

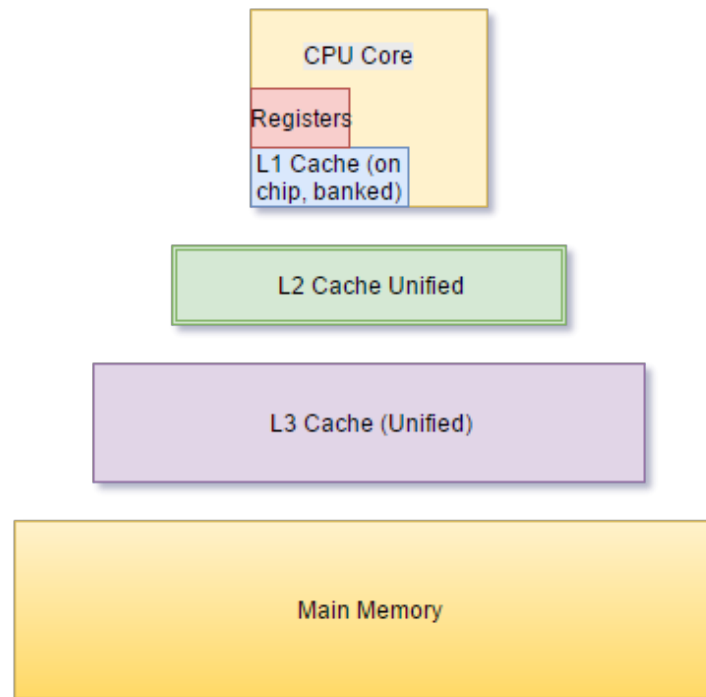
Cache Memory
Predpomnilnik

Characteristics / Lastnosti pomnilnikov

- Location
- Capacity
- Unit of transfer
- Access method
- Performance
- Physical type
- Physical characteristics
- Organisation
- Lokacija
- Velikost
- Enota prenosa
- Metode dostopa
- Učinkovitost
- Vrste tehnologije
- Fizikalne karakteristike
- Organizacija

Location / Lokacija

- CPU
 - Internal
 - External
- CPE
 - Notranji
 - Zunanji



Capacity / Velikost

- Word size
 - The natural unit of organisation
 - The word size of a CPU is typically chosen based on the size of the data bus (i.e., the number of bits that can be loaded from memory or stored into memory in one go) – but not necessarily (see next slide)
- Number of words
 - or Bytes
- Typical size
 - 8, 16, 32 bits (1, 2, 4 bytes)
 - Intel 16 bits WORD, 32 DWORD and 64 QWORD
- Velikost besede
 - Naravna enota organizacije
 - Velikost besede CPE je običajno izbrana glede na velikost podatkovnega vodila (tj. število bitov, ki jih je mogoče naložiti iz pomnilnika ali shraniti v pomnilnik naenkrat) – ne nujno tako (naslednja prosojnica)
- Število besed
 - ali običajno zlogov (angl. bytes)
- Tipična velikost
 - 8, 16, 32 bitov (1, 2, 4 zlogi)
 - Intel 16 bitov WORD, 32 DWORD and 64 QWORD

Unit of transfer / Enota prenosa

- Internal
 - Usually governed by data bus width
- External
 - Usually a block which is much larger than a word
- Addressable unit
 - Smallest location which can be uniquely addressed
 - Usually Word internally, but some systems allow to address on a byte level
 - On external memory blocks (records on magnetic tapes, cluster by Microsoft OS) – much larger than words
- Notranji pomnilniki
 - Ponavadi je določena s širino vodila
- Zunanji pomnilniki
 - Običajno je to blok, ki je precej večji kot beseda
- Naslovljiva enota
 - Najmanjša lokacija, ki jo lahko enolično naslovimo
 - Ponavadi Beseda pri notranjih pomnilnikih, vendar nekateri sistemi omogočajo naslavljanje na ravni bajtov
 - Na zunanjem pomnilniku je to blok (zapis na magnetnih trakovih, gruča v Microsoft OS) – veliko večji od besed.

Access methods (1) / Metode dostopa (1)

- Sequential
 - Start at the beginning and read through in order
 - Access time depends on location of data and previous location
 - e.g. tape
- Direct
 - Individual blocks have unique address
 - Access is by jumping to vicinity plus sequential search
 - Access time depends on location and previous location
 - e.g. disk
- Zaporedna
 - Začne se na začetku in bere v zaporedju
 - Dostopni čas je odvisen od trenutne in predhodne lokacije podatkov
 - Npr. trak
- Neposredna
 - Posamezni bloki imajo enoličen naslov
 - Dostop je omogočen s premikom v bližino in naknadnim zaporednim iskanjem
 - Dostopni čas je odvisen od trenutne in predhodne lokacije podatkov
 - Npr. disk

Access methods (2) / Metode dostopa (2)

- Random
 - Individual addresses identify locations exactly
 - Access time is independent of location or previous access
 - e.g. RAM
- Associative
 - Data is located by a comparison with contents of a portion of the store
 - Access time is independent of location or previous access
 - e.g. cache
- Naključna
 - Posamezni naslovi natančno določajo lokacijo
 - Dostopni čas je neodvisen od lokacije prejšnjega dostopa
 - Npr. RAM
- Asociativna
 - Podatki se iščejo s primerjanjem dela shranjene vsebine
 - Dostopni čas je neodvisen od lokacije prejšnjega dostopa
 - Npr. predpomnilnik

Performance / Zmogljivost

- Access time
 - Time between presenting the address and getting the valid data
 - Other types: time to position R/W mechanism at the desired location
- Memory Cycle time (primarily for RAM)
 - Time may be required for the memory to “recover” before next access (for signals to die out on signal lines or to regenerate data if they are read destructively)
 - Cycle time is access + recovery
- Transfer Rate
 - Rate at which data can be moved
- Dostopni čas
 - RAM: Čas med vročitvijo naslova in pridobitvijo veljavnih podatkov
 - Druge vrste: čas postavitve B/P mehanizma na željeno lokacijo
- Čas pomnilniškega cikla (primarno za RAM)
 - Po vsakem dostopu je potreben določen (mrtvi) čas, pred naslednjim dostopom (da signali ugasnejo na signalnih linijah ali da se podatki regenerirajo, če so brani destruktivno)
 - Čas cikla je dostopni čas + mrtvi čas
- Hitrost prenosa
 - Hitrost s katero lahko prenašamo podatke

Physical types / Vrste tehnologije

- Semiconductor
 - RAM, SSD
- Magnetic
 - Disk & Tape
- Optical
 - CD, DVD, HDDVD, Blu-ray
- Others
 - Bubble (discontinued in 80s)
https://en.wikipedia.org/wiki/Bubble_memory
 - CBRAM, Racetrack memory, NRAM, Millipede memory, FJG RAM
 - Hologram (future?)
https://en.wikipedia.org/wiki/Holographic_data_storage
- Polprevodnik
 - RAM, SSD
- Magnetni
 - Disk in trak
- Optični
 - CD, DVD, HDDVD, Blu-ray
- Ostali
 - Mehurčki
https://en.wikipedia.org/wiki/Bubble_memory
 - Hologrami
https://en.wikipedia.org/wiki/Holographic_data_storage

Physical characteristics / Fizikalne lastnosti

- Data degradation (data decay, data rot, bit rot)
 - Volatility
 - Volatile memory
 - Information decays naturally or is lost when electrical power is switched off
 - Nonvolatile memory
 - Once recorded, information remains there without deterioration until deliberately changed
 - No electrical power is needed to retain information
 - Example is magnetic-surface memory
 - Semiconductor memory may be either volatile (RAM) or nonvolatile (SSD)
 - Erasability
 - Nonerasable memory
 - Cannot be altered, except by destroying the storage unit
 - Semiconductor memory of this type is known as read-only memory (ROM)
 - Power consumption
- Razpad podatkov
 - Obstočnost
 - Neobstočni pomnilnik
 - Informacije razpadejo ali se izgubijo, ko je električna energija izklopljena
 - Obstočni pomnilnik
 - Ko so informacije zapisane, ostanejo tam nesoremenjene, dokler se namerno ne spremenijo
 - Električna energija ni potrebna za obstoj informacij
 - Primer predstavljajo magnetni pomnilniki
 - Polprevodniški pomnilnik je lahko obstojen (SSD) ali neobstočen (RAM)
 - Možnost brisanja
 - Neizbrisljiv pomnilnik
 - Ni ga mogoče spremeniti, razen z uničenjem pomnilniške enote
 - Polprevodniški pomnilnik te vrste je znan kot trajni pomnilnik (ROM)
 - Poraba energije

Computer memory and data storage types

General [show]

Volatile

RAM [hide]

Hardware cache (CPU cache · Scratchpad memory) · DRAM (eDRAM · SDRAM · SGRAM · LPDDR · QDRSRAM · EDO DRAM · XDR DRAM · RDRAM · SDRAM · DDR · GDDR · HBM) · SRAM (1T-SRAM) · ReRAM · QRAM · Content-addressable memory (CAM) · VRAM · Dual-ported RAM (Video RAM (dual-ported DRAM))

Historical [hide]

Williams–Kilburn tube (1946–1947) · Delay-line memory (1947) · Mellon optical memory (1951) · Selectron tube (1952) · Dekatron · T-RAM (2009) · Z-RAM (2002–2010)

Non-volatile

ROM [hide]

MROM · PROM (EPROM · EEPROM) · ROM cartridge · Solid-state storage (SSS) (Flash memory is used in: · Solid-state drive (SSD) · Solid-state hybrid drive (SSHD) · USB flash drive · IBM FlashSystem · Flash Core Module) · Memory card (Memory Stick · CompactFlash · PC Card · MultiMediaCard · SD card · SIM card · SmartMedia · Universal Flash Storage · SxS · MicroP2 · XQD card) · Programmable metallization cell

NVRAM [hide]

Memistor · Memristor · PCM (3D XPoint) · MRAM · Electrochemical RAM (ECRAM) · Nano-RAM · CBRAM

Early-stage NVRAM [hide]

FeRAM · ReRAM · FeFET memory

Analog recording [hide]

Phonograph cylinder · Phonograph record · Quadruplex videotape · Vision Electronic Recording Apparatus · Magnetic recording (Magnetic storage · Magnetic tape · Magnetic-tape data storage · Tape drive · Tape library · Digital Data Storage (DDS) · Videotape · Videocassette · Cassette tape · Linear Tape-Open · Betamax · 8 mm video format · DV · MiniDV · MicroMV · U-matic · VHS · S-VHS · VHS-C · D-VHS) · Hard disk drive

Optical [hide]

3D optical data storage (Optical disc · LaserDisc · Compact Disc Digital Audio (CDDA) · CD · CD Video · CD-R · CD-RW · Video CD · Super Video CD · Mini CD · Nintendo optical discs · CD-ROM · Hyper CD-ROM · DVD · DVD+R · DVD-Video · DVD card · DVD-RAM · MiniDVD · HD DVD · Blu-ray · Ultra HD Blu-ray · Holographic Versatile Disc) · WORM

In development [hide]

CBRAM · Racetrack memory · NRAM · Millipede memory · ECRAM · Patterned media · Holographic data storage (Electronic quantum holography) · 5D optical data storage · DNA digital data storage · Universal memory · Time crystal · Quantum memory · UltraRAM

Historical [hide]

Paper data storage (1725) · Punched card (1725) · Punched tape (1725) · Plugboard · Delay-line memory · Drum memory (1932) · Magnetic-core memory (1949) · Plated-wire memory (1957) · Core rope memory (1960s) · Thin-film memory (1962) · Disk pack (1962) · Twistor memory (~1968) · Bubble memory (~1970) · Floppy disk (1971)

V · T · E

Organisation / Organizacija

- Physical arrangement of bits into words
- For random-access memory the organisation is a key design issue
- Fizična organizacija bitov v besede
- Za pomnilnik z naključnim dostopom (znan tudi kot bralopisalni pomnilnik, RAM) je organizacija ključno vprašanje oblikovanja

The bottom line / Zaključek

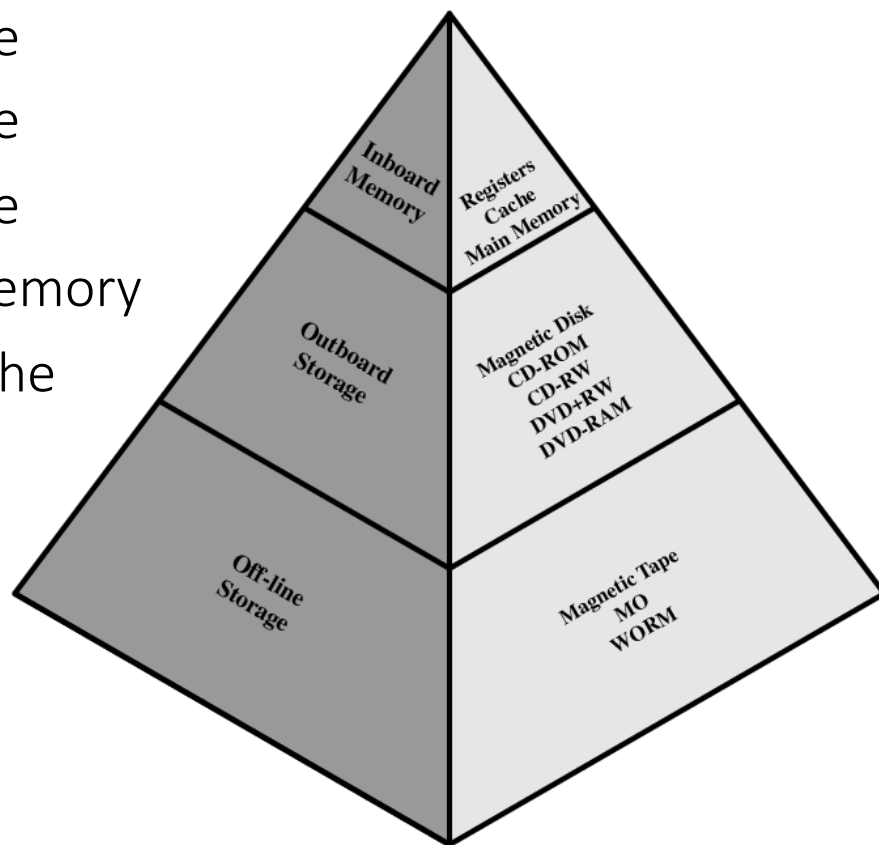
- How much?
 - Capacity
 - How fast? (access time)
 - Time is money
 - How expensive?
 - Trade-off:
 - Faster access time, greater cost per bit
 - Greater capacity, smaller cost per bit
 - Greater capacity, slower access time
 - The way out of the memory dilemma is not to rely on a single memory component or technology, but to employ a memory hierarchy
- Koliko?
 - Velikost
 - Kako hitro? (čas dostopa)
 - Čas je denar
 - Cena?
 - Kompromis:
 - Hitrejši čas dostopa, večja cena na bit
 - Večja velikost, manjši stroški na bit
 - Večja velikost, počasnejši čas dostopa
 - Izhod iz dileme je uporaba hierarhije pomnilnika.

