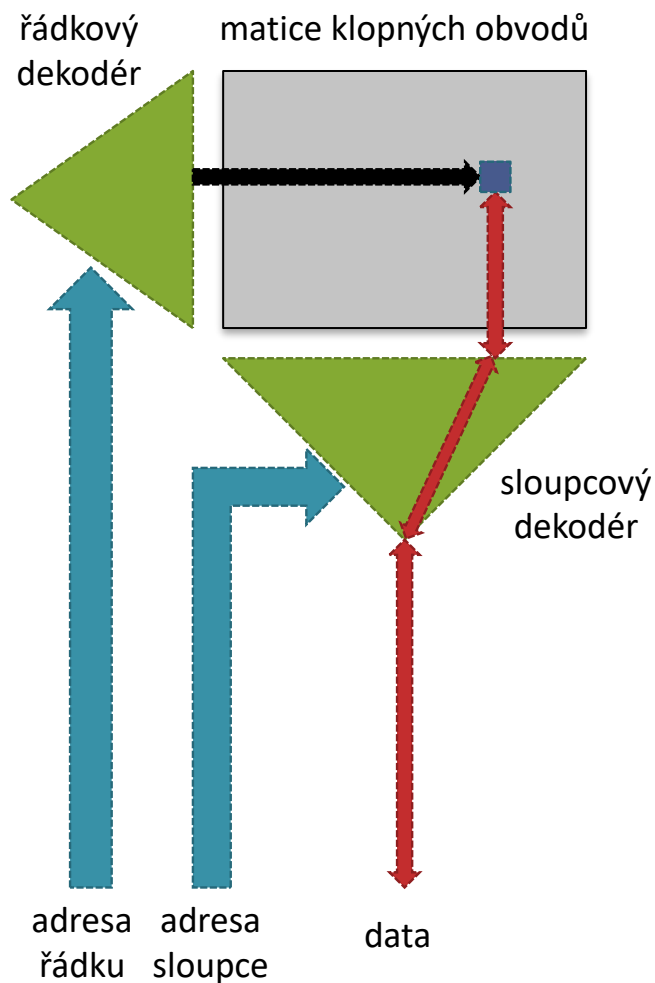


# Statická RAM

# Architektura paměti SRAM



## • Řádky a sloupce

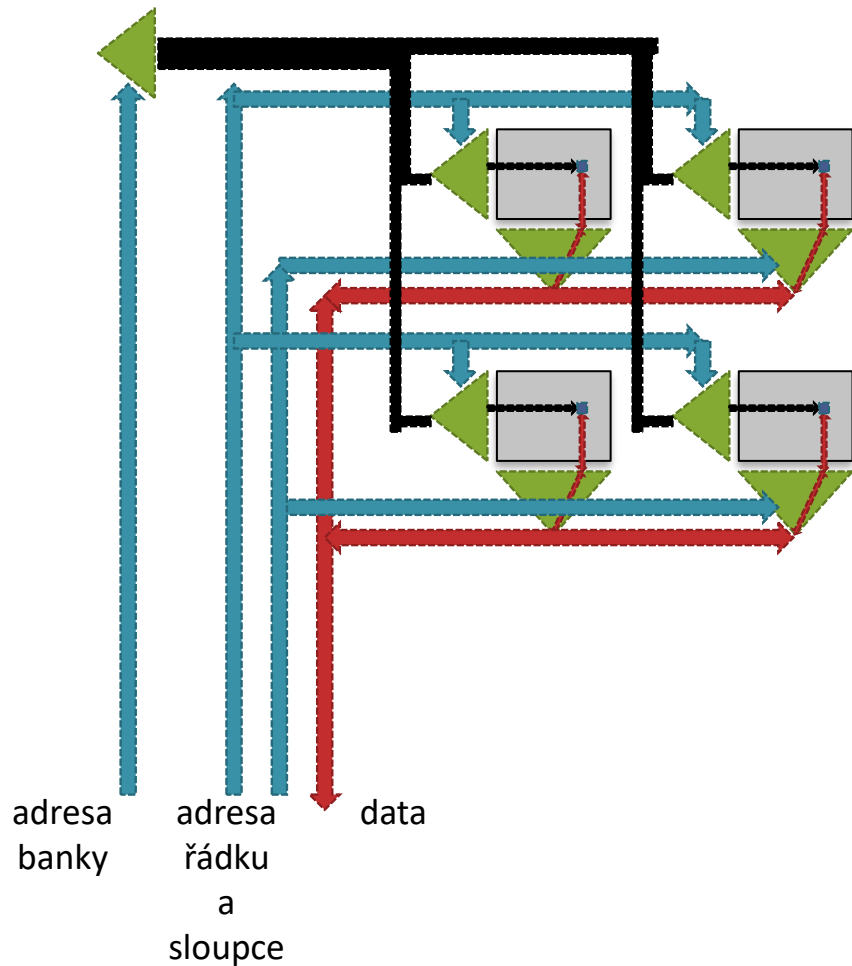
- Aktivace probíhá současně

## • Čtení

- Celý řádek vysílá své hodnoty
- Sloupcový dekodér vybírá data

## • Zápis

- Sloupcový dekodér posílá signály „zapiš 0“ a „zapiš 1“ do vybraných sloupců

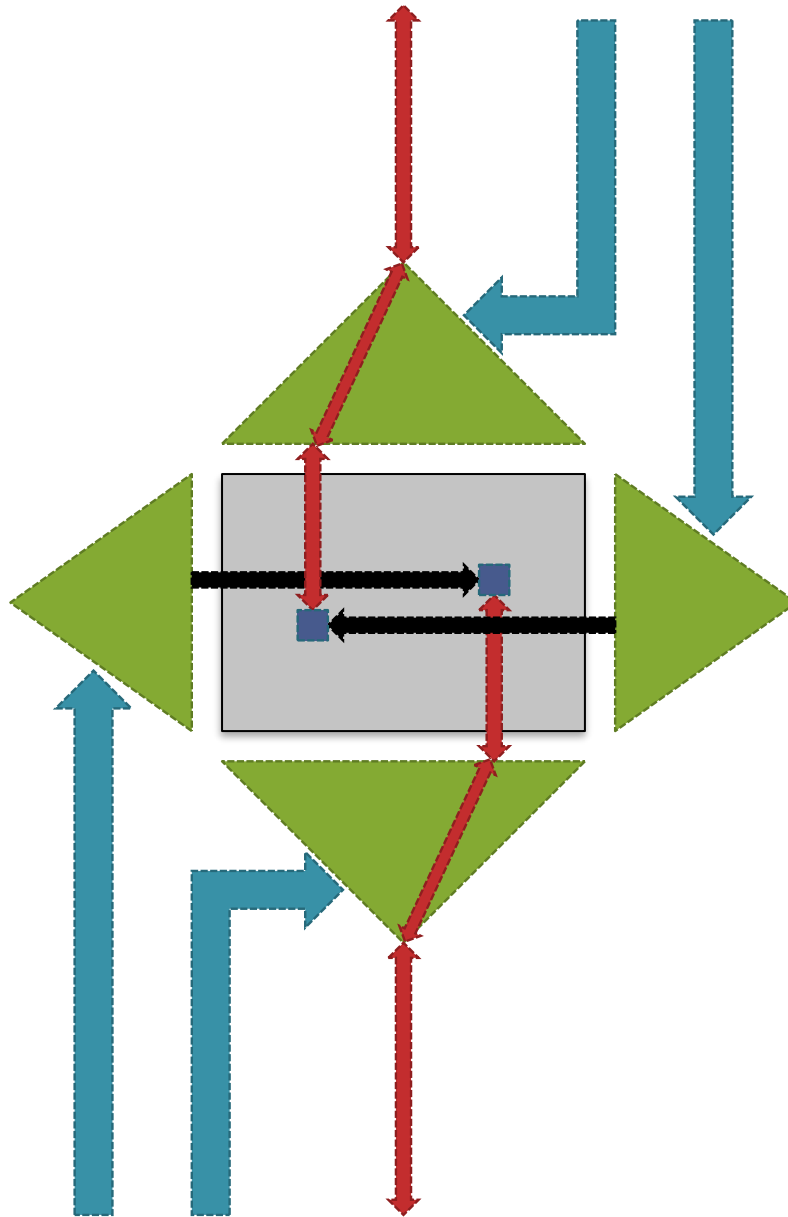


## •Banky

- Velké paměti jsou děleny na banky

