Memorijski sustav

(Građa računala, Arhitektura i organizacija računarskih sustava, str. 265-357

- Memorijska hijerarhija
- Osnovne organizacijske i tehnološke značajke memorijskog sustava
- Glavna (primarna) ili radna memorija
- Sekundarna memorija

Memorijska hijerarhija

- memorija ili spremnik (engl. storage) je vrlo važna komponenta računarskog sustava
- u njoj se pohranjuju instrukcije (programi), podaci, međurezultati i rezultati
- vrijeme koje se troši na komunikaciji između procesora i memorijske jedinice u velikoj mjeri utječe na performansu računarskog sustava

• Procesor bi trebao imati brzi i neprekidni pristup memorijskoj jedinici vrlo velikog kapaciteta

Problem: memorija koja bi radila jednakom brzinom kao i procesor i imala kapacitet nekoliko stotina giga bajtova (GB; giga = 2^{30}) tera bajtova (TB; tera = 2^{40}) znatno bi povećala cijenu računarskog sustava /odnos performansa- cijena/

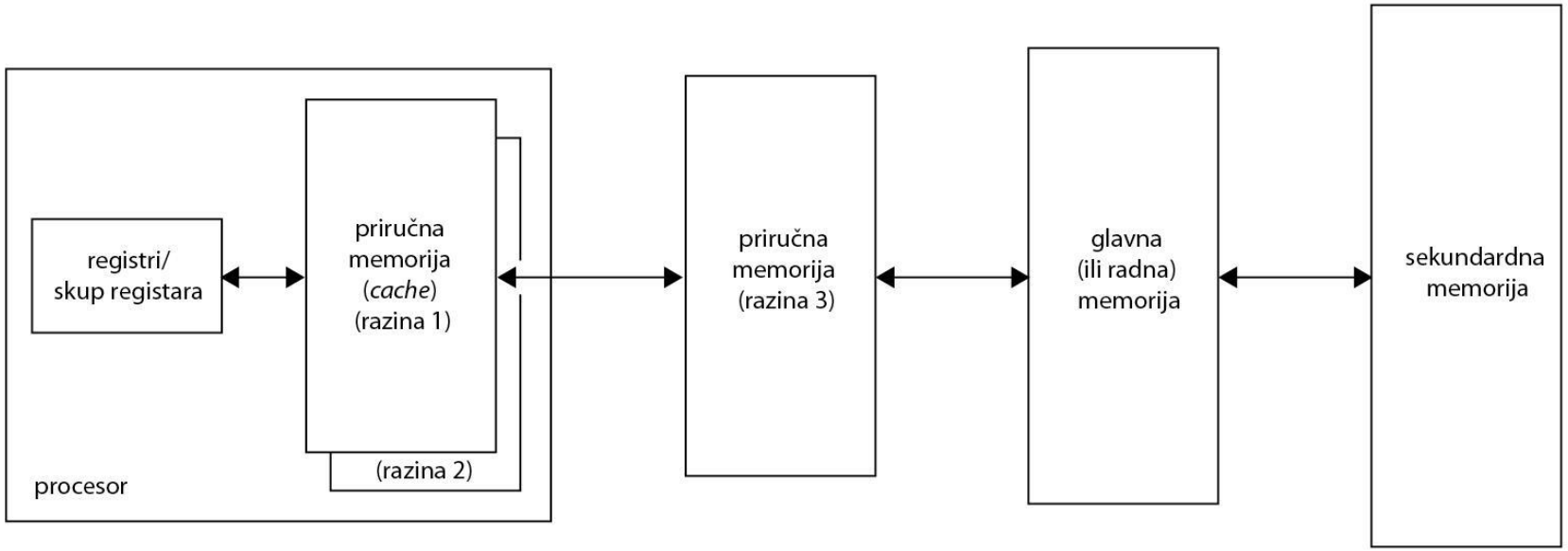
• Odnos između brzine procesora i memorije izražavamo pomoću latentnosti ili vremena odgovora

(vrijeme koje protekne između započinjanja i završetka nekog događaja)

- odnos latentnosti procesora memorije je 1: (3-5)

(npr. Pentium 4 – 15 ns; memorijski modul DDR SDRAM 52 ns)

Brzina memorije – vrijeme pristupa memoriji (engl. access time)
Kapacitet memorije: MB, GB, TB
Na temelju vremena pristupa, kapaciteta i cijene memorije u računarskom sustavu mogu se identificirati četiri glavne hijerarhijske razine:

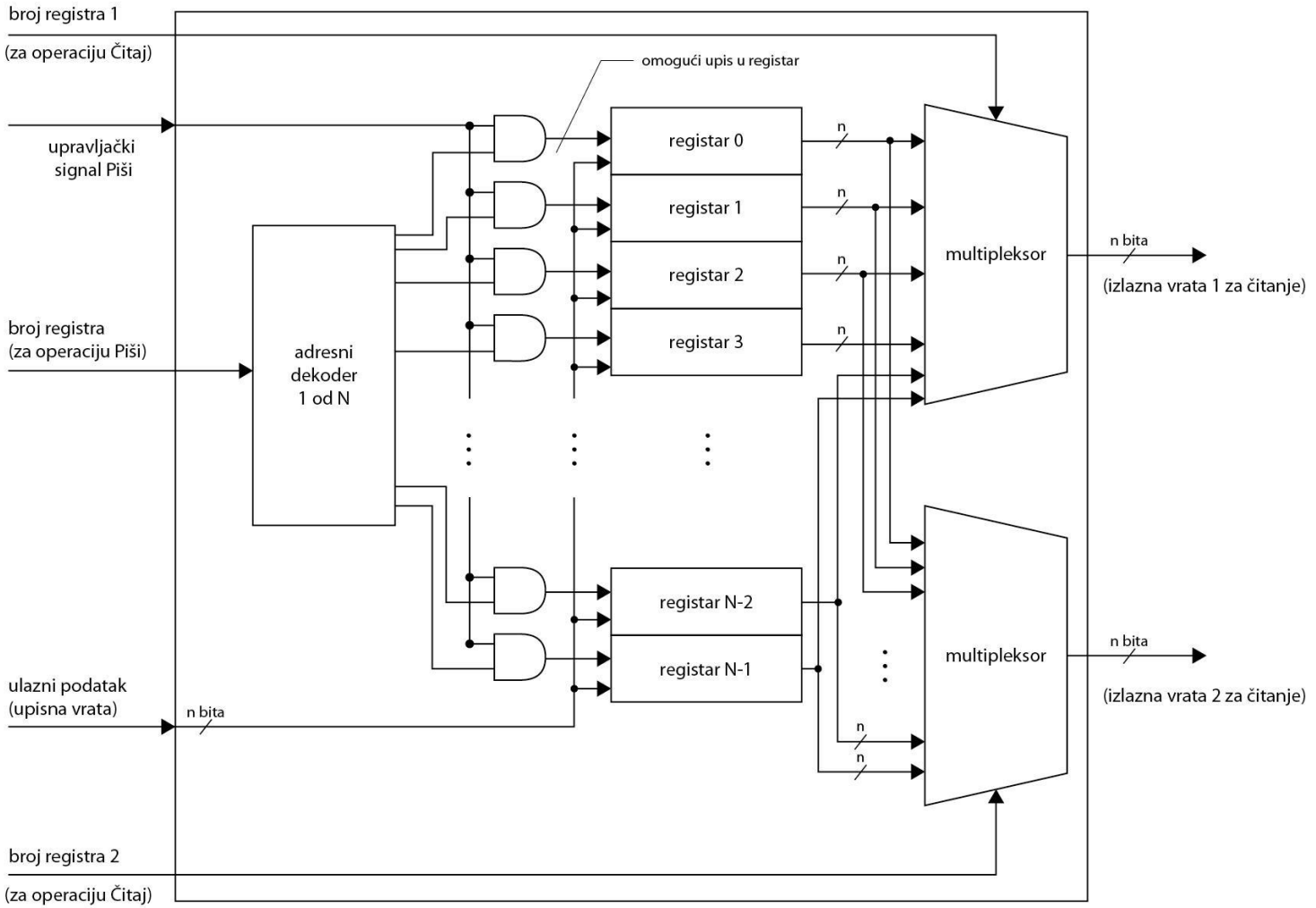


• Registri procesora ili skup registara opće namjene

- male ekstremno brze, višepristupne memorijske jedinice ostvarene na procesorskom čipu – imaju brzinu istog reda veličine kao i sklopovi procesora a kapacitet od npr. 16 ili 32 pa sve do nekoliko stotina i više riječi

- registri procesora obično su organizirani kao višepristupna memorijska jedinica koja oblikuje skup registara opće namjene

Izvedba skupa registara opće namjene s jednim upisnim vratima i dvojim izlaznim vratima



• Priručna (engl. cache) memorija

- većeg kapaciteta od skupa registara opće namjene ali nije nužno sporija

cache – skrivanje ili skrovito mjesto

- priručna memorija služi za pohranjivanje instrukcija (programskih segmenata) i podataka (podatkovnih segmenata)

razlika između skupa registara opće namjene i priručne memorije!

Priručna memorija organizirana kao:
- memorija za pohranjivanje instrukcija i podataka (I&D cache)
- izdvojena instrukcijska priručna memorija (I cache)
- izdvojena priručna memorija podataka (D cache)

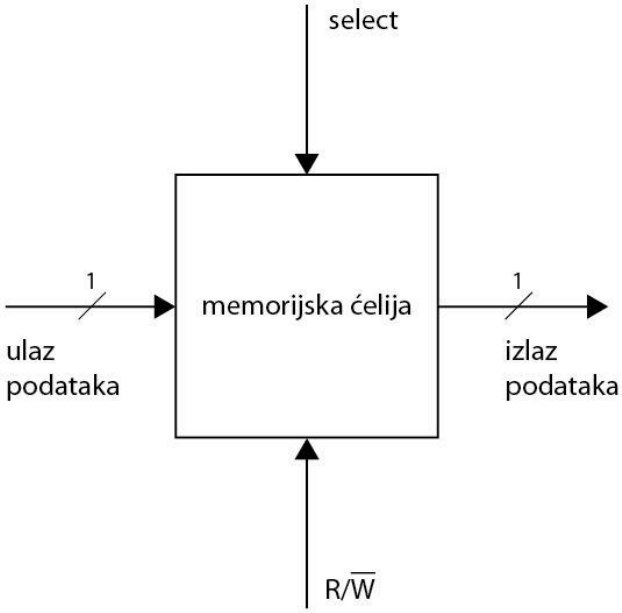
Procesor – I&D cache ili I cache + D cache
Fizički priručna memorija može biti smještena na samom procesorskom čipu (priručna memorija razine 1)
ili izvan procesorskog čipa (priručna memorija razine 2 i 3)
(Tehnologija VLSI dopušta izvedbu sve tri razine priručne memorije na procesorskom čipu)
Kapacitet priručne memorije – od nekoliko desetaka ili stotina KB pa sve do nekoliko MB i više

• Glavna (primarna) ili radna memorija

- prilično brza memorija velikog kapaciteta (nekoliko desetaka ili stotina GB) u kojoj se pohranjuju aktivni programi i podaci
-izvedena poluvodičkom tehnologijom sličnoj onoj koja se rabi za izvedbu skupa registara opće namjene ili priručne memorije ona je dva do pet puta sporija (veliki kapacitet, fizički "udaljena"od procesora)
VAŽNO: bez obzira na postojanje memorijske hijerarhije u računarskom sustavu, za procesor je mjerodavna jedino slika programa u napredovanju koja se odražava u sadržaju glavne memorije
Tehnologija: statički RAM (SRAM) i dinamički RAM (DRAM)

Memorijska ćelija

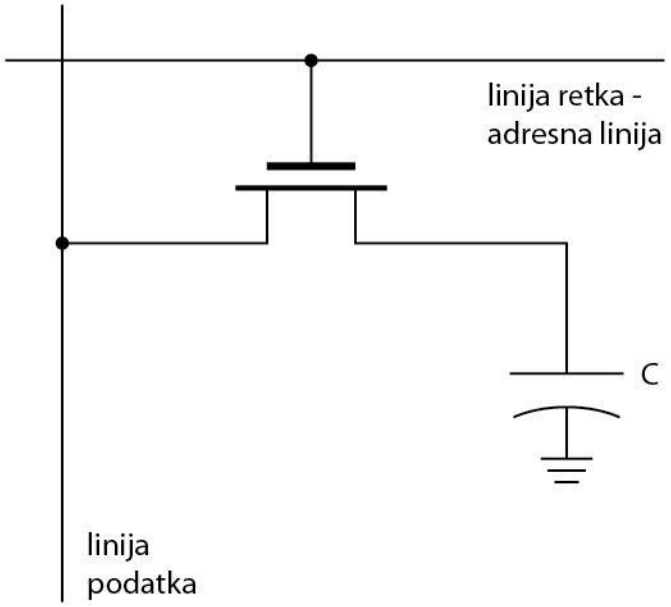
SRAM ili DRAM



- ima dva stabilna stanja koja se upotrebljavaju za prikaz 1 i 0
- omogućuje upis i pohranu (novog) stanja
- omogućuje čitanje pohranjenog stanja

• Statički se RAM sastoji od polja memorijskih ćelija koje se temelje na bistabilu. SRAM tipično zahtijeva šest tranzistora po bitu pohrane

• Dinamički RAM pohranjuje jednobitni podatak u jednotranzistorskoj memorijskoj ćeliji u obliku naboja u kondenzatoru



SRAM – važna brzina i kapacitet
DRAM – važna cijena po bitu i kapacitet

- gustoća (broj bitova po jedinici površine) DRAM-a je 4 - 8 puta veća u odnosu na SRAM
- SRAM je 30 – 100 puta brži (ali isto toliko puta skuplji!!!)

