ELEMENTI PROJEKTOVANJA KES MEMORIJE

- Pregled parametara za projetovanje kes memorije
- Racunari visokih performansi(HPC)
- HPC-superacunari, primena za naucne aplikacije koje obuhvataju velike kolicine podataka, proracuna, vektora, matrica I upotrebu paralelnih algoritama
- Projektovanje kes-a za HPC je razlicito od projektovanja za druge hardverske platforme

ELEMENTI PROJEKTOVANJA KES MEMORIJE

- Neka istrazivanja su pokazala da HPC aplikacije slabo rade na racunarskim arhitekturama koje koriste kes memorije
- Druga istrazivanja su pokazala da hijerarhija kesa moze da bude korisna u poboljsanju performanse ako se aplikacioni softver podesi da koristi kes

ELEMENTI PROJEKTOVANJA KES MEMORIJE

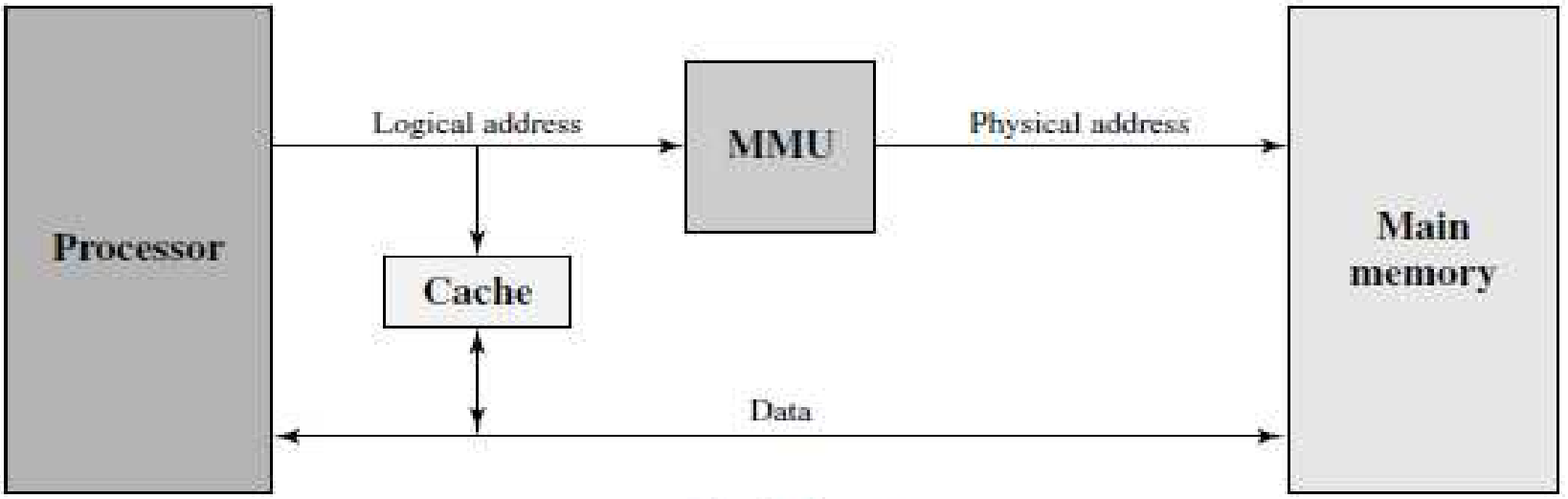
•Kljucni elementi za projektovanje koji sluze za klasifikaciju I medjusobno razlikovanje arhitekture kesa:
•Adresa kes memorije
•Velicina kesa
•Funkcija preslikavanja
•Algoritmi zamene
•Politika upisivanja
•Velicina reda
•Broj kes memorije

•Adresa kes memorije

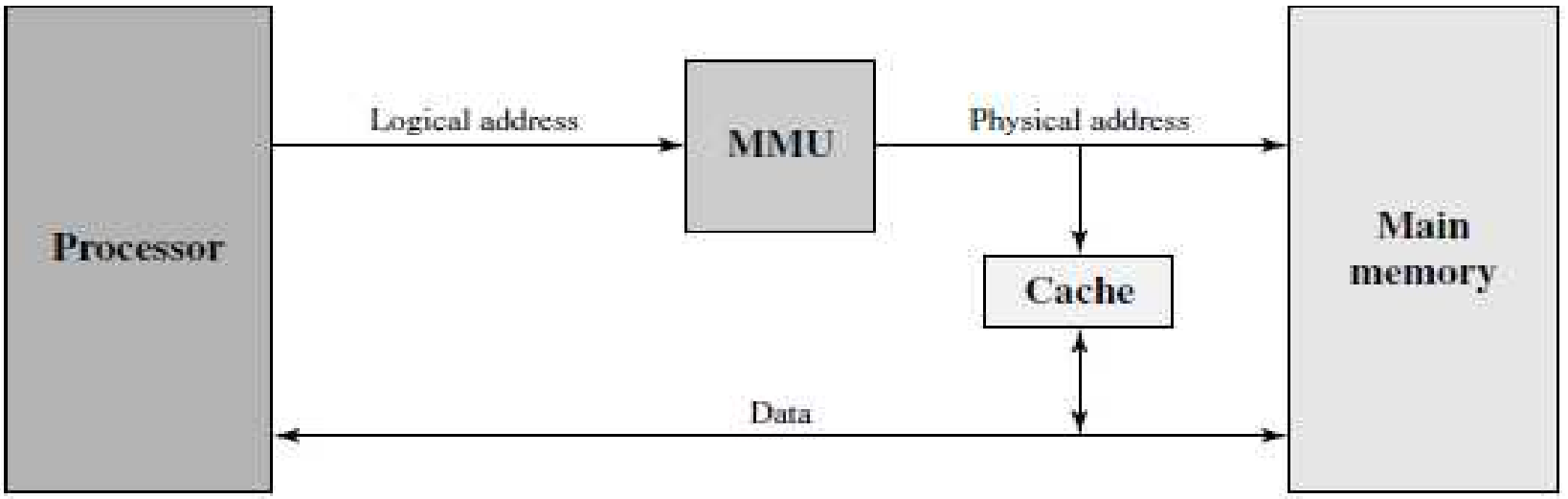
•Gotovo svi neugradjeni, a I mnogi ugradjeni procesori podrzavaju virtuelne memorije
•Virtuelna memorija je koncept koji programima dozvoljava da adresiraju memoriju sa logicke tacke gledista, bez obzira na kolicinu fizicki raspolozive memorije
•Kada se koristi virtuelna memorija, adresna polja masinskih instrukcija sadrze virtuelne adrese
•Za citanje I upis u glavnu memoriju, hardverska jedinica za upravljanje memorijom(MMU)pretvara svaku virtuelnu adresu u fizicku adresu u memoriji