## •Paralelismus

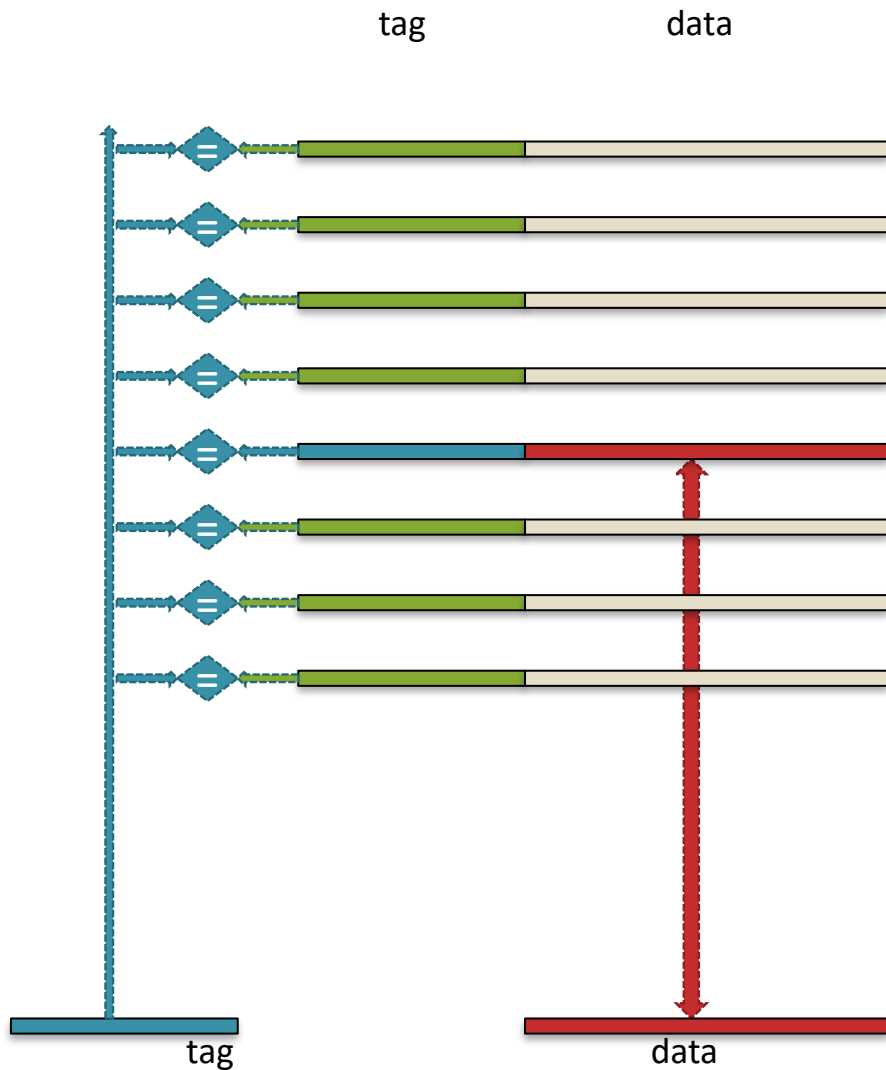
- Operace v různých bankách se mohou překrývat
- V jednom cyklu lze zahájit pouze jednu operaci
  - Společné sběrnice a dekodér banky brání plnému paralelismu



## • Dvoubránová paměť

- V každé buňce je zdvojeno její vnější rozhraní
  - cca 33% plochy navíc
- Přístup ze dvou bran je zcela nezávislý
  - Současný zápis různých hodnot do téže buňky je nežádoucí
- Používá se pro malé výkonově kritické paměti
  - registry, TLB

