5. Multiprocesorski sustavi – višeprocesorski MIMD sustavi

MIMD arhitektura obilježena je višestrukim instrukcijskim tokom i višestrukim tokom podataka. Svaki od procesora pribavlja i izvršava svoje vlastite instrukcije na svojim podacima. Višeprocesorski MIMD sustavi ili *multiprocesorski sustavi* iskorištavaju *paralelizam na razini procesa i dretvi*.

• U multiprocesorskom sustavu svaki procesor može izvršavati njemu dodijeljen *proces*.

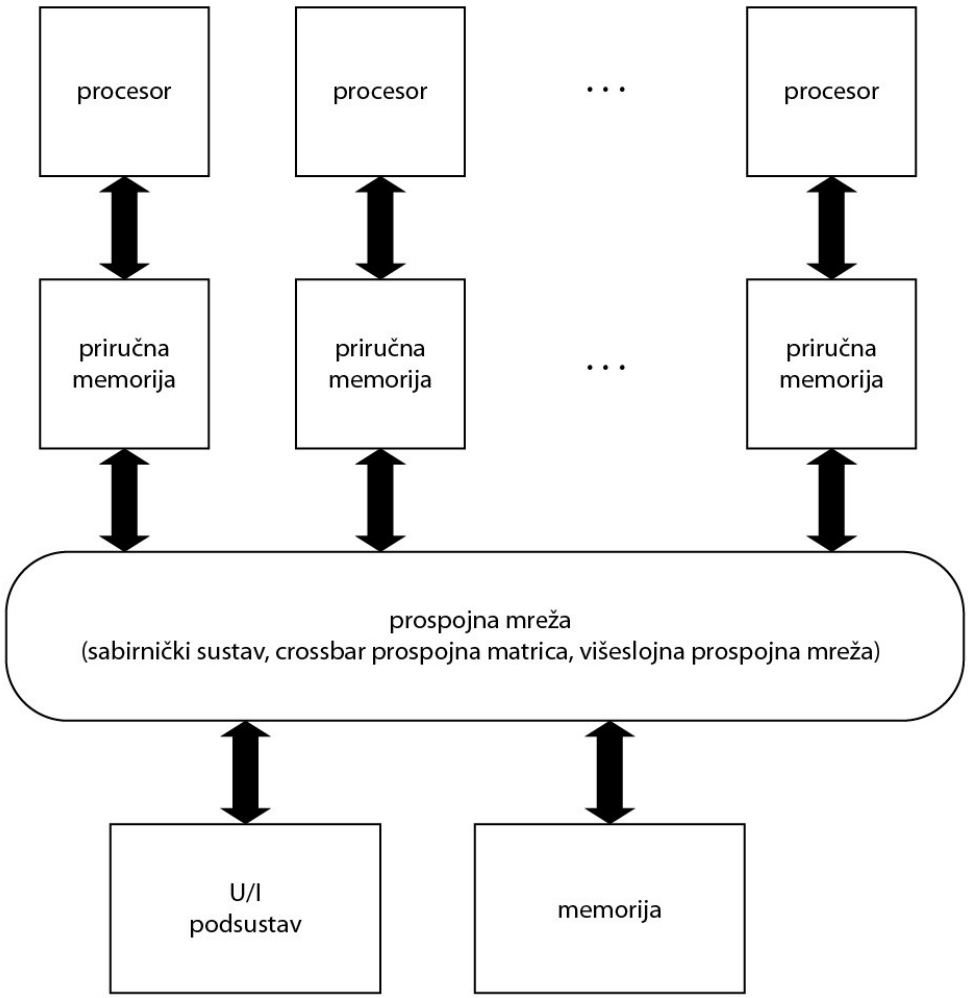
• Svaki od procesa može imati više dretvi tako da se izvođenje jednog procesa s većim brojem dretvi može povjeriti većem broju procesora. Višedretvena arhitektura temeljena na MIMD-u, u načelu, dopušta istodobno izvođenje većeg broja procesa s izdvojenim adresnim prostorima i izvođenje više dretvi koje dijele adresni prostor.

Osnovna značajka multiprocesorskog sustava jest veći broj procesora približno jednakih (vrlo često) identičnih obilježja koji na izvjestan način dijele zajednički memorijski prostor.

Ovisno o broju procesora i načinu organizacije memorijskog sustava multiprocesorski se sustavi mogu klasificirati u sljedeće skupine:

- sustavi s uniformnim pristupom memoriji UMA (engl. *Uniform Memory Access*),
- sustavi s neuniformnim pristupom memoriji NUMA (engl. *Nonuniform Memory Access*),
- sustavi samo s priručnom memorijom COMA (engl. *Cache-Only Memory Architecture*).

U *UMA modelu multiprocesorskog sustava* svi procesori dijele zajedničku fizičku memoriju i svi imaju jednako vrijeme pristupa svakoj od riječi u memoriji (uniformni, odnosno ujednačeni pristup memoriji). Svaki od procesora može imati i svoju vlastitu priručnu memoriju organiziranu u jednu ili više razina. Na sličan način kao što dijele memoriju, procesori dijele resurse U/I podsustava.



UMA model multiprocesorskog sustava

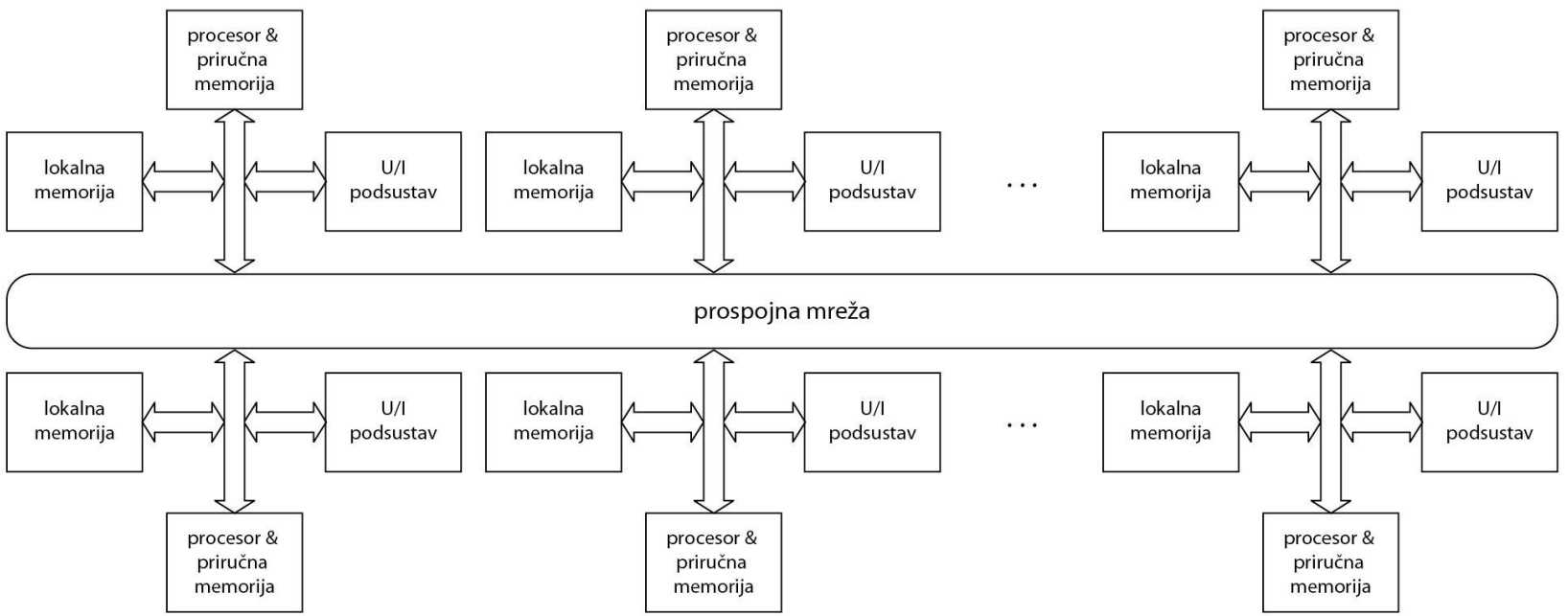
UMA model se još naziva i *multiprocesorski sustav sa središnjom dijeljenom memorijom* (engl. *centralized shared-memory*) ili *simetrični multiprocesorski sustav s dijeljenom memorijom* SMP (engl. *symmetric shared-memory multiprocessor*) – zato što memorija ima isti ("simetrični") odnos spram svih procesora.

• Komunikacija procesora sa zajedničkom memorijom i U/I podsustavom ostvaruje se prospojnom mrežom koja ovisno o zahtijevanoj propusnosti može biti ostvarena sabirničkim sustavom, crossbar prospojnom matricom ili višerazinskom prospojnom mrežom (npr. Omega).

Značajke:

• UMA model pogodan je za relativno mali broj procesora (manji od 100) jer je za taj broj procesora još uvijek moguće ostvariti dijeljenje i uniformni pristup memoriji. UMA model još se naziva i *čvrsto povezan multiprocesorski sustav* (engl. *tightly coupled*) zbog visokog stupnja dijeljenja zajedničkih resursa (memorije i U/I podsustava).
• UMA model multiprocesorskog sustava najčešće se koristi zato što je podesan za primjene opće namjene i izvedbu obrade dodjeljivanjem vremena u višekorisničkim okruženjima.

Multiprocesorski sustavi s neuniformnim pristupom memoriji NUMA nazivaju se još i *sustavi s porazdijeljenom memorijom* (engl. *distributed-memory multiprocessor*) imaju umjesto centralizirane memorije, memoriju porazdijeljenu procesorima.



Zbirka svih lokalnih memorija oblikuje globalni adresni prostor kojem mogu pristupiti svi procesori u sustavu. Zbog takve organizacije memorije razlikujemo dvije vrste pristupa memoriji:

- brzi pristup memoriji (kraće vrijeme pristupa) kada procesor pristupa svojoj lokalnoj memoriji,
- sporiji pristup (dulje vrijeme pristupa) kada procesor pristupa "udaljenoj" memoriji koja je, zapravo, lokalna memorija nekog drugog procesora. Dulje vrijeme pristupa uzrokovano je dodatnim kašnjenjima jer se "udaljenoj" memoriji pristupa kroz prospojnu mrežu.

• Multiprocesorski sustavi oblikovani u skladu s modelom NUMA imaju veliki broj procesora, npr. nekoliko stotina ili tisuća, i zato se za toliki broj procesora teško može realizirati središnja memorija sa zahtijevanom, odnosno prihvatljivom propusnosti (engl. *memory bandwidth*).

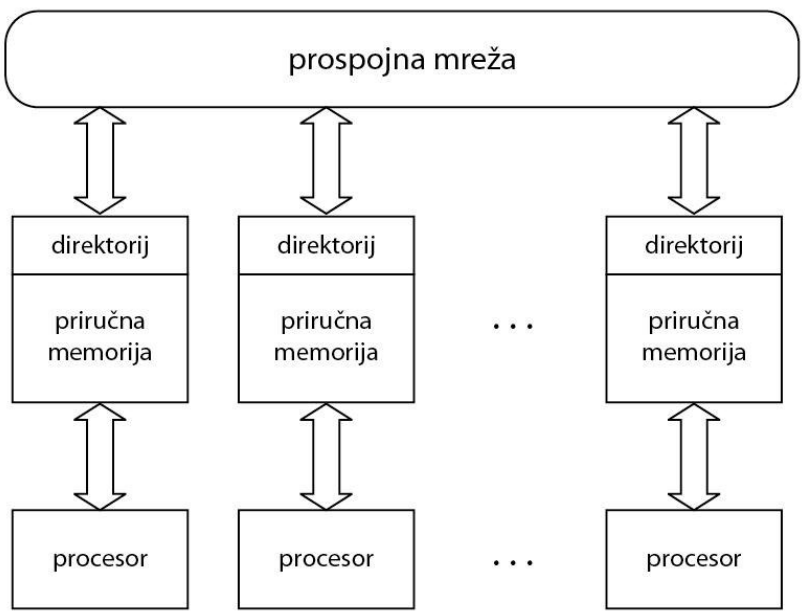
• Svaki čvor u NUMA modelu sastoji od procesora, lokalne memorije, U/I podsustava i sučelja za pristup prospojnoj mreži. Ovisno o izvedbi NUMA modela, čvor može biti sastavljen od određenog broja procesora i lokalnih memorijskih modula tako da čini procesorsku nakupinu (engl. *processor cluster*).

- Primjer takve organizacije je multiprocesorski sustav velikih razmjera (engl. *large-scale multiprocessor*) Cedar.

Multiprocesorski sustavi temeljeni na modelu NUMA često se zbog načina na koji je ostvarena veza između procesora nazivaju i *labavo povezanim* (engl. *loosely coupled*).

Multiprocesorski sustavi temeljeni na modelu COMA (Cache-Only Memory Architecture) su poseban slučaj NUMA multiprocesorskog sustava u kojem je umjesto glavne memorije porazdijeljena priručna memorija tako da sve priručne memorije oblikuju globalni adresni prostor.

Udaljeni pristup priručnoj memoriji ostvaruje se pomoću distribuiranih direktorija.



• Multiprocesorski sustavi temeljeni na UMA modelu dijele zajednički memorijski prostor tako da procesori mogu *međusobno izmjenjivati podatke instrukcijama load i store* (zato se i nazivaju sustavi s dijeljenom memorijom).

• U sustavima temeljenim na NUMA modelu komunikacija između procesora odvija se porukama koje procesori izmjenjuju preko prospojne mreže – takvi se multiprocesorski sustavi nazivaju još i *multiprocesorski sustavi s prosljeđivanjem poruka* (engl. *message-passing multiprocessor*).

• Organizacija *multiprocesorskog sustava s dijeljenom memorijom* najrašireniji te da mnogi proizvođači računala (npr. Sun, AMD, Intel, IBM) imaju proširenje svojih proizvodnih uniprocesorskih ili jednoprocesorskih linija na multiprocesorske sustave na čipu - *višejezgreni mikroprocesori* (engl. *multicore microprocessor*) ili kraće *višejezgreni procesori* (izraz "jezgra" (engl. *core*) upotrebljava se za procesor u višejezgrenom procesoru).

Višejezgreni je procesor, zahvaljujući tehnologiji visokog stupnja integracije, procesor koji ima dva ili više procesora (jezgri) na jednom čipu.

Osnovne značajke:

- manji potrošak snage,
- djelotvorna istodobna obrada više zadataka (programa, procesa, dretvi)

6. Koherencija priručne memorije u multiprocesorskom sustavu

• problem koherencije, i to za multiprocesorski sustav sa središnjom dijeljenom memorijom (UMA model)
• budući da svaki procesor ima svoju privatnu priručnu memoriju, ali dijeli i središnju zajedničku memoriju, u svakoj od priručnih memorija pohranjeni su "privatni" podaci koji se odnose na lokalni proces, ali i zajednički podaci koji se odnose na procese drugih procesora i koji su dohvaćeni i pohranjeni u pojedine priručne memorije na temelju komunikacije procesora, odnosno njihovim pristupom zajedničkoj memoriji.

- istodobno se dijeljeni podatak nalazi u zajedničkoj memoriji, a njegove kopije distribuirane su u priručne memorije pojedinih procesora.

Sada nastupa problem!

- Ako neki od procesora promijeni vrijednost zajedničkog podatka u svojoj priručnoj memoriji, taj se podatak treba promijeniti i u zajedničkoj memoriji, ali i u svim priručnim memorijama ostalih procesora koji koriste taj podatak (uvjet koherencije priručne memorije)

Primjer:
Pretpostavimo, radi jednostavnosti, da imamo multiprocesorski sustav koji ima samo dva procesora (jezgre) i da je ostvaren kao sustav sa središnjom dijeljenom memorijom. Dakle, sustav ima zajedničku memoriju, a svaki od procesora ima priručnu memoriju. Neka priručne memorije koriste tehniku obnavljanja sadržaja memorije „pohranjivanje-kroz“ (engl. *store-through)* .

