Универзитет „Св. Кирил и Методиј" во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Факултет за информатички науки и компјутерско инженерство, Скопје

Микропроцесорски системи

Подготовка за лабораториска вежба број 01

МЕМОРИЈА и ВЛЕЗ/ИЗЛЕЗ ВО АСЕМБЛЕР 8086

Меморија и влез/излез во асемблер 8086

Целта на оваа вежба е запознавање со меморијата, како и со работата со влез/излез на една програма.

Предмет на работа

1. Читање од тастатура и запишување на екран.
2. Податочна меморија и променливи.

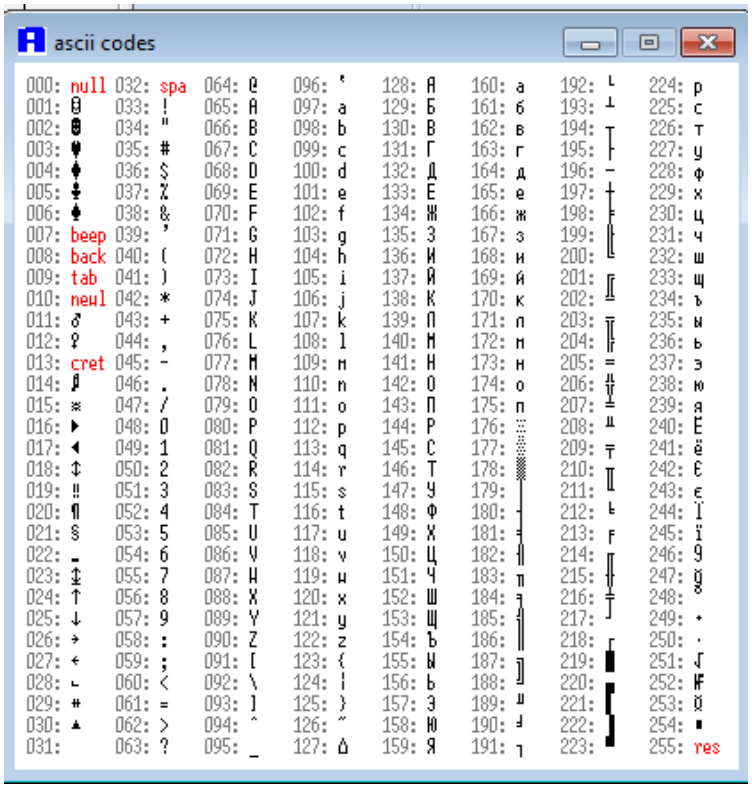
Читање од тастатура и запишување на екран

Со наредбите

```
MOV     dl,65d
MOV     ah,02h
INT     21h
```

се печати вредноста што е внесена во DL на екран. Со третата наредба се остварува DOS прекин (int - interrupt). Типот на прекин зависи од вредноста запишана во AH. Кога вредноста во AH е 2, како во конкретниот пример, по прекинот се печати знакот во DL на екранот. По печатењето, во AL се запишува вредноста од DL.

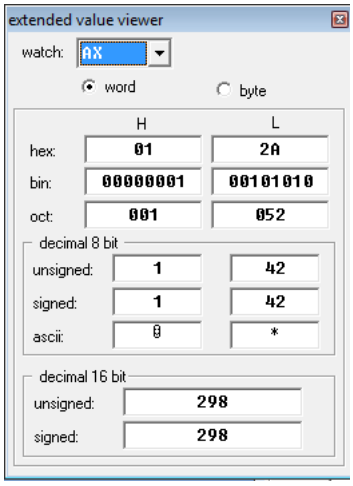
Со дадените инструкции на екранот ќе се отпечати знакот A чиј ASCII код е 65d. За преглед на ASCII кодовите може да ја изберете опцијата ASCII codes од менито во прозорецот со оригиналниот изворен код.