Memory hierarchy / Pomnilniška hierarhija

- Registers
 - In CPU
 - Internal or Main memory
 - May include one or more levels of cache
 - “RAM”
 - External memory
 - Backing storage
 - eg. SSD, hard drives, tapes
- Registri
 - v CPE
 - Notranji ali glavni pomnilnik
 - Lahko vsebuje več nivojev” predpomnilnika
 - “RAM” (pomnilnik z naključnim dostopom, bralno-pisalni pomnilnik)
 - Zunanji pomnilnik
 - Shranjevanje
 - Npr. SSD, trdi disk, trak

Hierarchy diagram / Diagram hierarhije

- Registers
- L1 Cache
- L2 Cache
- L3 Cache
- Main memory
- Disk cache
- Disk
- Optical
- Tape



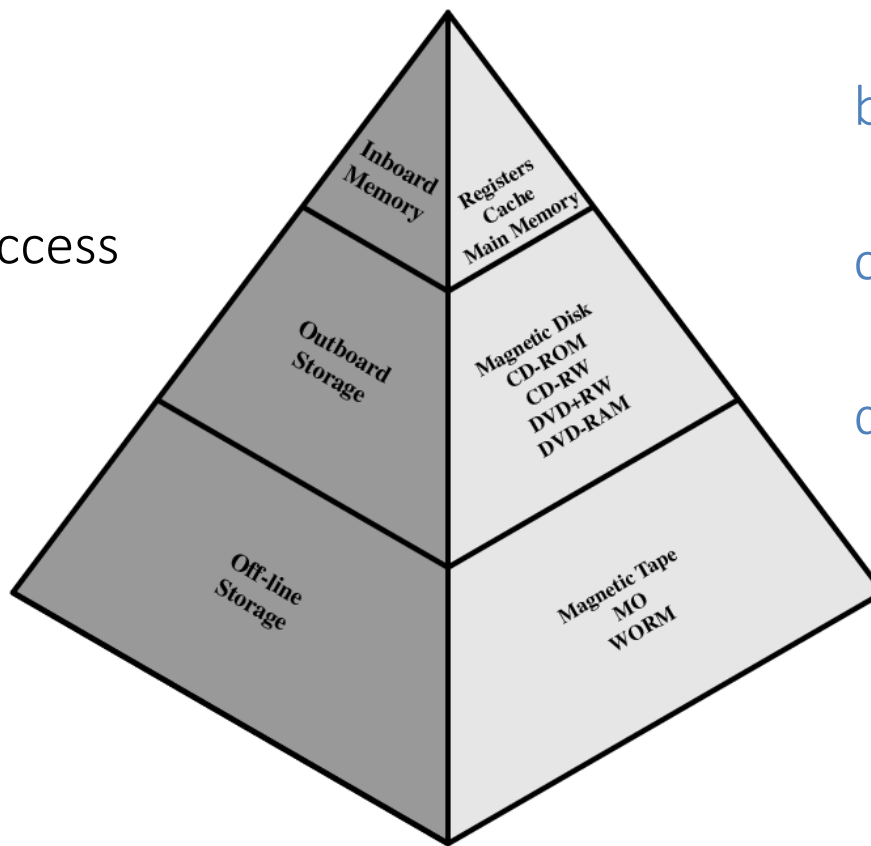
- Registri
- L1 predpomnilnik
- L2 predpomnilnik
- L3 predpomnilnik
- Glavni pomnilnik
- Diskovni predpomnilnik
- Disk
- Optični disk
- Trak

External memory / Zunanji pomnilnik

- Data are stored more permanently on external mass storage devices
- External, nonvolatile memory is also referred to as **secondary** memory or **auxiliary** memory
- Disk (or software) cache (software based)
 - A portion of main memory can be used as a buffer to hold data temporarily that is to be written to disk
 - A few large transfers of data can be used instead of many small transfers of data
 - Data can be retrieved rapidly from the software cache rather than slowly from the disk
- Podatki so trajneje shranjeni na zunanjih masovnih napravah
- Zunanji, obstojni pomnilnik se imenuje tudi sekundarni pomnilnik ali pomožni pomnilnik
- Diskovni (programski) predpomnilnik (programska operma)
 - Del glavnega pomnilnika se lahko uporablja kot vmesni pomnilnik za začasno shranjevanje podatkov, ki jih bomo zapisali na disk
 - Namesto številnih majhnih prenosov podatkov se lahko uporabi nekaj velikih prenosov podatkov
 - Podatke je mogoče hitro pridobiti iz predpomnilnika programske opreme in ne počasi z diska

Hierarchy characteristics / Značilnosti hierarhije

- a. Decreasing cost per bit;
- b. Increasing capacity;
- c. Increasing access time;
- d. Decreasing frequency of access to the memory by the processor.



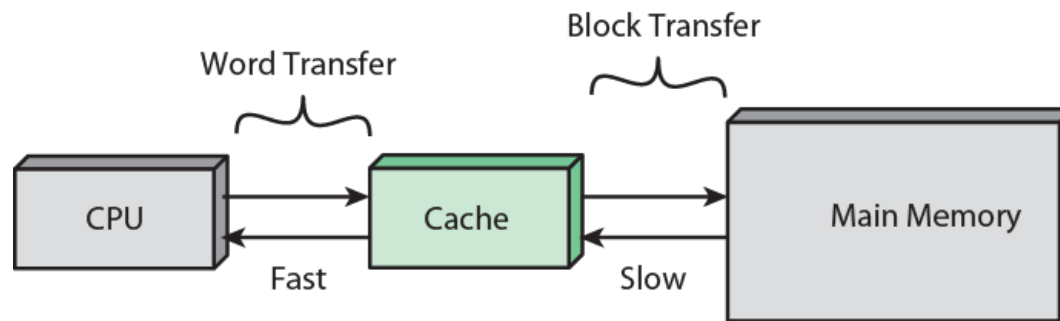
- a. Zmanjšanje stroškov na bit;
- b. Povečanje velikosti;
- c. Povečanje časa dostopa;
- d. Zmanjšanje pogostosti dostopa do pomnilnika s strani procesorja.

Locality of reference / Lokalnost referenc

- The basis for the validity of condition (d)
 - During the course of the execution of a program, memory references tend to cluster
 - e.g. loops, subroutines, arrays
 - Over a long period of time, the clusters in use change, but over a short period of time, the processor is primarily working with fixed clusters of memory references.
- Podlaga za veljavnost pogoja (d)
 - Med izvajanjem programa, so pomnilniške reference (naslovi) pogosto lokalne (združene v gruče)
 - npr. zanke, podprogrami, polja
 - V daljšem časovnem obdobju se gruče v uporabi spreminjajo, v kratkem času pa procesor zahteva podatke predvsem iz manjšega nabora gruč pomnilniških referenc.

Cache / Predpomnilnik

- Small amount of fast memory
- Sits between main memory and CPU
- May be located on CPU chip or module
- Majhna velikost hitrega pomnilnika
- Nahaja se med glavnim pomnilnikom in CPE
- Lahko se nahaja v procesorju ali posebnem modulu

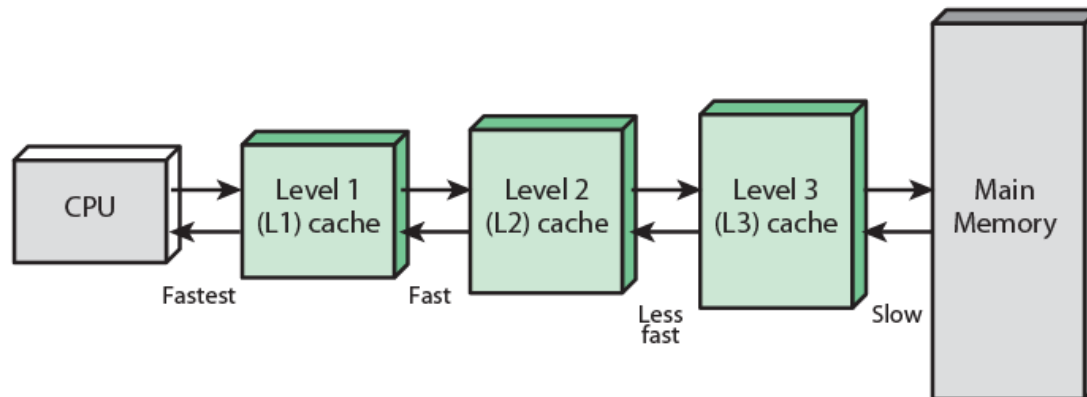


- The use of three levels exploits the fact that semiconductor memory comes in a variety of types which differ in speed and cost
- Uporaba treh ravni izkorišča dejstvo, da obstajajo različne vrste polprevodniških pomnilnikov, ki se razlikujejo v ceni in hitrosti

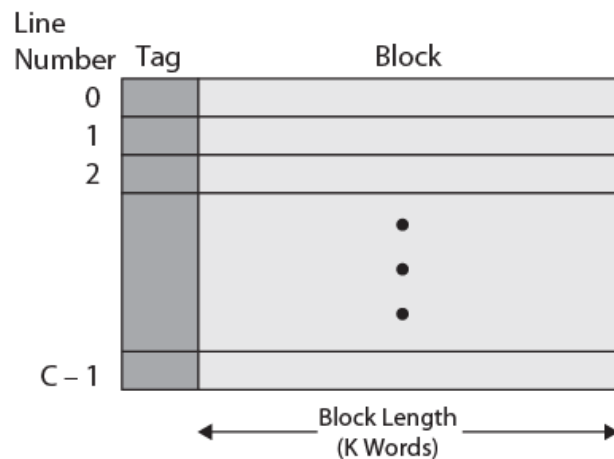
Multiple levels / Več ravni predpomnilnika

Three levels can be expanded to multiple levels to even further exploit the fact that semiconductor memory comes in a variety of types which differ in speed and cost.

Uporaba treh ravni se lahko razširi na še več ravni kar dodatno izkorišča dejstvo, da obstajajo različne vrste polprevodniških pomnilnikov, ki se razlikujejo v ceni in hitrosti



Cache/main memory structure / Sestava pred- in glavnega pomnilnika



(a) Cache

Main memory 2^n words each has n -bit address

M blocks = $2^n / K$

Cache consists of C blocks called lines

Each line contains K words (line size) + a tag + control bits (for recording changes, not show).

$C \ll M$

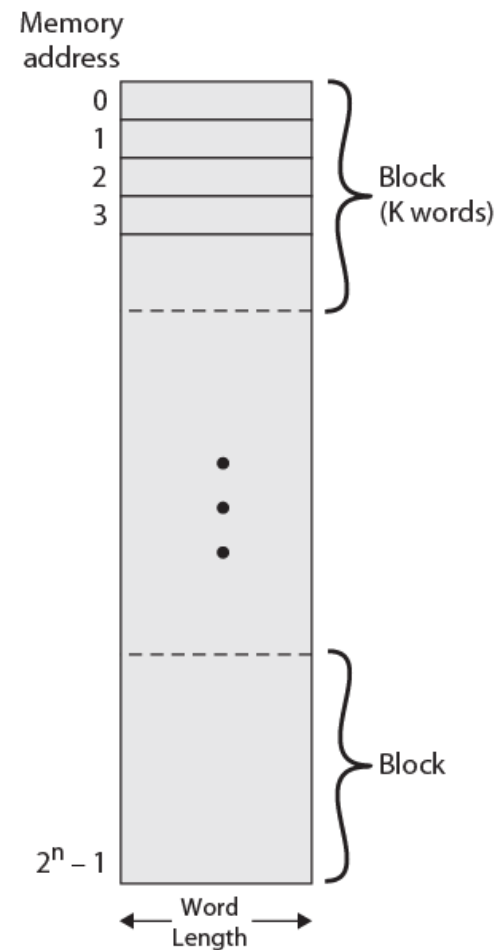
Glavni pomnilnik 2^n besed, vsaka ima n -bitni naslov

M blokov = $2^n / K$

Predpomnilnik je sestavljen iz C blokov, imenovanih vrstice

Vsaka vrstica vsebuje K besed (dolžina vestice) + oznako + kontrolne bite (za beleženje sprememb, ni prikazano).

$C \ll M$

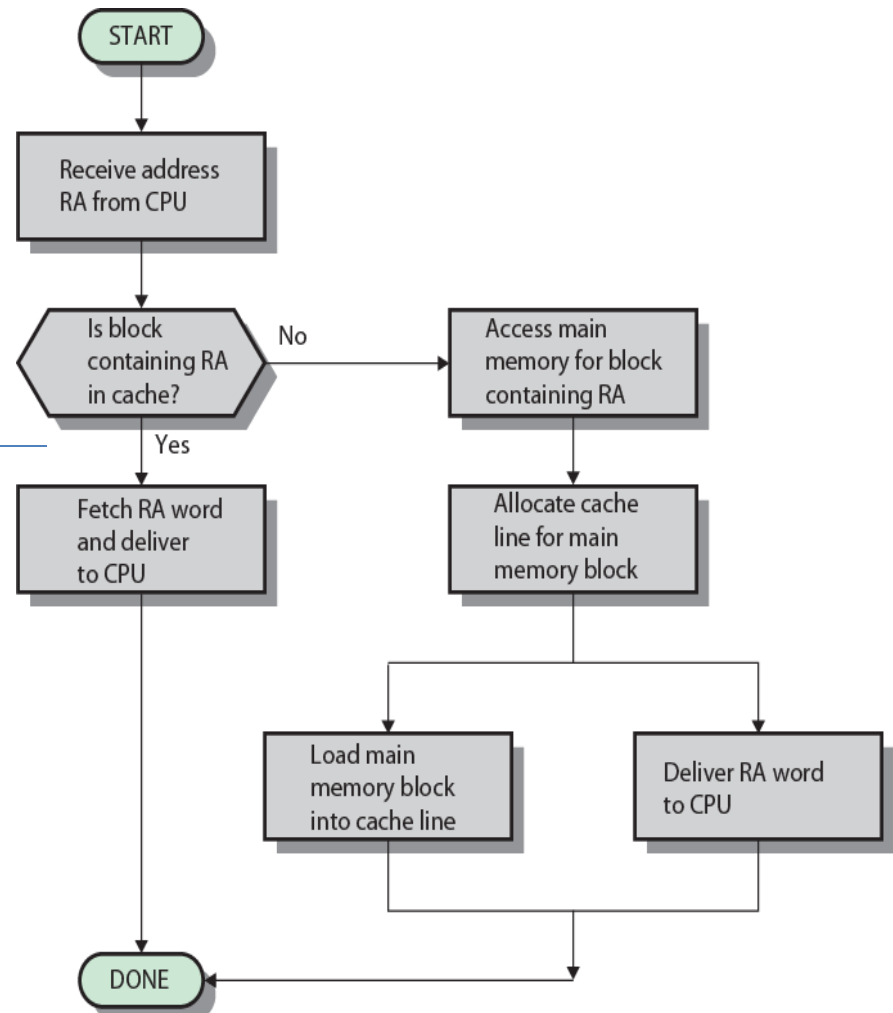


(b) Main memory

Cache operation – overview / Delovanje predpomnilnika – pregled

- CPU requests contents of memory location
- Check cache for this data
- If present, get from cache (fast)
- If not present, read required block from main memory to cache
- Then deliver from cache to CPU
- Cache includes tags to identify which block of main memory is in each cache slot

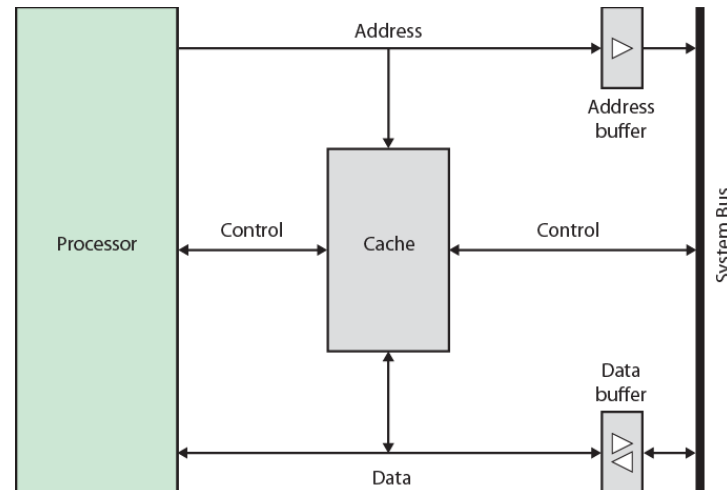
- CPU zahteva vsebino lokacije v pomnilniku
- Preveri v predpomnilniku ali vsebuje te podatke
- Če so prisotni, jih dobi iz predpomnilnika (hitro)
- Če niso prisotni, prebere zahtevani blok iz glavnega pomnilnika v predpomnilnik, ki jih potem dostavi CPU
- Predpomnilnik vsebuje oznake (angl. tags) za določitev, kateri blok iz glavnega pomnilnika se nahaja v pripadajoči vrstici/bloku predpomnilnika



