



Bölüm 4: Değişkenler

Mikroişlemciler



Değişkenler

- Bir değişken, bir bellek konumunu temsil eder.
- Programcı için, *var1* değişkeninin
 - *5A73:235B* adresi yerine kullanılması çok daha kolaydır.
- Derleyici iki tür değişkeni destekler.
 - BYTE: Bir bayt (8 bit) değerini temsil eder.
 - name DB value (*Define Byte*).
 - WORD: Bir sözcük (16 bit) değerini temsil eder.
 - name DW value (*Define Word*).
- counter DB 10h (counter adında bir BYTE değişkeni).
- sum DW 0FFFFh (sum adında bir WORD değişkeni).



Değişkenler

- Değişken adları,
 - harfle başlamalıdır,
 - sonrasında harf veya rakam kombinasyonları içerebilir.
- İsim belirtilmeyen değişkenler,
 - isim belirtilmemiş olabilir ancak bir adresleri vardır.
- Herhangi bir sayısal değer (ikili, onlu, onaltılı) alabilir.
- İlk değer atanmamış değişkenler için "?" sembolü kullanılır.



Örnek Kod

```
ORG 100h
MOV AL, var1    ; AL, var1 değişkeninin değerini alır.
MOV BX, var2    ; BX, var2 değişkeninin değerini alır.
RET             ; Programı sonlandırır.
VAR1 DB 7       ; BYTE türünde var1 değişkeni, değeri 7.
var2 DW 1234h   ; WORD türünde var2 değişkeni, değeri 1234h.
```



Örnek Kod

•

memory (1K) at: 0B56 : 0100 Disassemble from: 0B56 : 0100

Address	Hex	ASCII	Instruction
0100	A0 160	á	MOV AL, [00108h]
0101	08 008		MOV BX, [00109h]
0102	01 001	@	RET
0103	8B 139	i	POP ES
0104	1E 030	▲	XOR AL, 012h
0105	09 009		ADD [BX + SI], AL
0106	01 001	@	ADD [BX + SI], AL
0107	C3 195	†	ADD [BX + SI], AL
0108	07 007	•	ADD [BX + SI], AL
0109	34 052	4	ADD [BX + SI], AL
010A	12 018	‡	ADD [BX + SI], AL
010B	00 000		ADD [BX + SI], AL
010C	00 000		ADD [BX + SI], AL
010D	00 000		ADD [BX + SI], AL
010E	00 000		ADD [BX + SI], AL
010F	00 000		ADD [BX + SI], AL
0110	00 000		ADD [BX + SI], AL
0111	00 000		ADD [BX + SI], AL
0112	00 000		ADD [BX + SI], AL

variables



Örnek Kod

- Derleyici,
 - Kaynak kodu bir dizi bayta dönüştürür.
 - Değişken adlarını bellek konumlarıyla değiştirir.
 - Büyük-küçük harfe duyarsızdır ("VAR1" ve "var1" aynı değişken).
- COM dosyaları yüklendiğinde DS, CS yazmacının değerini alır.
- Bağıl konum (*offset*), bellek konumunun başlangıcından olan kaydırmadır.
- VAR1'in bağıl konumu 0108h, tam adresi 0B56:0108'dir.
- var2'nin bağıl konumu 0109h, tam adresi 0B56:0109'dur.
 - WORD türünde olduğu için 2 BYTE kaplar.
 - Düşük bayt düşük adreste saklanır, 34h, 12h'den önce yer alır.



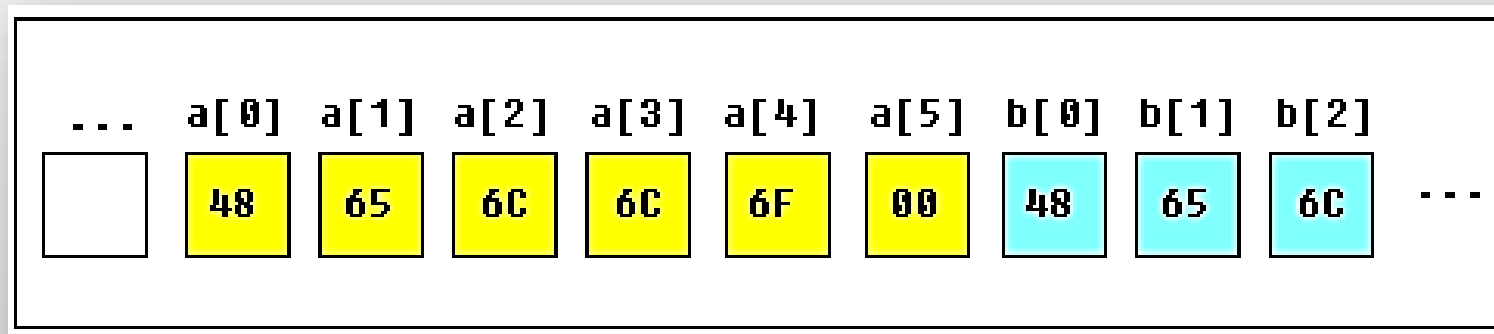
ORG 100h Direktifi

- Derleyiciye yürütülebilir dosyanın yükleneceği adresi söyler.
 - Değişken adları bellek adresleriyle değiştirilirken dikkate alınır.
- COM dosyalarında kullanılır.
- İşletim sistemi, CS'nin ilk 256 baytında programla ilgili bazı verileri tutar.
 - Komut satırı parametreleri gibi bilgiler bu alanda saklanır.
- EXE dosyaları 0000 ofsetinde yüklenir ve
 - Değişkenler için özel bir kesim (*segment*) kullanır.
- Direktifler, gerçek makine koduna dönüştürülmez.



Diziler

- Diziler, değişken zincirleri olarak düşünülebilir.
- Diziler, ardışık bellek konumlarında saklanan veri gruplarıdır.
- Tırnak içindeki metin otomatik olarak bayt dizisine dönüştürür.
- Dizi elemanlarına bellek adresi üzerinden erişilebilir.
- a DB 48h, 65h, 6Ch, 6Ch, 6Fh, 00h
- b DB 'Hello', 0.





Diziler

- Dizinin herhangi bir elemanına,
 - Köşeli parantez kullanarak erişilebilir.
 - Örneğin: `MOV AL, a[3]`
 - Bellek indis yazmaçları BX, SI, DI, BP kullanarak erişilebilir.
 - Örneğin: `MOV SI, 3` ve `MOV AL, a[SI]`
- Büyük bir dizi tanımlamak için DUP işleci kullanılır.
 - `number DUP (value(s))`
 - `c DB 5 DUP(9) → c DB 9, 9, 9, 9, 9`
 - `d DB 5 DUP(1, 2) → d DB 1, 2, 1, 2, 1, 2, 1, 2, 1, 2`
- DW dizi tanımlamak için kullanılamaz.!