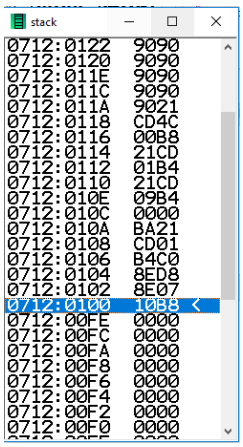
Со наредбите

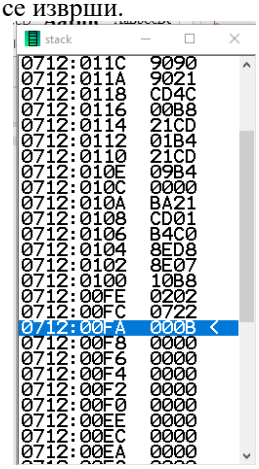
```
MOV     ah,01h
INT     21h
```

се чита вредност од тастатура (еден знак) и истиот се запишува во AL. При извршување на овие инструкции се појавува прозорецот за извршување каде треба да внесете вредност. Ако од тастатура внесеме знак * ќе се смени вредноста во регистерот AL и неговата состојба е следна:



При извршување на инструкцијата за прекин се менува и стекот. Имено нека пред извршувањето на инструкција INT за прекин, состојбата на стекот е како на сликата дадена лево (View->stack). Со извршувањето на инструкцијата за прекин INT на стекот прво ќе се стави вредноста на FLAG регистерот (0202H), па CS регистерот (0722H) и IP регистерот (000B) и стекот ќе изгледа како на сликата дадена десно. Да забележиме дека потоа вредноста на регистрите CS и IP ќе се сменат и ќе покажуваат на следната инструкција во рутината за прекин која треба да се изврши.

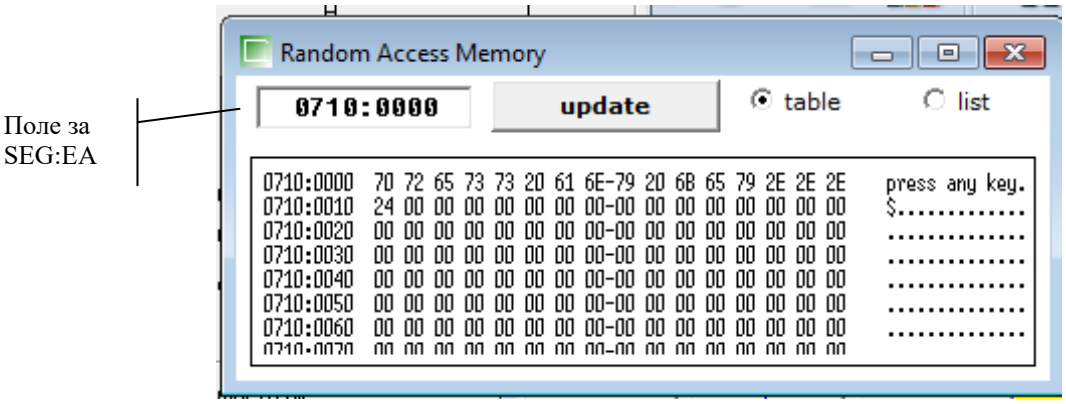




Со инструкцијата IRET вредностите на IP, CS и FLAG регистрите ќе бидат извадени од стек. Потоа програмата ќе продолжи онаму каде што застанала пред прекинот.

Податочна меморија и променливи

За преглед на меморијата во емулаторот има посебен прозорец. Во емулаторот изберете ја опцијата View-> Memory од главното мени. Ќе го добиете следниот прозорец:



Логичката адреса на податок од меморија е во облик SEG: EA. Првиот дел го означува сегментот, а вториот ефективната адреса или поместувањето. За пристап до податочниот сегмент во првиот дел од адресата ставете ја вредноста која се наоѓа во регистерот DS. За пристап до стек сегментот, ставете ја вредноста која се наоѓа во SS. Слично е и за пристап до кодниот сегмент и додатниот сегмент. Ако ефективната адреса ја ставите да биде 0, ќе ја погледнете меморија на избраниот сегмент од почеток.