Typical cache organisation with system bus / Tipična organizacija predpomnilnika z vodilom

Two operations (load memory block to cache, deliver to CPU) occurring in parallel

Dve operaciji (nalaganje bloka pomnilnika v predpomnilnik, dostava CPE) potekata vzporedno

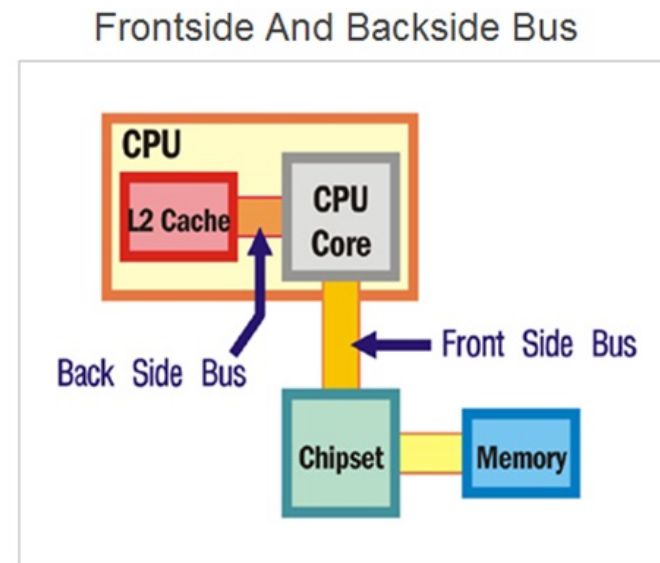
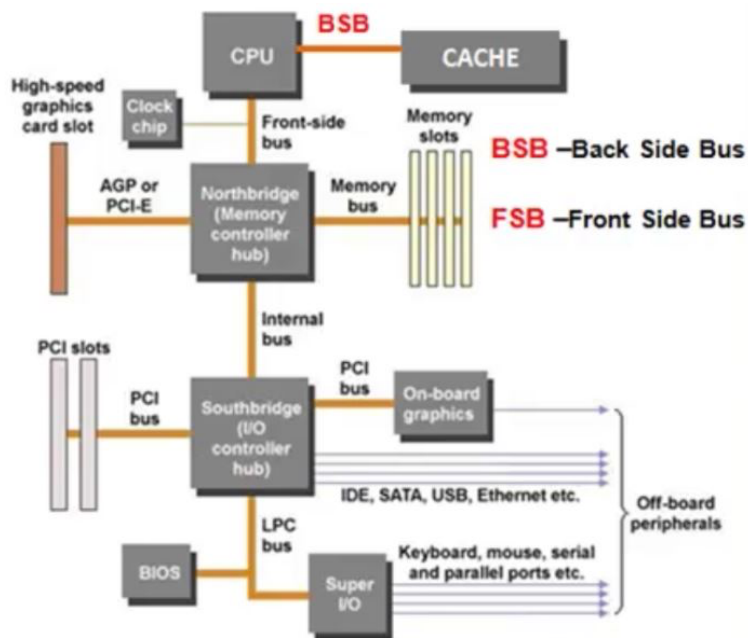


- When a cache hit occurs, the data and address buffers are disabled and communication is only between processor and cache
- When a cache miss occurs, the desired address is loaded onto the system bus and the data are returned through the data buffer to both the cache and the processor.
- Ko pride do zadetka v predpomnilniku, so medpomnilniki podatkov in naslovov onemogočeni in komunikacija poteka samo med procesorjem in predpomnilnikom
- Ko pride do nezadetka v predpomnilniku, se želeni naslov naloži na sistemsko vodilo in podatki se vrnejo skozi podatkovni medpomnilnik tako v predpomnilnik kot v procesor.

Other organisations / Druge organizacije

Cache can physically be interposed between the processor and the main memory and any other organisation is possible.

Predpomnilnik je lahko fizično vstavljen med procesor in glavni pomnilnik oz. možna je katera koli druga organizacija.



Cache design / Načrtovanje predpomnilnika

- Addressing
- Size
- Mapping Function
- Replacement Algorithm
- Write Policy
- Line Size (block)
- Number of Caches
- Naslavljanje
- Velikost
- Preslikovalna funkcija
- Zamenjevalni algoritem
- Strategija pisanja
- Velikost vrstice (bloka)
- Število predpomnilnikov

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

Depends on the computer: HPC vs desktop

Ovisno od računalnika: HPC vs namizni

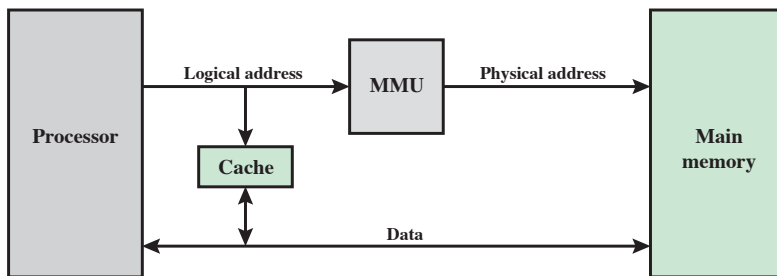
Cache addresses - Virtual memory / Naslavljanje - Navidezni naslovi

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

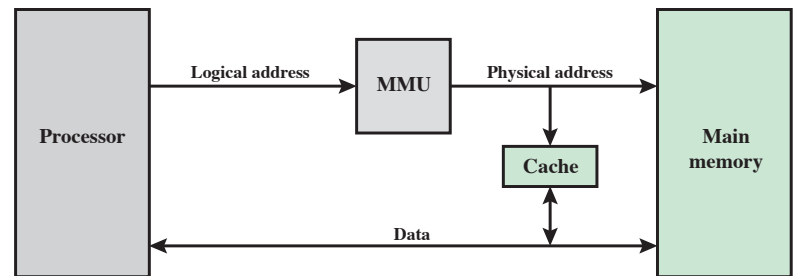
- Facility that allows programs to address memory from a logical point of view, without regard to the amount of main memory physically available
- When used, the address fields of machine instructions contain virtual addresses
- For reads to and writes from main memory, a hardware memory management unit (MMU) translates each virtual address into a physical address in main memory
- Koncept ki programom omogoča naslavljanje pomnilnika, ne glede na količino fizično razpoložljivega glavnega pomnilnika
- Naslovi strojnih ukazov vsebujejo navidezne naslove
- Za branje iz in pisanje v glavni pomnilnik enota za upravljanje pomnilnika (MMU) prevede vsak navidezni naslov v fizični naslov v glavnem pomnilniku

Logical (virtual) and physical cache / Logični (navidezni) in fizični predpomnilnik

- Where does cache sit?
 - Between processor and virtual memory management unit
 - Between MMU and main memory
- Kje se nahaja predpomnilnik?
 - Med CPU in enoto za upravljanje z virtualnim pomnilnikom (MMU)
 - Med MMU in glavnim pomn.



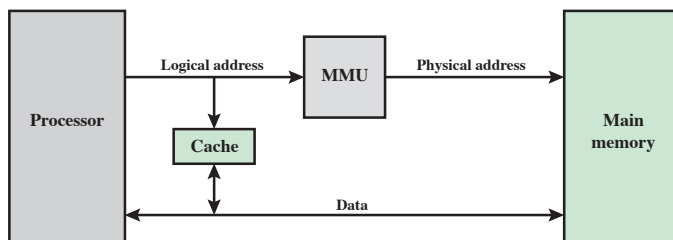
(a) Logical Cache



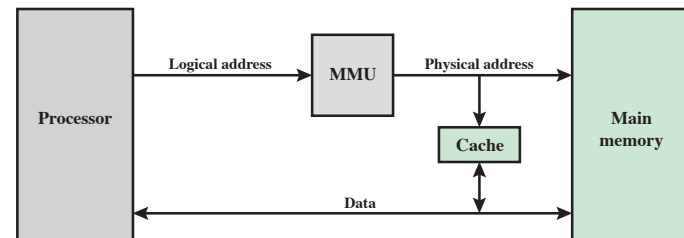
(b) Physical Cache

Cache addressing / Naslavljanje

- Logical cache (virtual cache) stores data using virtual addresses
 - Processor accesses cache directly, not thorough MMU
 - Pro: Cache access faster, before MMU address translation
 - Con: Virtual addresses use same address space for different applications
 - Must flush cache on each context switch or additional bits are used to refer to a particular application
- Physical cache stores data using main memory physical addresses
 - Logični (virtualni) hrani podatke z uporabo virtualnih naslovov
 - CPU dostopa do predpomnilnika neposredno in ne preko MMU
 - Poz: Hitrejši dostop do predpomnilnika, pred prevedbo naslova v enoti MMU
 - Neg: Virtualni naslovi uporabljajo isti naslovni prostor za različne aplikacije
 - Moramo izprazniti predpomnilnik ob vsaki zamenjavi konteksta ali pa uporabiti dodatne bite za določitev različnih programov
 - Fizični predpomnilnik hrani podatke z uporabo fizičnih naslovov glavnega pomnilnika



(a) Logical Cache



(b) Physical Cache

Size does matter / Velikost je pomembna

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

- Cost
 - More cache is expensive
- Speed
 - More cache is faster (up to a point)
 - Checking cache for data takes time
- Size available on the CPU


Cache performance is sensitive to the nature of the workload. It is impossible to arrive at a single “optimum” cache size.

- Cena
 - Narašča z velikostjo predpomnilnika
- Hitrost
 - Več predpomnilnika doprinese k večji hitrosti (a le do določene mere)
 - Iskanje podatkov v predpomnilniku rabi čas
- Prostor, ki je na voljo v CPE

Učinkovitost predpomnilnika je občutljiva na naravo obremenitve. Nemogoče je doseči eno samo "optimalno" velikost predpomnilnika.


Size does matter / Velikost je pomembna

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

	
General information	
Launched	November 5, 2020; 2 years ago
Designed by	AMD
Common manufacturer(s)	TSMC (core die) GlobalFoundries (I/O die)
Cache	
L1 cache	64 KB (per core)
L2 cache	512 KB (per core)
L3 cache	32 MB (per CXX) 64 MB (per 3D V-Cache CCD) 16 MB (in APUs)

Ice Lake	
General Info	
Launched	September 2019 (availability)
Cache	
L1 cache	80 KiB per core (32 instructions + 48 data)
L2 cache	512 KiB per core
L3 cache	Up to 8 MiB, shared

	
General information	
Launched	September 27, 2022; 7 months ago
Designed by	AMD
Common manufacturer(s)	TSMC
Cache	
L1 cache	64 KB (per core)
L2 cache	1 MB (per core)
L3 cache	32 MB (per CCD) 96 MB (per 3D V-Cache CCD) 16 MB (in Phoenix)

	
Alder Lake	
General information	
Launched	November 4, 2021; 17 months ago ^[1]
Marketed by	Intel
Designed by	Intel
Common manufacturer(s)	Intel
Product code	80715
Performance	
Max. CPU clock rate	1.0 GHz to 5.5 GHz, P-cores 700 MHz to 4.0 GHz, E-cores
Cache	
L1 cache	80 KB (32 instructions + 48 data), per P-core 96 KB (64 instructions + 32 data), per E-core
L2 cache	1.25 MB per P-core 2 MB per E-core module
L3 cache	Up to 30 MB, shared

zEC12	
General Info	
Launched	2012
Designed by	IBM
Performance	
Max. CPU clock rate	5.5 GHz
Cache	
L1 cache	64+96 KB/core
L2 cache	1+1 MB/core
L3 cache	48 MB/chip

z15	
General information	
Launched	2019
Designed by	IBM
Performance	
Max. CPU clock rate	5.2 ^[1] GHz
Cache	
L1 cache	128 KB instruction 128 KB data per core
L2 cache	4 MB instruction 4 MB data per core
L3 cache	256 MB shared
Architecture and classification	
Technology node	14 nm ^[1]
Instruction set	z/Architecture
Physical specifications	
Cores	12 ^[1]
History	
Predecessor	z14
Successor	Telum

ARM Cortex-A77	
General Info	
Launched	2019
Designed by	ARM Holdings
Max. CPU clock rate	to 3.0 GHz in phones and 3.3 GHz in tablets/laptops
Cache	
L1 cache	128 KiB (64 KiB I-cache with parity, 64 KiB D-cache) <i>per core</i>
L2 cache	256–512 KiB
L3 cache	1–4 MiB

Historical view of cache sizes of some processors

^a Two values separated by a slash refer to instruction and data caches.

^b Both caches are instruction only; no data caches.

Processor	Type	Year of Introduction	L1 Cachea	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 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
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	1MB	—
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
Intel Core i7 EE 990	Workstaton/ server	2011	6 × 32 kB/32 kB	1.5 MB	12 MB
IBM zEnterprise 196	Mainframe/ Server	2011	24 × 64 kB/ 128 kB	24 × 1.5 MB	24 MB L3 192 MB L4

Mapping function / Preslikovalna funkcija

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

Because there are fewer cache lines than main memory blocks, an algorithm is needed for mapping main memory blocks into cache lines.

Three techniques can be used:

- Direct
- Associative
- Set-associative

Example used for describing the techniques

- Cache of 64kByte
 - Cache lines of 4 bytes
 - 16K (2^{14}) lines of 4 bytes
- 16MBytes main memory
 - 24 bit address
 - ($2^{24}=16M$)

Ker imamo manj vrstic predpomnilnika kot blokov glavnega pomnilnika, potrebujemo algoritem za preslikavo glavnih pomnilniških blokov v predpomnilnik.

