

# Asemblery

!!!! KOMENT(ARE TU ALEBO FIITKAR) !!!!

**!!! KOMENTARE POMOCOU STYLU SUBTITLE !!!**

rozumna url na tento dokument: <http://is.gd/asm2012>

rozumna **PREVIEW** url: [http://is.gd/asm2012\\_preview](http://is.gd/asm2012_preview)

skuska 2010 [https://www.dropbox.com/sh/ec5tyi8bmjkg1h/YfQzuM9Kv1/Asm\\_skuska2010.png](https://www.dropbox.com/sh/ec5tyi8bmjkg1h/YfQzuM9Kv1/Asm_skuska2010.png)

riesenie tu:

<http://liho.php5.sk/ASM0910L.png>

//vobec sa mi nepaci kolko tam je technickych otazok. Takze zakladna otazka:

**JE NA SKUSKU NEJAKE MINIMUM?? :)**

v dokumente Podmienky absolvovania predmetu a aj na prvej prednaske (ako to mam napisane v zosite) **je uvedene cislo 31b**

/// uzitocna vec....flagy by liho

príznakový register (F) - - - - O D I T S Z - A - P - C (16 bitov, - znamená nevyužitý bit)

**C** = carry = nastaví sa na 1, ak nastal prenos z najvyššieho rádu

**P** = parity = nastaví sa na 1, ak výsledok operácie má párny počet jednotiek

**A** = auxiliary = nastaví sa na 1 ak nastal prenos medzi 3. a 4. rádom

**Z** = zero = nastaví sa na 1, ak výsledok operácie je nula

**S** = sign = znamienko, skopíruje sa z najvyššieho rádu výsledku operácie

**T** = trap = na debugovanie, vieš donútiť procesor zastať po každom kroku

**I** = interrupt = hnusný bit, ktorý keď je nastavený na 0 tak procesor nemôže spraviť prerušenie

**D** = direction = smer pri spracovávaní reťazcov (1 je zľava do prava - od vyšších bitov k nižším)

**O** = overflow = nastaví sa na 1 ak došlo k pretečeniu

prehľad veľkosti:

procesor	fyzicky	virtualny	segment
086	1MB		
286	16MB	1GB	64kB
386/486	4GB	64TB	4GB

**1.) 009A**

**2.) 1111 1010 B**

//filip: nejake vysvetlenie?

//stclaus: operacny kod (v Intellovskej architekture) je jednobajtovy, cize by to malo byt 8-bitov

**3.)**

MOV AX, SEG D1

MOV ES, AX

MOV DS, ES

//ada: preco sa to robi cez es?? ci len kvoli tomu komentaru??

//stclaus: povedal by som, ze hej

//joffo: moc sa mi to nezda, podla definicie mov nemoze mat operandy segReg, segReg <http://>

[www.ousob.com/ng/asm/ng11587.php](http://www.ousob.com/ng/asm/ng11587.php)

#### 4.) NIE

//povodna odp: ANO

// isto? Ved LABEL je z Reserved Words.

<http://www.x-hacker.org/ng/asm/ng43e58.html>

//stclaus: tak v tom pripade podla mna nie

// urcite nie skusal som to, label nemoze byt

//joffo:menim na NIE

#### 5.) 1111011B

6.) programy napisane pre tento procesor pojdu rovnako spustiť aj na vyšších procesoroch

#### 7.) OR BH, 11111111B

#### 8.) XOR AX, AX

9.) double words =   32 bitov  
                          4 bajty  
                          2 slova

//rackom: nech sa vyjadri nejaký kompetentný človek, či je to správne lebo to čo je napísané v tej skúske veľmi nesedí s týmto:

[http://en.wikipedia.org/wiki/Word\\_\(computer\\_architecture\)](http://en.wikipedia.org/wiki/Word_(computer_architecture))

(tabuľka dolu)

//matus: no záleží čo tým myslíš, v prednáške má základné typy údajov ako double word čo je 32 bitov, 4 bajty a 2 slova, tu sa ale pýta na double word ako na odvodený typ čo je veľmi matúce

//ada: asi to bude to isté: D:D ... a vidíš aj prednášky si môžeme kukat

//stclaus: nevidím dôvod, prečo by to malo byť inak ako 32b, neviem prečo tam dával množné číslo...

