Charakteristika dalších verzí procesorů v PC

Cíl přednášky

- Poukázat na principy tvorby architektur nových verzí personálních počítačů.
- Prezentovat aktuální pojmy.

Úvod

- Zvyšování výkonu cestou paralelizace výpočtu.
- Paralelní systémy lze třídit z hlediska počtu toků instrukcí a počtu toků dat:

  SI – systém s jedním tokem instrukcí (Single Instruction stream)

  MI – systém s několika toky instrukcí (Multiple Instruction stream)

  SD – systém s jedním tokem dat (Single Data stream)

  MD – systém s několikanásobným tokem dat (Multiple Data stream)
- Použití těchto dvou hledisek vede ke vzniku **čtyř základních typů počítačových systémů označovaných zkratkami SISD, SIMD, MISD a MIMD**.

Kombinace základních architektur

- **SISD** (Single Instruction stream, Single Data stream) – klasický jednoprocesorový počítač von Neumannova typu zpracovávající data sériově.
- **SIMD** (Single Instruction stream, Multiple Data stream) – pole procesorů zpracovávající paralelně pole hodnot podle společného programu.
- **MIMD** (Multiple Instruction stream, Multiple Data stream) – multiprocesorový systém, v němž každý procesor je řízen samostatným programem a pracuje s jinými daty než ostatní procesory.
- **MISD** – soustava procesorů pracujících podle různých programů na společných datech (označováno jako enfant terrible této klasifikace) – v praxi neobsazená alternativa.

Uplatnění SIMD

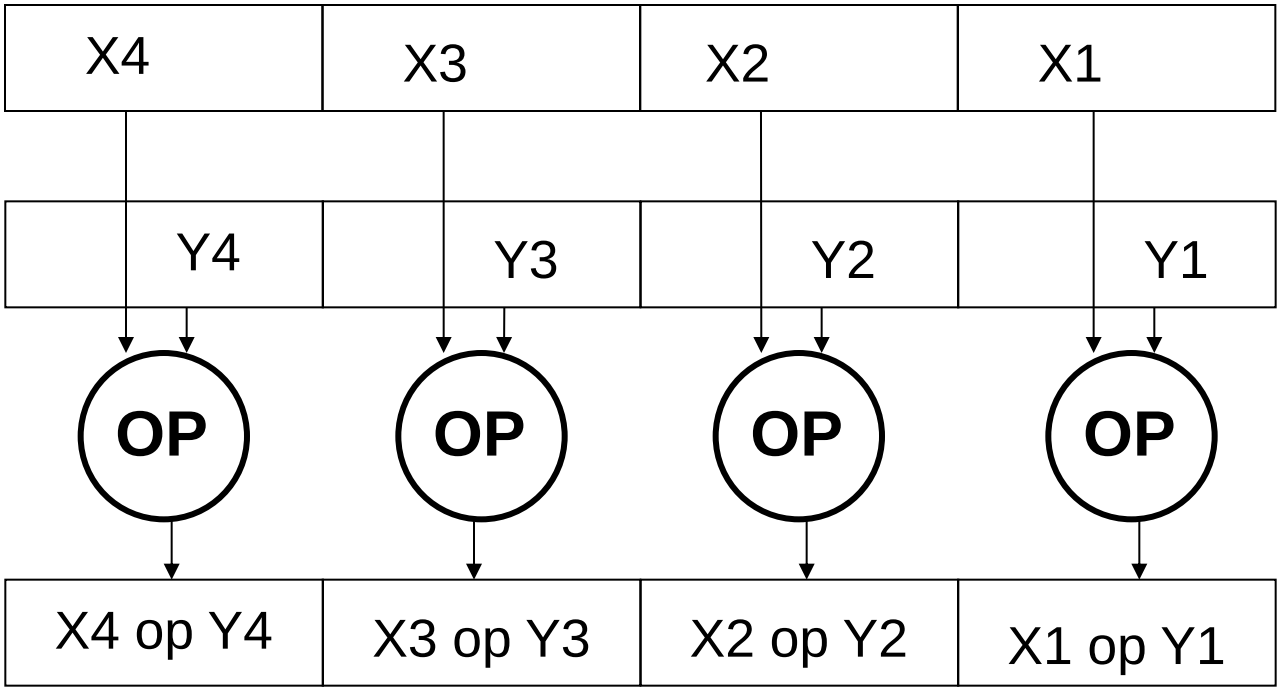
- Typický rys multimediálních aplikací – paralelně prováděné operace na stejných typech dat.
- Strategie SIMD – jeden tok instrukcí na více procesorech, instrukce operuje na násobných ale shodných typech dat.
- Příklad: technologie MMX, která operuje paralelně s násobnými daty ve zkomprimované formě uloženými v 8/16 bitovém registru.
- Jiný případ: stejná hodnota se přičítá k různým datům, častá situace v multimediálních aplikacích.
- Změna jasu celého obrazu: každý pixel sestává z informace o třech složkách jasu R, G, B. Pokud potřebujeme změnit jas celého obrazu, přečteme R, G, B pixelu z paměti, přičteme (odečteme) patřičnou hodnotu a výsledek zapíšeme zpět do paměti.

