

Principy počítačů a operačních systémů

Instrukce

Zimní semestr 2007/2008

Architektura z pohledu programátora

- styčný bod mezi návrhářem počítače a programátorem
- v abstraktním smyslu programátor nemá znát fyzický návrh – obvodové řešení
- musí ale znát logický návrh, tedy např. registrový a adresový model

Instrukční sada

Nutné vlastnosti

- funkční úplnost
 - ♦ musí dovolit uživateli formulovat libovolnou úlohu zpracování jeho dat na vyšší úrovni
- účinnost
 - ♦ často opakované funkce by měly být provedeny rychle a za pomoci malého počtu instrukcí
- jasně definované zdroje
 - ♦ požadavky na zdroje jasně určitelné

Instrukční sada

Doporučené vlastnosti

- ortogonalita
 - ♦ definice instrukcí, datových typů a způsobů adresování jsou nezávislé
- kompatibilita
 - ♦ s existujícím softwarem i hardwarem

Obsah instrukce

Každá instrukce musí obsahovat

- operační kód
 - ♦ co se má vykonat
- odkaz a určení zdrojových operandů
 - ♦ s čím se to má vykonat
- umístění a určení výsledku
 - ♦ co se má udělat s výsledkem
- odkaz na další instrukci

Modalita instrukce

Variabilita operace a operandů

- modalita operačního kódu
 - ♦ např. směr rotace
- modalita operandů
 - ♦ např. velikost a umístění
- modalita ochrany (paměti)
 - ♦ např. úroveň procesu

Modalita operačního kódu

Upřesnění operace

- směr a vzdálenost rotace
- způsob testování podmínky v podmíněném skoku
- směr operace load/store
- použití operandů (paměť / registry)
- úprava výsledku po provedení operace

Modalita operandů

Specifikace operandů

- způsob adresace
 - ♦ přímé, nepřímé, ...
- velikost a typ operandů
 - ♦ stejnou instrukci možno vykonat nad různými daty
- ovlivňuje přípravu instrukce
 - ♦ vícestupňové načítání
 - ♦ vícenásobné načítání

Modalita ochrany

Kontext běžícího vlákna

- přístupová práva k operandům
 - ♦ ochrana paměti
- právo vykonání instrukce
 - ♦ ochrana OS, procesu

Syntaxe instrukční sady

Formát zápisu instrukce v paměti

- nutná součást návrhu instrukční sady
- určuje části, ze kterých je instrukce složena
 - ♦ operační kód (instrukční kód)
 - ♦ argumenty
- pro dekodér musí být co nejjednodušší

Operační kód

Specifikace operace v instrukci

- vyžadována jasná definice, co má instrukce provést
- z návrhu počítače plyne, co znamenají jednotlivé části operačního kódu

... později

Dělení instrukcí (1)

Podle počtu operandů

- bezadresové
- jednoadresové
- dvouadresové
- tříadresové
- (čtyřadresové)

Bezadresové instrukce

Bez operandů

- ♦ NOP, RET

Operandy dány implicitně

- konkrétní operand dán operačním kódem
 - ♦ CLI, TBA
- zásobníková architektura
 - ♦ všechny operandy jsou na zásobníku
 - ♦ instrukce ze zásobníku vyzvedne potřebné operandy
 - ♦ provede se požadovaná operace
 - ♦ případný výsledek je uložen zpět na zásobník

Jednoadresové instrukce

Explicitní adresa jednoho operandu

- dvě adresy operandů implicitní
 - ♦ druhý zdrojový operand a umístění výsledku
- obvyklé pro akumulátorovou architekturu

ADD x ... Acc := Acc + x

Dvouadresové instrukce

Explicitní adresa dvou operandů

- jedna z adres je použita jak pro operand, tak pro výsledek
- velmi obvyklé

SUB A,4 ... $A := A - 4$

Tříadresové instrukce

Všechny operandy adresovány explicitně

- zdrojové operandy, umístění výsledku
- se zvyšováním rychlosti pamětí roste flexibilita počítače a dává velké možnosti dobrým kompilátorům
 - ♦ „vyhněte se problémům s udržováním dat v omezené sadě registrů a používejte jako velkou sadu registrů primární paměť.“
- pro větší délku instrukcí a výslednou délku instrukčního slova se nepoužívá často
 - ♦ prakticky použitelné především ve spojení s registry