- Pretpostavimo da oba procesora (*procesor 1* i *procesor 2*) dijele zajedničku varijablu V koja je pohranjena u središnjoj dijeljenoj memoriji na lokaciji X.
- Pretpostavimo, također, da se u trenutku t = 0 varijabla V ne nalazi u priručnim memorijama *procesora 1* i *procesora 2*.

• U trenutku t = 1 *procesor 1* dohvaća (čita) varijablu V iz središnje memorije i pohranjuje je u svoju priručnu memoriju.
• U sljedećem trenutku (t = 2) *procesor 2* dohvaća (čita) varijablu V iz središnje memorije i pohranjuje je u svoju priručnu memoriju
• U ovom trenutku (t = 2) koherencija podataka nije narušena: oba procesora imaju pohranjenu vrijednost varijable V jednaku onoj koja je pohranjena na memorijskoj lokaciji X u središnjoj memoriji
• Pretpostavimo da u sljedećem trenutku (t = 3) *procesor 1* mijenja vrijednost varijable V u V' i pohranjuje je u svoju priručnu memoriju. Uporabom tehnike „pohranjivanje-kroz" *procesor 1* upisuje novu vrijednost V' na memorijsku lokaciju X u središnjoj memoriji.

• Sada u vremenskom trenutku t = 3 imamo sljedeću situaciju: *procesor 1* u svojoj priručnoj memoriji ima pohranjenu novu vrijednost varijable V, tj. V', središnja dijeljena memorija ima također na memorijskoj lokaciji X pohranjenu novu vrijednost V'.

• Međutim, *procesor 2* ima u svojoj priručnoj memoriji pohranjenu staru vrijednost varijable V. Ako u sljedećem trenutku (t = 4) *procesor 2* dohvaća varijablu V iz svoje priručne memorije – dohvatit će staru vrijednost V, a ne V'. Došlo je do povrede koherencije – umjesto prave vrijednosti V' (koja je pohranjena i priručnoj memoriji *procesora 1* i u središnjoj memoriji), *procesor 2* koristit će se starom vrijednosti V.

Problem (ne)koherencije priručne memorije rješava se programski ili sklopovski.

-Relativno jednostavno programsko rješenje sastoji se u tome da prevodilac tijekom prevođenja programa označi informaciju (podatke) koja se smije pohraniti (engl. *cacheable*) i onu koja se ne smije pohraniti u priručnu memoriju (engl. *non-cacheable*). Sve zajedničke varijable koje se mogu mijenjati upisivanjem označavaju se kao nepodesne (*non-cacheable*) za pohranu u priručne memorije i njima procesori jedino mogu pristupiti tako da pristupe središnjoj dijeljenoj memoriji.

Ovo rješenje degradira performansu sustava zbog pojačanog prometa između središnje dijeljene memorije i procesora.

Multiprocesorski sustavi obično koriste sklopovska rješenja kojima se održava koherencija priručnih memorija.

Protokoli kojima se održava koherencija u multiprocesorskim sustavima nazivaju se *protokoli koherencije priručne memorije* (engl. *cache coherence protocol*) i mogu se klasificirati na:

- *snooping* protokole (engl. *snoop* – njuškati),
- protokole koji se temelje na direktoriju (engl. *directory based*).

Snooping protokol prvenstveno je namijenjen multiprocesorskim sustavima koji se temelje na prospojnoj mreži koja je izvedena kao zajednička sabirnica.

Taj se protokol može koristiti i u multiprocesorskim sustavima sa složenijom prospojnom mrežom (npr. višerazinskom) uz izvjesne sklopovske preinake kojima se omogućuje odašiljanja informacije svim procesorima o promašajima u priručnoj memoriji.

Protokoli koji se temelje na *direktoriju* obično se koriste u multiprocesorskim sustavima koji imaju prospojnu mrežu izvedenu kao višerazinsku.

Protokoli ove vrste mogu se temeljiti na jednom središnjem direktoriju u kojem je pohranjena informacija o statusu dijeljenih blokova fizičke memorije – središnji direktorij sadržava kopije svih direktorija priručnih memorija s informacijom u kojim se priručnim memorijama kopije blokova podataka nalaze.

Snooping protokolom postiže se koherencija podataka u priručnim memorijama i središnjoj dijeljenoj memoriji tako da svi upravljači priručnih memorija (engl. *cache controller*) nadgledaju događaje na zajedničkoj sabirnici ("njuškaju" zajedničku sabirnicu) i određuju trebaju ili ne trebaju kopirati dijeljeni blok podataka.

Održavanje koherencije temelji se na dvjema operacijama:

- čitanju i
- pisanju.

Višestruke kopije koje se nalaze u priručnim memorijama pojedinih procesora nisu problem kada je riječ o operaciji čitanja, međutim, procesori imaju i ekskluzivno pravo upisivanja podataka u te dijeljene blokove podataka.

Koherencija podataka u multiprocesorskom sustavu zahtijeva da bilo koji procesor koji koristi dijeljeni blok podataka i kada čita podatak mora čitati njegovu novu vrijednost (onu nakon operacije upisivanja)

Snooping protokol mora, na temelju nadgledanja sabirnice kojom se i ostvaruju prijenosi dijeljenih blokova podataka između središnje dijeljene memorije i priručne memorije, locirati sve priručne memorije koje dijele taj podatak koji će se upisati.

Posljedica upisivanja (promjene) dijeljenog podatka može biti dvojaka (ovisno o vrsti protokola):

- sve ostale kopije dijeljenog bloka podataka koje se nalaze u ostalim priručnim memorijama proglašavaju se nevažećima (tehnika "piši i proglasi nevažećim"; engl. *write-invalidate*)
- aktivira postupak osvježavanja svih ostalih kopija dijeljenog bloka u ostalim priručnim memorijama (tehnika "piši i obnovi"; engl. *write-update*).

U tehnici "piši i proglasi nevažećim" procesor koji će modificirati dijeljeni podatak mora, neposredno prije nego što to učini, generirati poseban signal "nevažeće" (engl. *invalidation signal)* i postaviti ga na sabirnicu te time obavijestiti sve druge priručne memorije da su kopije dijeljenog bloka podataka nevažeće.

Kada upravljači priručnih memorija prime taj signal, provjeravaju sadržava li ona taj dijeljeni blok podataka, a ako sadržava, taj blok podataka u priručnoj memoriji proglašava nevažećim. Pokušaj čitanja tog podatka od strane nekog procesora, zbog nevažećeg bloka, izazvat će promašaj tako da će se iz središnje dijeljene memorije dohvatiti blok s modificiranom (novom) vrijednosti podatka.

Tehnika "piši i obnovi" umjesto da kopije dijeljenog bloka podataka proglasi nevažećim, dopušta procesoru koji je modificirao podatak da taj modificirani podatak pošalje preko sabirnice tako da se obnove sve kopije u ostalim priručnim memorijama.

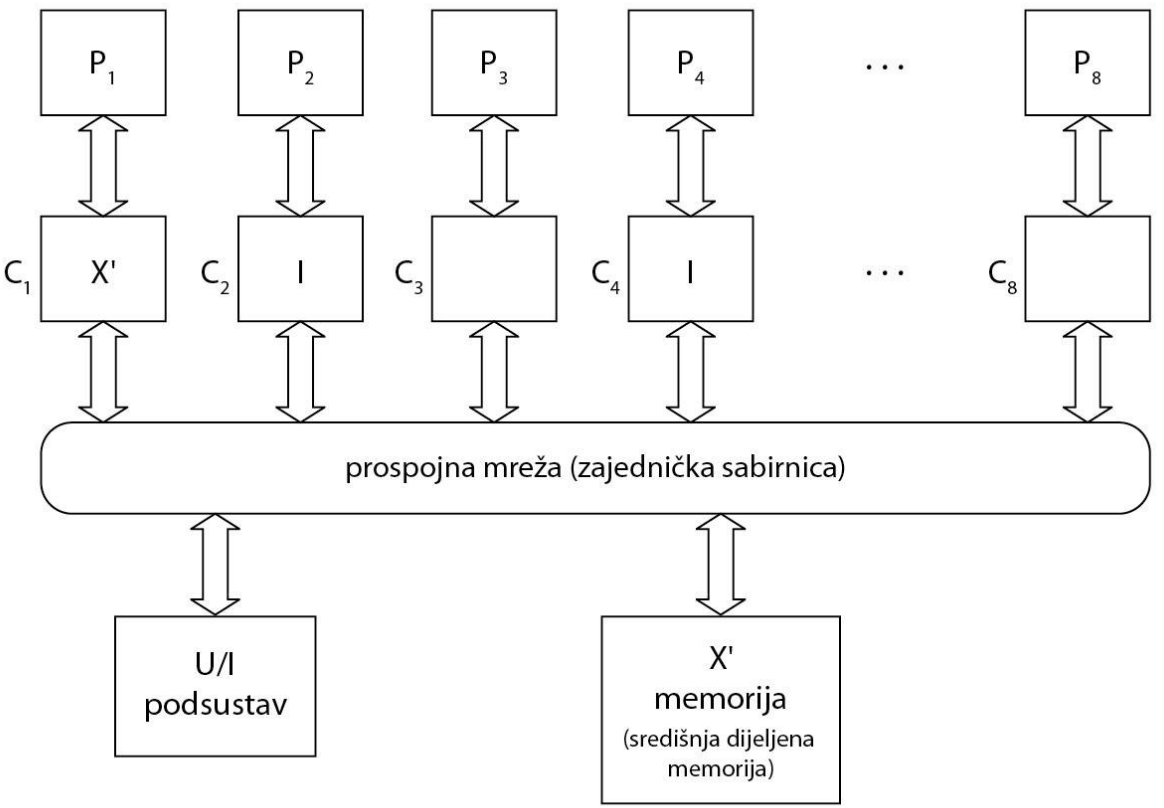
Tehnika "piši i obnovi" slična je postupku "pohranjivanja-kroz" jer se svaka promjena zajedničkog podatka šalje preko sabirnice središnjoj dijeljenoj memoriji, samo što se ovdje obnavljaju i sadržaji priručnih memorija.

- Svaka promjena (upisivanje u zajednički dijeljeni blok podataka) ima za posljedicu prijenos tog podatka zajedničkom sabirnicom.

Primjer:
Osnovna zamisao dviju tehnika *snooping* protokola u multiprocesorskom sustavu sa središnjom dijeljenom memorijom.

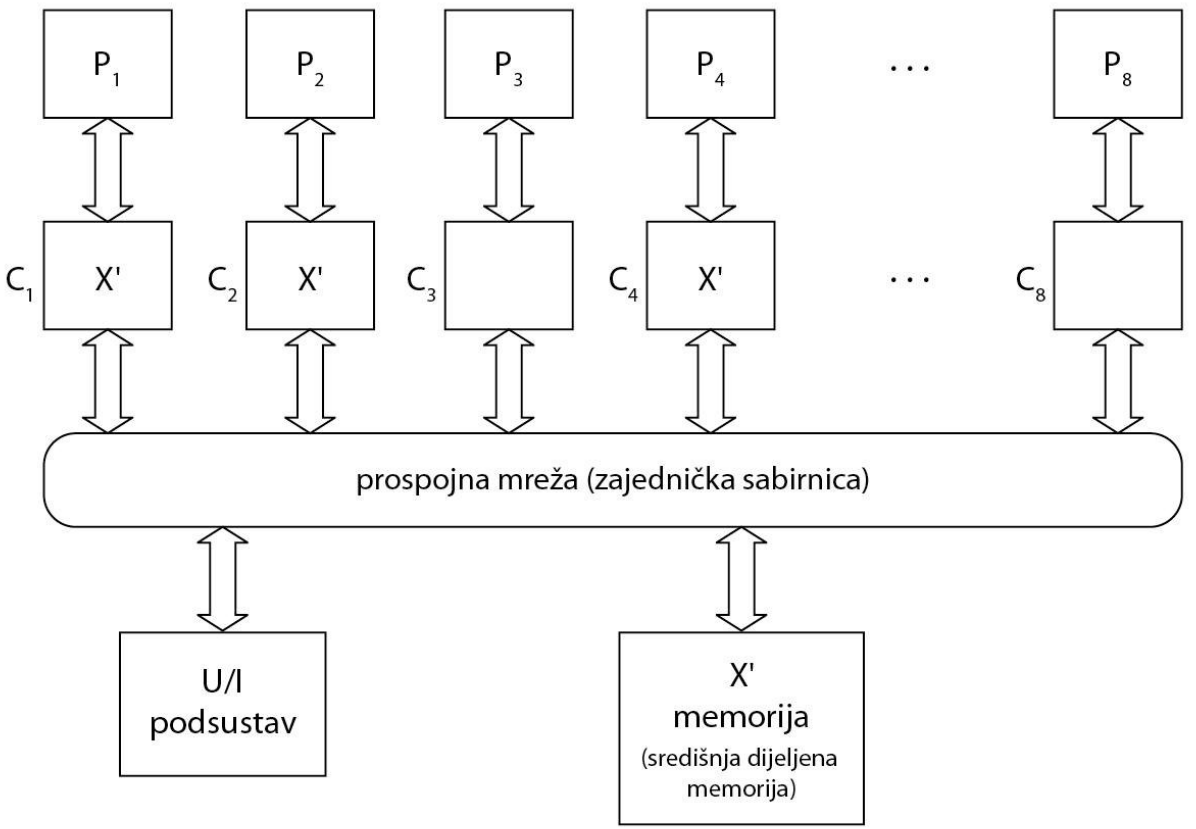
Pretpostavimo da imamo u sustavu osam procesora P_1, P_2, ... , P_8 koji imaju svoje priručne memorije C_1, C_2, ... , C_8 te da se kao prospojna mreža koristi zajednička sabirnica. Pretpostavimo da se dijeljeni blok podataka X nalazi u priručnim memorijama procesora P_1, P_2, i P_4, tj. u C_1, C_2 i C_4

Pretpostavimo da je procesor P_1 promijenio podatak u bloku X (promijenjeni sadržaj bloka X označit ćemo s X').
- Stanje multiprocesorskog sustava nakon operacije koja odgovara tehnici "piši i proglasi nevažećim“:



I - nevažeći (engl. *invalidate*).

- Stanje multiprocesorskog sustava nakon operacije koja odgovara snooping protokolu "piši i obnovi" . Vidimo da su obnovljeni dijeljeni blokovi X u X' u priručnim memorijama C_2 i C_4 i u središnjoj dijeljenoj memoriji



- jedan od poznatijih *snooping* protokola koherencije priručne memorije MESI (engl. *modified-exclusive-shared-invalid*) koji se djelotvorno koristi kako u jednoprocesorskim sustavima, tako i u multiprocesorskim sustavima.

7. Sinkronizacija procesa i dretvi

• u paralelnim procesnim okruženjima dva ili više konkurentna procesa (ili dretve) trebaju međusobno komunicirati (na primjer, tijekom izvođenja *proces 2* u nekom trenutku treba podatke koje generira *proces 1*)

• Komunikacija između procesa temelji se na osnovnom aksiomu da se ne mogu predvidjeti ili pretpostaviti relativne brzine procesa. To znači da bi se ostvarila komunikacija ili izmjena podataka između *procesa 1* i *procesa 2* u točno definiranoj točki, oba procesa moraju biti sinkronizirana.