Vývoj technologie SIMD na úrovni instrukcí

- SIMD – Single Instruction, Multiple Data, jeden typ instrukce se provádí na více datech.

  Typická operace SIMD – obr. 1.

  Dvě skupiny operandů, každá obsahuje 4 zkomprimované datové prvky (X1, X2, X3, X4 a Y1, Y2, Y3, Y4).



Obr. 1 – operace v architektuře SIMD

Co dále nastalo v architekturách PC

- Takto to bylo u technologie MMX.
- Technologie MMX umožňovala realizovat operace SIMD na zkomprimovaných slabikách, slovech, dvouslovech (vše celá čísla) uložených v osmi 64 bitových registrech.
- Registry MMX – 64 bitů a XMM – 128 bitů.
- Specielní instrukce byly šity na míru pro multimediální a komunikační aplikace.

- **Pentium III** – model SIMD byl změněn na Streaming SIMD Extensions (SSE).
- Rozdíl oproti klasickému SIMD:
  - Operace na operandech obsahujících čtyři komprimovaná čísla v pohyblivé řádové čárce.
  - Operandy uloženy buď v paměti nebo v osmi 128 bitových registrech (registry XMM).
  - Specielní množina instrukcí vytvořená pro procesory MMX byla rozšířena o dalších 64 instrukcí.

Pentium 4

Pentium 4 – byl použit model **Streaming SIMD Extensions 2 (SSE2)** s těmito vlastnostmi:

- Operace se provádějí na těchto operandech:

  dvě čísla v pohyblivé řádové čárce, dvojnásobná přesnost,

16 zkomprimovaných slabik,

8 zkomprimovaných slov,

4 zkomprimovaná dvouslova,

2 zkomprimovaná čtyřslova

(čtyři poslední alternativy – čísla typu integer).

- Operandy mohou být v paměti nebo registrech.
- Podpora aritmetiky SIMD pro práci se 64 bitovými operandy typu integer.
- Instrukce pro konverzi mezi datovými typy (původními a novými).

• **Trend: SIMD na úrovni instrukcí, rozšiřování množiny datových typů (změna v množině instrukcí).**

Dílčí závěry

- Možnost, jak vytvářet nové architektury – stimulace nasazováním do nových aplikací (v architekturách CISC je to snadnější proces než v architekturách RISC).
- Architektury takových procesorů se nijak výrazně nelišily – extenzivní rozvoj.
- MMX – představitel uplatnění takových postupů.
- Jiná alternativa – zavádění nových prvků a nových myšlenek do architektur počítačů.

Vláknové architektury

- Míra posuzování výkonnosti procesoru – rychlost provádění instrukcí:

  míra MIPS = f x IPC

  MIPS – Millions Instructions per Second

  jednotka výkonnosti počítače, která udává počet zpracovaných instrukcí za sekundu

  f - kmitočet procesoru

  IPC – Instructions per Cycle
- Jak to řešit?

Jak řešit zvyšování výkonu procesoru?

- 1. možnost:

  zvyšovat kmitočet synchronizace procesoru
- 2. možnost:

  zvyšovat počet instrukcí, které jsou v jednom strojovém cyklu dokončeny.

Jak řešit zvyšování výkonu procesoru?

- Dosavadní přístup k řešení tohoto problému
  1. zřetězené zpracování instrukcí.
  2. superskalární architektury
  3. využití architektur SIMD (obecně – využití možností, které nabízejí klasické architektury)

  Další zvyšování počtu kroků ve skalárních a superskalárních architekturách – složité řízení a zvyšování příkonu – existuje mez těchto architektur.

- Nové postupy uplatňované v moderních architekturách PC: vláknové architektury

  Principy těchto architektur jsou uplatňovány v moderních konstrukcích PC, navazují na principy využívané ve skalárních a superskalárních architekturách.

Vláknové architektury - pojmy

- **Proces**: program běžící na počítači. Je charakterizován těmito pojmy:

  Resource ownership (přidělení zdrojů)

  Procesu jsou přiděleny prostředky potřebné pro realizaci – paměťový prostor, V/V prostředky, V/V zařízení, soubory na externích pamětech.

  Scheduling/execution (plánování/provedení)

  V rámci jednoho procesu může být spuštěno více programů – je nutno plánovat. Proces je charakterizován stavem procesu: běží, připraven, ukončen,…
- **Důležitý prvek: přepínání procesů**

Vláknové architektury - pojmy

- **Vlákno:**

  Jednotka práce uvnitř procesu. Vlákno (program) – samostatně běžící část aplikace.
- Provádí se sekvenčně, vlákno je přerušitelné.