Uporabijo se lahko tri tehnike preslikav:

- Neposredna
- Asociativna
- Niz asociativna

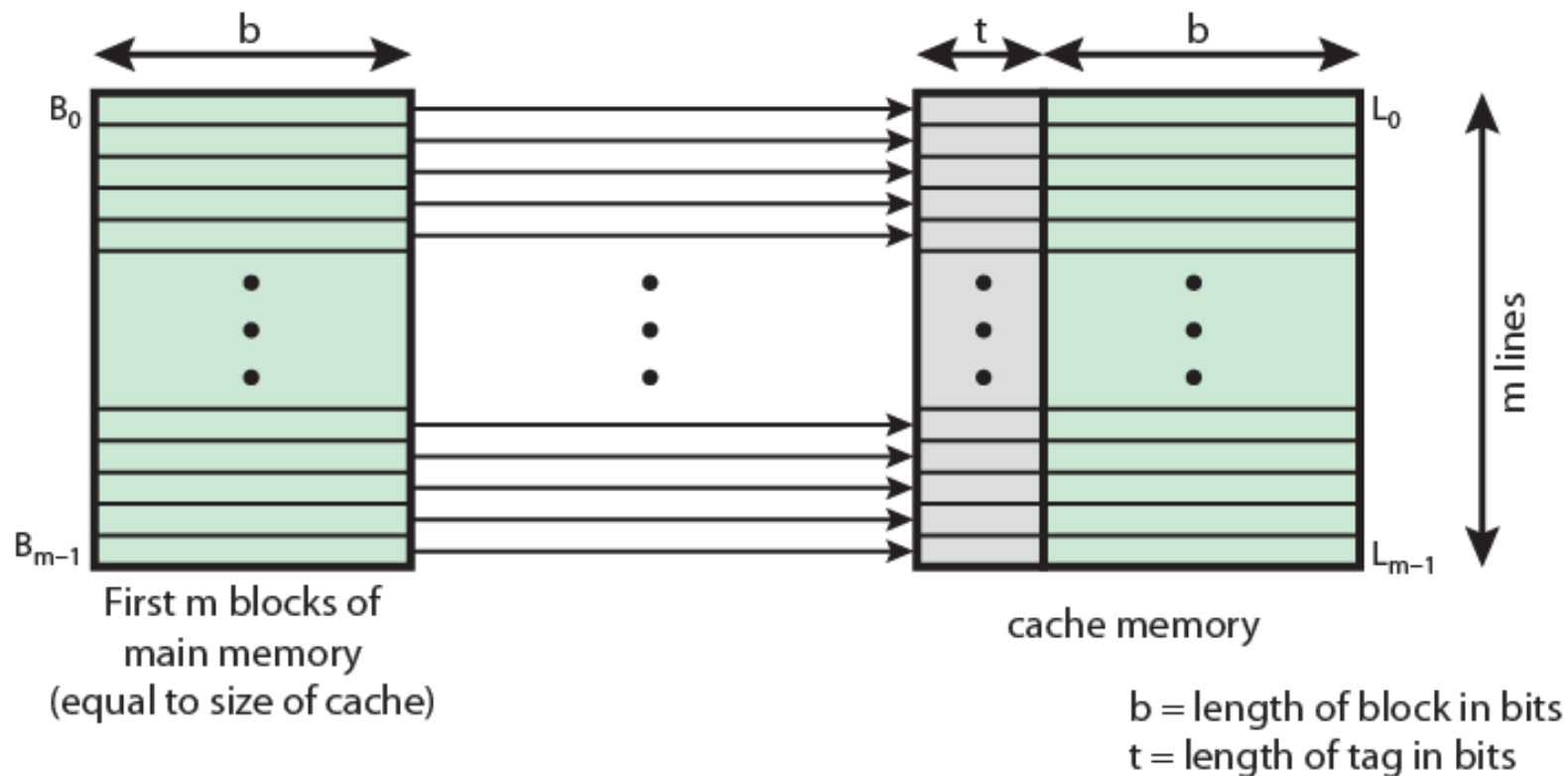
Primer uporabljen za opis tehnik preslikav

- Predpomnilnik velikosti 64kB
 - Velikost vrstice je 4 zloge
 - 16K (2^{14}) vrstic po 4 zloge
- 16MB glavnega pomnilnika
 - 24 bitni naslovi
 - ($2^{24} = 16M$)

Direct mapping / Neposredno preslikovanje

- Each block of main memory maps to only one cache line
 - i.e. if a block is in cache, it must be in one specific place
 - Address is divided in two parts:
 1. Least Significant w bits (LSB) identify unique word
 2. Most Significant s bits specify one memory block. The MSBs are split into
 - a cache line field r and
 - a tag of $s-r$ (most significant)
 - More: a couple of slides later
- Vsak blok pomnilnika se preslika v natančno eno vrstico predpomnilnika
 - Nek blok pomnilnika se vedno nahaja v isti vrstici predpomnilnika
 - Naslov je sestavljen iz dveh delov
 1. Manj pomembnih w bitov (LSB) enolično določajo besedo bloka
 2. Bolj pomembnih s bitov (MSB) določa blok pomnilnika
 - r bitov določa vrstico predpomnilnika
 - $s-r$ bitov je uporabljenih za oznako
 - Več čez nekaj prosojnic

Direct mapping from cache to main memory / Neposredna preslikava iz predpomnilnika v glavni pomnilnik



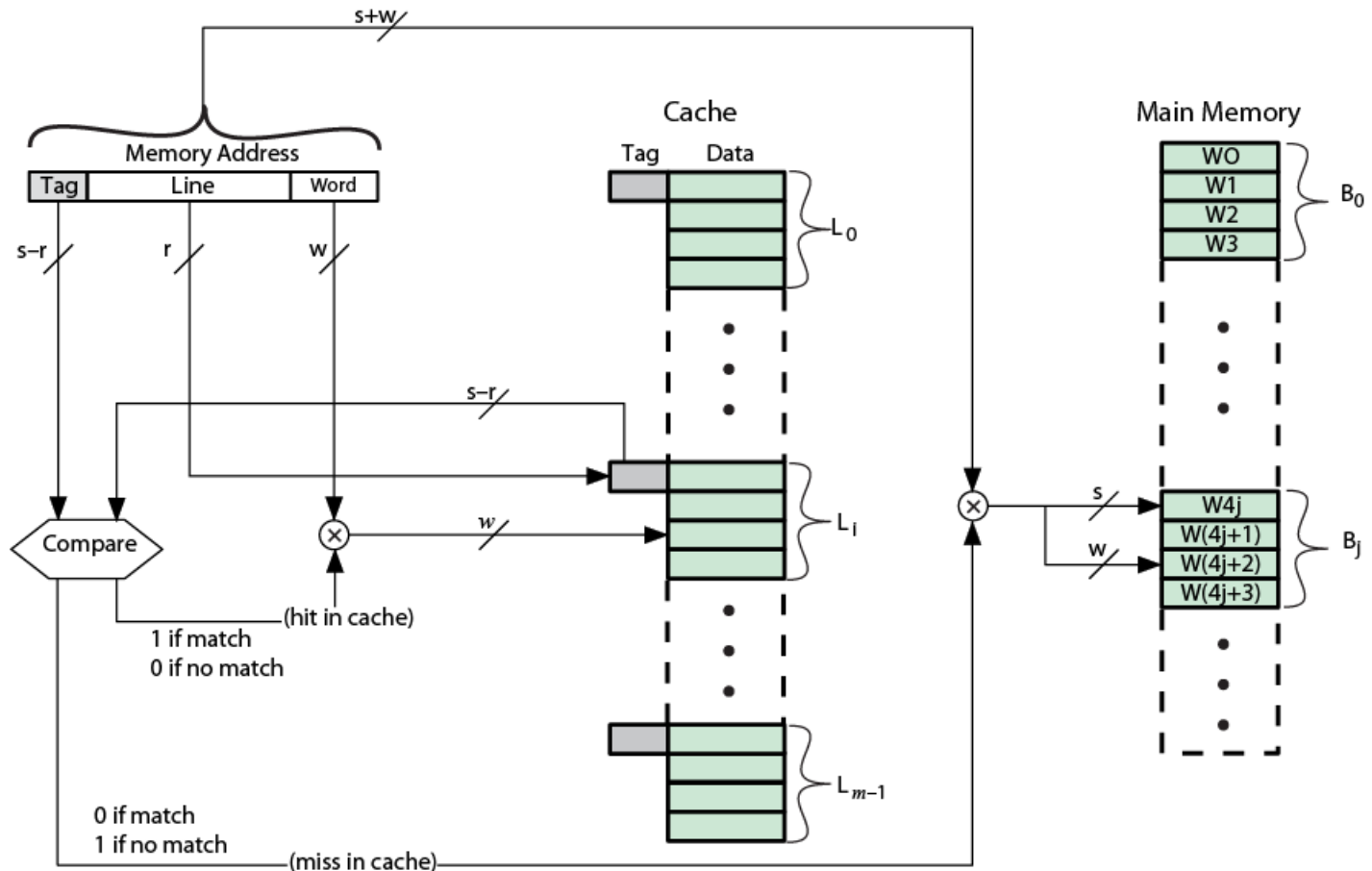
$$i = j \text{ modulo } m$$

Direct mapping- cache line table /

Neposredno preslikovanje – predpomnilniška tabela vrstic

Cache line / Vrstica predpomnilnika	Main Memory blocks held / Pripadajoči bloki glavnega pomnilika
0	0, m, 2m, 3m, ... , $2^s - m$
1	1, m+1, 2m+1, ... , $2^s - m + 1$
...	
m-1	m-1, 2m-1, 3m-1, ... , $2^s - 1$

Blocks in main memory = 2^s / Blokov v glavnem pomnilniku = 2^s



Address length = $(s + w)$ bits

Number of addressable units = $2^{(s+w)}$ words or bytes

Block size = line size = 2^w words or bytes

Number of blocks in main memory = $2^{(s+w)}/2^w = 2^s$

Number of lines in cache = $m = 2^r$

Size of cache = $2^{(r+w)}$ words or bytes

Size of tag = $(s - r)$ bits

Dolžina naslova = $(s + w)$ bitov

Število naslovljivih enot = $2^{(s+w)}$ besed

Velikost bloka = velikosti vrstice = 2^w besed

Število blokov v glavnem pomnilniku = $2^{(s+w)}/2^w = 2^s$

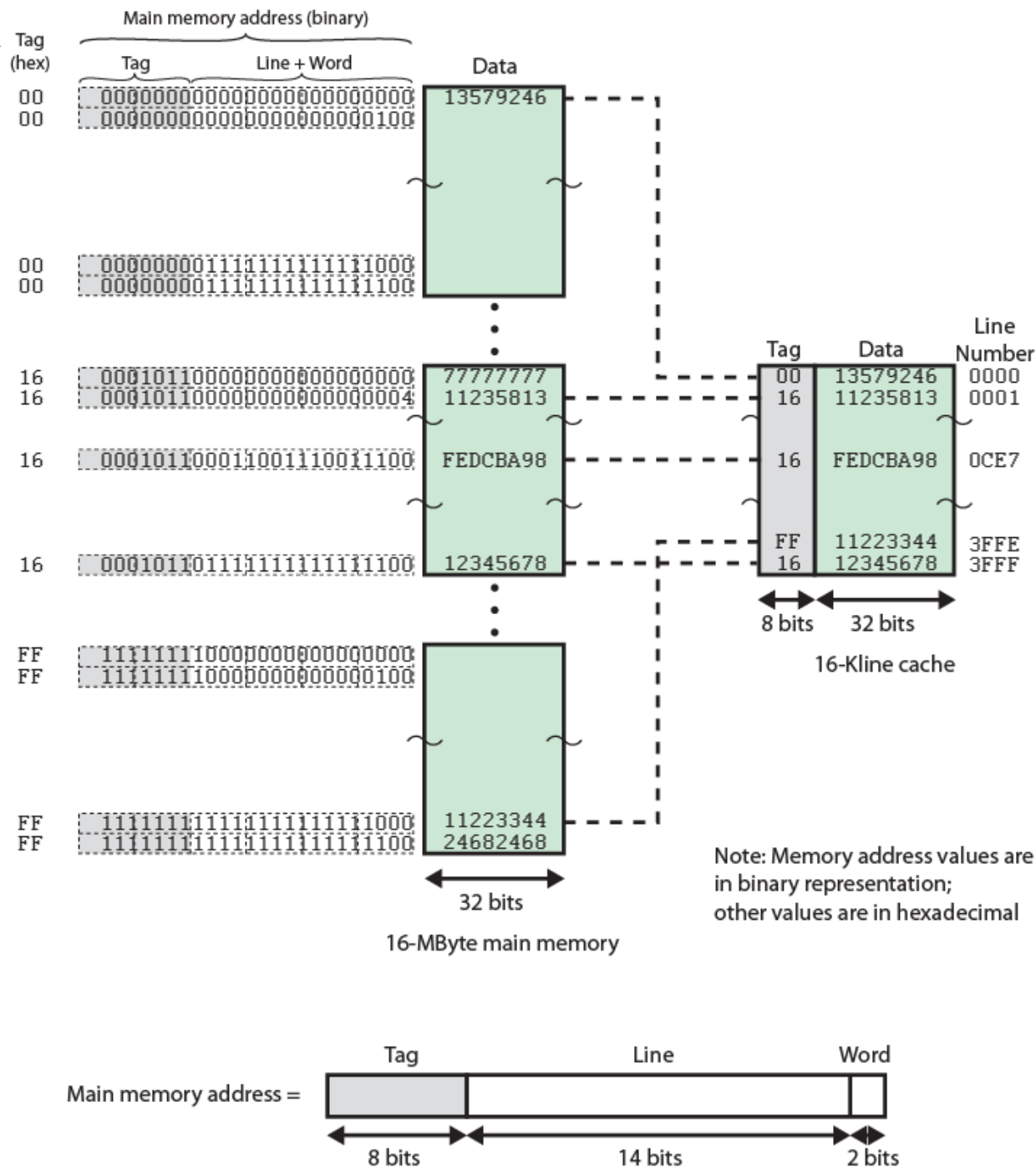
Število vrstic v predpomnilniku = $m = 2^r$

Velikost predpomnilnika = $2^{(r+w)}$ besed ali zlogov

Velikost oznake = $(s - r)$ bitov

Direct mapping example / Primer nepos. preslik.

- 24 bit address
 - 2 bit word identifier (4 byte block)
 - 22 bit block identifier
 - 8 bit tag (=22-14)
 - 14 bit slot or line
 - No two blocks in the same line have the same Tag field
 - Check contents of cache by finding line and checking Tag
- 24 bitni naslov
 - 2 bitni identifikator besede (4 zlogi sestavljajo blok)
 - 22 bitni identifikator bloka
 - 8 bitna oznaka (= 22- 14)
 - 14 bitna vrstica
 - Ne obstajata dva bloka, ki bi ob enaki vrstici imela enako oznako
 - Preveri vsebino predpomnilnika z iskanjem vrstice in preverjanjem oznake



Direct mapping pros & cons /

Neposredno preslikovanje – prednosti in slabosti

Pros

- Simple
- Inexpensive

Cons

- Fixed location for given block
 - If a program accesses 2 blocks that map to the same line repeatedly, cache misses are very high

Prednosti

- Enostavno
- Poceni

Slabost

- Fiksna lokacija za dani blok
 - Če program želi doseči dva izmenjujoča bloka iz iste vrstice, potem je cena zgrešitve v predpomnilniku zelo visoka