//mnicky: IMO je WORD podľa architektúry, čiže ak je 16bitová, tak word == 16b => double word = 32b

//cstx: <http://www.ousob.com/ng/masm/ng486c3.php>

4B sú správne

#### 10.) XOR CL,CL

#### 11.) 0D

//ada: tá 0 lebo má byť dvojciferný výsledok

#### 12.) indexovy

#### 13.) nepriamy

//povodna odpoveď: **indexovobazovy** je zle (vid. diskusia)

//matus no neviem neviem, minimálne indexové keďže register [SI]

//nirhu podľa [http://en.wikipedia.org/wiki/MOV\\_\(x86\\_instruction\)](http://en.wikipedia.org/wiki/MOV_(x86_instruction)) memory to memory are NOT possible, aj TASM hadze "Illegal memory reference", čiže podľa mňa indexovy

//mnicky: podľa NG: <http://www.ousob.com/ng/masm/ng2d6e3.php> - takže ja by som povedal že ide o 'register indirect', čiže IMO **nepriamy** - čo vy na to?

//joffo:indexovobazovy má to v prednáškach

//stclaus: ja by som asi povedal nepriamy, ale nie som si istý

//mne sa ten indexovo-bazovy nejak nezdá, malo by ísť o kombináciu indexového a bazového v rámci jedného operandu, tu to tak nieje, dal by som nepriame

//tak to zmeníme?

//dury: ja by som povedal tiež nepriamy, keby to bol indoxovo-bázovy, muselo by to byť v jednom operande  
 //joffo:nj, zrejme to teda bude ten nepriamy  
 //rootpd: na zaklade diskusie z minulorocneho fiitkara to je nepriame (tretia strana posledny post), joffo ty si pisal ze to je v prednaskach, mozes povedat kde konkretne?  
 //joffo:no ono ked som to detailne prezrel v poznatkach tak ono to bolo tak pisali ludia vyssie, teda v jednych zatvorkach

#### 14.) 1

//matus je to hodnota bitu najvysieho radu vysledku a to je 1  
 //filip: neznamena S sign, co je znamienko + alebo - ? A vyjde kladne cislo, takže 0. ci?  
 //bally: aj podla mna 0, nech skusam co skusam 0  
 //mnicky: kedze je to vlastne 243 - 114, tak vysledok je kladny, cize znamienkovy bit nie je nastaveny, cize 0 :)  
 //ale pozor, to su 8 bitove registre, vysledok je 8 bitovy, najvyssi bit je 1 tak aj S (sign) bude 1, skusal som to v debuggeri: ked som dal operandy DL a BH, tak **S-flag sa nastavil na 1** (pri DX a BX to bolo 0)  
 //stclaus: takže dalsi chytak?  
 //rootpd: vyzerá to tak, kazdopadne S nezaujima znamienko ale najvyssi bit a ten je v tomto pripade 1

#### 15.) 0

16.) RET      X      X

#### 17.)

```
VSTUP          PROC
MOV AH,01H
INT 21H
RET
VSTUP          ENDP
```

//mnicky: a nemalo by tam skorej byť daco ako: "MOV AH, 01H" na druhom riadku? (je to to iste, ale co ak to bude kontrolovať automaticky?)  
 // mov ax,01h alebo mov ah,01h?  
 //ja by som dal radšej AH (v príručke je aj tak napísané On entry: AH; Returns: AL)  
 //mov ah,01h podľa mna...ma to byť instrukcia, tá musí byť v ah...mov ax,01h by znamenalo, že až po presiahnutí 8 bitov by použilo ah, inak by stacil al...moj názor  
 //dury: samozrejme tam má byť AH... funkcia INT 21H kontroluje hodnotu, ktorá je uložená v AH registry:  
[http://en.wikipedia.org/wiki/INT\\_21H](http://en.wikipedia.org/wiki/INT_21H)  
 //joffo:menim na AH

#### 18.) -119 + 128 = 9 = 1000 1001

//mnicky: IMO je logickejsi postup:  
 1. previesť kladne cislo (cize 119) do dvojkovej - dostaneme 0111 0111  
 2. invertovať bity - dostaneme 1000 1000  
 3. pripočítať doplňok, cize jednotku k najnižšiemu radu - dostaneme **1000 1001**  
 // simon: konečne to mi to niekto vysvetlil ;- ) super postup mnicky  
 //dury: mnickyho postup je správny

**19.)**  $0111\ 0000 = 16+32+64 = 112$

// prvý bit je 0, takže sa žiadne zmeny nekonajú

// čo by sa dialo keby bol 1 prvý bit, potom sa odpocita 128, nie?