  Využívá zdroje přidělené procesu a data, která potřebuje, proto jsou ve hře pouze mechanismy plánování/provedení (nikoliv přidělení zdrojů).
- **Důležitý prvek: přepínání vláken** – probíhá mezi vlákny jednoho procesu.
- Přepínání vláken - jednodušší proces než přepínání procesů.
- Přepínání procesů – úklid obsahu registrů, …..

Vláknové architektury - pojmy

- Vláknové architektury musí být podporovány operačním systémem – není to pouze o podpoře vybavení počítače (hardware).
- Rozdíl mezi vláknovými architekturami – objem a typ vybavení (hardware), kterým je vláknová architektura podporována.
- Každé vlákno musí např. mít svůj vlastní čítač adres (anglicky program counter).

Příklady vícevláknových architektur

- Na známých architekturách si zobrazíme možnosti zpracování programu formou vláken.
- Budeme rozlišovat:

  prokládané vícevláknové zpracování

  blokové vícevláknové zpracování

  současné vícevláknové zpracování

  zpracování na více procesorech

Příklady vícevláknových architektur

- Prokládané vícevláknové zpracování

  Procesor zpracovává dvě nebo více vláken.

  Mezi vlákny se přepíná s každým synchronizačním pulsem.

- Blokové vícevláknové zpracování

  Instrukce jsou prováděny sekvenčné tak dlouho, dokud nedojde k události, která může způsobit zpoždění (např. cache miss). Pak se přepne na nové vlákno.

Příklady vícevláknových architektur

- Současné vícevláknové zpracování

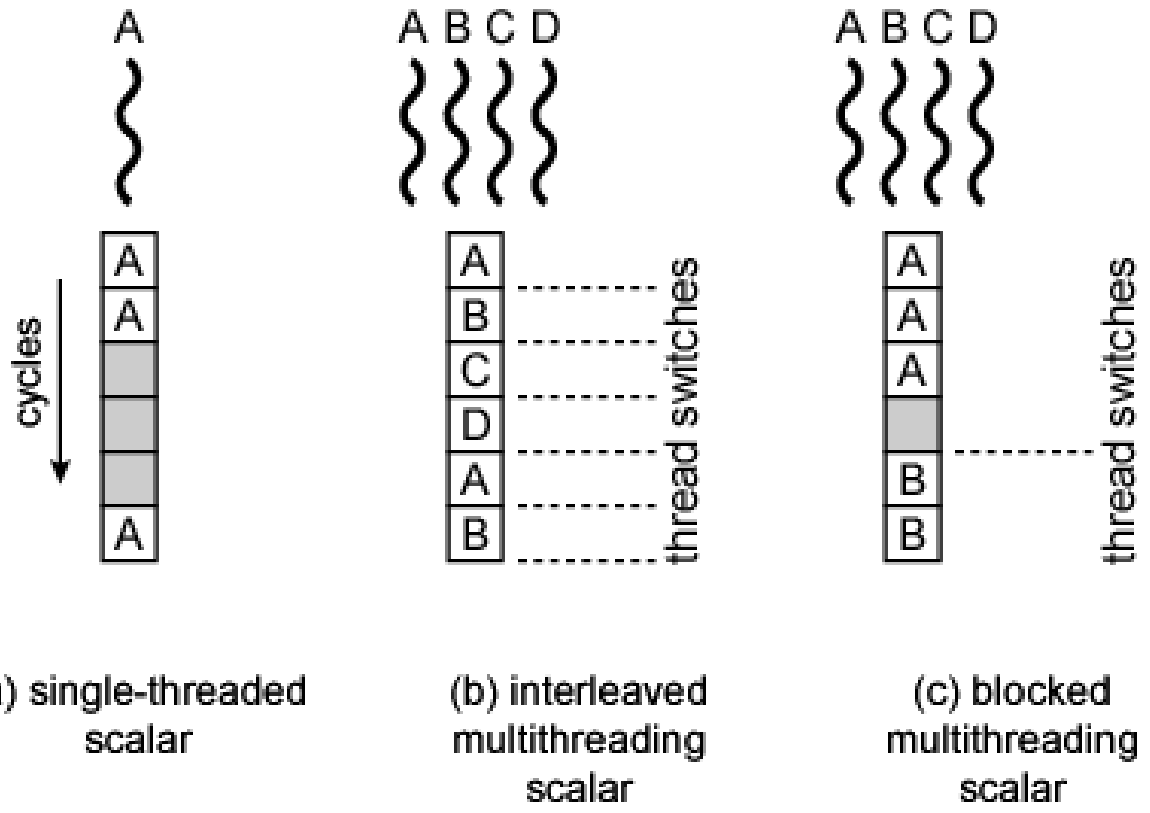
  Instrukce z jednotlivých vláken jsou zpracovávány v prováděcích jednotkách superskalárního procesoru.

