



Bölüm 8: Makrolar

Mikroişlemciler



Makrolar

- Prosedürlere benzer.
- Derleme aşamasında gerçek komutlar ile değiştirilen geçici yapılardır.
- Kod içerisinde kullanılmayan makrolar görmezden gelinir.
- Makrolar, prosedürlerden farklı olarak,
 - kullanıldıkları kodun üzerinde tanımlanmalıdır.



Makro Tanımlama

```
name MACRO [parametreler,...]  
    <komutlar>  
ENDM
```



Örnek Kod Parçası

```
MyMacro MACRO p1, p2, p3
    MOV AX, p1
    MOV BX, p2
    MOV CX, p3
ENDM
```



```
MOV AX, 00001h
MOV BX, 00002h
MOV CX, 00003h
MOV AX, 00004h
MOV BX, 00005h
MOV CX, DX
```

```
ORG 100h
MyMacro 1, 2, 3      ; Makro kullanımı
MyMacro 4, 5, DX     ; Başka bir örnek
RET
```











Örnek Kod Parçası

emulator: z01.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

	H	L
AX	00	04
BX	00	02
CX	00	03
DX	00	00
CS	07 00	
IP	01 0C	
SS	07 00	
SP	FF FE	
BP	00 00	
SI	00 00	
DI	00 00	
DS	07 00	
ES	07 00	

07 00: 01 0C

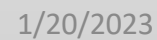
07100:	B8	184	7
07101:	01	001	0
07102:	00	000	NULL
07103:	BB	187	7
07104:	02	002	0
07105:	00	000	NULL
07106:	B9	185	7
07107:	03	003	7
07108:	00	000	NULL
07109:	B8	184	7
0710A:	04	004	7
0710B:	00	000	NULL
0710C:	BB	187	7
0710D:	05	005	7
0710E:	00	000	NULL
0710F:	8B	139	7
07110:	CA	202	7
07111:	C3	195	7
07112:	90	144	E
07113:	90	144	E
07114:	90	144	E
07115:	90	144	E

07 00: 01 0C

```
MOV AX, 00001h
MOV BX, 00002h
MOV CX, 00003h
MOV AX, 00004h
MOV BX, 00005h
MOV CX, DX
RET
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

screen source reset aux vars debug stack flags







Prosedür Özellikler

- Prosedür CALL komutu ile çağrılır.
- Prosedür bir bellek adresinde bulunur.
- Çağrıldığında CPU kontrolü bu bölgeye aktarır.
- RET komutu ile kontrol, programın ana kısmına geri döner.
- Dönüş adresini saklamak için yığın kullanılır.
- Aynı prosedür 100 kez çağrılırsa dosyanın boyutunu çok az arttırır.
 - CALL komutu 3 byte yer kaplar,
- Prosedürlere parametre geçmek için yığın veya yazmaçlar kullanılır.
- Prosedürü sonlandırmak için prosedür adı ve ENDP direktifi yazılır.



Makro Özellikler

- Makro sadece adı yazılarak kullanılır.
- Makrolar program kodunda doğrudan genişletilir.
- Aynı makro 100 kez kullanılırsa, derleyici makroyu 100 kez genişletir.
 - Yürütülebilir dosya boyutu giderek büyür.
- Makroya parametre geçmek için makro adının ardından değerler yazılır.
- Makro sonlandırmak için ENDM direktifi yeterlidir.
- Macro tanımında etiketler varsa,
 - birden fazla kez kullanıldığında *Duplicate declaration* hatası alınabilir.
 - değişken, etiket, prosedür adları için LOCAL direktifi kullanılır.



Örnek Kod Parçası

```
MyMacro2 MACRO
    LOCAL label1, label2
    CMP  AX, 2
    JE  label1
    CMP  AX, 3
    JE  label2
label1:
    INC  AX
label2:
    ADD  AX, 2
ENDM
```



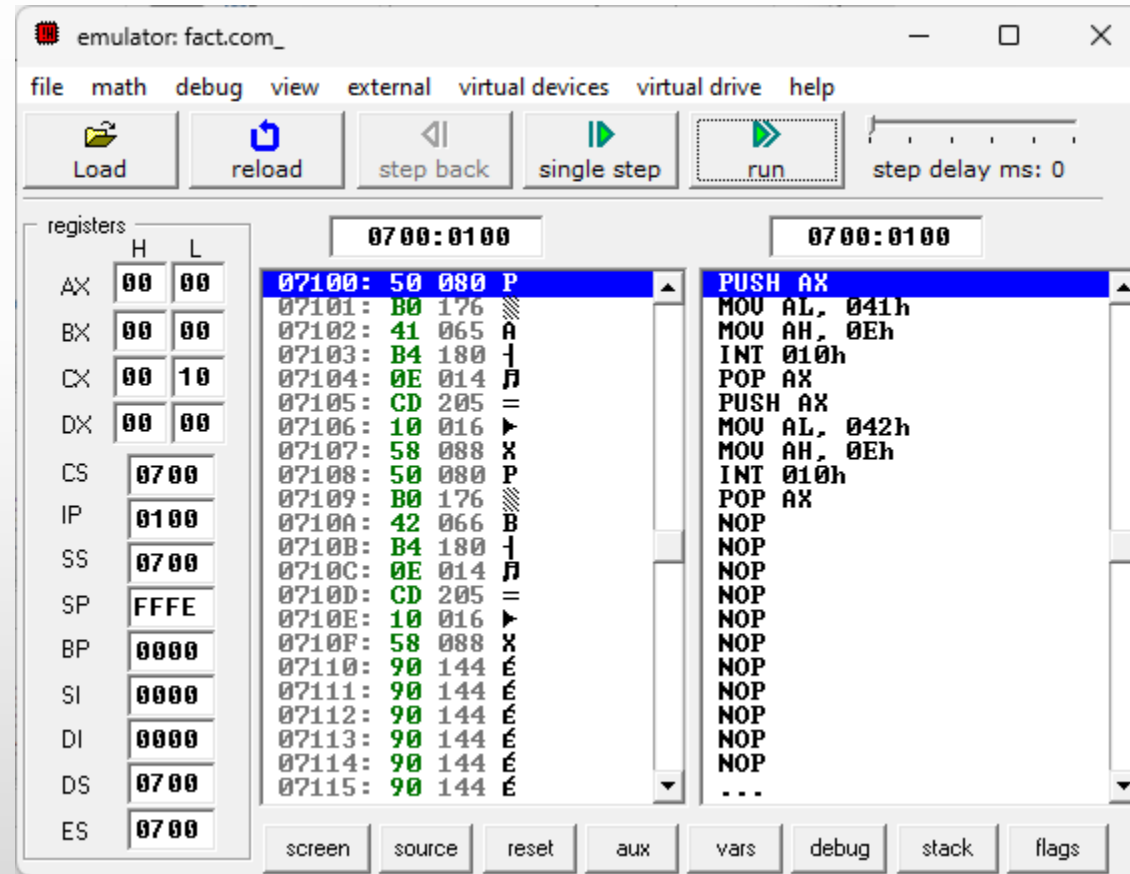
Ekrana Karakter Yazdırma

; this macro prints a char in AL and advances the current cursor position