# Asociativní paměť

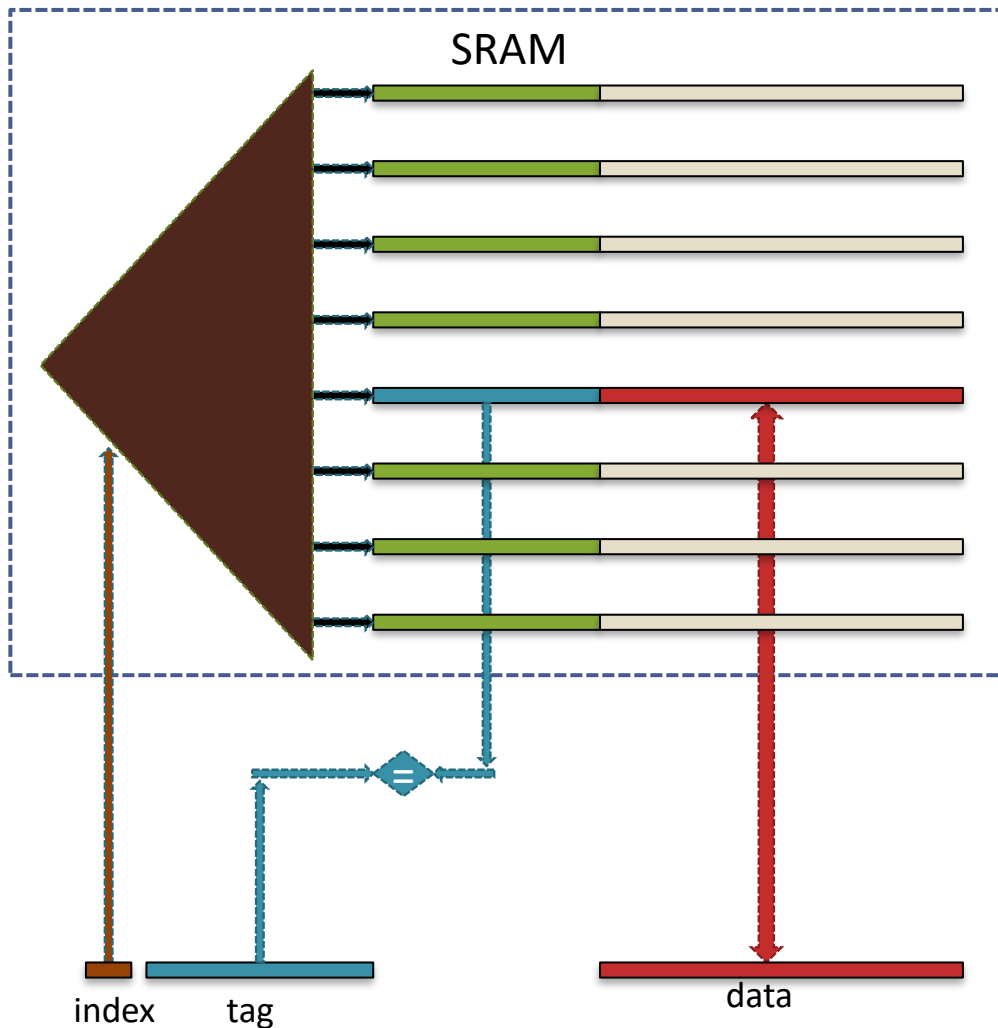


## • Plně asociativní paměť

- Každý řádek má svůj komparátor
- Vyžaduje strategii výměny při zaplnění

## • Drahá

- nejvýše desítky řádků
  - TLB1



## •Přímo mapovaná asociativní paměť

- Základem běžná paměť (SRAM)
- Společný komparátor
- Nevyžaduje strategii výměny při zaplnění
- Umožňuje číst data ještě před dokončením porovnání

## •Prakticky nepoužitelná

- Kolize jsou v reálných aplikacích příliš pravděpodobné





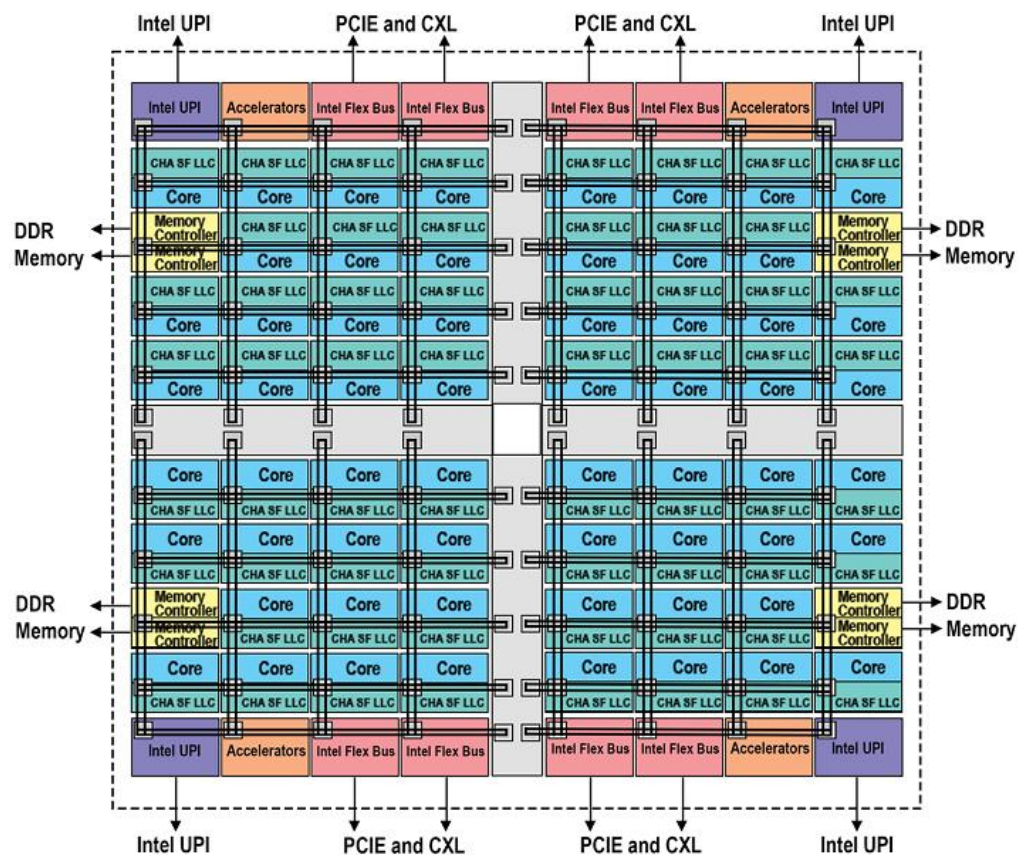
## ▶ LLC = L2 nebo L3

- ▶ Obvykle společná pro všechna/některá jádra na čipu
- ▶ Typické velikosti 1-2 MB per core
  - V posledních letech neroste
- ▶ Cache Line = 512, výjimečně 1024 bitů
  - Obvykle odpovídá velikosti atomické operace DDR3/4/5 DRAM
- ▶ Latence 20..30 ns
  - U serverových CPU větší; v posledních letech roste!
- ▶ Komunikace LLC – DRAM – vždy celá Cache Line
- ▶ Komunikace s nižšími úrovněmi – obvykle 128 nebo 256 bitů najednou

## ▶ Paralelismus

- ▶ Celý proces přístupu do cache je pipeline
  - lze zahájit 1 přístup každé 1-3 cykly procesoru
- ▶ Cache je dělena na banky
  - přístup do různých bank může být paralelní