- Zpracování na více procesorech

  Více procesorů, každý z nich zpracovává samostatné vlákno.

Skalární procesory a vlákna

A, B, C, D – vlákna



Single-threaded – jednovláknový, multithreading – vícevláknový, blocked - blokový

Skalární procesory a vlákna

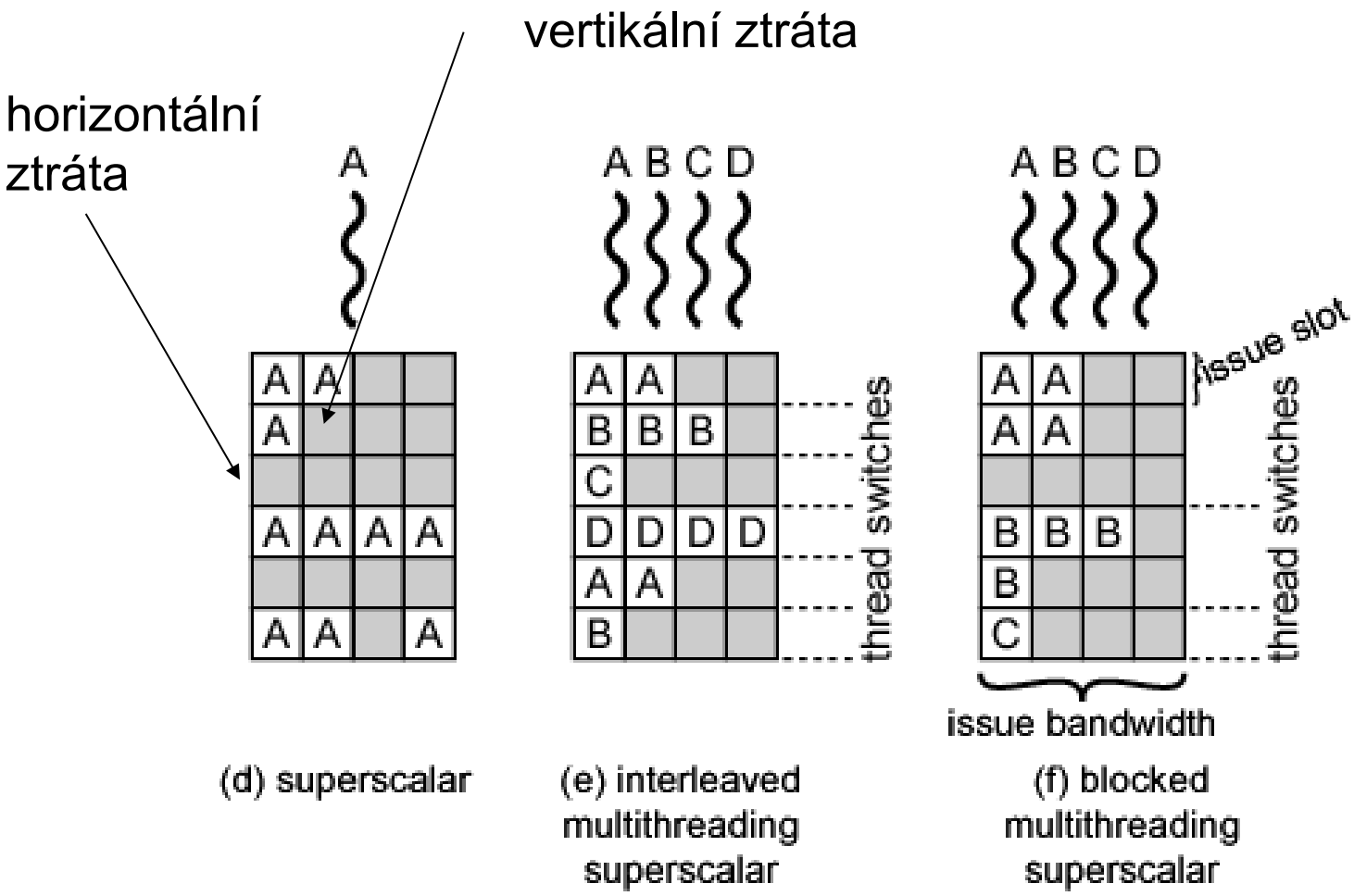
- Přepínání:

  Obr. b – mezi vlákny se přepíná bez režie (každé vlákno má své pracovní registry)

  Je to podmíněno také tím, že mezi vlákny nejsou žádné závislosti.

  Obr. c – nutný jeden cyklus na přepnutí

Superskalární procesory a vlákna



Superskalární procesory a vlákna

- Architektura d)

  Dlouhodobě byla viděna jako nejvýkonnější alternativa paralelního zpracování.