SRAM – za izvedbu priručnih memorija
DRAM – za izvedbu glavne memorije

Godina	Kapacitet čipa	Vrijeme memorijske periode (ns)
1980.	64 Kb	250
1983.	256 Kb	220
1996.	64 Mb	110
2000.	256 Mb	90
2004.	1 Gb	70
2006.	2 Gb	60

Tablica 9.2. Brzina i kapacitet DRAM za razdoblje od 1980. do 2006.

• Sekundarna memorija

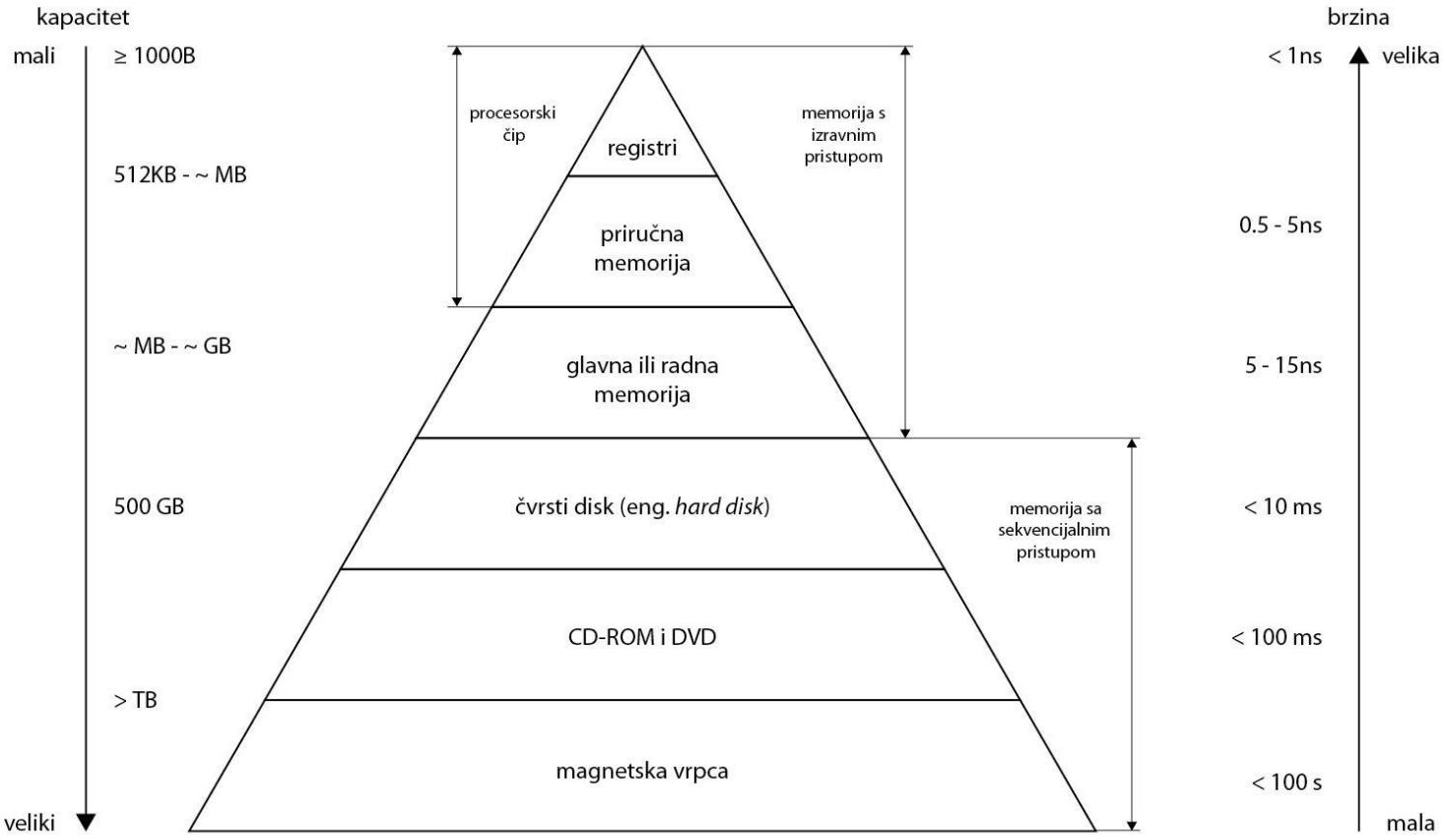
- vrlo velikog kapaciteta (nekoliko stotina GB ili nekoliko TB) ali puno sporija u odnosu na glavnu memoriju (vrijeme pristupa ms)

(magnetski diskovi, magnetske vrpce, optički diskovi)

Razina	1	2	3	4
Vrsta memorije	Registri ili skup registara	Priručna memorija	Glavna ili radna memorija	Sekundarna memorija (jedinica diska)
Kapacitet	< 1 KB	< 16 MB	< 512 GB	> T (tera) B
Tehnologija	Višeulazno-izlazna memorija CMOS	Na procesorskom čipu ili izvan procesorskog čipa realizirana memorija CMOS SRAM	CMOS DRAM	Magnetski disk
Vrijeme pristupa (ns)	0.25 - 0.5	0.5 - 25	50 - 250	5 000 000

Tablica 9.1. Tipične vrijednosti vremena pristupa i kapaciteta memorije za četiri memorijske razine.

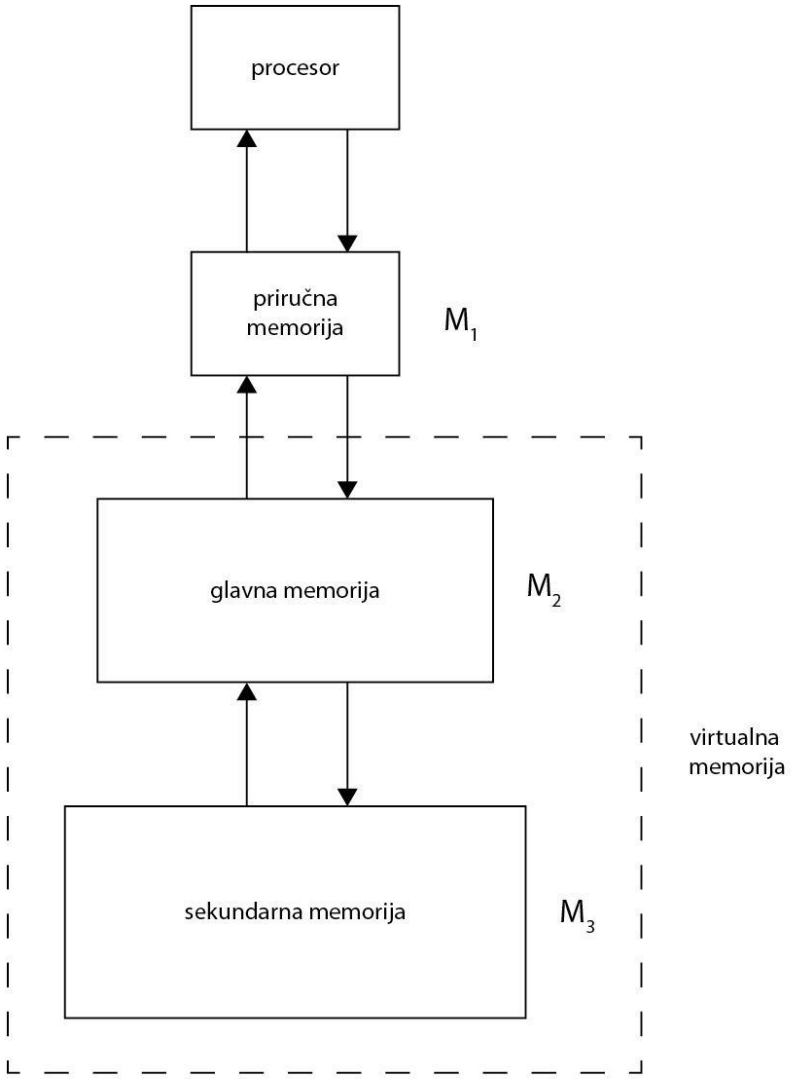
Simbolički prikaz memorijske hijerarhije u računarskom sustavu



Kapacitet: C
Cijena: P
Vrijeme pristupa: t

$C_1 < C_2$
$P_1 > P_2$
$t_1 < t_2$

$C_2 < C_3$
$P_2 > P_3$
$t_2 < t_3$

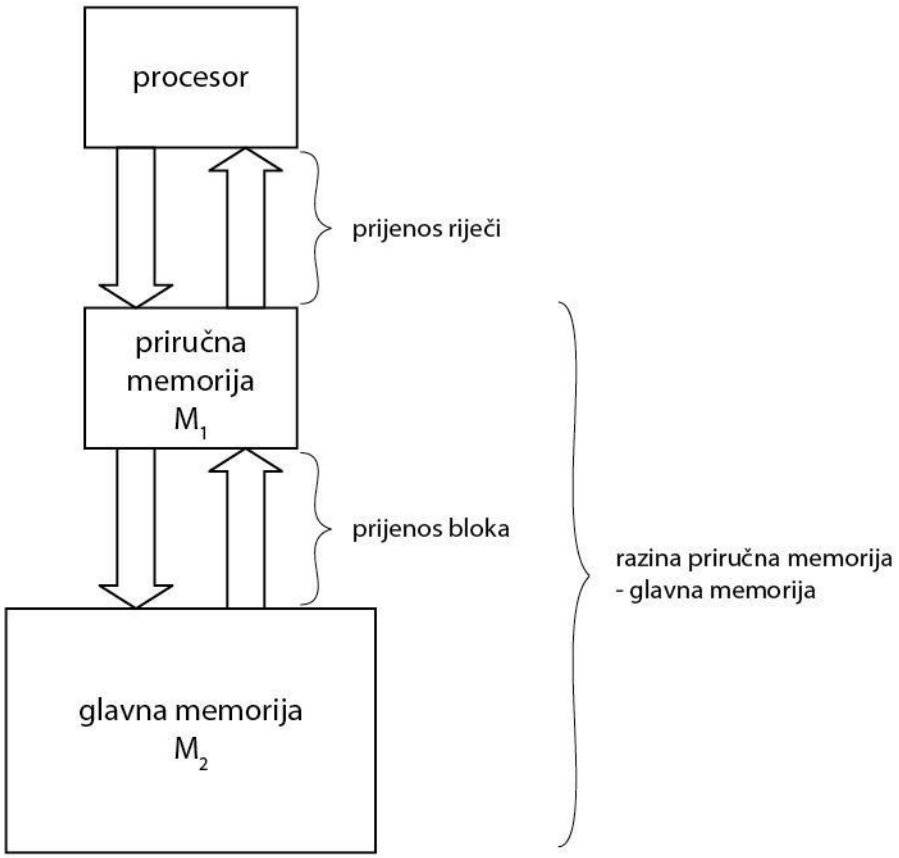


Priručna (engl. cache) memorija

(Građa računala, Arhitektura i organizacija računarskih sustava, str. 311 – 336)

Priručna memorija – mala i brza memorija koja sadrži kopiju sadržaja dijela glavne memorije – tekuće aktivne segmente programa i podataka