•Adresa kes memorije

•Kada se koriste virtuelne adrese, projektant sistema moze da odabere da stavi kes izmedju procesora I MMU(jedinica za upravljanje memorijom) ili izmedju MMU I glavne memorije, slika sledeci slajd



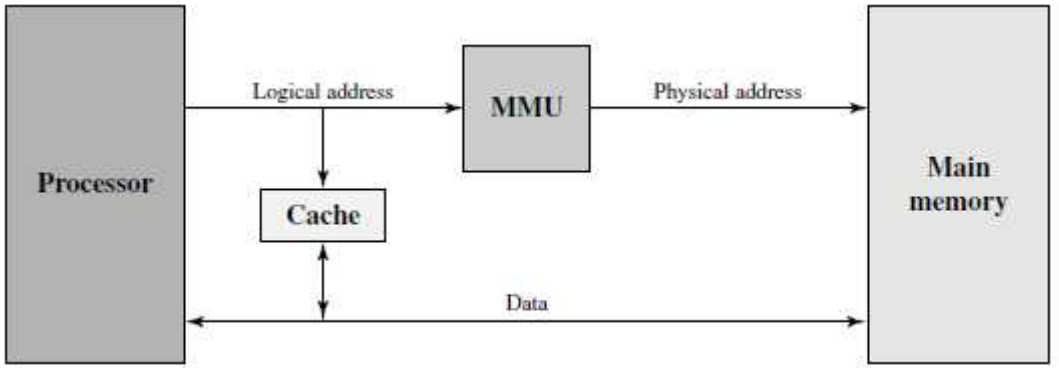
(a) Logical cache



(b) Physical cache

Figure 4.7 Logical and Physical Caches

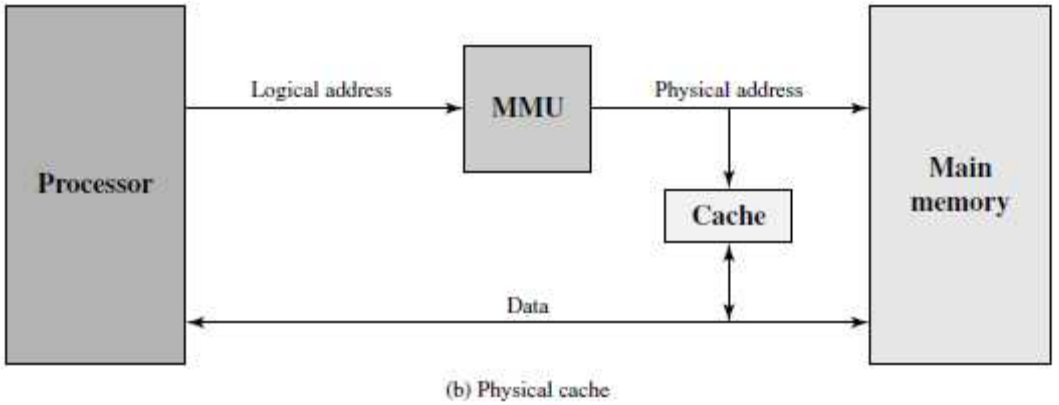
•Adresa kes memorije



(a) Logical cache

•Logicki kes poznat I kao virtuelni kes skadisti podatake koristeci virtuelne adrese
•Procesor pristupa kesu direktno bez prolaska kroz MMU

•Adresa kes memorije



(b) Physical cache

•Fizicki kes skladisti podatke koristeci fizicke adrese glavne memorije

•Adresa kes memorije

•Prednost logickog kesa je da je brzina pristupa kesu veca od brzine pristupa fizickom kesu, zbog toga sto MMU izvodi pretvaranje adrese
•Nedostatak je sto svaka primena koja koristi virtuelnu adresu, dobija isti adresni prostor virtuelne memorije
•To znaci da svaka primena vidi virtuelnu adresu koja pocinje na adresi 0.

•Adresa kes memorije

•Ista virtuelna adresa u dve razlicite primene odnosi se na dve razlicite fizicke adrese
•Kes memorija mora da se potpuno ocisti sa svakom promenom konteksta primene ili moraju da se dodaju esktra bitovi svakom redu kes memorije, da bi se identifikovao virtuelni prostor na koji se ta adresa odnosi

Velicina kes memorije

- Bilo bi dobro da je kes memorija dovoljno mala, jer sa aspekta cene, prosecna cena po bitu bude slicna ceni glavne memorije
- Na drugoj strani bilo bi dobro da je dovoljno velika
- Sto je veci kes, veci je broj logickih kola koja se koriste za adresiranje, pa su velike kes memorije nesto sporije od manjih
- Raspolozive povrsine cipa I ploce takodje ogranicavaju velicinu kesa

