#31 APO-1

Petr Svec

Architektura pocitace

Klasickty von Neumanovsky pocitac:

- Procesor
  - Radic
  - ALU (Arithetic-Logical Unit)
- Pamet
- I/O System

Radic se dal sestava z casti datove a vlastni ridici casti. Datovou lze navic rozdelit na registry a dalsi obvody.
Dulezite registry:

- Program Counter (PC)
- Instruction Register (IR)
- Stack Pointer (SP)

Zakladni cyklus pocitace (predchazi mu inicializace registru apod.):

1. Cteni instrukce
   - PC → adresa Hlavni Pameti (HP)
   - Cteni obsahu
   - Prectena data se ulozi do IR
   - Inkrementace PC o delku intrukce
2. precteni opcode(kod instrukce - typicky nekolik prvnich bitu v zavilsti na velikosti instrukcni sady)
3. provedeni instrukce - zahrnuje i dalsi veci jako je cteni operandu
4. Pokud doslo k preruseni, tak se zpracuje.
5. skok na bod 1.

Preruseni jsou dva druhy a u obou dochazi k preruseni sekvence vykonavaneho kodu a prechazi se na obsluhu.

- **Vnejsi preruseni** - asynchroni obsluha vnejsi udalosti, napr. reakce OS na event ze vstupni periferie.
- **Vnitrni preruseni**(Vyjimka) - primo od CPU, ktere takto reaguje na problem pri zpracovani instrukce napr. deleni nulou
- **Synchronni soft. preruseni** - zamerne preruseni vznikle vlozenim patricne instrukce na misto v kodu.

Kazda instrukce se sklada z opcode a operandu. Operandy muzou byt registry(resp. jejich "indexy"), adresa (napr. pro skoky)

RISC/CISC

- **RISC**(Reduced Instruction Set Computer) je architektura CPU, ve ktere je velice omezena intrukcni sada (IS) napr. jen na scitani, bitove posuny, nejake cteni/ukladani atd. Zbytek se realizuje softwarove. Vsechny instrukce maji pevny format a delku a take stejnou dobu vykonani. Velikost IS taky implikuje potrebu mnohem mensiho poctu tranzistoru, coz je mimo jine duvod proc treba ARMy moc nezerou. Zastupci: ARM, MIPS, PowerPC
- **CISC**(Complex Instruction Set Computer) je architektura (prozatim) vetsiny uzivatelskych CPU. Instrukcni sada byva casto pomerne velka, instrukce nemusi byt stejne dlouhe a jejich vykonani muze trvat ruzne dlouho. Zastupci: x86, amd64

Site procesoru a paralelni architektury

existuji 4 typy struktur procesoru:

1. **SISD** - Klasicky procesor tj. jednoduchy tok instrukci i dat
2. **SIMD** - Procesorove pole - jed. tok instrukci, vicenasobny tok dat. napr. GPU
3. **MISD** - Piplene
4. **MIMD** - Multiprocesorove pole

Propojovaci site
Zajistuji propojeni a komunikaci mezi procesory, deli se na staticke a dynamicke, pricemz ve statickych jsou spoje nemenne a v dynamickych naopak. U dynamickych jsou spojovaci spinace rizeny bud lokalne, kdy ma kazda skupina svuj radic, nebo centralne, kdy existuje jen jeden.
Staticke prop. site jsou: linearni, stromova, kruhova, mrizova, hvezdicova, krychlova a polygonalni(uplny graf)

- procesory mohou byt skalarni nebo vektorove
- skalarni procesory jsou bezne procesory
- vektorove p. provadi jednu operaci nad nekolika operandy → odstranuje se tim rezije spojena indexovanim jednolivych prvku vektoru.

Paralelizovane SISD

- samotne SISD je vetsina pocitacu zalozena na von Neumanove architekture
- palarelizovane SISD jsou systemy postavene na architekture VLIW(= very long instruction word), zalohovane systemy a systemy pouzivajici pipelining

Architektura VLIW

- velice dlouhe instrukce, ktere jsou rozdelene na nekolik casti/useku, ktere jsou zpracovavany paralelne
- jednotlive exekucni jednotky(EJ) jsou propojeny
- operacni pamet je rozdelena podle usporadani EJ tj. kazdy blok pameti odpovida jedne EJ

Zalohovane systemy - systemy, ve kterych bezi paralelne nekolik SISD tj. vsechny provadi stejny vypocet nad stejnou sadou dat ,pricemz vysledek ze vsech jednotlivych procesoru/pocitacu je porovnavan v komparatoru. Cilem je zvyseni spolehlivosti a bezpecnosti.

