Řetězové instrukce, konvence volání, externí funkce

ISU cv 9

http://www.fit.vutbr.cz/~isakin/isu

Zásobníkový rámec (Stack frame) - opakování

- Co to je?
- Na co to je?

Zásobníkový rámec (Stack frame) - opakování

- Oblast mezi EBP a ESP
- Vymezení prostoru pro funkci
- Vytváří se při volání podprogramu, kdy jsou **parametry předány přes zásobník**.
- Jednotlivé prvky zásobníku jsou vyhrazeny pro:
  - Parametry podprogramu
  - Lokální proměnné podprogramu
  - Návratová adresa z volajícího podprogramu (uložený EIP)
- Obsahuje minimálně návratovou adresu.

Zásobníkový rámec - opakování

```
fce:
    ???

    ; kód funkce

    ???
    ret
```

Zásobníkový rámec - opakování

```
fce:
    push ebp
    mov ebp, esp
    ; kód funkce
    mov esp, ebp
    pop ebp
    ret
```

Zásobníkový rámec - opakování

```
fce:
    enter 0,0

    ; kód funkce

    leave
    ret
```

Lokální proměnné

Lokální proměnné

- alokujeme po vytvoření nového dna (**EBP**) posunutím vrcholu zásobníku (**ESP**), přistupujeme *“do mínusu”*

```
push ebp      ; zálohuj staré dno
mov ebp, esp  ; vytvoř nové dno
sub esp, 12   ; rezervuj 3 lokální proměnné

mov [ebp - 8], dword 5 ; vložení do lokální proměnné
mov eax, [ebp - 8]     ; vyčtení

mov esp, ebp  ; smazání proměnných
pop ebp       ; obnova starého dna
```

Ukázka

- cv9_A.asm

Řetězové instrukce

Řetězové instrukce

- Pro usnadnění práce s poli (vektory, řetězová data)
- Používají **nepřímé adresování** pomocí registrů **ESI** a **EDI**, které (podle instrukce) automaticky zvedají/snižují
- Velikost položek pole (a posuvu) **ne**určujeme pseudo-instrukcemi (byte/word/dword), ale je **přímo součástí názvu instrukce**, např: MOVS(B|W|D)
- MOVSB, CMPSB, SCASB, STOSB, LODSB
- Příznak **DF** (direction flag) určuje **směr pohybu**

Řetězová instrukce - MOVS

- Přesune znak z jednoho řetězce do druhého (**ESI** → **EDI**)
- Směr posuvu příznakem **DF** (Direction Flag), CLD a STD
- Automaticky inkrementuje **ESI** a **EDI** (při **DF**==0)

```
section .data
    string1 db "Ahoj"
    string2 db "xxxx"
section .text
    mov esi, string1
    mov edi, string2
    movsb  ; Výsledek ve string2: "Axxx"
    movsb  ; Výsledek ve string2: "Ahxx"
```

Řetězové instrukce - LODS a STOS

- LODS - načte symbol z řetězce do AL
- STOS - uloží hodnotu z AL do řetězce

```
mov edi, string2
mov al, 'B'
stosb   ; výsledek string2: "Bxxx"
stosb   ; výsledek string2: "BBxx"

mov esi, string1
lodsb   ; v AL symbol "A"
```

Řetězové instrukce - CMPS a SCAS

- CMPS - Porovná znaky ze dvou řetězců
- SCAS - porovná hodnotu z AL se znakem v řetězci

```
mov esi, string1
mov edi, string2
cmpsb  ; Výsledek: vygenerované příznaky SF, AF, PF, CF

mov al, "A"
mov esi, string1
scasb  ; Výsledek: vygenerované příznaky AF, CF
```

Řetězové instrukce - REP, REPE, REPNE

- Pro opakování řetězové instrukce (místo smyčky)
- `rep` - Dekrementuje **ECX** a opakuje instrukci dokud není 0.
- `repe  ; Repeat while Equal (ZF == 1)`
- `repne ; Repeat while Not Equal (ZF == 0)`

```
mov esi, src ; ESI = zdroj
mov edi, dst ; EDI = cíl
mov ecx, 5   ; ECX = počet
cld          ; DF = 0 (doprava)
; zkopiruj 8b, ECX--, opakuj dokud ECX != 0
rep movsb
```

Ukázka

- cv9_B.asm

Úkol

- cv9_1.asm

Konvence volání

Konvence volání funkcí

- Definuje způsob předání parametrů:
  - **zleva doprava** `fce(1,2,3,4)`
  - **zprava doleva** `fce(4,3,2,1)`
- Určuje kdo ze zásobníku uklidí parametry:
  - **volající**: ten, kdo volá funkci instrukcí CALL
  - **volaný**: ten, kdo je volán = volaná funkce

Konvence volání funkcí

Konvence volání	Parametry funkce	Zásobní k uklízí	Dekorace jmen (pro jazyk C)	Použito v
`pascal`	zleva doprava	volaný	`symbol`	Pascal
`cdecl`	zprava doleva	volající	`_symbol`	Jazyk C
`stdcall`	zprava doleva	volaný	`_symbol@4`	Win32 API
`fastcall`	první dva parametry v ECX a EDX, zbytek zprava doleva	volaný	`@symbol@8`	různé

Uklízení argumentů ze zásobníku

- U konvence, kde argumenty uklízí **volaný** (volaná funkce) je potřeba posunout ESP až po zavolání RET (což nejde), proto lze při volání RET definovat, jak moc se má vrátit:

```
ret     ; pouze skok na return adresu
ret 8   ; skok na return adresu a uklizení args ze zásobníku (2*4)
```

Externí C funkce

Volání externích C funkcí

- funkce implementované mimo náš zdrojový soubor
- Externí funkce deklarujeme direktivou
  extern (_extern, CEXTERN)
- Funkce z “rw32.inc” **nejsou** externí, protože `%include` vloží kód knihovny do našeho kódu
- **Pozor!** externí funkce **nezálohují** hodnoty registrů **EAX**, **EBX**, **ECX** a **EDX**!

Volání externích C funkcí

- C funkce:
  atoi/atol, strlen, strcmp, strstr, qsort, malloc, free

Ukázka

- cv9_C.asm

Úkoly

- cv9_2.asm
- cv9_3.asm

Příští týden TEST 2