Velicine kes memorija nekih procesora

Table 4.3 Cache Sizes of Some Processors

Processor	Type	Year of Introduction	L1 Cache[a]	L2 Cache	L3 Cache
IBM 360/85	Mainframe	1968	16 to 32 kB	—	—
PDP-11/70	Minicomputer	1975	1 kB	—	—
VAX 11/780	Minicomputer	1978	16 kB	—	—
IBM 3033	Mainframe	1978	64 kB	—	—
IBM 3090	Mainframe	1985	128 to 256 kB	—	—
Intel 80486	PC	1989	8 kB	—	—
Pentium	PC	1993	8 kB/8 kB	256 to 512 KB	—
PowerPC 601	PC	1993	32 kB	—	—
PowerPC 620	PC	1996	32 kB/32 kB	—	—
PowerPC G4	PC/server	1999	32 kB/32 kB	256 KB to 1 MB	2 MB
IBM S/390 G4	Mainframe	1997	32 kB	256 KB	2 MB
IBM S/390 G6	Mainframe	1999	256 kB	8 MB	—
Pentium 4	PC/server	2000	8 kB/8 kB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 kB/32 kB	8 MB	—
CRAY MTA[b]	Supercomputer	2000	8 kB	2 MB	—
Itanium	PC/server	2001	16 kB/16 kB	96 KB	4 MB
SGI Origin 2001	High-end server	2001	32 kB/32 kB	4 MB	—
Itanium 2	PC/server	2002	32 kB	256 KB	6 MB
IBM POWER5	High-end server	2003	64 kB	1.9 MB	36 MB
CRAY XD-1	Supercomputer	2004	64 kB/64 kB	1 MB	—
IBM POWER6	PC/server	2007	64 kB/64 kB	4 MB	32 MB
IBM z10	Mainframe	2008	64 kB/128 kB	3 MB	24–48 MB

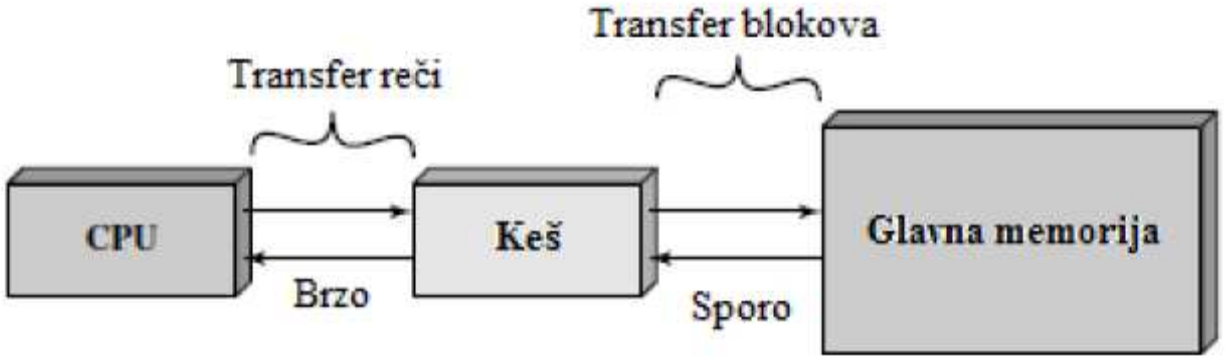
Funkcija preslikavanja

- Namena keš memorije je da pruži brzinu koja je blizu onoj koju imaju najbrže raspoložive memorije, a u isto vreme da obezbedi veliki kapacitet memorije po ceni jeftinih vrsta poluprovodničkih memorija.

Funkcija preslikavanja

- Keš memorija sadrži kopiju delova glavne memorije.
- Kada procesor pokuša da čita reči iz memorije, prvo se proverava da li je ta reč u kešu i ako jeste, reč se isporučuje procesoru.
- Ako nije, blok glavne memorije, koji se sastoji od nekog fiksnog broja reči učitava se u keš i onda se reč isporučuje procesoru

