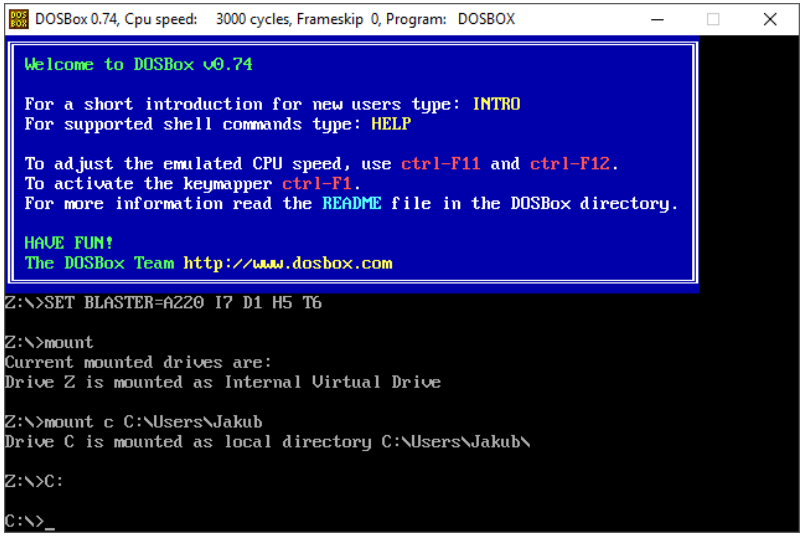
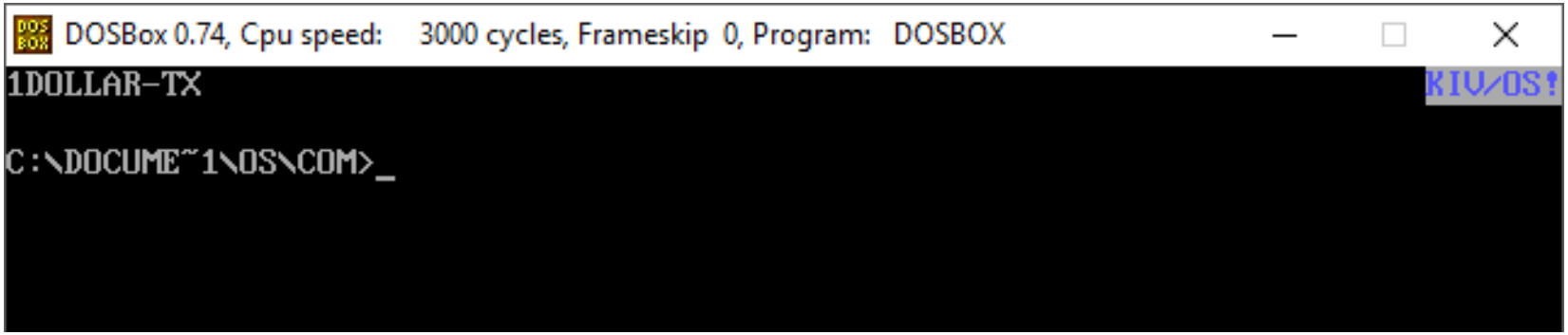
PRŮBĚŽNÁ PREZENTACE PŘEDMĚTU OS