```
PUTC  MACRO  char
    push    ax
    mov     al, char
    mov     ah, 0eh
    int     10h
    pop     ax
ENDM
ORG 100h
START:
    PUTC    'A'
    PUTC    'B'
```

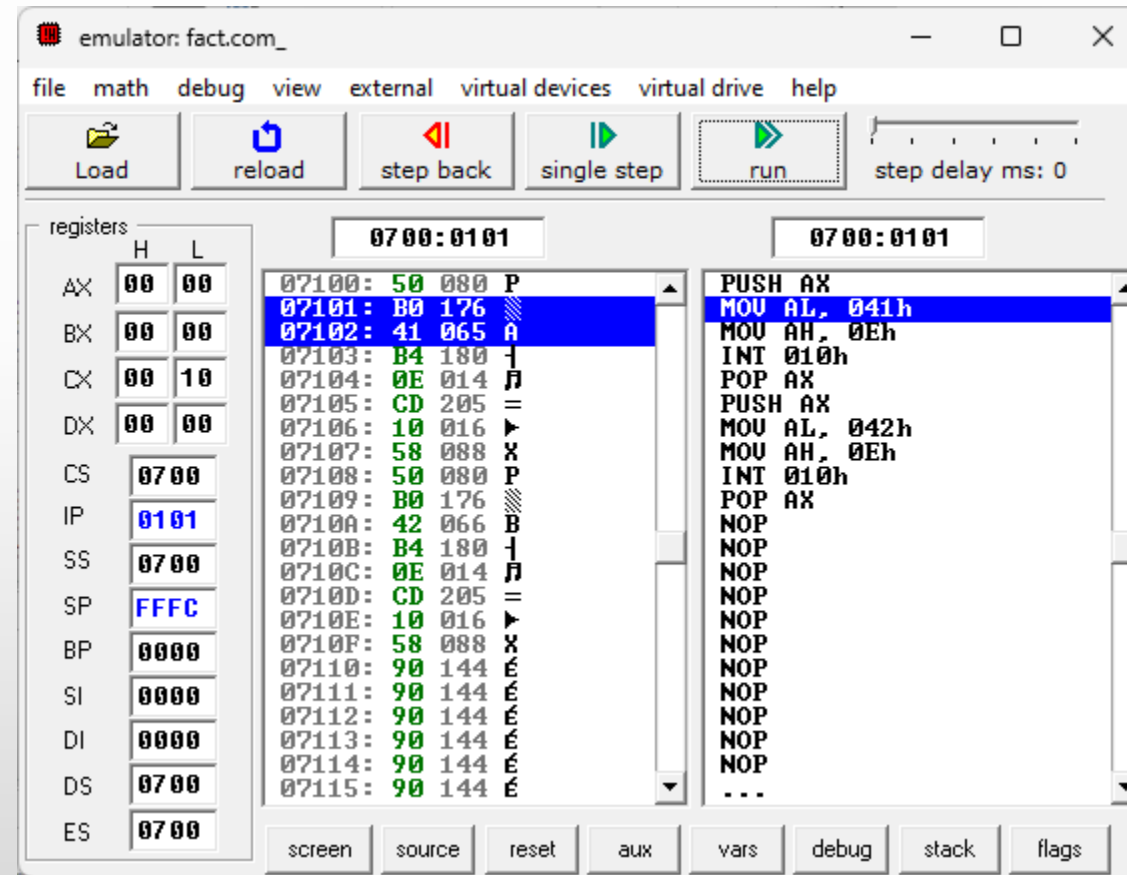


Ekрана Karakter Yazdırma



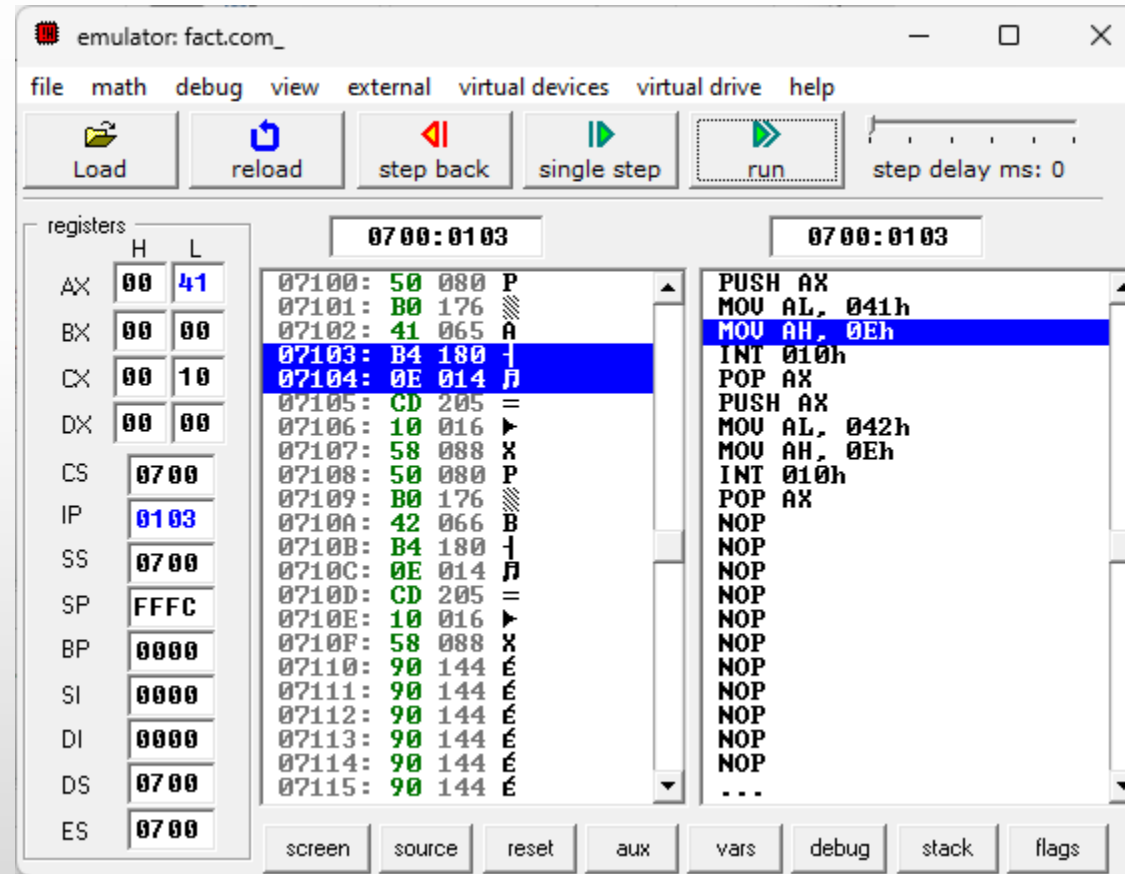


Ekрана Karakter Yazdırma



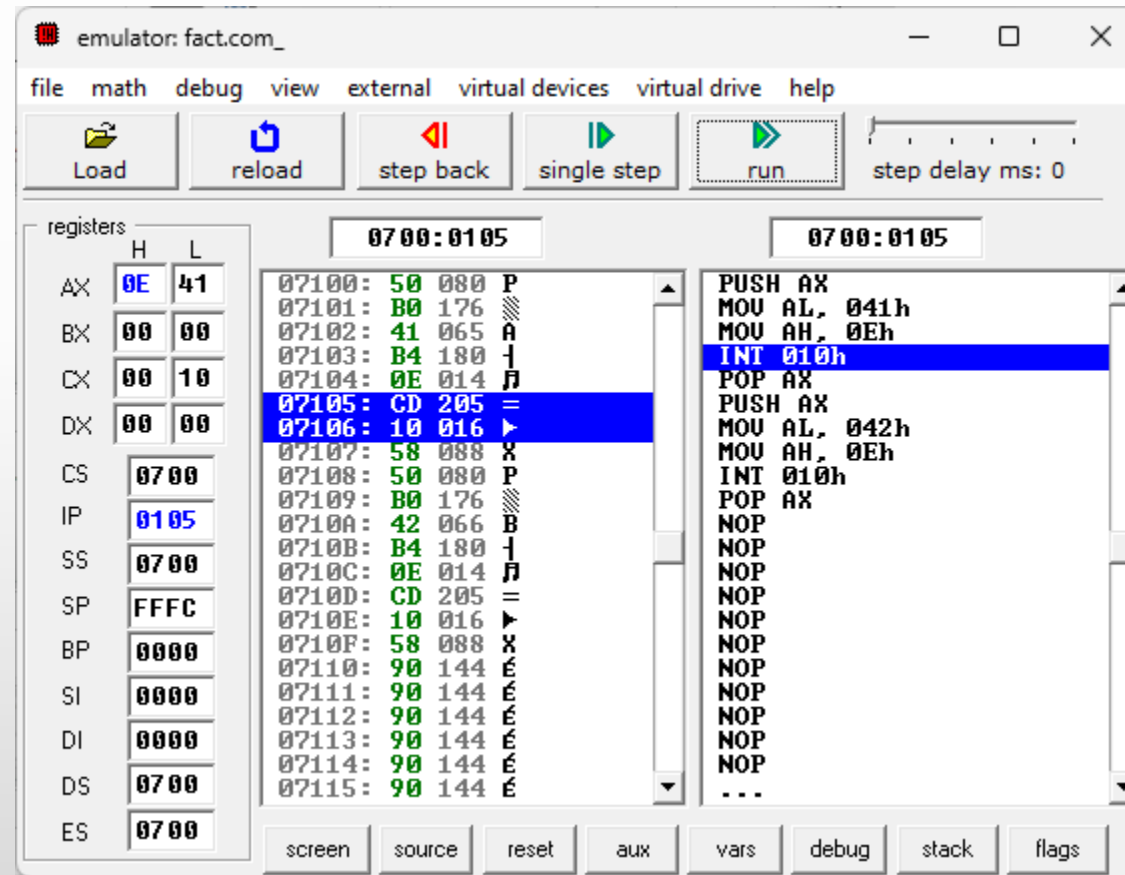


Ekрана Karakter Yazdırma



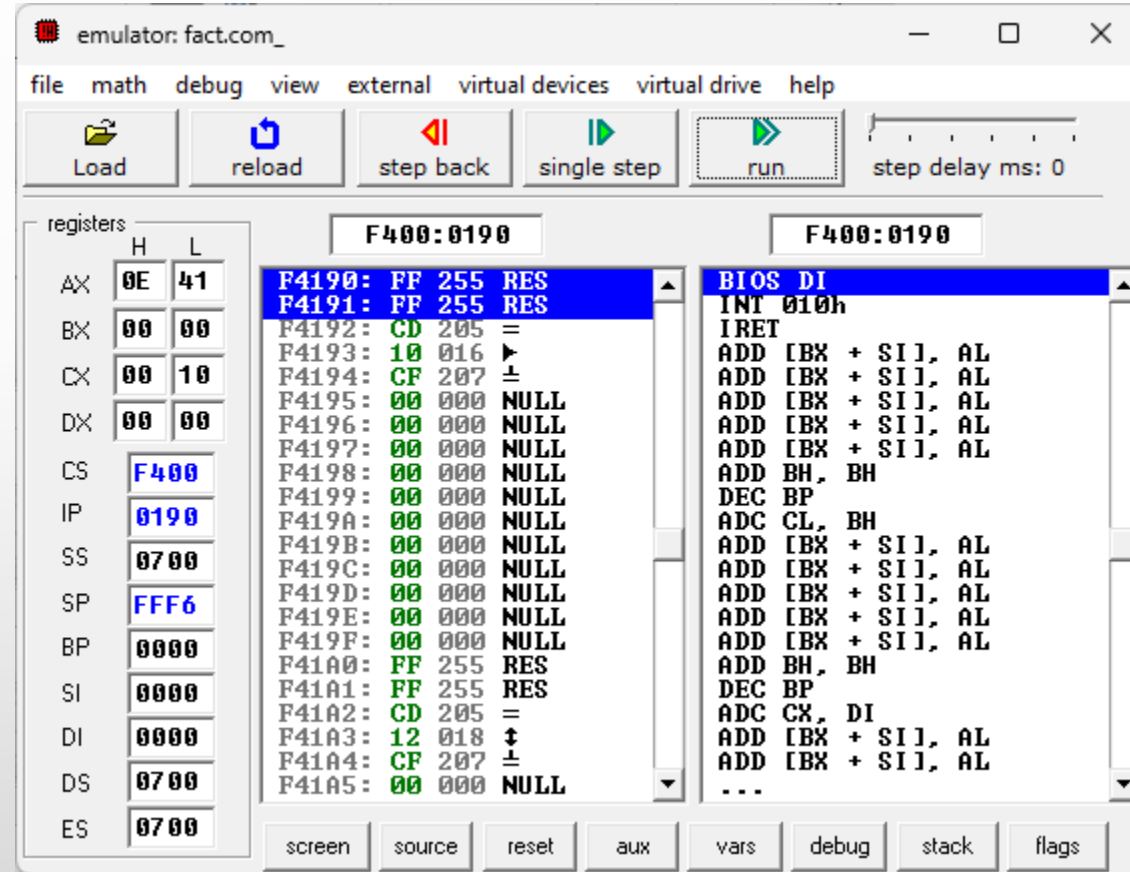


Ekрана Karakter Yazdırma





Ekrana Karakter Yazdırma





Ekrana Karakter Yazdırma

emulator: fact.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

	H	L
AX	0E	41
BX	00	00
CX	00	10
DX	00	00
CS	F400	
IP	0194	
SS	0700	
SP	FFF6	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0194

F4190:	FF	255	RES
F4191:	FF	255	RES
F4192:	CD	205	=
F4193:	10	016	▶
F4194:	CF	207	±
F4195:	00	000	NULL
F4196:	00	000	NULL
F4197:	00	000	NULL
F4198:	00	000	NULL
F4199:	00	000	NULL
F419A:	00	000	NULL
F419B:	00	000	NULL
F419C:	00	000	NULL
F419D:	00	000	NULL
F419E:	00	000	NULL
F419F:	00	000	NULL
F41A0:	FF	255	RES
F41A1:	FF	255	RES
F41A2:	CD	205	=
F41A3:	12	018	↑
F41A4:	CF	207	±
F41A5:	00	000	NULL

BIOS DI
INT 010h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADC CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADC CX, DI
ADD [BX + SI], AL
ADD [BX + SI], AL
...

screen source reset aux vars debug stack flag

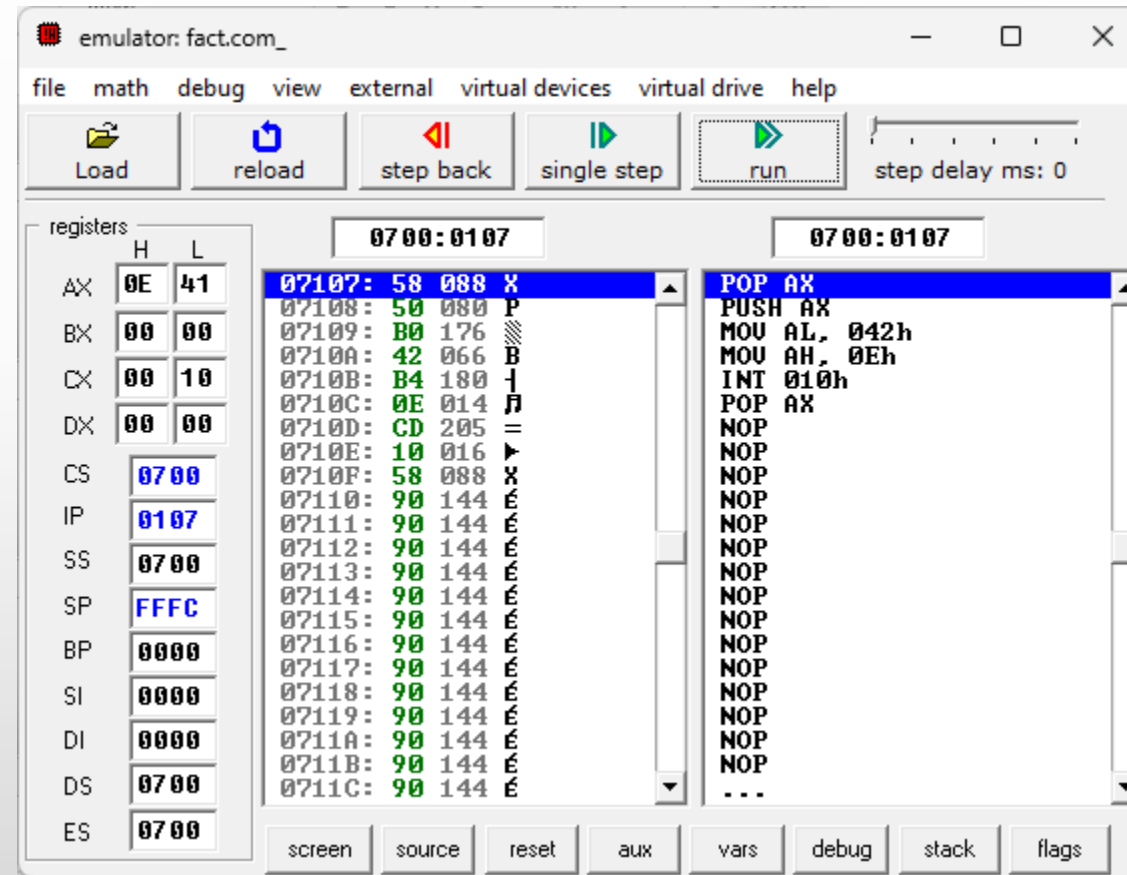
