4.5 Подобрување на перформансите

- Техники за подобрување на перформансите на процесорот
  - **кеш** (cache) меморија
  - **нередоследно** извршување на инструкциите (вон редослед; out-of-order) со:
    - **преименување** на регистрите (register renaming)
  - **шпекулативно** извршување (speculative execution) засновано на:
    - претскажување на **разгранувања** (branch prediction)
    - претскажување на **вредности** (value prediction)

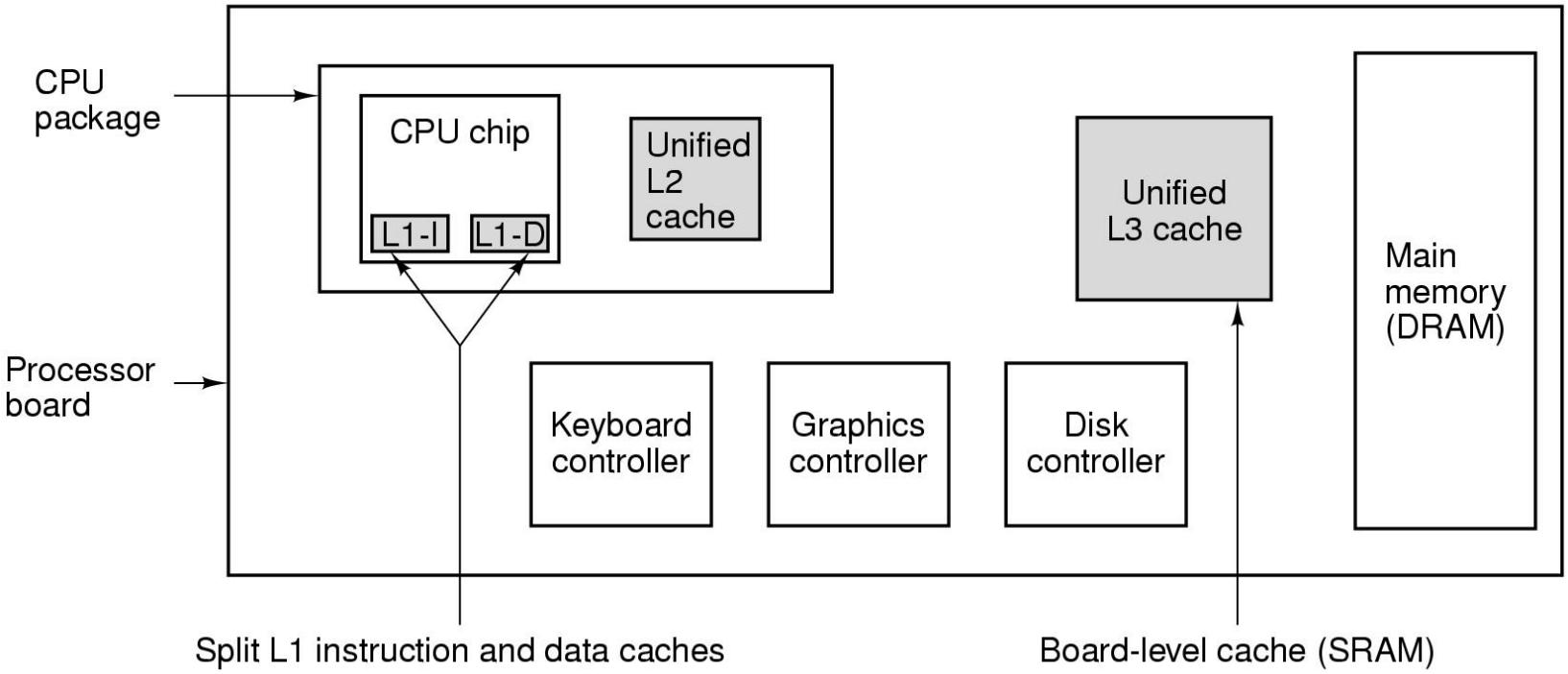
4.5.1 Кеш меморија

- Мала брза меморија во која се чуваат најнеодамна (најскоро) употребуваните мемориски зборови, со што се забрзува пристапот до нив
- Доколку доволно голем процент од потребните мемориски зборови се наоѓаат во кеш-меморијата, ефективното доцнење при комуникацијата со меморијата може значително да се намали
- Основна техника (работи мошне ефикасно):
  - **раздвоена кеш-меморија** (split cache) – одделни кеш-мемории за инструкции (instruction cache) и податоци (data cache)
  - мемориските операции можат да се иницираат независно кај двете кеш-мемории, удвојувајќи ја, на тој начин, пропусната моќ на мемориските систем
    - **Пример:** 32-битна и 8-битна мемориска порта кај Mic-1 микроархитектурата, секоја од нив со посебна кеш-меморија