# Cache poslední úrovně



- ▶ LLC blocks are paired with individual cores
  - ▶ The more cores, the more LLC cache
  - ▶ LLC blocks form a combined associative memory
    - Uniform config: Any LLC search logically operates over all blocks
    - The chip may be reconfigured to 2 or 4 independent LLC blocks (NUMA)

## ▶ L2 v systémech s L3

- ▶ Každé jádro má vlastní
- ▶ Typické velikosti 1 - 2 MB
- ▶ Cache Line téměř vždy 512 bitů
- ▶ Latence ~5 ns
- ▶ Komunikace s okolními úrovněmi – obvykle 128 nebo 256 bitů najednou

## ▶ Velikost L2 a L3 je dnes podobná - proč jsou odděleny?

- ▶ L2 je privátní
  - Řádky používané více jádry jsou v L2 replikovány
    - Zabírají více prostoru
    - Zápisy způsobují přesuny dat mezi jádry
- ▶ L3 je společná pro všechna jádra
  - Řádky používané více jádry jsou v L3 jen jednou
    - Efektivnější využití prostoru - celkově se do L3 vejde více dat než do L2
    - Komplikovanější vyhledávání - větší latence

## ▶ L1D

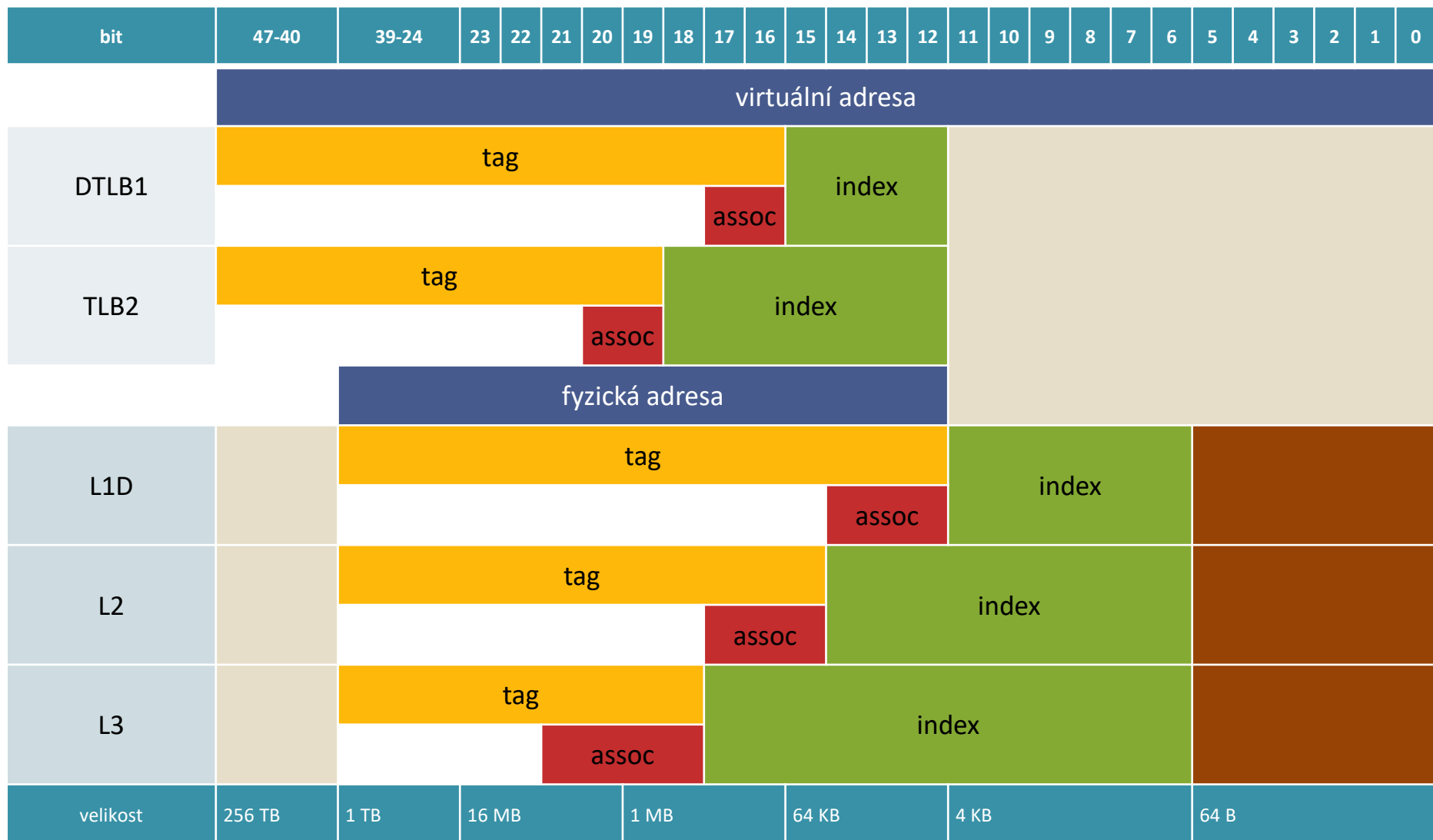
- ▶ Pouze pro data (instrukce mají vlastní L1I cache)
- ▶ Každé jádro má vlastní
  - Systém musí řešit přístup k datům v L1/L2 cache sousedního jádra
- ▶ Typické velikosti 16-64 KB
- ▶ Cache Line téměř vždy 512 bitů
- ▶ Latence 1-2 ns
- ▶ Komunikace s vyšší úrovní – obvykle 128 nebo 256 bitů najednou
- ▶ Komunikace s jádrem – podle šířky operandu instrukce (8 až 512 bitů)