emulator screen (80x25 chars)

A

clear screen change font 0/16



Ekрана Karakter Yazdırma





Ekрана Karakter Yazdırma

emulator: fact.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

	H	L
AX	0E	42
BX	00	00
CX	00	10
DX	00	00
CS	F400	
IP	0194	
SS	0700	
SP	FFF6	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

F400:0194

F4190:	FF	255	RES
F4191:	FF	255	RES
F4192:	CD	205	=
F4193:	10	016	▶
F4194:	CF	207	±
F4195:	00	000	NULL
F4196:	00	000	NULL
F4197:	00	000	NULL
F4198:	00	000	NULL
F4199:	00	000	NULL
F419A:	00	000	NULL
F419B:	00	000	NULL
F419C:	00	000	NULL
F419D:	00	000	NULL
F419E:	00	000	NULL
F419F:	00	000	NULL
F41A0:	FF	255	RES
F41A1:	FF	255	RES
F41A2:	CD	205	=
F41A3:	12	018	↑
F41A4:	CF	207	±
F41A5:	00	000	NULL

BIOS DI
INT 010h
IRET

ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADC CL, BH
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD BH, BH
DEC BP
ADC CX, DI
ADD [BX + SI], AL
ADD [BX + SI], AL
...

screen source reset aux vars debug stack fla

emulator screen (80x25 chars)

AB

clear screen change font 0/16



Üs Alma

POWER macro b, e, r

mov ax, 1	; geçici sonucu 1 yap
mov cx, e	; üs değerini cx yazmacına yükle
mov bx, b	; taban değerini bx yazmacına yükle

powerloop:

mul bx	; taban ile geçici sonucu çarp
loop powerloop	; üs kere döngüyü tekrarla
mov r, ax	; sonucu result değişkenine yaz

endm

Üs Alma



start:

```
    POWER base, exponent, result    ; power makrosunu çağır
```

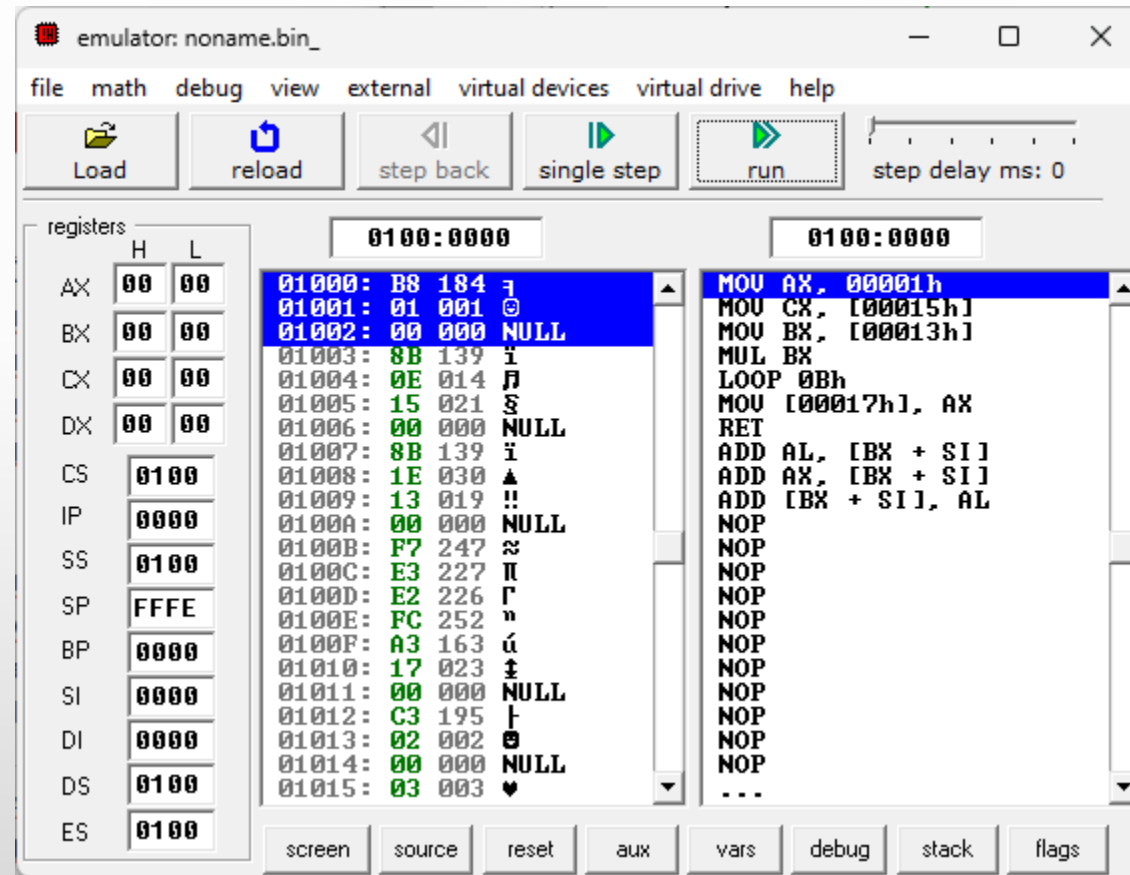
```
ret
```

