Bölüm 7: Akış Kontrol

Mikroişlemciler

Program Akış Kontrolü

- Koşulsuz atlamalar (*jumps*)
- Koşullu atlamalar (*jumps*)
  - Tek bir bayrağı test eden atlama komutları
  - İşaretli sayılar için atlama komutları
  - İşaretsiz sayılar için atlama komutları
- Döngüler (l*oops*)

Koşulsuz Atlamalar

- JMP (Jump): program içinde kontrolü başka bir noktaya aktarır.
  - JMP etiket
- Etiket tanımlamak için adı yazılır ve sonuna ":" eklenir.
- Etiket herhangi bir karakter kombinasyonu olabilir,
  - ancak bir sayı ile başlayamaz.
- Etiket, ayrı bir satırda veya başka bir komutun önünde tanımlanabilir.
- JMP, kontrolü hem ileri hem de geri yönlendirebilir.
- Mevcut kod segmenti içinde (65,535 bayt) herhangi bir yere atlayabilir.

Koşulsuz Atlamalar

```
org     100h
mov     ax, 5           ; set ax to 5.
mov     bx, 2           ; set bx to 2.
jmp     calc            ; go to 'calc'.
back:   jmp stop        ; go to 'stop'.
calc:
add     ax, bx          ; add bx to ax.
jmp     back            ; go 'back'.
stop:
ret                     ; return to operating system.
```

Tek Bir Bayrağı Test Eden Atlama Komutları

Instruction	Description	Condition
JZ , JE	Jump if Zero (Equal).	ZF = 1
JC , JB, JNAE	Jump if Carry (Below, Not Above Equal).	CF = 1
JS	Jump if Sign.	SF = 1
JO	Jump if Overflow.	OF = 1
JPE, JP	Jump if Parity Even.	PF = 1
JNZ , JNE	Jump if Not Zero (Not Equal).	ZF = 0
JNC , JNB, JAE	Jump if Not Carry (Not Below, Above Equal).	CF = 0
JNS	Jump if Not Sign.	SF = 0
JNO	Jump if Not Overflow.	OF = 0
JPO, JNP	Jump if Parity Odd (No Parity).	PF = 0

Tek Bir Bayrağı Test Eden Atlama Komutları

- JZ, JE: Sıfıra eşitse atlama yap.
  - Koşul: ZF = 1 Zıt Komut: JNZ, JNE
- JC, JB, JNAE: Taşma durumunda atlama yap.
  - Koşul: CF = 1 Zıt Komut: JNC, JNB, JAE
- JS: Negatifse atlama yap.
  - Koşul: SF = 1 Zıt Komut: JNS
- JO: Taşma durumunda atlama yap.
  - Koşul: OF = 1 Zıt Komut: JNO
- JPE, JP: Çiftlik durumunda atlama yap.
  - Koşul: PF = 1 Zıt Komut: JPO

Tek Bir Bayrağı Test Eden Atlama Komutları

- Atlama komutları sabit uzunluktadır (iki bayt).
- Bağıl konum (*Offset*) 1 baytta saklanır.
  - -128 bayt geriye veya 127 bayt ileriye atlama yapabilir.
- Değer her zaman işaretli bir sayıdır.
- JE, JZ; JNE, JNZ ile aynı makine koduna derlenir.
- JC, JB, JNAE; JNC, JNB, JAE ile aynı makine koduna derlenir.

Tek Bir Bayrağı Test Eden Atlama Komutları

```
jnc a
jnb a
jae a

mov ax, 4
a: mov ax, 5
ret
```

İşaretli Sayılar İçin Atlama Komutları

Instruction	Description	Condition
JE , JZ	Jump if Equal (=). Jump if Zero.	ZF = 1
JNE , JNZ	Jump if Not Equal (<>). Jump if Not Zero.	ZF = 0
JG , JNLE	Jump if Greater (>). Jump if Not Less or Equal (not <=).	ZF = 0 and SF = OF
JL , JNGE	Jump if Less (<). Jump if Not Greater or Equal (not >=).	SF <> OF
JGE , JNL	Jump if Greater or Equal (>=). Jump if Not Less (not <).	SF = OF
JLE , JNG	Jump if Less or Equal (<=). Jump if Not Greater (not >).	ZF = 1 or SF <> OF

