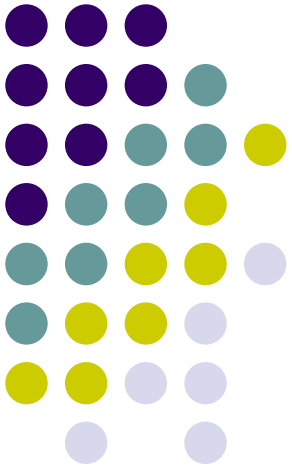
Cache paměti (2. část)

Hardware



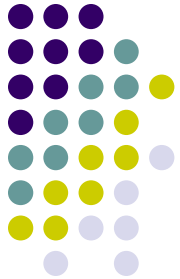
Plně asociativní cache

- Příklad obsahu primitivní cache s kapacitou 8B

Klíč	Data	platnost
12345h	A7h	1
2A4D1h	FFh	1
00000h	00h	0
FF2C5h	14h	1
145ADh	BCh	1
75683h	11h	1
A1122h	22h	1
71243h	5Ch	1

Ke každému uloženému bajtu (8 bitů dat) je třeba uložit také klíč (v tomto příkladu 20 bitů, ale reálně by to byly 32 bitů, protože na procesorech používáme 32 bitovou adresaci) a ke každý klíč musí mít svůj komparátor. Uložení dat je tedy **velmi drahé**. **Mnohem více místa v cache zabírají klíče a jejich komparátory než užitečná data!**

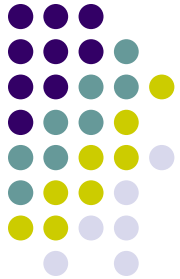
Plně asociativní cache



- Tento způsob cache paměti má své nevýhody:
  - V každém řádku tabulky se musí kromě dat (což je jen 8 bitů) uchovávat také klíč (který je mnohem větší než užitečná data – typicky 32 bitů)
  - Aby bylo možné data vyhledat, je nutné velké množství komparátorů – pro každý uložený bajt jeden komparátor
  - K uložení každého bajtu a zajištění možnosti jeho okamžitého vyhledání je v tomto typu cache potřeba extrémní množství tranzistorů
  - Daleko více místa v cache zabírají klíče a jejich komparátory než užitečná uložená data

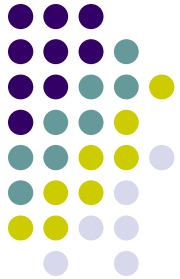
- Z těchto důvodů se plně asociativní paměti dnes již prakticky nepoužívají

Plně asociativní cache



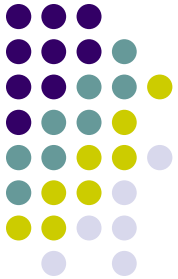
- Efektivita ukládání dat v plně asociativní cache se dá zvýšit, pokud se data budou ukládat ve větších blocích
- Blok = více bajtů uložených naráz
- Spolu s bajtem, který má být cachován, uložíme i několik jeho „sousedů“
- Je velmi pravděpodobné, že pokud se často používá bajt ležící na *adrese x*, budou se v budoucnu zřejmě používat i bajty ležící v jeho okolí (tedy na *adresách x-1, x+1, x+2*…. apod.)
- Bajty se tedy mohou do cache ukládat po dvou, po čtyřech, po osmi… na jeden řádek s jedním společným klíčem
- Pracujeme pak s blokem dvou, čtyř, osmi, šestnácti… bajtů naráz
- Vzhledem k tomu, že bajty tohoto bloku leží v operační paměti za sebou, stačí jako klíč uložit adresu prvního bajtu celého bloku
- Například pokud se budou do paměti cache ukládat data po blocích velkých 4B
  - Bajty 4Ah, FFh, 7Eh, 9Dh leží v operační paměti za sebou na adresách 12345678h, 12345679h, 1234567Ah, 1234567Bh
  - Do cache uložíme 4-bajtový blok s klíčem 12345678h (adresa prvního bajtu)
  - **Klíč:12345678h Data: 4A FF 7E 9D**
  - Na každém řádku je teď uloženo 4x8 bitů dat (32 b) a 32-bitový klíč – uložení dat je mnohem efektivnější

Plně asociativní cache



- Pokud se budou do paměti cache ukládat data po blocích velkých 8B
  - Bajty 4Ah, FFh, 7Eh, 9Dh, 05h, BFh, AAh, 98h leží v operační paměti za sebou na adresách 12345678h, 12345679h, 1234567Ah, 1234567Bh, 1234567Ch, 1234567Dh, 1234567Eh, 1234567Fh
  - Do cache uložíme **8-bajtový blok** s klíčem 12345678h (adresa prvního bajtu)
  - **Klíč:12345678h Data: 4A FF 7E 9D 05 BF AA 98**
  - Na každém řádku je teď uloženo 8x8 bitů dat (64 b) a 32-bitový klíč
  - Množství užitečných dat je již vyšší než počet bitů, které ukládáme jako klíč
  - K uložení **8 bajtů** teď stačí **jeden 32-bitový klíč** a jeden komparátor