```
base        dw    2        ; taban değeri
```

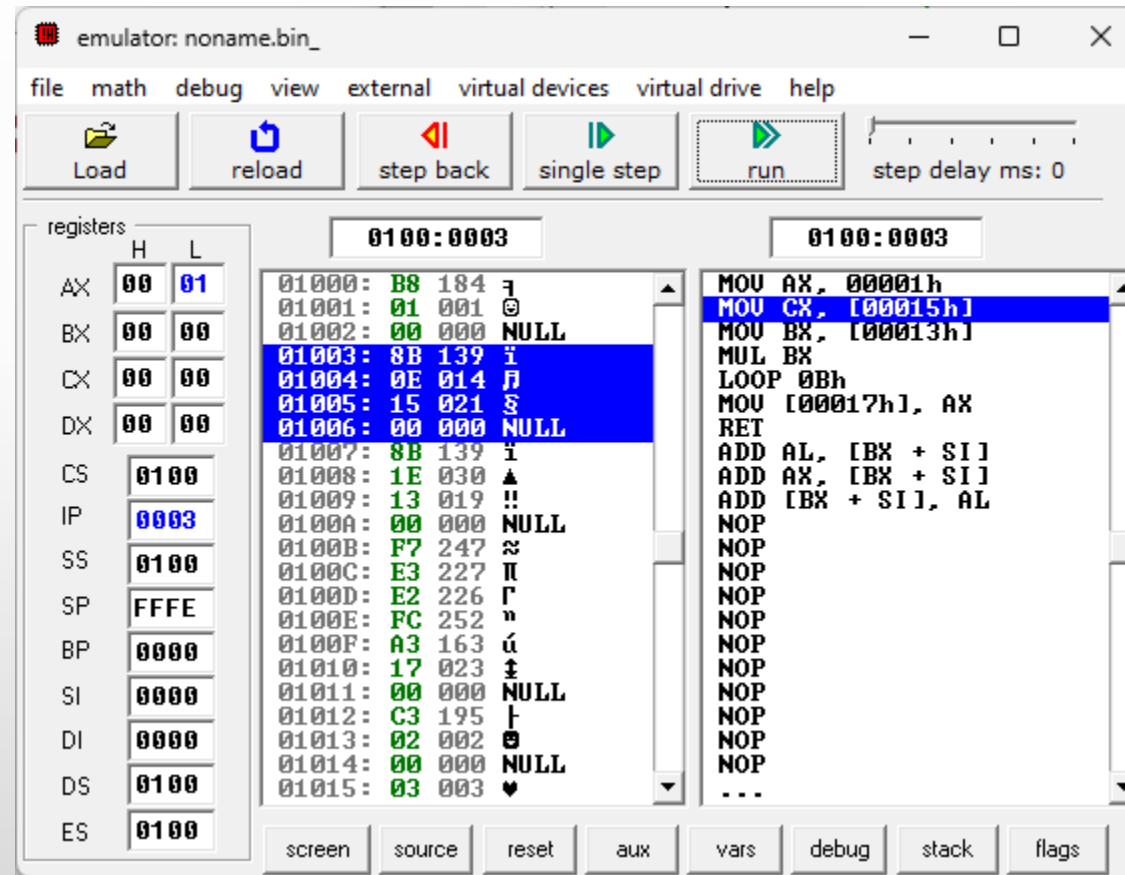
```
exponent    dw    3        ; üs değeri
```