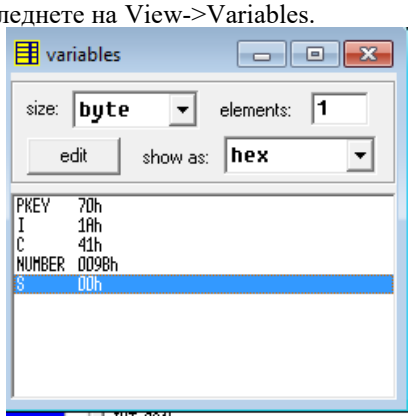
По извршување наредбите објаснети погоре за доделување вредноста на DS и ES во емулаторот ќе може да ги видите вредностите на овие регистри и истите да ги вметнете во прозорецот на меморијата (означено на сликата погоре).

Што има во податочниот сегмент?

Во податочниот сегмент може и да се додаваат променливи. Ќе го напишеме следниот код:

```
i               db      1Ah
c               db      'A'
number          dw      155d
s                       db      ? ; неиницијализирана променлива
```

Проверете каде овие податоци се запишани во податочниот сегмент од меморијата. Постои и посебен прозорец каде што може да ги видите сите декларирани променливи и кои се нивните вредности. Прозорецот може да го погледнете на View->Variables.



Формат на инструкцијата MOV за читање

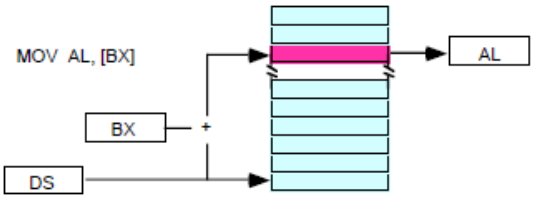
За читање од некоја мемориска локација, потребно е да му кажеме на процесорот 8086 од која локација да чита, и каде да го смети податокот . Постојат повеќе начини на пристап до податоците од меморија, а тука ќе го прикажеме **индиректниот регистарски режим** кој го користи регистарот BX. Постојат и други режими дадени на вежбите.

Кога се чита податок од меморија тој може да се запише *исклучиво* во регистер (од меморија во меморија и од меморија во променлива не може да се направи директен трансфер). Форматот на инструкцијата е следен:

```
MOV <register>, [BX]
```

каде што `<register>` претставува местото во кој регистер ќе се запише податокот, а `[BX]` означува податок кој што е сместен на локацијата DS:BX. Да забележиме дека доколку се користи регистарот BX за адресирање на поместувањето, се подразбира дека станува збор за податочниот регистер. Истото можеше и експлицитно да се напише со инструкцијата:

```
MOV <register>, DS[BX]
```



На пример во вашата програма напишете ги следните инструкции.

```
MOV   BX,100
MOV   CX,[BX]
```

Со првата наредба во BX ќе се зачува вредноста 100. Со следната наредба вредностите запишани во податочниот сегмент со поместување 100 и 101 ќе се запишат во регистерот CX (CH и CL, соодветно). Од меморија се читаат два бајти затоа што регистерот CX има големина од 2 бајти. Погледнете го податочниот сегмент во меморијата и регистерот CX по извршување на инструкциите.