Değişkenin Adresini Alma

- LEA ve OFFSET, değişkenin adresini almak için kullanılır.
- LEA, indisli değişkenlerin adresini almak için de kullanılabilir.
- Değişkenin adresi, prosedüre parametre geçirmek için çok kullanışlıdır.



Değişkenin Adresini Alma

```
ORG 100h
MOV     AL, VAR1           ; VAR1 değerini AL yazmacına kopyala.
LEA     BX, VAR1           ; VAR1 adresini BX yazmacına kopyala.
MOV     BYTE PTR [BX], 44h ; VAR1 değerini güncelle.
MOV     AL, VAR1           ; VAR1 değerini AL yazmacına kopyala.
RET
VAR1    DB    22h
END
```



Değişkenin Adresini Alma

```
ORG 100h
MOV     AL, VAR1           ; VAR1 değerini AL yazmacına kopyala.
MOV     BX, OFFSET VAR1    ; VAR1 adresini BX yazmacına kopyala.
MOV     BYTE PTR [BX], 44h ; VAR1 değerini güncelle.
MOV     AL, VAR1           ; VAR1 değerini AL yazmacına kopyala.
RET
VAR1    DB    22h
END
```



Değişkenin Adresini Alma

- LEA BX, VAR1 ve MOV BX, OFFSET VAR1 aynı makine koduna derlenir:
 - MOV BX, num
 - num, değişkenin bağıl konum değerinin 16 bitlik temsili.
- Sadece BX, SI, DI, BP yazmaçları köşeli parantez içinde kullanılabilir.
- LEA (Load Effective Address), değişkenin adresini yazmaca yükler.
- OFFSET, değişkenin bağıl konum değerini sağlar.



Sabitler (Constants)

- Değişkenlere benzer.
- Ancak program derlendiğinde var olurlar.
- Bir sabitin değeri tanımlandıktan sonra değiştirilemez.
- Sabitler, programın farklı bölümlerinde aynı değer kullanılmasını sağlar.
- Sabit tanımlamak için EQU direktifi kullanılır:
 - name EQU <herhangi bir ifade>.
 - k EQU 5
 - MOV AX, k



Değişkenler Arasında Veri Taşıma

DATA SEGMENT

value1 DW 10 ; İki byte'lık bir değişken tanımlama

value2 DW 0 ; İkinci bir değişken tanımlama

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA ; DATA kesiminin adresini AX yazmacına taşı

MOV DS, AX ; DS yazmacına DATA kesiminin adresini yükle

MOV AX, value1 ; value1 değişkeninin değerini AX yazmacına taşı

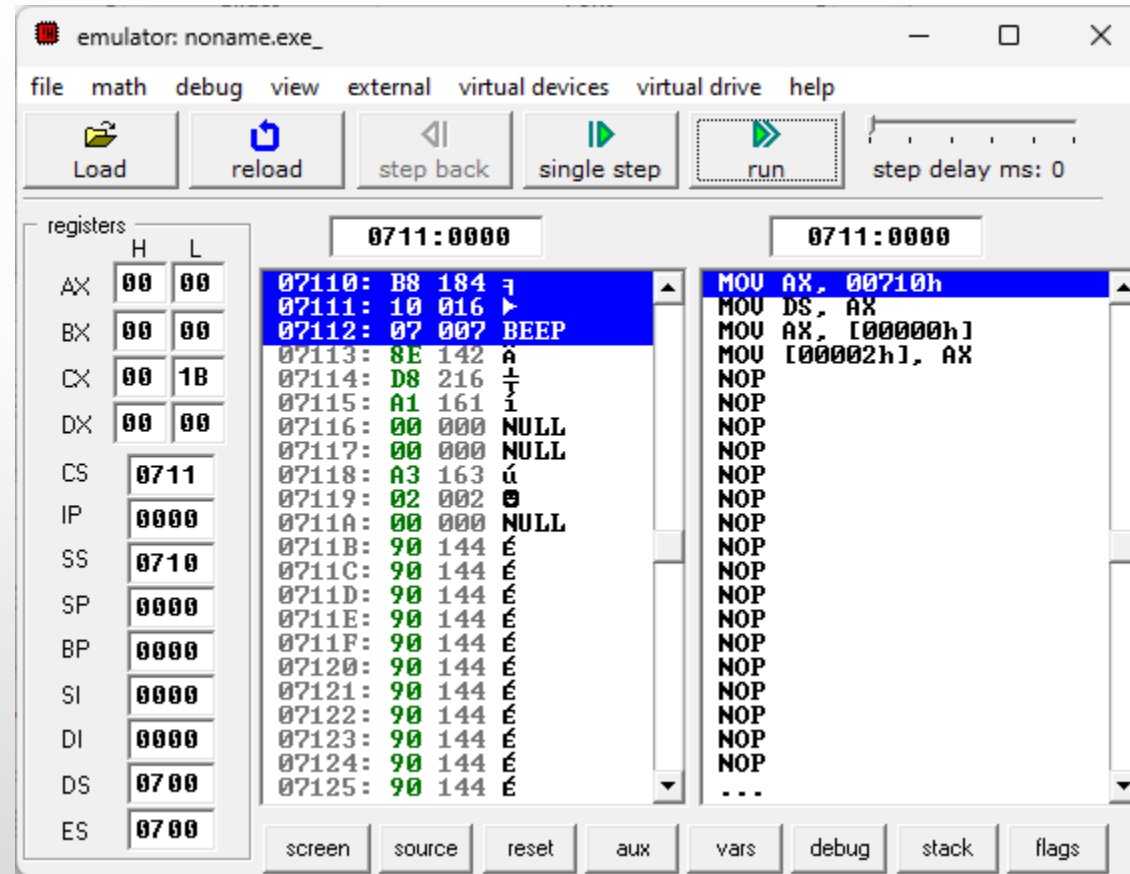
MOV value2, AX ; AX yazmacındaki değeri value2 değişkenine taşı