- jedan važan detalj – sinkronizacija mora biti ugrađena programski u procese (ili dretve), a ne ostvarena procesorima, npr. nekim satnim mehanizmom – signalom vremenskog vođenja.

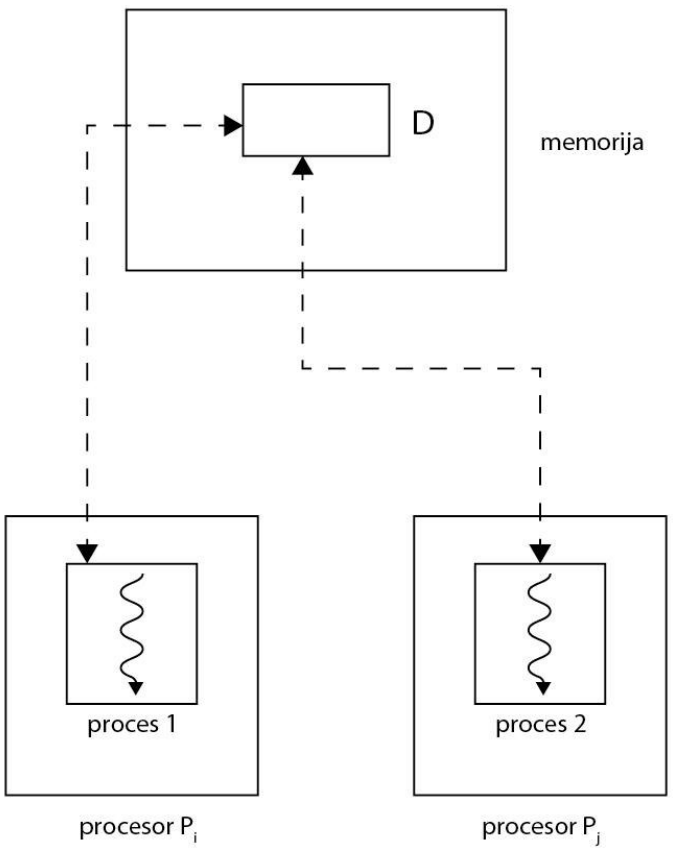
Ovisno o tome jesu li procesi ili dretve takve da se međusobno natječu ili konkuriraju (na primjer, međusobno se natječu za dobivanje nekog resursa), ili su pak kooperativni, tj. međusobno surađuju, razlikujemo dvije vrste sinkronizacije:

i) međusobno isključivanje (engl. *mutual exclusion*),

ii) uvjetovana sinkronizacija (engl. *condition synchronization*).

-Dva procesora P_i i P_j i dva nezavisna procesa *proces 1* i *proces 2* koji se paralelno izvršavaju.

- dva procesa međusobno natječu za neki resurs



U nekoj točki izvršavanja oba procesa, *proces 1* i *proces 2*, zahtijevaju modifikaciju zajedničke strukture podataka D koja se nalazi u središnjoj dijeljenoj memoriji.

-Pretpostavimo da programski segmenti *procesa 1* i *procesa 2* izgledaju ovako:

proces 1:
ld r1, D
addi r1, 1
st r1, D
...

proces 2:
ld r2, D
addi r2, 2
st r2, D
...

Budući da su oba procesa konkurentna i da nemamo pretpostavke o njihovim relativnim brzinama, može se dogoditi da se instrukcije gornjih dvaju programskih odsječaka međusobno *isprepliću* što vodi nepredvidljivom rezultatu:
- ako pretpostavimo da je početna vrijednost za D bila 0, onda vrijednost D-a nakon *procesa 1* i *procesa 2* kada se oba izvrše može biti 1, 2 ili 3, ovisno o relativnim vremenima u kojima se instrukcije izvode.

Da bi se ta neodređenost izbjegla, odnosno da bismo imali jamstvo da će vrijednost za D biti 3 kadgod se procesi 1 i 2 izvrše, zahtijeva se da gornji programski odsječci budu *nedjeljivi,* što znači da pristupaju i modificiraju D na međusobno isključivi način i promatraju se kao *kritični programski odsječci.*

- mehanizam *zaključavanja* (engl. *locking*), i to na sljedeći način: Pretpostavimo da smo strukturi podataka D pridružili "ključ" *k* i dvije operacije *lock(k)* i *unlock(k)* kojima se *k* postavlja u 1 ("zaključana brava"), odnosno 0 ("otključana brava"). Proces može izvesti operaciju *lock(k)* samo kada je *k* = 0, odnosno kada je brava otključana. Rezultat operacije *lock* je postavljanje *k* u 1 i zato onemogućavanje ostalih procesa da izvedu operaciju *lock(k).*

proces 1:

```
...
lock (k)
ld r1, D
addi r1, 1
st r1, D
unlock(k)
...
```

proces 2:

```
...
lock(k)
ld r2, D
addi r2, 2
st r2, D
unlock(k)
```

Pretpostavka: inicijalna vrijednost k = 0

Problem *uvjetovane sinkronizacije*:
-procesi *proces 1* i *proces 2* kooperativni procesi, odnosno procesi koji međusobno surađuju. Pretpostavimo da *proces 1* generira niz rezultata i pohranjuje ih slijedno u D, a *proces 2* slijedno uzima te podatke iz niza i konzumira ih.
Radi jednostavnosti, pretpostavimo da D može pohranjivati samo jednu vrijednost tako da se mora osigurati sljedeće:
procesi *proces 1* i *proces 2* kooperativni procesi, odnosno procesi koji međusobno surađuju.

Pretpostavimo da *proces 1* generira niz rezultata i pohranjuje ih slijedno u D, a *proces 2* slijedno uzima te podatke iz niza i konzumira ih. Radi jednostavnosti, pretpostavimo da D može pohranjivati samo jednu vrijednost tako da se mora osigurati sljedeće:

i) kad *proces 1* generira vrijednost V_i i smješta je u D, on čeka sve dok *proces 2* ne konzumira vrijednost V_i. Tek će poslije toga pohraniti sljedeću vrijednost V_{i+1} u D;

ii) nakon što *proces 2* konzumira vrijednost V_i iz D, on čeka sve dok *proces 1* ne generira sljedeću vrijednost V_{i+1}. Tek poslije toga ponovo čita iz D.

proces 1:

...
lock (k_1)
generiraj vrijednost i pohrani je u
D
unlock(k_2)
...

proces 2:

...
lock (k_2)
konzumiraj vrijednost iz D
unlock(k_1)
...

Pretpostavka: početna vrijednosti k_1 jednaka 0, a početna vrijednost k_2 jednaka 1

Iz operacijskih sustava znamo da postoje dva osnovna pristupa koji se primjenjuju za izvedbu mehanizma za obje vrste sinkronizacije: sinkronizacija uporabom *dijeljenih varijabli* (engl. *shared variables*) uporabom *test-and-set* primitiva, *fetch-and-add* primitiva, semafora ili na temelju *prosljeđivanja poruka*.

Prvi mehanizam sinkronizacije pogodan je i zato prevladava u multiprocesorskim sustavima sa središnjom dijeljenom memorijom, dok se drugi koristi u multiprocesorskim sustavima s porazdijeljenom memorijom.

8. Višejezgreni procesori

- Višedretvenost (engl. *multithreading*) podrazumijeva da se više dretvi izvodi u jednom procesoru tako da se izvođenje dretvi međusobno isprepliće, odnosno dretve se naizmjenično izvode u dodijeljenim funkcijskim jedinicama procesora uz nužno prospajanje konteksta sadržanog u tablici dretvi, i to nakon svake izmjene dretve.

Dva su glavna pristupa višedretvenosti:

- finozrnata višedretvenost (engl. *fine-grained multithreading*),
- grubozrnata višedretvenost (engl. *coarse-grained multithreading*).

Finozrnata višedretvenost podrazumijeva prospajanje dretvi nakon svake instrukcije.
-Procesori s takvom značajkom nazivaju se još i "bačvasti", odnosno *barrel* procesori

Te su instrukcije međusobno nezavisne jer pripadaju različitim dretvama tako da se protočna struktura djelotvorno iskorištava.
- povećani broj promašaja priručne memorije zbog narušavanja lokalnosti programa. Obično se izvođenje dretvi u tom slučaju temelji na kružnom prioritetu (engl. *round-robin*) uz "preskakanje" dretvi koje su u stanju zastoja.

Višedretveni procesori imaju sklopovski podržano brzo prospajanje konteksta dretvi (tzv. *hardware based fast context switching*) koje se temelji na tome da svaka dretva ima dodijeljene fizičke registre za pohranu konteksta dretvi

Primjer:

Višedretveni procesor Tera podržava 128 dretvi, pri čemu je svakoj dretvi dodijeljen 41 64-bitni registar u procesoru. Jezgre, osim sklopovski podržanog brzog prospajanja konteksta, imaju i dinamičku izvedbu preimenovanja registara kojim se rješavaju hazardi WAW i WAR koji se nazivaju još i *lažne zavisnosti*

Grubozrnata višedretvenost predviđa prospajanje dretvi samo onda kada nastupa dulji zastoj u tekućoj dretvi (npr. promašaj u priručnoj memoriji). Za kraće zastoje, na primjer one izazvane u instrukcijskoj protočnoj strukturi uslijed hazarda, ne predviđa se izmjena dretvi te je to jedan od glavnih nedostataka grubozrnate višedretvenosti.

- Razlog da se ne koristi prospajanje dretvi kod zastoja izazvanih u protočnoj strukturi leži u procjeni cijene i količine posla za pražnjenje protočne strukture da bi se u nju mogao uputiti instrukcijski tok druge dretve.
- Izmjena dretvi i prospajanje njihova konteksta može se događati i pri svakoj *load* instrukciji (neovisno o tome je li se dogodio promašaj) ili nakon programskog odsječka (bloka instrukcija) koji pripadaju jednoj dretvi.

Grubozrnata i finozrnata višedretvenost može se kombinirati s paralelizmom na razini instrukcija ILP, ali i sa superskalarnosti (višestrukim protočnim strukturama u jednom procesoru). Tri su procesorske konfiguracije moguće u tom slučaju:

- superskalarnost s grubozrnatom višedretvenosti,
- superskalarnost s finozrnatom višedretvenosti,
- superskalarnost sa simultanom višedretvenosti.

Superskalarni procesori s gubozrnatom višedretvenosti izvode istodobno veći broj instrukcija koje pripadaju istoj dretvi sve do trenutka kada nastupi dulji zastoj, u tom se trenutku prospaja kontekst dretvi i nastavlja se s izvođenjem druge dretve.

• Superskalarni procesori s finozrnatom višedretvenosti isprepliću dretve i na taj način eliminiraju možebitne zastoje u izdavanju instrukcija. Budući da samo jedna dretva izdaje instrukcije tijekom periode signala vremenskog vođenja, još uvijek postoje ograničenja u paralelizmu na razini instrukcija. Višejezgreni procesori tvrtke Sun nazvani T1(Niagara 1) i T2 (Niagara 2) koriste finozrnatu višedretvenost.

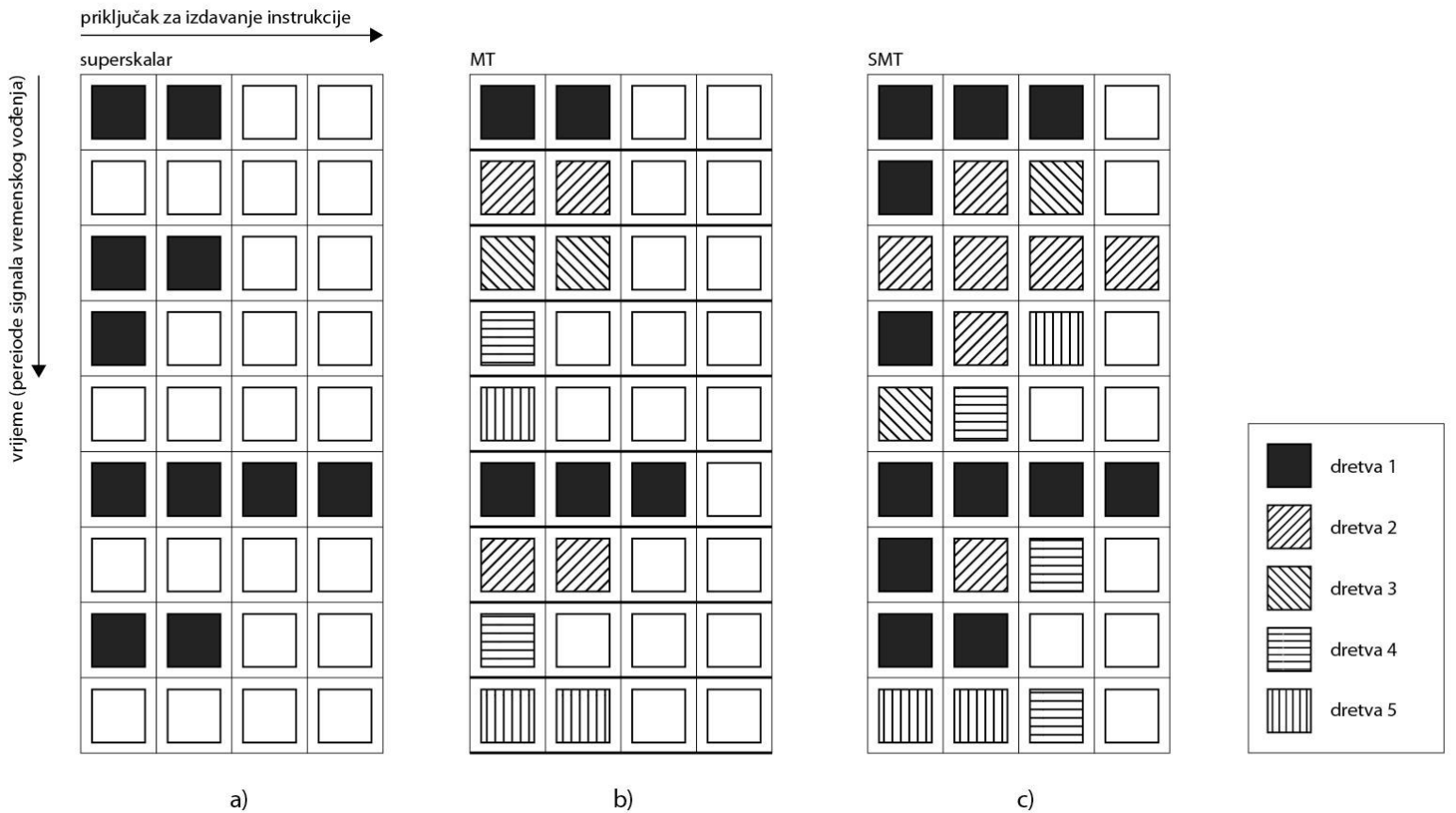
Simultana višedretvenost SMT (engl. *simultaneous multithreading*) zasniva se na činjenici da suvremeni (superskalarni) procesori imaju paralelizam na razini funkcijskih jedinica veći od onog koji jedna dretva može iskoristiti.

-dinamičkim raspoređivanjem izdaje *istodobno* više instrukcija iz *nezavisnih* dretvi u istoj periodi signala vremenskog vođenja.