İşaretli Sayılar İçin Atlama Komutları

- JE, JZ: Eşitse atlama yap.
  - Koşul: ZF = 1 Zıt Komut: JNE, JNZ
- JNE, JNZ: Eşit değilse atlama yap.
  - Koşul: ZF = 0 Zıt Komut: JE, JZ
- JG, JNLE: Büyükse atlama yap.
  - Koşul: ZF = 0 ve SF = OF Zıt Komut: JNG, JLE
- JL, JNGE: Küçükse atlama yap.
  - Koşul: SF ≠ OF Zıt Komut: JNL, JGE
- JGE, JNL: Büyük veya eşitse atlama yap.
  - Koşul: SF = OF Zıt Komut: JNGE, JL
- JLE, JNG: Küçük veya eşitse atlama yap.
  - Koşul: ZF = 1 veya SF ≠ OF Zıt Komut: JNLE, JG

İşaretli Sayılar İçin Atlama Komutları

- <> işareti eşit değil anlamına gelir.

```
mov ax, 5
mov bx, 5
cmp ax, bx
je  equal_message
jmp not_equal_message
equal_message:          ; Eşitse yapılacak işlemler
jmp end_program
not_equal_message:      ; Eşit değilse yapılacak işlemler
end_program:
```

İşaretsiz Sayılar İçin Atlama Komutları

Instruction	Description	Condition
JE , JZ	Jump if Equal (=). Jump if Zero.	ZF = 1
JNE , JNZ	Jump if Not Equal (<>). Jump if Not Zero.	ZF = 0
JA , JNBE	Jump if Above (>). Jump if Not Below or Equal (not <=).	CF = 0 and ZF = 0
JB , JNAE, JC	Jump if Below (<). Jump if Not Above or Equal (not >=). Jump if Carry.	CF = 1
JAE , JNB, JNC	Jump if Above or Equal (>=). Jump if Not Below (not <). Jump if Not Carry.	CF = 0
JBE , JNA	Jump if Below or Equal (<=). Jump if Not Above (not >).	CF = 1 or ZF = 1

İşaretsiz Sayılar İçin Atlama Komutları

- JE, JZ: Eşitse atlama yap.
  - Koşul: ZF = 1 Zıt Komut: JNE, JNZ
- JNE, JNZ: Eşit değilse atlama yap.
  - Koşul: ZF = 0 Zıt Komut: JE, JZ
- JA, JNBE: Büyükse atlama yap.
  - Koşul: CF = 0 ve ZF = 0 Zıt Komut: JNA, JBE
- JB, JNAE, JC: Küçükse atlama yap.
  - Koşul: CF = 1 Zıt Komut: JNB, JAE, JNC
- JAE, JNB, JNC: Büyük veya eşitse atlama yap.
  - Koşul: CF = 0 Zıt Komut: JNAE, JB
- JBE, JNA: Küçük veya eşitse atlama yap.
  - Koşul: CF = 1 veya ZF = 1 Zıt Komut: JNBE, JA

İşaretsiz Sayılar İçin Atlama Komutları

```
mov ax, 5
mov bx, 7
cmp ax, bx
ja  jump_above
jmp not_jump_above
jump_above:      ; ax büyükse yapılacak işlemler
jmp end_program
not_jump_above:  ; ax küçükse veya eşitse yapılacak işlemler
end_program:
```

CMP ve Atlama Komutları

- Sayısal değerleri karşılaştırmak için CMP (compare) komutu kullanılır.
- CMP komutu, SUB (çıkarma) komutunu gerçekleştirir.
- Örnek 1: 5 ve 2'yi karşılaştır,
  - 5 - 2 = 3
  - Sonuç sıfır değil (Zero Bayrağına 0 atanır).
- Örnek 2: 7 ve 7'yi karşılaştır,
  - 7 - 7 = 0
  - Sonuç sıfır! (Zero Bayrağına 1 atanır, JZ veya JE atlama yapar).