END START

CODE ENDS

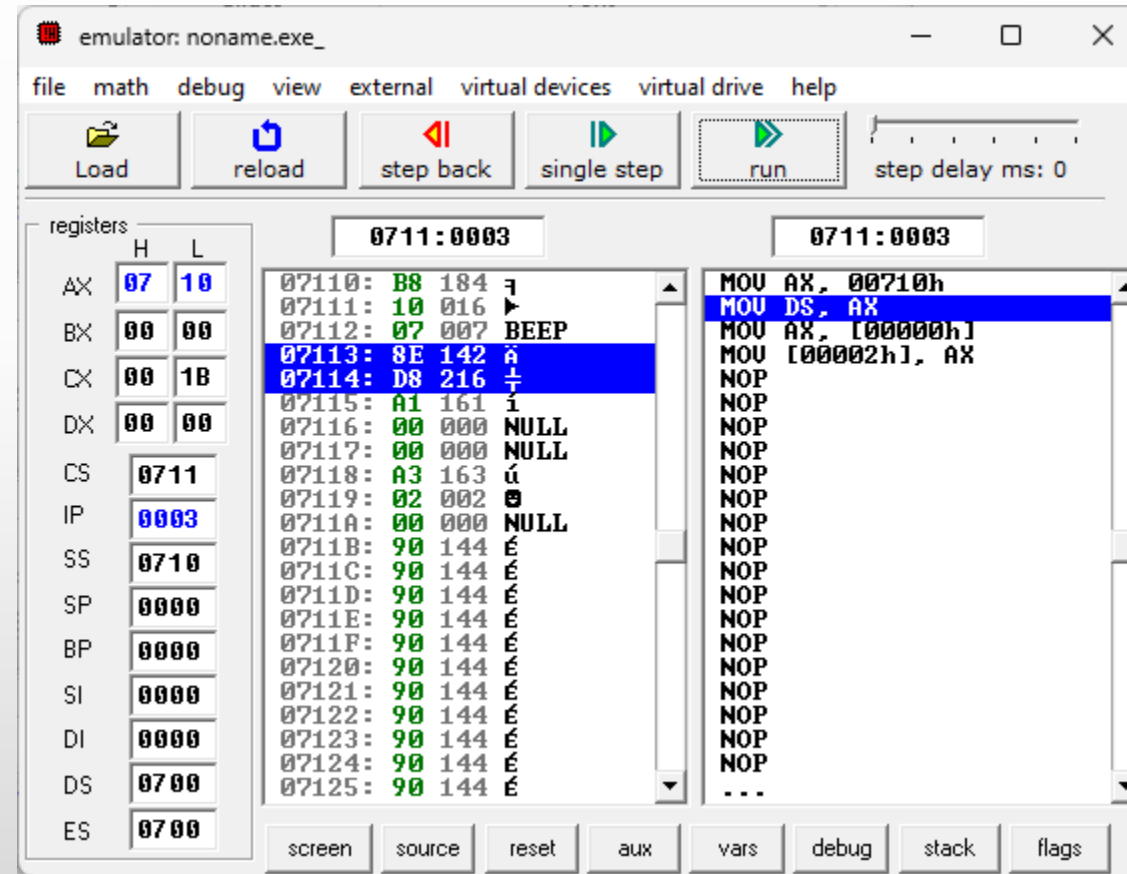


Değişkenler Arasında Veri Taşıma



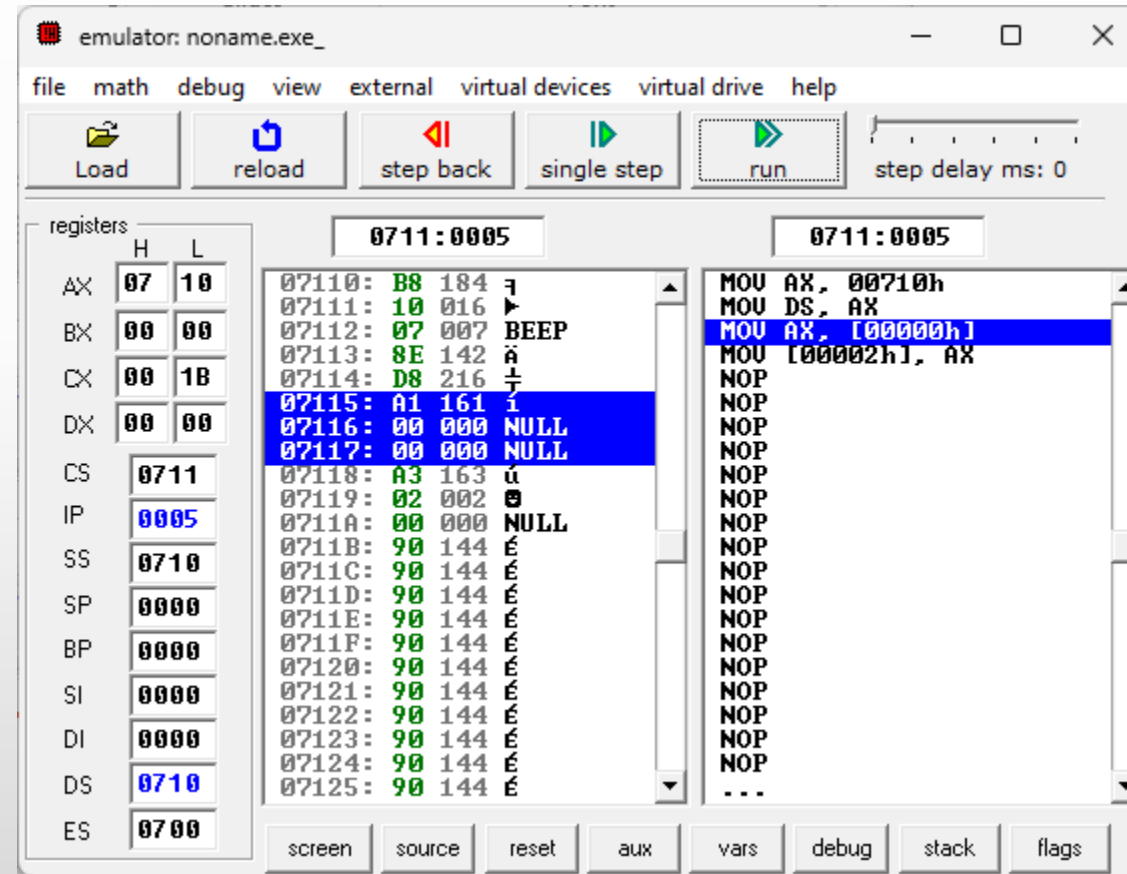


Değişkenler Arasında Veri Taşıma



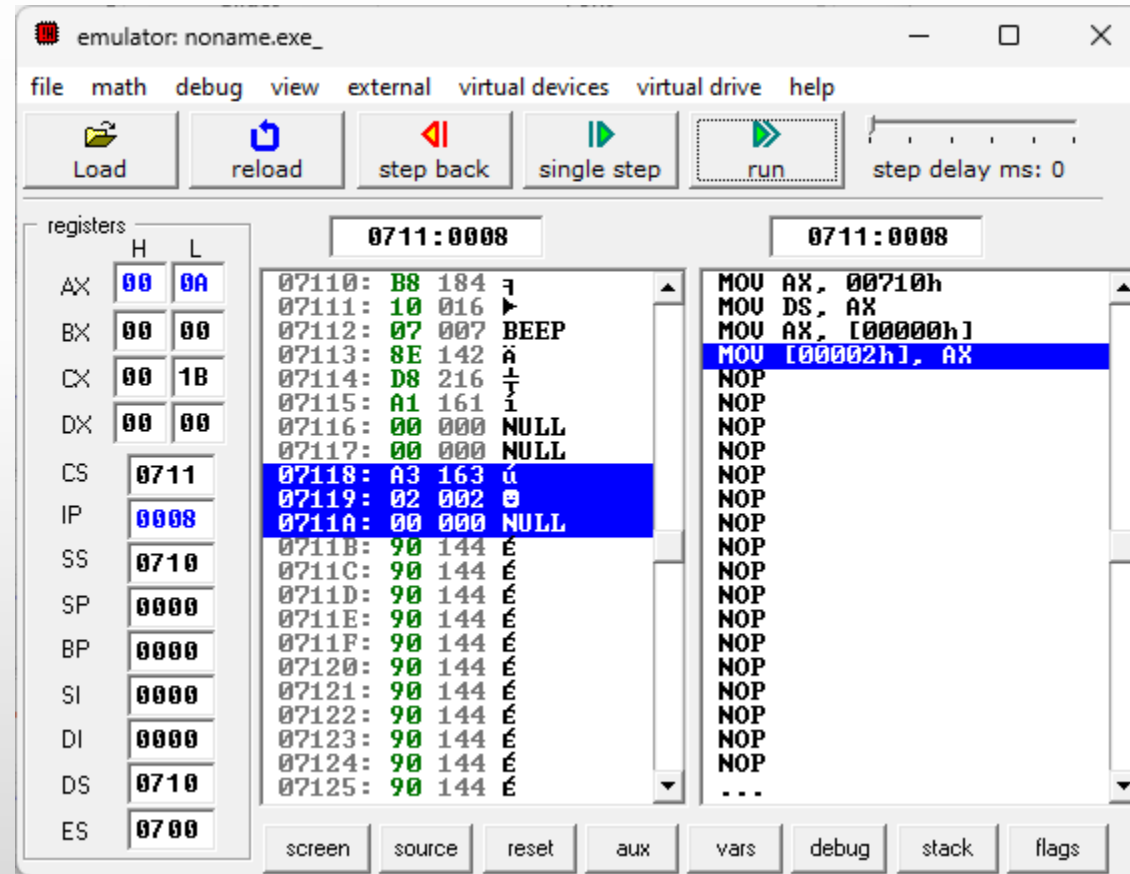


Değişkenler Arasında Veri Taşıma





Değişkenler Arasında Veri Taşıma





Değişkenleri Toplama İşlemi

DATA SEGMENT

value1 DW 105h ; İki byte'lık değişken tanımla, başlangıç değeri 105h