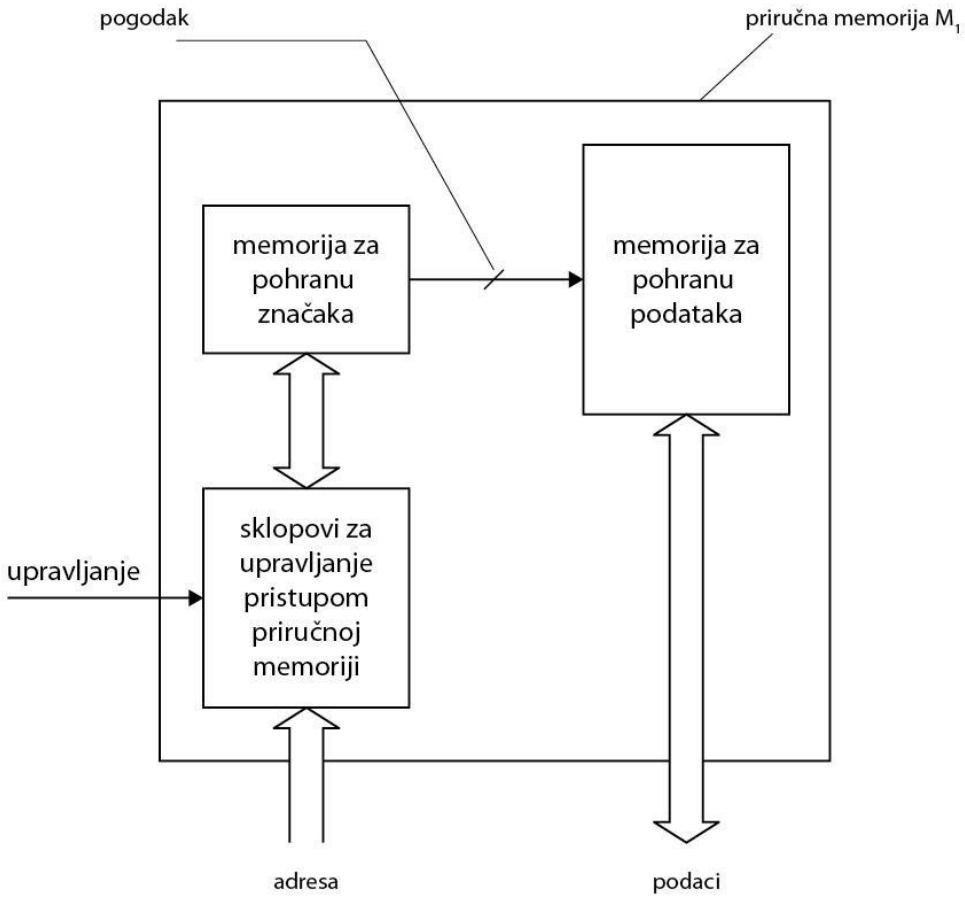
- između procesora i M_1 prenose se podaci u obliku riječi
- između M_2 i M_1 prenosi se blok podataka (priručni blok, linija)



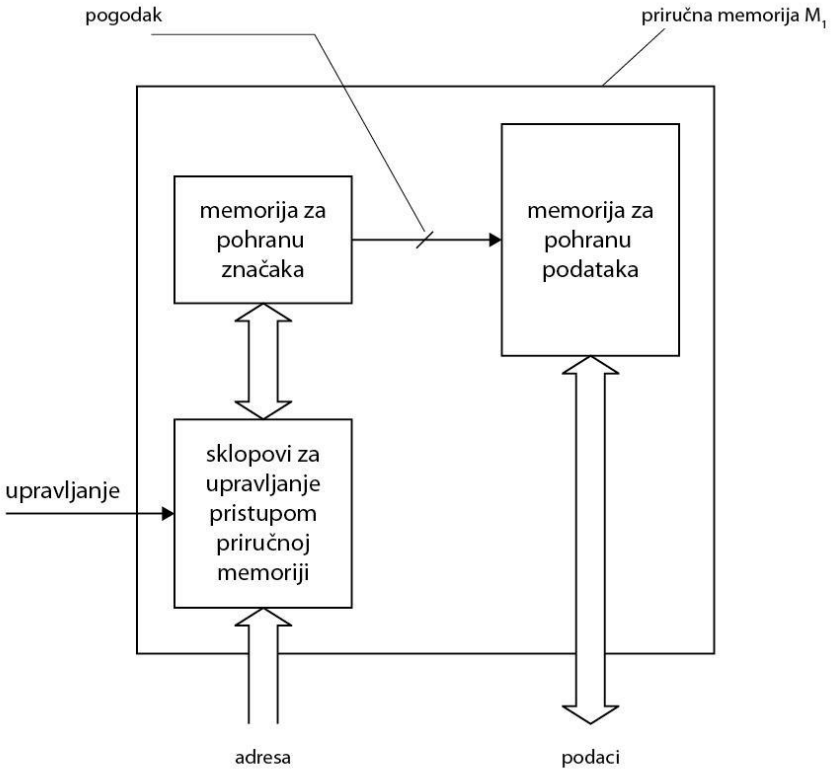
Interna organizacija priručne memorije

Opaska: ovdje se podaci podrazumijevaju u širem smislu – instrukcije i podaci

Kopija sadržaja dijela glavne memorije pohranjuje se u memoriji za pohranu podataka koja je organizirana u male blokove – priručni blokovi ili linije



• Svaki je priručni blok označen svojom bločnom adresom koja se naziva značka (ili oznaka) tako da priručna memorija "zna" kojem dijelu glavne memorije odgovara dotični blok
• Značke su pohranjene u memoriji za pohranu značaka
• Glavna memorija je također podijeljena na blokove koji su jednake veličine kao i priručni blokovi



Organizacija:

• Glavna memorija koja se sastoji od 2^M slijednih riječi, razdijeljena je na slijedne blokove koji se sastoje od b riječi

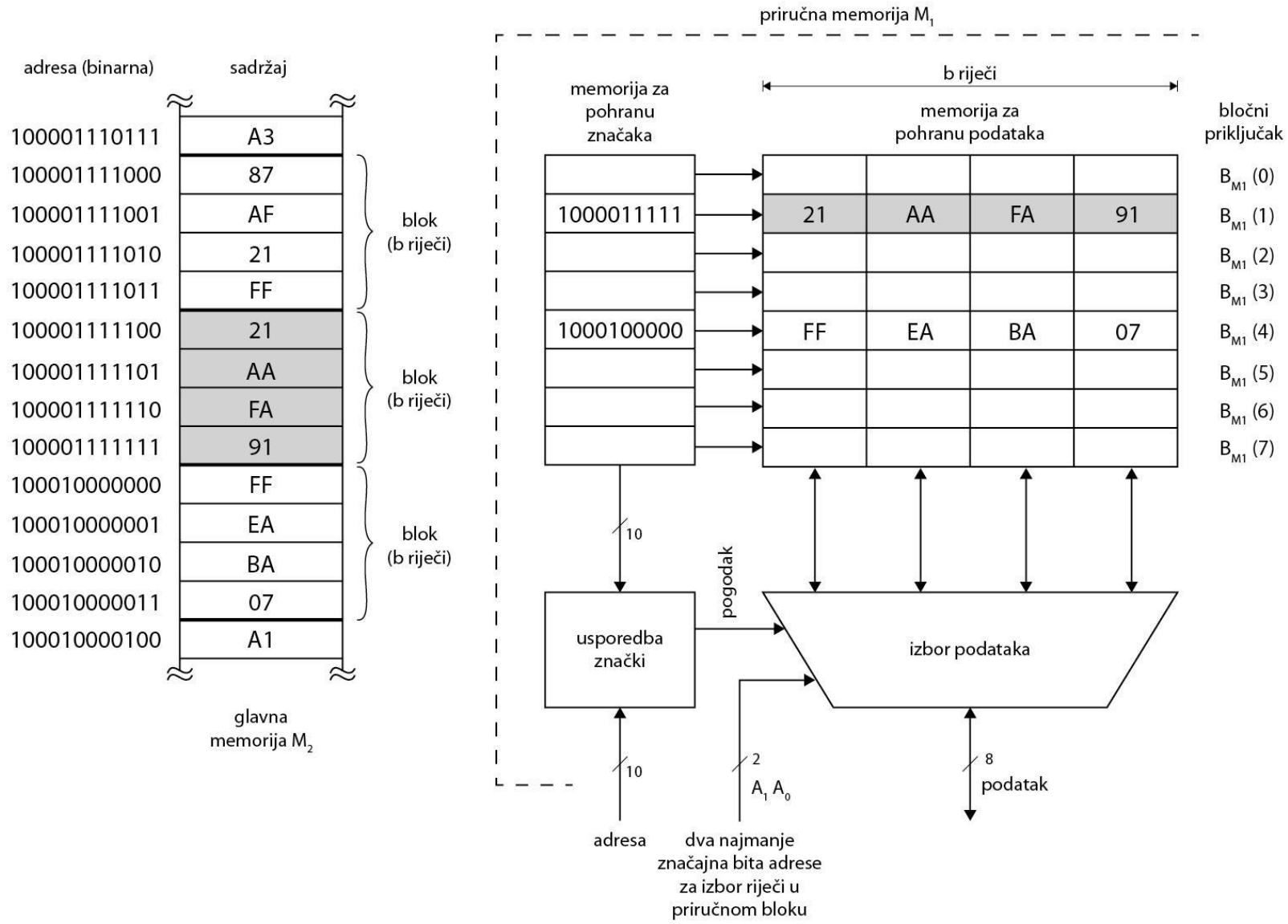
- glavna memorija ima $B_{M2} = 2^M / b$ blokova; b je također potencija broja 2; $b = 2^m$, pri čem je $m << M$

Memorija za pohranu podataka (u priručnoj memoriji!) sastoji se od B_{M1} priručnih blokova ili linija koji su također veličine $b = 2^m$

- svakom je priručnom bloku pridružena značka (pohranjena u memoriji za pohranu značaka) koja sadržava informaciju o adresi bloka pokazujući na početnu lokaciju odgovarajućeg bloka u glavnoj memoriji

Vrijedi $B_{M2} >> B_{M1}$

Priručni blok i značka smještaju se u priručnoj memoriji u tzv. bločni priključak

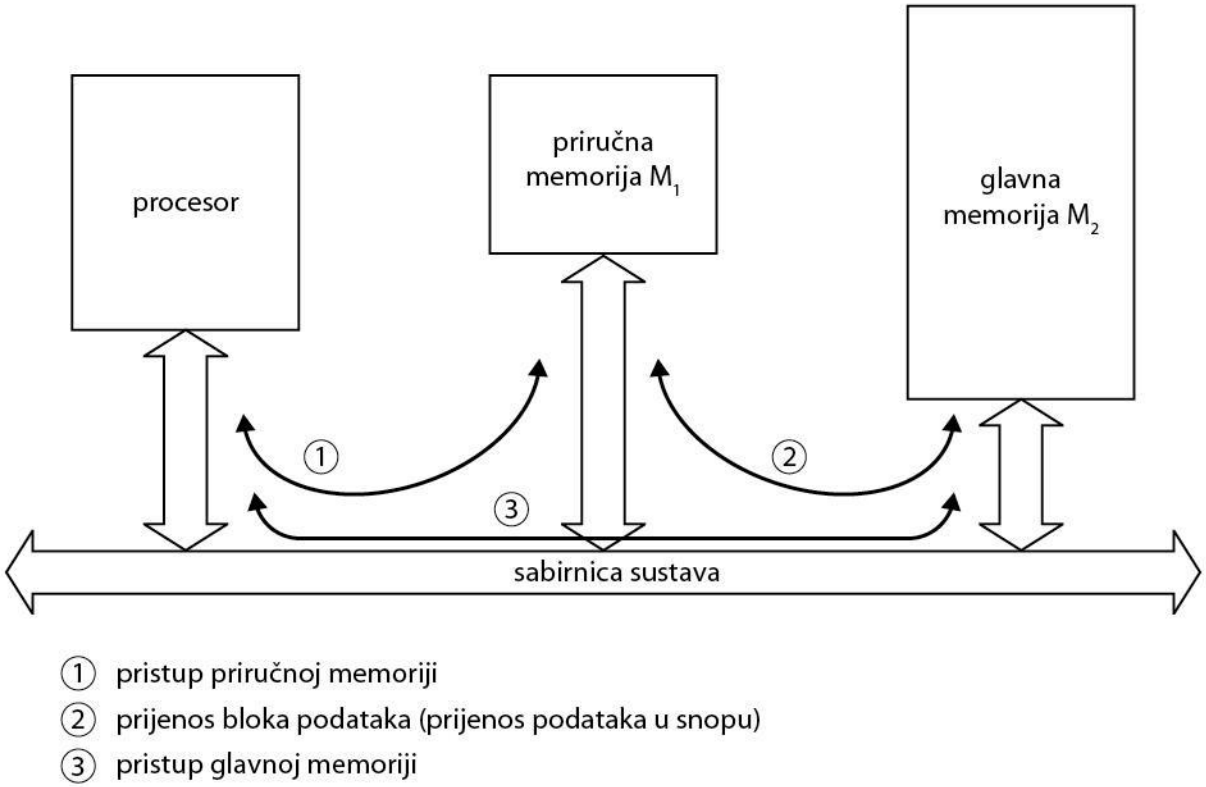

adresa (binarna)
sadržaj
100001110111
A3
100001111000
87
100001111001
AF
100001111010
21
100001111011
FF
100001111100
21
100001111101
AA
100001111110
FA
100001111111
91
100010000000
FF
100010000001
EA
100010000010
BA
100010000011
07
100010000100
A1
blok (b riječi)
blok (b riječi)
blok (b riječi)
glavna memorija M2
priručna memorija M1
b riječi
memorija za pohranu značaka
memorija za pohranu podataka
bločni priključak
1000011111
21
AA
FA
91
1000100000
FF
EA
BA
07
BM1 (0)
BM1 (1)
BM1 (2)
BM1 (3)
BM1 (4)
BM1 (5)
BM1 (6)
BM1 (7)
10
pogodak
usporedba znački
izbor podataka
10
2
A1 A0
8
podatak
adresa
dva najmanje značajna bita adrese za izbor riječi u priručnom bloku