Příklad zápis výrazu pomocí různých instrukcí

Výpočet hodnoty

$$Y = (A - B) \div (C + D \times E)$$

| 3–adresové | 2–adresové | 1–adresové | 0–adresové |
|------------|------------|------------|------------|
| SUB Y,A,B | MOV Y,A | LOAD D | PUSH A |
| MUL T,D,E | SUB Y,B | MUL E | PUSH B |
| ADD T,T,C | MOV T,D | ADD C | SUB |
| DIV Y,Y,T | MUL T,E | STORE Y | PUSH C |
| | ADD T,C | LOAD A | PUSH D |
| | DIV Y,T | SUB B | PUSH E |
| | | DIV Y | MUL |
| | | STORE Y | ADD |
| | | | DIV |
| | | | POP Y |

Zápisu výrazu

Infixová notace

$$(A-B) \div (C+D \times E)$$

Postfixová notace

$$AB-CDE \times + \div$$

Prefixová notace

$$\div - AB + C \times DE$$

Převody mezi notacemi

- pomocí gramatiky výrazu

Gramatika aritmetického výrazu

EXPR := EXPR '+' TERM
| EXPR '-' TERM
| TERM ;

TERM := TERM '×' FACT
| TERM '÷' FACT
| FACT ;

FACT := '(' EXP ')'
| identifier ;

Reprezentace výrazu

Syntaktický strom

- uzly ~ operace
- listy ~ operandy

Průchod stromem

- in-order

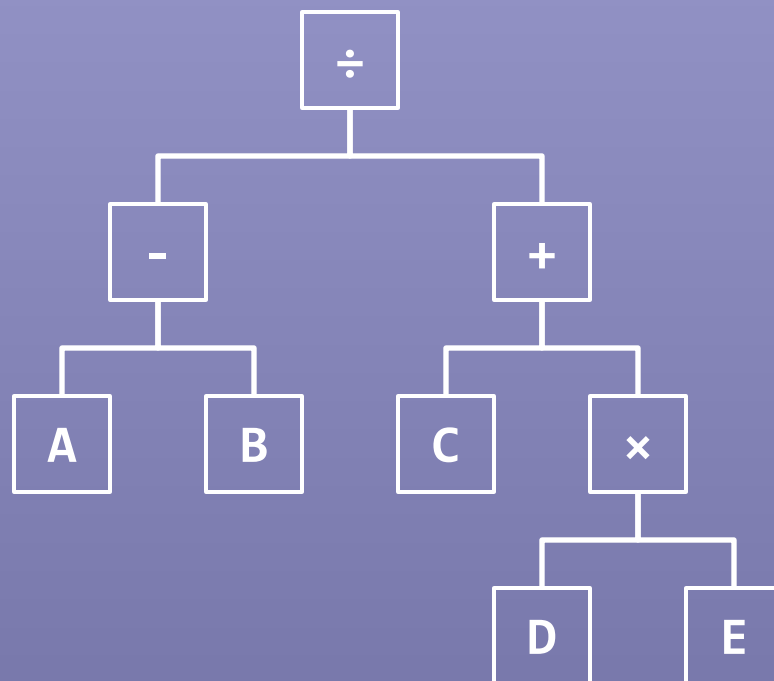
$(A-B) \div (C+D \times E)$

- post-order

$AB-CDE \times + \div$

- pre-order

$\div - AB + C \times DE$



Reverzní polská notace

Zápis výpočtu pomocí zásobníku

- vstup: výraz v RPN
- výstup: číslo na vrcholu zásobníku

Zpracování výrazu

- číslo
 - uložit na zásobník
- operace
 - vyzvednout potřebné operandy
 - provést operaci
 - výsledek uložit na zásobník

Dělení instrukcí (2)

Podle typu operace

- aritmetické operace
- logické operace
- operace pro přesun dat uvnitř počítače
- operace pro vstup/výstup dat do/z počítače
- řídicí operace

Aritmetické instrukce

Základní výpočetní operace

- $+$ $-$ $*$ $/$
- abs, neg, inc, dec

Speciální aritmetické operace

- práce v plovoucí řádové čárce
- logaritmy, odmocniny, ...