CMP ve Atlama Komutları

```
include "emu8086.inc"
org     100h
mov     al, 25      ; set al to 25.
mov     bl, 10      ; set bl to 10.
cmp     al, bl      ; compare al - bl.
je      equal       ; jump if al = bl (zf = 1).
putc    'n'         ; if it gets here, then al <> bl,
jmp     stop        ; so print 'n', and jump to stop.
equal:              ; if gets here,
putc    'y'         ; then al = bl, so print 'y'.
stop:
ret                 ; gets here no matter what.
```

Döngüler (Loops)

Instruction	Operation And Jump Condition
LOOP	decrease cx, jump to label if cx not zero.
LOOPE	decrease cx, jump to label if cx not zero and equal (zf = 1).
LOOPNE	decrease cx, jump to label if cx not zero and not equal (zf = 0).
LOOPNZ	decrease cx, jump to label if cx not zero and zf = 0.
LOOPZ	decrease cx, jump to label if cx not zero and zf = 1.
JCXZ	jump to label if cx is zero.

Döngüler (Loops)

- Döngüler, bir koşula bağlı olarak bir kod bloğunun tekrarlanmasını sağlar.
- LOOP: CX sıfır olmadığı sürece belirtilen etikete atlama yapar.
- LOOPE, LOOPZ:
  - CX sıfır olmadığı ve ZF = 1 olduğu sürece etikete atlama yapar.
- LOOPNE, LOOPNZ :
  - CX sıfır olmadığı ve ZF = 0 olduğu sürece etikete atlama yapar.
- JCXZ: CX sıfır olduğunda belirtilen etikete atlama yapar.

Döngüler (Loops)

```
include "emu8086.inc"
org 100h
mov cx, 5                ; CX döngü tekrar sayısı 5 ata.
dongu:
   ; Döngü İçeriği
   loop dongu            ; CX sıfır değilse dongu etiketine atla
jmp dur                  ; Döngü bittiğinde dur etiketine atla
dur:
   ret
```

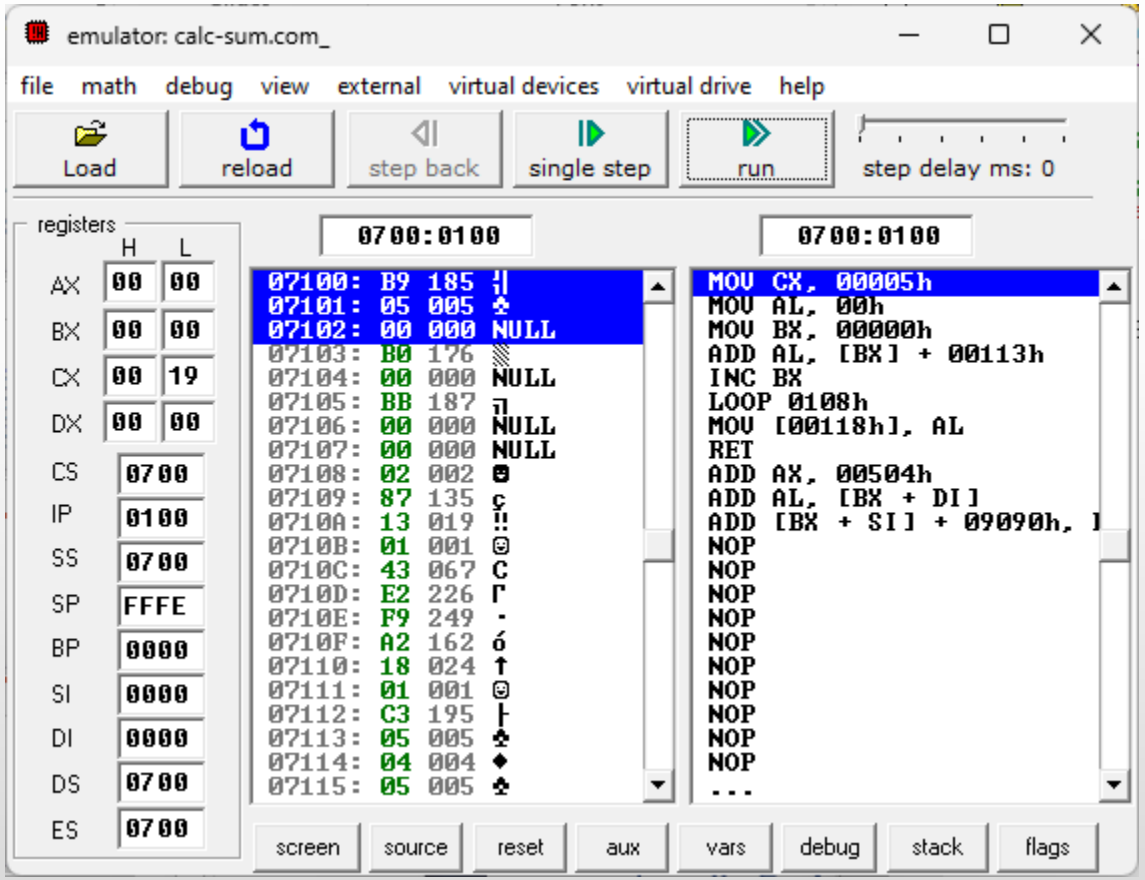
Döngüler (Loops)

```
1   org 100h
2   mov bx, 0   ;Toplam adım.
3   mov cx, 5
4   k1:
5    add bx, 1
6    mov al, '1'
7    mov ah, 0eh
8    int 10h
9    push cx
10  mov cx, 5
11  k2:
12    add bx, 1
13    mov al, '2'
14    mov ah, 0eh
15    int 10h
16    loop  k2   ; İç döngü.
17   pop cx
18 loop k1       ; Dış döngü.
19 ret
```