Paralelni systemy SIMD
SIMD systemy jsou ty, ve kterych se zpracovavaji dobre rozdelitelna data. Idealnim prikladem

jsou matice, ktere jsou takovym gró tohoto oboru, jelikoz se s nimi numericky pocitaji diferencialni rovnice, pouzivaji se k reprezentaci obrazu apod. Konkretni aplikace z toho plynouci: pocitani aerodynamiky, meteorologie, temer cokoliv co se tyka zpracovani obrazu.

Princip prace spociva v soucasnem zpracovani nekolika prvku napr. nekolik prvku z pole, kdy procesory z procesoroveho pole(PP) provadeji synchronne stejnou operaci(instrukci). Procesory jsou rizene spolecnym radicem.

1. SIMD s lokalni pameti
   - PP je rizeno univerzalnim pocitacem/procesorem, ktery zpracovava nadrizeny program → rozhoduje o maticovych ulohach a zabezpecuje prenos dat na procesory v poli
   - radic PP sam zpracovava skalarni a ridici instrukce, zatimco vektorove nechava zpracovat PP
   - kazdy procesor ma svou vlastni pamet operandu
   - procesory si mezi sebou posilaji data pres propojovaci sit
2. SIMD se sdilenou pameti
   - narozdil od SIMD s lok. pameti je v tomto pripade pamet od procesoru oddelena a komunikace probiha pres propojovaci sit
   - pridelovani pameti do procesoru zajistuje radic
   - pocet pametovych modulu muze byt jiny nez pocet procesoru

Paralelni systemy MIMD
- kazdy procesor zpracovava instrukce a data sveho vlasntiho programu
- deli se na tesne vazane a volne vazane

1. tesne vazane
   - kazdy procesor ma malou vyrovnavaci pamet pro data
   - procesory maji sdilenou operacni pamet, ktera je oddelena od procesoru a pripojena spolu s periferiemi na propojovaci sit, pres kterou komunikuji s CPU
   - periferie maji malou autonomii
   - propojovaci sit umoznuje lib. propojeni
2. volne vazane
   - procesory maji vlastni (lokalni)pamet a vlastni periferie, pricemz lokalni pamet obsahuje jak program, tak data
   - propojovaci sit byva staticka
     - Hierarchicka organizace sbernic - procesory nejnizsi urovne spolu s pametmi seskupeny do clusteru, ktere jsou pripojeny komunikacnimi moduly na sbernice vyssi hierarchie
     - Organizace do n-rozmerne krychle(nebo mrize) - kazdy proc. modul ma ma 8(4) komunikacnich procesoru pro pripojeni. Casti krychle lze dynamicky pridelovat pro ruzne ulohy. Je vyzadovan nadrizeny pocitac.
   - rizeni je slozitejsi nez u volne vazanych, ale na druhou stranu jsou tyto systemy odolnejsi vuci porucham a vypocty lze navic znekolikanasobit podle potreby. Jsou pouzivany tam, kde je treba vysoka spolehlivost

NUMA(Non-Uniform Memory Access)
- pouziva se u MIMD systemu, kde ma zajistit kratsi cekani na zapis nebo cteni do pameti tim, ze kazdemu procesoru je poskytnuta samostatna pamet
- pouziva tesnejsi vazbu vice CPU v uzlu, ktere jsou dale propojene do vetsich celku

Hierarchie pameti

- duvod hierarchizace - rychlejsi pamet je drazsi, hierarchizaci dostaneme system rychlosti se blizici tomu, ktery by pouzival vyhradne tu nejrychlejsi z pameti z hierarchie.
- hlavni myslenka - bezici programy pouzivaji v dany okamzik ke svemu behu jen cast adresoveho prostoru Kterou cast bude program pouzivat se rozhoduje podle nasledujicich dvou faktoru:

- Casova lokalita - co se pouzilo nedavno se brzo pouzije znova (soft. cykly, promenne)
- Prostorova lokalita - polozky blizko aktualne pouzivanych budou brzo treba (sekvenci provadeni kodu)

- pametova hierarchie se bezne sklada (smerem od te nejblize procesoru) z nekolika cache (L1,L2 a cim dal casteji i L3), operacni pameti a pevneho disku.
- data v pameti se hledaji smerem od CPU