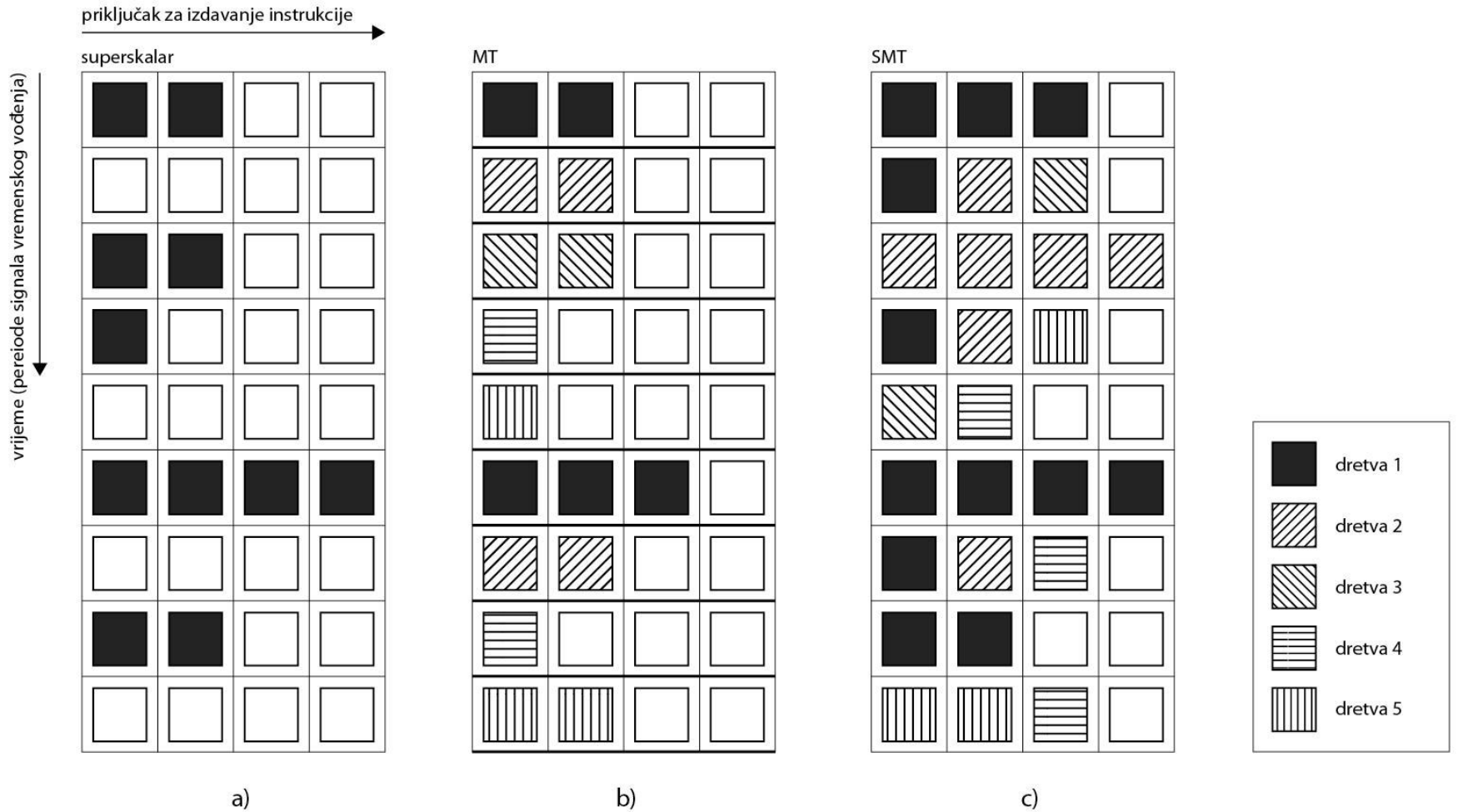
U SMT-u se iskorištava paralelizam na razini dretvi i paralelizam na instrukcijskoj razini u kombinaciji s izdavanjem više instrukcija u jednoj periodi signala vremenskog vođenja.

Primjeri procesora koji se temelje na SMT zamislima su Intelovi višejezgreni procesori Nehalem i Pentium D te IBM-ovi višejezgreni procesori Power 5, Power 6 i Power 7.

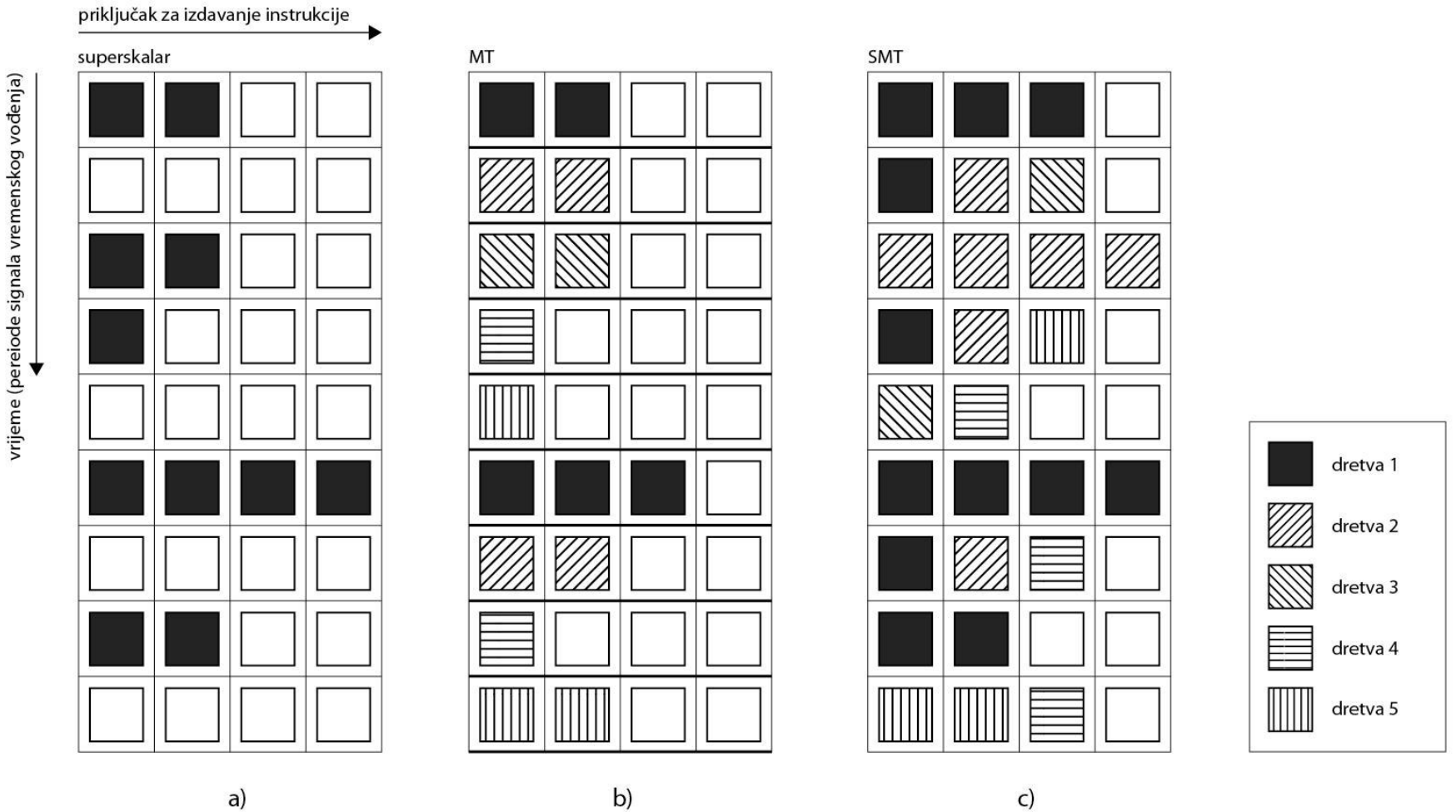
- razlika između superskalarnosti, višedretvenosti (MT) i simultane višedretvenosti (SMT), poslužit ćemo se jednostavnim modelom obrade koji su predložili S. J. Eggers i suradnici.



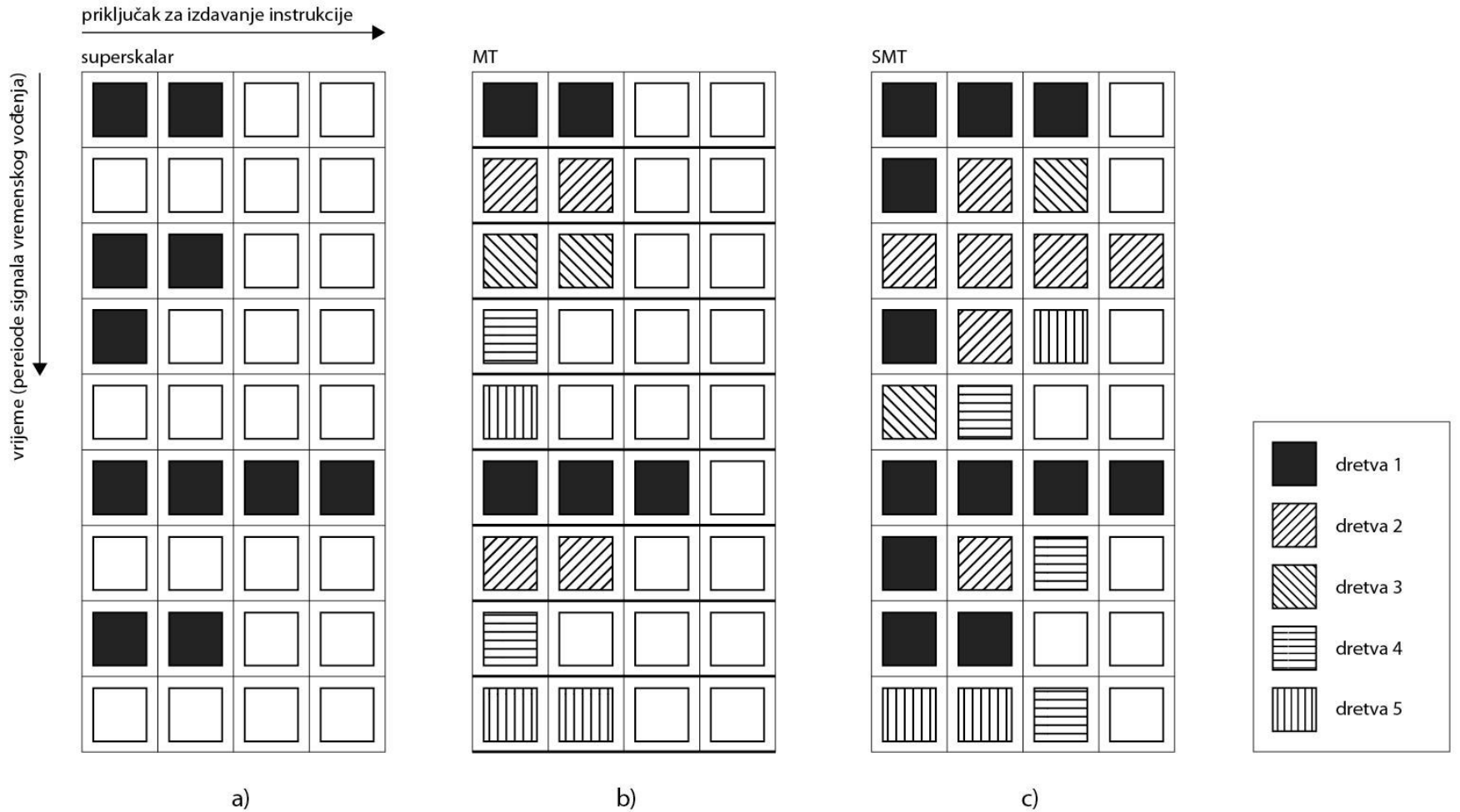
x os – priključci za izdavanje instrukcija
y os – vrijeme (periode signala vremenskog vođenja)



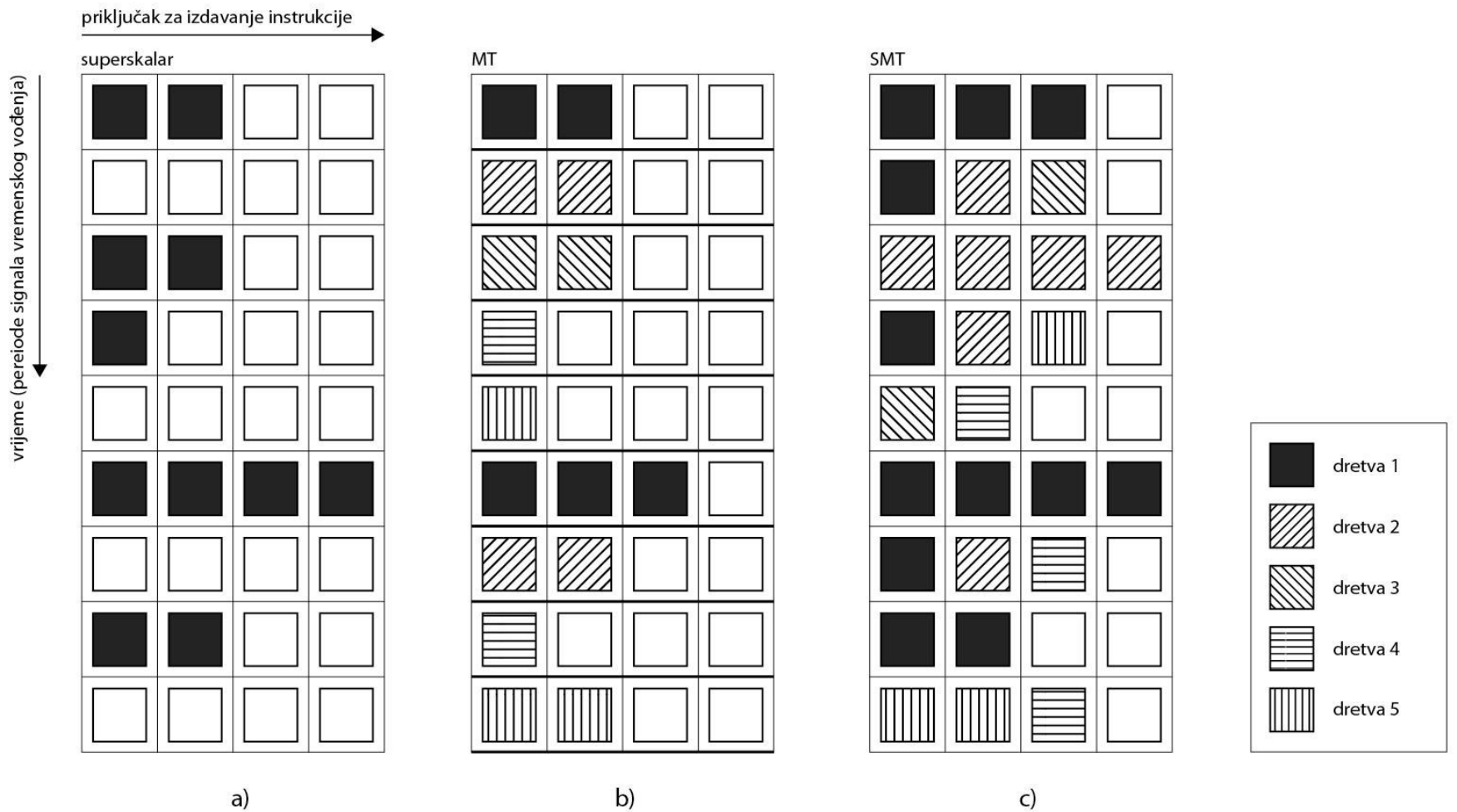
Nekorišteni priključci za izdavanje instrukcija mogu se promatrati kao "horizontalno neiskorišteni priključci" i kao "vertikalno neiskorišteni priključci"



Pod horizontalno neiskorištenim priključcima podrazumijevamo slučaj kada su u jednom retku jedan ili više priključaka neiskorišteni (ali ne svi!).



Vertikalno neiskorišteni priključci događaju se kada su tijekom jedne periode *svi* priključci neiskorišteni – to nastupa zbog dulje latencije (duljeg zastoja u izvođenju) instrukcije, npr. pristupa memoriji, kada je privremeno spriječeno daljnje izdavanje instrukcija.