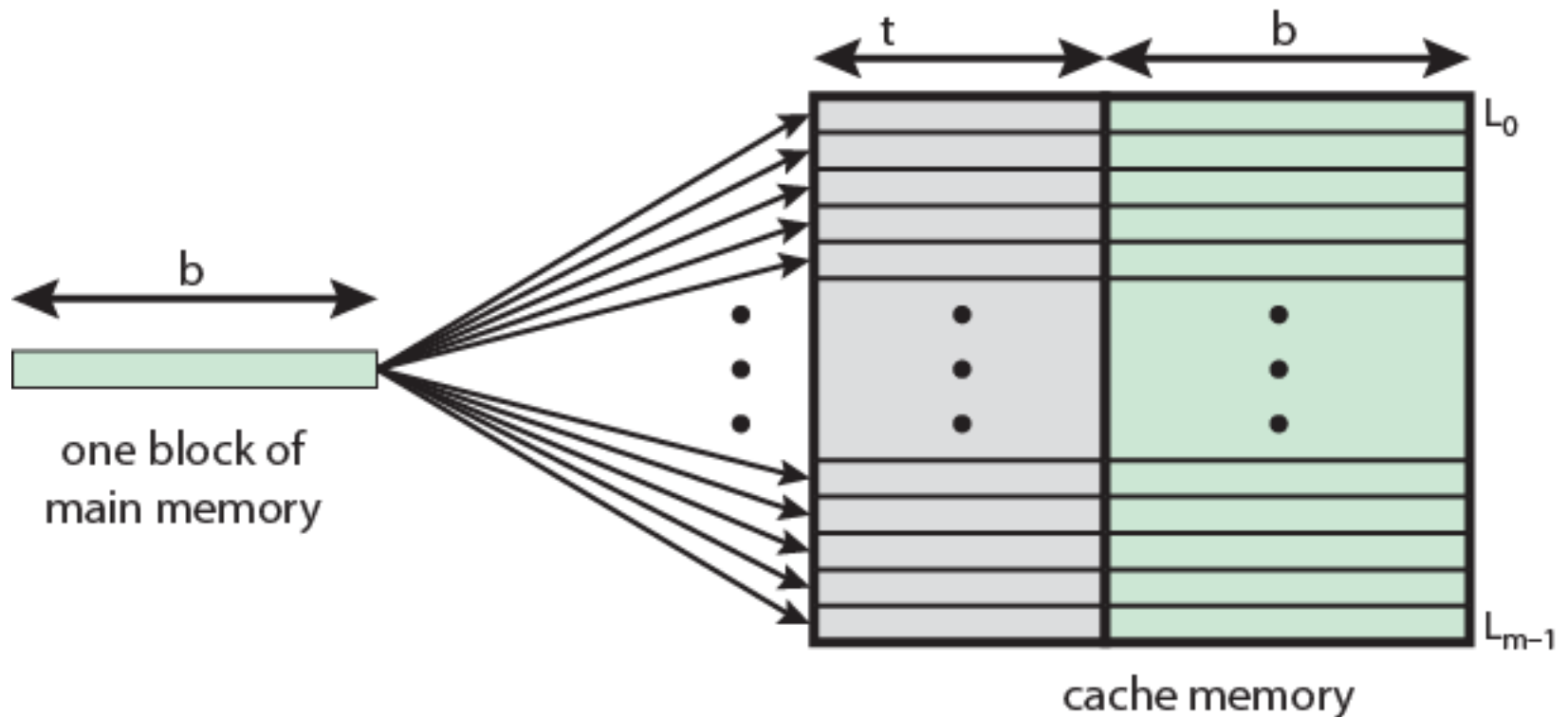
Victim cache / Deložirni predpomnilnik

- Lower miss penalty
- Remember what was discarded
 - Already fetched
 - Use again with little penalty
- Fully associative
- 4 to 16 cache lines
- Between direct mapped L1 cache and next memory level
- Zmanjša kazni zgrešitve
- Zapomni si kaj je bilo deložirano
 - Že prevzeto
 - Ponovna uporaba ob majhni kazni
- Polno asociativni predpomnilnik
- Velik 4 do 16 predpomnilniških vrstic
- Se nahaja med L1 predpomnilnikom in naslednjim pomnilniškim nivojem

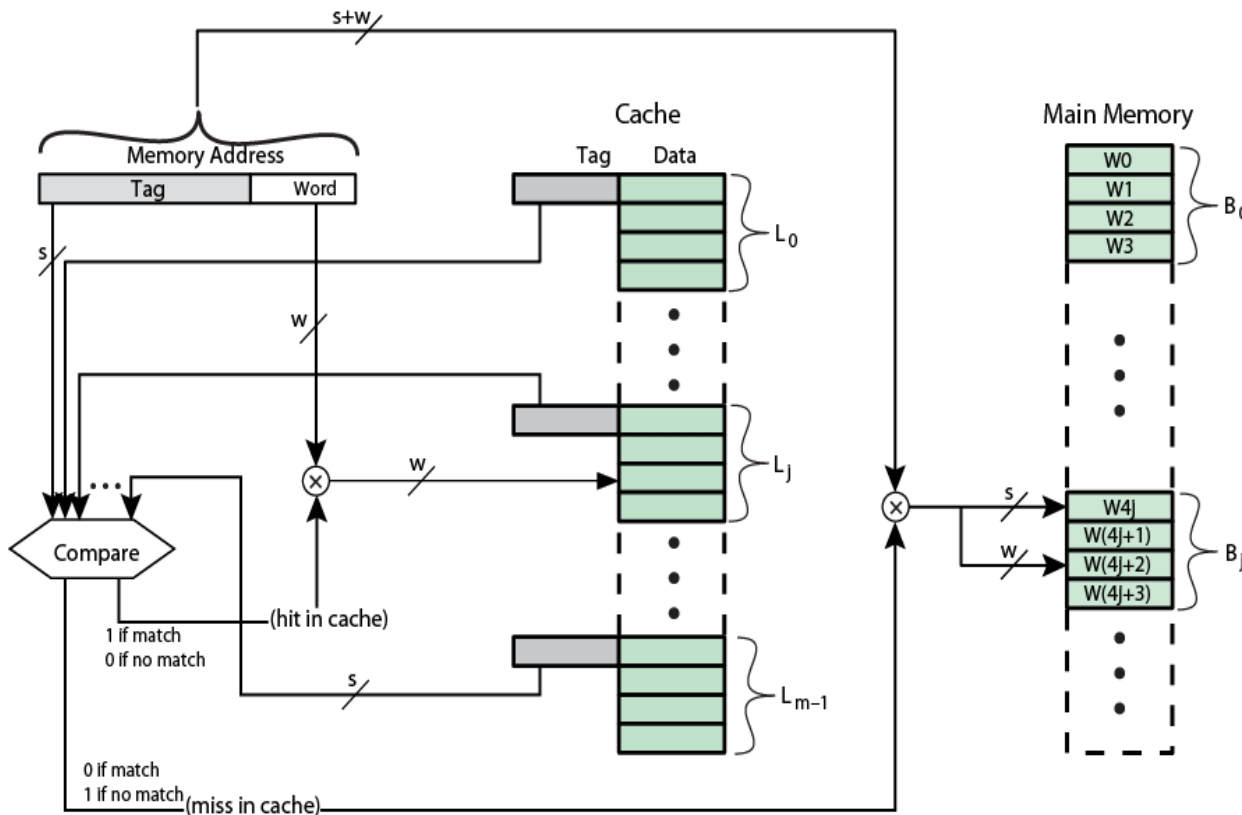
Associative mapping / Asociativno preslikovanje

- A main memory block can load into any line of cache
- Memory address is interpreted as tag and word
- Tag uniquely identifies block of memory
- Every line's tag is examined for a match
- Cache searching gets expensive
- Blok iz glavnega pomnilnika se lahko naloži v katerokoli vrstico predpomnilnika
- Pomnilniški naslov je predstavljen z oznako in besedo
- Oznaka enolično določa blok pomnilnika
- Oznaka vsake vrstice se preveri pri iskanju ujemanja
- Iskanje po predpomnilniku je zaradi tega drago

Associative mapping from cache to main memory /
Asociativno preslikovanje iz predpomnilnika v glavni pomnilnik



Fully associative cache organization / Organizacija polno asociativnega predpomnilnika



- Address length = $(s + w)$ bits
- Number of addressable units = $2^{(s+w)}$ words or bytes
- Block size = line size = 2^w words or bytes
- Number of blocks in main memory = $2^{(s+w)} / 2^w = 2^s$
- Number of lines in cache = undetermined
- Size of tag = s bits

- Dolžina naslova = $(s + w)$ bitov
- Število naslovljivih enot = $2^{(s+w)}$ besed
- Velikost bloka = velikost vrstice = 2^w besed
- Število blokov v glavnem pomnilniku = $2^{(s+w)} / 2^w = 2^s$
- Število vrstic v predpomnilniku = nedoločeno
- Velikost oznake = s bitov

Diagram illustrating a 16 Kline Cache and 16 MByte Main Memory. The cache is a 16-line structure, and the main memory is 16 MByte.

Cache Structure:

Line Number	Tag (22 bits)	Data (32 bits)
0000	3FFFFE	11223344
0001	058CE7	FEDCBA98
...
3FFD	3FFFFD	33333333
3FFE	000000	13579246
3FFF	3FFFFF	24682468

Main Memory Structure:

Address (hex)	Data (32 bits)
000000	13579246
...	...
24682468	24682468
...	...
33333333	33333333
11223344	11223344
24682468	24682468

Cache Line 3FFFFF (Current State):

Field	Value
Data	24682468
Cache line	3FFFFF

Cache Line 3FFFFF (Target State):

Field	Value
Data	24682468
Cache line	3FFFFF

Main Memory Address (Binary):

Tag (hex)	Tag (binary)	Word (binary)	Data (hex)
000000	000000000000000000000000	000000000000000000000000	13579246
000001	000000000000000000000000	000000000000000000000100	

Cache Line 058CE6 (Current State):

Field	Value
Data	058CE6
Cache line	058CE6

Cache Line 058CE7 (Current State):

Field	Value
Data	058CE7
Cache line	058CE7

Cache Line 058CE8 (Current State):

Field	Value
Data	058CE8
Cache line	058CE8

Main Memory Address (Binary):

Tag (hex)	Tag (binary)	Word (binary)	Data (hex)
333333	111111111111111111111111	111111111111111111111111	33333333
112233	111111111111111111111111	111111111111111111111111	11223344
246824	111111111111111111111111	111111111111111111111111	24682468

Note: Memory address values are in binary representation; other values are in hexadecimal.

Note: Memory address values are in binary representation; other values are in hexadecimal



Set associative mapping /

Niz asociativna preslikava

- Cache is divided into sets (ways)
- Each set contains a number of lines (k)

Two types:

- k-way set-associative mapping
 - A given block maps to any line in a given set. E.g. block B can be in any line of set I
 - typically used for higher degrees of associativity (high number of sets)
- k-way associative direct mapping
 - The first k lines of main memory are direct mapped into the k lines of each way
 - The direct-mapped implementation is typically used for small degrees of associativity (small values of ways-sets)

- Predpomnilnik je razdeljen na nize
- Vsak niz vsebuje določeno število vrstic (k)

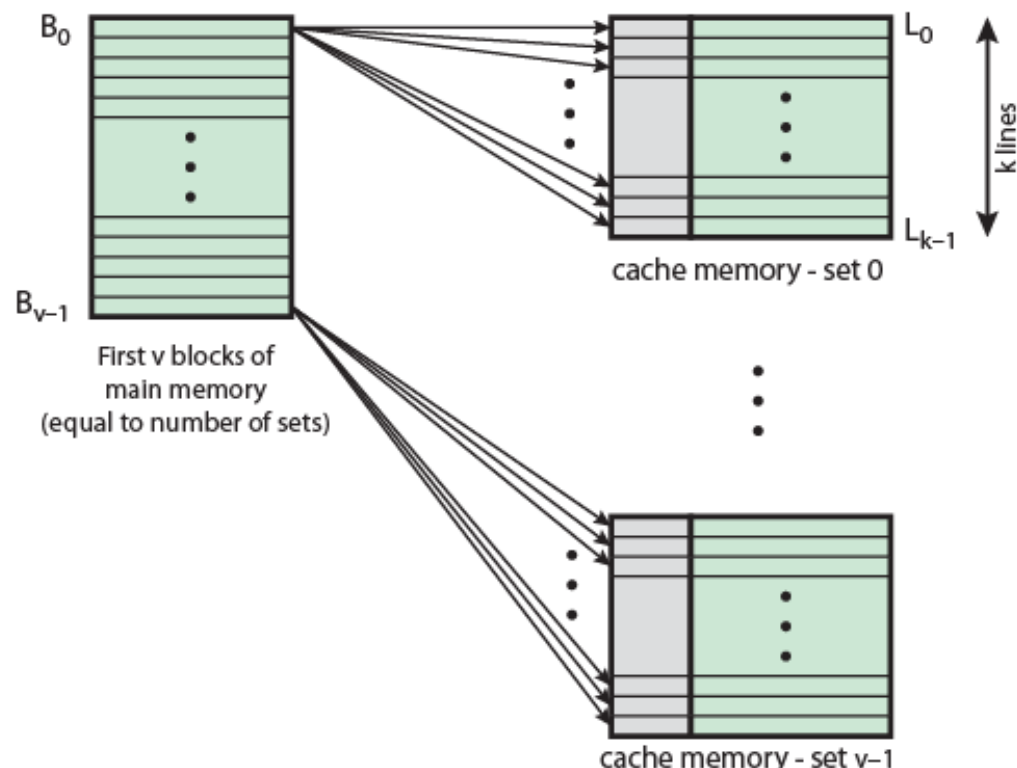
Dva tipa:

- k-niz niz-asociativna preslikava
 - Dani blok se lahko preslika v katerikoli vrstico pripadajočega niza. Npr. blok B je lahko v katerikoli vrstici niza I
 - Običajno se uporablja za višje stopnje asociativnosti (visoko število nizov)
- k-niz neposredna preslikava
 - Prvih k vrstic pomnilnika je neposredno preslikanih v k vrstic vsakega niza
 - Neposredno preslikana izvedba se običajno uporablja za majhne stopnje asociativnosti (majhno število nizov)

Mapping from main memory to cache: k-way set-associative / Asociativna preslikava v k -ti niz

A given block maps to any line in a given set.

Dani blok se lahko preslika v katerokoli vrstico pripadajočega niza.



$$i = j \text{ modulo } v$$

i = cache set number

j = main memory block number

m = number of lines in the cache

v = number of sets

k = number of lines in each set

v = number of sets

k = number of lines in each set

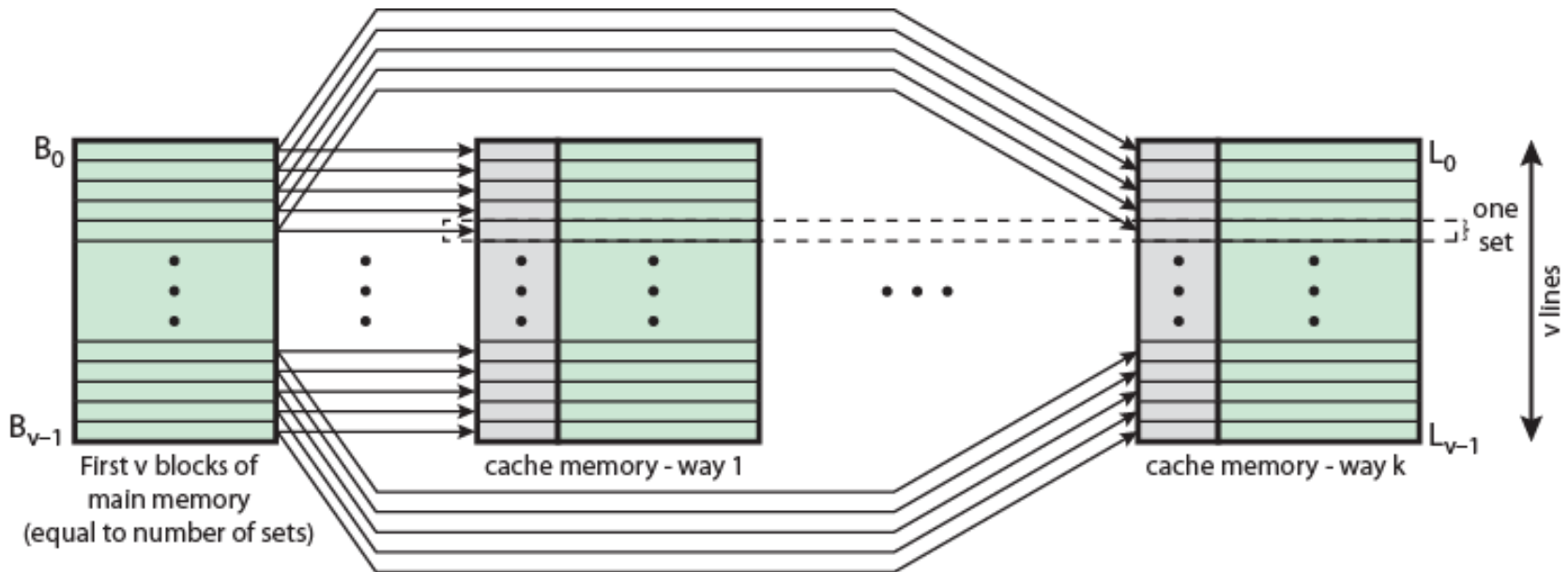
v = number of sets

k = number of lines in each set

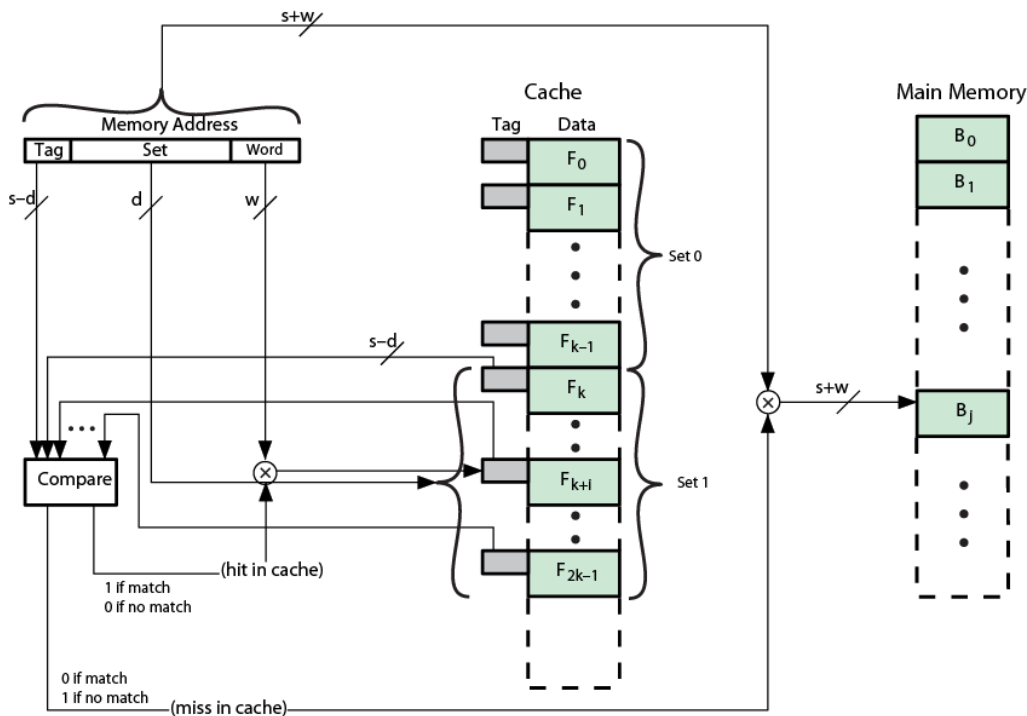
Direct mapping from main memory to cache: k-way Associative / Neposredna preslikava v niz dolžine k