Funkcija preslikavanja



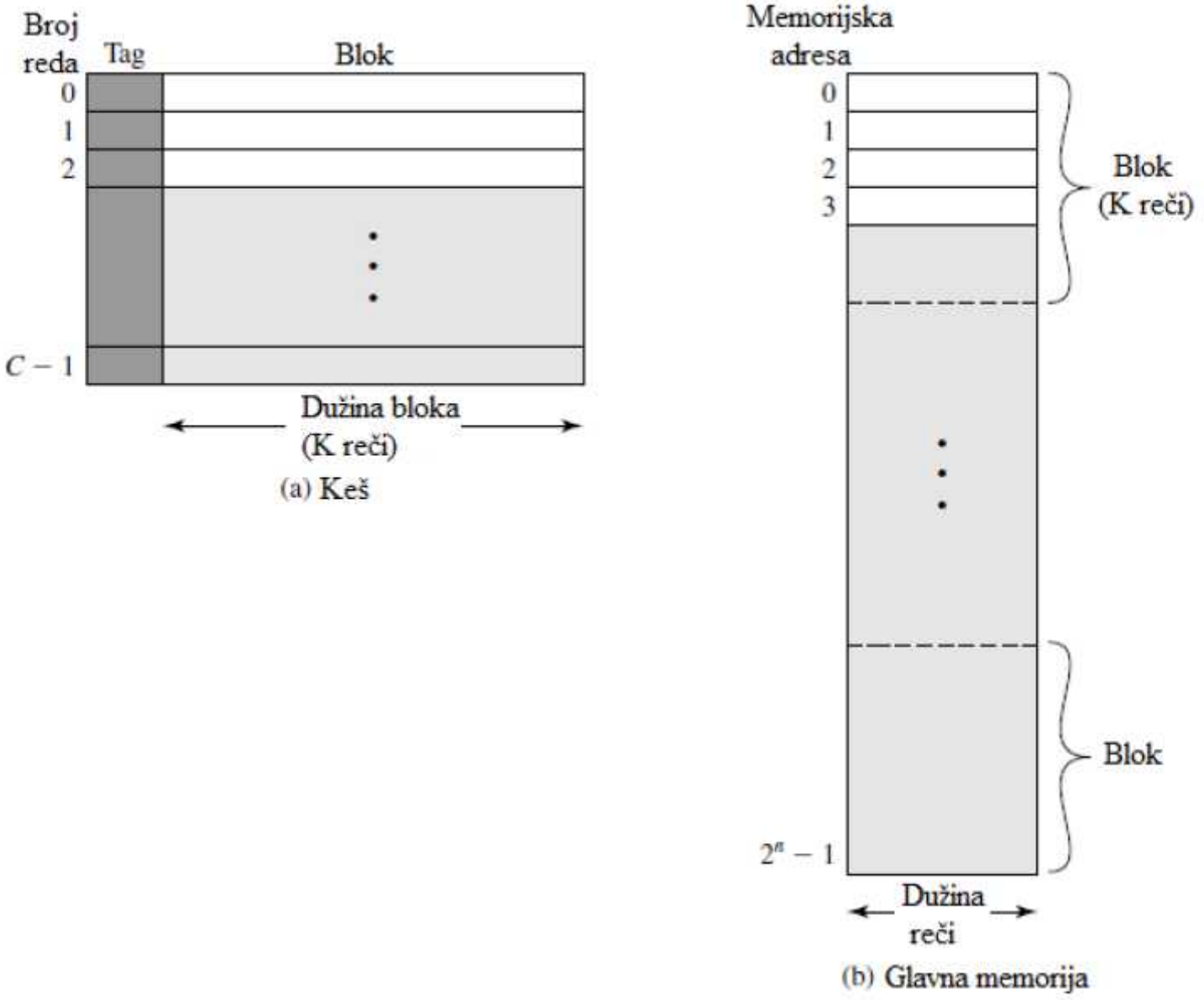
Odnos procesora, keša i glavne memorije

Funkcija preslikavanja

• Zbog fenomena lokalnosti reference, kada se blok podataka donese u keš da bi se zadovoljila jedna referenca memorije, verovatno će biti budućih referenci na tu istu memorijsku lokaciju ili na drugu reč u bloku.

Funkcija preslikavanja

- Glavna memorija se sastoji od 2^n adresibilnih reči, gde svaka lokacija (reč) ima jedinstvenu n-bitnu adresu .
- U svrhu preslikavanja, ta memorija je zamišljena da se sastoji od izvesnog broja blokova fiksne dužine, svaki po *k* reči.
- To znači da postoji $M = 2^n/k$ blokova u glavnoj memoriji.



Organizacija keša i glavne memorije

Funkcija preslikavanja

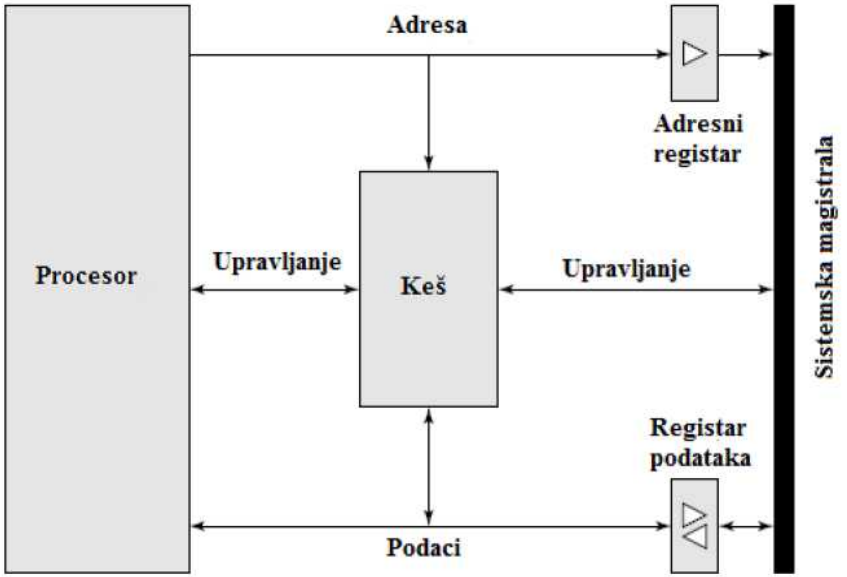
- Keš memorija se sastoji od *C* redova.
- Svaki red sadrži *k* reči, plus tag od nekoliko bitova.
- Broj reči u redu se zove veličina reda.
- Broj redova u kešu je mnogo manji od broja blokova u glavnoj memoriji (*C* << *M*)

Funkcija preslikavanja

- U bilo kom trenutku, neki podskup blokova memorije se nalazi u redovima u kešu.
- Ako se čita reč u bloku memorije, taj blok se prenosi u jedan od redova keša.
- S obzirom na to da ima više blokova od redova, pojedinačni red se ne može jedinstveno i trajno dodeliti određenom bloku.

Funkcija preslikavanja

• Prema tome, svaki red poseduje tag, koji predstavlja deo adrese glavne memorije i koji identifikuje blok koji je trenutno uskladišten u kešu.