```
result      dw    ?        ; sonuç
```

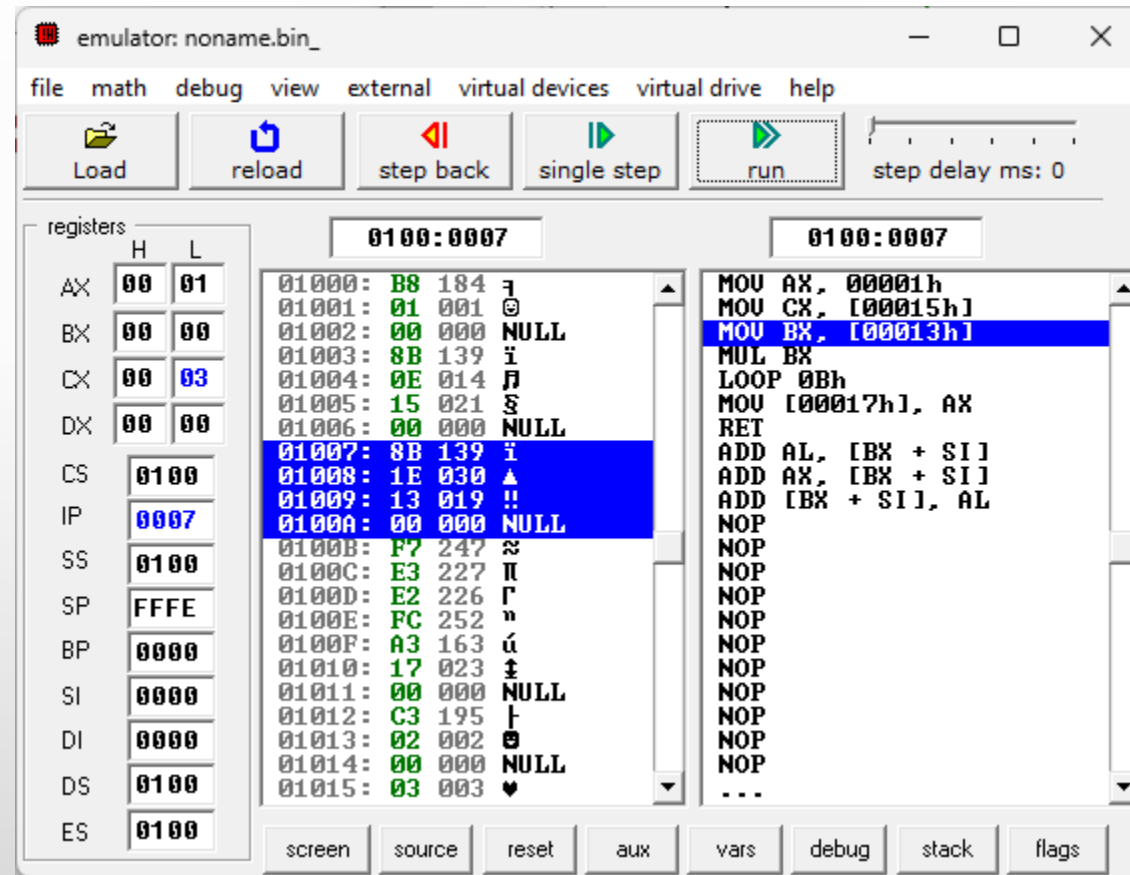
Üs Alma



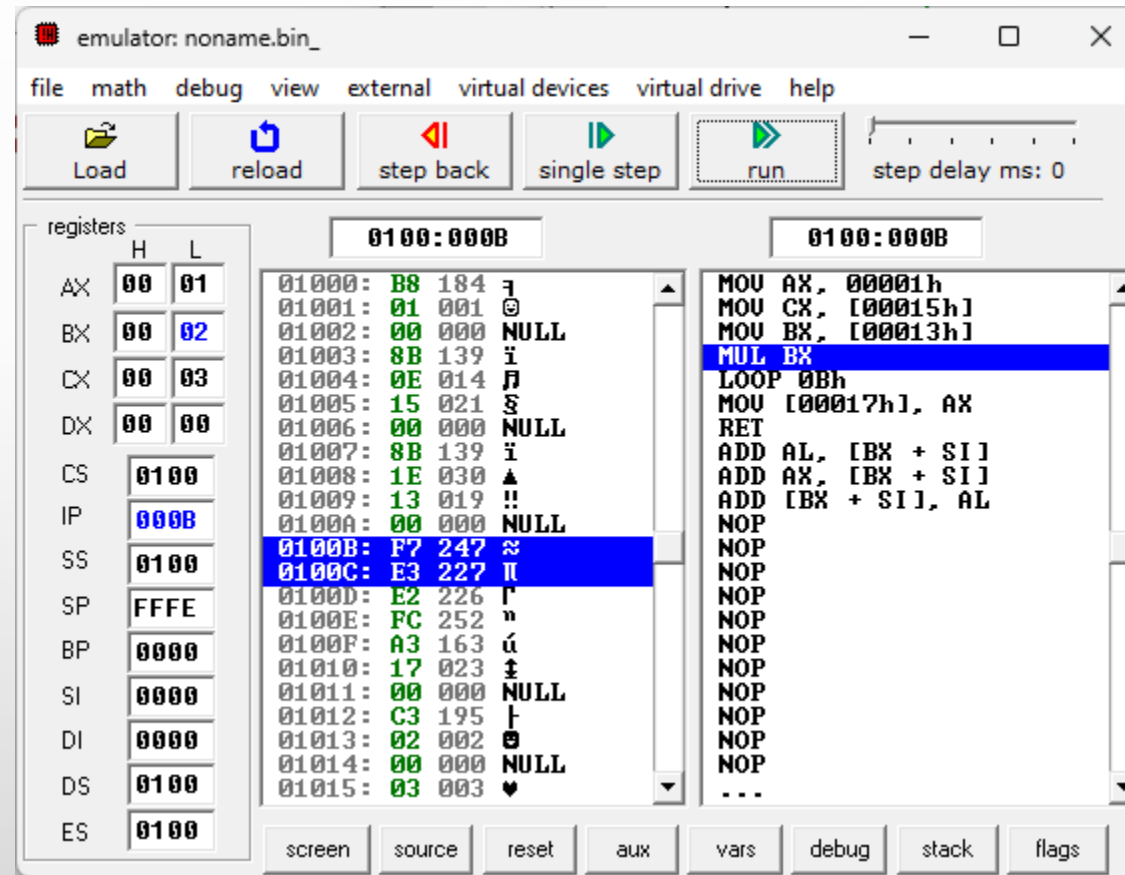
Üs Alma



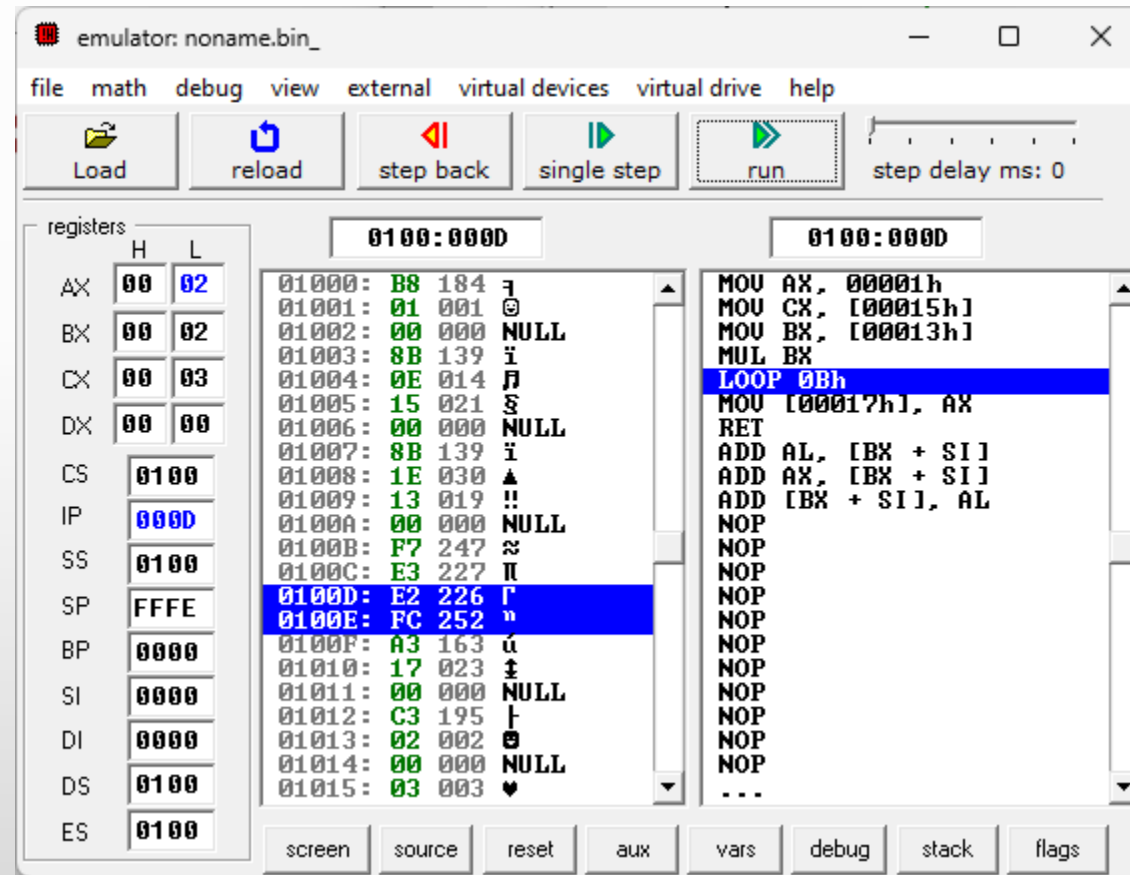