// podľa postupu z príkladu 18:

$0111\ 0000 - 1 = 0110\ 1111$

invert:  $1001\ 0000$  -prvá nemôže byť jednotka!! takže beriem pôvodne  $0111\ 0000 = 112$

// ak by bolo  $0001\ 0000 = -16$

**20.)** 0

**21.)**  $129H = 0001\ 0010\ 1001$

**22.)**  $123 = 001\ 010\ 011 = 0101\ 0011 = 83$

//mnicky: nie je ľahšie spočítať  $1 * 8^2 + 2 * 8^1 + 3 * 8^0 = 83 ? :-)$

//simon:  $3 + 5*16 = 83$ , tiež celkom easy

//dury: simon to si sa ako k tomu dostal?

**23.)**

1111 1011

1011

//doplňkový kód = odpočíta sa 128 keď máte najvyšší bit jednotkový

//napr.  $1000\ 0001 = 1 - 128 = -127$

//mnicky: no ja som videl doplňkový kód definovaný inak (vid. 18.), a podľa tejto (všeobecnejšej) definície, platnej pre ľubovoľný počet bitov je správna odpoveď navyše **aj 1011**

//stclaus: v otázke je stvorbitový, alebo osembitový, takže asi aj mnickyho odpoveď

**24.)**

//podľa mňa 2. možnosť, aj 3. je logická ale robí sa to na čo najlepšie posunutie doprava nie?

//Matho: [08/09 C](#) riešenia - 52. otázka => 2 a 3 súčasne

**25.)**

CMP A, B

JA ULOZ

MOV C, B

JMP VON

ULOV: MOV C, A

VON:

//filip: netuším čo robí JA, takže to niekto potvrdte

//JA je jump if above

//mnicky: je to OK

**26.)**

MOV CL, AH

MOV AH, AL

MOV AL, CL

**27.)** XOR AH,BH

**28.)**

- ak nebol stlačený Enter, skonči. Inak vynuluj register AL a skonči.

- ak bol stlačený Enter, vynuluj register AL a skonči. Inak skonči.

//Toto druhé je určite dobre?? nemalo by to byť len, keď bol stlačený enter, tak koniec? bez nulovania AL  
//je tam JNZ, takže ak nie je podmienka splnená (nebol stlačený enter), skoci na KON a skonci. Ak bol stlačený enter, zbežne XOR AL, AL a potom KON.

## 29.) ASCII kod stlačenej klavesy

### 30.) alokuje pole o veľkosti 256 bajtov

//a naplni ho blbostami, popripade tam necha tie, ktoré tam boli predtým

//Matho: Nenaplna blbostami, ten otaznik znamena ze neinicializuje pamat (ako malloc v C)

### 31.)

[J+2]

[K+2]

//mnicky: ake je racio skryte za tymto?

//stclaus: tomuto ani ja nerozumiem

//vysvetlenie by Lih0:

dajme tomu, že máš 4B pamäť takúto:

0 3AH

1 2BH

2 17H

3 3EH

... v tom prvom presunul všetky 4 bajty naraz,

v tom druhom musel najskôr

MOV AX, [J] (16 bitov = 3A2BH)

to si hodil aj s prvými dvoma bajtami Kčka do lčka

MOV AX, [J+2] (16 bitov = 173EH)

k tomu hodil K+2 cez ADC ale aby mu v carry flagu

ostala 1tka ak by náhodou...

### 32.) ANO EQU 'ANO'

//predtým bolo: ANO DB 'ANO\$'

//ada: asi lepsie ako v lihovom priklade...

//rackom: aj ja si myslim, len by to mohol este niekto potvrdiť

//filip: ste si istý aj tým \$? ved ten by teoreticky mohol byť v ďalšom riadku a s konstantou by nemal nič

//nirhu: nemalo by to byť ANO EQU 'ANO\$' keďže to má byť konštanta?

//filip: asi ano. A teraz s \$ alebo bez?

//mnicky: to je otázka. ja by som dal bez, lebo podľa <http://www.ousob.com/ng/masm/ng4cb7d.php> sa

stringy píše **bez dolara**. Dolar je iba konvencia používaná niektorými DOS funkciami, nie preprocesorom.

Upravil som aj odpoveď.

### 33.) X DB 04H DUP(9)

### 34.) 0000 0001

### 35.) 0000 1001

### 36.) 2343 = 0010 0011 0100 0011

### 37.)

XOR AX

XOR BX

XOR CX

//ada: mozno?? moze byt xor s jednym parametrom??