JAKUB VÁVERKA
DAVID BOHMANN
VÁCLAV JANOCH

ZADÁNÍ

Simulace operačního systému DOS



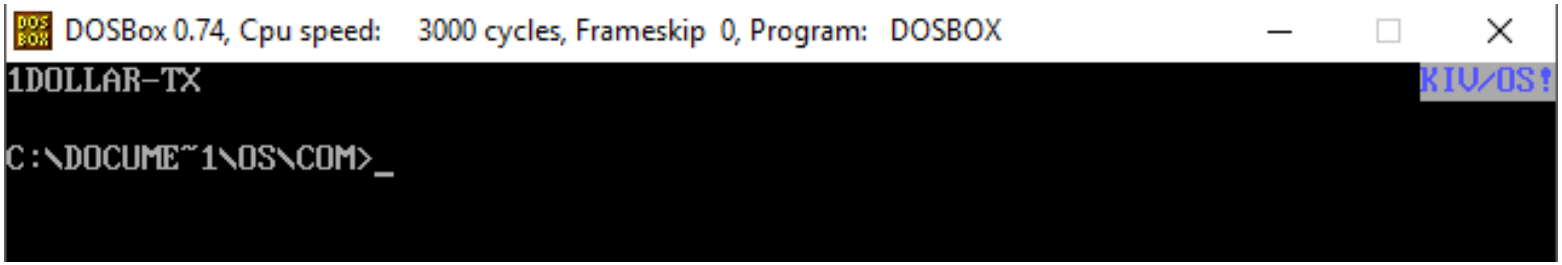


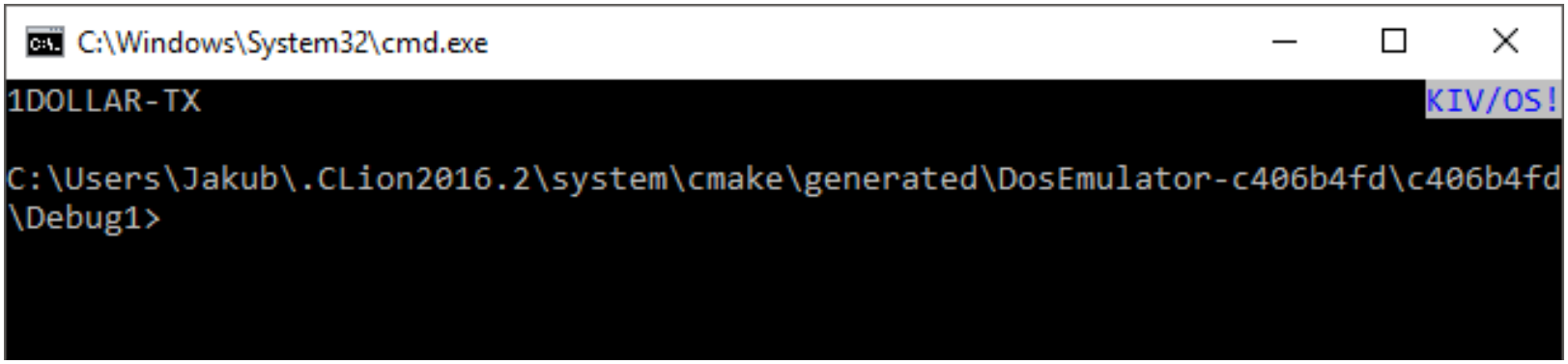
CO TO VLASTNĚ DĚLÁ?

```
40  decrypt:
41
42    mov al, byte ptr ds:[edx]
43    mov al, byte ptr es:[esi+eax]
44    add byte ptr ds:[bx], al
45
46    inc ebx
47    inc edx
48
49    dec cx
50    cmp cx, 0
51    jnz decrypt
```

