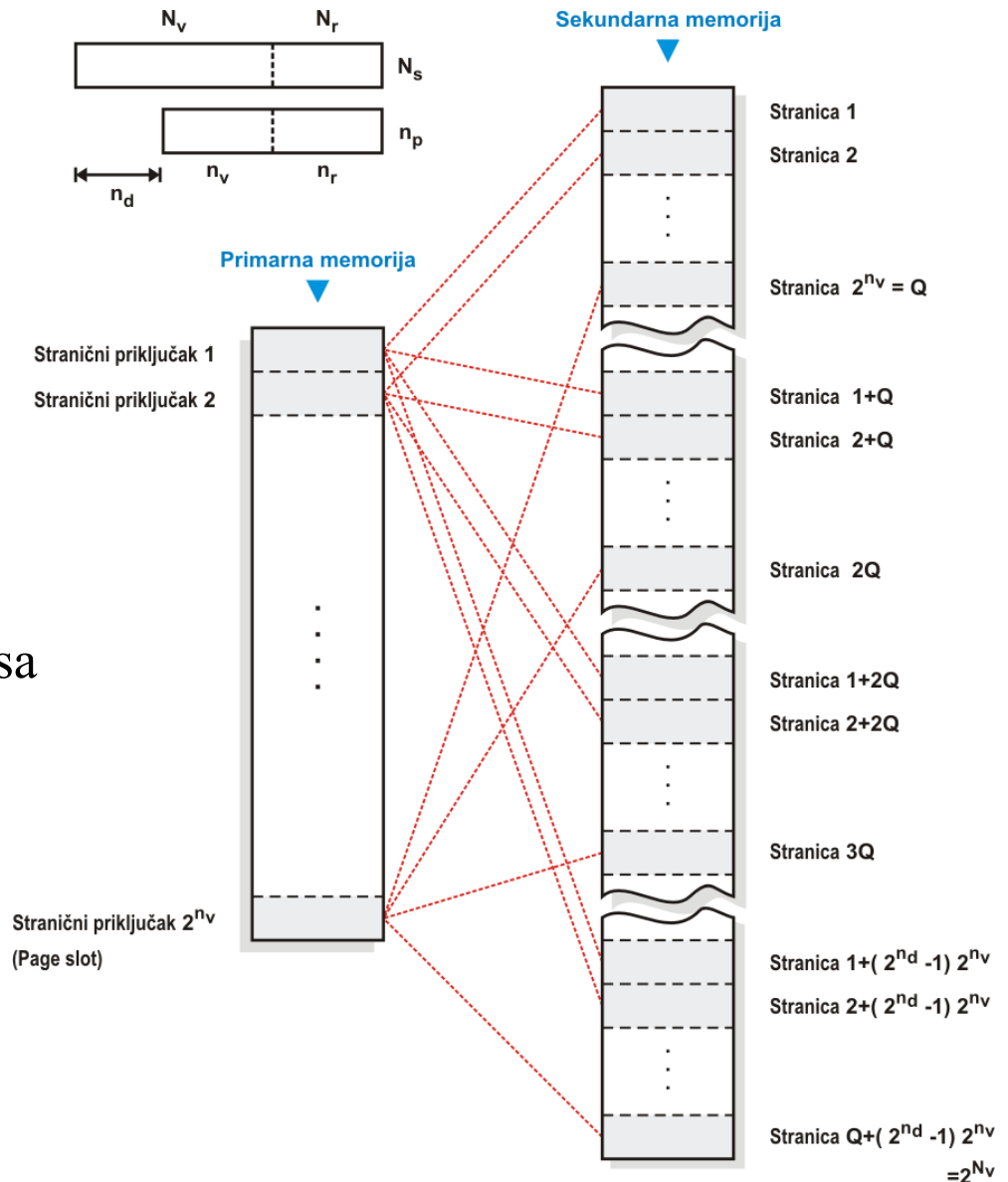
Izravno preslikavanje:

- svaka stranica iz sekundarne memorije može se smjestiti samo na određeni stranični priključak:

$$j = i \text{ (modulo } B_p\text{)}$$

Stranica iz sekundarne memorije sa straničnim brojem i smješta se na stranični priključak j ;