## ▶ Paralelismus

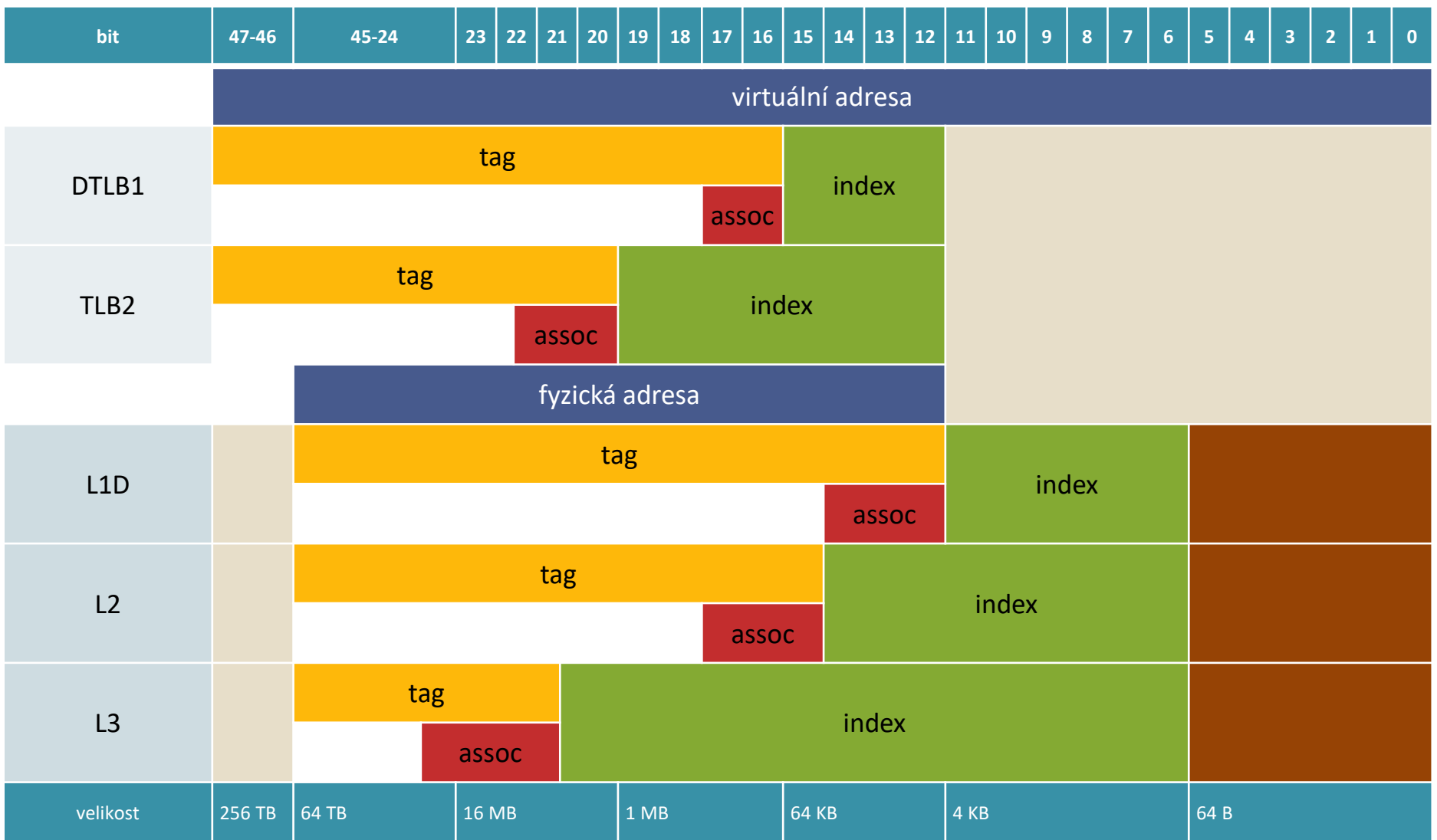
- ▶ Celý proces přístupu do cache je pipeline
  - v každém cyklu může začít nový přístup
- ▶ Cache může mít více bran
  - v každém cyklu mohou začít dva přístupy, pokud vedou na jiné adresy
  - komunikace s vyšší úrovní běží na pozadí (eject, prefetch)