Prema načinu uključivanja priručne memorije razlikujemo dvije osnovne organizacije:

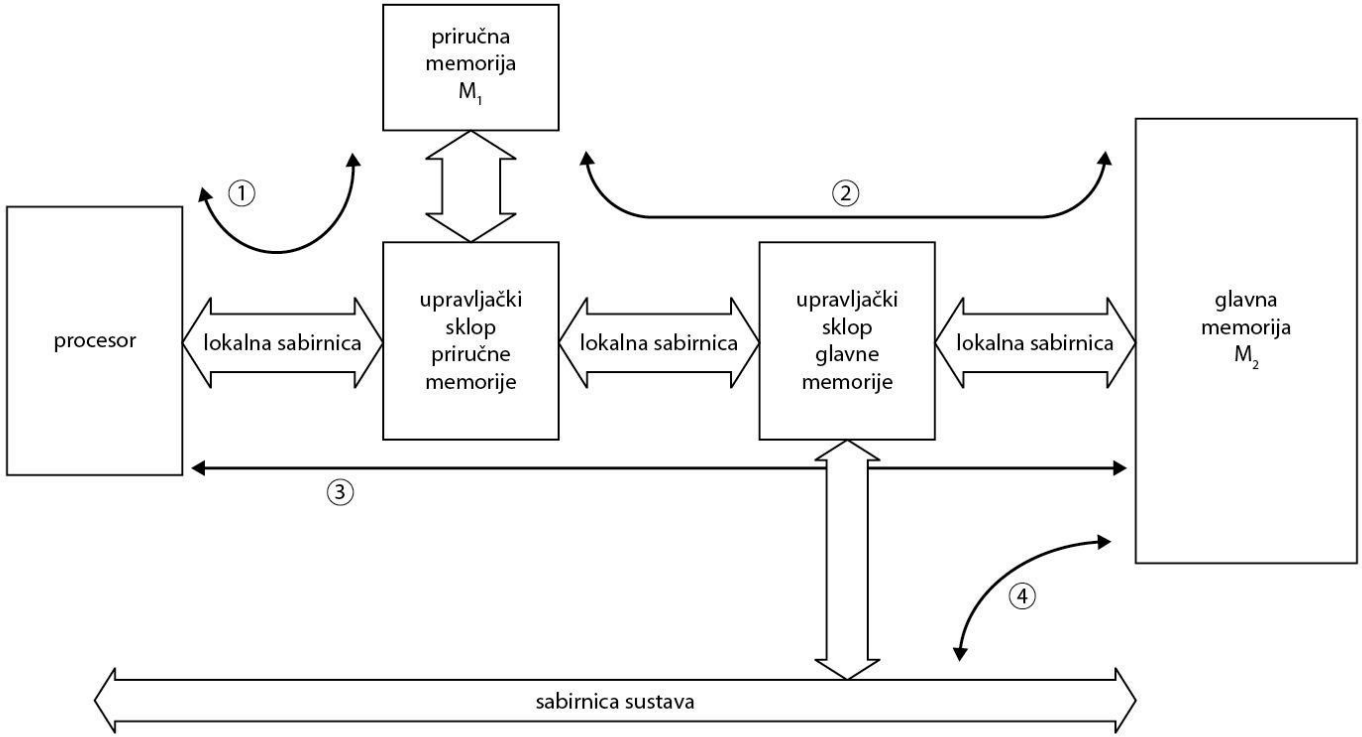
a) “pogled sa strane” (engl. look-aside)

b) “pogled kroz” (engl. look-through)

a)



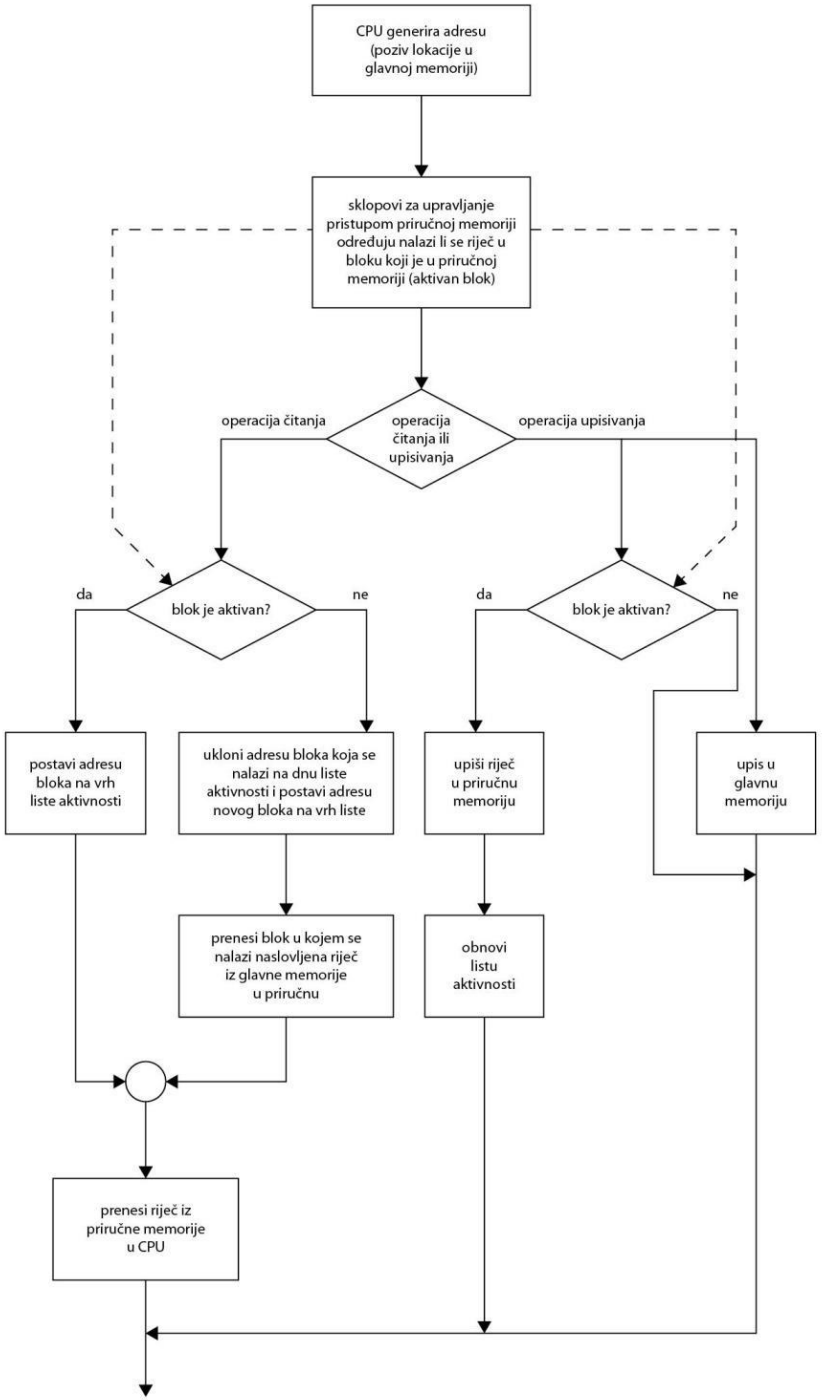
b)



① pristup priručnoj memoriji
② prijenos bloka podataka
③ pristup glavnoj memoriji
④ pristup glavnoj memoriji u kojem ne sudjeluje procesor (npr. DMA)

Djelovanje priručne memorije

- utvrđuje se nalazi li se riječ koju referencira procesor u priručnoj memoriji, tj. utvrđuje se nalazili se ona u nekom bloku u bločnom priključku priručne memorije (uspoređuju se značke sa značajnijim bitovima adrese – ako se tijekom uspoređivanja dogodi podudaranje – pogodak! /blok koji sadržava referenciranu riječ nalazi se u priručnoj memoriji – aktivan blok/)



Dogodio se pogodak!

- Operacija čitanja – referencirana se riječ dohvaća iz priručne memorije i pribavlja procesoru /glavna memorija ne sudjeluje u toj operaciji/

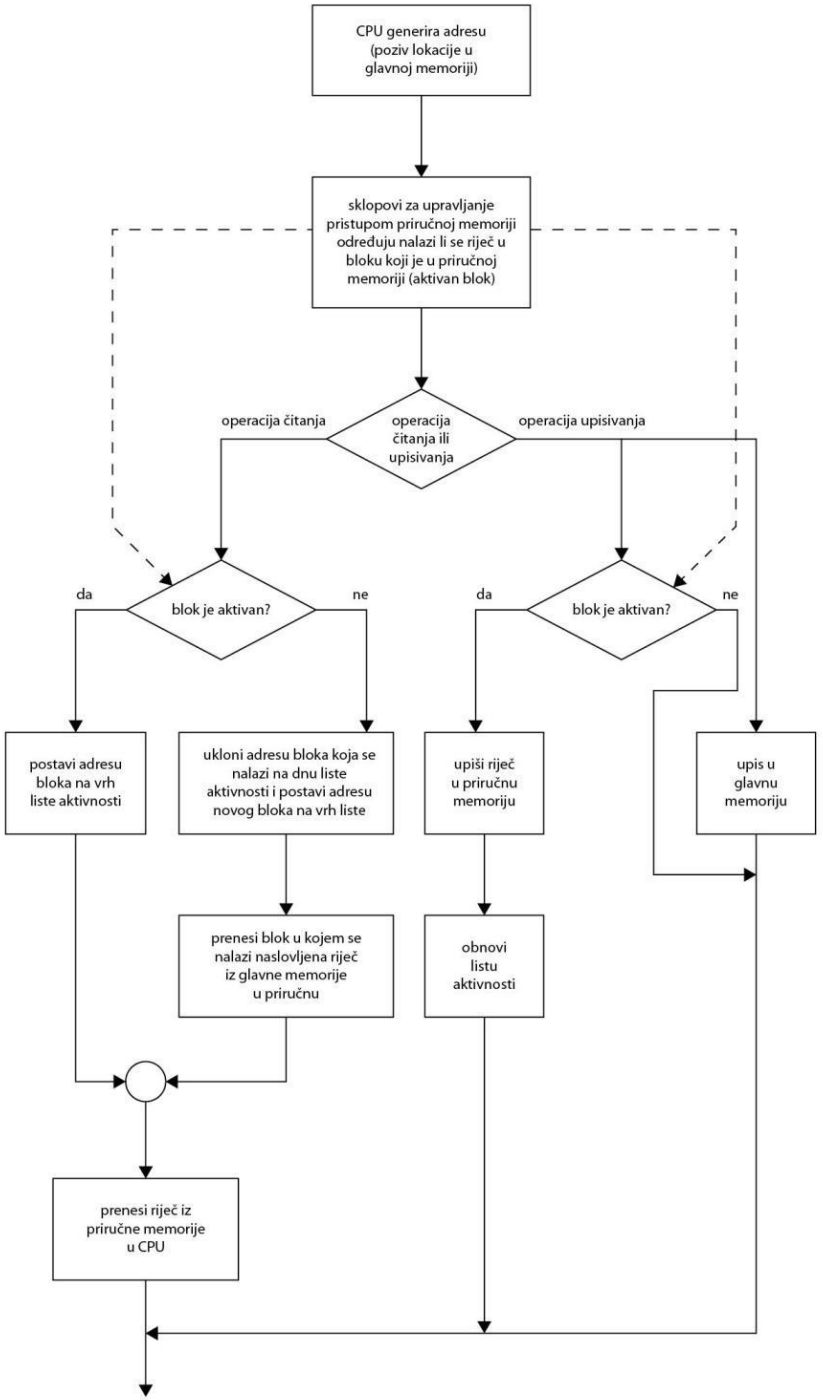
- Operacija pisanja – istodobno se podatak upisuje u odgovarajuću lokaciju priručne memorije ali i u lokaciju glavne memorije. Takav se postupak obnavljanja glavne memorije naziva “pohranjivanje-kroz” (engl. store-through)
/inačica upisivanje u međuspremnik (engl. write buffer)/

Druga metoda obnavljanja sadržaja glavne memorije naziva se “kopiranje nazad” (engl. write back, copy back) i sastoji se od toga da se podatak upisuje samo u lokaciju u priručnoj memoriji, a da se upis označi posebnom zastavicom pridruženom bloku. Ta se zastavica naziva “prljavi bit” (engl. dirty bit) i njome se označava blok koji će se morati upisati natrag u glavnu memoriju tijekom zamjene blokova.

Dogodio se promašaj!

Uspoređivanje adrese sa značkama bilo je neuspješno!

- Operacija čitanja – adresa se prosljeđuje glavnoj memoriji. Modul za rukovanje u slučaju promašaja na temelju informacije o zamjeni blokova, određuje koji se blok zamjenjuje u priručnoj memoriji blokom koji treba dohvatiti iz glavne memorije. Nakon ili tijekom prijenosa bloka, referencirana se riječ šalje procesoru.