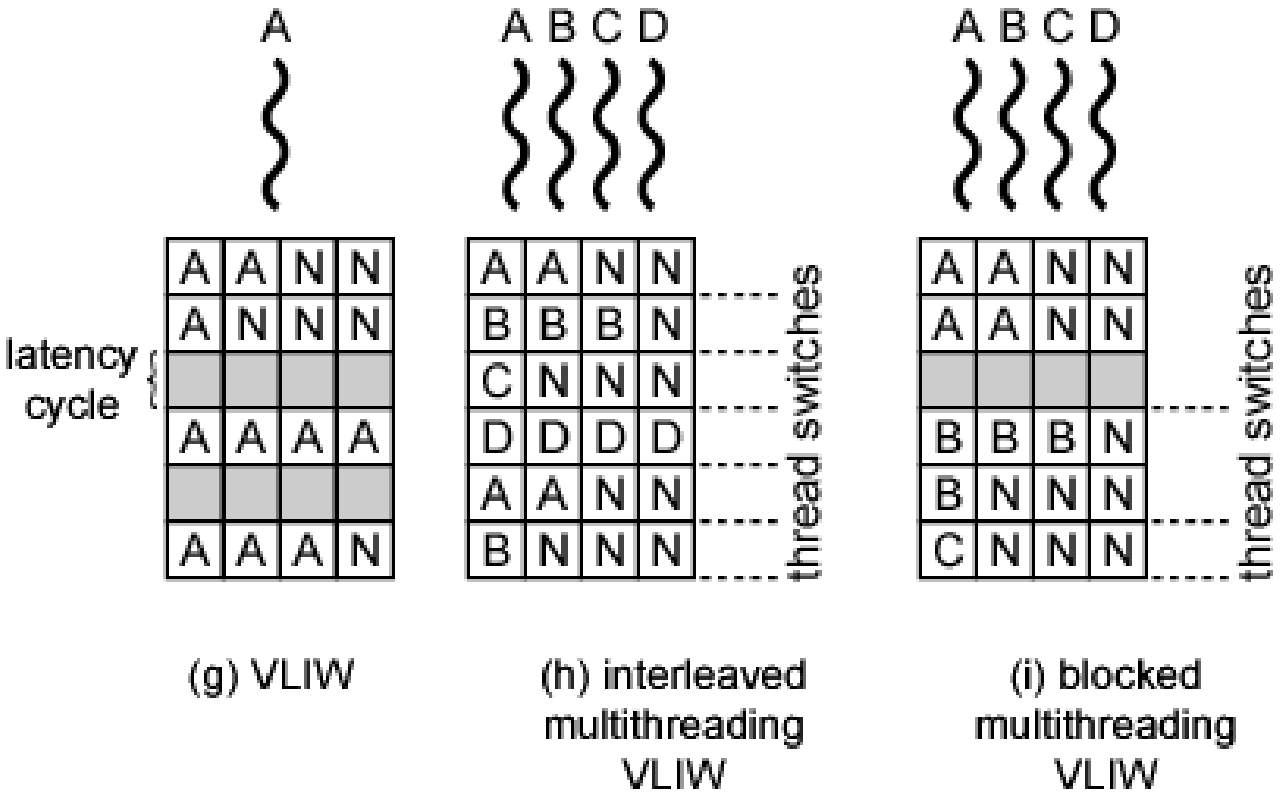
  Horizontální/vertikální ztráta – např. uplatnění párovacích pravidel.
- Architektura e)

  V rámci každého vlákna se provádí co nejvíce instrukcí – uplatnění párovacích pravidel.
- Architektura f)

  Jako jeden blok se provádějí instrukce pouze jednoho vlákna

Architektury na bázi VLIW a vlákna

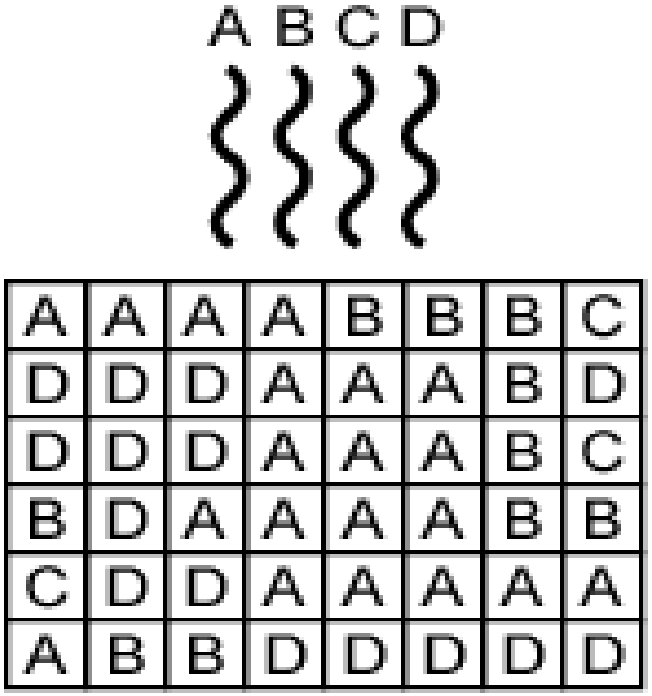
VLIW - Very Long Instruction Word
Uplatněno např. v architektuře IA-64.