value2 DW 103h ; İkinci bir değişken tanımla

result DW ? ; Sonucu saklayacak değişken, başlangıç değeri belirsiz (?)

DATA ENDS

START:

MOV AX, DATA ; DATA kesiminin adresini AX yazmacına yükle

MOV DS, AX ; DS yazmacına DATA kesiminin adresini yükle

MOV AX, value1 ; value1 değişkeninin değerini AX yazmacına yükle

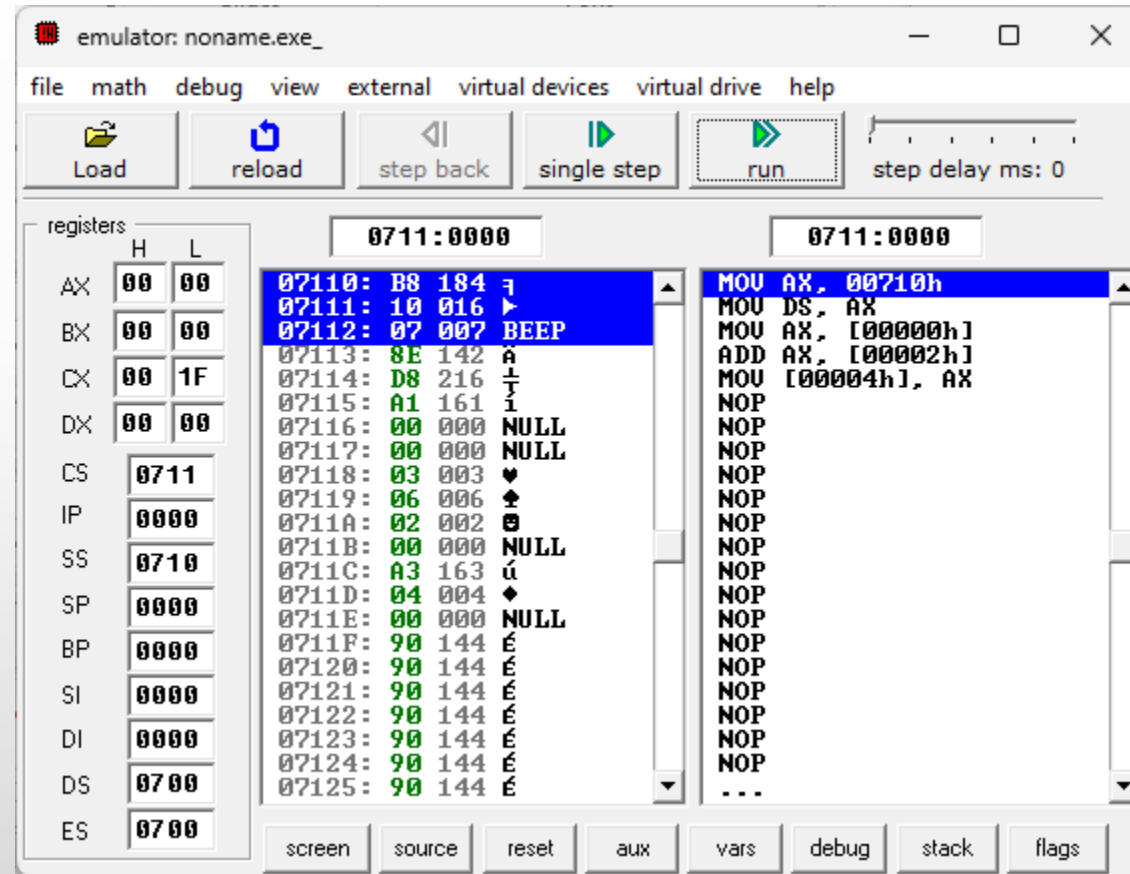
ADD AX, value2 ; value2 değişkeninin değerini AX yazmacındaki değere ekle