//filip: podla mna iba dva moze mat

//ada: tak co tam potom ma byt?? makro by malo ten parameter reg nahradit s AX, BX, CX

//filip: netusim

//mnicky: IMO je rec o: <http://www.ousob.com/ng/masm/ng537d1.php> a nasiel som este podobny priklad na [http://www.shsu.edu/~csc\\_tjm/fall2000/cs272/ch8.html](http://www.shsu.edu/~csc_tjm/fall2000/cs272/ch8.html). Podla vsetkeho je to tak, ako to mame. Jediný problém je, že XOR musí mať dva parametre :D

//ja: pytaju sa na vysledok po preklade, nie ci to bude funkčne

### 38.) 5, 7

//ada: mozem vymazať aj všetko okrem 1 a 6?:D:D

//mnicky: ano, mozes. bude to funkčne makro :D

### 39.)

INIT MACRO DSNAME

PUSH AX

MOV AX, **SEG** DSNAME

MOV DS, AX

MOV ES, **AX**

ENDM

//mnicky: este mu tam jaksi chyba riadok na POP AX .... :DD

//stclaus: na to tam nebol priestor :P

//xsivakp: nemoze byt v tom predposlednom riadku aj MOV ES, AX?

//hmmm nemalo by tam byt mov ax, offset dsname?

//dury: offset by tam nemal byť, lebo premenná predstavuje segment, s tým MOV ES,AX ti neviem poradiť teoreticky by to malo ísť

//Matho: nie offset ale SEG “..odovzdanou cez formálny parameter DSNAME, ktorý bude predstavovať ADRESU niektorého segmentu ....” A to MOV ES,DS tam nemože byť, instrukcia MOV nepozná operandy seg-reg, seg-reg, takže tam musí byť MOV ES,AX alebo MOV ES,SEG DSNAME

### 40.) MACRO ENDM X X X

//mnicky: nema to skor byt: **MACRO ENDM X X X** ??

//predchadzajuca odpoved: MENO\_MAKRA MACRO X X X

### 41.) 40H \* 4H = 100H, preto 100H - 103H

//kazda instrukcia (ze vraj) ma velkost 4 (neviem co) takže das 40H\*4 = 100H a kedze mas velkost 4 tak ide to od 100H-103H (100H,101H,102H,103H)

//stclaus: tie preruseniasu totiz ulozené v tabulke, ktorá ma 4 stĺpce. Každé prerušenie ma svoj vlastný riadok. V tom riadku su adresy CS:IP, cize adresa instrukcie, ktoru ma vykonat po prerusení. Preto 40h \* 4

//mnicky: Uz len to mi nie je jasne, ako moze byt 40h \* 4h rovne 100h ?

//stclaus: pretoze prve prerusenie 00H zacina adresou 0h az po 03h, prerusenie 01h zacina od 04h, prerusenie 02 zacina 08h

//stclaus: 40h \* 4 = sestnast 0h a sestnast je 10h:P

//mnicky: ok, diky ;)

### 42.)

Bazova adresa segmentu = DS = 200h  
Efektivna adresa = displacement = posunutie v ramci segmentu = 1234h  
Fyzicka adresa =  $16 * DS + EA = 16 * 200h + 1234h = 2000h + 1234h = \underline{3234h}$   
//filip: preco  $16 * ??$   
//stclaus: pretoze registre v i8086 su 16-bitove, ale adresova zbernica ma 20-bitov. Fyzicka adresa sa teda rata ako:  
FA =  $16 * \text{adresa segmentu} + \text{relativna adresa}$   
tento vzorec sa obmieval na kazdej prednaske, takze by to malo tak byt:P

#### 43.)

$A00000h + 1000h = A01000H$ , cize cely segment lezi na  $A00000h - A01000h$

#### 44.) 20FFH

ADD AX, DX obsah registra DX nemeni // meni len AX

#### 45.) vyzerá ze CD AB

//mnicky: vid. aj

#### 46.) Lokalne

//z fiitkara:

Selektor je veľký 16bitov  
prvých 13 udáva index do tabuľky deskriptorov  
potom nasleduje 1 bit.. ak je 0 odkazuje na globalnu tabuľku deskriptorov GDT a ak 1 tak na lokalnu LDT  
posledne dva bity reprezentuju Requestor's Privilege Level a teda RPL to môže mať hodnoty od 0 po 3  
pri čom 0 znamená žiadanie najvyšších privilégii.