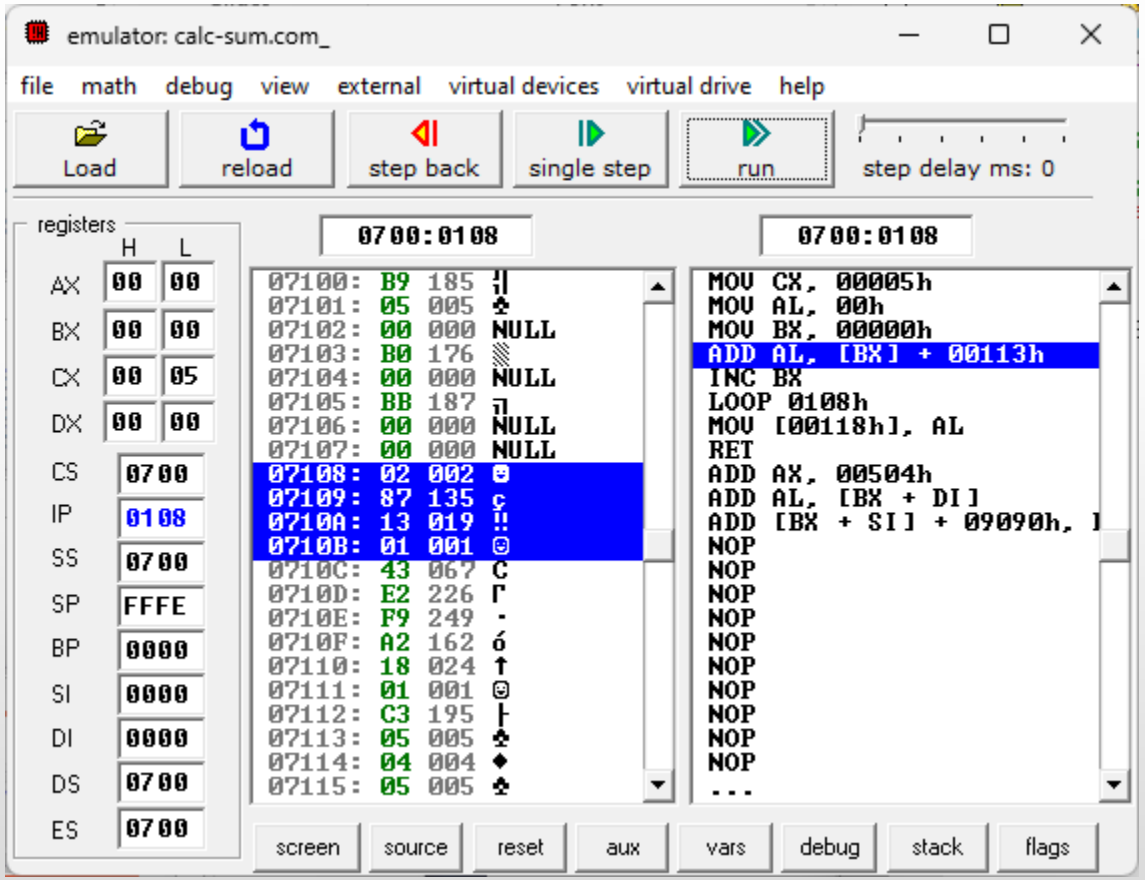
Dizi Elemanları Toplamı

```
org 100h
mov cx, 5 ; eleman sayısı
mov al, 0  ; toplam al yazmacında tutulacak
mov bx, 0 ; bx indis olarak kullanılacak
next: add al, vector[bx]  ; elemanları topla
inc bx   ; sonraki eleman
loop next   ; cx=0 olana kadar dön
mov m, al   ; sonucu m değişkenine atar
ret
vector db 5, 4, 5, 2, 1
m db 0
```

