Logické instrukce, instrukce posunů a rotací, skokové instrukce

Ing. Dominika Regéciová
Výzkumná skupina formálních modelů

Vysoké učení technické v Brně, Fakulta informačních technologií
Božetěchova 1/2, 612 66 Brno - Královo Pole

iregeciova@fit.vutbr.cz

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Příklad na začátek

- **Převeďte číslo $(2.625)_{10}$ do binární soustavy na 4 desetinná místa**
- **Převeďte číslo $(-50)_{10}$ do binární soustavy na 8 bitů v doplňkovém kódu**

Příklad na začátek

- **Převeďte číslo $(2.625)_{10}$ do binární soustavy na 4 desetinná místa**
  - $(10.1010)_2$
- **Převeďte číslo $(-50)_{10}$ do binární soustavy na 8 bitů v doplňkovém kódu**
  - $(11001110)_2$

Informace k testu

- **Musíte dojít na svoji skupinu cvičení**
- **Můžete získat až 6 bodů**
- **Nejsou povoleny žádné pomocné materiály**
- **Témata:**
  - Řešení jednoduchého matematického problému v asembleru
  - Posuvy, rotace, práce s řetězci, aritmetické operace, . . . tedy dnešní a předchozí cvičení

Logické operace

A	B	AND	OR	XOR	NOT A	NOT B
0	0	0	0	0	1	1
0	1	0	1	1	1	0
1	0	0	1	1	0	1
1	1	1	1	0	0	0

```
AND   DEST, SOURCE    ;DEST = DEST ∧ SOURCE
OR    DEST, SOURCE    ;DEST = DEST ∨ SOURCE
XOR   DEST, SOURCE    ;DEST = DEST ⊕ SOURCE
NOT   DEST            ;DEST = ¬ DEST
```

- Pracují s jednotlivými bity
- Všechny nastavují příznaky v registru EFLAGS (CF, OF, PF, SF, ZF)
- Operandy instrukcí musí mít stejnou velikost

Logické operace AND a TEST

- **AND** op1, op2
  - op1 = op1 AND op2
  - Nastavení určitých bitů na 0
- **TEST** op1, op2
  - op1 AND op2
  - TEST neukládá výsledek operace, pouze nastavuje příznaky
  - Když chci porovnat hodnoty na 0 (rychlejší než `cmp`)
  - Testování nějakého bitu (flagu)

Logické operace OR a XOR

- **OR** op1, op2
  - op1 = op1 OR op2
  - Nastavení určitých bitů na 1
  - Porovnání čísel na nulovou nebo nenulovou hodnotu
- **XOR** op1, op2
  - op1 = op1 XOR op2
  - Rychlé nulování registru (Pozor! Ovlivňují se příznaky v EFLAGS)
  - Kryptografie - šifrování/dešifrování dat

Mapování (J. Husa)

- Logické instrukce lze použít k nastavení konkrétních bitů na konkrétní hodnoty.
- Nastavení nejnižšího bitu `AL` na jedničku.

```
OR    AL, 0b00000001
```

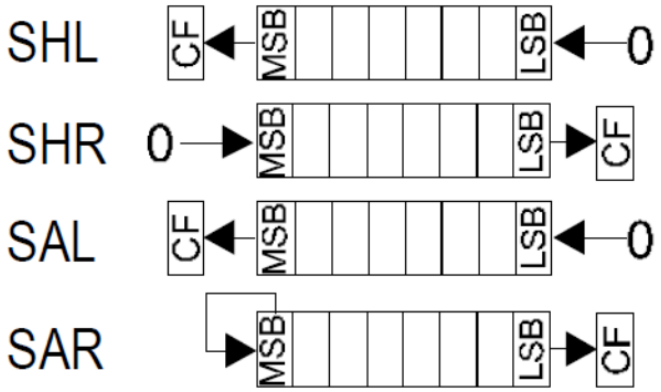
- Nastavení nejnižšího bitu `AL` na nulu.

```
AND   AL, 0b11111110
```

- Úkol:
  - Načtěte ze vstupu řetězec (login).
  - Převeďte malá písmena na velká a vypište.

Posuvy

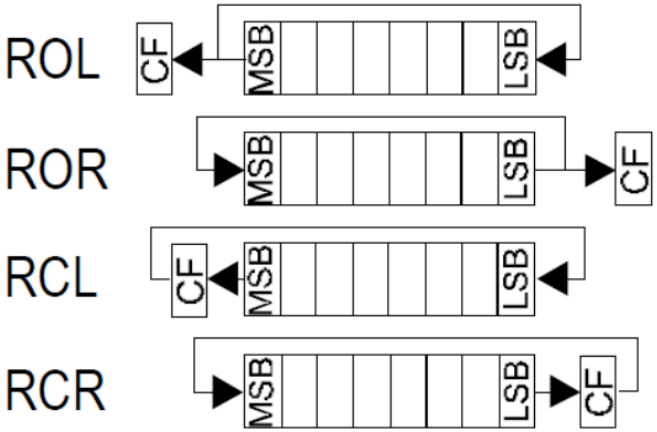
- `SHL DEST, const`
- `SHR DEST, const`
- `SAL DEST, const`
- `SAR DEST, const`



- Posun bitů v registru doleva nebo doprava
- Druhým operandem může být registr `CL`
- Úkol:
  - Vytvořte program, který násobí číslo deseti bez instrukcí `mul` nebo `imul`.

Rotace

- `ROL DEST, const`
- `ROR DEST, const`
- `RCL DEST, const`
- `RCR DEST, const`



- Podobné posuvům, ale bity se do registru vracejí
- Opět lze použít `CL`

Nepodmíněné skoky

- Instrukce nepodmíněného skoku **JMP** `navesti`
- Program bude pokračovat od pozice `navesti`
- Pozor, může způsobit zacyklení

```
%include "rw32-2018.inc"
section .data
section .text
_main:
    mov AL, 42
    jmp navesti
    inc AL
navesti:
    call WriteInt8
    ret
```

Podmíněné skoky

- Skoky jsou prováděny na základě hodnot registru EFLAGS
- Instrukce jsou ve tvaru **Jcc** `navesti` nebo **JNcc** `navesti`, kde **N** značí negaci a **cc** konkrétní podmínku
  - **JE** `cil` *; skoc, pokud jsou hodnoty stejne*
  - **JNE** `cil` *; skoc, pokud nejsou hodnoty stejne*
  - **JA** `cil` *; skoc, pokud je vetsi (bez znamenka)*
  - **JG** `cil` *; skoc, pokud je vetsi (se znamenkem)*
- Úkol:
  - Upravte příklad `5.1.jmp.asm` tak, aby obsahoval podmíněný skok pro případ, že je hodnota registru AL větší než 0.

Dobré vědět . . .

Zajímavý kanál na Youtube, nejen pro ISU:
https://www.youtube.com/user/Computerphile/videos

Doporučuji video: Endianness Explained With an Egg