#32 APO-2

Petr Svec

Mnohaurovnova struktura pocitacu a virtualizace

Struktura

Viceurovnova struktura ma za cil usnadnit praci programatorum, pridanim dodatecnych urovni abstrakce - predevsim jazyku vyssi urovne.

- druhou nejnizsi urovni (znaceno L2) je strojovy jazyk tj. nuly a jednicky (prvni nema ani moc smysl uvazovat, jelikoz se k ni normalni smrtelnik ani nedostane viz. dale).
- pro lidi samozrejme necitelny, coz je duvodem existence vyssich urovni, ale na druhou stranu to nejrychlejsi co na danem hardwaru dostaneme, takze kazdy program ve vysledku bezi na tehle urovni.

Virtualni pocitac
- na kazde vyssi urovni nez je strojova zavadime virtualni pocitac M_i s jazykem L_i. Program napsany v L_i se preklada do L_{i-1} nebo je interpretovan v M_{i-1} pro $i > 2$ (pri i = 1 jsme na strojovem jazyce).
- pro pripomenuti:

  - interpretace = program v L_{i-1} zpracovava L_i jako data
  - kompilace = prevod instrukci z L_i na instrukce v L_{i-1}

Bezna struktura soucasneho pocitace
Serazeno od nejnizsi uroven po nejvysi se soucasny pocitac sklada z nasledujicich urovni:

1. L1 - mikrokod (mikroinstrukce) - preklada strojovy kod na instrukce pro praci s obvody - interpretovany, ale velice rychly. Byl pridan az pozdeji tj. po strojovem kodu.
2. L2 - strojovy kod
3. L3 - uroven operacniho systemu (OS)
4. L4 - assembly lang.(ASM)
5. L5 - high lvl lang.

Pozn. ve slidech je uveden OS jako nizsi nez assembly lang. Osobne mi to prijde podivny, jelikoz napr. u RISCovych procesoru bez instrukce nasobeni je nejprve treba funkci OS, ktere nasobeni realizuji abychom dostali kod v ASM. Dalsim prikladem je keyword "new" v C++ na alokaci pameti, ktere se take musi preves na systemove volani.

Strojova uroven - M_2
- definovana instrukcni sadou
- instrukcni format se sklada z OPcode (kod instrukce) a 0-3 adres/registru (pri 0 je adresa/registr implicitni)
- u RISC je nejcastejsi 3 adresovy format u CISC 2 adresovy

Virtualizace

Principem virtualizace je skryti nizsich vrstev a implementace vrstvy, ktera se tvari jako samostatny HW.

V. se deli na nekolik skupin:

- Na urovni jazyka a byte kodu - virtualni stroje (Java, .NOT)
- Emulace jine pocit. architektury
- Nativni virtualizace - izolovane prostredi poskytujici shodny typ architektury pro nemodifikovany OS
- Virtualizace s plnou podporou HW (napriklad vetsina novych CISCovych procesoru ji v sobe ma)
- Castecna virtualizace - typicky jen adresni prostory
- Paravirtualizace
- Virtualizace na urovni OS - oddelena uzivatelska rozhrani

Hypervizor (Virtual machine monitor - VMM)
- program, ktery zajistuje zakladni praci s hostovanymi OS, mezi ktere patri:

- monitorovani hostovanych OS, osetrovani vyjimek a emulace privilegovanych intrukci
- emulace periferii, preruseni, ktere mohou generovat, predavani dat do fyzickych zarizeni na hostujicim OS/systemu.
- rozdeleni zdroju mezi hostovane OS

Hypervizor muze byt implementovany jako program/proces uzivatelskeho prostoru v hostitelskem OS (QEMU), s pouzitim HW akcelerace a podpory v jadre OS (KVM) nebo jako samostatny system vyuzivajici system v jedne domene (jednom virtualizovanem prostoru vyhrazenem pro jeden hostovany OS) pro komunikaci s HW a z nej pak posila data ostatnim (XEN).

Xen (to je virtualizacni "program"/system)
V paravirtualizaci (XEN) systemova volani aplikace bezici v hostovanem OS nevolaji primo hostujici kernel, ale hypervizor. Hypervizor prenasi cast sve prace na hostujici sys. misto aby provadel syscall ve virtualizovane domene. Nevyhodou je, ze hostovany sys. musi byt upraveny na miru hypervizoru.

Klasicka architektura

Pozn. nasledujici text je z vetsi casti z wiki, jelikoz ve slidech neni temer nic uzitecnyho krome obrazku.

Klasickou architekturou je myslena CISC a (dle toho mala co je ve slidech) konkretne architektura m68k tj. Motorola rada 68xxx.