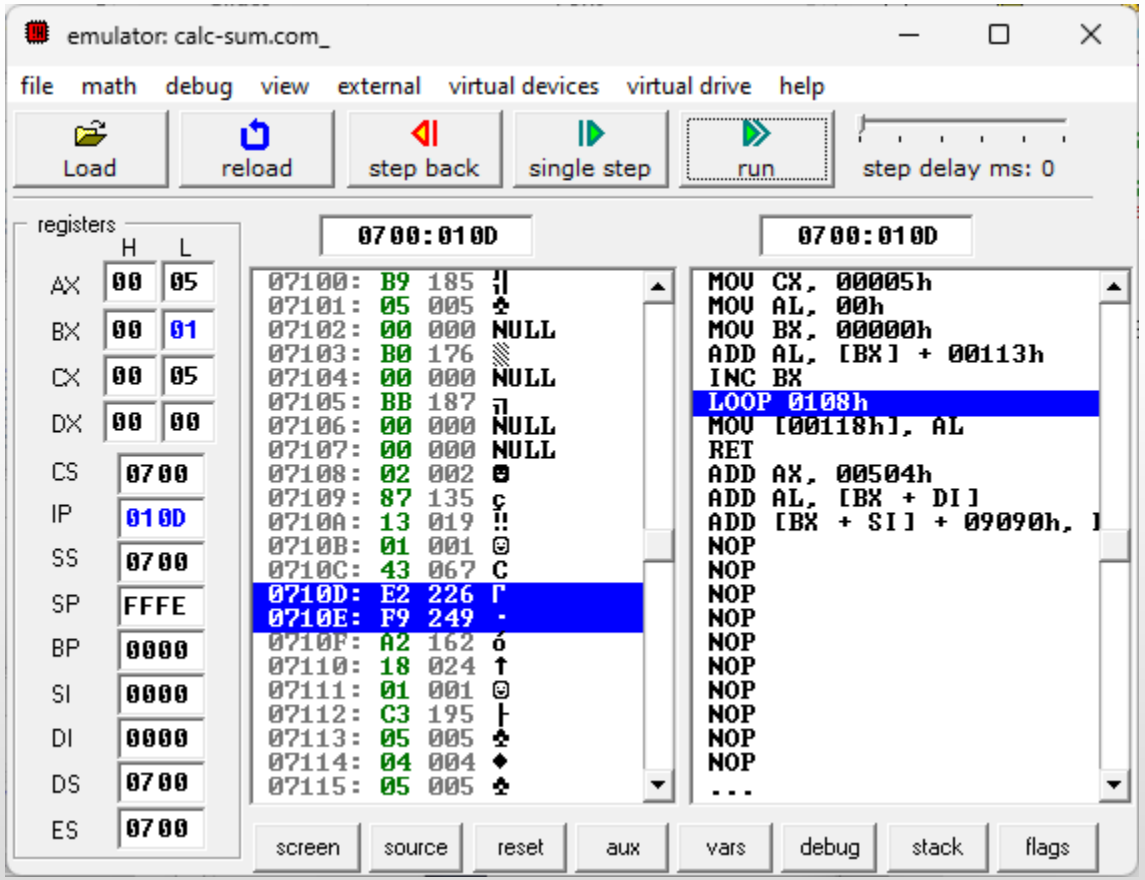
Dizi Elemanları Toplamı