[http://en.wikipedia.org/wiki/X86\\_memory\\_segmentation](http://en.wikipedia.org/wiki/X86_memory_segmentation)

// Spread: potvrdzujem <http://files.gamepub.sk/Bakalar/ASM/SkuskaASP2009.pdf> otázka 8

#### 47.) Globalnej

//filip: podľa príkladu 46. to bude globalna (14 bit je 0). Myslím ale že tam bude aj niečo z premenných, prerušení a instrukcií. Predsa len sa da vybrať viac odpovedí. Ci? :)

#### 48.) 280000H - 290000H

// G=0 => limit je v Bajtoch

//G=1 => limit je vyjadrený v 4KBoch ale neviem čo s tým ďalej

// matussvk: som síce amatér, no ak G = 1 znamená, že limit je v násobkoch 4K blokov, tak vide možnosť  
2: 280000H - 290000H, pretože  $10H * 4K = 10H * 4096 = 10000H$  a  $280000H + 10000H = 290000H$

//tak je to super...mame odpoved...dakujem

#### 49.) 02502H

//mnicky: vie niekto kde sa tam nabera tá nula? Ved  $DS + SI + 2 = 200H + 500H + 2H \neq 2502H$

//stclaus: vid príklad 42:

DS = 200 h

SI = 500 h

posunutie indexu = 2h => efektivna adresa =  $500h + 2h = 502h$

Fyzicka adresa =  $16 * \text{bazova adresa segmentu} + \text{efektivna adresa}$

FA =  $16 * 200h + 502h = 2000h + 502h = \underline{2502h}$

// vobec neberieme uvahu BX????

#### 50.) RCR DX,1

//mnicky: <http://www.ousob.com/ng/masm/ng1fd19.php>

### 51.) 020h

//stclaus: adresa je prvych 13 bitov zo selektora, cize posledne tri bity (ta 3), nepatria do adresy, takže 100h. Asi:PP - ZLE!!

//no ked posledne tri nepatria do adresy, preco ich uvazujes ako nuly? ak nepatria tak ich daj uplne prec a ostane ti 0020H takže podľa mňa to je spravne

//mnicky: presne, aj ja si myslim ze by to malo byt **20H**

//stclaus: ano, uz som si to aj ja uvedomil. 103h je 000100000011, prvych 13 bitov je 00010 0000 co je 20h

### 52.) DS = 1000h EAX = 3000h

$$FA = 16 * 1000h + 3000h = 10000h + 3000h = \underline{\underline{13000h}}$$

urcite  $16 * ?$  nesuvisi to stym : 80386

to je pravda, tam mozu byt ine velkosti tych zbernic..

•To form a physical memory address, appropriate segment registers contents (16-bits) are shifted left by four positions and then added to the 16-bit offset address formed using one - to ano, ale ci aj v 386

zdroj :<http://dopice.sk/2AR> strana 10 ..cca :D

//stclaus: tak potom je to ok, a plati to aj tu

//mnicky: hej, niekde na Wikipedii som cital, ze kvoli kompatibilite ponechal Intel 16b aj v novsich architekturah....

### 53.) DI = destination index = index cieľového reťazca

### 54.) 64TB

The 80386 CPU supports 16K number of segments and thus the total virtual space of 4Gbytes \* 16K = 64 Terrabytes. <http://dopice.sk/2AR> : Summary of 80386

### 55.) DW 0ABCDh

//ada: vie niekto na isto??

//stclaus: povedal by som DW ABCDh

DB by to nemalo byt, kedze konstanta ma 2B a DB alokuje len jeden. Ci sa mylim?

//DW 0ABCDH, konstanta zacina cislom.

//stclaus::) dobre vediet

//mnicky: chytaak :D



skuska08-09 <https://www.dropbox.com/s/m4ilktzexur2alx/ASPSkuska0809RT-v1.png>

nejasne priklady:

//mnicky: to znamena ze sa sem pisu akoze iba tie, ktore nevieme naisto vyratat?

//ada: ked nieco nevies napis, my sme take "jednoduché" vynechali

**14.) 11000001 01000000 00000000 00000000**

//gondy: <http://www.h-schmidt.net/FloatConverter/IEEE754.html>

**15.) +12**

//gondy: to iste cislo ako 14, len signed bit je nulovy => kladne cislo

**16.) 125**

//125 normalne dekadicky to je

**17.) AH**

//mnicky: to si akoze fakt mam pamatat kam ktora funkcia vracia?