Za SMT vidimo da u svakoj periodi signala vremenskog vođenja procesor "izabire" instrukcije za izvođenje iz svih dretvi.
- iskorištava se paralelizam na razini instrukcija izborom instrukcija iz bilo koje dretve.
- Nakon toga procesor dinamički raspoređuje resurse procesora između instrukcija i osigurava visoku razinu iskoristivosti sklopovskih resursa.
- Ako neka od dretvi ima visok stupanj paralelizma na razini instrukcija, onda se te instrukcije izvode i pritom je vrlo malo horizontalno neiskorištenih priključaka.

- Ako paralelizam na razini instrukcija za jednu dretvu nije dovoljno visok, onda se izabire još jedna dretva (ili više njih) s nižim stupnjem paralelizma koja će popuniti prazne horizontalne priključke. Na taj se način postiže uklanjanje i vertikalnih neiskorištenih priključaka i u velikoj mjeri horizontalnih neiskorištenih

Posebno zanimljiv pristup arhitekturi procesora je pristup koji se naziva *treća generacija DPL* (engl. *Data Parallel Processor*) u kojem se kombiniraju vektorske instrukcije, izdavanje instrukcija izvan redoslijeda i preimenovanje registara te simultano izvođenje više dretvi.
Procesori s tim značajkama nose oznaku SMV – *Simultaneous Multithreaded Vector* – simultano višedretveni vektorski procesori.

IBM POWER4 dvojezgreni procesor

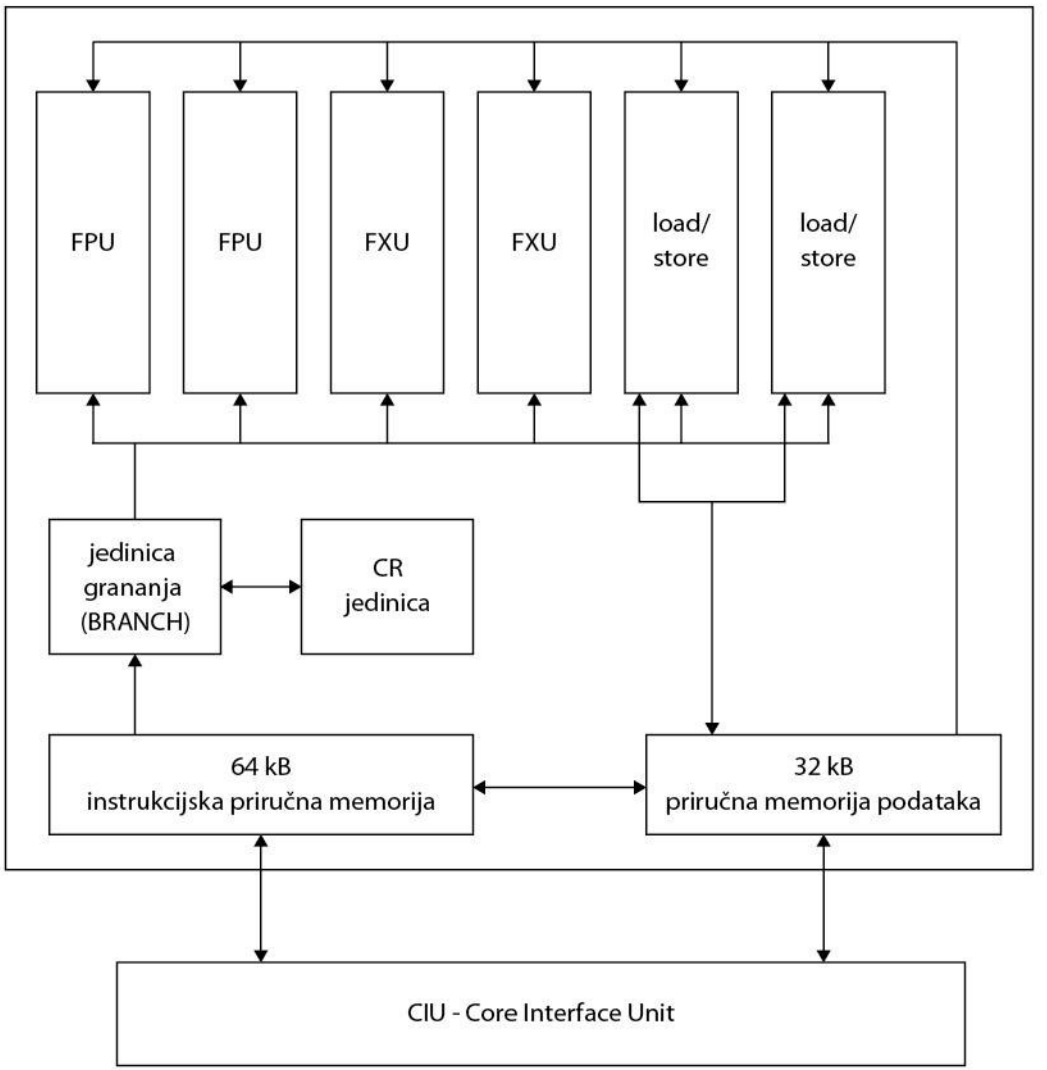
-Godine 2001. tvrtka IBM oblikovala je prvi komercijalni 64-bitni multiprocesorski sustav na čipu (višejezgreni procesor) opće namjene – POWER4
- dvije jezgre – dva superskalarna procesora na čipu
- dvojezgreni procesor namijenjen je iskorištavanju paralelizma na razini dretvi. Značajke POWER4 arhitekture su superskalarnost i izvođenje instrukcija izvan redoslijeda.

Jezgra:

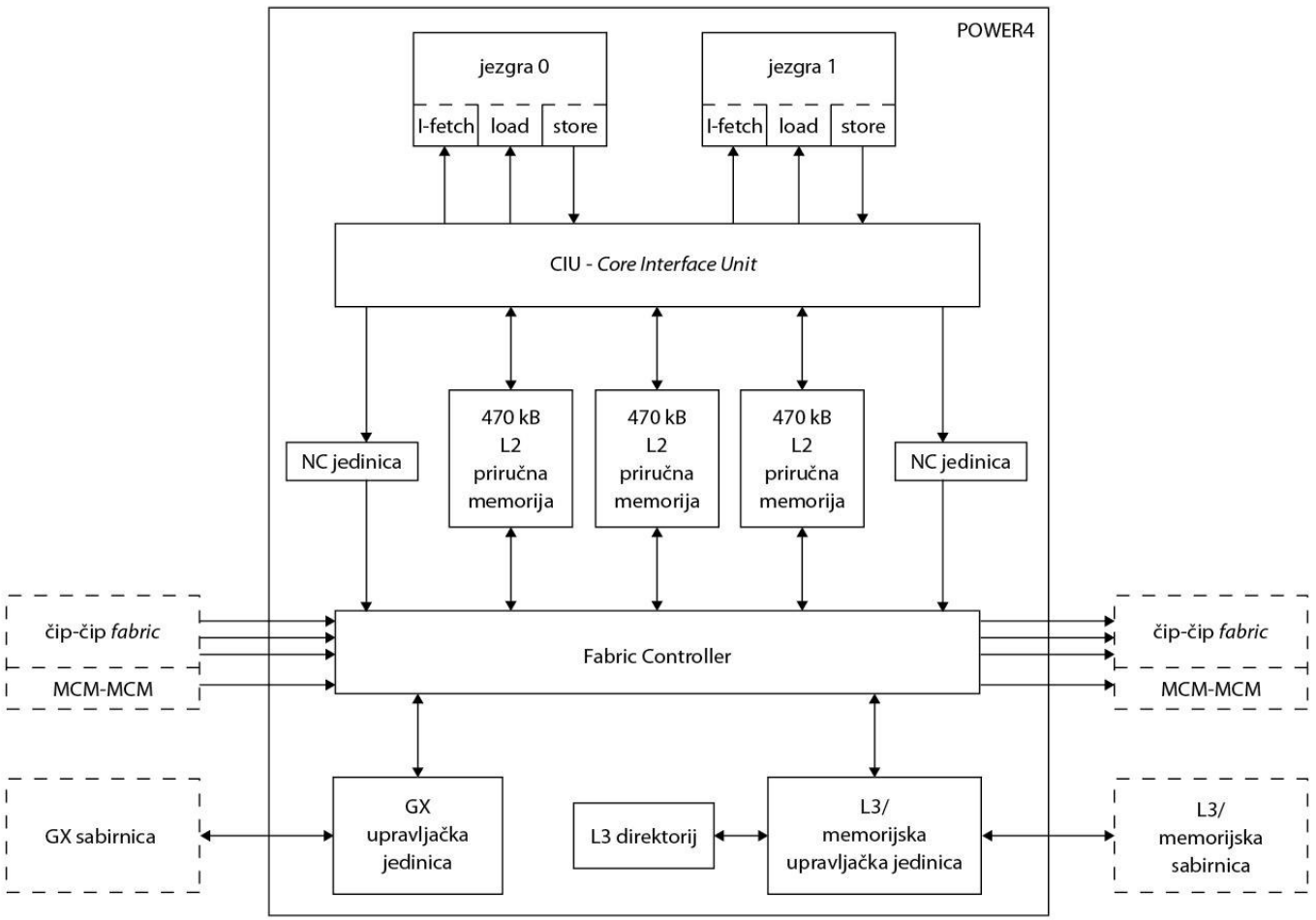
• Osam nezavisnih izvršnih jedinica:

- FPU
- FXU
- load/store
- jedinica grananja BR
- logička jedinica CR
- priručna memorija L1 kapaciteta 64 KB (I cache) / 32 KB (D cache)

Superskalarna jezgra može izvoditi do osam operacija u jednoj periodi signala vremenskog vođenja



- dvojezgreni procesor POWER4



Značajke

• dva superskalarna procesora na čipu

• dijeljena zajedničku priručnu memoriju druge razine (L2) (organizirana u tri bloka –svaki po 470 KB)

• direktorij za priručnu memoriju razine L3 (kapaciteta 32 MB)

• dvije jedinice NC – *Non-Cacheable* odgovorne su za rukovanje operacijama koje ne koriste priručne memorije jezgara

• *Fabric Controller* glavni je upravljač za sabirničku strukturu i komunikaciju s L2/L3 upravljačima priručne memorije te za komunikaciju više POWER4 procesora

• sabirnički upravljač GX upravlja komunikacijom s U/I jedinicama i podržava dvije tzv. GX sabirnice širine 4 bajta.

• POWER4 može se koristiti i u konfiguraciji s više čipova POWER4 uporabom MCM – *Multi-Chip Module* tako da se mogu kombinirati četiri POWER4 čipa u jednom pakovanju.

Tehnološke značajke:

-Višejezgreni procesor POWER4 ostvaren je u 180 nm VLSI tehnologiji SOI (Silicon on Insulator) CMOS i ima 174 milijuna tranzistora, radi na frekvenciji od 1.3 GHz pri čemu mu je potrošak snage oko 115 W.
- Poboljšana verzija procesora POWER4+ radi na frekvenciji 1.9 GHz i ima 184 milijuna tranzistora na čipu površine 267 mm^2 (tehnologija 0.13 μm SOI CMOS).

Ultra SPARC T1 i T2

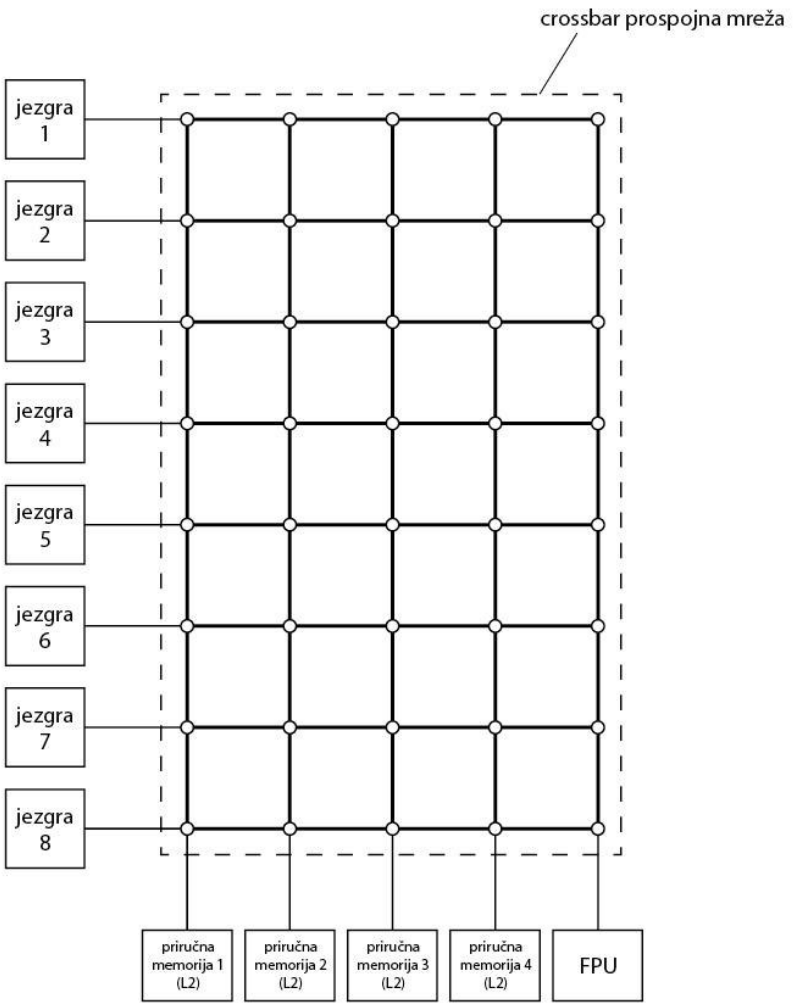
-2005. tvrtka Sun najavila je višejezgreni (osam jezgri) procesor Ultra SPARC T1 (Niagara)

- Jedna od arhitektonskih značajki mu je da je orijentiran na djelotvorno iskorištavanje paralelizma na razini dretvi (višedretvenost), više no na razini instrukcija.
- Jezgra ima protočnu strukturu koja se sastoji od šest protočnih segmenata (pribavi, izbor dretve, dekodiraj, izvrši, pristup memoriji, upiši natrag) – klasična RISC protočna struktura
- Zbog relativne jednostavnosti protočne strukture instrukcije grananja i instrukcije *load* prouzrokuju kašnjenje od tri periode signala vremenskog vođenja (!?), međutim, to se kašnjenje "prekriva" izvođenjem instrukcija drugih dretvi.

Jezgra izdaje jednu instrukciju i upućuje je u protočnu strukturu – ne koristi se tehnika izdavanja i završavanja instrukcija izvan redoslijeda što u velikoj mjeri pojednostavnjuje izvedbu procesora

- Svaka od osam jezgri podržava istodobno četiri dretve, tako da, zapravo, višejezgreni procesor T1 kombinira višedretvenost na razini procesora (jezgre) i hiperdretvenost – istodobno izvođenje većeg broja dretvi u osam jezgri.
- T1 rabi finozrnatu višedretvenost, odnosno da ima značajke "bačvastog" procesora (engl. *barrel CPU*) - svaka jezgra u svakoj periodi signala vremenskog vođenja prospaja dretve.