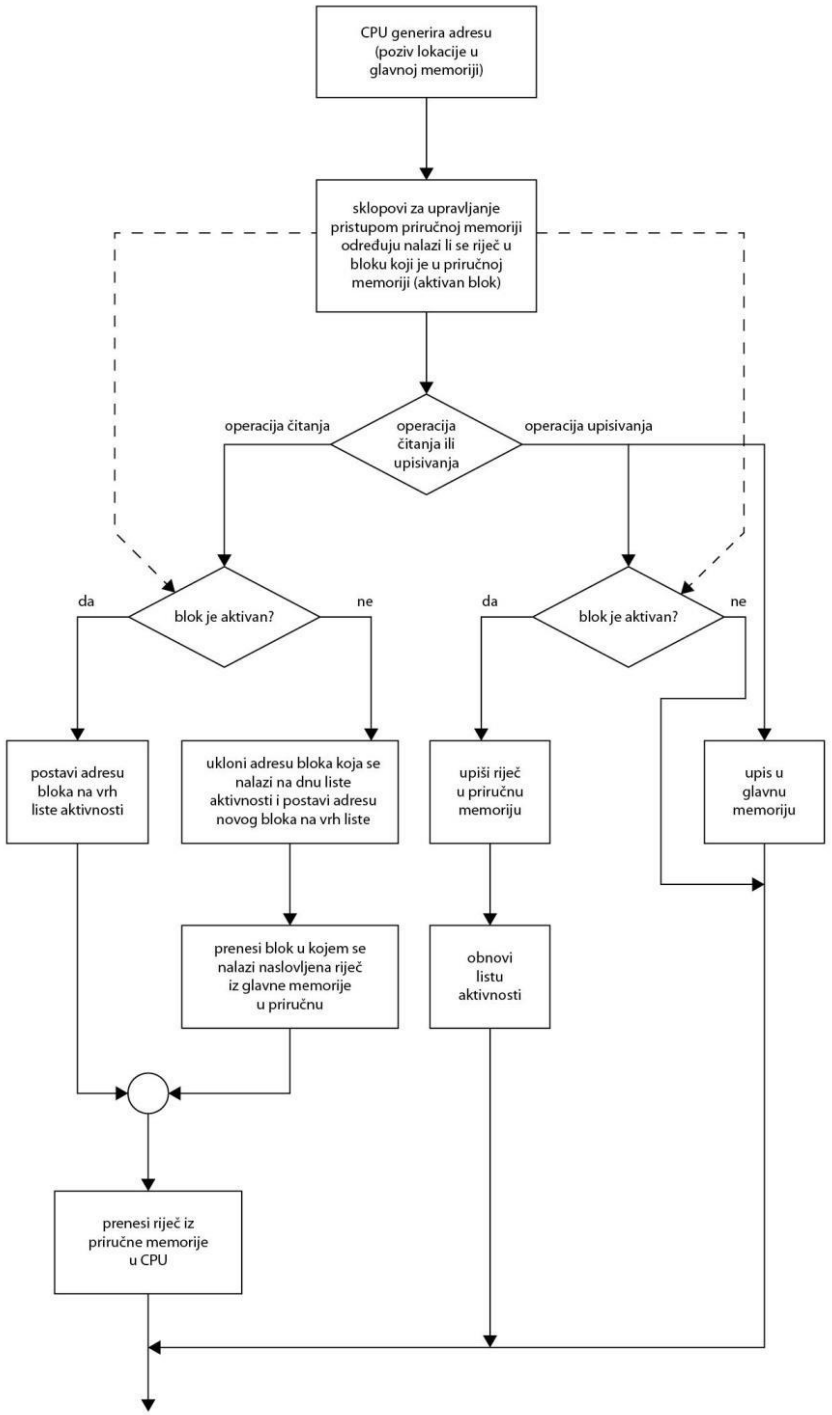


Dogodio se promašaj!

- Operacija pisanja – dva pristupa:

1) podatak se izravno upisuje u glavnu memoriju – sadržaj priručne memorije se ne mijenja;
/tehnika *write-no alloction/*

2) odgađa se upis u memoriju da bi se prenio blok u priručnu memoriju i tek onda obavi upis u priručnu memoriju
/tehnika *write allocation/*



Performansa priručne memorije

Omjer pogotka H (engl. hit ratio)

H = (broj referenciranja u kojima je postignut pogodak) /
(ukupan broj referenciranja)

Omjer promašaja MR = 1 - H (engl. miss ratio)

U nekom vremenskom intervalu možemo ocijeniti srednje vrijeme pristupa memorijskom sustavu t_A:

$$t_A = t_{M1} \times H + (1-H) \times t_{M2,}$$

gdje je t_{M1} vrijeme pristupa priručnoj memoriji,

t_{M2} vrijeme pristupa glavnoj memoriji

Primjer:

t_{M2} = 250 ns

t_{M1} = 25 ns

Omjer pogotka H neka je 0.98, što je vrijednost koja se nalazi u okviru tipičnih vrijednosti za priručne memorije

$$t_A = t_{M1} \times H + (1-H) \times t_{M2} = 25 \times 0.98 + (1- 0.98) \times 250$$

$$t_A = 24.5 + 5.0 = 29.5 \text{ ns}$$

uvođenjem priručne memorije srednje vrijeme pristupa memoriji popravilo se skoro za jedan razred veličine

Lokalnost

Visoka vrijednost omjera pogotka H (tipično 0.9 – 0.98) temelji se na lokalnim svojstvima programa i podataka (sljed memorijskih lokacija kojima programi pristupaju nije slučajan; pokazuje težnju grupiranju pristupa malom dijelu ukupnog adresirljivog memorijskog prostora)

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 vremenskom trenutku *t* referencirani u posljednjih *h* pozivanja:

$$WS(t, h) = \{i \in N \mid i \in r_{t-h}, r_{t-h+1}, \ldots, r_t\}$$

skup blokova N = {1, 2, ... , n} koji čine neki program

R – slijed naslovljavanja stranica:

$$R = (r_1, r_2, ..., r_k, ...)$$

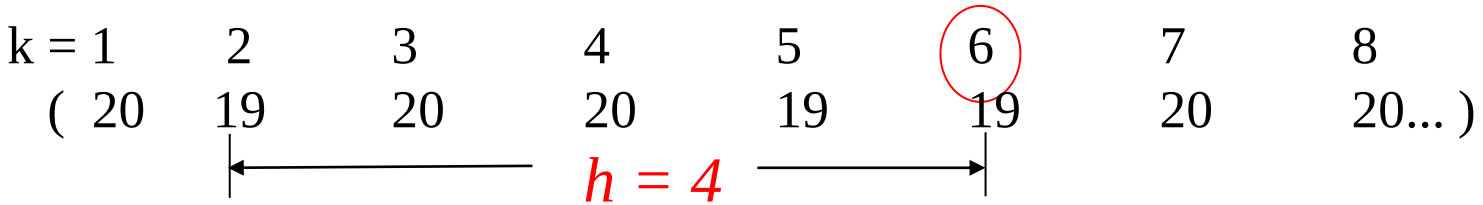
WS(*k*, *h*) – skup blokova koji se javljaju u "oknu" naslovljavanja veličine *h* (gledajući unatrag u slijedu naslovljavanja stranica).

Primjer:

Neka je slijed naslovljavanja blokova:

R = (20, 19, 20, 20, 19, 19, 20, 20, 21, 4, 5, 6, 20)

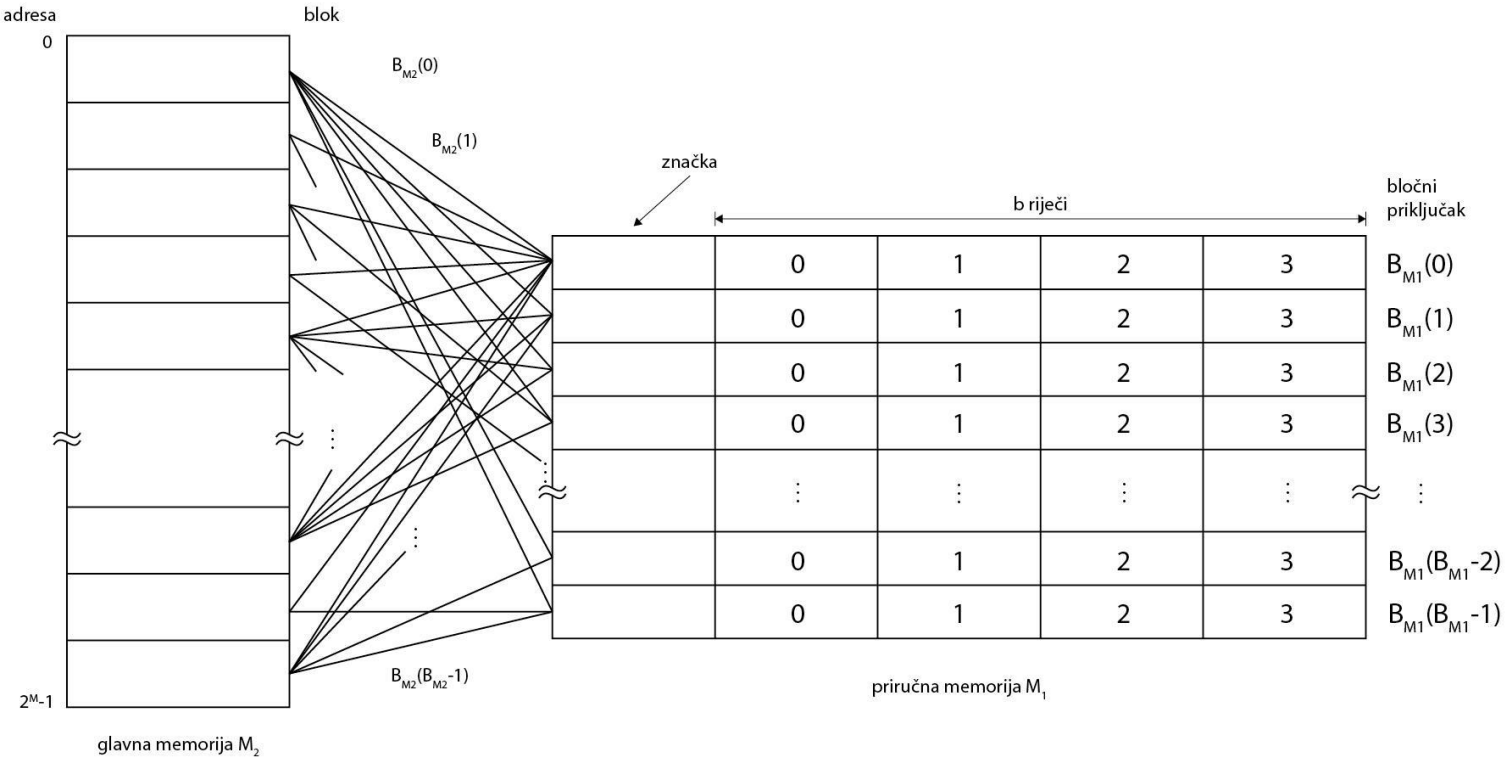
WS(*6*, *4*) = { 19, 20}



Organizacija priručne memorije

Način smještanja blokova u bločne priključke

- priručna memorija s potpuno asocijativnim preslikavanjem (engl. fully associative mapping)



- Priručna memorija s izravnim preslikavanjem

Pravilo: Blok $B_{M2}(i)$ iz glavne memorije smješta se u bločni priključak $B_{M1}(j)$ priručne memorije pri čemu vrijedi

$$j = i \text{ (modulo } B_{M1})$$

Primjer:

- kapacitet priručne memorije 64 KB, veličina priručnog bloka 4 bajta;