MOV result, AX ; AX yazmacındaki değeri result değişkenine taşı

END START

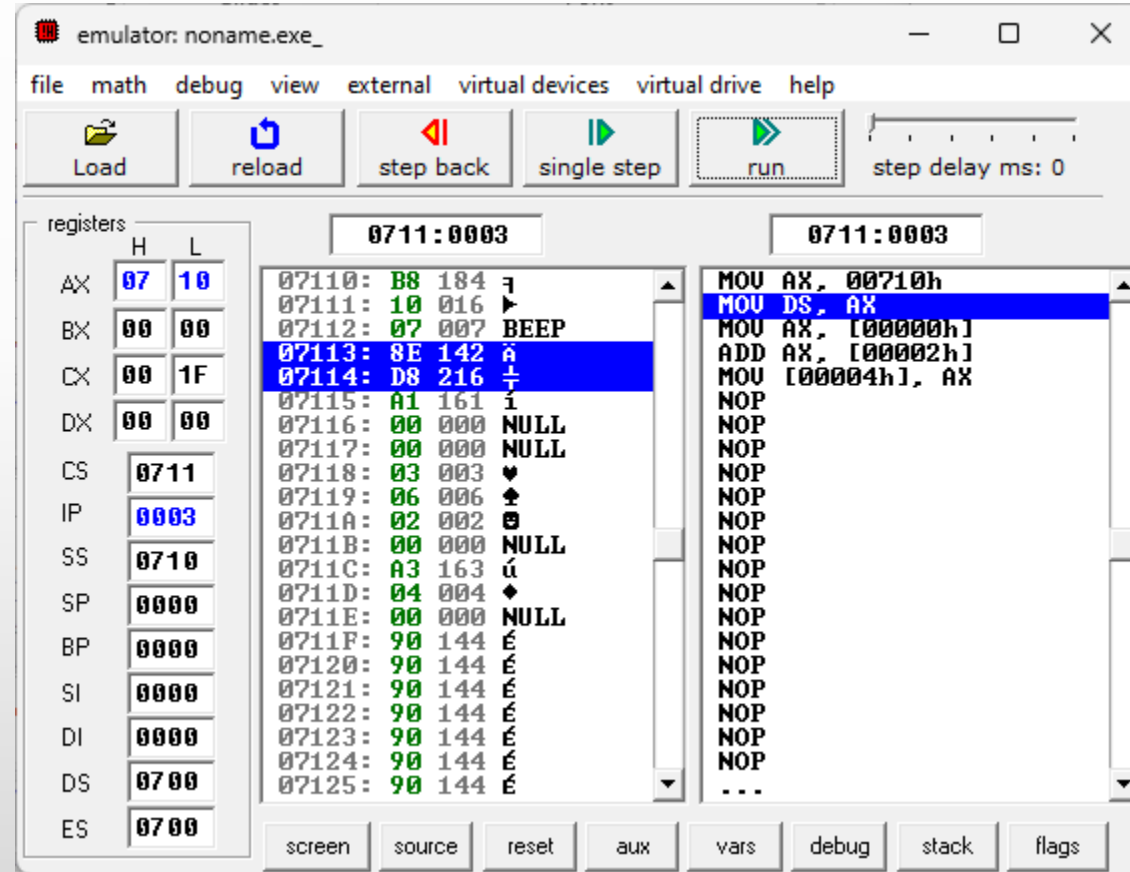


Değişkenleri Toplama İşlemi



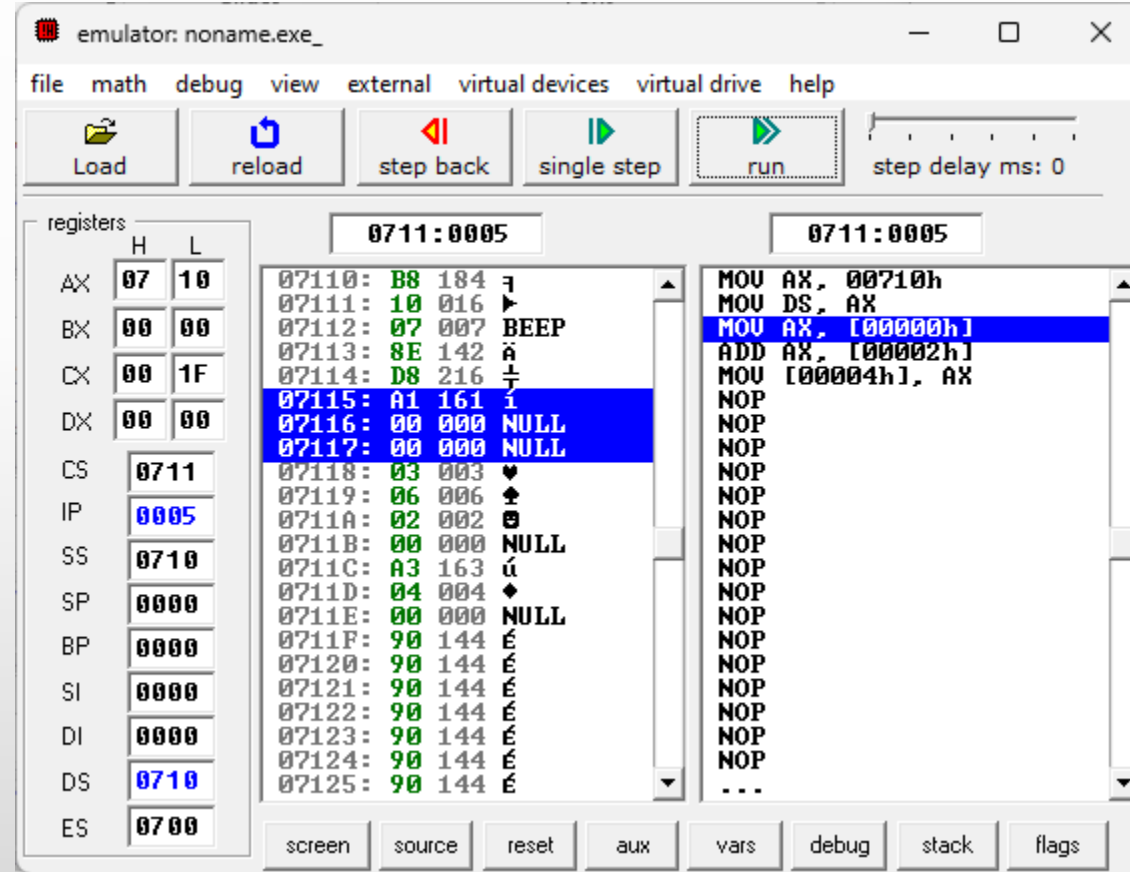


Değişkenleri Toplama İşlemi



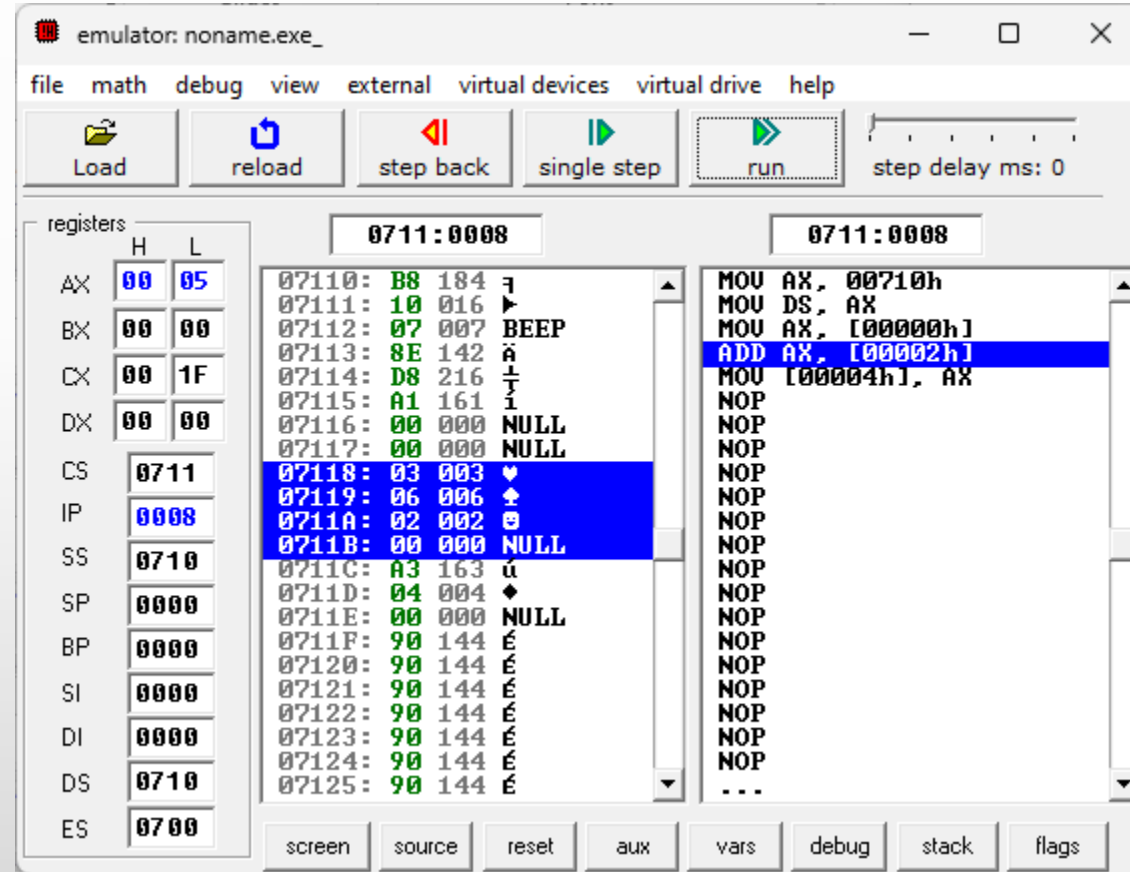


Değişkenleri Toplama İşlemi



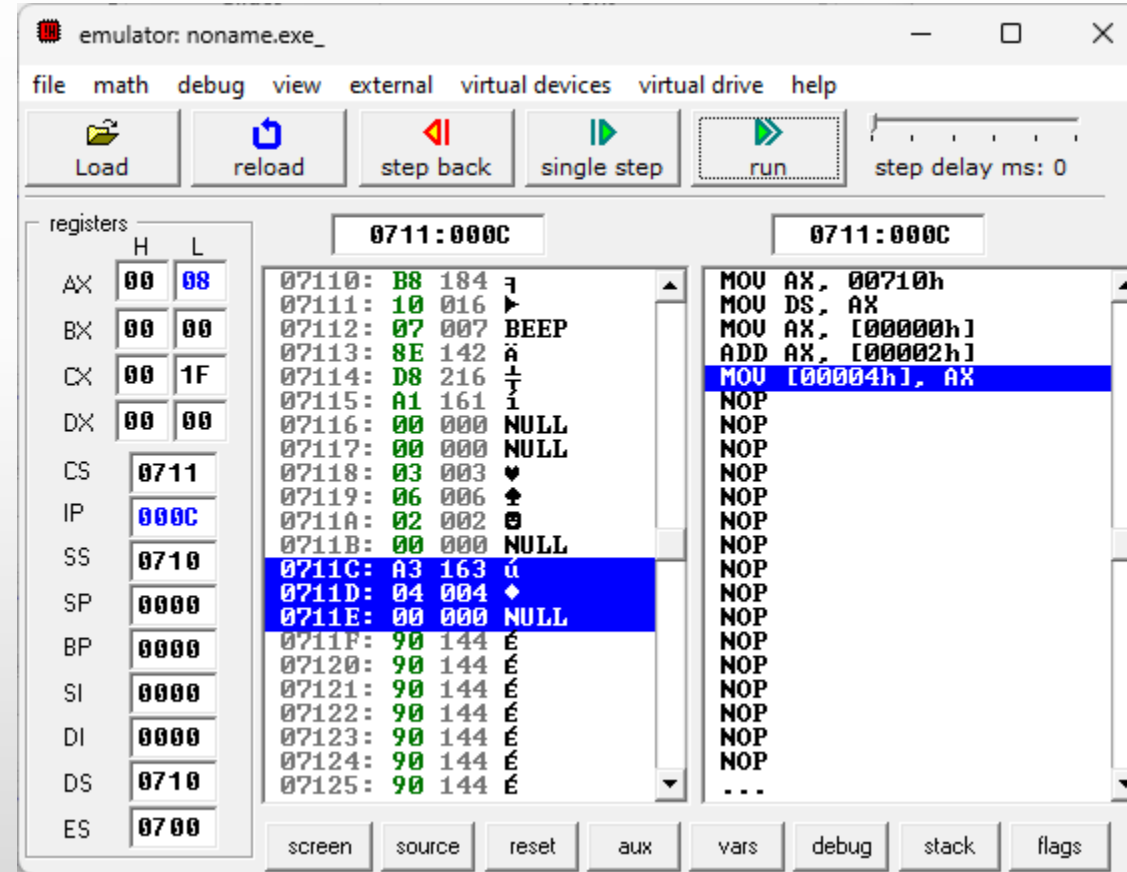


Değişkenleri Toplama İşlemi





Değişkenleri Toplama İşlemi







İki Değişkenin Değerini Karşılaştırma

DATA SEGMENT

value1 DW 10 ; Birinci değişken, başlangıç değeri 10

value2 DW 5 ; İkinci değişken, başlangıç değeri 5

max_value DW ? ; Büyük değeri tutacak değişken, ilk değeri belirsiz (?)