Organizacija višejezgrenog procesora Sun Ultra SPARC T1 (Niagara 1)



Arhitektonske značajke višejezgrenog procesora T1:

•finozrnata višedretvenost na razini instrukcije,

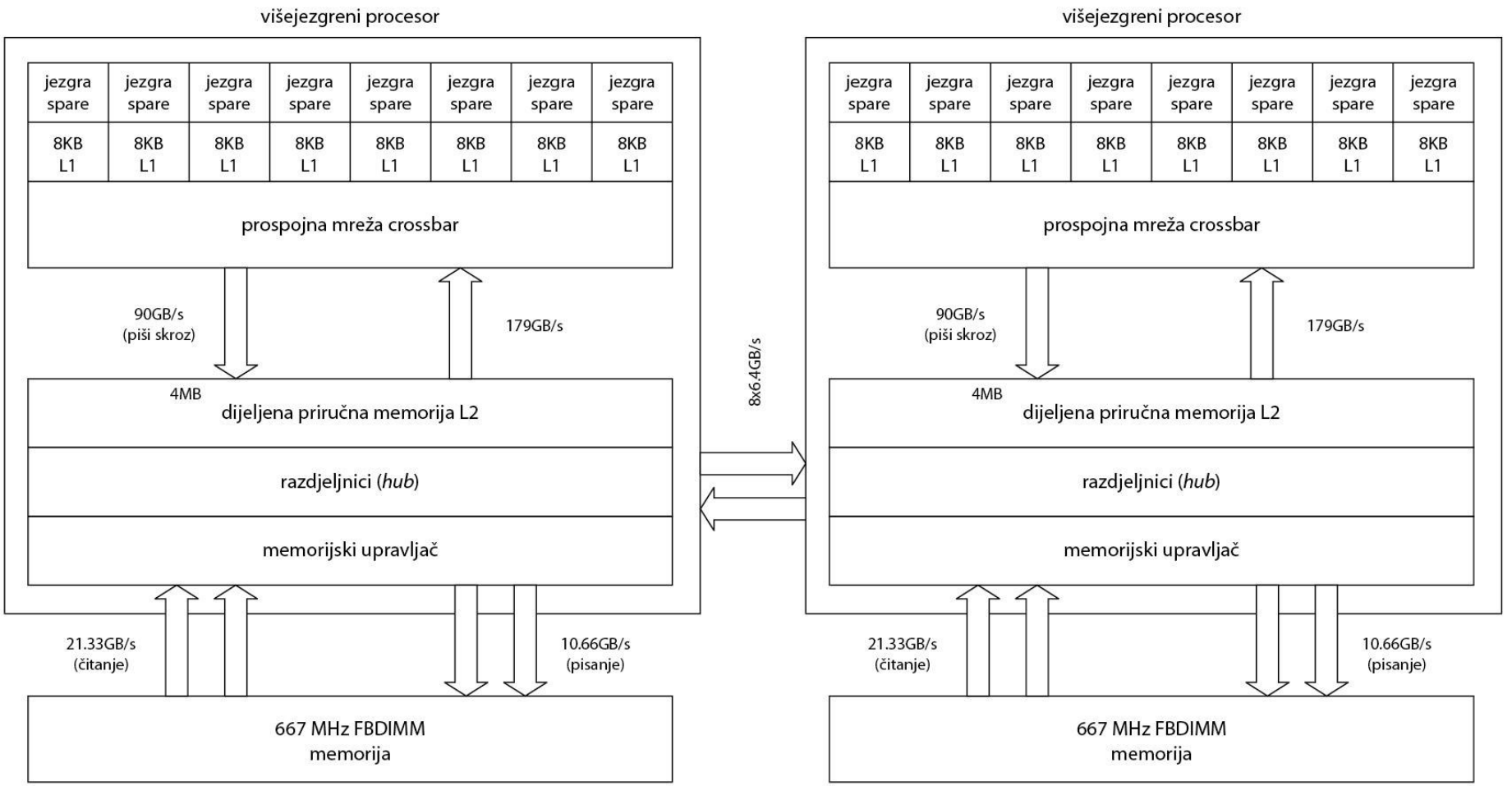
- - osam jezgri,
- - četiri dretve po jezgri,
- - jedna FPU,
- - crossbar prospojna mreža,
- - relativno jednostavna protočna struktura sastavljena od šest poput RISC protočnih segmenata,
- - jezgra ima priručne memorije razine L1: 16 KB instrukcijske memorije i 8 KB priručne memorije podataka (blok ili linija duljine je 64 bajta),
- - četiri odvojene priručne memorije razine L2 svaka po 750 KB

Tehnološke značajke:
- izveden je u 90 nm CMOS VLSI tehnologiji,
- ima 279 milijuna tranzistora ostvarenih na silicijskom čipu površine 379 mm^2,
- potrošak snage mu je 79 W,
- maksimalna frekvencija signala vremenskog vođenja je 1.2 GHz.
- Vršna performansa višejezgrenog procesora T1 iznosi oko 9600 MIPS-a i oko 1200 MFLOPS-a

U listopadu 2007. na tržištu se pojavio Ultra SPARC T2 (kodnog imena Niagara 2) koji je poboljšana verzija prethodnika T1.

- Procesor T2 je realiziran u 65 nm VLSI tehnologiji, ima osam jezgri i svaka se jezgra može istodobno koristiti s osam dretvi – 64 dretvi istodobno
- Svaka je jezgra ostvarena kao protočna jedinica s osam protočnih segmenata i ima i dvije aritmetičko-logičke jedinice koje dijele grupe od po četiri dretve
-Umjesto jedne FPU jedinice, sada ih je u T2 osam – po jedna FPU za svaku jezgru.
- Kapacitet priručne memorije L2 4 MB
- Frekvencija 1.6 GHz

Konfiguracija 2 x 8-jezgrena procesora SunUltra SPARC T2 (Niagara 2)

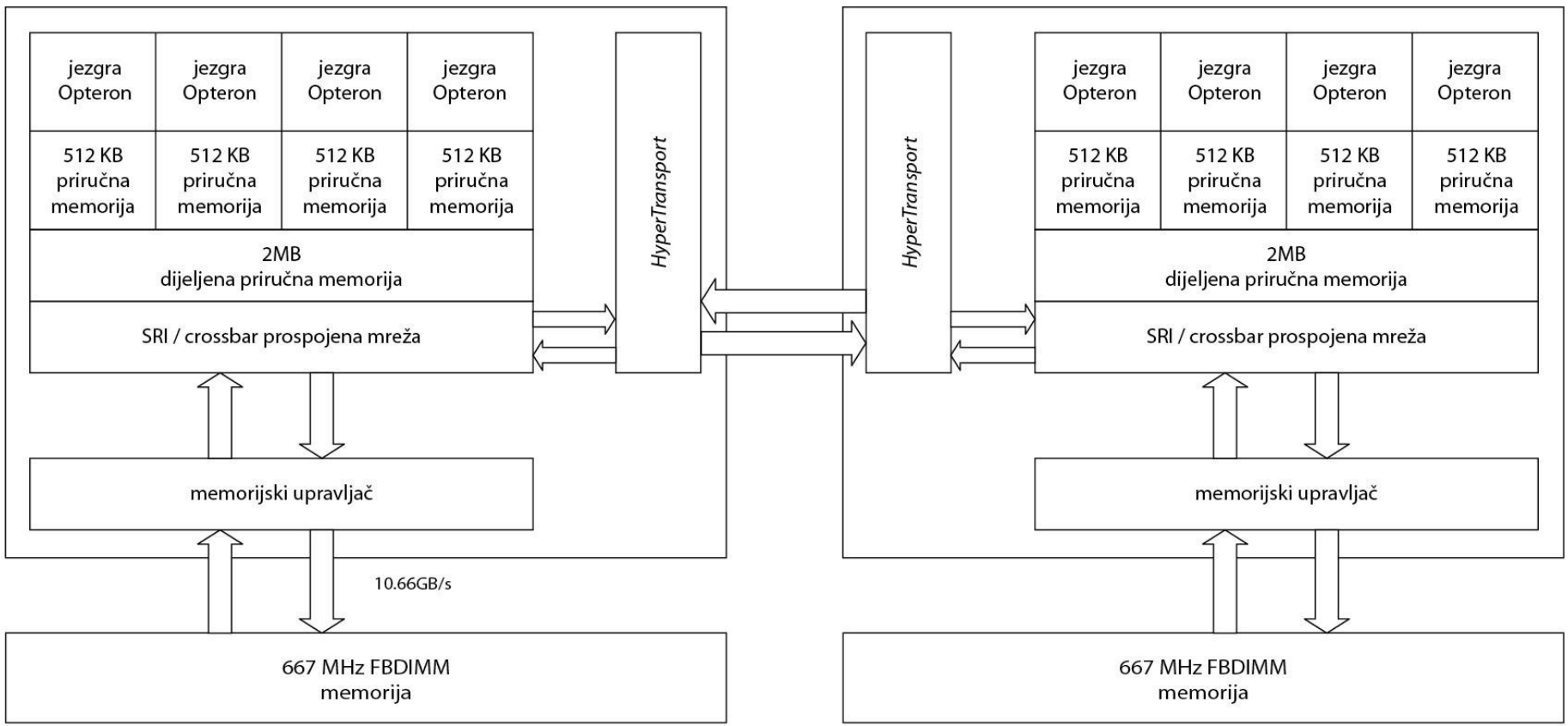


FBDIMM - Fully-biffered DIMM

AMD Opteron X4 2356

• Višejezgreni procesor Opteron X4 2356 tvrtke AMD jedan je iz brojne porodice 4-jezgrenih procesora serija 1300, 2300, 4100 i 8300, različitih kodnih imena – od Budapest (serija 1300) do Shangai (serija 8300)
• Opteron X4 2356, kodnog imena Barcelona, ima četiri jezgre po čipu, a koristi se kao multiprocesorski sustav koji ima dva podnožja (engl. *CPU slot*, *CPU socket*) tako da oblikuje sustav s 8 jezgri ("broj jezgri po podnožju")

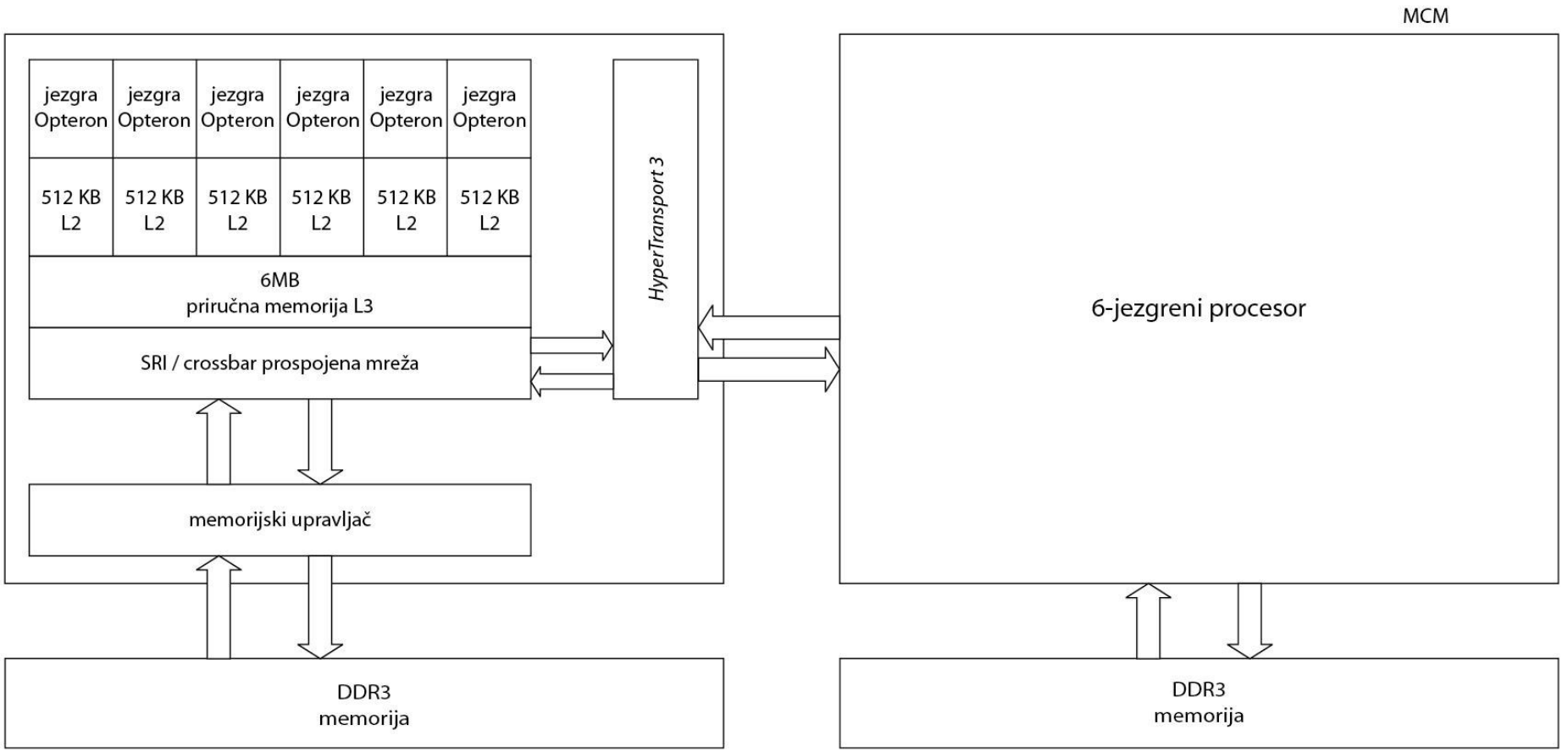
Organizacija višejezgrenog procesora Opteron X4 2356



Značajke:
-Jezgra je složeni procesor koji ima 6-segmentnu instrukcijsku protočnu strukturu,
- 72 fizička registra,
- međuspremnik za cjelobrojne i FP operacije u repu čekanja (engl. *operation queue*),
- međuspremnik za *load/store* operacije u repu čekanja,
- tri ALU za cjelobrojne operacije,
- tri FPU (zbrajalo, množilo, mješovite operacije),
- instrukcijska priručna memorija (razina L1, 64 KB) i priručnu memoriju podataka (razina L1, 64 KB).
- Svaka od jezgri ima i dodatnu 512 KB lokalnu priručnu memoriju, tzv. *victim cache* koja sadržava blokove podataka koji su "izbačeni" iz priručne memorije tijekom zamjene blokova

Svaka od jezgri podržava osam dretvi.
-Na čipu je ostvareno posebno tzv. sistemsko sučelje SRI (*System Request Interface*) i prospojna mreža crossbar koja ostvaruje vezu između *quasi-victim* priručne memorije, memorijskog upravljača i sučelja za komunikaciju s višejezgrenim procesorom (*HyperTransport link*) koji se nalazi u drugom podnožju.
- za povezivanje višejezgrenih procesora "četiri jezgre po podnožju" u 8-jezgrenu konfiguraciju AMD Opteron koristi LGA (engl. *land grid array*) podnožje tipa F (*Socket F*) koje ima preko 1200 priključaka.
- Višejezgreni procesor radi s frekvencijom signala vremenskog vođenja od 2.3 GHz i postiže vršnu performansu od 74 GFLOP/s

Krajem ovog desetljeća tvrtka AMD razvila je 12-jezgreni procesor Opteron kodnog imena "Magny Cours"