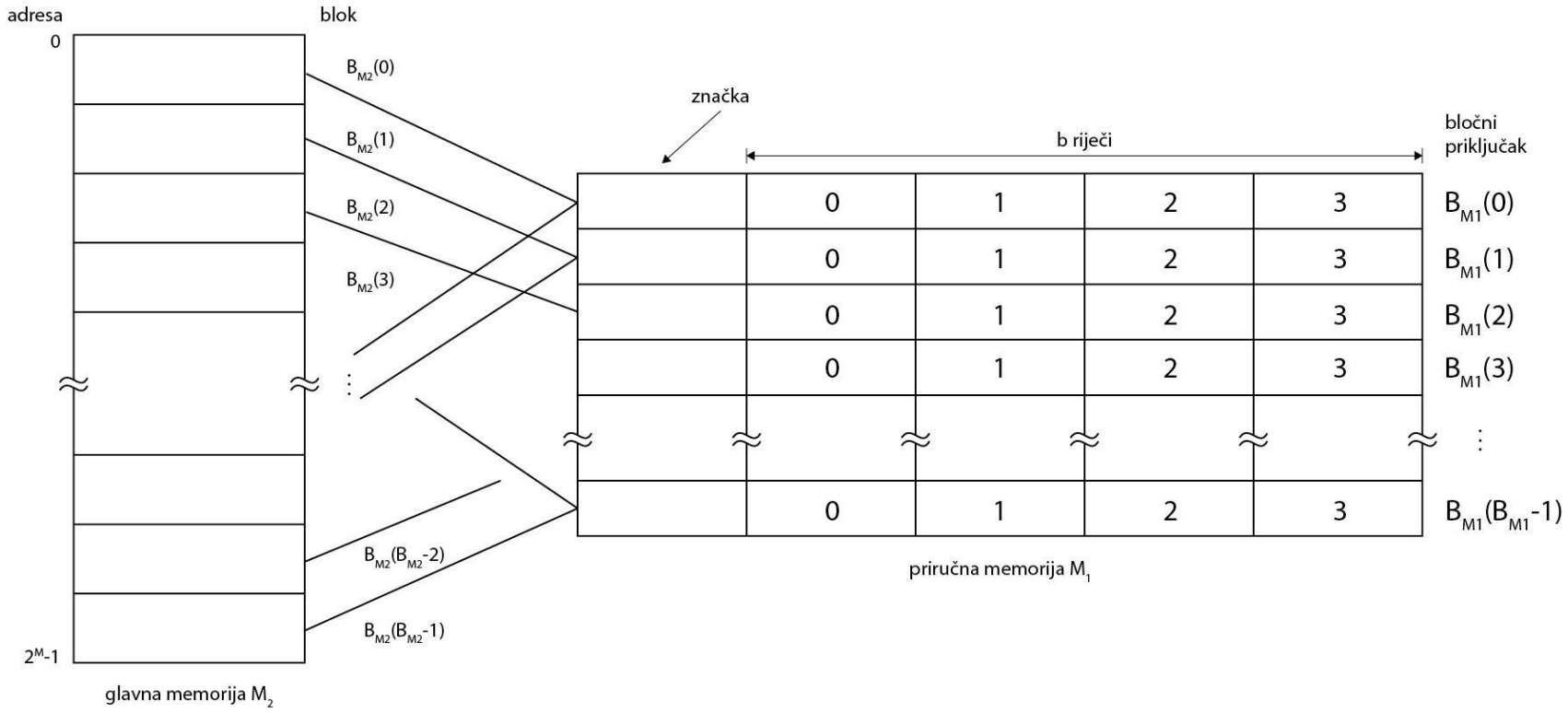
$$B_{M1} = 64 \text{ K} / 4 = 16 \text{ K} = 16384$$

- kapacitet glavne memorije 16 MB

$$B_{M2} = 16 \text{ M} / 4 = 4 \text{ M} = 4\ 194304$$

Blok $B_{M2}(32780)$ preslikat će se u $B_{M1}(j)$ pri čemu je j:

$$j = 32780 \text{ (modulo } 16384) = 12$$



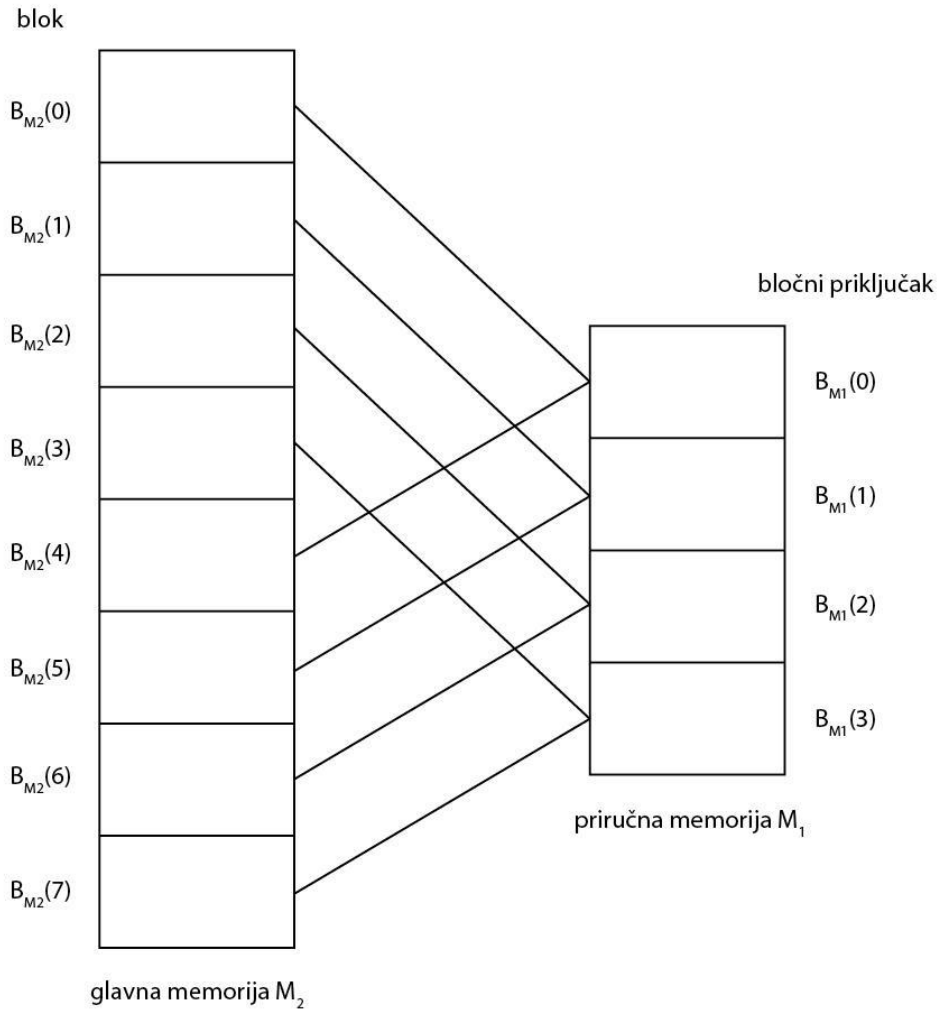
Priručna memorija s izravnim preslikavanjem

$B_{M1} = 4$
$B_{M2} = 8$

$B_{M1}(j)$; j = 0, 1, 2, 3
$B_{M2}(i)$; i = 0,1, ... , 7

Npr. $B_{M2}(6)$ se preslikava u
j = i (modulo B_{M1})
j = 6 (modulo 4) = 2

$B_{M2}(7) \rightarrow B_{M1}(3)$



Priručna memorija sa skupnim asocijativnim preslikavanjem (engl. set-associative mapping)

- kombinacija prethodno opisanih dvaju načina smještanja blokova

bločni priključci priručne memorije organizirani su u skupine tako da mehanizam preslikavanja dopušta da se blok iz glavne memorije priključuje u bilo koji bločni priključak koji pripada odgovarajućoj skupini

- priručna memorija ima B_S skupina bločnih priključaka
- svaka skupina ima k bločnih priključaka (ukupno je B_S x k bločnih priključaka u priručnoj memoriji)

Blok iz glavne memorije $B_{M2}(i)$ može se priključiti u bilo koji od k slobodnih bločnih priključaka skupine j, pri čemu je

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

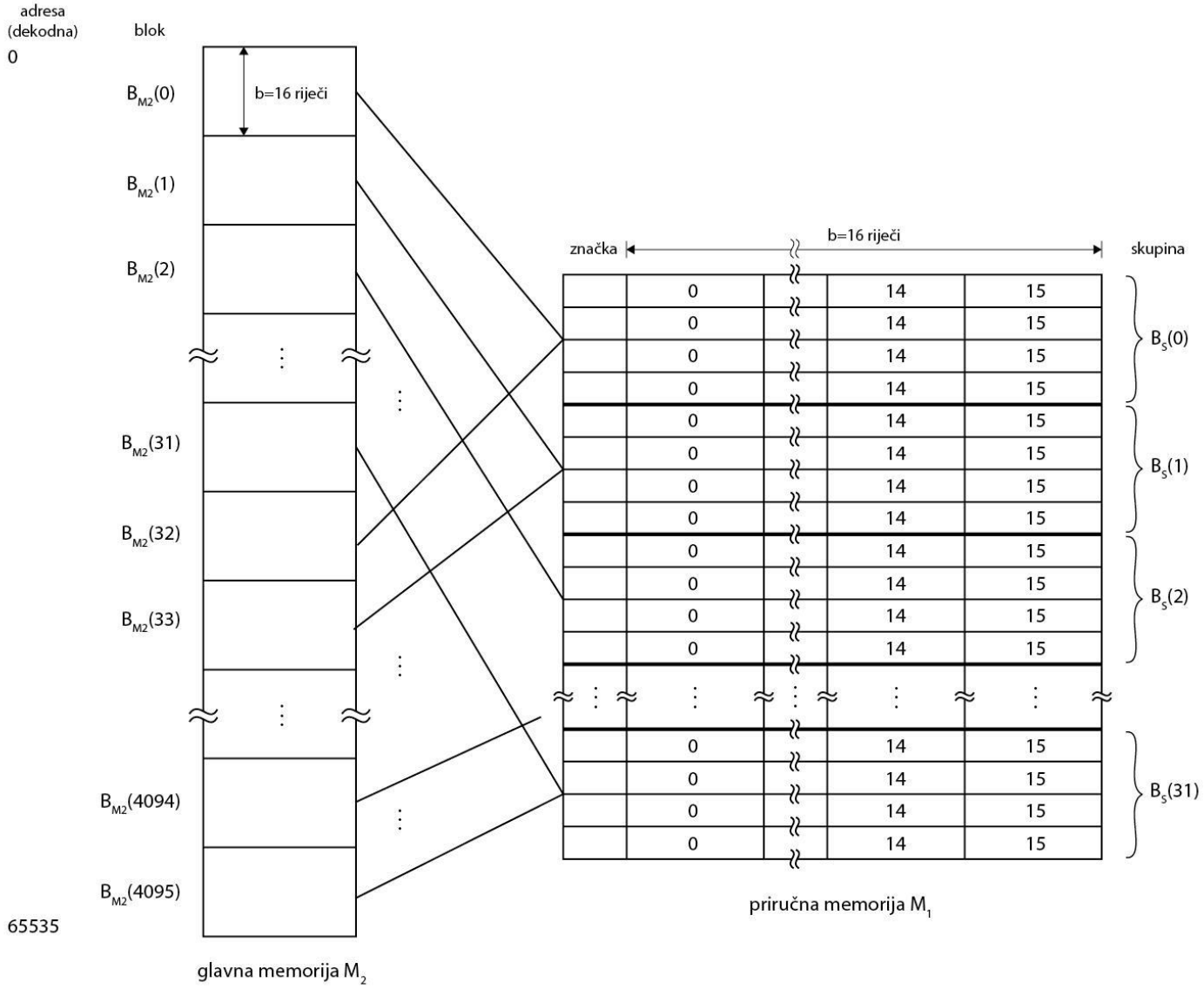
Pretpostavimo da imamo priručnu memoriju koja ima B_{M1} = 128 bločnih priključaka organiziranih u 32 skupine B_s = 32; B_s (0) - B_s (31). Svaki bločni priključak ima b = 16 riječi.

U svakoj skupini imamo B_{M1} / B_s = 4 bločna priključka ("četveroputna asocijativna memorija")

Glavna memorija neka je kapaciteta 64 K riječi i organizirana je u B_{M2} = 64 K / b = 64 K / 16 = 4096 blokova. Glavna memorija ima blokove B_{M2} (0) - B_{M2} (4095).

Primjer preslikavanja:
Neka se referencirana riječ nalazi u bloku B_{M2} (255); i = 255.
Blok B_{M2} (255) može se priključiti na jedan od četiri slobodna priključka skupine j:

$$j = i \text{ (modulo } B_s \text{)} = 255 \text{ (modulo 32)} = 31$$


adresa
(dekodna)
0
blok
b=16 riječi
$B_{M2}(0)$
$B_{M2}(1)$
$B_{M2}(2)$
$B_{M2}(31)$
$B_{M2}(32)$
$B_{M2}(33)$
$B_{M2}(4094)$
$B_{M2}(4095)$
65535
glavna memorija M_2
značka
b=16 riječi
skupina
0
14
15
$B_S(0)$
$B_S(1)$
$B_S(2)$
$B_S(31)$
priručna memorija M_1

Način zamjene blokova

- kad se novi blok iz glavne memorije treba prenijeti u priručnu memoriju i kad su svi bločni priključci zauzeti, potrebno je donijeti odluku o tome koji se od blokova u priručnoj memoriji mora zamijeniti

- u slučaju priručne memorije s izravnim preslikavanjem odluka je već donesena – j = i (modulo B_{M1})
- za priručne memorije s potpuno asocijativnim preslikavanjem i priručne memorije sa skupnim asocijativnim preslikavanjem odluka se donosi na temelju algoritma zamjene

Algoritmi zamjene:

• FIFO
• LRU (Least Recently Used) izabire se blok za zamjenu koji je najdulje u priručnoj memoriji najduže bez da bude referenciran
• NRU (Not Recently Used) – algoritmom se zamjenjuje blok koji nije u posljednje vrijeme bio referenciran
/”posljednje vrijeme” – ovisno o implementaciji/
• LFU (Least Frequently Used)
• Random

Itanium-2	L1	L2	L3
Kapacitet	16 KB I priručna memorija 16 KB D priručna memorija	256 KB I&D priručna memorija	3 MB I&D priručna memorija
Način smještanja blokova u bločne priključke	Skupna asocijativna (4-putna)	Skupna asocijativna (8-putna)	Skupna asocijativna (24-putna)
Veličina bloka (linije)	64 bajta	128 bajta	128 bajta
Algoritam zamjene blokova	LRU	NRU	NRU
Način obnavljanja sadržaja glavne memorije	Pohranjivanje-skroz	Kopiranje nazad	Kopiranje nazad
Detekcija/ispravljanje pogreške	ECC	ECC (SECDED)	ECC (SECDED)
Mjesto implementacije	Procesorski čip	Procesorski čip	Procesorski čip

Tablica 10.2. Osnovne značajke priručne memorije za Intel Itanium-2 (EPIC ISA)

11. VIRTUALNI MEMORIJSKI SUSTAV

Građa računala, Arhitektura i organizacija računarskih sustava, str. 337- 357

- 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 ili nekoliko G bajtova
- Sekundarna memorija – nekoliko desetaka ili stotina (i više) G bajtova ili nekoliko desetak T bajtova

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

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

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.

