32 AP0

Petr Svec

Mnohaúrovňová organizace počítače, virtuální stroje. Virtuální programovací prostředí a virtualizační techniky. Klasická registrově orientovaná architektura s kompletní instrukční sadou. Standardní systémové a I/O sběrnice počítačových systémů

1 Mnohaúrovňová struktura počítačů a virtualizace

1.1 Struktura

Viceúrovňová struktura má za cíl usnadnit práci programátorům, přidáním dodatečných úrovní abstrakce - především jazyků vyšší úrovně.

Druhou nejnižší úrovní (značeno L2) je strojový jazyk tj. nuly a jedničky. První úroveň nemá ani moc smysl uvažovat, protože se k ní normální smrtelník ani nedostane, viz. dále.

L2 je pro lidi samozřejmě nečitelný, což je důvodem existence vyšších úrovní, ale na druhou stranu je to to nejrychlejší, co na daném HW dostaneme, takže každý program ve výsledku běží na téhle úrovni.

1.2 Virtuální počítač

Na každé vyšší úrovni i než je strojová zavádíme virtuální počítač M_i s jazykem L_i. Program napsaný v L_i se překládá do L_{i-1} nebo je interpretován v M_{i-1} pro $i > 2$ (při i = 1 jsme na strojovém jazyce).

Pro připomenutí:

interpretace program v L_{i-1} zpracovává L_i jako data

kompilace instrukce z L_i jsou převedeny na instrukce v L_{i-1}

1.3 Běžná struktura současného počítače

Současný počítač se skládá z následujících úrovní:

L1 Mikrokód (mikroinstrukce) - překládá strojový kód na instrukce pro práci s obvody - interpretovaný, ale velice rychlý. Byl přidál až po strojovém kódu.

L2 Strojový kód

L3 Úroveň operačního systému (OS)

L4 Assembly lang. (ASM)

L5 High lvl lang.

Pozn.: ve slidech je uveden OS jako nižší než assembly lang. Osobně mi to přijde podivné, jelikož např. u RISCových procesorů bez instrukce násobení je nejprve potřeba funkce OS, která násobení realizuje, abychom dostali kód v ASM. Dalším příkladem je klíčové slovo "new" v C++ na alokaci paměti, které se také musí převést na systémové volání.

1.4 Strojová úroveň - M_2

- definovaná instrukční sadou
- instrukční formát se skládá z OPcode (kód instrukce) a 0-3 adres/registrů (při 0 je adresa/registr implicitní)
- u RISC je nejcastejsi 3 adresovy format u CISC 2 adresovy

1.5 Virtualizace

Principem virtualizace je skrytí nižších vrstev a implementace vrstvy, která se tváří jako samostatný HW.

Virtualizace se dělí na několik skupin:

- Na úrovni jazyka a byte kódu - virtualní stroje (Java, .NET)
- Emulace jiné počítačové architektury
- Nativní virtualizace - izolované prostředí poskytující shodný typ architektury pro nemodifikovaný OS
- Virtualizace s plnou podporou HW (například většina nových CISCových procesorů jí v sobě má)
- Částečná virtualizace - typicky jen adresní prostory
- Paravirtualizace
- Virtualizace na úrovni OS - oddělená uživatelská rozhraní

1.5.1 Hypervizor (Virtual machine monitor - VMM)

VVM je program, který zajišťuje základní práci s hostovanými OS, mezi které patří

- Monitorování hostovaných OS, ošetřování výjimek a emulace privilegovaných instrukcí
- Emulace periferií, přerušení, které mohou generovat, předávání dat do fyzických zařízení na hostujícím OS/systému
- Rozdělení zdrojů mezi hostované OS

Hypervizor může být implementovaný jako program/proces uživatelského prostoru v hostitelském OS (QEMU), s použitím HW akcelerace a podpory v jádře OS (KVM) nebo jako samostatný systém využívající systém v jedné doméně (jednom virtualizovaném prostoru vyhrazeném pro jeden hostovaný OS) pro komunikaci s HW a z něj pak posílá data ostatním (XEN).

1.5.2 Virtualizační program/systém Xen

V paravirtualizaci (XEN) nevolají systémová volání aplikace běžící v hostovaném OS přímo hostující kernel, ale hypervizor. Hypervizor přenáší část své práce na hostující systém místo aby prováděl syscall ve virtualizované doméně. Nevýhodou je, že hostovaný systém musí být upravený na míru pro hypervizor.

2 Klasická architektura

Pozn.: následující text je z větší části z wiki, jelikož ve slidech není téměř nic užitečného kromě obrázku.

Klasickou architekturou je myšlena CISC (dle toho mála co je ve slidech) a konkrétně architektura m68k tj. Motorola řada 68xxx.