Funkcija preslikavanja

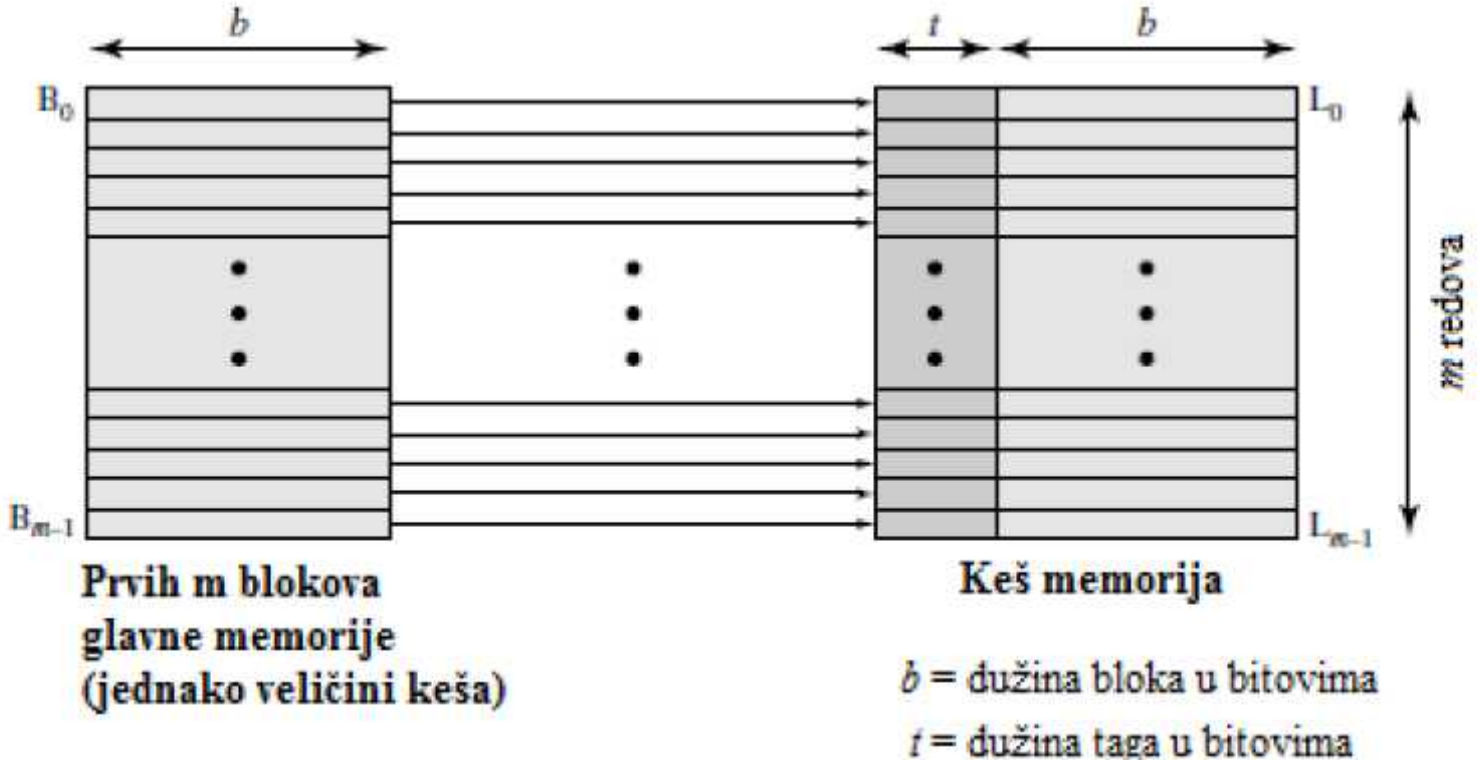
- S obzirom na to da ima manje redova keša od blokova u glavnoj memoriji, potreban je algoritam za preslikavanje blokova glavne memorije u redove keša.
- Pored toga, potreban je način na koji se određuje koji blok glavne memorije trenutno zauzima red u kešu. Najčešće se koriste tri tehnike preslikavanja:

Funkcija preslikavanja

- Direktno preslikavanje,
- Asocijativno preslikavanje,
- Asocijativno preslikavanje skupa (set-asocijativno preslikavanje).

Direktno preslikavanje

•Ovo je najjednostavnija tehnika preslikavanja gde se svaki blok glavne memorije može preslikati samo u jedan mogući red keša



Direktno preslikavanje

- Preslikavanje se izvršava po sledećem obrascu:

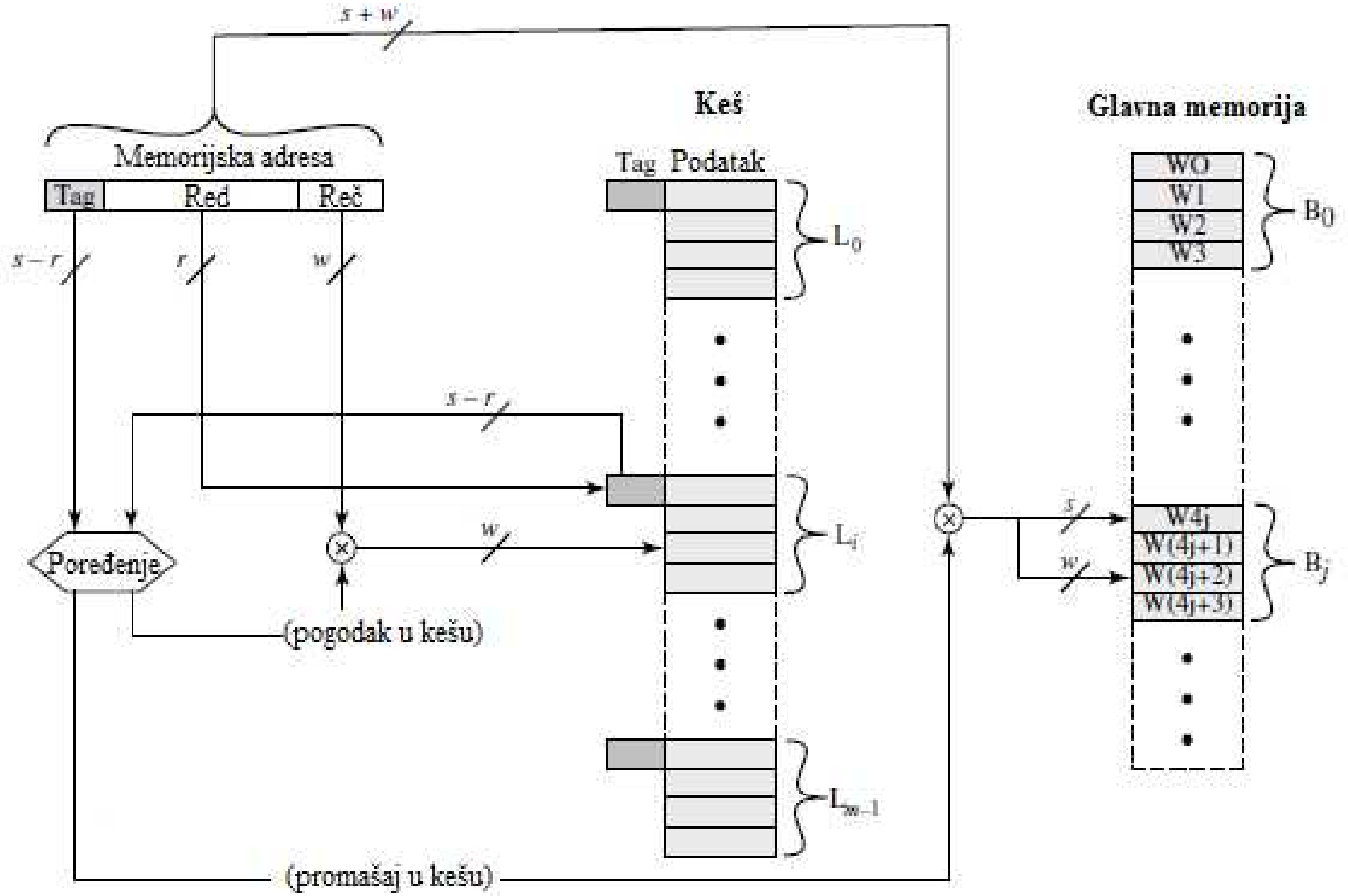
$$i = j \text{ moduo } m,$$

gde je:

i – broj reda u kešu,

j – broj bloka u glavnoj memoriji,

m – broj redova u kešu.