- m68k ma jak 16-bit tak 32-bit modely
- big endian
- ma 8 datovych a 8 adresovych registru, pricemz 8. adresovy je User Stack Pointer (USP) - jeho chovani je ovlivneno nasledujicim registrem
- status register (= PSW - Program Status Word) je 16-bitovy. Spodnich 8-bitu je pro tzv. Condition Code Register (CCR), hornich 8 pro system (sys. byte).

CCR (cisla jsou indexy PSW)

0 - carry bit - nastavi se na 1 pri prenosu z nejvice vyznam. bitu pri arit. operaci nebo kdyz je treba vypujcka pri odecitani

1 - overflow bit - pri preteceni na 1

2 - zero bit - pokud jsou vsechny bity operandu nebo vysledku nulove

3 - negative bit - pokud je nejvice vyznamny bit operandu nebo vysledku roven 1 (negativni vysledek v doplnkovem kodu)

4 - extended carry - pouziva se pri rozsireni operaci na praci s vicenasobnou presnosti, ma stejnou hodnotu jako carry

Pozn. zbyle 3 bity nejsou ve slidech primo zmineny, ale na obrazku v nich uvedenem jsou nastaveny na 0. vic jsem nehledal
System byte

- 8,9,10 - interrupt mask - definuje uroven preruseni pro kterou je prijeti zadosti blokovane
- 14,15 - trace bits - pokud je nektery z nich nastaveny, tak dojde ke generovani vyjimky po provedeni kazde instrukce nebo po instrukcich ovlivnujicich tok kodu

11. a 12. jsou 0

Systemove a IO sbernice

Pozn. Ve slidech je vse ukazovano na prikladu PC, takze nedostatek obecnosti je zamerny a neni to chyba.

sbernice(bus)

- spousta dratu, na ktery se pripojuji zarizeni v ramci pocitace, zjednodussuje komunikaci oproti point-to-point spojeni vsech komponent
- v jednom pocitaci neni typicky jen jedna sbernice, ale hned nekolik - procesorova, systemova, lokalni a IO sbernice
- podle typu "obsahu", ktery obsluhuje existuji 3 typy - adresova, datova, ridici. Muze byt i kombinovana

Nejblize CPU jsou Front-side Bus (FSB) a Back-side Bus (BSB). FSB je pripojeni na North Bridge - systemovy cip, ktery obsluhuje pameti a dnes jiz prehistoricke AGP (na to se pripojovalo GPU). V soucasny dobe je North b. vetsinou jiz soucasti CPU (AMD a Intel). BSB jsou zadni vratka do pameti pro L2 cache, ktera z ni cte primo (BSB je na cteni z pameti zoptimalizovana, a tak pomerne rychla).
S North b. je spojeny South bridge, na ktery se uz pripojuji ostatni sbernice a zarizeni (PCI, PCIe, USB etc.).

dvoubodove spojeni (point-to-point)

- pri malem poctu zarizeni je nejjednodussi, pri vetsim ale znacne komplikovane
- poskytuje vyssi vykon nez sbernice, jelikoz se zadne zarizeni nemusi o sve spojeni delit s jinym

Multiplexovana sbernice(time division multiplexing) - sbernice, kde jsou data nebo adresy vetsi velikosti nez jsou vodice schopne prenest, a tak se deli na casti. Priklad: sbernice prenese najednou 16-bit, ale nase data maji 32-bit → rozdeli se a poslou na dvakrat. Jsou take pripady,

kdy se posilaji adresy i data po stejnych dratech.

PCI
- PCI je sbernice z dob kdy dinosauri chodili po Zemi
- komunikace je paralelni. To mimo jine vyzaduje, aby byl pritomen arbitr sbernice - komponenta urcujici, ktere z PCI zarizeni posila po sbernici data (a adresy) v dany okamzik.
- vzdy, kdyz chce lib. PCI zarizeni posilat data po sbernici musi nejprve pozadat arbitra o prideleni prava
- kazdeho prenosu/transakce se ucasni 2 zarizeni - initiator(Bus Master) a target. Arbitr take resi pripad vice bus masteru (multimaster sbernice)
- prenos po sbernici ma 2 faze - adresovou a datovou
- zarizeni na sbernici maji spolecny 4-bit prerusovaci podsystem (narozdil od ostatnich dratu nejsou paralelni)
- sbernice je multiplexovana, adresy i data se posilaji pres stejne piny PCI zarizeni

PCI vs. PCI-Express (PCIe)
- PCIe ma narozdil od PCI prepinac, na ktere jsou pripojena jednotliva PCIe zarizeni. Prepinac samostny je je spojeny se sbernici. Komunikace je timpadem, narozdil od PCI, seriova
- rychlost PCIe zarizeni se udava poctem linek (x1, x16 atd.). Kazda linka se sklada z jednoho nebo vice proudu a ten je tvoren 2 vodici (jeden v kazdem smeru)
- komunikace je full duplex - 1 bit za takt v obou smerech