Üs Alma



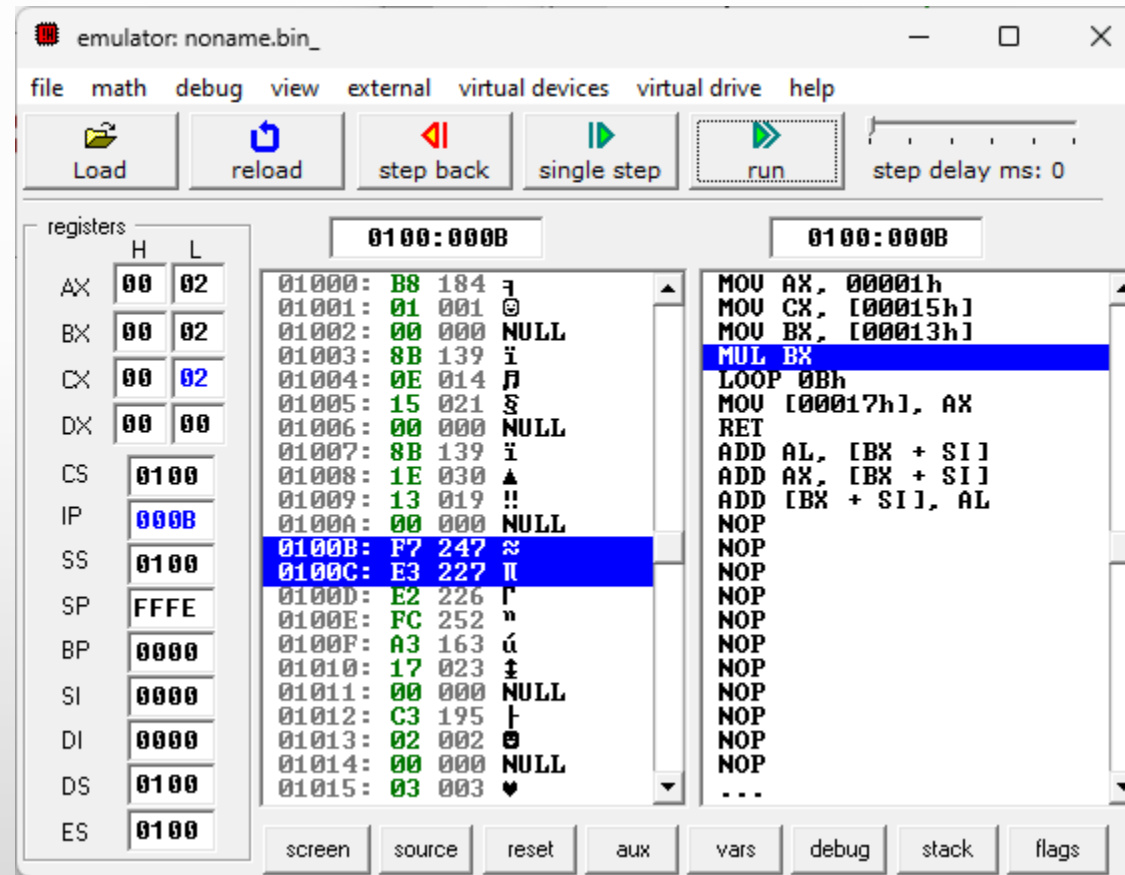
Üs Alma



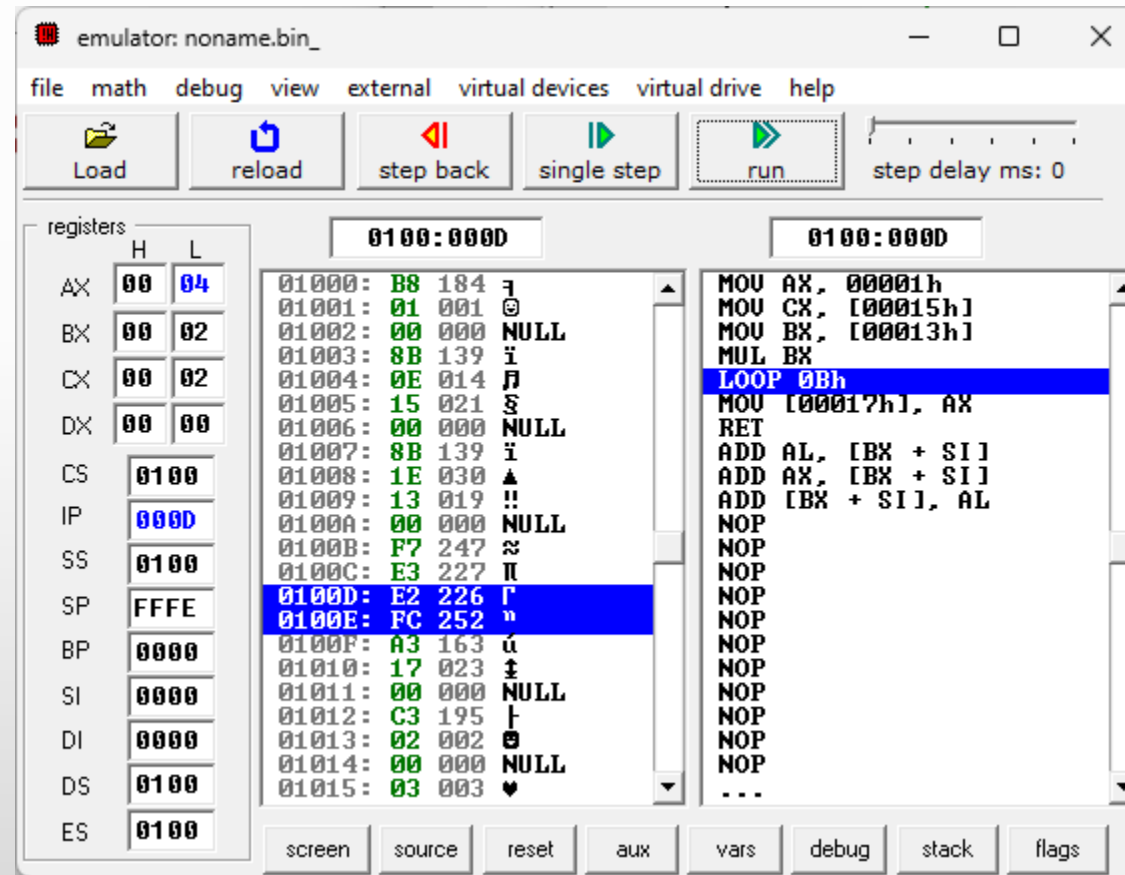
Üs Alma



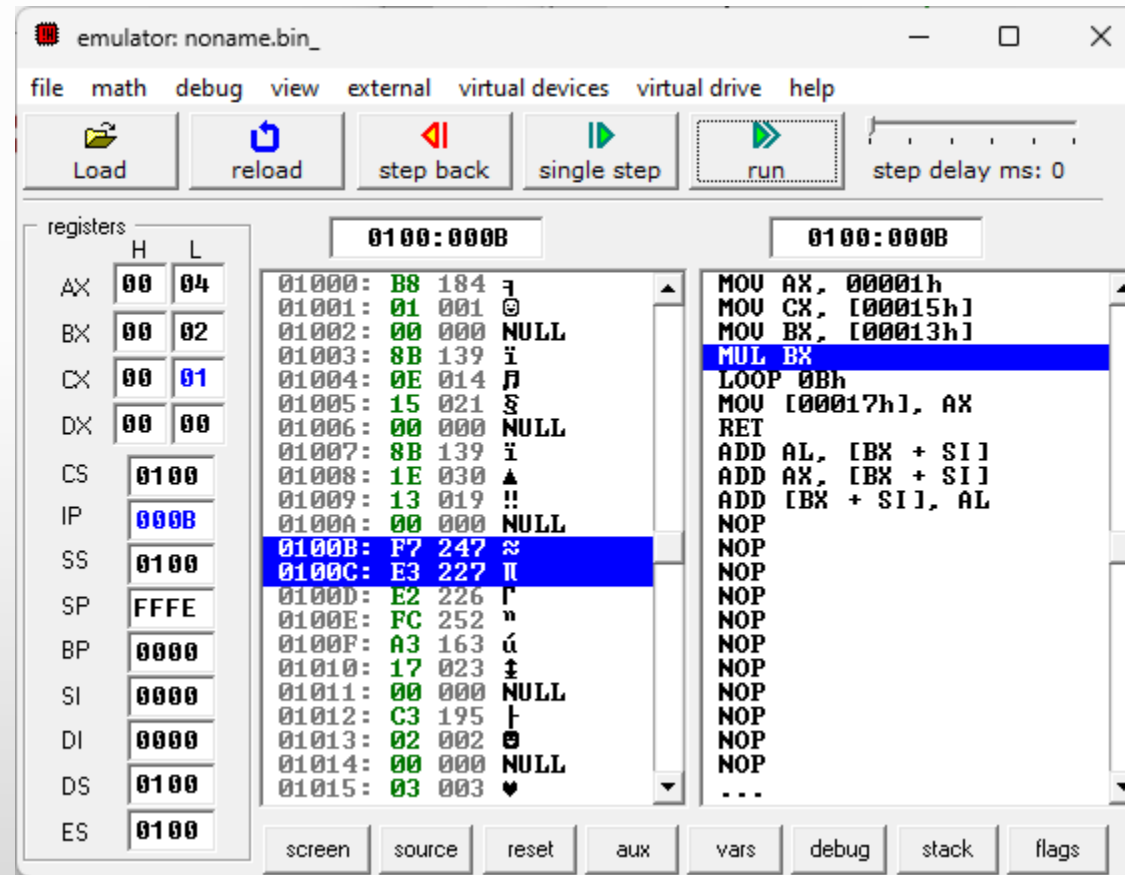
Üs Alma