Konverze, překlady

Logické instrukce

Booleovské operace

- AND, OR, NOT, XOR

Porovnávání, test

- pouze nastavení příznaků

Posuny a rotace

- pozor na znaménko

Přesuny dat a vstup/výstup

Load/store architektury

- načíst obsah registru z paměti
- uložit obsah registru do paměti
- ALU realizuje operace pouze mezi registry
 - ♦ přesuny mezi registry
 - ♦ operace s obsahem registrů

Jiné operace

- přesuny v paměti, řetězcové operace
- vstup/výstup na periferní zařízení

Řídící instrukce

Skoky

- nepodmíněný, podmíněný
- volání podprogramu, návrat
 - ♦ vstupní parametry
 - ♦ návratová adresa

Řízení

- halt, wait, nop
- cli, sti

Určení argumentů instrukce

Implicitní

- parametry dány použitou instrukcí

Explicitní

- součástí zápisu instrukce je odkaz na parametry
- nutno jasně definovat při návrhu instrukční sady

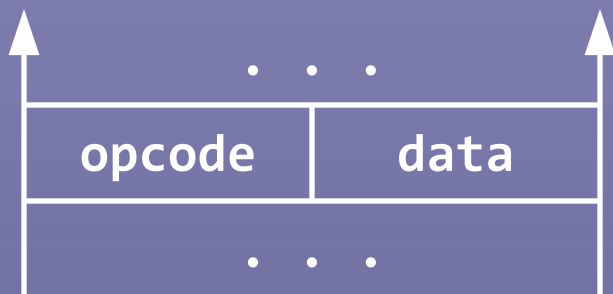
Způsoby adresace

- immediate – bezprostřední zápis v instrukci
- direct – v instrukci zapsána adresa operandu
- indirect – odkaz do paměti, kde je adresa oper.
- indexed – k adrese je přičten index
- based – adresa tvoří posunutí vzhledem k bázi
- relative – vzhledem k adresovému čítači

Immediate Mode

Operand je obsažen v instrukci

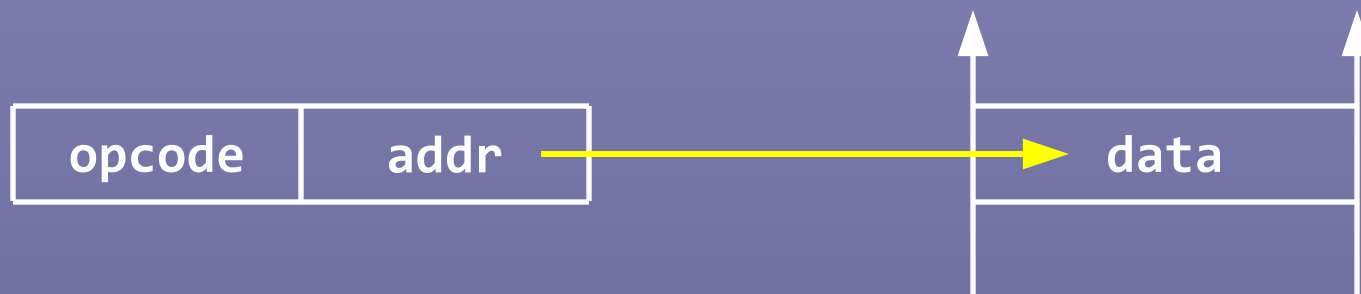
- data jsou za běhu kódu konstantní
- po načtení instrukce není třeba přistupovat do paměti
- velikost operandu je omezená



Direct Mode

V instrukci je zapsána adresa operandu

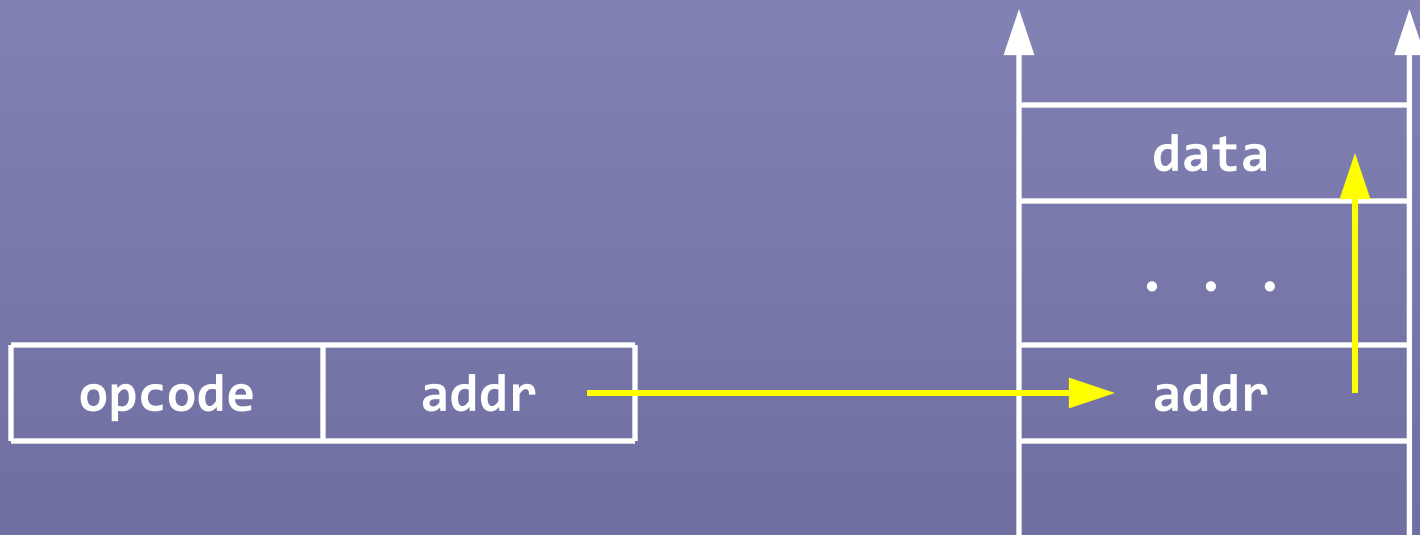
- k vykonání instrukce je třeba navíc 1 přístup k paměti
- rozsah adres limitován velikostí instrukce
- adresa operandu je konstatní, data se mohou měnit