B_p – broj straničnih priključaka



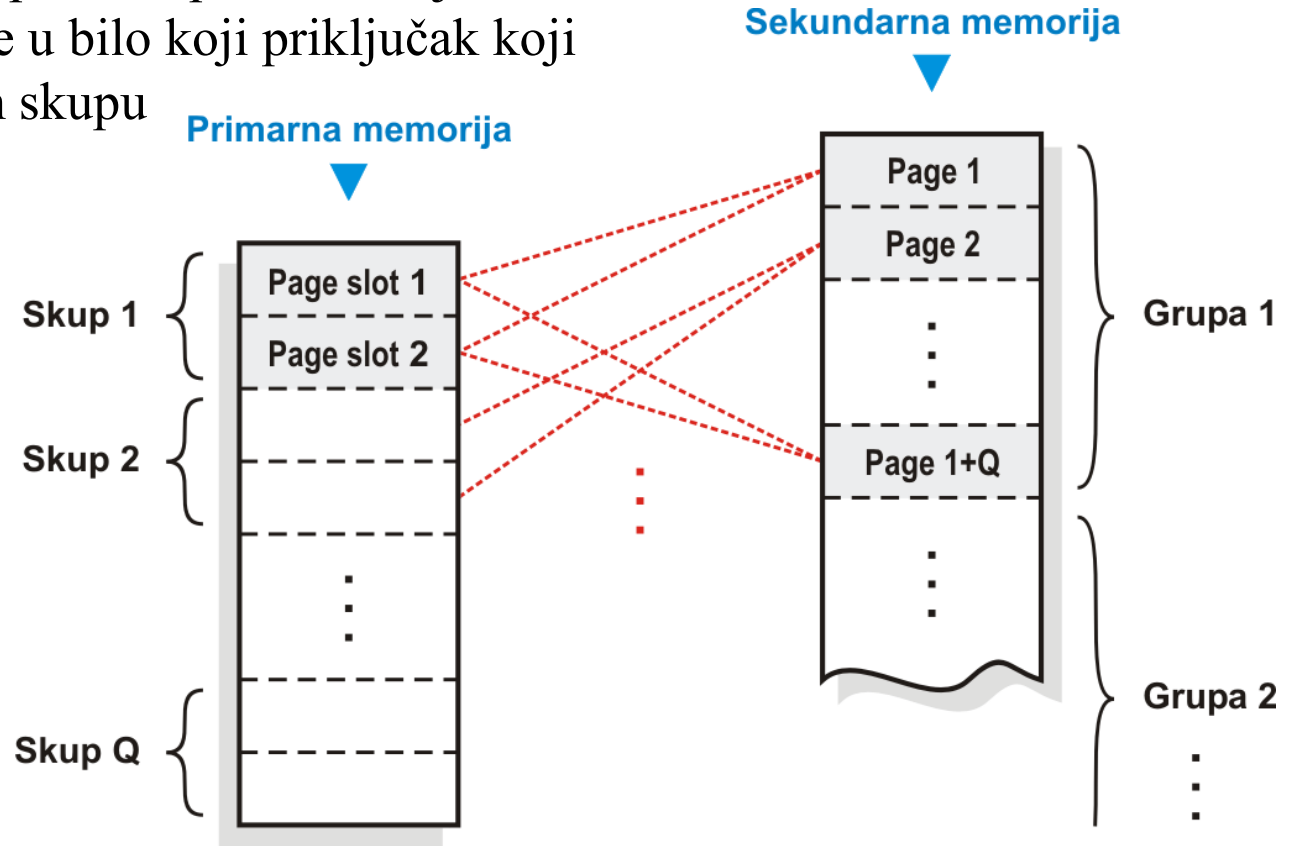
Skupno asocijativno preslikavanje (engl. Set associative):

Stranični priključci primarne memorije grupirani su u skupove tako da je dopušteno preslikavanje stranice iz sekundarne memorije u bilo koji priključak koji pripada odgovarajućem skupu