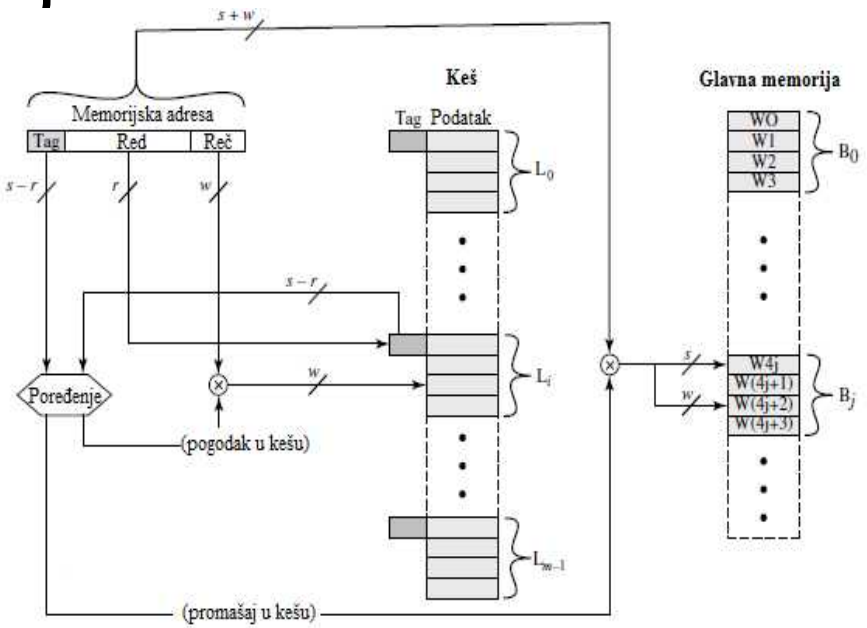

s + w
Memorijska adresa
Tag
Red
Reč
s − r
r
w
Keš
Tag
Podatak
L0
Li
Lm−1
s − r
w
Poređenje
(pogodak u kešu)
(promašaj u kešu)
Glavna memorija
W0
W1
W2
W3
B0
W4j
W(4j+1)
W(4j+2)
W(4j+3)
Bj
s
w

Direktno preslikavanje

- Funkcija preslikavanja se lako implementira koristeći adrese.
- Kada se pristupa kešu, svaka adresa glavne memorije može da se posmatra kao da se sastoji iz tri dela.
- Najmanje značajnih *w* bitova identifikuju jedinstvenu reč unutar bloka glavne memorije.
-

Direktno preslikavanje

- To znači da jedan blok glavne memorije sadrži 2^w reči.
- Ostalih *s* bitova određuju 2^s blokova glavne memorije.
- Logika keša interpretira tih *s* bitova kao tag od *s-r* bitova i polje reda od *r* bitova.
- Polje reda identifikuje jedan od $m = 2^r$ redova keša.



Direktno preslikavanje

•Kod direktnog preslikavanja, prvo se čita red u adresi i tag u kešu koji odgovara tom redu
• Tag se poredi sa tagom u memorijskoj adresi.
•Ako su isti, blok se nalazi u kešu i reč se čita.
• Ako nisu isti, kažemo da je došlo do promašaja u kešu i pristupa se glavnoj memoriji.
•Blok iz glavne memorije u kojem se nalazi tražena reč se prebacuje u keš memoriju.
•Tada se čita blok *s* i reč na adresi *w* iz tog bloka.

Direktno preslikavanje

Rezime:

- Dužina adrese: $s + w$ bitova,
- Kapacitet memorije: 2^{s+w} reči,
- Veličina bloka (ili reda): 2^{w} reči,
- Broj blokova u glavnoj memoriji: 2^{s},
- Broj redova u keš memoriji: 2^{r},
- Veličina taga: $s - r$ bitova.

Direktno preslikavanje

•Kod direktnog preslikavanja, prvo se čita red u adresi i tag u kešu koji odgovara tom redu
• Tag se poredi sa tagom u memorijskoj adresi.
•Ako su isti, blok se nalazi u kešu i reč se čita.
• Ako nisu isti, kažemo da je došlo do promašaja u kešu i pristupa se glavnoj memoriji.
•Blok iz glavne memorije u kojem se nalazi tražena reč se prebacuje u keš memoriju.
•Tada se čita blok *s* i reč na adresi *w* iz tog bloka.

Direktno preslikavanje

•Tehnika direktnog preslikavanja je jednostavna i jeftina za implementaciju.
• Njen glavni nedostatak je to što postoji fiksna lokacija u kešu za svaki blok glavne memorije.
•Ako se dogodi da program stalno traži reči iz dva različita bloka koji se preslikavaju u isti red, onda će se blokovi stalno izbacivati iz keša, a verovatnoća pogotka će biti mala.
•Ovaj fenomen se zove “uzaludan rad” ili *thrashing*

Direktno preslikavanje

- U tabeli su dati redovi keša u koje se preslikavaju blokovi operativne memorije

Red u kešu	Dodeljeni blokovi glavne memorije
0	$0, m, 2m, \ldots, 2^s - m$
1	$1, m + 1, 2m + 1, \ldots, 2^s - m + 1$
⋮	⋮
$m - 1$	$m - 1, 2m - 1, 3m - 1, \ldots, 2^s - 1$

Primer

- Računar poseduje operativnu memoriju veličine 4 kB i keš memoriju veličine 64 B.
- Veličina bloka u memoriji je 8 B.
- Prikazati organizaciju glavne memorije, keša i polje fizičke adrese, ako je u pitanju direktno preslikavanje keša.