DATA ENDS

ASSUME DS:DATA ; DATA kesiminin adresini DS yazmacına ata



İki Değişkenin Değerini Karşılaştırma

START:

```
MOV AX, DATA    ; DATA kesiminin adresini AX yazmacına yükle
MOV DS, AX        ; DS yazmacına DATA kesiminin adresini yükleme
MOV AX, value1    ; Birinci değişkenin değerini AX yazmacına yükle
CMP AX, value2    ; AX yazmacındaki değeri ikinci değişken ile karşılaştır
JGE greater_or_equal ; value1, value2'den büyük veya eşitse atla
MOV AX, value2    ; İkinci değişkenin değerini AX yazmacına yükle
JMP store_max     ; Karşılaştırma sonrası büyük değeri saklayan kısma atla
```

greater_or_equal:

```
MOV AX, value1    ; Birinci değişkenin değerini AX yazmacına yükle
```

store_max:

```
MOV max_value, AX ; AX yazmacındaki değeri max_value değişkenine taşı
```

END START



İki Değişkenin Değerini Karşılaştırma

emulator: noname.exe_

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	00
BX	00	00
CX	00	1F
DX	00	00
CS	0710	
IP	0006	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

0710:0006

```
07106: B8 184 7
07107: 10 016 7
07108: 07 007 BEEP
07109: 8E 142 A
0710A: D8 216 1
0710B: A1 161 i
0710C: 00 000 NULL
0710D: 00 000 NULL
0710E: 3B 059 ;
0710F: 06 006 1
07110: 02 002 0
07111: 00 000 NULL
07112: 7D 125 >
07113: 05 005 1
07114: A1 161 i
07115: 02 002 0
07116: 00 000 NULL
07117: EB 235 6
07118: 03 003 v
07119: A1 161 i
0711A: 00 000 NULL
0711B: 00 000 NULL
```

0710:0006

```
MOV AX, 00710h
MOV DS, AX
MOV AX, [00000h]
CMP AX, [00002h]
JNL 019h
MOV AX, [00002h]
JMP 01Ch
MOV AX, [00000h]
MOV [00004h], AX
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

screen source reset aux vars debug stack flags



İki Değişkenin Değerini Karşılaştırma

emulator: noname.exe_

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	07	10
BX	00	00
CX	00	1F
DX	00	00
CS	0710	
IP	0009	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

0710:0009

```
07106: B8 184 3
07107: 10 016 3
07108: 07 007 BEEP
07109: 8E 142 ä
0710A: D8 216 ÷
0710B: A1 161 í
0710C: 00 000 NULL
0710D: 00 000 NULL
0710E: 3B 059 ;
0710F: 06 006 ±
07110: 02 002 8
07111: 00 000 NULL
07112: 7D 125 >
07113: 05 005 ±
07114: A1 161 í
07115: 02 002 8
07116: 00 000 NULL
07117: EB 235 δ
07118: 03 003 ♥
07119: A1 161 í
0711A: 00 000 NULL
0711B: 00 000 NULL
```

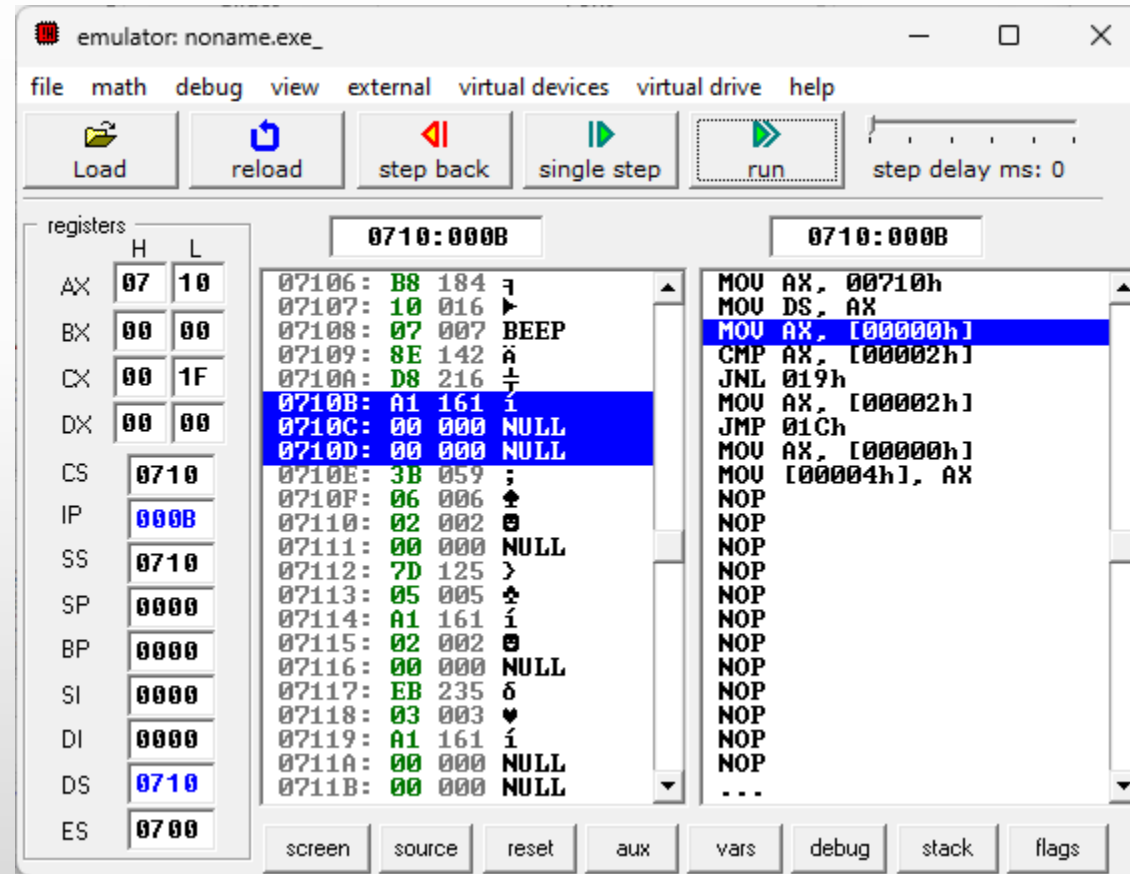
0710:0009

```
MOV AX, 00710h
MOV DS, AX
MOV AX, [00000h]
CMP AX, [00002h]
JNL 019h
MOV AX, [00002h]
JMP 01Ch
MOV AX, [00000h]
MOV [00004h], AX
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

screen source reset aux vars debug stack flags



İki Değişkenin Değerini Karşılaştırma





İki Değişkenin Değerini Karşılaştırma

emulator: noname.exe_

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	0A
BX	00	00
CX	00	1F
DX	00	00
CS	0710	
IP	000E	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0710	
ES	0700	