- m68k má jak 16-bit tak 32-bit modely
- Big endian
- Má 8 datových a 8 adresových registrů, přičemž 8. adresový je User Stack Pointer (USP) - jeho chování je ovlivněno nasledujícím registrem
- status register (= PSW - Program Status Word) je 16-bitový. Spodních 8-bitů je pro tzv. Condition Code Register (CCR), horních 8 pro systém (sys. byte).

2.1 CCR

Čísla jsou indexy PSW

0 - carry bit - nastaví se na 1 při přenosu z nejvíce významného bitu při aritmetické operaci nebo když je třeba výpůjčka při odečítání

1 - overflow bit - při přetečení na 1

2 - zero bit - pokud jsou všechny bity operandů nebo výsledku nullové

3 - negative bit - pokud je nejvíce významný bit operandu nebo výsledku roven 1 (negativní výsledek v doplňkovém kódu)

4 - extended carry - používá se při rozšíření operací na práci s vícenásovnou přesností, má stejnou hodnotu jako carry

Pozn.: zbylé 3 bity nejsou ve slidech přímo zmíněny, ale na obrázku v nich uvedeném jsou nastaveny na 0. Víc jsem nehledal

2.2 System byte

- 8,9,10 - interrupt mask - definuje úroveň přerušení, pro kterou je přijetí žádosti blokované
- 14,15 - trace bits - pokud je některý z nich nastavený, tak dojde ke generování výjimky po provedená každé instrukce nebo po instrukcích ovlivňujících tok kódu
- 11 a 12 - jsou 0

3 Systemové a IO sběrnice

Pozn.: Ve slidech je vše ukazováno na příkladu PC, takže nedostatek obecnosti je záměrný a není to chyba

3.1 Dvoubodové spojení (point-to-point)

- Při malém počtu zařízení je nejjednodušší, při větším ale značně komplikované
- Poskytuje vyšší výkon než sběrnice, jelikož se žádné zařízení nemusí o své připojení dělit s jiným

3.2 Sběrnice (bus)

- Spousta drátů, na které se připojují zařízení v rámci počítače, zjednodušuje komunikaci oproti point-to-point spojení všech komponent
- V jednom počítači není typicky jen jedna sběrnice, ale hned několik - procesorová, systémová, lokální a IO sběrnice
- Podle typu "obsahu", který obsluhuje existují 3 typy - adresová, datová, řídící. Může být i kombinovaná

Nejblíže CPU jsou Fronst-Side-Bus (FSB) a Back Side Bus (BSB). FSB je připojení na North Bridge - systémový čip, který obsluhuje paměti a dnes již prehistorické AGP (na to se připojovalo GPU). V současné době je North Bridge většinou již součástí CPU (AMD a Intel). BSB jsou zadní vrátka do paměti pro L2 cache, která z ní čte přímo (BSB je na čtení z paměti zoptimalizovaná, a tak poměrně rychlá).

S North Bridge je spojený South Bridge, na který se připojují ostatní sběrnice a zařízení (PCI, PCIe, USB etc.).

3.3 Multiplexovaná sběrnice (time division multiplexing)

Sběrnice, kde jsou data nebo adresy větší velikosti než jsou vodiče schopné přenést, a tak se dělí na části. Příklad: sběrnice přenese najednou 16-bit, ale naše data mají 32-bit, proto se data rozdělí a pošlou na dvakrát. Jsou také případy, kdy se posílají adresy i data po stejných drátech.

3.4 PCI

- PCI je sběrnice z dob, kdy dinosauři chodili po Zemi
- Komunikace je paralelní. To mimo jiné vyžaduje, aby byl přítomen arbitr sběrnice - komponenta určující, které z PCI zařízení posílá po sběrnici data (a adresy) v daný okamžik
- Vždy, když chce libovolné PCI zařízení posílat data po sběrnici, musí nejprve požádat arbitra o přidělení práva
- Každého přenosu/transakce se účastní 2 zařízení - initiator (Bus Master) a target. Arbitr také řeší případ více Bus masterů (multimaster sběrnice)
- Přenos po sběrnici má 2 fáze - adresovou a datovou
- Zařízení na sběrnici mají společný 4-bit přerušovací podsystém (narozdíl od ostatních drátů nejsou paralelní)
- Sběrnice je multiplexovaná, adresy i data se posílají přes stejné piny PCI zařízení

3.5 PCI vs. PCI-Express (PCIe)

- PCIe má narozdíl od PCI přepínač, na který jsou připojena jednotlivý PCIe zařízení. Přepínač samotný je spojený se sběrnicí. Komunikace je tím pádem, na rozdíl od PCI, sériová
- Rychlost PCIe zařízení se udává počtem linek (x1, x16 atd.). Každá linka se skládá z jednoho nebo více proudů a každý proud je tvořen 2 vodiči (jeden v každém směru)
- Komunikace je full duplex - 1 bit za takt v obou směrech