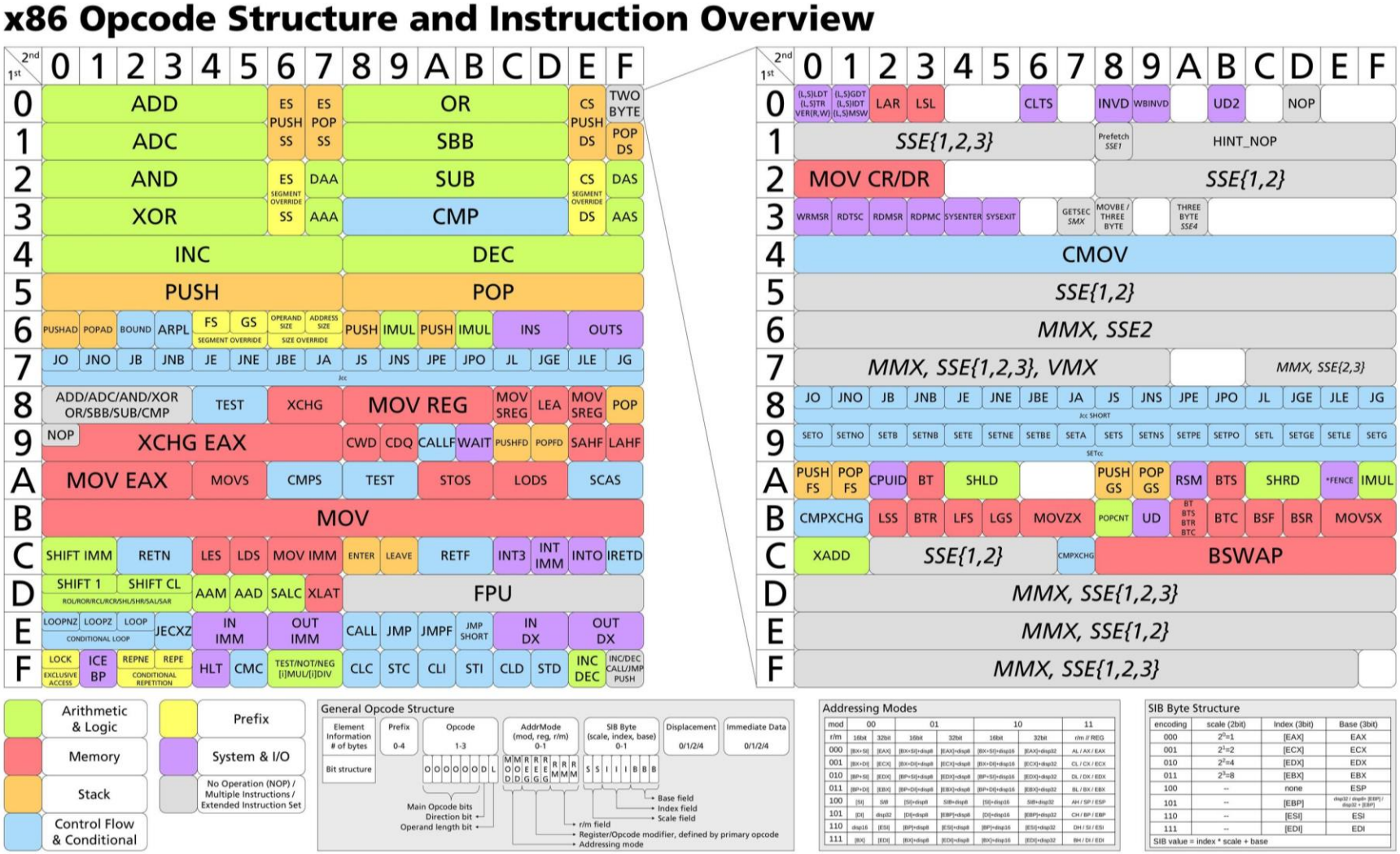

```
77  key1 db 20h, 3ah, 23h, 32h, 0bh, 3dh, 1fh, 13h, 4ch, 19h, 05h, 07h, 07h, 00h
78  key2 db 00h, 0ah, 11h, 08h, 03h, 1dh, 1ah, 08h, 04h, 2ch, 3fh, 33h, 1ah, 41h
79  key3 db 01h, 00h, 08h, 0bh, 0ch, 07h, 0ah, 05h, 02h, 09h, 06h, 03h, 04h, 00h
```

CO MÁME?


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
1DOLLAR-TX
KIV/OS!
C:\DOCUME~1\OS\COM>_


C:\Windows\System32\cmd.exe
1DOLLAR-TX
KIV/OS!
C:\Users\Jakub\.CLion2016.2\system\cmake\generated\DosEmulator-c406b4fd\c406b4fd
\Debug1>

JAK TO ŠLO?



v1.0 – 30.08.2011

Contact: Daniel Plohmann – +49 228 73 54 228 – daniel.plohmann@fkie.fraunhofer.de

Source: Intel x86 Instruction Set Reference

Opcode table presentation inspired by work of Ange Albertini

CO BYLO EASY?

Skok do sebe

```
32  ;nasledujuci kod nastavi eax na 1
33  ; je to jmp -1 nasledovany inc eax, ktery je ted 0
34  db 0ebh, 0ffh, 0c0h
```

CO REKTILO?

Nexorovaný segmentový register

```
16  mov dx,cs
17  mov ds,dx
18  mov es, dx
19
20  ;zacneme s desifrovanim
21
22  xor eax, eax
23  xor ecx, ecx
24  xor ebx, ebx
25  xor edx, edx
26  xor esi, esi
```

Přímý zápis do grafického bufferu

```
63  mov word ptr [di+146], 794Bh  ;K
64  mov word ptr [di+148], 7949h  ;I
65  mov word ptr [di+150], 7956h  ;V
66  mov word ptr [di+152], 792Fh  ;/
67  mov word ptr [di+154], 794Fh  ;O
68  mov word ptr [di+156], 7953h  ;S
69  mov word ptr [di+158], 7921h  ;!
```

DĚKUJEME VÁM ZA POZORNOST

Nějaké otázky?