//jj :D

//mnicky: a kde mam potom zarucene, ze mi tam neda SAHF alebo SMSW ? :DD

//LOL... :D

//Logic: AL . AX / source ; Source is byte

AH . remainder

or

AX . DX:AX / source ; Source is word

DX . remainder

**18.) DX**

**20.) 2400H**

**21.) 34 12**

**22.) ESP,EIP**

//ada: vie niekto naisto??

//stclaus: ano, ostatne urcite nemeni. CS urcite nie, kedze je to typu NEAR, to je jediny, co by este mohol menit

//mnicky: co je EIP? Preto tam nema IP? (asi som nebol na prednaskach :D)

//<http://www.eecg.toronto.edu/~amza/www.mindsec.com/files/x86regs.html>

//to su registre v 32-bitovych systemoch

**23.) 124A0H**

//podla mna 124A0H = 1200H \* 10H + 150H + 250H + 100H

//stclaus: Pozor pri scitavani 150h + 250h nie je 400h!

**24.) MOV CL, BL**

## 25.) BP, SP

//stclaus: O tom by sa podla mna dalo polemizovat. Na vrchole SP je totiz hodnota IP, ktoru nesmieme porusit. Ked sa chceme dostat ku parametrom, musime ku stacku pristupovat cez BP, aby sme neporusili jeho strukturu. A teda spravime MOV BP, SP, aby nam BP ukazovalo na vrchol stacku a potom si posunieme tak, aby ukazoval na ten parameter, ktory chceme. Cize spravna odpoved by mala byt BP s tym, ze aj SP sa tam scasti pouziva, tak neviem... Ak ma byt len 1 spravna, tak urcite **BP**  
//mnicky: hm, ale filozoficky ponate, samotne BP ti na to nestaci, klucovym je proste SP, nie?  
//stclaus: toto by sa podla mna dalo vyhadat, nech oznacis cokolvek z tych dvoch  
//filip: ale ved odpovedi moze byt aj viac. Takze BP aj SP.

## 26.) 25000H

//preco?  
//stclaus: CS: 1A00H, IP: B000H Fyzicka adresa:  $16 * CS + IP = 1A00H * 10H + B000H = 1A000H + B000H = 1(A+B)000H = \mathbf{25000H}$   
 $Ah+Bh = 10+11 = 21 = 15H$

## 27.) 21100H

//povodne: 11100H je nespravne, vid nizsie  
//Kudlohlapec: podla mna by to malo byt 21100H lebo BP by mal standardne brat zo STACKU, je to tak aj tu co vypracoval Liho minuly rok: <http://liho.php5.sk/ASM0809bL.png> otazka 18.  
//stclaus: Musim suhlasit s Kudlohlavcem. V prednaske o indexovani mam napisane pri bazovom indexovani: AK je pouzite BP, tak adresa segmentu je SS, ak BX, tak je to DS. Vypocet Fyz,adresy:  $16 * \text{bazova adresa segmentu (u nas hodnota SS)} + \text{efektivna adresa (BP + DI)}$ .

## 28.) 64KB

//ada: mozno....  
//ak to niekto viete, pls napiste aj preco  
//mnicky: lebo ked si das vyhladat <http://is.gd/1n6rug>, tak 64KB sa tam casto vyskytuje :) **+2**  
//stclaus: zakerna otazka! Tabulka moze mat max  $2^{13}$  (8k) deskriptorov, kedze selektor obsahuje 13-bit adresu. Avsak kazdy deskriptor ma 8B, cize max velkost je **64kB**, ako je spominane vyssie. Uz som ale chcel pisat odpoved 8k....

## 29.) CS, EIP(IP), ESP(SP)

//ada: bez sance...D:D  
//ak teda instrukcia CALL typu FAR je to iste ako long jump :D  
//stclaus: zabudli ste na SP. Kedze sa pridava na stack, meni sa aj SP, tu konkretne sa posuva o -4.  
//joffo:mozno nekomu pomoze pochopit <http://www.ousob.com/ng/asm/ng17284.php>

## 30.) 20FFH

## 31.) RCR DX,1

//ada: mozno srnky tusia....  
//asi RCR DX,1  
//je to RCR (= rotate carry right, to napovie:P)  
//mnicky: prip. aj <http://www.ousob.com/ng/masm/ng1fd19.php> :)

## 32.) A1000H

## 33.) 4GB

## 34.) vo funkcii F(), tesne pred volanim G()

### 35.) 64KB

//povodne bolo 4kb

//rackom: mozno

//nebude to **64KB**?? lebo to je max velkost segmentu len nvm ci aj vo virtualnom rezime.

//nebude to 1MB?to nie...ale link je ok...