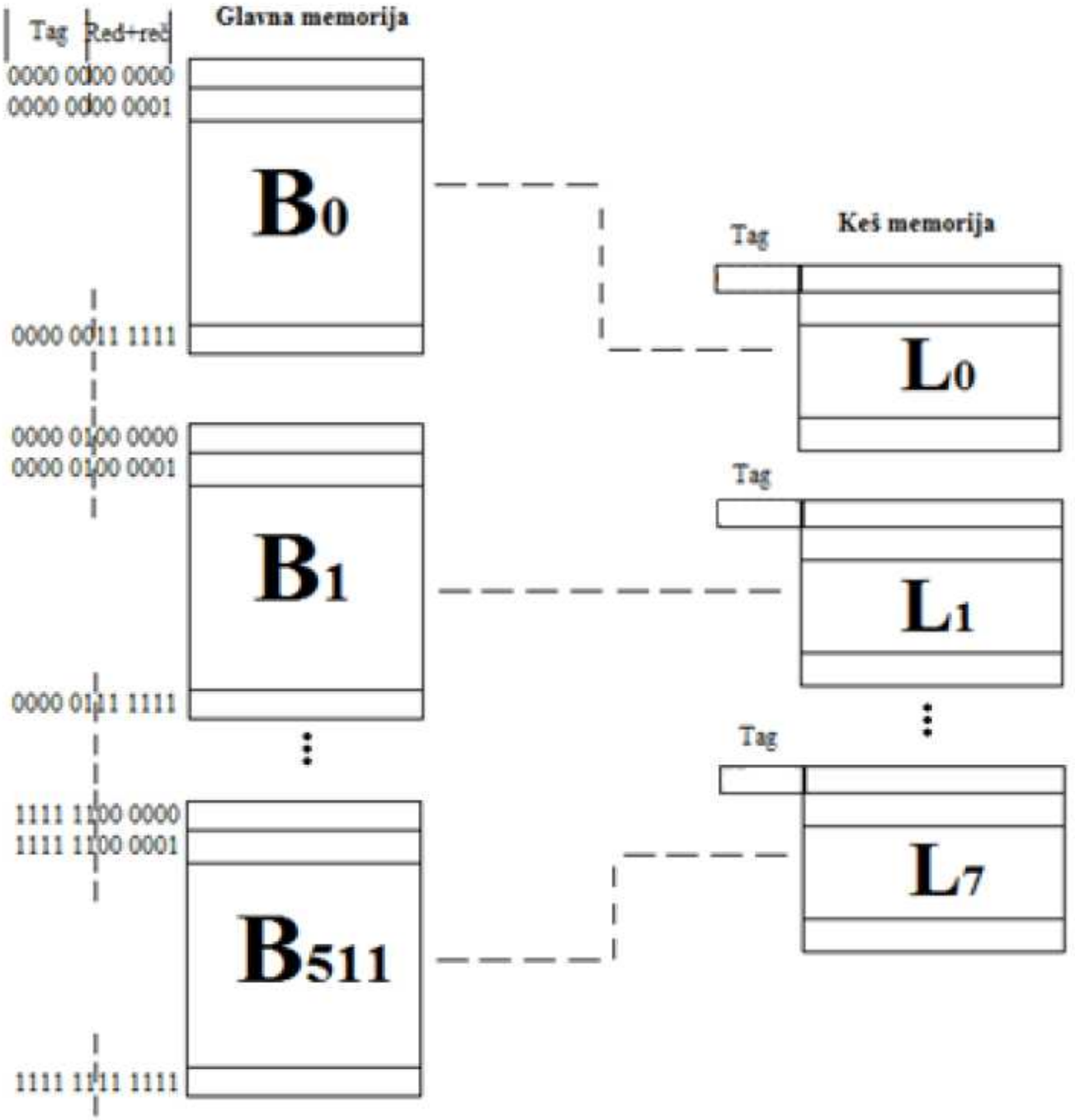
Primer

$C_M = 4\ \text{kB} = 2^2 \cdot 2^{10} = 2^{11}\ \text{B}$ → dužina memorijske adrese je n = 12 bita.