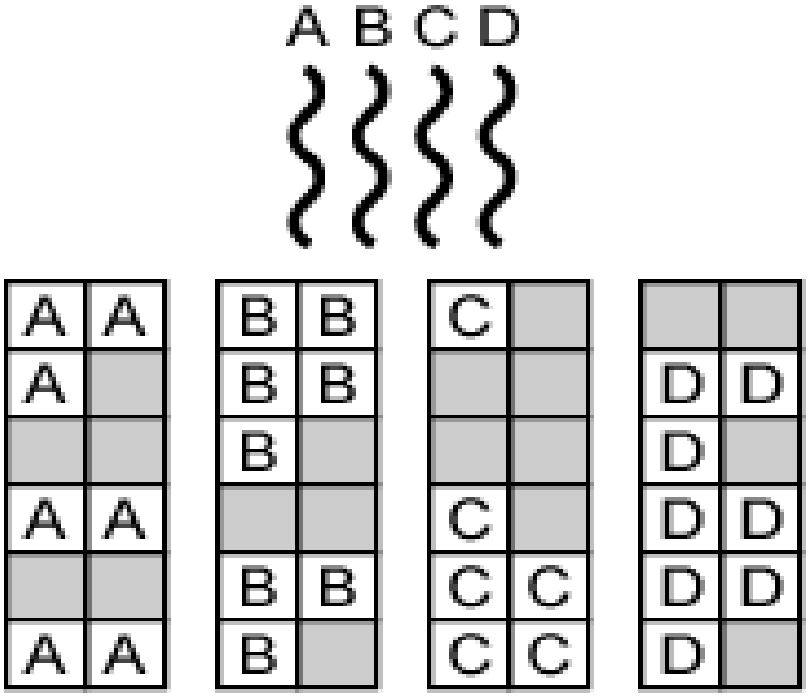
Architektury na bázi VLIW a vlákna

- Princip: během kompilace se do jednoho dlouhého slova vloží instrukce, mezi nimiž nejsou závislosti a mohou být realizovány paralelně.
- Pokud se nepodaří zaplnit instrukcemi celou šířku slova, pak se vloží NOP.

Výkonné vláknové architektury



(j) simultaneous multithreading (SMT)



(k) chip multiprocessor

Další pojmy

- **Simultaneous multithreading (SMT)** – technika pro zvýšení efektivity superskalárních architektur.
- Existence několika nezávislých vláken – lepší využití hardware, který je v každé frontě k dispozici.
- Pozor – zkratka SMT má i jiný význam: Surface Mounted Technology.

Další pojmy – pojem core

- Pojem core je odlišný od pojmu procesor.
- Core má svou vlastní sadu EU (execution units).

  Důsledek: dvě vlákna, každé existující na samostatném core, „nesoutěží“ o zdroje (tzn. EU).
- Jiná alternativa: core může sestávat z více logických procesorů.
- Příklad multi-core HT Technology: core se jeví operačnímu systému jako 2 procesory, dovoluje proto současné plánování dvou procesů.
- Použití: multi-threading aplikace.

V čem spočívala implementace vlastností Intel Netburst micro-architecture

- **Rychlé vyrovnávací paměti integrované do čipu (on-chip cache)**
- Na rozdíl od dřívějších typů to není pouze klasická rychlá vyrovnávací paměť L1, ale:
  - rychlá vyrovnávací paměť L1 pro data kapacity 8 kB,
  - rychlá vyrovnávací paměť L2, 8 way, kapacity 256 kB **(**v terminologii Intel **Advanced Transfer Cache**), instrukce a data
  - rychlá vyrovnávací paměť typu **Execution Trace Cache** kapacity 12K µop,

  Pojem **Advanced Transfer Cache** – vyrovnávací paměť L2 (integrována do čipu procesoru), pracuje na stejném kmitočtu jako procesor) – pojem fy Intel.