- ▶ Překlad virtuálních adres na fyzické
  - ▶ Často dvě úrovně TLB
    - DTLB1 pouze pro data, cca. 64 záznamů (pro 4 KB stránky)
      - odpovídá 64-256 KB adresového prostoru
    - TLB2 společná, cca. 1024 záznamů
  - ▶ Latence DTLB1 bývá započtena do latence L1D
    - L1D bývá „virtually-indexed-physically-tagged“
  - ▶ Latence TLB2 v řádu 2-3 ns
  - ▶ Není-li záznam v TLB:
    - „page walk“ - procesor realizuje 2-4 přístupy do paměti (cache)
    - „page fault“ – řeší operační systém
- ▶ Paralelismus
  - ▶ TLB bývá vícebránová – obsluhuje čtení i zápisy paralelně
  - ▶ Virtually-indexed cache
    - L1 cache indexována virtuální adresou (nebo ofsetem uvnitř stránky)
    - Překlad v TLB může běžet paralelně s adresací L1 cache
    - Vyžaduje speciální opatření pro případ násobného mapování téže fyzické adresy

# Typické použití bitů adresy (Intel Sandy Bridge)



# Typické použití bitů adresy (Intel Xeon E7 v4 - 2017)



# Typická latence a propustnost – Intel Sandy Bridge (2011)

	latence (cykly)	operací/ cyklus	B/operace	B/cyklus	GB/s [3 GHz CPU]
registry	0 (započteno v čase instrukce)	teoreticky 14, prakticky 3-6	4-16	až 112 SIMD až 160	teoreticky 480
L1D	4	3 (2*R+1*W)	4-16	12-48	36-144
L2	12	1	32	32	96
L3	26-31	1	32	32	96
DDR3-1600 dual channel	cca 120	2/15	64	8.5	25.6