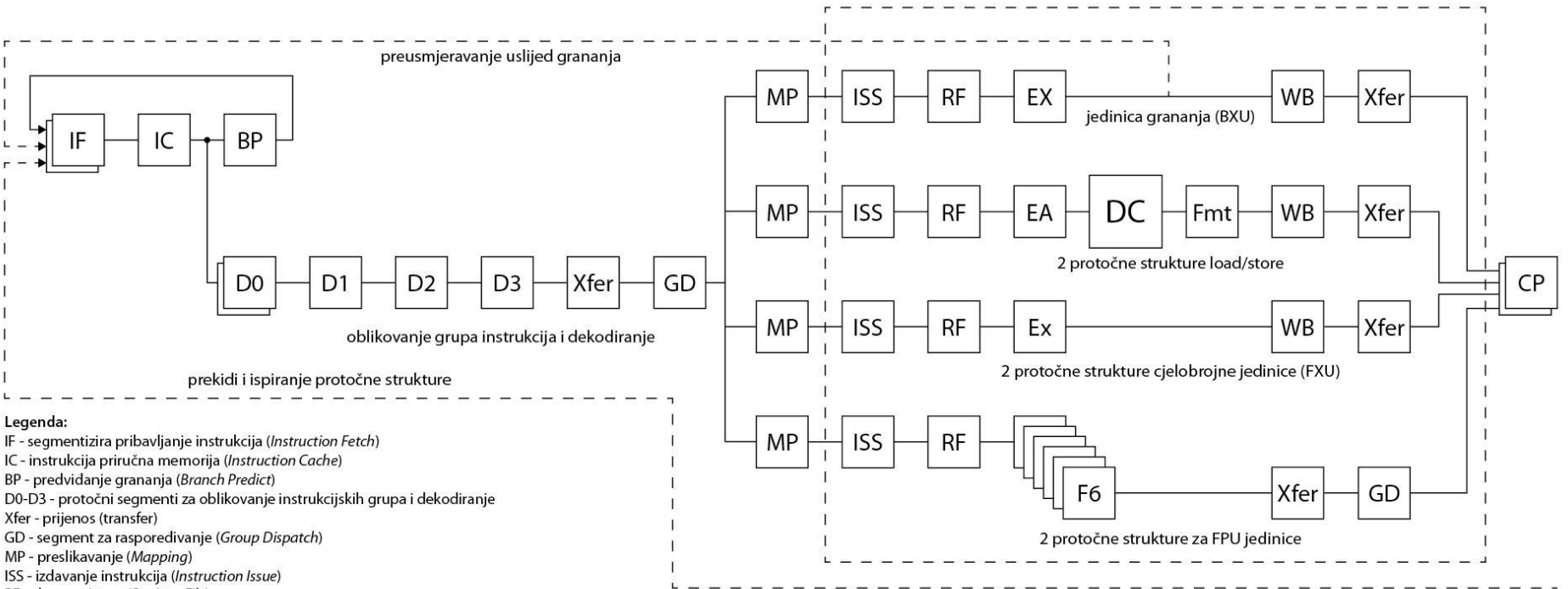
-Svaka jezgra Opteron je superskalarni procesor (izdaje do tri instrukcije u jednoj periodi signala vremenskog vođenja) i koristi izdavanje i završavanje instrukcija izvan redoslijeda
- Jezgra pribavlja (iz instrukcijske priručne memorije) i dekodira do tri x86-64 instrukcije tijekom svake periode signala vremenskog vođenja.
- Instrukcije promjenjive duljine pakiraju se u tzv. makrooperacije (*mops*) čvrste duljine.
- Svaka od Opteron jezgri ima instrukcijsku priručnu memoriju i priručnu memoriju podataka (svaka kapaciteta 64 KB; razina L1).
- Svakoj je jezgri pridružena i priručna memorija razine L2 kapaciteta 512 KB (nalazi se na procesorskom čipu) te zajednička dijeljena priručna memorija razine L3 (6 MB)

IMB Power5 dvojezgreni SMT procesor, Power6 i Power7

-POWER5 je dvojezgreni SMT procesor koji podržava 64-bitnu PowerPC arhitekturu.
- Na jednom čipu su ostvarena dvije identične jezgre pri čemu svaka od njih podržava jednu fizičku dretvu i dvije logičke dretve,
- pored instrukcijske priručne memorije razine L1 kapaciteta 32 KB i priručne memorije podataka razine L1 kapaciteta 32 KB, procesor ima i priručnu memoriju razine L2 kapaciteta 1.875 MB (koju dijele obje jezgre) i direktorij za priručnu memoriju razine L3.
- Direktorij je namijenjen priručnoj memoriji kapaciteta 36 MB koja je realizirana izvan procesorskog čipa.

- Procesor POWER5 koristi crossbar prospojnu mrežu.
- Jezgra POWER5 podržava dva načina rada: simultanu višedretvenost (SMT način rada) u kojem se izdaje više instrukcija iz različitih dretvi u jednoj periodi signala vremenskog vođenja, te jednodretveni način rada ST (engl. *single-threaded mode*).
- U oba slučaja (SMT i ST) pribavlja se osam instrukcija iz iste dretve, međutim, njihovo izdavanje će biti, ako je riječ o SMT, takvo da su iz različitih dretvi.

Instrukcijska protočna struktura za jezgru SMT procesora POWER5



Legenda:
IF - segmentizira pribavljanje instrukcija (*Instruction Fetch*)
IC - instrukcija priručna memorija (*Instruction Cache*)
BP - predvidanje grananja (*Branch Predict*)
D0-D3 - protočni segmenti za oblikovanje instrukcijskih grupa i dekodiranje
Xfer - prijenos (transfer)
GD - segment za rasporedivanje (*Group Dispatch*)
MP - preslikavanje (*Mapping*)
ISS - izdavanje instrukcija (*Instruction Issue*)
RF - skup registara (*Register File*)
EX - izvrši (*Execute*)
EA - računanje efektivne adrese (*Address Computation*)
DC - priručna memorija podataka (*Data Cache*)
F6 - protočna struktura FP jednica
Fmt - formatiranje podataka
WB - upisivanje natrag (*Write Back*)
CP - grupa izvršenih instrukcija (*Group Commit*)

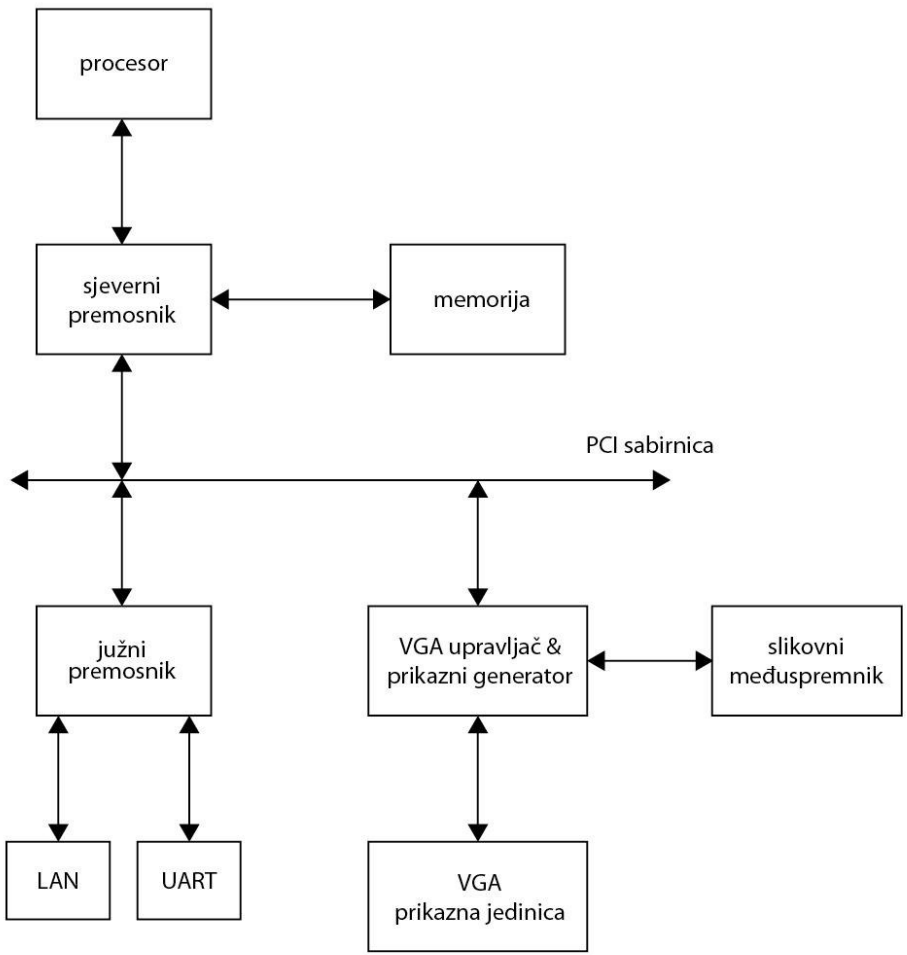
Višejezgreni procesor POWER5 radi s frekvencijom signala vremenskog vođenja od 1.9 GHz,
- ostvaren je u 130 nm VLSI tehnologiji na čipu površine 389 mm^2, ima oko 200 milijuna tranzistora
- potrošak snage od oko 130 W.

Poboljšane verzija POWER5+ (2005. i 2006. godina) rade s frekvencijom signala vremenskog vođenja do 2.3 GHz i imaju priručnu memoriju razine L3 ostvarene na procesorskom čipu. Power5+ realiziran je u 90 nm tehnologiji na komadiću silicija površine 389 mm^2.

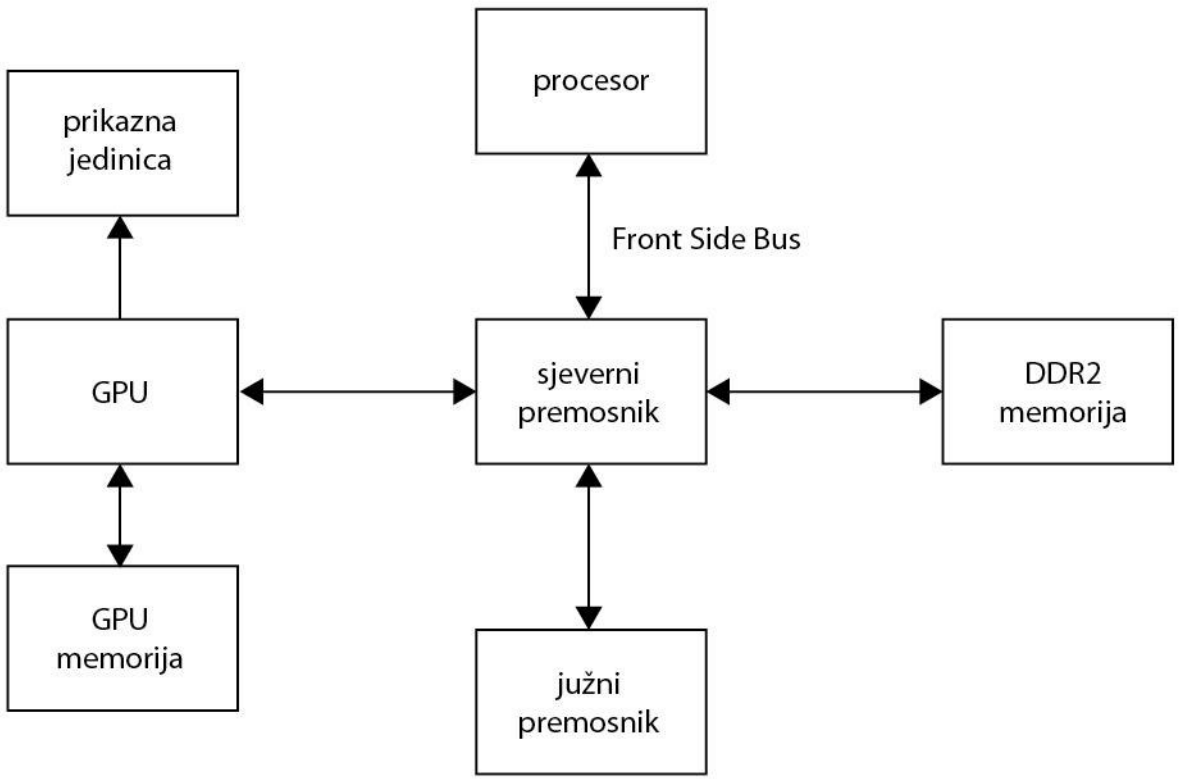
- Poboljšana verzija SMT procesora, također s dvije jezgre, je POWER6 procesor (2007.) koji radi na frekvenciji 5.0 GHz. Svaka jezgra ima dvije cjelobrojne aritmetičke jedinice, dvije jedinice za operacije brojevima s pomičnim zarezom FPU, jedinicu za izvođenje SIMD instrukcija (*AltiVec unit*) i jednu FP jedinicu za operacije decimalnim brojevima (engl. *decimal floating-point unit).*
- Procesor POWER7 (2010.) je prvi IBM-ov 8-jezgreni SMT procesor i predstavlja procesor za poslužitelje nove generacije. (Svaka od jezgri podržava četiri istodobne višedretvene operacije, dakle ukupno 32 istodobno izvršljive dretve.)
- Jezgra ima 12 izvršnih protočnih jedinica
- Procesor POWER7 ostvaren je u 45 nm VLSI tehnologiji, ima 1.2 milijarde (!) tranzistora na čipu površine 567 mm^2.

9. Grafički procesori (grafičke procesne jedinice)

Uobičajena konfiguracija osobnog računala početkom devedesetih godina:



Konfiguracija osobnog računala u prvom desetljeću ovog stoljeća:



Veza GPU – sjeverni premosnik –PCI-express (PCIe)

- krajem devedesetih godina razvoj tehnologije VLSI omogućio je da se VGA upravljaču pridoda sklopovlje (tzv. sklopovski akceleratori) kojim se ubrzavaju neke funkcije definirane u *logičkoj grafičkoj protočnoj strukturi* (engl. *logical graphics pipeline*)
- Početkom 2000. počinje se koristiti naziv *grafički procesor* jer se u jednom čipu objedinjuju sve uobičajene funkcije potrebne za djelotvorni prikaz 2D i 3D grafike.
- Grafički procesor postaje čvrsto definirana funkcijska protočna struktura namijenjena visoko kvalitetnom prikazu jednakom filmskom (engl. *film-quality rendreing*) u stvarnom vremenu.

- krajem 1999. tvrtka NVIDIA uvela je termin "grafička procesna jedinica" GPU – *Graphics Processing Unit* kojim je opisala procesor na čipu GeForce 256 u kojem su integrirane funkcije koje podržavaju *logičku grafičku protočnu strukturu.*
- Grafička procesna jedinica GeForce 256 obrađivala je 15 M poligona (krpica) u sekundi i 480 M slikovnih elemenata u sekundi. Imao je čvrsto definiranu 32-bitnu FP (engl. *floating-point*) jedinicu za transformacije vrhova, procesor za proračun osvjetljenja (engl. *lighting processor*) i čvrsto definiranu cjelobrojnu funkciju koja se odnosila na preslikavanje boje i teksture na poligone.

Trenutačno stanje (2010.) je sljedeće: grafički su se procesori pretvorili u *programirljive paralelne procesore* koji svojom procesnom snagom premašuju višejezgrene (engl. *multicore*) procesore.

S arhitektonskog gledišta, grafički procesor je visoko paralelni, više dretveni procesor s vrlo velikim brojem jezgri namijenjen vizualnom računanju (engl. *visual computing*). *Vizualno računanje* je novi pojam koji se odnosi na složenu mješavinu grafičke obrade i računanja tako da omogućuje vizualnu interakciju u stvarnom vremenu računalno oblikovanih objekata uporabom grafike, slika i videa.

- Grafičkim procesorima pridodane su nove instrukcije i memorija tako da podržavaju više programske jezike i programska okruženja opće namjene. Grafički procesori sada se mogu programirati u programskim jezicima sličnim C i C++ i postaju procesori opće namjene s vrlo velikim brojem jezgri (engl. *manycore*)!
- definiran je i novi *model programiranja* jer velika procesna moć koja se odnosi na operacije brojevima s pomičnim zarezom te veliki stupanj paralelizma ostvaren vrlo velikim brojem procesora, odnosno jezgri, nudi djelotvornu primjenu grafičkih procesora na područjima izvan računalne grafike (na primjer, obrada digitalnih slika i videa, složeni postupci optimizacije uporabom evolucijskih algoritama, postupci kriptiranja/dekriptiranja podataka i sl.).