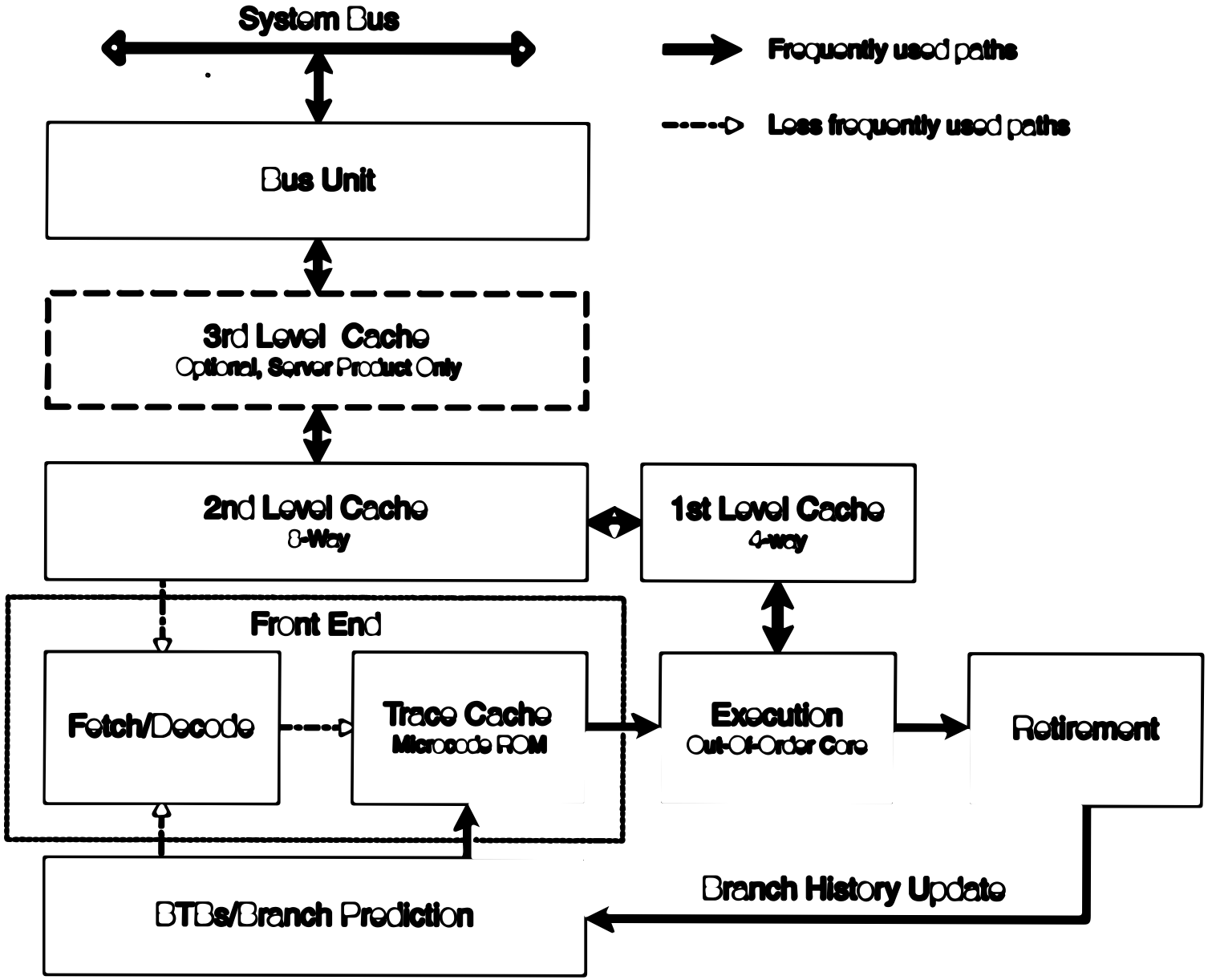
Rychlost komunikace procesoru s okolím

- Pentium III komunikovalo s okolím na frekvenci 100/133 MHz, Pentium 4 - komunikace s okolím synchronizována kmitočtem 266 MHz (později přenosy realizovány 4x v rámci jednoho cyklu).
- Dnes: 400/533/1066/1333 MHz.
- **Trend: zvyšování rychlosti komunikace procesoru s okolím.**

- Pojem FSB (Front Side Bus) – rozhraní procesoru (pouzdro)

Vlastnosti Intel NetBurst Micro-Architecture

- Hyperskalární (zřetězení instrukcí – více jak 10 jednotek) a superskalární architektura (minimálně dvě fronty).
- Dokonalé (?) uplatnění principů zřetězení, přičemž různé komponenty jsou synchronizovány různými frekvencemi, některé vyššími, některé nižšími frekvencemi než je frekvence procesoru.
- Synchronizace procesoru - frekvence 3 - 4 Ghz.
- Zvyšování kmitočtu procesoru – výrazná snaha o zvýšení počtu jednotek (kvůli zvyšování kmitočtu je to nutnost).
- Provádění mikroinstrukcí mimo pořadí.
- Konstrukce obvodů tak, aby se častěji prováděné instrukce realizovaly rychleji.
- **Trend: snaha o uplatnění principů architektur RISC na úrovni provádění mikroinstrukcí v procesorech Intel.**