Indirect Mode

V instrukci je adresa s adresou operandu

- k vykonání instrukce jsou třeba 2 přístupy navíc:
 - ♦ načtení adresy operandu
 - ♦ přístup k operandu



Indexed Mode, Base Mode

Adresa složena ze dvou částí

- počáteční/bázová adresa
- posunutí vzhledem k počátku
- velmi podobné, někdy se nerozlišuje

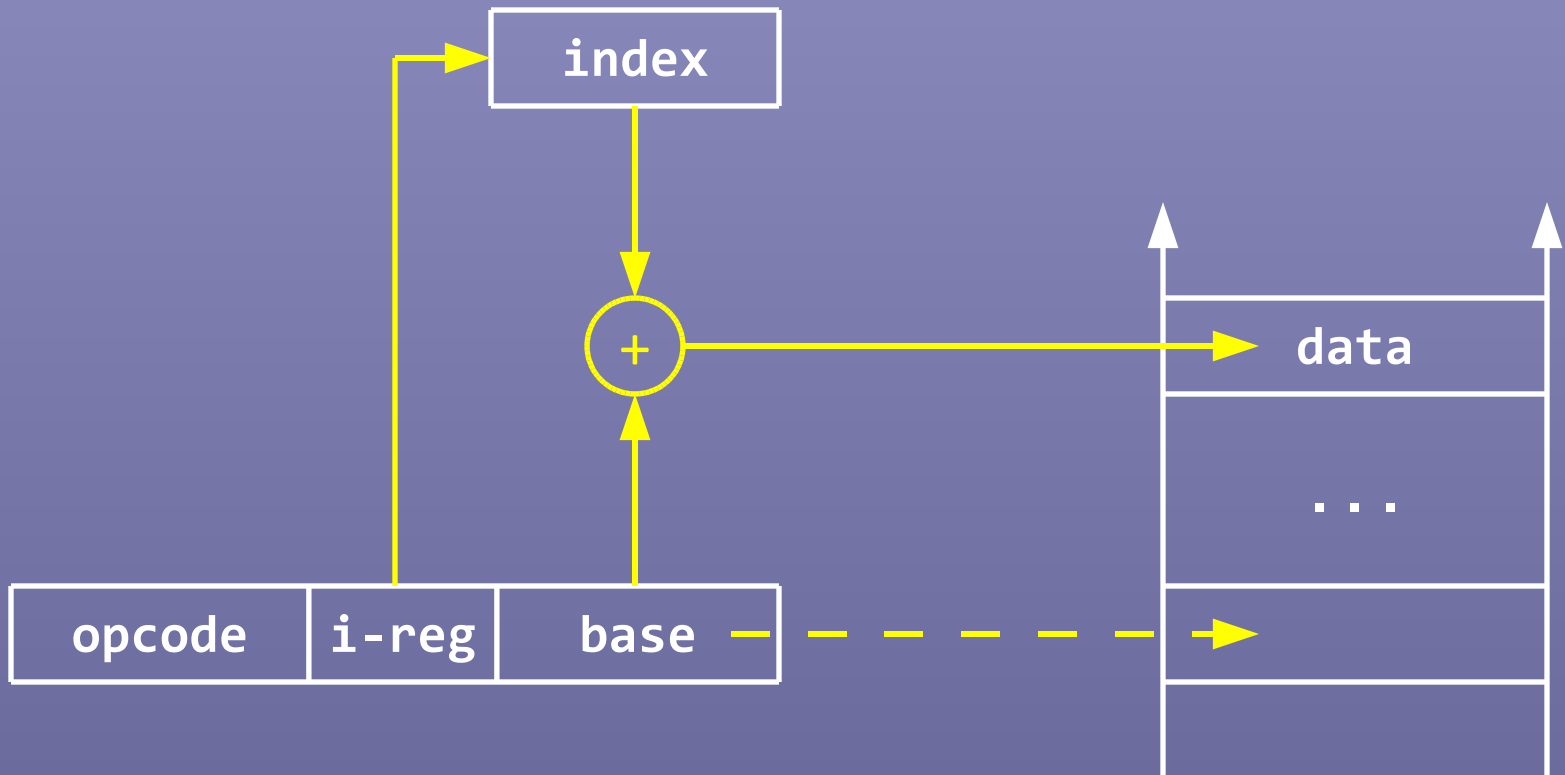
Pokud se rozlišuje...