Формат на инструкцијата MOV за запишување

За запишување на некоја мемориска локација, потребно е да му кажеме на процесорот 8086 на која локација да го запише податокот и од каде да го земе податокот кој што ќе се запише. За да се запише вредност во меморија, истата мора да се прочита од регистер. Форматот на инструкцијата е сличен како и читање, со тоа што редоследот на операндите на инструкцијата MOV е променет:

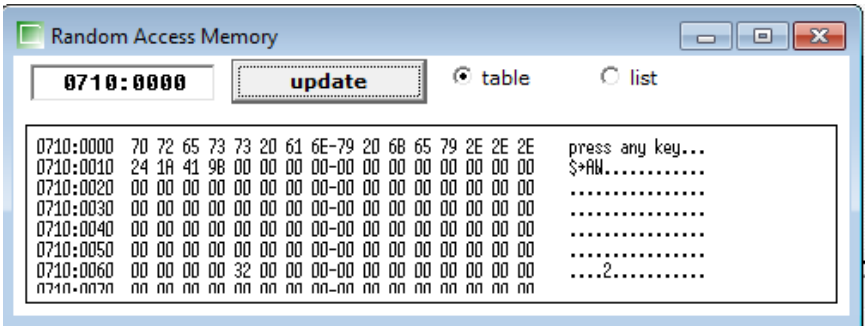
```
MOV [BX], <register>
```

Во регистарот `<register>` е сместен податокот кој што треба да се запише, а `[BX]` ја означува адресата каде што треба да се запише податокот. Физичката адреса каде што ќе се запише податокот е DS: BX.

На пример во вашата програма напишете ги следните инструкции.

```
MOV   BX,100
 MOV   [BX], 50
```

Со првата наредба во BX ќе се зачува вредноста 100. Со следната наредба вредноста 50 ќе се зачува во податочниот сегмент со поместување 10h. Погледнете го податочниот сегмент во меморијата.



Во 100тиот бајт во сегментот е запишана вредност 50, што е всушност ASCII кодот за 2.

Нека е дадена следната инструкција. Вредноста во променливата с (погоре таа променлива беше декларирана) треба да се запише во мемориска локација DS:BX. Оваа инструкција има грешка затоа што од променлива во меморија, како и од меморија во меморија, инструкцијата MOV не може да направи податочен трансфер.

```
MOV   [BX], c
```

За да се оствари целта вредноста од променливата с да се ископира во мемориската локација DS:BX потребно е да се извршат две инструкции со кои посредно преку некој регистер ќе се оствари саканиот трансфер:

```
MOV   AH, c
MOV   [BX], AH
```

Со првата инструкција вредноста во променливата с се копира во AH, а во втората вредноста во AH Се копира во DS:BX. Со овие две наредби на локација DS:BX се наоѓа ASCII кодот на A.

Покрај дадените има и други варијанти на инструкцијата MOV со која се прави податочен трансфер. Тие се дадени во следниот дел.

Други примери за инструкцијата MOV

Има повеќе форми во кои може да се искористи инструкцијата MOV. Тука се дадени повеќе примери. Извршете ги следните инструкции. Во коментар се објаснети инструкциите и е дадено кои од нив се невалидни и зашто.

```
MOV AX, BX ;(регистерски режим) Се копира вредност од BX во AX
MOV AL, [10h] ;(директен режим) Се копира еден бајт од DS:10h во AL
MOV BX, 10h   ;(непосреден режим) Се копира еден 10h во BX
MOV AX, DS:[BX] ;(регистерски индиректен режим)
                ; се копираат два бајти од DS:10h во AX
MOV AX, DS:[CX] ;грешка: не може да се користи
                ; регистерот CX за ефективна адреса

MOV c, i          ;грешка: не може директно да се копира вредност
                  ; од една во друга променлива
MOV AL, i      ;Се копира вредноста од i во AL
MOV AX, i     ;грешка: i и AX имаат различна големина

MOV AX, [BX+10]  ;(регистерски индиректен режим со поместување)
                 ; се копираат два бајти од DS: (10h+BX) во AX
MOV AX, i[BX]  ;грешка: поместувањето I мора да биде од тип dw

MOV AX, [BX+number] ;(регистерски индиректен режим со поместување)
                    ; се копираат два бајти од DS: (2+BX) во AX
                    ; 2 е адресата на number
```