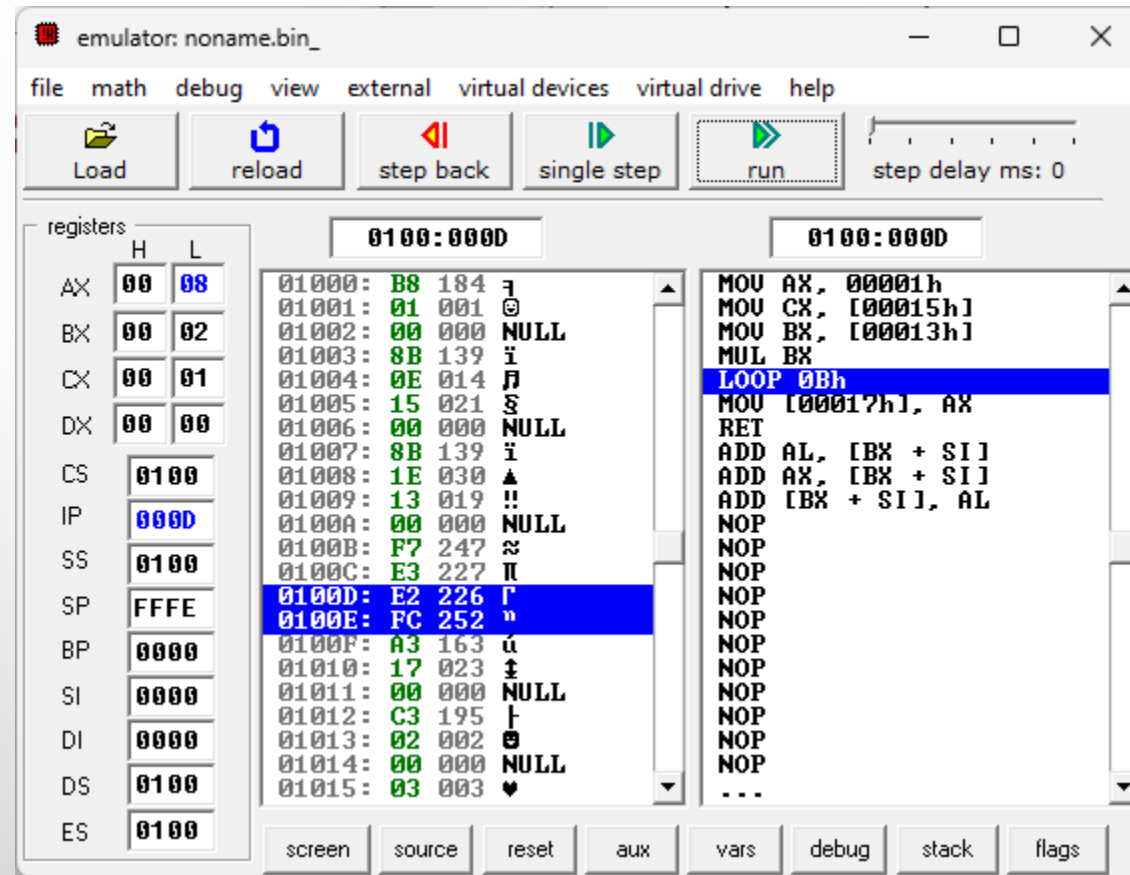
Üs Alma



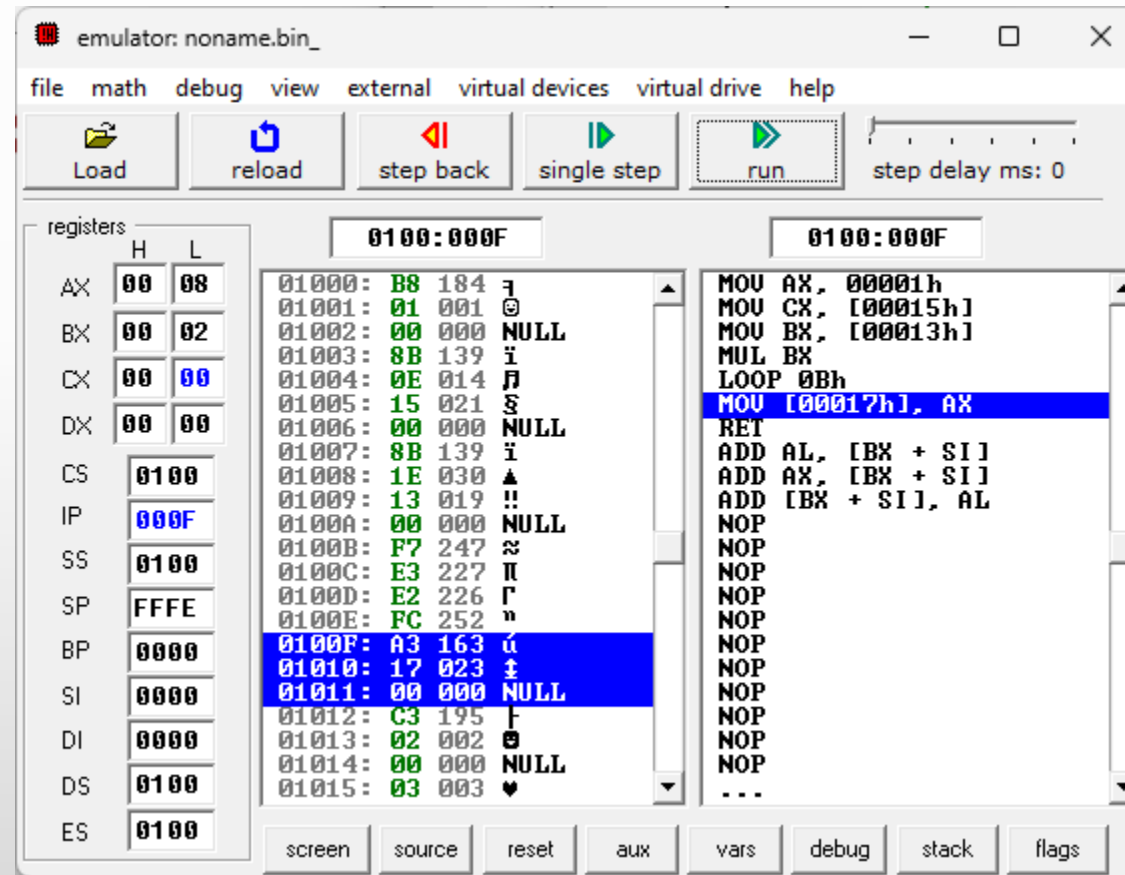
Üs Alma



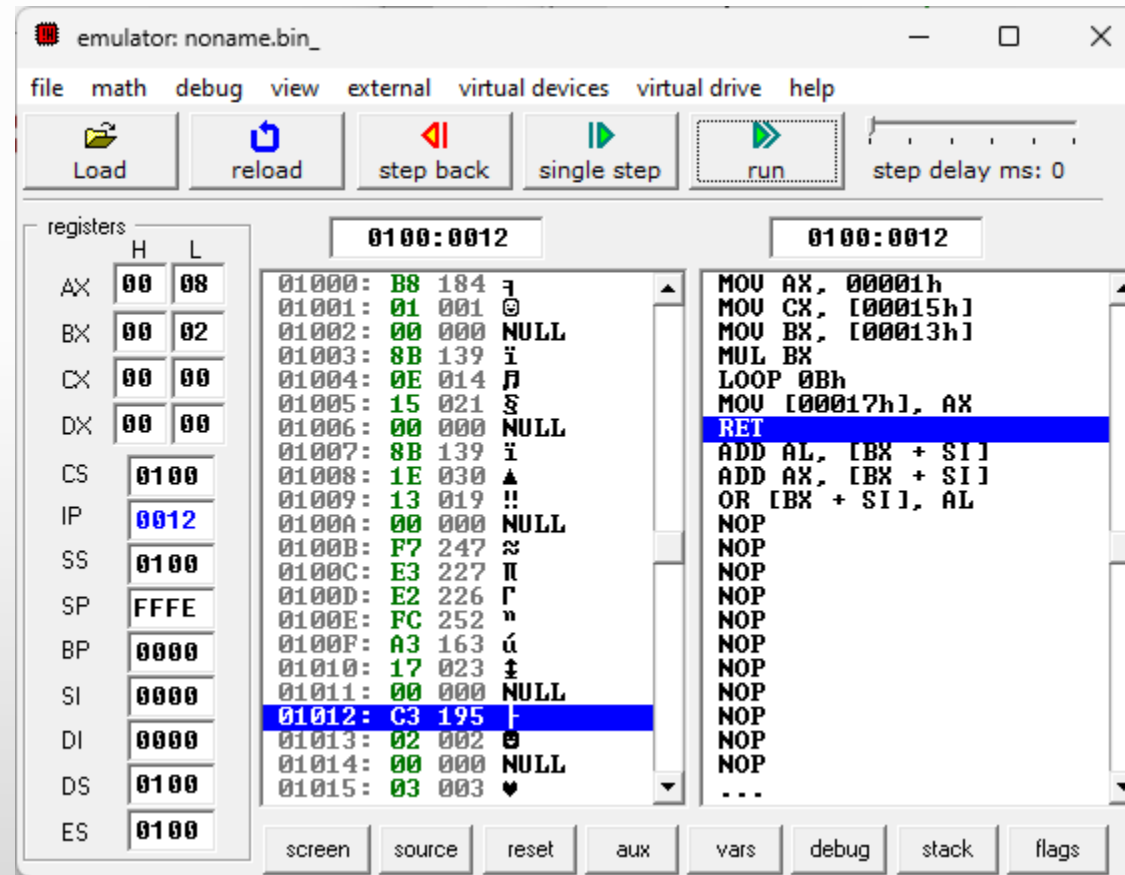
Üs Alma



Üs Alma



Üs Alma





SON