The first k lines of main memory are
direct mapped into the k lines of each
way

Prvih k vrstic pomnilnika je neposredno
preslikanih v k vrstic vsakega niza



k -way set associative cache organization / Organizacija niz-asociativnega predpomnilnika s k izbirami



- Address length = $(s + w)$ bits
- Number of addressable units = $2^{(s+w)}$ words or bytes
- Block size = line size = 2^w words or bytes
- Number of blocks in main memory = $2^{(s+w)}/2^w = 2^s$
- Number of lines in set = k
- Number of sets = $v = 2^d$
- Number of lines in cache = $m = kv = k * 2^d$
- Size of cache = $k * 2^{(d+w)}$ words or bytes
- Size of tag = $(s - d)$ bits

Dolžina naslova = $(s + w)$ bit

Število naslovljivih enot = $2^{(s+w)}$ besed ali bajtov

Velikost bloka = velikost vrstice = 2^w besed ali bajtov

Število blokov v glavnem pomnilniku = $2^{(s+w)}/2^w = 2^s$

Število vrstic v nizu = k

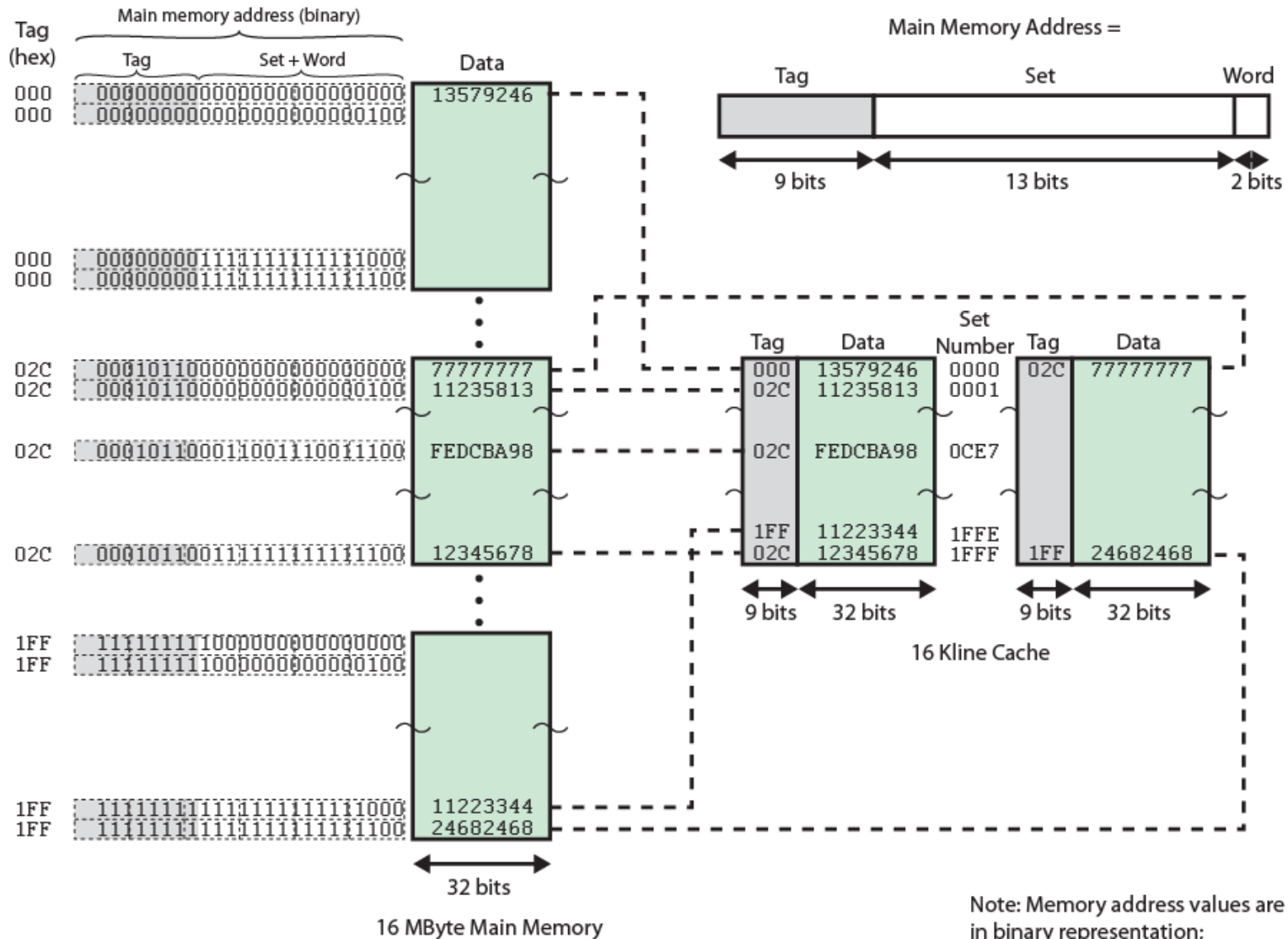
Število nizov = $v = 2^d$

Število vrstic v predpomnilniku = $m = kv = k * 2^d$

Velikost predpomnilnika = $k * 2^{(d+w)}$ besed ali bajtov

Velikost oznake = $(s - d)$ bitov

Two-way set associative mapping- example / Dva-niz set asociativna preslikava- primer



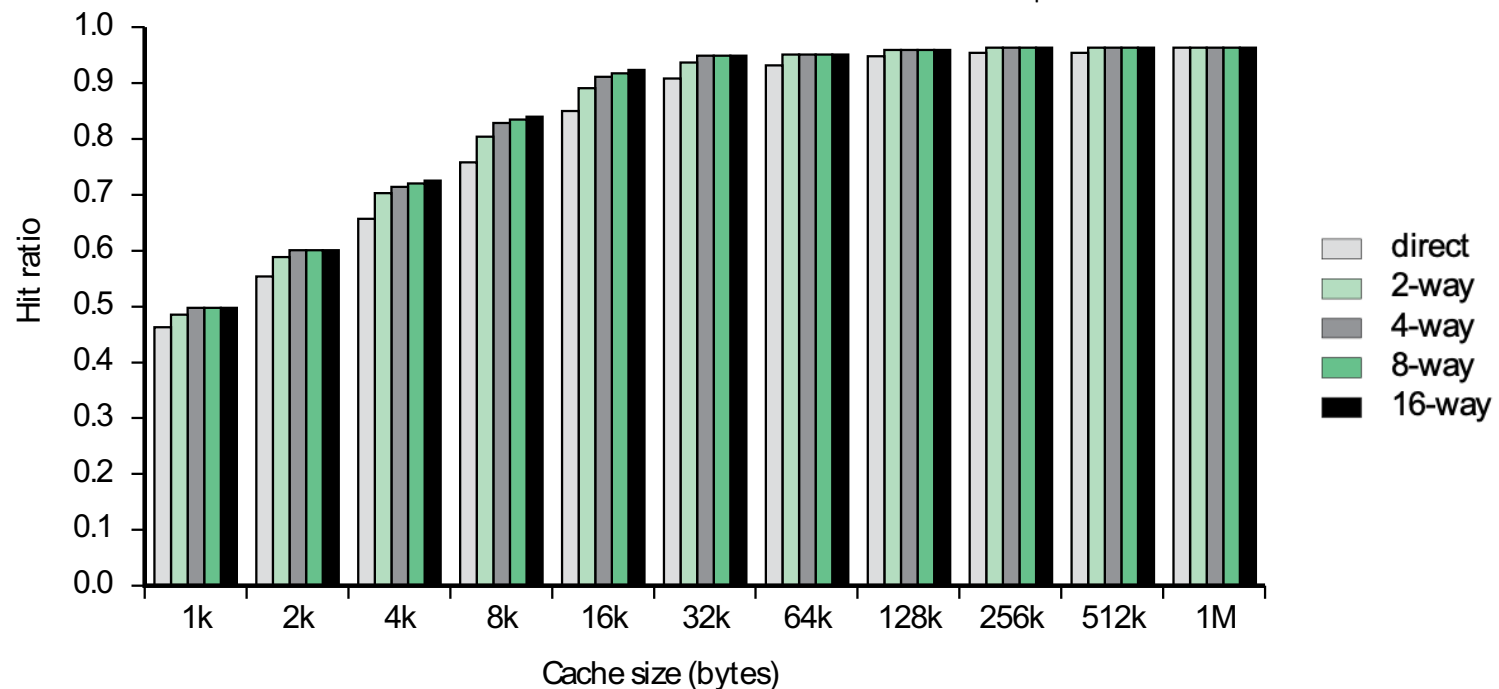
Note: Memory address values are in binary representation; other values are in hexadecimal

Examples / Različni primeri

- $v = \text{number of sets}$
- $k = \text{number of lines in a set}$
- $m = \text{number of all lines}$
- $v = m, k = 1$ set-associative reduces to direct mapping
- $v = 1, k = m$ set-associative reduces to associative mapping
- Most common is two lines per set $v = m/2, k = 2$
- Four-way set associative ($v = m/4, k = 4$) modest improvement for a small cost
- Further increases have little effect
- $v = \text{število nizov}$
- $k = \text{število vrstic v nizu}$
- $m = \text{število vseh vrstic}$
- $v = m, k = 1$ niz-asociativna postane direktna preslikava
- $v = 1, k = m$ niz-asociativna postane asociativna preslikava
- Najpogosteje sta uporabljeni dve vrstici na niz $v = m / 2, k = 2$
- 4-niz niz-asociativna preslikava ($v = m / 4, k = 4$) prinese skromno izboljšavo za majhne stroške
- Nadaljnja povečanja št. nizov ima majhen učinek

Direct and set associative cache - performance differences / Razlike v učinkovitosti neposrednega in asociativnega predpomnilnika

- Significant up to at least 64kB for 2-way
- Difference between 2-way and 4-way at 4kB much less than 4kB to 8kB
- Cache complexity increases with associativity
- Not justified against increasing cache to 8kB or 16kB
- Above 32kB gives no improvement
- Signifikantna do vsaj 64kB za niz dolžine 2
- Razlika med nizom dolžine 2 in 4 pri 4kB mnogo manjša kot prehod med 4kB in 8kB
- Kompleksnost predpomnilnika se z višjo asociativnostjo večja
- Glede na povedano smiselno povečanje do 8kB ali 16kB
- Nad 32kB ne pridobimo na učinkovitosti



Replacement algorithms / Zamenjevalni algoritmi

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

- Once the cache has been filled, when a new block is brought into the cache, one of the existing blocks must be replaced
- For direct mapping there is only one possible line for any particular block and no choice is possible
- For the associative and set-associative techniques a replacement algorithm is needed
- To achieve high speed, an algorithm must be implemented in hardware
- Ko je predpomnilnik zapolnjen, je treba povoziti zapolnjeno vrstico z novim blokom iz pomnilnika
- Za neposredno preslikavo obstaja le ena možna vrstica za kateri koli posamezni blok in ni možna izbira
- Za asociativne in niz asociativne preslikave je potreben zamenjevalni algoritem
- Za doseganje visokih hitrosti je algoritem implementiran s strojno opremo

Common replacement algorithms / Pogosti zamenjevalni algoritmi

- Least recently used (LRU)
 - Most effective
 - Replace that block in the set that has been in the cache longest with no reference to it (USE bit)
 - Because of its simplicity of implementation, LRU is the most popular replacement algorithm
- First-in-first-out (FIFO)
 - Replace that block in the set that has been in the cache longest
 - Easily implemented as a round-robin or circular buffer technique
- Least frequently used (LFU)
 - Replace that block in the set that has experienced the fewest references
 - Could be implemented by associating a counter with each line
- Random
- Najdlje neuporabljen (LRU)
 - Najbolj učinkovita
 - Zamenjaj blok v nizu, ki je bil najdlje v predpomnilniku, brez sklica nanj
 - Zaradi enostavnosti izvajanja je LRU najbolj priljubljen algoritem
- Prvi pride prvi gre (FIFO)
 - Zamenjaj blok, ki je v predpomnilniku najdlje časa
 - Implementacija z round-robin ali krožna tehnika
- Najmanj pogosto uporabljen (LFU)
 - Zamenjaj blok, ki je imel najmanj zadetkov
 - Implementacija s števcem za vsako vrstico
- Naključno

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

Write policy / Strategija pisanja

When a block in the cache is to be replaced, and it changed we must not overwrite a cache block unless main memory is up to date.

Ko je treba blok v predpomnilniku zamenjati in se je spremenil, ne smemo prepisati bloka predpomnilnika dokler tudi glavni pomnilnik ni posodobljen.