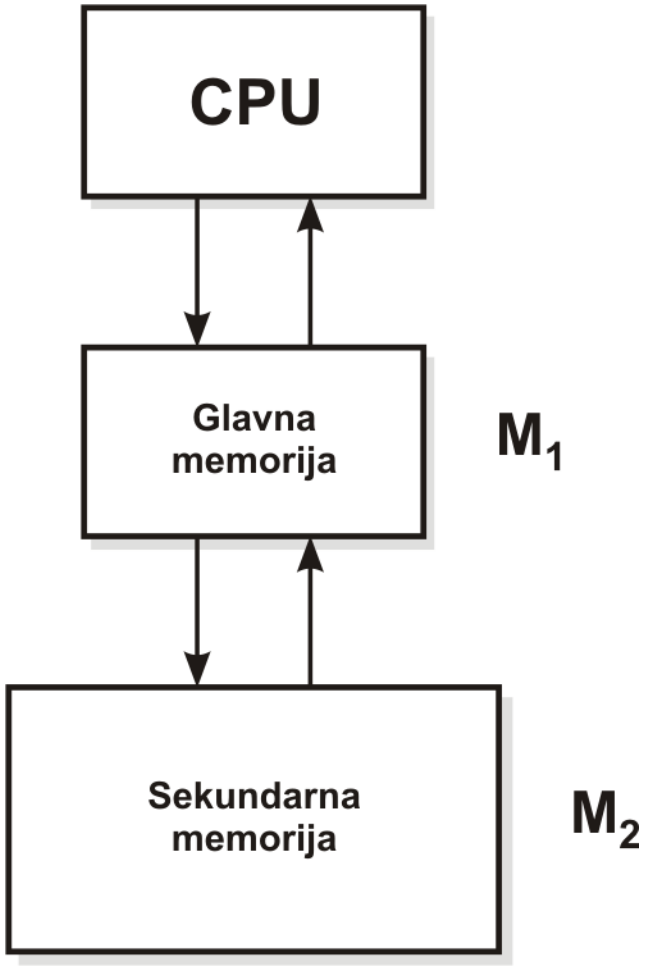
Memorijska hijerarhija:

(M_1, M_2, M_3, ... 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: $c_i > c_{i+1}$
$t_{ai} < t_{ai+1}$
$S_i < S_{i+1}$

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 >> FAP odrediti funkciju f:

$$f: LAP \rightarrow FAP$$

$f: LAP \rightarrow FAP$

$LAP = \{0, 1, 2, ..., N-1\}$
$FAP = \{0, 1, 2, ..., M-1\}$

vrijedi: $N >> M$

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

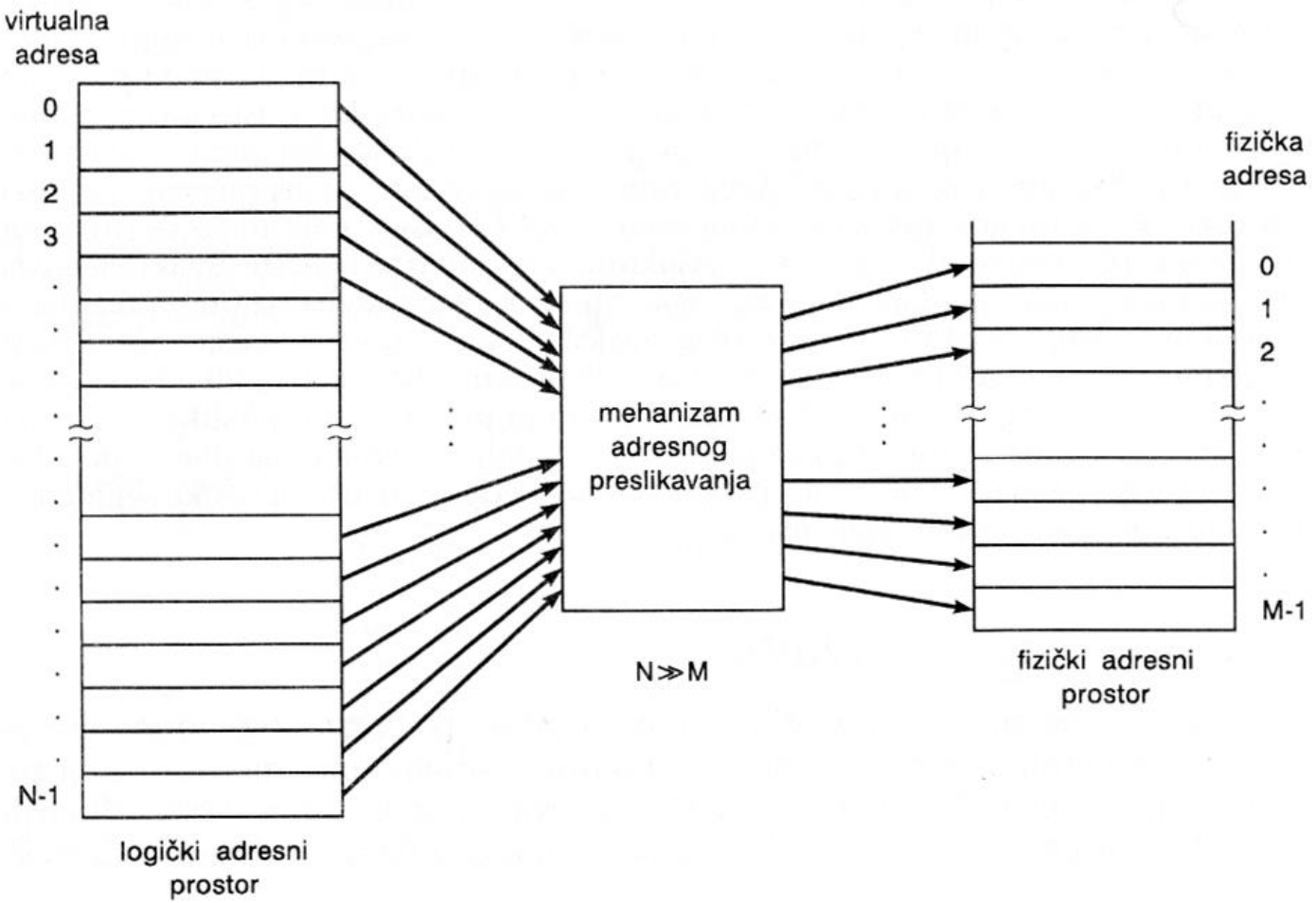
Neka je $a \in$ LAP

Funkcija f je definirana kao:

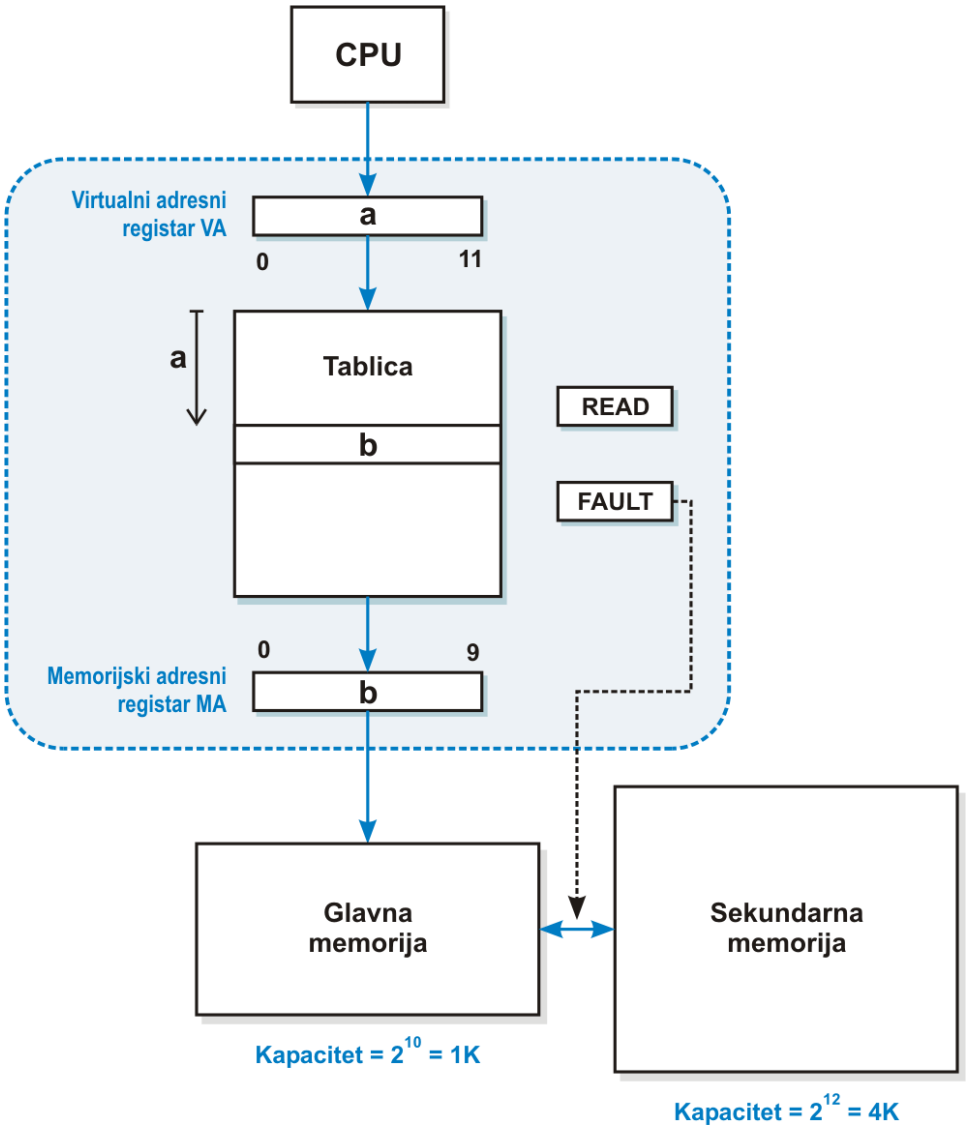
f(a) = a' ako se podatak s virtualnom adresom a nalazi na adresi a' u fizičkoj memoriji ($a' \in$ FAP)

f(a) = ϕ označava PROMAŠAJ (engl. Missing-item fault)

Adresno preslikavanje:

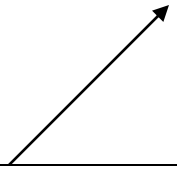


Denningov model



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!