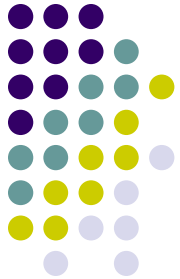
Plně asociativní paměť



- Jako klíč tedy ukládáme pouze adresu prvního bajtu celého bloku (adresa dalších bajtů v bloku se dá dopočítat, bajty ležely v paměti za sebou)
- Efektivita se dá dále zvýšit, pokud nebudeme jako klíč ukládat celou tuto adresu
- Pokud při ukládání 8-bajtového bloku budeme postupovat tak, že blok smí vždy začínat pouze adresou, které je dělitelná číslem 8, bude mít všech 8 adres shodných prvních 29 bitů
- (adresa dělitelná číslem 8 má v binárním zápisu na konci tři nuly)
- `12345678h = 00010010000110100010101100111100`*`0`*
- `12345679h = 00010010000110100010101100111100`*`1`*
- `1234567Ah = 00010010000110100010101100111101`*`0`*
- `1234567Bh = 00010010000110100010101100111101`*`1`*
- `1234567Ch = 00010010000110100010101100111110`*`0`*
- `1234567Dh = 00010010000110100010101100111110`*`1`*
- `1234567Eh = 00010010000110100010101100111111`*`0`*
- `1234567Fh = 00010010000110100010101100111111`*`1`*

- Jako klíč celého bloku teď bude stačit uvést 29 bitů!
- **Klíč:00010010000110100010101100111 1 Data: 4A FF 7E 9D 05 BF AA 98**

Plně asociativní cache



- Pokud cache používá blok velký **2 Bajty**
  - První bajt bloku musí ležet na sudé adrese (dělitelné dvěma)
  - Klíč nebude obsahovat poslední bit adresy
- Pokud cache používá blok velký **4 Bajty**
  - První bajt bloku musí ležet na adrese dělitelné 4
  - Klíč nebude obsahovat poslední 2 bity adresy
- Pokud cache používá blok velký **8 Bajtů**
  - První bajt bloku musí ležet na adrese dělitelné 8
  - Klíč nebude obsahovat poslední 3 bity adresy
- Pokud cache používá blok velký **16 Bajtů**
  - První bajt bloku musí ležet na adrese dělitelné 16
  - Klíč nebude obsahovat poslední 4 bity adresy
- Čím větší je blok, tím kratší bude klíč

Plně asociativní cache

- Příklad obsahu primitivní cache s kapacitou 32B, pracující po blocích 4B se sedmibitovým klíčem

Klíč	Data	platnost
1010100b	A7 12 C5 14	1
0010100b	11 2E 7E FF	1
0000000b	00 00 00 00	0
1110111b	14 00 14 00	1
1010101b	B2 01 12 00	1
1010111b	11 2A 11 32	1
0011000b	22 2B 2E F1	1
1000000b	5C 51 11 10	1

Tento záznam v paměti cache znamená, že je v ní z hlavní operační paměti zkopírován blok 4 bajtů z adres, jejichž horních 7 bitů je 1010101b

Tedy z adres **1010101***00*b až **1010101***11*b

Na těchto adresách leží bajty B2, 01,12, 00

Plně asociativní cache

- Příklad:
- Na jaké adrese v hlavní operační paměti leží tento bajt?

Klíč	Data	platnost
1010100b	A7 12 C5 14	1
0010100b	11 2E 7E FF	1
0000000b	00 00 00 00	0
1110111b	14 00 14 00	1
1010101b	B2 01 12 00	1
1010111b	11 2A 11 32	1
0011000b	22 2B 2E F1	1
1000000b	5C 51 11 10	1

U záznamu je uveden klíč **0010100**

První bajt tohoto bloku leží na adrese
Adresa **001010000** *(přidaly se 2 nuly)*

Bajt 2E leží za prvním bajtem
Adresa **001010001**

Na další adrese leží bajt 7E
Adresa **001010010**

A poslední bajt FF leží na
Adresa **001010011**

Plně asociativní cache

- Příklad:
- Na jaké adrese v hlavní operační paměti leží tento bajt?