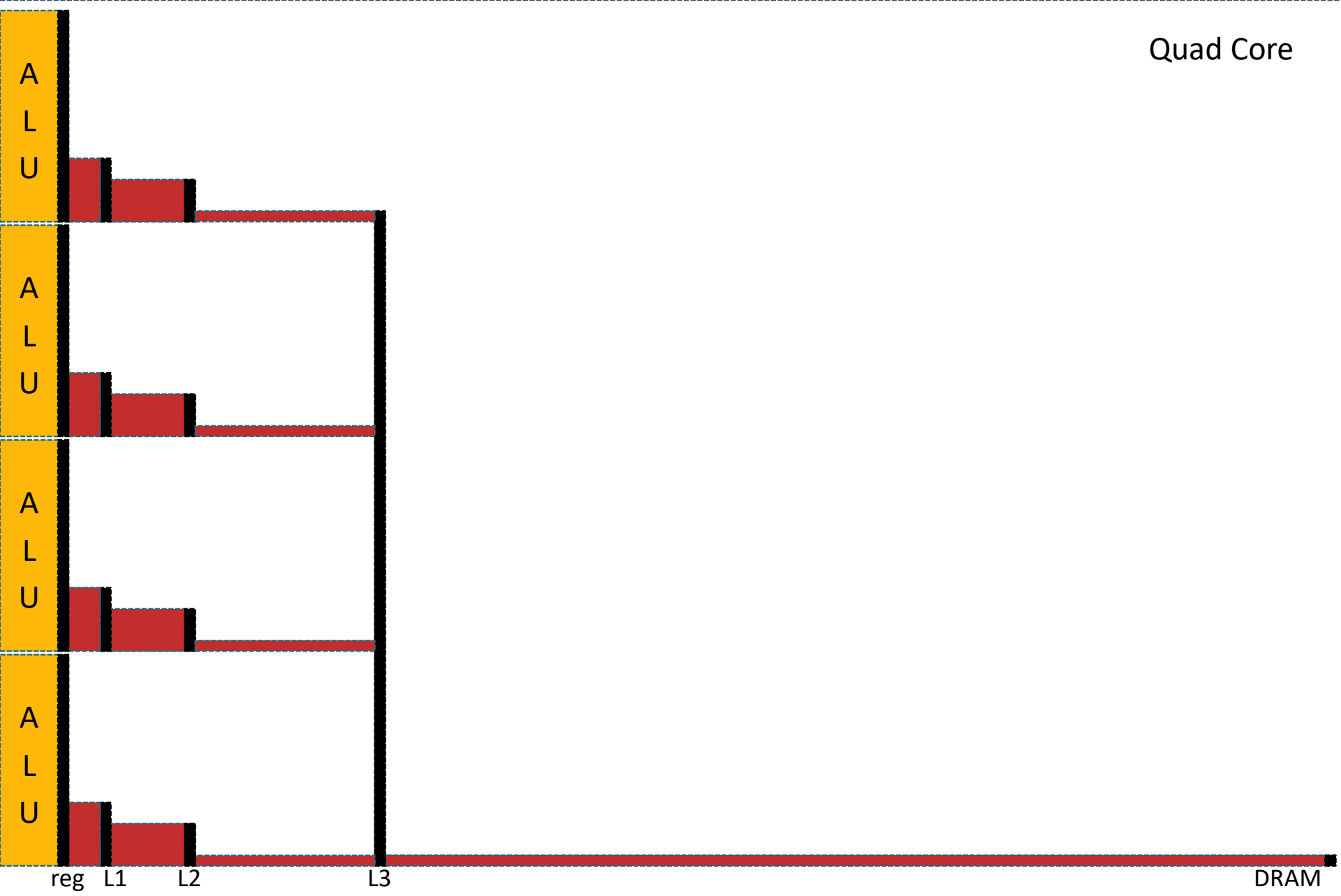
A  
L  
U





# Typická latence a propustnost – Intel Sandy Bridge

Quad Core



# Typická latence a propustnost – Intel Broadwell (Xeon E5 v4 - 2014)

	latence (cykly)	operací/ cyklus	B/operace	B/cyklus	GB/s [3 GHz CPU]
registry	0 (započteno v čase instrukce)	teoreticky $15 * R + 4 * W$	4-32	až 152 SIMD až 288	teoreticky 864
L1D	4	$3 (2 * R + 1 * W)$	4-32	12-96	36-288
L2	12	1	64	64	192
L3	~22	1	32	32	96
DDR4-2400 dual channel	cca 120	1/4	64	16	60

# Typická latence a propustnost – Intel Sapphire Rapids (2023)

cache (per core)	latence (cykly)	operací/cyklus	B/operace	B/cyklus	GB/s [3 GHz CPU]
registry	0 (započteno v čase instrukce)	?	4-64	?	?
L1D (48 KB)	4	3 (2*R+2*W)	4-64	<224	<700
L2 (2MB)	16	1	64	64	192
L3 (1.875MB)	~100	?	64	?	?
DDR5-4800 octo channel	cca 180	?	64	?	200

# Typická latence a propustnost – Intel Xeon Gold 6130 (2019) [parlab]

cache (per core)	latence (cykly)	operací/cyklus	B/operace	B/cyklus	GB/s [3 GHz CPU]
registry	0 (započteno v čase instrukce)	2 AVX512 4 scalar	$(1..3)*(4..64)$	< 384	? < 1000 per core
L1D (32 KB)	4	3 (2*R+1*W)	4..64	192	140 per core
L2 (1MB)	14	1	64	64	110 per core
L3 (1.375MB)	~75	?	64	?	?
DDR4 6 channel	cca 220	?	64	?	~80 total ~5 per core

- ▶ Velikosti, latence a architektury cache jsou různé
  - ▶ Spolehlivě lze odhadnout pouze dolní mez L1 (16KB)
  - ▶ Ladit software na přesné velikosti všech úrovní je téměř nerealizovatelné
    - autotuning, NUMA awareness
    - cache-oblivious algoritmy
- ▶ Velikost Cache Line lze odhadnout spolehlivě – 64B