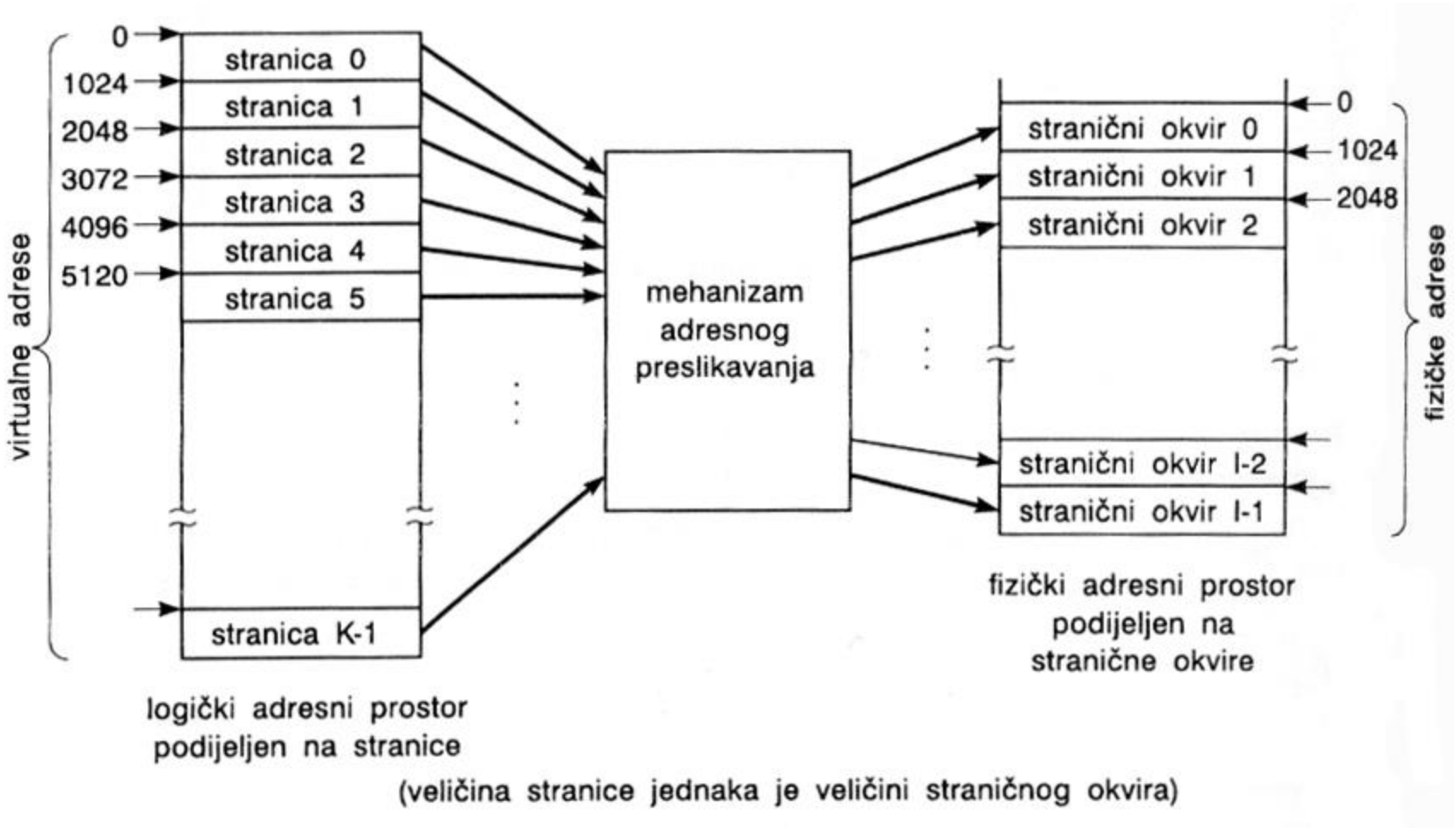
Rješenje:

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!

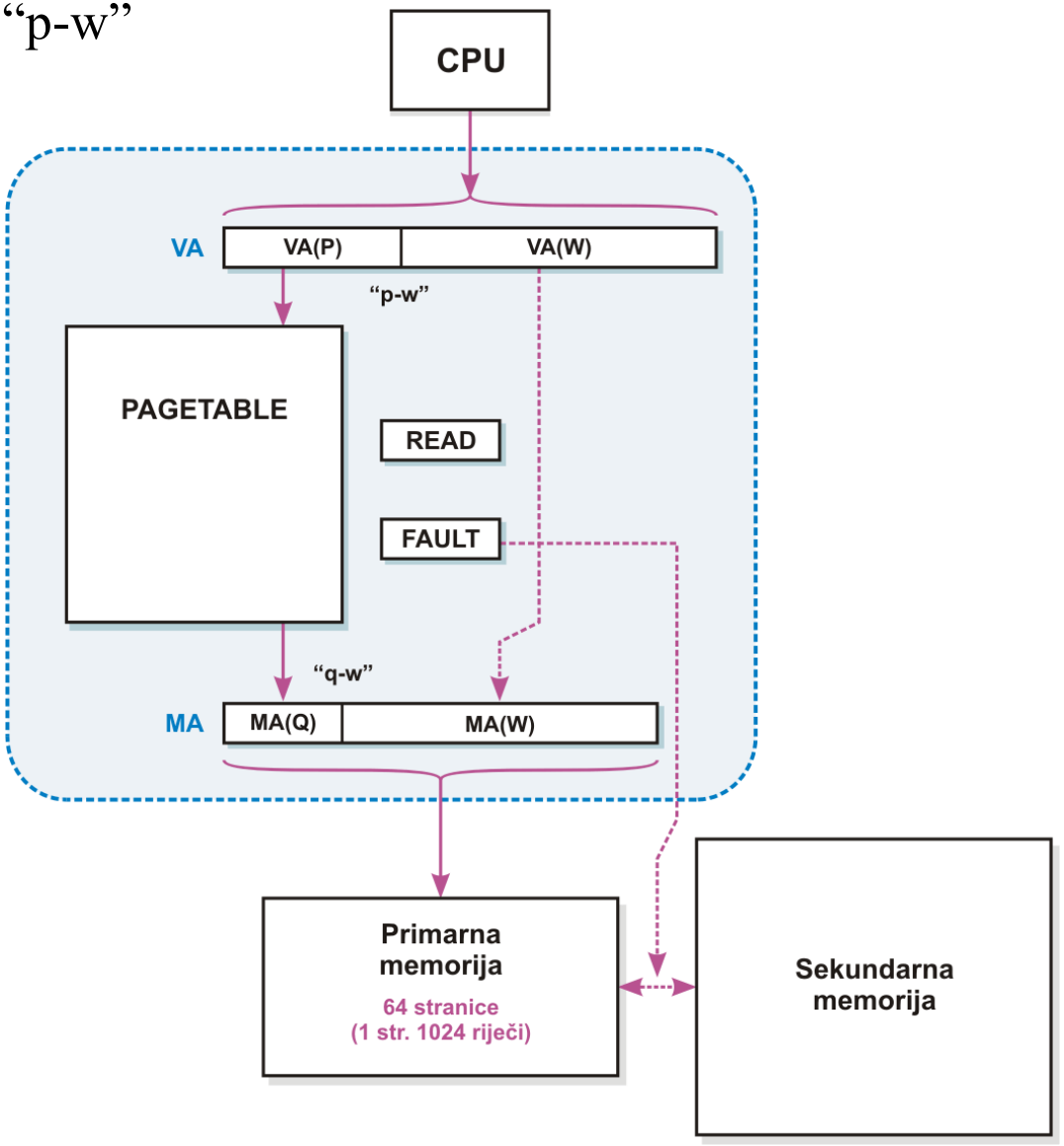
Blokovi = stranice /ako su čvrste duljine/
Blokovi = segmenti /ako su promjenjive duljine/

Straničenje (engl. Paging)

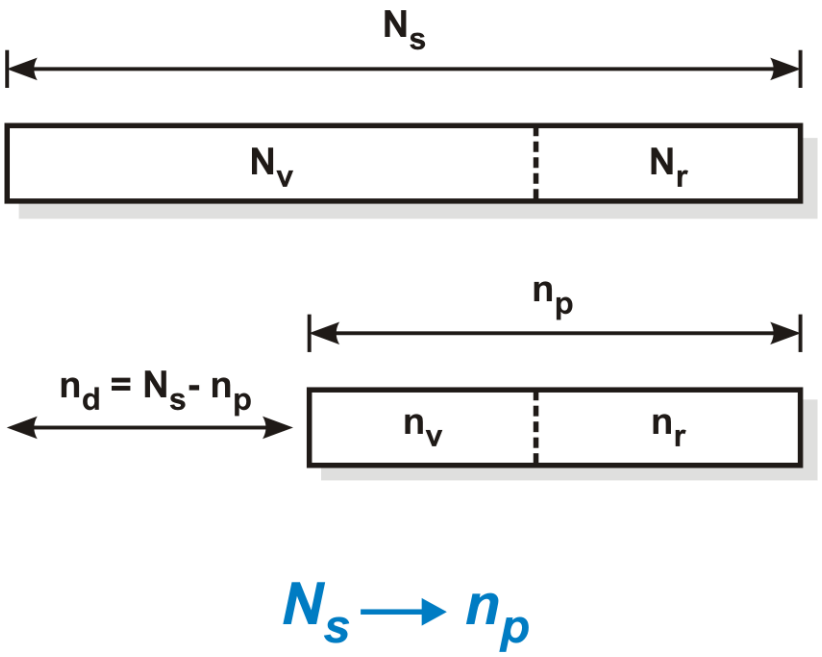


Translacija zadane virtuane adrese “p-w” u fizičku adresu “q-w”:

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)

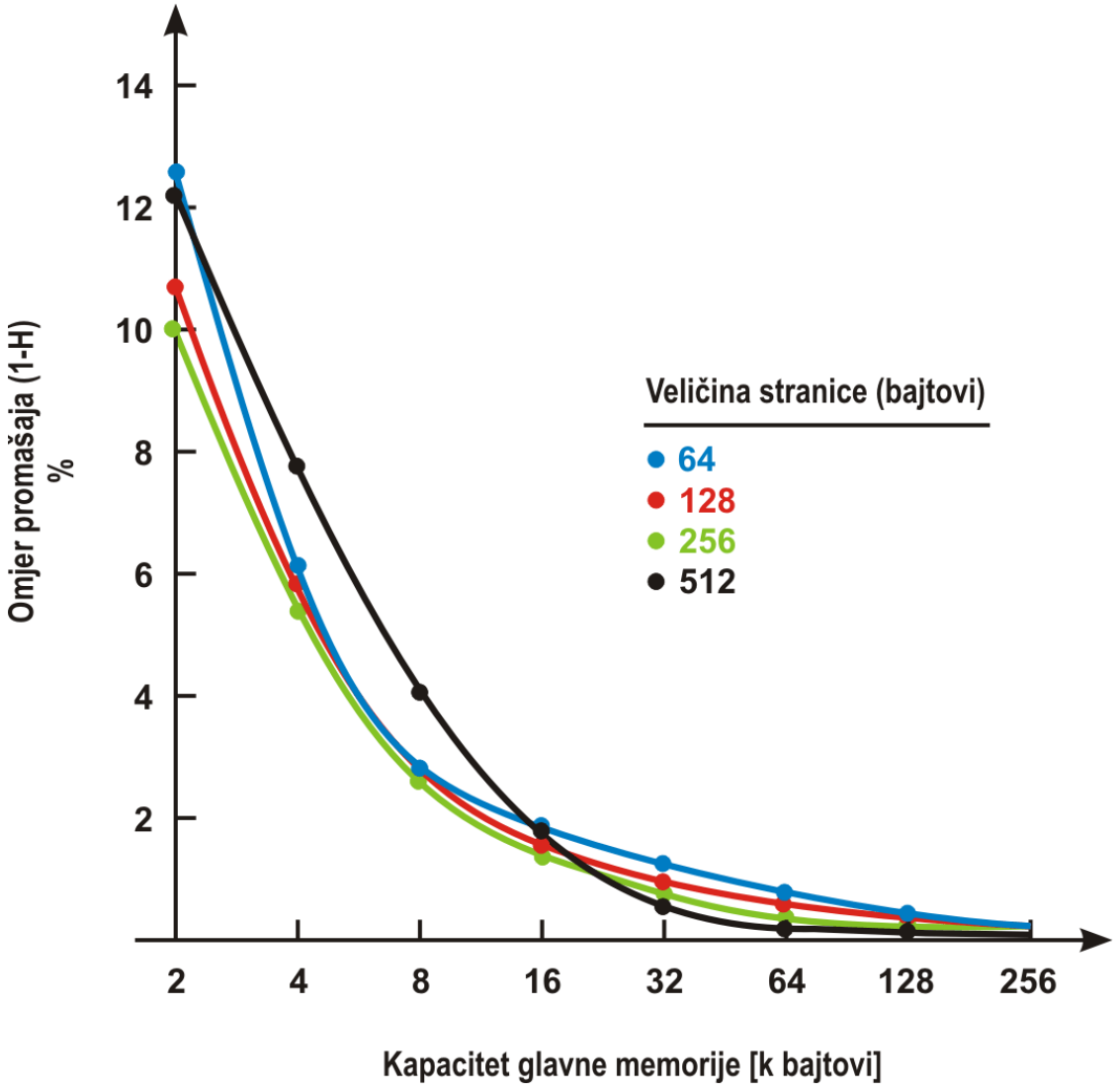
$$Miss\_ratio = 1 - H$$

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

Primjer:
Način smještanja stranica:
- potpuno asocijativno preslikavanje

Način zamjene stranica:
- LRU algoritam zamjene stranica



Algoritmi zamjene stranica

Situacija:

- nova se stranica iz sekundarne memorije treba prenijeti u glavnu memoriju a u njoj nema slobodnog straničnog priključka

Odluka:

- koja će se stranica iz glavne memorije odstraniti i na taj način osloboditi stranični priključak

Algoritmi zamjene:

- Slučajni izbor
- FIFO
- LRU (engl. Least Recently Used)
- OPT! (optimalna strategija zamjene stranica zahtijeva potpuno znanje o budućim referenciranjima stranica)

Primjer višekorisničkog virtualnog sustava