Problems

- I/O may address main memory directly and change a block that is in the cache
- I/O may need a block that has changed in the cache and written back to MM
- Multiple CPUs may have individual caches and a block may be in more than one of them and one cache changes

Težave

- V/I lahko neposredno naslovi glavni pomnilnik in spremeni blok, ki je v predpomnilniku
- V/I potrebuje blok, ki je bil spremenjen v predpomnilniku in še ni bil zapisan v glavni pomnilnik
- Več CPE ima lahko ločene predpomnilnike in isti blok se lahko nahaja v več kot enem od in en predpomnilnik se spremeni

Several write policies exist

Obstaja več politik pisanja

Write through / Pisanje skozi

- All writes go to main memory as well as cache
- Multiple CPUs can monitor main memory traffic to keep local (to CPU) cache up to date
- Lots of traffic
- Slows down writes
- Vsa pisanja se izvajajo v glavni pomnilnik kot tudi v predpomnilnik
- Več CPU enot lahko spremlja promet glavnega pomnilnika, tako da ima vsak CPU svoj predpomnilnik usklajen z glavnim pomnilnikom
- Veliko prometa
- Upočasnitev pisanja

Write back / Pisanje nazaj

- Updates initially made in cache only
- Update (dirty) bit for cache slot is set when update occurs
- If block is to be replaced, write to main memory only if update bit is set
- Posodobitve so najprej narejene samo v predpomnilniku
- Ob posodobitvi vrstice/bloka se postavi umazan bit
- Če je blok v vrstici potrebno zamenjati, se ga prepiše v glavni pomnilnik samo v primeru postavljenega umazanega bita

Problems

- Other caches get out of sync
- I/O must access main memory through cache
- 15% of memory references are writes (up to 50% for HPC)

Težave

- Ostali predpomnilniki niso več usklajeni
- I/O mora dostopati do glavnega pomnilnika preko predpomnilnika (poveča kompleksnost)
- 15% pomnilniških referenc obsega pisanje (HPC do 50%)

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

Line Size / Velikost vrstice

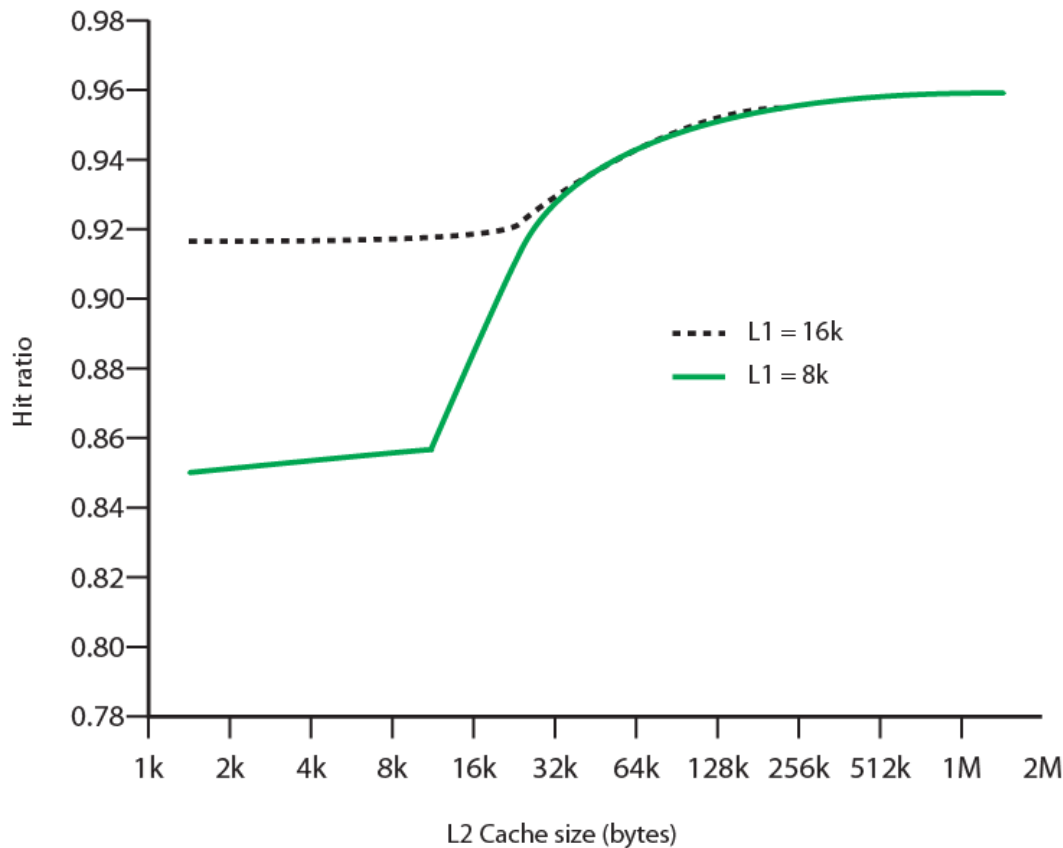
- Retrieve not only desired word but a number of adjacent words as well
- Increased block size will increase hit ratio at first
 - The principle of locality
- Hit ratio will decrease as block becomes even bigger
- Larger blocks
 - Reduce number of blocks that fit in cache
 - Data overwritten shortly after being fetched
 - Each additional word is less local so less likely to be needed
- No definitive optimum value has been found
 - 8 to 64 bytes seems reasonable
 - For HPC systems, 64- and 128-byte most common
- Ne preberemo/zapišemo samo želeno besedo ampak tudi vse sosednje znotraj bloka
- Povečana velikost bloka bo sprva povečala verjetnost zadetka
 - Lastnost lokalnosti
- Verjetnost zadetka se zmanjša, ko se velikost bloka povečuje
- Večji bloki
 - Zmanjša se število blokov, ki „pašejo“ v predpom.
 - Bloki so prepisani kmalu po tem, ko jih prevzamemo
 - Vsaka nadaljnja posamezna beseda je manj lokalna in zato je manj verjetno, da bo potrebvana
- Optimalna vrednost še ni bila najdena
 - Smiselna velikost je od 8 do 64 zlogov
 - Za HPC sisteme so običajne velikosti od 64 do 128 zlogov

Multilevel caches / Večnivojski predpomnilniki

Cache Addresses	Write Policy
Logical	Write through
Physical	Write back
Cache Size	Line Size
Mapping Function	Number of caches
Direct	Single or two level
Associative	Unified or split
Set Associative	
Replacement Algorithm	
Least recently used (LRU)	
First in first out (FIFO)	
Least frequently used (LFU)	
Random	

- High logic density enables caches on chip
 - Faster than bus access
 - Frees bus for other transfers
- Common to use both on and off chip cache
 - L1 on chip, L2 off chip in static RAM
 - L2 access much faster than DRAM or ROM
 - L2 often uses separate data path
 - L2 may now be on chip
 - Resulting in L3 cache
 - Bus access or now on chip...
- Visoka gostota logike omogoča postavitev predpomnilnikov na CPE
 - Dostop je hitrejši kot preko vodil
 - Vodilo se lahko uporablja za druge prenose
- Pogosto se je predpomnilnik uporabil tako na kot izven čipa
 - L1 na čipu, L2 ni na čipu kot statičen RAM
 - L2 dostop mnogo hitrejših kot do dinamičnega RAMa ali ROMa
 - L2 uporablja ločene podatkovne poti in ne systemskega vodila
 - L2 je dandanes že na čipu
 - Tako pridemo do L3 predpomnilnika
 - Dostop preko zunanega vodila, dandanes že na čipu ...

Hit Ratio (L1 & L2) for 8 kbytes and 16 kbyte L1 / Razmerje zadetkov (L1 & L2) Za 8 kB in 16 kB L1



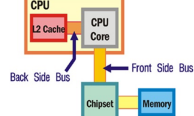
- A simulation study of two-level cache performance
- Assumption: both caches have the same line size
- A hit is counted if the desired data appears in either the L1 or the L2 cache
- L2 has little effect on the total number of cache hits until it is at least double the L1
- If the L2 cache has the same line size and capacity as the L1 cache, its contents will more or less mirror those of the L1 cache.

-
- Simulacijska študija zmogljivosti dveh ravni predpomnilnika
 - Predpostavka: oba predpomnilnika imata enako velikost vrstice
 - Zadelek: če se želeni podatki pojavijo v predpomnilniku L1 ali L2
 - L2 ima majhen učinek na število zadetkov, dokler ni vsaj dvakrat večji od L1
 - Če ima L2 enako velikost in dolžino vrstice kot L1, bo njegova vsebina bolj ali manj odražala vsebino L1.

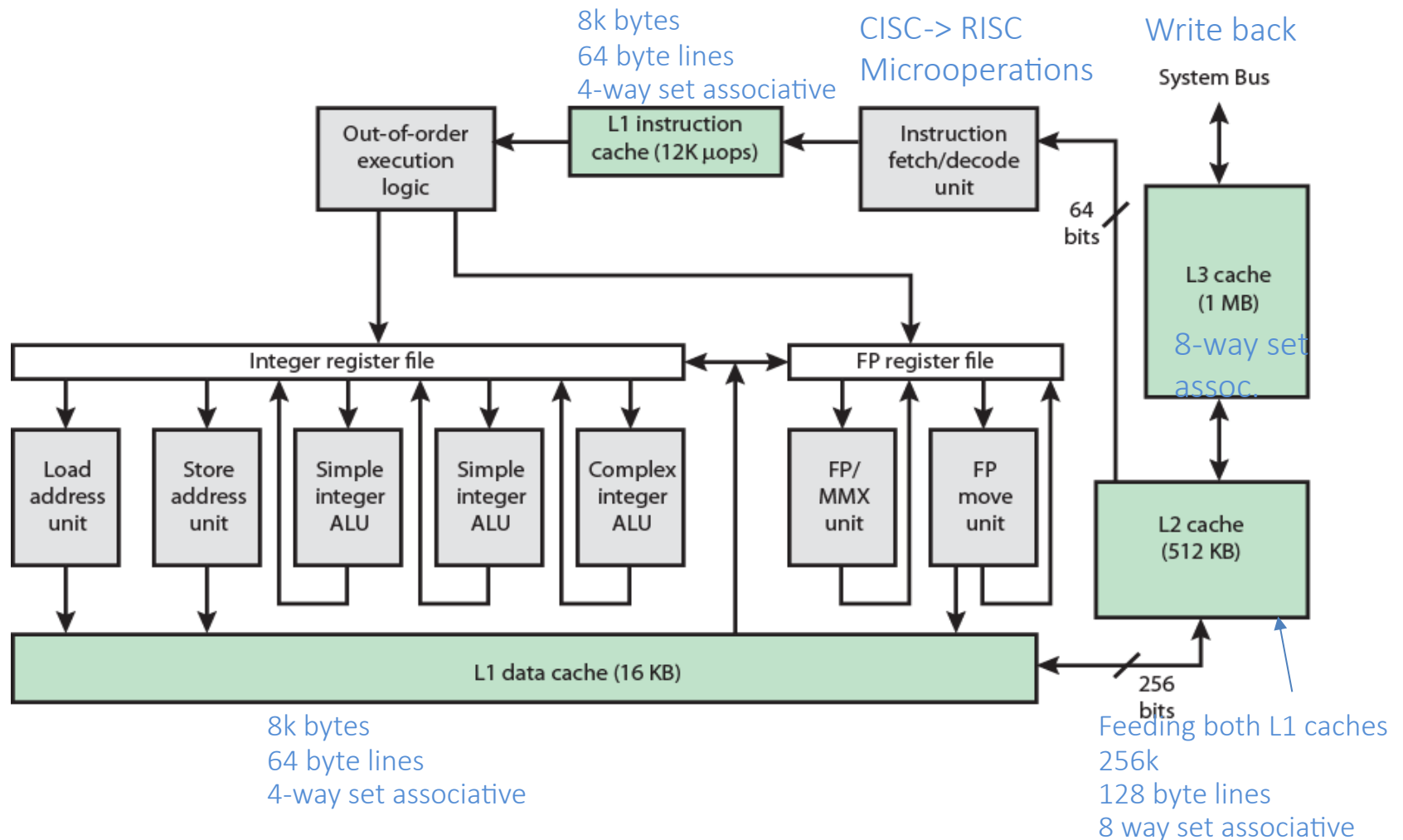
Unified vs split caches / Enoten ali razdeljen predpomnilnik

- One cache for data and instructions or two, one for data and one for instructions
- Advantages of unified cache
 - Higher hit rate
 - Balances load of instruction and data fetch
 - Only one cache to design & implement
- Advantages of split cache (Harvard)
 - Eliminates cache contention between instruction fetch/decode unit and execution unit
 - Important in pipelining (superscalar)
- En predpomnilnik za podatke in ukaze ali dva predpomnilnika (en za podatke, drugi za ukaze)
- Prednosti enotnega predpomnilnika (von Neuman)
 - Višja verjetnost zadetka
 - Uravnava „obremenitev“ prevzemanja ukazov in podatkov (v primeru, da je enih ali drugih več)
 - Samo en predpomnilnik, ki ga je potrebno izvesti
- Prednosti razdeljenega predpomnilnika (Harvard)
 - Odstrani tekmovanje med enoto za prevzemanje ukazov in izvajalno enoto (podatki)
 - Zelo pomembno pri cevovodnosti (superskalarni računalniki)

Intel cache evolution / Razvoj predpomnilnika Intel

Problem	Solution	Processor on which feature first appears
External memory slower than the system bus.	Add external cache using faster memory technology.	386
Increased processor speed results in external bus becoming a bottleneck for cache access.	Move external cache on-chip (8kB with each line 16B), operating at the same speed as the processor.	486
Internal cache is rather small, due to limited space on chip	Add external L2 cache using faster technology than main memory	486
Contention occurs when both the Instruction Prefetcher and the Execution Unit simultaneously require access to the cache. In that case, the Prefetcher is stalled while the Execution Unit's data access takes place.	Create separate L1 data and instruction caches.	Pentium
Increased processor speed results in external bus becoming a bottleneck for L2 cache access.	Create separate back-side bus that runs at higher speed than the main (front-side) external bus. The BSB is dedicated to the L2 cache.	Pentium Pro 
	Move L2 cache on to the processor chip.	Pentium II
Some applications deal with massive databases and must have rapid access to large amounts of data. The on-chip caches are too small.	Add external L3 cache.	Pentium III
	Move L3 cache on-chip.	Pentium 4

Pentium 4 block diagram / blokovna shema (diagram)



Pentium 4 Core Processor

- Fetch/Decode Unit
 - Fetches instructions from L2 cache
 - Decode into micro-ops
 - Store micro-ops in L1 cache
- Out of order execution logic
 - Schedules micro-ops
 - Based on data dependence and resources
 - May speculatively execute
- Execution units
 - Execute micro-ops
 - Data from L1 cache
 - Results in registers
- Memory subsystem
 - Include L2, L3 cache and systems bus
- Enota za prevzem in dešifriranje
 - Prevzema ukaze iz predpom. L2
 - Jih dešifrira v mikro ukaze
 - Shranjuje mikro ukaze v predpom. L1
- Logika izvrševanja ukazov
 - Razporeja mikro ukaze
 - Glede na odvisnost od podatkov in virov
 - Lahko špekulativno izvrši
- Izvršilne enote
 - Izvajajo mikro ukaze
 - Podatki iz predpomnilnika L1
 - Rezultati v registrih
- Podsistem pomnilnika
 - Vključuje L2, L3 predpomnilnika in sistemsko vodilo