<http://www.scribd.com/doc/28346119/80286-80386-80486-and-Pentium-Microprocessor>  
strana 3

//mnicky: tak aky je teda zaver? Kolko?

//asi to bude 64KB --->[http://bitsavers.org/pdf/amd/\\_dataSheets/80286\\_Nov85.pdf](http://bitsavers.org/pdf/amd/_dataSheets/80286_Nov85.pdf) : strana c.: 9  
ale ci to plati aj pre VM...neviem

//mnicky: "80286 was a 16-bit microprocessor. Although in protected mode the CPU could address up to 16 MB of memory, this was implemented using memory segments. Maximum size of memory segment was still 64 KB." <http://www.cpu-world.com/CPUs/80286/index.html>

- bude to tych 64kb podla mna, lebo velkost segmentu musi byt rovnaka aj pri fyzickej pamati aj pri virtualnej, pokiaľ si z OSiek pamatas ako to funguje :)

### 38.) 11100h

BP = 1000h, SS: 2000h, BP: 1000h, DI: 100h, SI: 200h

MOV AL, DS:[BP+DI]

//stclaus: co je MOV AL, DS:[BP+DI]? Znamena to, ze sa pozrera namiesto do SS do DS?

<http://www.ousob.com/ng/asm/ng31473.php>

: Segment-Override Operator

segmentRegister:expression

Depending on the instruction and operand types, the effective address of an operand is computed relative to ES, DS, and SS. By using this operator, these defaults can be overridden.

//povodne by sa teda pozeral do SS (kedze je tam BP), ale tak sa pozera do DS

Fyzicka adresa = 16 \* **DS** + BP + DI = 10h \* 1000h + 1000h + 100h = **11100h**

### 39.) AB CD

//mnicky: ja by som dal **CD AB**

// 0ABCDH Big Endian => AB CD

// 0ABCDH Little Endian => CD AB

//mnicky: okej, tak AB CD :)

### 40.) MOV EDX, EBX

### 41.) zasobnik

//mnicky: a to je asi vsetko, nie?

//+ESP ? <http://www.unixwiz.net/techtips/win32-callconv-asm.html>

//mnicky: ja by som ale povedal, ze ta (navratova) adresa sa nachadza na zasobniku. na ESP je iba adresa najvrchnejšej položky zasobníka, ale nie priamo navratova adresa....

//chapem co myslis ..asi ...keby otazka bola : ...presne po volani... tak by si tam dal aj ESP, ci?...hmm asi si to nemyslel takto... hejhej...uz viem...lebo ESP je index ... a pod indexom je adresa...

//mnicky: nie, nedal ani vtedy. nejde ani tak o to, ze sa na zasobnik mozu potom ukladat aj dalsie veci, ako skorej o to, ze tie veci su ulozene na \_zasobniku\_ a nie v ESP. ESP iba sprostredkuva pristup k zasobniku, to je vsetko....

### 43.) ulozenie navratovej adresy

//mozno aj : poskytuje priestor pre dočasne ukladanie dat ; ale kedze otazka je : hlavne funkcie zas. PRI VOLANI PODPROGRAMU...tak neviem

//mnicky: ja by som tam asi dal aj na dočasne ukladanie dat (akoze aby sa mi neprepisali registre), ved

to je vlastne to iste ako ulozenie si obsahu IP na zasobnik, akurat obsahom su tu moje data  
//jj je to logicke, napis tam  
//mnicky: pockajme si radsej co povedia dalsi, nie som si uplne isty, je to vsetko velmi individualne. Som zvedavy aj to, ako to bude posudzovat na skuske, lebo tu by sa dalo pri mnohych veciach hadat, pri viacerých otazkach :)  
//stclaus: ja by som dal aj ODOVZDAVANIE PARAMETROV. Ci?  
//dal by som aj alokáciu lokálnych premenných  
// ale ved ide o **volanie** podprogramu - ja myslim ze pri **volani** sa iba ulozi navratova adresa

#### 44.) CR0 bit PG

//nie CRO bit PG ?  
//mnicky: ano, ma to byt **PG**, kedze tu ide o **strankovanie** (predtym bolo PE) a PE je virtualny rezim...  
// CR = control register, PG = paging, PE = Protected Mode Enable

#### 46.) 21000H

//stclaus: Fyzicka adresa =  $10H * SS + BP = 10H * 2000H + 1000H = 21000H$   
//joffo:preco SS a nie DS?  
//lebo BP :D