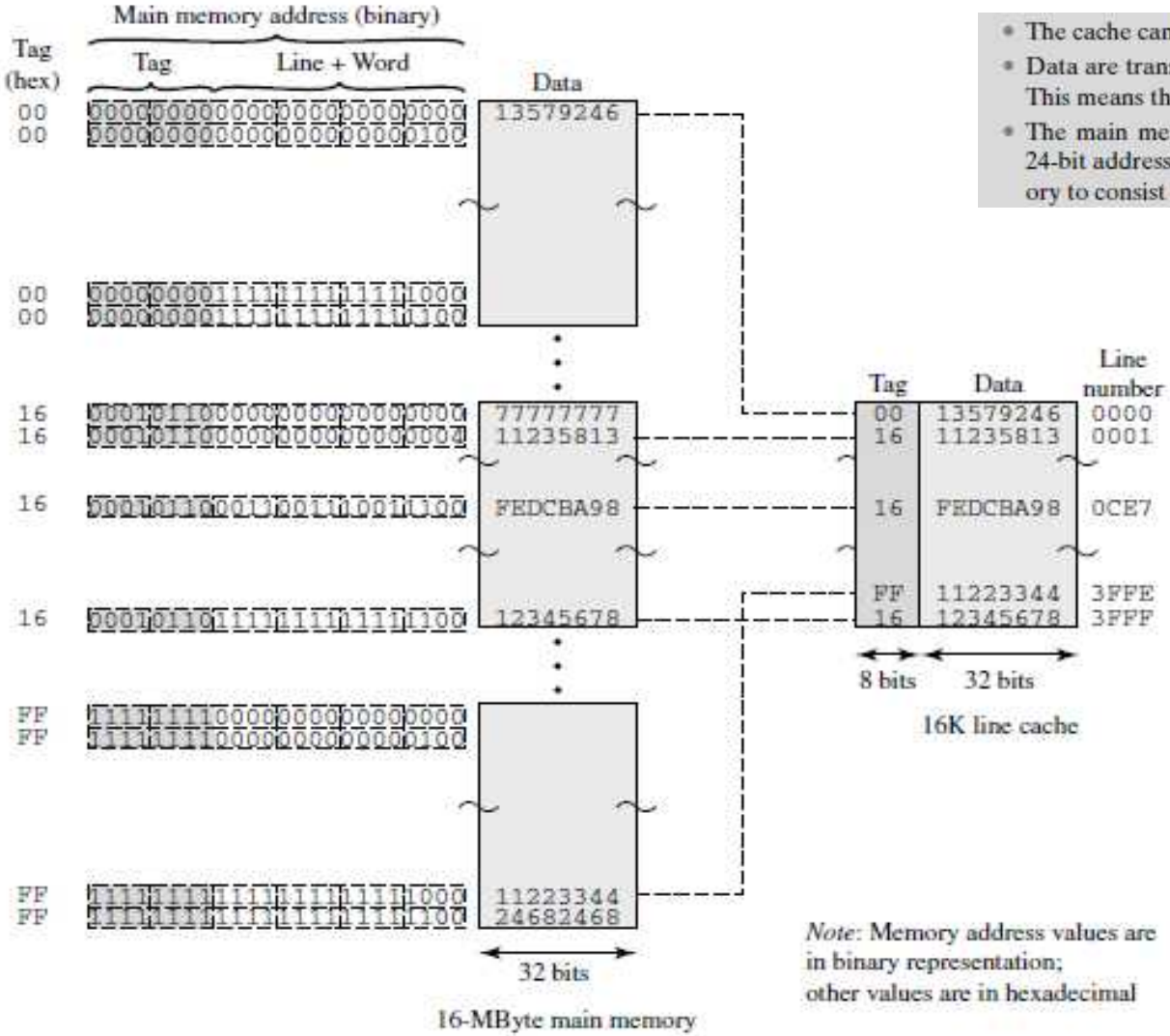
$K = 8\ \text{B} = 2^3\ \text{B}$ → w = 3 bita

$M = \frac{C_M}{K} = \frac{2^{12}}{2^3} = 2^9 = 512$ blokova → s = 9 bitova

$C = \frac{C_K}{K} = 8$ → postoji 8 redova u kešu → r = 3 bita


Tag
Red+red
Glavna memorija
0000 0000 0000
0000 0000 0001
B0
0000 0011 1111
0000 0100 0000
0000 0100 0001
B1
0000 0111 1111
1111 1100 0000
1111 1100 0001
B511
1111 1111 1111
Tag
Keš memorija
L0
Tag
L1
Tag
L7

Primer 2



- The cache can hold 64 KBytes.
- Data are transferred between main memory and the cache in blocks of 4 bytes each. This means that the cache is organized as 16K $= 2^{14}$ lines of 4 bytes each.
- The main memory consists of 16 Mbytes, with each byte directly addressable by a 24-bit address ($2^{24} = 16$M). Thus, for mapping purposes, we can consider main memory to consist of 4M blocks of 4 bytes each.

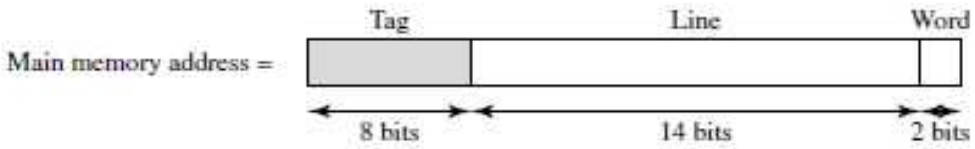
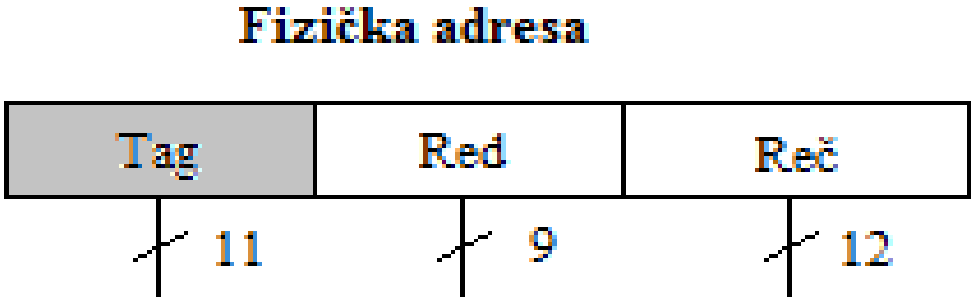


Figure 4.10 Direct Mapping Example

Kes sistem se predstavlja 24-bitnom adresom.
Da bi se pristupilo odredjenom redu koristi se 14-bitni broj reda kao indeks u kesu.
Ako 8-bitni broj taga odgovara broju taga koji je uskladisten u tom redu, onda se 2-bitni broj reci koristi da se izabere jedan od 4 bajta u to redu.
U suprotnom 22-bitno polje za tag plus red koristi se da donese blok iz glavne memorije.
Stvarna adresa za donosenje je 22-bitno polje za tag plus red, spojeno sa dva 0 bita, tako da se donesu cetiri bajta pocevsiod granice bloka.

Primer 3

Računar poseduje memorijski podsistem koji se sastoji iz operativne memorije i keša sa direktnim preslikavanjem.
Odrediti kapacitet glavne memorije i keša, ako su polja fizičke adrese prikazana na slici.



Fizička adresa

$C_M = 2^n = 2^{32}$ b $= 2^2 \cdot 2^{30} = 4$ GB → kapacitet glavne memorije je 4 GB

$K = 2^w = 2^{12}$ B $= 2^2 \cdot 2^{10} = 4$ kB → veličina bloka (ili reda) je 4 kB

$M = \frac{C_M}{K} = 2^{20}$ → broj blokova u operativnoj memoriji

$C = 2^r = 2^9 = 512$ → broj redova u kešu

$C_K = C \cdot K = 2^{21}$ B $= 2$ MB → kapacitet keš memorije.