Virtualizace pameti

- virtualni pamet (VP) je uroven abstrakce postavena nad vsemi zdroji dostupne pameti
- umoznuje procesu/programu dotazovat se pouze na logicke adresy (LA) a nezabyvat se jestli to je napr. v RAM nebo na HDD. VP prostor tedy muze byt mnohem vetsi nez je velikost fyzicke pameti.
- prevod mezi VA a fyzickou adresou casto zajistuje procesor (hardwarove)
- bezne implementovano pomoci strankovani, kde je strankovana jak operacni pamet, tak misto na HDD
- jednotka LA prostoru je stranka u fyzickeho adr. prostoru (FAP) to je ramec -kazdemu procesu je prideleno urcite mnozstvi stranek. ty si kazdy proces drzi ve sve Tabulce stranek
- nekolik prvnich bitu logicke adresy je indexem do tabulky stranek, na kterem lezi hledana fyzicka adresa (FA). Zbytek tvori offset jak ve strance, tak v ramci (za predpokladu, ze jsou stejne velke). Spolu s FA je na stejnem indexu take nekolik priznaku - validity bit (pritomnost stranky ve FAP), dirty(obsah modifikovan), access rights
- v pripade, ze je ramec prazdny, tak se hledana stranka pomoci DMA nacte do onoho ramce.
- pokud je nedostatek pameti, tak se odebere nejaky (nejake) ramec (ramce) na zaklade nektereho z algoritmu na vyber "obeti" (nejspise LRU - Last recently used). V pripade, ze byt aktivni dirty bit, tak se puvodni data nejprve zapisi na disk. Nakonec se aktualizuje page table.

Prerusovaci a I/O podsystem

I/O podsystem
Druhy periferii:

1. vystupni
2. vstupni
3. obousmerne - napr. HDD

Metody prenosu z/na periferie:

1. Programovy kanal - pooling(= cekani ve smycce), nejhloupejsi reseni
2. Programovy kanal s prerusenim. IO operace je zahajena na zadost programu pomoci OS. Existuji dve varianty:
   - synchronni - program ceka na dokonceni IO operace
   - asynchronni - p. neceka na dokonceni IO a muze bezet soubezne s ni. Jakmile jsou data dostupna perif. vyvola preruseni a dojde k jejich. precteni

3. Direct memory access (DMA)
4. Autonomni kanal

DMA
- prenos je realizovany specialni jednotkou bez prime ucasti OS. Ten jen nastavi parametry prenosu = kolik bytu a do jakeho bufferu a na jake adrese. Nasledne radic periferie inicializuje DMA prenos, ktery probiha dokud neni precten pozadovany pocet bytu definovany OS. Po ukonceni prenosu je vygenerovano preruseni.
- vyhodou je, ze velke datove prenosy nevytesnuji data z cache
- DMA radic nemusi mit jen disky, ale napr. i sitove rozhrani

Autonomni kanal(Bus Master DMA) (Pozn. to co je ve slidech k BM DMA je temer totozny s beznym DMA, presto to uvadim. na wiki jsem nasel ale neco navic.)
- inteligence presunuta do zarizeni
- periferie je doplnena o vlastni radic
- prubeh prenosu

1. nadrizeny procesor vlozi sekvenci datovych bloku do pameti
2. nakonfiguruje nebo primo naprogramuje radic periferie. ta provede sekveni prenosu
3. po uplnem nebo castecnem preruseni je informovan procesor

- - - - - -
Dle wikipedie se Bus Master DMA od klasickeho lisi tim, ze periferie muze dostat kontrolu nad pametovou sbernici a zapsat tak primo do pameti bez jakehokoliv zapojeni ze strany CPU.

Vyjimky a preruseni

- Vyjimky
  - pro MIPS napr. mat. preteceni, nactena neznama instr., systemove volani
  - nedostupnost dat nebo selhani zapisu
- Preruseni
  - maskovatelna - lze zakazat ve stavovem slovu CPU
  - nemaskovatelna - casto osetreni HW chyb, hlidaci obvod

Vyjimky jsou prijaty temer vzdy, preruseni jen pokud jsou nemaskovatelna nebo povolena.

Zpracovani vyjimky/preruseni:

1. stavove slovo(PSW) a PC se ulozi bud na zasobnik nebo do spec. registru
2. PC se nastavi na adresu obsluzne rutiny pripadajici dane vyjimce pripadne i cislu zdroje preruseni, ktera je nasledne vykonana
3. V zavislosti na typu vyjimky/preruseni doje ke specifickemu zpracovani
4. provede se instrukce CPU pro uvedeni do stavu pred zpracovanim vyjimky/preruseni (instrukce navratu z preruseni) a obnovi se puvodni registry (PC a PSW)

Pri spravnem obslouzeni vyjimky by puvodni program nemel primo poznat, ze k preruseni doslo.

Urceni zdroje vyjimky/preruseni

- soft. hledani
  - veskera preruseni a vyj. spousteji rutinu od stejne adresy. rutina zjisti duvod preruseni ze stavoveho registru (!= PSW)
- vektorova obsluha preruseni
  - cislo zdroje zjisti CPU
  - v pameti se nachazi na pevnem miste (specifikovane ridicim registrem VBR) tabulka vektoru preruseni, CPU prevede cislo zdroje na index a z nej nacte v poli slovo, kt. vlozi do PC
- nevektorova obsluha vice pevne urcenych adres podle priorit
- casto i kombinovane

Async vs. sync preruseni: async. nejsou vazane na instrukci, zatimco sync ano