  - ▶ Přesunovat do procesoru méně než 64B je plýtvání
    - spojové seznamy či stromy s malými uzly jsou neefektivní
- ▶ Velikost TLB1 je velmi malá
  - ▶ Přístup na větší počet míst paměti je problematický pro TLB, ne pro L1
    - L1 udrží stovky živých míst, TLB pouze desítky
    - Bucket-sort nebo merge může být TLB-bound
      - více než 64 živých bodů vzdálených od sebe více než 4 KB
    - Bloky dynamicky alokovaných dat mohou být velmi vzdálené
      - 100-prvkový vyhledávací strom se nemusí vejít do DTLB1

Nešikovně naprogramovaný kód může dobrý překladač napravit

Nevhodnou datovou strukturu překladač nenapraví  
(a nepomůže ani naprogramování v assembleru)

- ▶ Registry poskytují cache úrovně 0
  - ▶ Použití této „cache“ řídí překladač
    - Programátor může vytvořit šanci nebo zlepšit podmínky úpravou kódu
  - ▶ Velikost řádky je velikost registru (podle řešené úlohy; SIMD: 128/256 bit)
  - ▶ Velikost “L0-cache” dána počtem registrů (Intel: 8/16, některé obsazené)
    - 32 až 512 B
    - Pro danou úlohu a platformu programátor dokáže určit
  - ▶ Lze programovat na míru (týká se jádra algoritmu)
- ▶ Velikost řádku Cache je stabilní - 64B
  - ▶ Lze programovat na míru (týká se zejména datových struktur)
- ▶ V intervalu 4KB-16MB leží mnoho hranic důležitých velikostí
  - ▶ Bez znalosti konkrétní varianty CPU nelze určit přesnou polohu hranic
  - ▶ Každé zdvojnásobení velikosti úlohy může dramaticky změnit poměry
  - ▶ Nejvhodnější je přístup převzatý z Cache-Oblivious algoritmů
    - Rekurzivní dělení úlohy na části