4.5.1 Кеш меморија

- Кај пософистицираните мемориски системи, можат да постојат две, три, или повеќе нивоа на кеш-меморија
  - L1 (на самиот CPU чип): 16 KB – 64 KB
  - L2 (дел од „процесорскиот пакет“): 512 KB – 1MB
  - L3 (на матичната плоча): неколку MB (SRAM)
- Кеш мемориите се **инклузивни**
  - целата содржина на L1 кеш-меморијата е содржана и во L2 кеш-мемоијата
  - целата содржина на L2 кеш-меморијата е содржана и во L3 кеш-меморијата

4.5.1 Кеш меморија



4.5.1 Кеш меморија

- Кеш мемориите користат два вида на локалност на мемориските адреси:
  - **просторна локалност (spatial locality)**
    - **АКО** неодамна се пристапило на некоја мемориска локација, **ТОГАШ**, со голема веројатност, наскоро ќе им се пристапи и на мемориските локации кои имаат **нумерички слични** (соседни) адреси
  - **временска локалност (temporal locality)**
    - **АКО** неодамна се пристапило на некои мемориски локации, **ТОГАШ**, со голема веројатност, наскоро **повторно** ќе им се пристапи
      - **Забелешка:** принципот на временска локалност се применува при донесување на одлука „што треба да се исфрли од кеш-меморијата" при промашување