- způsob realizace a použití
 - ♦ indexed: v programu pro přístup k datům
 - ♦ based: v OS pro implementaci ochrany/segmentace

Indexed Mode

Přístup k datům program

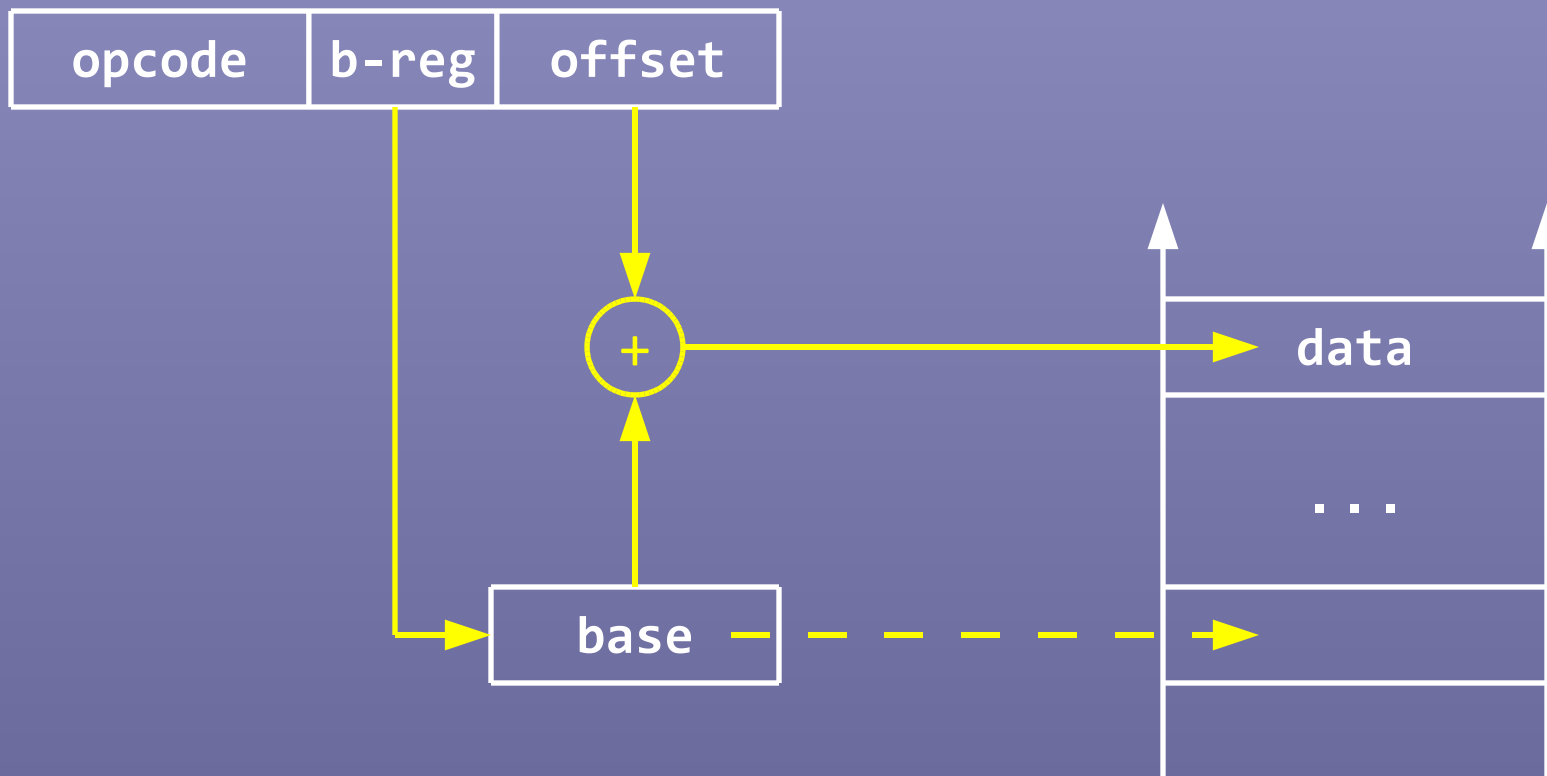
- báze je statická, index/offset se mění