Stranica s indeksom i iz sekundarne memorije može se priključiti **na bilo koji slobodni priključak skupine j** :

$$j = i \text{ (modulo } B_s)$$

$$B_s = Q - \text{broj skupova}$$



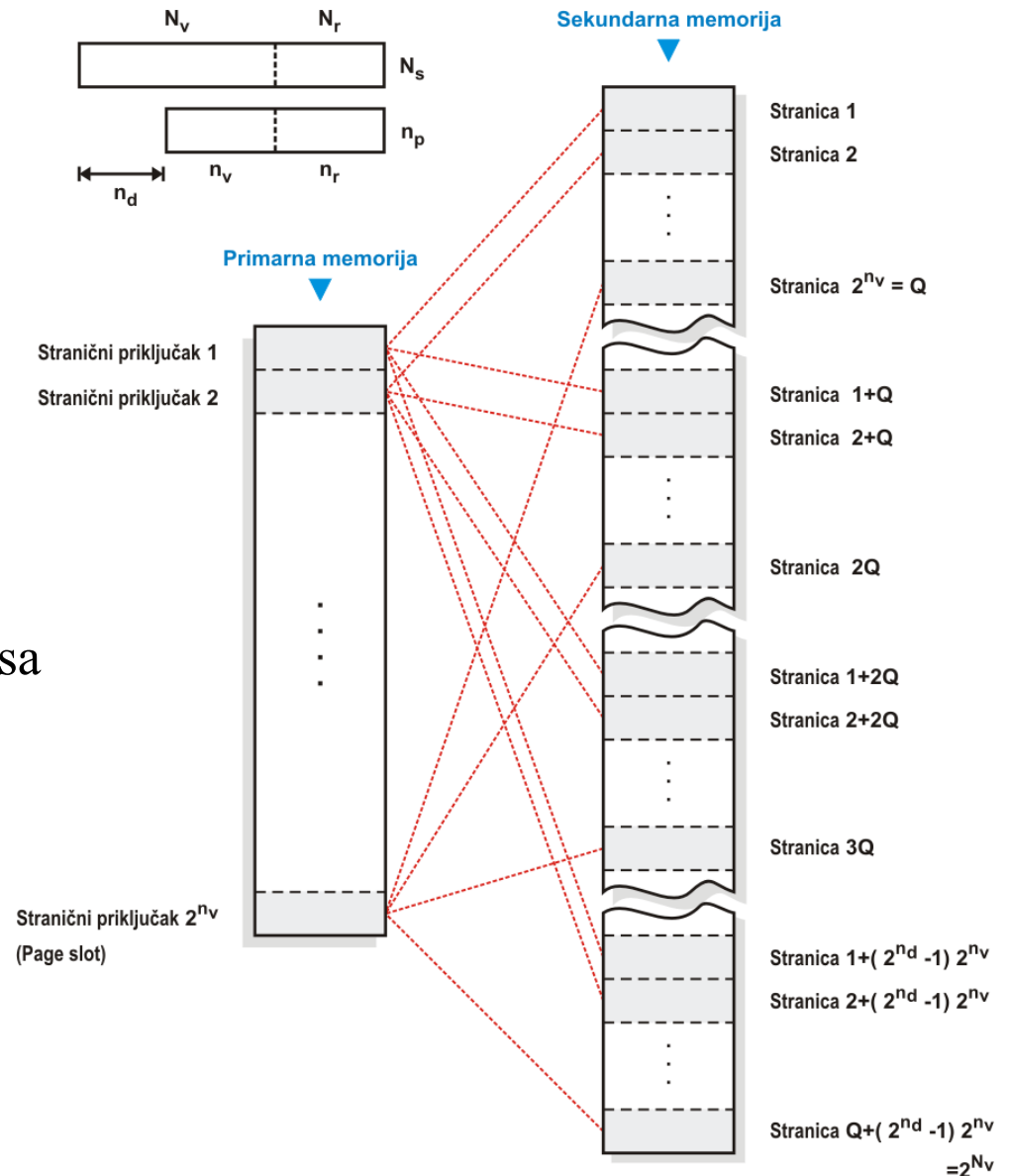
Izravno preslikavanje:

- svaka stranica iz sekundarne memorije može se smjestiti samo na određeni stranični priključak:

$$j = i \text{ (modulo } B_p\text{)}$$

Stranica iz sekundarne memorije sa straničnim brojem i smješta se na stranični priključak j ;

B_p – broj straničnih priključaka



Skupno asocijativno preslikavanje (engl. Set associative):

Stranični priključci primarne memorije grupirani su u skupove tako da je dopušteno preslikavanje stranice iz sekundarne memorije u bilo koji priključak koji pripada odgovarajućem skupu

Stranica s indeksom i iz sekundarne memorije može se priključiti **na bilo koji slobodni priključak skupine j** :

$$j = i \pmod{B_s}$$

$$B_s = Q - \text{broj skupova}$$