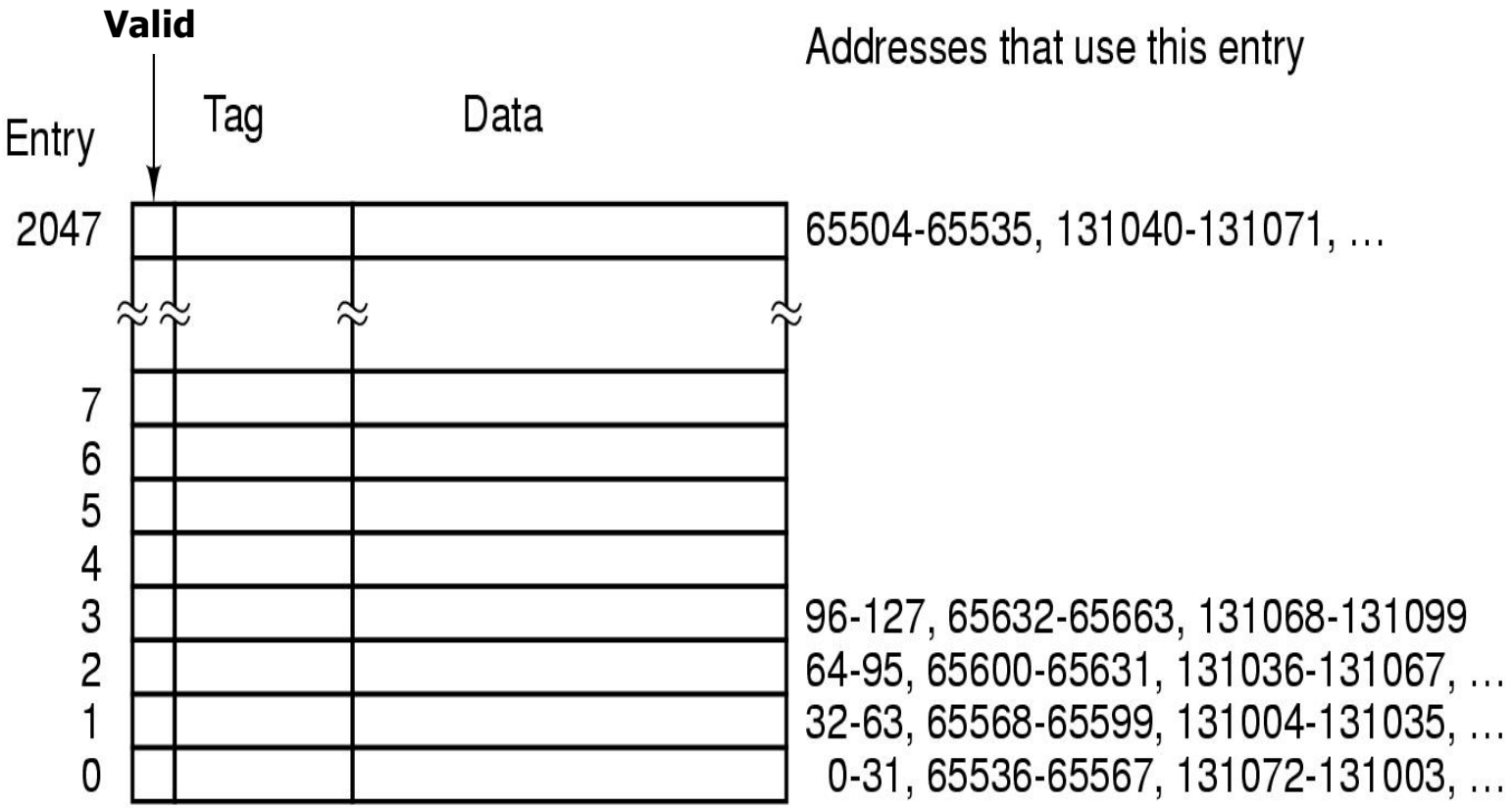
4.5.1 Кеш меморија

- Модел:
  - Главната меморија се дели на блокови со фиксна големина – **кеш линии** (со големина 4 – 64 последователни бајтови)
  - Во секој момент, некои кеш линии се содржани во кеш-меморијата
  - При комуникација со меморијата, се проверува дали бараниот збор е содржан во кеш-меморијата(?)
    - Ако зборот е најден, веднаш може да се искористи соодветната вредност, без да се пристапува до главната меморија
    - Ако зборот не е најден, една од кеш-линиите се исфрла од кеш-меморијата, а на нејзино место (од главната меморија, или од кеш-меморијата од пониското ниво) се презема и се внесува бараната кеш-линија

4.5.1.1 Директно-пресликана кеш меморија

- **Директно-пресликана кеш меморија** (Direct-Mapped Cache) – наједноставна кеш-меморија
- Пример:
  - 2048 ($\mathbf{2^{11}}$) редици – секоја од нив може да содржи само една кеш-линија од главната меморија
  - Должина на кеш-линија = 32 бајти
  - Капацитет = 2048 x 32 = 64 KB
  - Секоја редица се состои од три дела:
    - Поле **Valid** (1 бит) – покажува дали податоците во соодветната кеш-линија се валидни или не (при стартирање на системот, сите редици имаат бит Valid=0)
    - Поле **Tag** (16 бита) – ја идентификува соодветната линија од меморијата од која потекнуваат податоците
    - Поле **Data** (32 бајти)– содржи копија од податоците во меморијата (кеш-линија)