Based Mode

Relokace, ochrana paměti

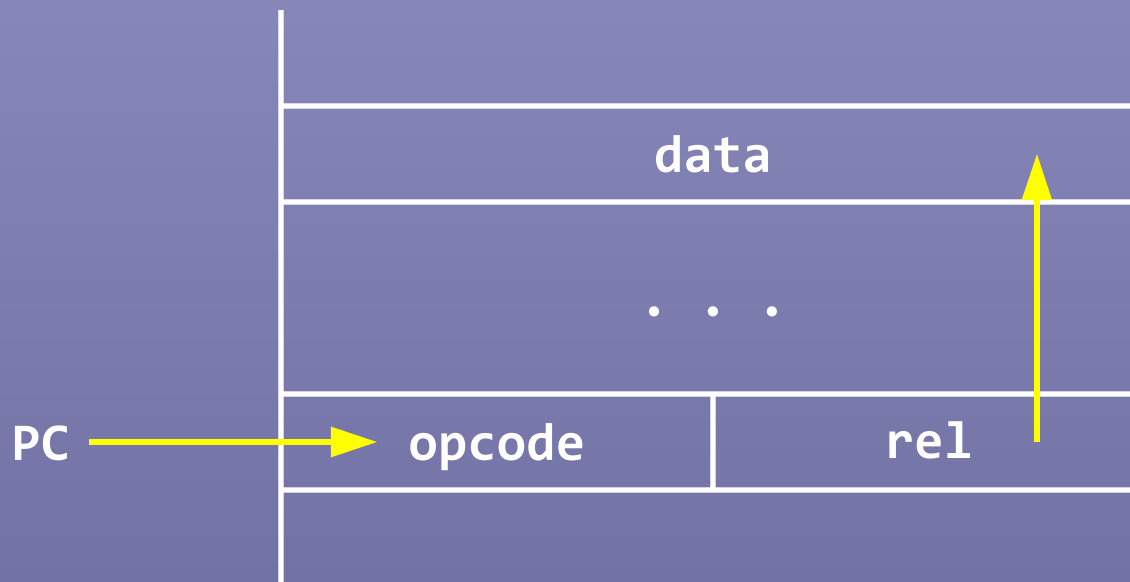
- báze je součást kontextu, offset je statický



Relative Mode

Vzhledem k programovému čítači

- relativní posun



Adresace s použitím registrů

Adresy v instrukcích z domény registrů

- register addressing
 - ♦ jako direct, ale adresová část určuje registr s operandem
- register indirect
 - ♦ jako indirect, ale adresová část určuje registr, který obsahuje adresu operandu

Automatické změny argumentů

Např.

| | |
|---------|--------------|
| 1. pre | a. increment |
| 2. post | b. decrement |

- automatický posun na další zpracovávaná data je součástí logiky instrukce
- typické pro přesuny dat a konverze
 - ♦ Intel IA-32: movs, lods, stos, ins, outs, ...
- typické v kombinaci s nepřímým adresováním

Kombinované režimy

Kombinace různých způsobů

- $\text{base} + \text{index} * \text{scale} + \text{displacement}$
 - ♦ base, index ... registry
 - ♦ scale ... $\{ 1, 2, 4, 8 \}$
 - ♦ displacement ... konstanta v instrukci
- Intel IA-32
 - ♦ $\text{segment} + \text{base} + \text{index} * \text{scale} + \text{displacement}$

Literatura

W. Stallings

- Computer Organization & Architecture

A. Tanenbaum

- Structured Computer Organization

J. Bayer et al.

- Počítače pro řízení

N. J. Davis

- Computer Organisation