0710:000E

```
07106: B8 184 3 MOV AX, 00710h
07107: 10 016 3 MOV DS, AX
07108: 07 007 BEEP MOV AX, [00000h]
07109: 8E 142 3 CMP AX, [00002h]
0710A: D8 216 3 JNL 019h
0710B: A1 161 3 MOV AX, [00002h]
0710C: 00 000 NULL JMP 01Ch
0710D: 00 000 NULL MOV AX, [00000h]
0710E: 3B 059 3 MOV [00004h], AX
0710F: 06 006 3 NOP
07110: 02 002 3 NOP
07111: 00 000 NULL NOP
07112: 7D 125 3 NOP
07113: 05 005 3 NOP
07114: A1 161 3 NOP
07115: 02 002 3 NOP
07116: 00 000 NULL NOP
07117: EB 235 3 NOP
07118: 03 003 3 NOP
07119: A1 161 3 NOP
0711A: 00 000 NULL NOP
0711B: 00 000 NULL ...
```

screen source reset aux vars debug stack flags



İki Değişkenin Değerini Karşılaştırma

emulator: noname.exe_

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	0A
BX	00	00
CX	00	1F
DX	00	00
CS	0710	
IP	0012	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0710	
ES	0700	

0710:0012

```
07106: B8 184 3 MOV AX, 00710h
07107: 10 016 3 MOV DS, AX
07108: 07 007 BEEP MOV AX, [00000h]
07109: 8E 142 3 CMP AX, [00002h]
0710A: D8 216 3 JNL 019h
0710B: A1 161 3 MOV AX, [00002h]
0710C: 00 000 NULL JMP 01Ch
0710D: 00 000 NULL MOV AX, [00000h]
0710E: 3B 059 3 MOV [00004h], AX
0710F: 06 006 3 NOP
07110: 02 002 3 NOP
07111: 00 000 NULL NOP
07112: 7D 125 3 NOP
07113: 05 005 3 NOP
07114: A1 161 3 NOP
07115: 02 002 3 NOP
07116: 00 000 NULL NOP
07117: EB 235 3 NOP
07118: 03 003 3 NOP
07119: A1 161 3 NOP
0711A: 00 000 NULL NOP
0711B: 00 000 NULL ...
```

screen source reset aux vars debug stack flags





İki Değişkenin Değerini Karşılaştırma

emulator: noname.exe_

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	0A
BX	00	00
CX	00	1F
DX	00	00
CS	0710	
IP	001C	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0710	
ES	0700	

0710:001C

0711C:	A3	163	ü
0711D:	04	004	♦
0711E:	00	000	NULL
0711F:	90	144	é
07120:	90	144	é
07121:	90	144	é
07122:	90	144	é
07123:	90	144	é
07124:	90	144	é
07125:	90	144	é
07126:	90	144	é
07127:	90	144	é
07128:	90	144	é
07129:	90	144	é
0712A:	90	144	é
0712B:	90	144	é
0712C:	90	144	é
0712D:	90	144	é
0712E:	90	144	é
0712F:	90	144	é
07130:	90	144	é
07131:	90	144	é

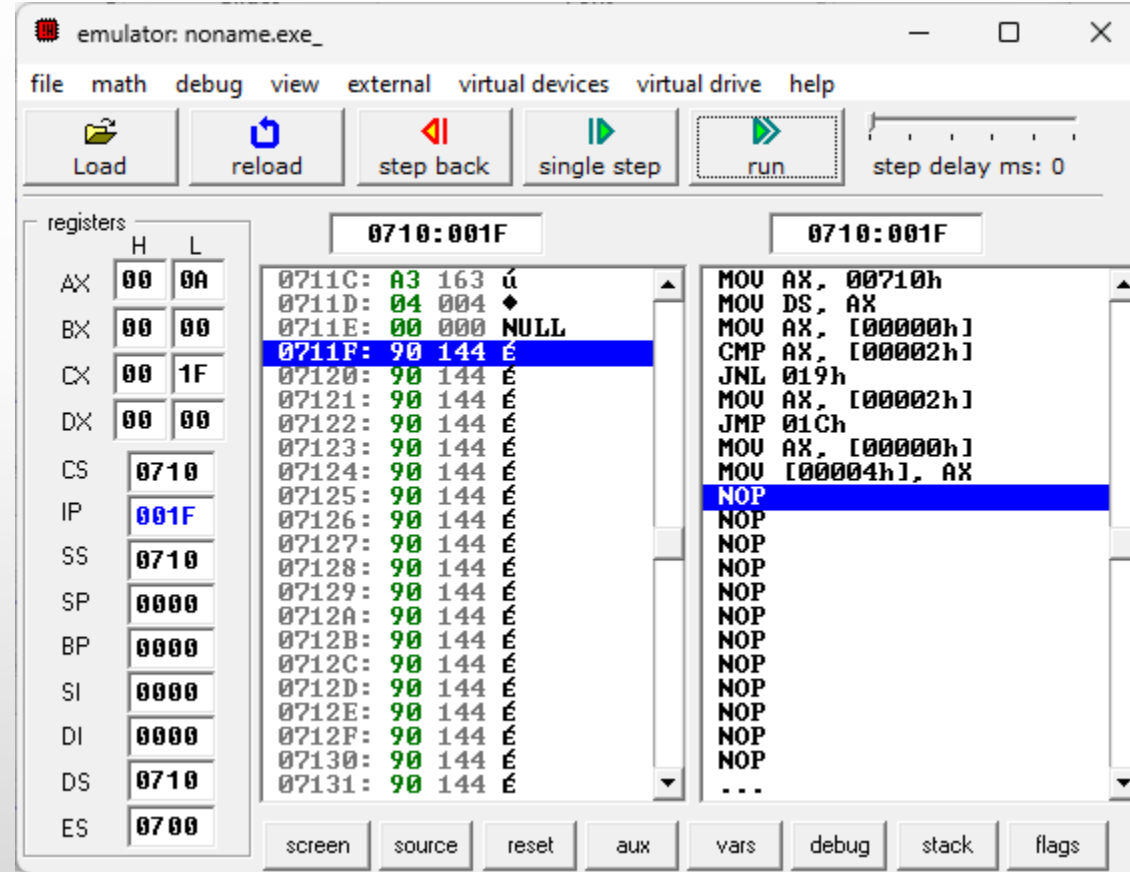
0710:0019

```
MOV AX, 00710h
MOV DS, AX
MOV AX, [00000h]
CMP AX, [00002h]
JNL 019h
MOV AX, [00002h]
JMP 01Ch
MOV AX, [00000h]
MOV [00004h], AX
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

screen source reset aux vars debug stack flags



İki Değişkenin Değerini Karşılaştırma





SON