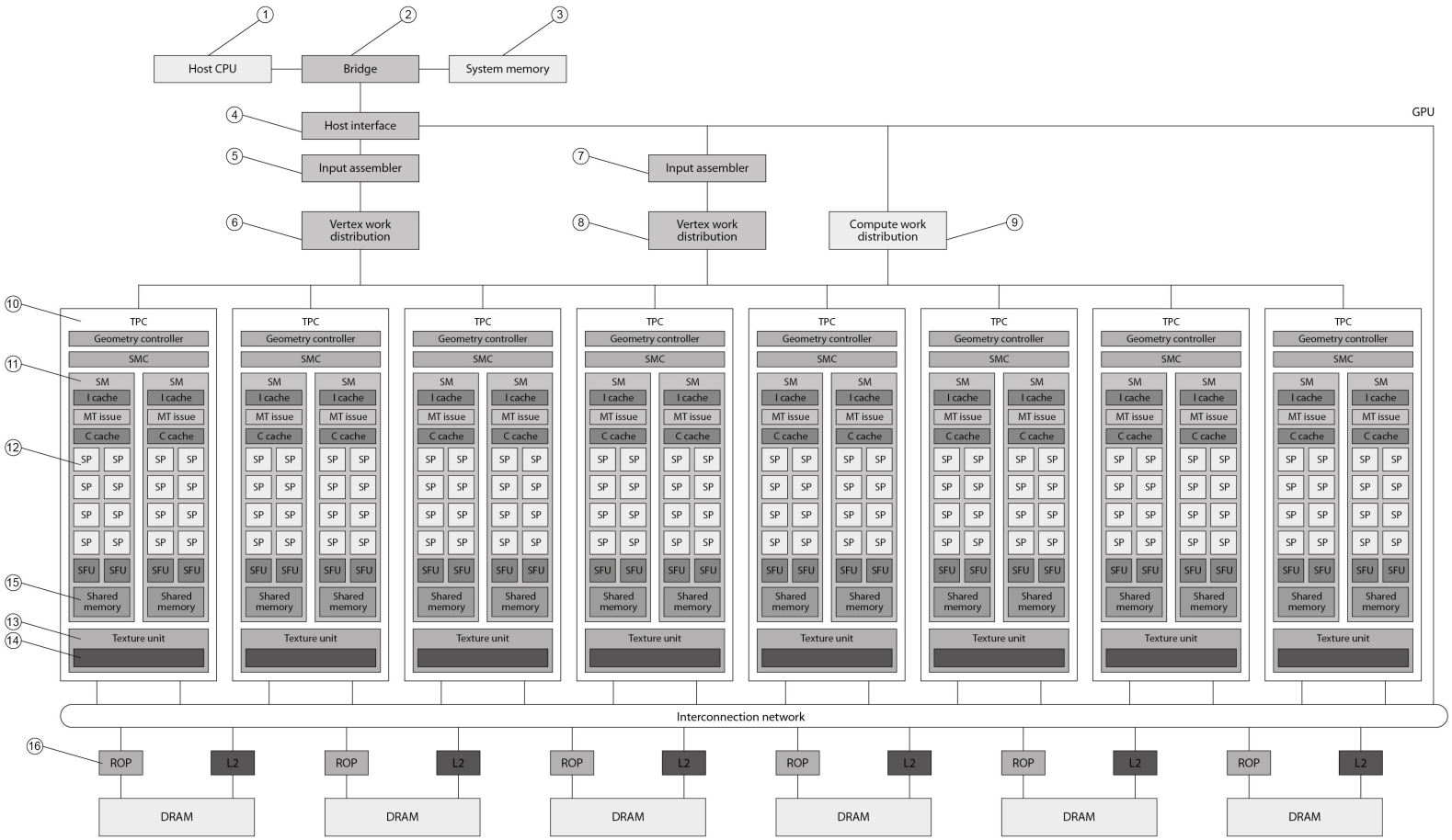
Jedan od modela paralelnog programiranja namijenjen grafičkim procesorima (ali i višejezgrenim procesorima) nove generacije je CUDA (*Compute Unified Device Architecture*) koji predstavlja i programsku platformu temeljenu na C/C++ programskom jeziku (CUDA C i CUDA C++). Model CUDA je zapravo SPMD (*Single-Program Multiple Data*) model u kojem programer piše program za jednu dretvu koja se potom instancira ili oprimjeruje u velikom broju dretvi na velikom broju procesora u grafičkoj procesnoj jedinici.

NVIDIA Tesla – unificirana arhitektura za grafiku i računanje

U studenom 2006. tvrtka NVIDIA definirala je novu arhitekturu pod nazivom Tesla za grafički procesor GeForce 8800.

Značajke:
- ujednačena visokoparalelna arhitektura procesora (grafičkom procesoru koji se temelji na Tesla arhitekturi nema razlike između procesora za sjenčanje vrhova i procesora za sjenčanje slikovnih elemenata.)
- Programi za tu vrstu aplikacija razvijaju se u višim programskim jezicima sličnim C i C++u skladu s modelom paralelnog programiranja CUDA

Blok-dijagram grafičkog procesora GeForce 8800



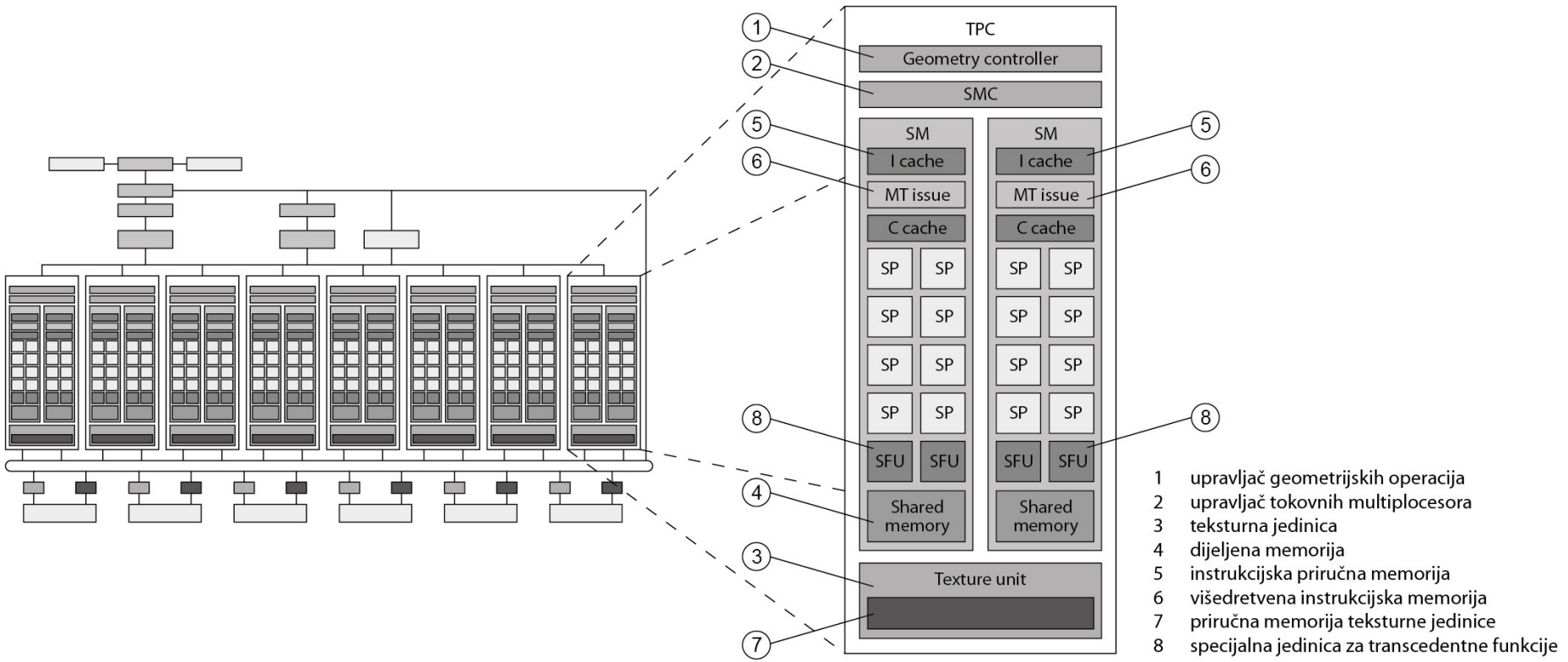
1 CPU domaćin
2 premosnik
3 sistemska memorija
4 sučelje za CPU domaćina
5 ulazni zbirnik
6 raspoređivanje poslova za procesore za sjenčanje vrhova
7 blok za rasterizaciju
8 raspoređivanje poslova za procesore za sjenčanje slikovnik elemenata
9 raspoređivanje računalnih poslova
10 TPS - teksturno-procesorska nakupina
11 SM - tokovi multiprocesor
12 SP - tokovni procesor
13 teksturna jedinica
14 priručna memorija
15 dijeljena memorija
16 procesor za rasterske operacije

Sustavno se arhitektura grafičkog procesora GeForce 8800 može opisati kao hijerarhijska organizacija koja se sastoji od:

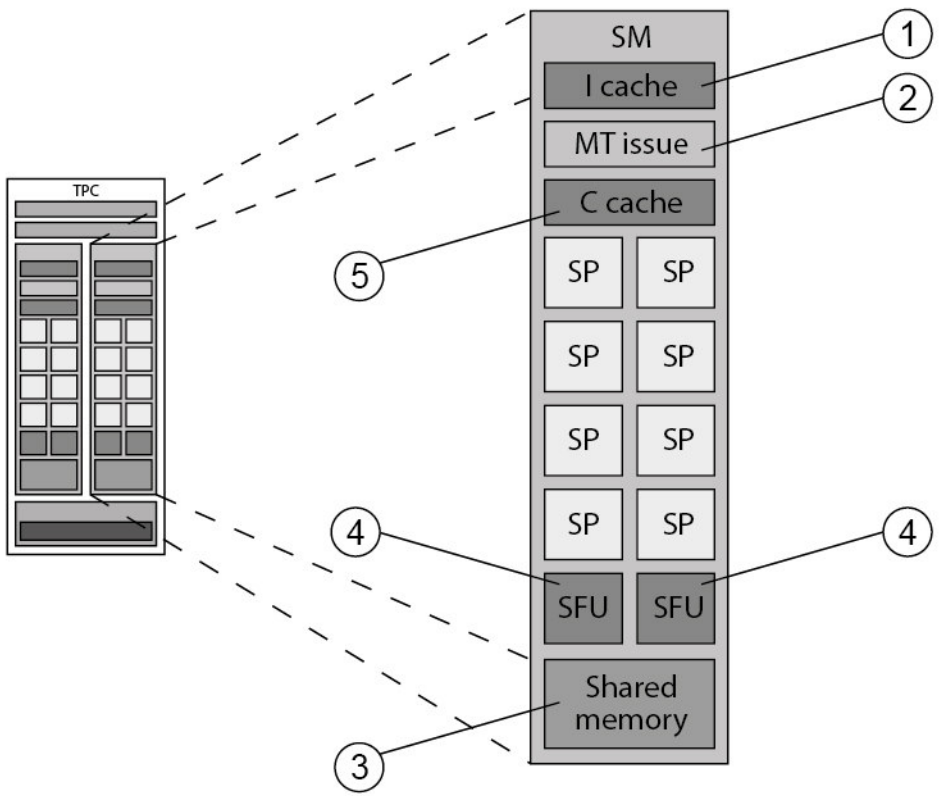
i) polja tokovnih procesora SPA (*Streaming Processor Array*),
ii) teksturno-procesorske nakupine TPC (*Texture /Processor Cluster*),
iii) tokovnih multiprocesora SM (*Streaming-Multiprocessor*),
iv) tokovnih procesora SP (*Streaming-Processor*) i procesora za rasterske operacije ROP (*Raster Opertions Processor*).

- Grafički procesor ima *polje tokovnih procesora* SPA (*Streaming Processor Array*) koje se sastoji od 128 tokovnih procesora (ili tokovnih procesorskih jezgri) SP (engl. *streaming-processor core*) – 16 x 8
-Tokovni procesori SP organizirani su u 16 tokovnih multiprocesora SM (engl. *streaming-multiprocessor*)
/Svaki tokovni procesor SP je višedretveni procesor koji podržava 96 istodobnih dretvi/
- tokovni multiprocesori SM grupirani su u osam nezavisnih procesorskih jedinica koje se nazivaju teksturno-procesorske nakupine TPC (engl. *texture/processor cluster*)
- Teksturno-procesorska nakupina ima po dva tokovna multiprocesora SM, a svaki od njih ima osam tokovnih procesora. Iz toga slijedi da jedan tokovni multiprocesor SM može istodobno izvršavati do 8×96 = 768 dretvi.

Teksturno-procesorska nakupina TPC



Tokovni multiprocesor SM



1 instrukcijska priručna memorija
2 všiedretvena instrukcijska jedinica
3 dijeljena memorija
4 specijalna jedinica za transcedentne funkcije
5 priručna ispisna memorija

- Procesorska jezgra SP GeForce 8800 je višedretvena i da rukuje istodobno s 96 dretvi.
- Jezgra ima skup od 1024 32-bitnih registara kojima podržava 96 dretvi, skalarnu jedinicu za množenje i zbrajanje brojeva s pomičnim zarezom MAD (*multiply-add*) tako da svaki tokovni multiprocesor SM ima osam takvih jedinica.
- Svaka jezgra podržava 32-bitnu i 64-bitnu cjelobrojnu aritmetiku te logičke PTX (*Parallel Thread eXecution*) instrukcije.
- Procesori su povezani sa šest 64-bitnim DRAM modula pomoću prospojne mreže.
- Svaki od šesnaest tokovnih multiprocesora SM ima, osim osam jezgri SP, instrukcijsku priručnu memoriju (I *cache*),

Svaki tokovni multiprocesor SM ima još i:
- višedretvenu instrukcijsku jedinicu za pribavljanje i izdavanje instrukcija MT
- priručnu ispisnu (*read-only cache*) memoriju za pohranu konstanti C-Cache (*Constant-cache*),
-dvije specijalne jedinice SFU (*Special Function Unit*) te zajedničku, dijeljenu memoriju

Performansa grafičkog procesora GeForce 8800 ovisno o izvedbi (Ultra ili GTX) je između 518 GFLOPS-a i 576 GFLOPS-a.
Usporedba GeForce 8800 GTX i 4-jezgrenog procesora Intel Xeon (2.8 GHz) pokazuje da je grafički procesor otprilike sedam puta brži u računanju brze Fouriereve transformacije FFT (*Fast Fourier Transform*)

Krajem 2010. na tržištu se pojavio grafički procesor, odnosno grafička kartica NVIDIA GeForce GTX 580 marketinški najavljen kao "najbrži grafički procesor na planeti" koji se temelji na arhitekturi NVIDIA Fermi. GTX 580 ima 512 tokovnih procesora SP koji su organizirani u 16 tokovnih multiprocesora SM i 48 procesora za rasterske operacije ROP.

Opaska: Podrobniji opis višejezgrenih i grafičkih procesora može se naći u knjizi S. Ribarić, *Građa računala, Arhitektura i organizacija računarskih sustava*, Algebra, Zagreb, izlazi u ožujku 2011. (520 str.)