4.5.1.1 Директно-пресликана кеш меморија

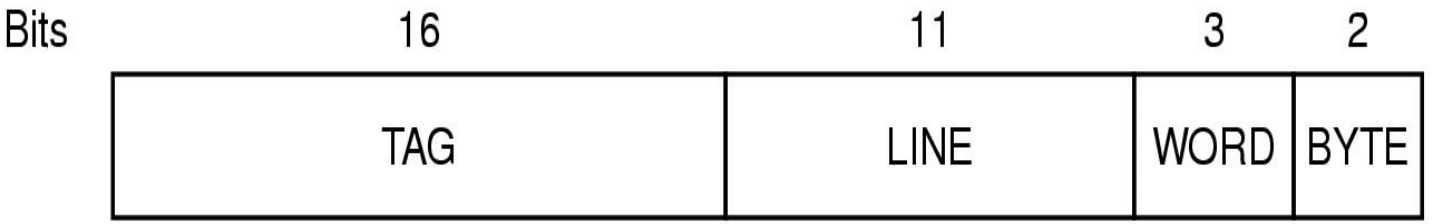


4.5.1.1 Директно-пресликана кеш меморија

- Секој меморискиот збор може да биде сместен само на едно единствено место во кеш-меморијата – ако се знае меморискиата адреса, тогаш постои само едно место на кое може да се бара соодветниот збор (ако не е таму, тогаш воопшто не е во кеш-меморијата!)
- За запишување и читање на податоци од кеш-меморијата, 32-битната адреса е поделена на четири компоненти:
  - Поле **Tag** – соодветствува на истоименото поле содржано во една редица од кеш-меморијата
  - Поле **LINE** – ја означува редицата од кеш-меморијата која ги содржи бараните податоци (доколку се присутни во неа)
  - Поле **WORD** – означува кој збор од линијата е потребен
  - Поле **BYTE** – означува кој бајт од зборот е потребен

4.5.1.1 Директно-пресликана кеш меморија

- 32-битна адреса



- Кога процесорот ќе генерира мемориска адреса, хардверот ги одвојува 11-те LINE битови и ги користи како индекс (покажувач) кон една од 2048-те **редици**

4.5.1.1 Директно-пресликана кеш меморија

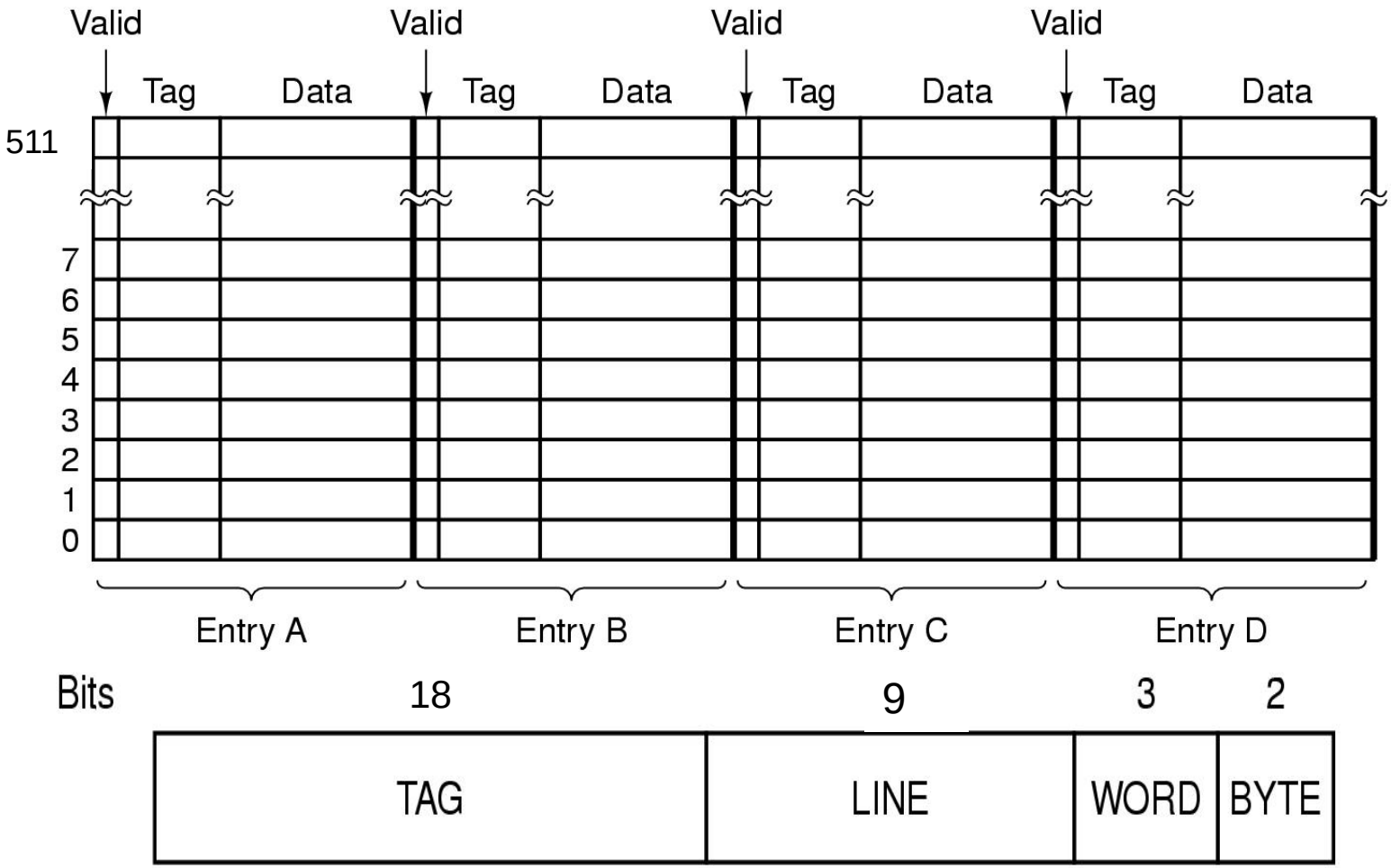
- Ако содржината на редицата е валидна, се споредува **TAG** полето од адресата со **Tag** полето од кеш-меморијата
  - Ако **TAG=Tag**, тогаш има совпаѓање – редицата го содржи бараниот збор – **погодок** (cache hit)
  - Ако **TAG<>Tag**, тогаш има несовпаѓање – бараната линија не е присутна во кеш-меморијата – **промашување** (cache miss) – 32-бајтната кеш-линија која што била побарана се презема од главната меморијата и се запишува во кеш-меморијата, а тековната содржина се пребришува
    - **Забелешка**: ако постојната редица од кеш-меморијата била менувана, тогаш мора да биде запишана назад во главната меморија пред да биде исфрлена!