Pentium 4 design reasoning

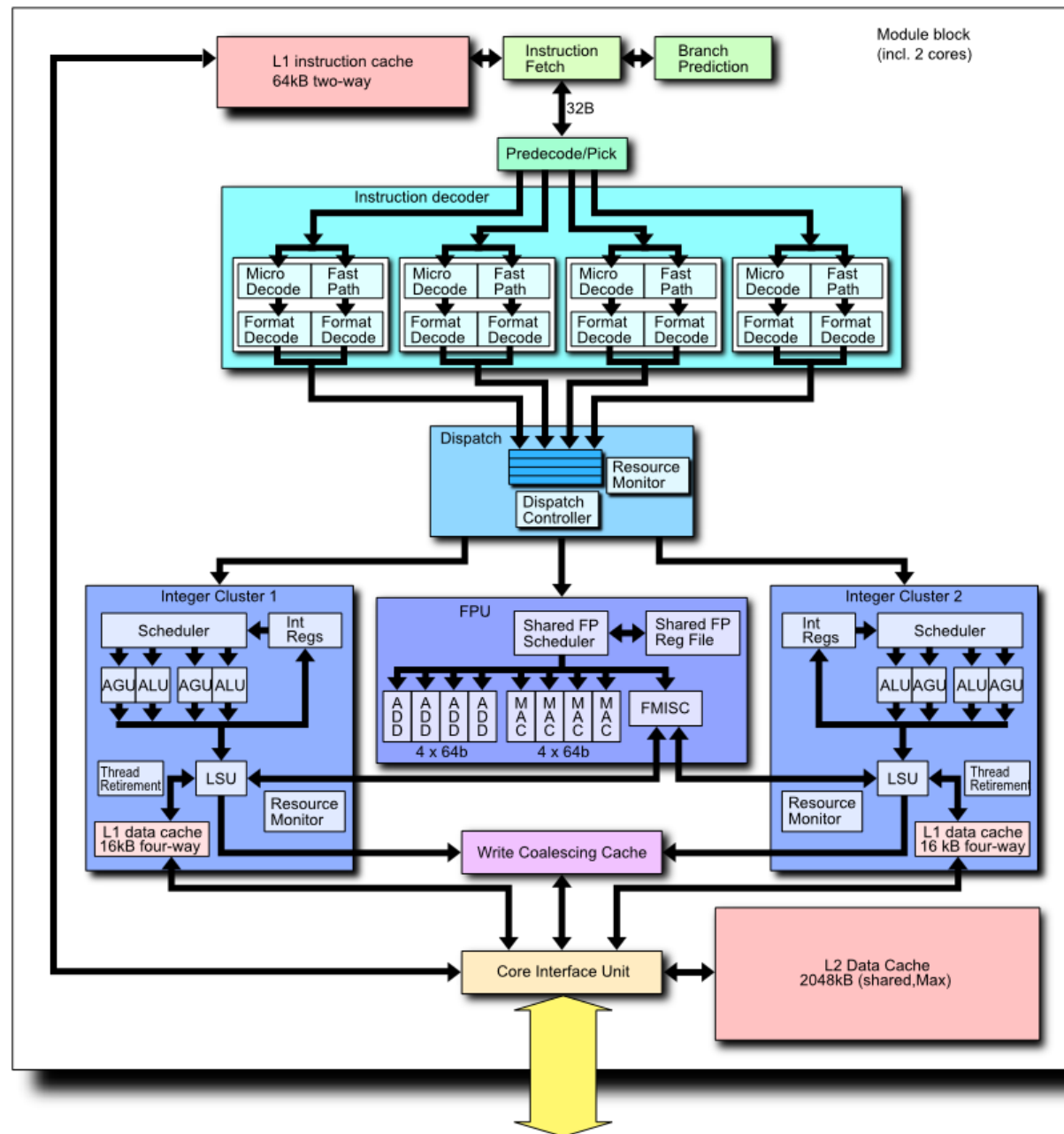
- Decodes instructions into RISC like micro-ops before L1 cache
- Micro-ops fixed length
 - Superscalar pipelining and scheduling
- Pentium instructions long & complex
- Performance improved by separating decoding from scheduling & pipelining
- Data cache is write back
 - Can be configured to write through
- Dešifrira ukaze v RISC mikro ukaze pred predpomnilnikom L1
- Mikro ukazi so fiksne dolžine
 - Superskalarni cevovodi in razvrščevanje
- Navodila za Pentium so dolga in zapletena
- Zmogljivost se izboljša z ločevanjem dešifriranje od razvrščanja in cevovoda
- Predpomnilnik podatkov uporablja pisanje nazaj
 - Lahko se konfigurira za pisanje skozi

AMD Bulldozer – Diagrams with cache hierarchy

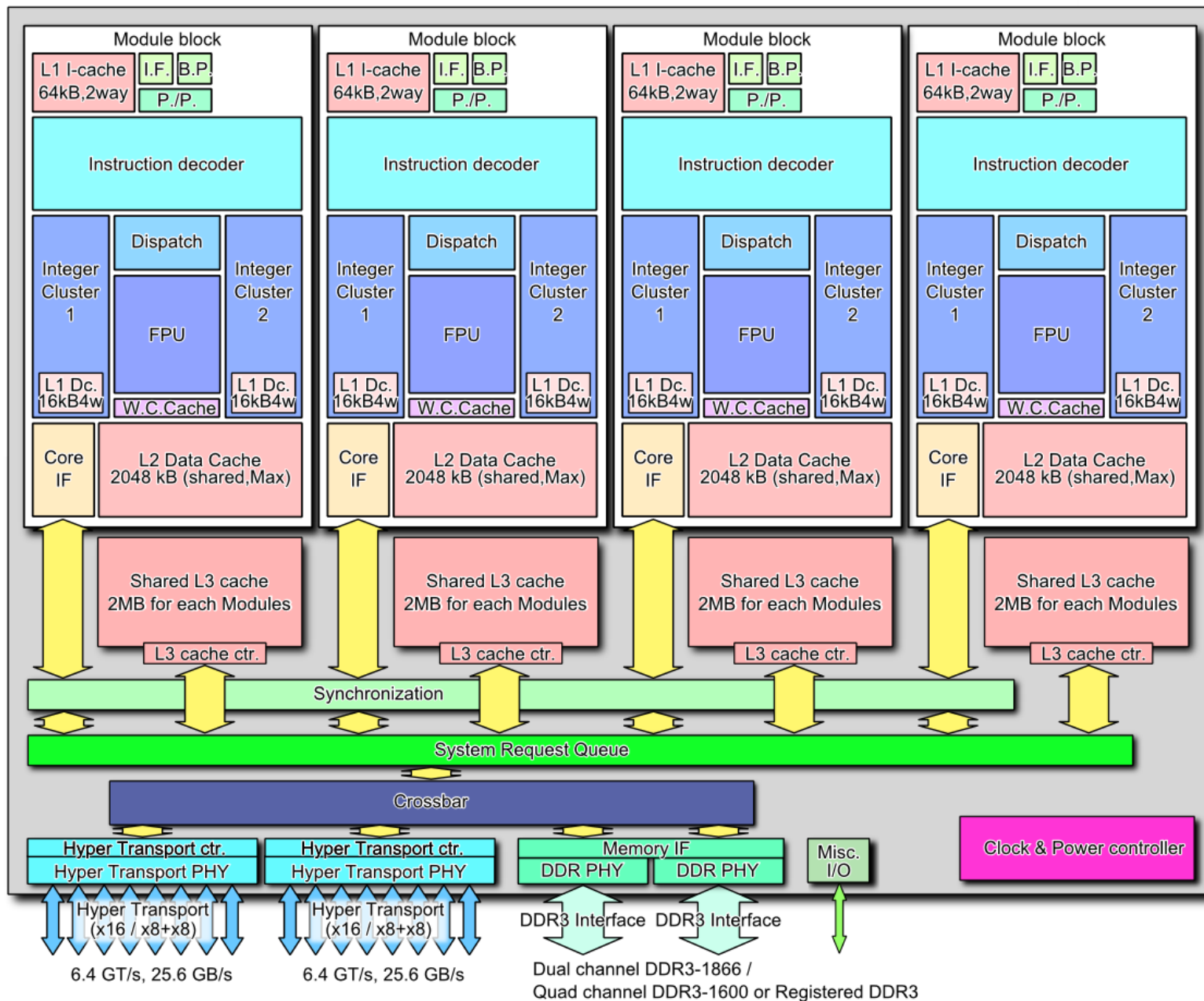
- Successors:
 - 2012 Piledriver - Family 15h (2nd-gen)
 - 2014 Steamroller - Family 15h (3rd-gen)
 - 2015 Excavator - Family 15h (4th-gen)
 - 2017 AMD Zen (Ryzen, Ryzen Threadripper, Epyc, Athlon)
 - 2018 Zen+ (Ryzen, Ryzen Threadripper)
 - 2019 Zen 2
 - 2020 Zen 3
 - 2022 ? Zen 4
- Nasledniki:

AMD Bulldozer-

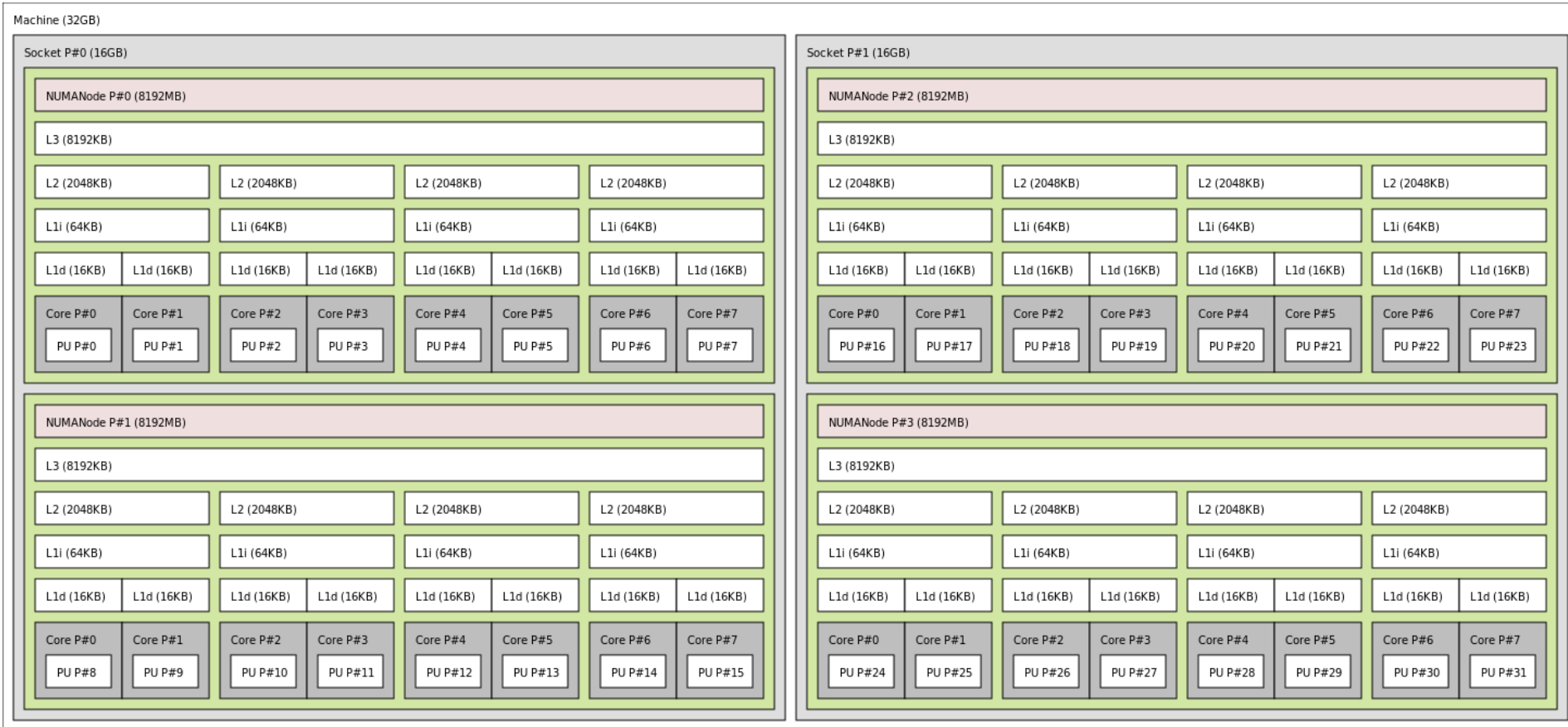
Block diagram
of one module
showing
2 integer
clusters



AMD
Bulldozer-
Block
diagram
of a 4
module
design
with
8 integer
clusters



AMD Bulldozer- Memory topology / Topologija spomina



ARM cache features / Lastnosti predpomnilnika

Core	Cache Type	Cache Size (kB)	Cache Line Size (words)	Associativity	Location	Write Buffer Size (words)
ARM720T	Unified	8	4	4-way	Logical	8
ARM920T	Split	16/16 D/I	8	64-way	Logical	16
ARM926EJ-S	Split	4-128/4-128 D/I	8	4-way	Logical	16
ARM1022E	Split	16/16 D/I	8	64-way	Logical	16
ARM1026EJ-S	Split	4-128/4-128 D/I	8	4-way	Logical	8
Intel StrongARM	Split	16/16 D/I	4	32-way	Logical	32
Intel Xscale	Split	32/32 D/I	8	32-way	Logical	32
ARM1136-JF-S	Split	4-64/4-64 D/I	8	4-way	Physical	32

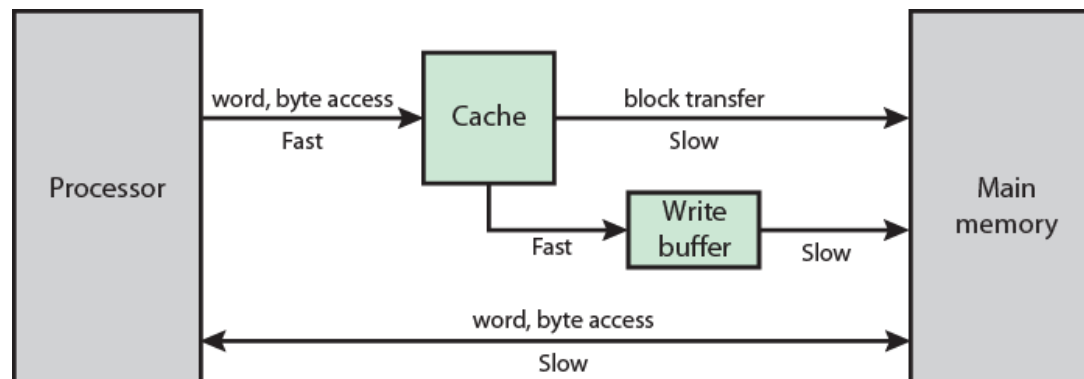
ARM cache organisation / Organizacija predpomnilnika

Small FIFO write buffer

- Enhances memory write performance
- Between cache and main memory
- Small c.f. cache
- Data put in write buffer at processor clock speed
- Processor continues execution
- External write in parallel until empty
- If buffer full, processor stalls
- Data in write buffer not available until written
 - So keep buffer small

Majhen medpomnilnik za pisanje FIFO

- Pohitri zapisovanja v pomnilnik
- Med predpomnilnikom in glavnim
- Majhna c.f. predpomnilnik
- Podatki se vnesejo v medpomnilnik za zapisovanje z hitrostjo procesorja
- CPE nadaljuje z izvajanjem
- Zunanje zapisovanje vzporedno, ni prazen
- Če je predpomnilnik poln, se CPE ustavi
- Podatki v medpomnilniku za pisanje niso na voljo, dokler niso napisani
 - Zato naj bo medpomnilnik majhen



Questions? / Vprašanja?

Key terms / Ključni pojmi

access time, associative mapping, cache hit, cache line, cache memory, cache miss, cache set, data cache, direct access, direct mapping, high-performance computing (HPC), hit, hit ratio, instruction cache, L1 cache, L2 cache, L3 cache, line, locality, logical cache, memory hierarchy, miss, multilevel cache, physical address, physical cache, random access, replacement algorithm, secondary memory, sequential access, set-associative mapping, spatial locality, split cache, tag, temporal locality, unified cache, virtual address, virtual cache, write back, write through

Questions / Vprašanja

- **4.1** What are the differences among sequential access, direct access, and random access?
- **4.2** What is the general relationship among access time, memory cost, and capacity?
- **4.3** How does the principle of locality relate to the use of multiple memory levels?
- **4.4** What are the differences among direct mapping, associative mapping, and set-associative mapping?
- **4.5** For a direct-mapped cache, a main memory address is viewed as consisting of three fields. List and define the three fields.
- **4.6** For an associative cache, a main memory address is viewed as consisting of two fields. List and define the two fields.
- **4.7** For a set-associative cache, a main memory address is viewed as consisting of three fields. List and define the three fields.
- **4.8** What is the distinction between spatial locality and temporal locality?
- **4.9** In general, what are the strategies for exploiting spatial locality and temporal locality?