System Bus
Frequently used paths
Less frequently used paths
Bus Unit
3rd Level Cache
Optional, Server Product Only
2nd Level Cache
8-Way
1st Level Cache
4-way
Front End
Fetch/Decode
Trace Cache
Microcode ROM
Execution
Out-Of-Order Core
Retirement
Branch History Update
BTBs/Branch Prediction

Funkce Front End

- Sestává ze dvou bloků:
  - Fetch/Decode
  - Execution Trace Cache
- Front End má tuto funkci:
  - čtení instrukcí, jejich dekódování a náhrada posloupnostmi mikrooperací (Intel rozlišuje tyto typy instrukcí: instruction, complex instruction, special purpose instruction).
  - přenos dříve dekódovaných instrukcí z Execution Trace Cache,
  - predikci výsledků instrukcí skoku.
- **Trend: architektura moderních mikroprocesorů fy Intel usiluje o originální řešení problémů různých zpoždění, k nimž dochází při provádění instrukcí. Mezi tato zpoždění patří, např.:**
  - **doba potřebná na dekódování instrukcí**
  - **doba potřebná na řešení problémů větvení programu (skoky).**

Execution Trace Cache

- **Princip:**
  - Instrukce jsou rozdekódovány (Translation Engine) do posloupnosti mikrooperací (μop).
  - Instrukce jsou reflektovány posloupností mikrooperací – tyto posloupnosti se nazývají traces (kopie, obrazy) a jsou uloženy do Execution Trace Cache.
  - Posloupnosti mikrooperací jsou uloženy tak, jak odpovídá toku programu.
- Instrukce skoku - v Execution Trace Cache jsou k těmto mikrooperacím uloženy do stejných řádků výsledky těchto skoků => zvyšuje se pravděpodobnost správné predikce, zvyšuje se podíl kódu prováděného z Execution Trace Cache.

Execution Trace Cache

- Pentium 4 – Execution Trace Cache uchovala až 12K μoperací.
- Procesor Pentium 4 - častěji prováděné instrukce byly realizovány z Execution Trace Cache, menší množství instrukcí se provádělo z paměti mikroprogramů (microcode ROM).
- Výsledek: jistá část instrukcí (počet závisí na velikosti Execution Trace Cache) má svou reprezentaci uloženou v Execution Trace Cache, pouze pro malou část je využívána paměť mikroprogramů.
- **Trend: zkrátit čas potřebný k dekódování instrukcí a zajištění přístupu k mikroprogramům.**

Jednotka Out-of-Order Core

- Jednotka Out-of-Order Core umí přeuspořádat provádění instrukcí tak, aby neutrpěla logika programu a byla přitom zohledněna konkrétní kriteria pro přeuspořádání.
- Princip: pokud nemůže být některá μop provedena, protože nemá k dispozici data, provede se jiná μop => tímto způsobem je možné odstranit možná zpoždění, která vzniknou v důsledku nedostupnosti dat.
- V mechanismech rozhodujících o zahájení realizace konkrétní μoperace se bere v úvahu také dostupnost potřebných hardwarových prostředků.
- Pořadí provádění mikrooperací se může modifikovat podle toho, zda jsou k dispozici hodnoty operandů pro patřičné mikrooperace a jsou volné hardwarové prostředky pro její realizaci.
- **Trend: optimalizovat principy realizace instrukcí zpracovávaných ve frontě s cílem zkrátit jejich provádění.**

Zpětné uspořádání výsledků posloupnosti instrukcí

- Tuto činnost provádí jednotka, kterou Intel označuje jako Retirement Section.
- Za ukončení mikrooperace se považuje stav poté, co je výsledek uložen do cílového registru.
- Jednotka Reorder Buffer (ROB) provádí zpětné uspořádání výsledků μinstrukcí tak, aby odpovídaly původní posloupnosti.
- Jednotka Retirement Section uchovává informaci o tom, jak dopadly skoky a předává ji do BTB (Branch Target Buffer).
- Podle této informace se obnovuje obsah BTB.

Techniky předvídání výsledků skoků

- Předvídání výsledků skoků je velmi důležité pro procesory zpracovávající fronty instrukcí.
- Techniky předvídání skoků (**Branch Prediction**) umožňují pokračovat ve zpracování programu na správném místě předtím, než je instrukce skoku zpracována a je znám výsledek skoku.
- Pojem **Branch Delay** reprezentuje zpoždění, které nastane, pokud dojde při předvídání výsledku skoku k omylu.
- Správná předpověď - Branch Delay je nulové.
- Mylná předpověď - Branch Delay bude odpovídat délce fronty (do fronty se musí načíst instrukce z jiné adresy než bylo předpovězeno a ta se do prováděcí jednotky dostane za takový počet cyklů, který odpovídá počtu komponent podílejících se na zpracování).
- **Trend: v oblasti předpovídání výsledků skoků se usiluje o inovaci těchto technik.**