#### 48.) RET, LOOPNZ, JBE

//podla mna : loopnz,JBE  
//mnicky: rozmyslal som nad nimi, kedze pouzitim JMP vlastne skacu, cize musia upravovat obsah IP asi.... Len v NG je pisane ze si ho neukladaju, tak neviem ci rovno neskocia bez toho, ze by ho zmenili (ale to sa asi neda, co?)....  
//hmmmm...IP je pointer na dalsiu instrukciu ze?...podla mna sa neda len tak skakat hore dole  
//mnicky: asi mas pravdu. Som ja ale odbornik :)

[http://en.wikibooks.org/wiki/X86\\_Assembly/Control\\_Flow#Loop\\_Instructions](http://en.wikibooks.org/wiki/X86_Assembly/Control_Flow#Loop_Instructions)

//juri: čiže, keď je to pointer na ďalšiu inštrukciu, tak by sa mal meniť po vykonaní akejkoľvek inštrukcie, nie? Aj v prednáške píše Čičák o IP, že sa mení vždy.

//stclaus: zas nestastne formulovana otazka. Ano, po kazdej instrukcii sa zmeni, posunie sa nizsie. Ale tieto tri instrukcie ten obsah priamo menia obsah IP a daju don inu hodnotu ako adresu nasledujucej instrukcie

//joffo: urcite aj ret meni IP <http://www.ousob.com/ng/asm/ng17284.php>

<http://i.imgur.com/vWjgW.jpg> ...takze tak :)

skuska 08/09 v2 <https://www.dropbox.com/s/553mtyu56gwcasq/ASPSkuska0809RT-v2.gif>

dalsie otazky: (narazil som na test 2009 z AIS, odpovede su vyznacene AISom).

Vyberam len zaujimave otazky

<https://www.dropbox.com/s/7dq30ikggje1yds/2009.htm>

//ja tam tie odpovede nevidim len ohodnotenie napravo ze kolko bodov dostal, moznosti vidim v pohode ale vyznacenie tam nemam ziadne ci spravne alebo nespravne

//skus to otvorit v exploreri :)

Maximálny počet byteov inštrukcie procesora 8086 je

a)2 b) **6** c) 4 d) 14 e) 1

Základné atribúty premennej sú:

**Segment**      **Typ**      **Názov**      **Offset**      Počet byteov

Inštrukcie implicitne meniace obsah zásobníka sú:

**PUSH**              **INC**      **CMP**      **LEA**      **CALL**

Ktorý bit príznakového registra ovláda spracovanie prerušení:

**AF**      **DF**      **CF**      **IF**      **ZF**

Koncepcia von Neumannovho počítača je charakteristická týmito vlastnosťami:

- procesor vyberie z pamäte takú inštrukciu, na ktorú ukazuje SP register
- procesor vyberie z pamäte takú inštrukciu, ktorej adresa sa nachádza v inštrukčnom registri
- pred vykonaním inštrukcie procesor zisťuje, či sú operandy platné
- **inštrukcie sa vykonávajú podľa poradia ako sú uložené v pamäti, pokiaľ sa nevykona inštrukcia skoku**
- **procesor vyberie z pamäte takú inštrukciu, na ktorú ukazuje IP register**

Nech obsahy registrov sú nasledujúce: DL=0F3H, BH=72H.

Potom obsah bitu S príznakového registra po inštrukcii SUB DL,BH je:

0      1

//stclaus: toto je výsledok z AIS, čiže správny. takže naozaj to znamená najvyšší bit, a nie znamienko

Nech obsahy registrov sú nasledujúce: DL=0F3H, BH=72H.

Potom obsah bitu C príznakového registra po inštrukcii SUB DL,BH je:

0      1

Nech obsahy registrov sú nasledujúce: AX=1001H, DX=20FFH.

Potom obsah bitu A príznakového registra po inštrukcii ADD AX, DX je:

0      1

//A - Auxiliary carry - prenos medzi 3 a 4 bitom

Nech obsahy registrov sú nasledujúce: DL=0F3H, BH=72H.

Potom obsah bitu P príznakového registra po inštrukcii SUB DL,BH je:

0      1

//P - parity: obsahuje výsledok párneho počtu jednotiek?

//joffo: zaujímavá skúška, hlavne so správnymi odpoveďami

<http://files.gamepub.sk/Bakalar/ASM/SkuskaASP2009.pdf>

//Tu otázku s tým busy niekto nevie? 08/09 35. otázka

//stclaus: pozeral som to, v poznámkach z prednášok som nenášiel, nie je to podstatné: P

//okej :D