Klíč	Data	platnost
1010100b	A7 12 C5 14	1
0010100b	11 2E 7E FF	1
0000000b	00 00 00 00	0
1110111b	14 00 14 00	1
1010101b	B2 01 12 00	1
1010111b	11 2A 11 32	1
0011000b	22 2B 2E F1	1
1000000b	5C 51 11 10	1

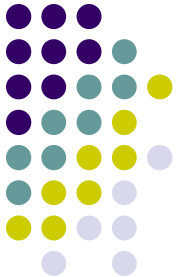
U záznamu je uveden klíč **1010111**

První bajt tohoto bloku leží na adrese
Adresa **1010111***00 (přidaly se 2 nuly)*

Jednotlivé bajty tohoto bloku tedy leží na adresách:

1010111*00*	11h
1010111*01*	2Ah
1010111*10*	11h
1010111*11*	32h

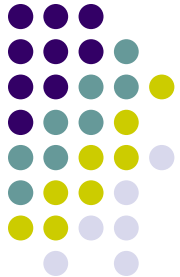
Plně asociativní cache



- Příklad:
- Mikroprocesor používá 32-bitovou adresaci
- Do paměti cache se ukládají bloky **8B**
- Je třeba uložit bajt z adresy AB462DF6h
- Jaké další adresy budou uloženy do cache v jednom bloku spolu s bajtem z adresy AB462DF6h?
- Jaký bude u tohoto záznamu uveden klíč?

- Adresa AB462DF6h není dělitelná číslem 8
- Blok musí začínat adresou, která je dělitelná číslem 8
- Blok bude tedy začínat nějakou jinou nižší adresou a náš bajt z adresy AB462DF6h nebude prvním bajtem tohoto bloku
- Adresa AB462DF6h vypadá binárně takto: 10101011010001100010110111110110
- Nejbližší nižší adresa dělitelná číslem 8 je: 10101011010001100010110111110***000***
- Blok tedy musí začínat adresou **AB462DF0h**
- Klíč bude mít délku **29 bitů** – 10101011010001100010110111110
- V jednom bloku budou uloženy bajty z adres AB462DF0h až AB462DF7h

Plně asociativní cache



- Příklad:
- Mikroprocesor používá 32-bitovou adresaci
- Do paměti cache se ukládají bloky **4B**
- Je třeba uložit bajt z adresy AB462DF6h
- Jaké další adresy budou uloženy do cache v jednom bloku spolu s bajtem z adresy AB462DF6h?
- Jaký bude u tohoto záznamu uveden klíč?

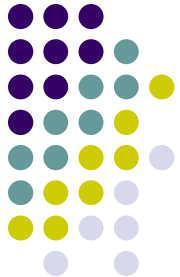
- Adresa AB462DF6h není dělitelná číslem 4
- Blok musí začínat adresou, která je dělitelná číslem 4
- Blok bude tedy začínat nějakou jinou nižší adresou a náš bajt z adresy AB462DF6h nebude prvním bajtem tohoto bloku
- Adresa AB462DF6h vypadá binárně takto: 10101011010001100010110111110110
- Nejbližší nižší adresa dělitelná číslem 4 je: 10101011010001100010110111110100
- Blok tedy musí začínat adresou **AB462DF4h**
- Klíč bude mít délku **30 bitů** – 101010110100011000101101111101
- V jednom bloku budou uloženy bajty z adres AB462DF4h až AB462DF7h

Plně asociativní cache

- Příklad:
- Mikroprocesor používá 32-bitovou adresaci
- Do paměti cache se ukládají bloky **16B**
- Je třeba uložit bajt z adresy AB462DF6h
- Jaké další adresy budou uloženy do cache v jednom bloku spolu s bajtem z adresy AB462DF6h?
- Jaký bude u tohoto záznamu uveden klíč?

- Adresa AB462DF6h není dělitelná číslem 16
- Blok musí začínat adresou, která je dělitelná číslem 16
- Blok bude tedy začínat nějakou jinou nižší adresou a náš bajt z adresy AB462DF6h nebude prvním bajtem tohoto bloku
- Adresa AB462DF6h vypadá binárně takto: 10101011010001100010110111110110
- Nejbližší nižší adresa dělitelná číslem 16 je: 10101011010001100010110111110000
- Blok tedy musí začínat adresou **AB462DF0h**
- Klíč bude mít délku **28 bitů** – 1010101101000110001011011111
- V jednom bloku budou uloženy bajty z adres AB462DF0h až AB462DFFh

Plně asociativní cache



- Příklad:
- Mikroprocesor používá 32-bitovou adresaci
- Do paměti cache se ukládají bloky **32B**
- Je třeba uložit bajt z adresy AB462DF6h
- Jaké další adresy budou uloženy do cache v jednom bloku spolu s bajtem z adresy AB462DF6h?
- Jaký bude u tohoto záznamu uveden klíč?

- Adresa AB462DF6h není dělitelná číslem 32
- Blok musí začínat adresou, která je dělitelná číslem 32
- Blok bude tedy začínat nějakou jinou nižší adresou a náš bajt z adresy AB462DF6h nebude prvním bajtem tohoto bloku
- Adresa AB462DF6h vypadá binárně takto: 10101011010001100010110111110110
- Nejbližší nižší adresa dělitelná číslem 32 je: 10101011010001100010110111***00000***
- Blok tedy musí začínat adresou **AB462DE0h**
- Klíč bude mít délku jen **27 bitů** – 101010110100011000101101111
- V jednom bloku budou uloženy bajty z adres AB462DE0h až AB462DFFh