4.5.1.2 Ограничено-асоцијативна кеш меморија

- **ПРОБЛЕМ**: ако програмата пристапува кон податоци на локација X, а потоа кон податоци од локација X+65536 (или некоја соседна локација од истата кеш-линија), тогаш ќе се предизвика преземање на нова кеш-линија од главната меморија и пребришување на тековната содржина
  - ако тоа се случува често, доаѓа до деградација на перформансите(!)
- **МОЖНО РЕШЕНИЕ:** ако во секоја редица од кеш-меморијата се запишуваат по две или повеќе кеш-линии, тогаш секоја редица ќе претставува **множество** од кеш-линии (set)
- **n-насочна ограничено-асоцијативна кеш-меморија** (n-way Set-Associative Cache) – кеш-меморија со **n** можни позиции за секоја адреса (најчесто, n=2 или n=4)
- **НЕДОСТАТОК**: Секогаш треба да се проверува множество од **n** кеш-линии за да се утврди дали бараната линија е присутна

4.5.1.2 Ограничено-асоцијативна кеш меморија

- Пример: **4-насочна** ограничено-асоцијативна меморија (со ист капацитет)



4.5.1.2 Ограничено-асоцијативна кеш меморија

- **Прашање:** која од кеш-линиите да се исфрли кога треба да се преземе нова кеш-линија од меморијата?
- Ефикасен алгоритам – **LRU** (Least Recently Used) – се исфрла линијата која **најдолго време не била употребена**
  - За линиите од едно множество се чува информација („топ-листа“) која ги подредува во склад со обраќањата до нив:
    - Штом ќе биде побарана (употребена) некоја од линиите, таа се става на врвот од листата
    - Линијата која се наоѓа на дното од листата е таа која најдолго време не била употребена и затоа е кандидат за исфрлање од кеш-меморијата(!)

4.5.1.2 Ограничено-асоцијативна кеш меморија

- **Прашање:** доколку процесорот запишува (write) некој мемориски збор, дали истиот треба **веднаш** да се запише во главната меморија?
- Постојат две можности:
  - **Запишување <u>низ</u> кеш-меморијата** (write through) – зборот се запишува во кеш-меморијата, а истовремено и во главната меморија (на тој начин, содршината на главната меморија е секогаш ажурирана – "up to date")
  - **Одложено запишување** (write back; write deferred) – зборот се запишува во главната меморија многу **подоцна**, дури кога линијата треба да биде пребришана (во согласност со LRU алгоритмот)