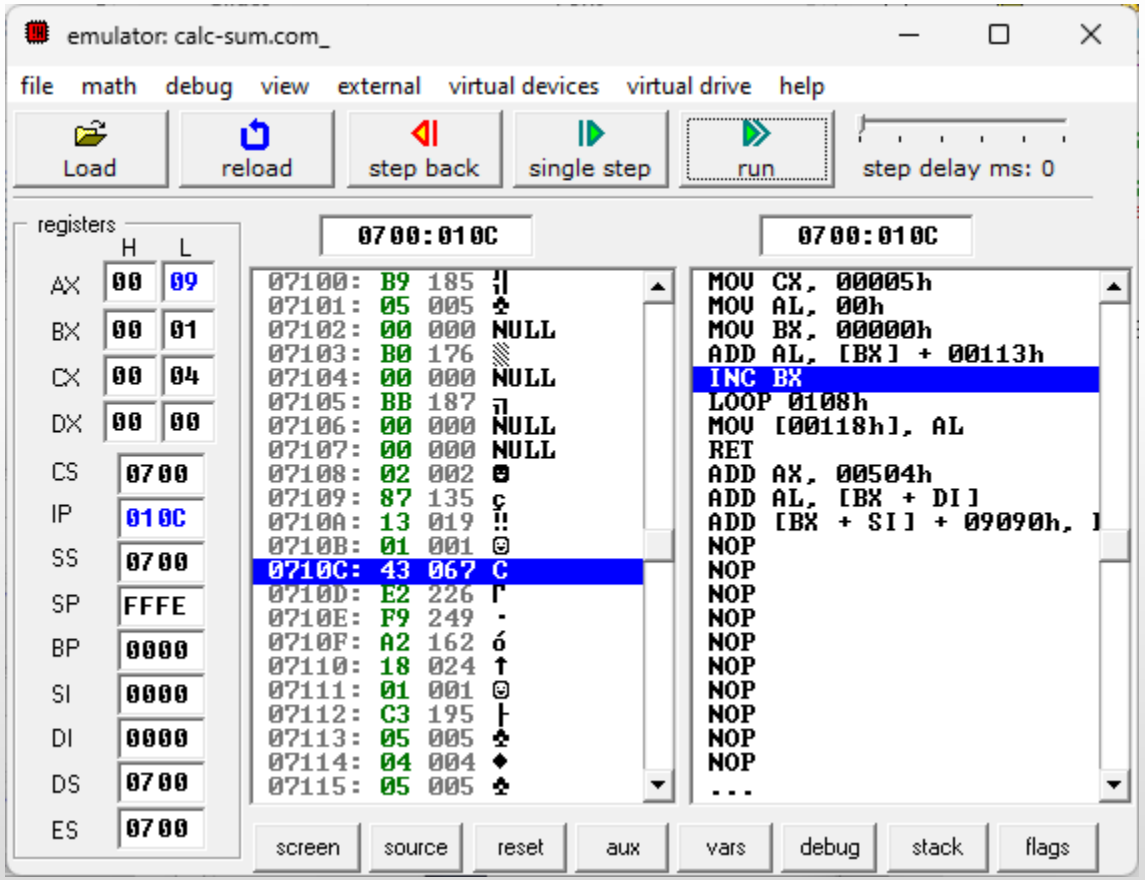
Dizi Elemanları Toplamı



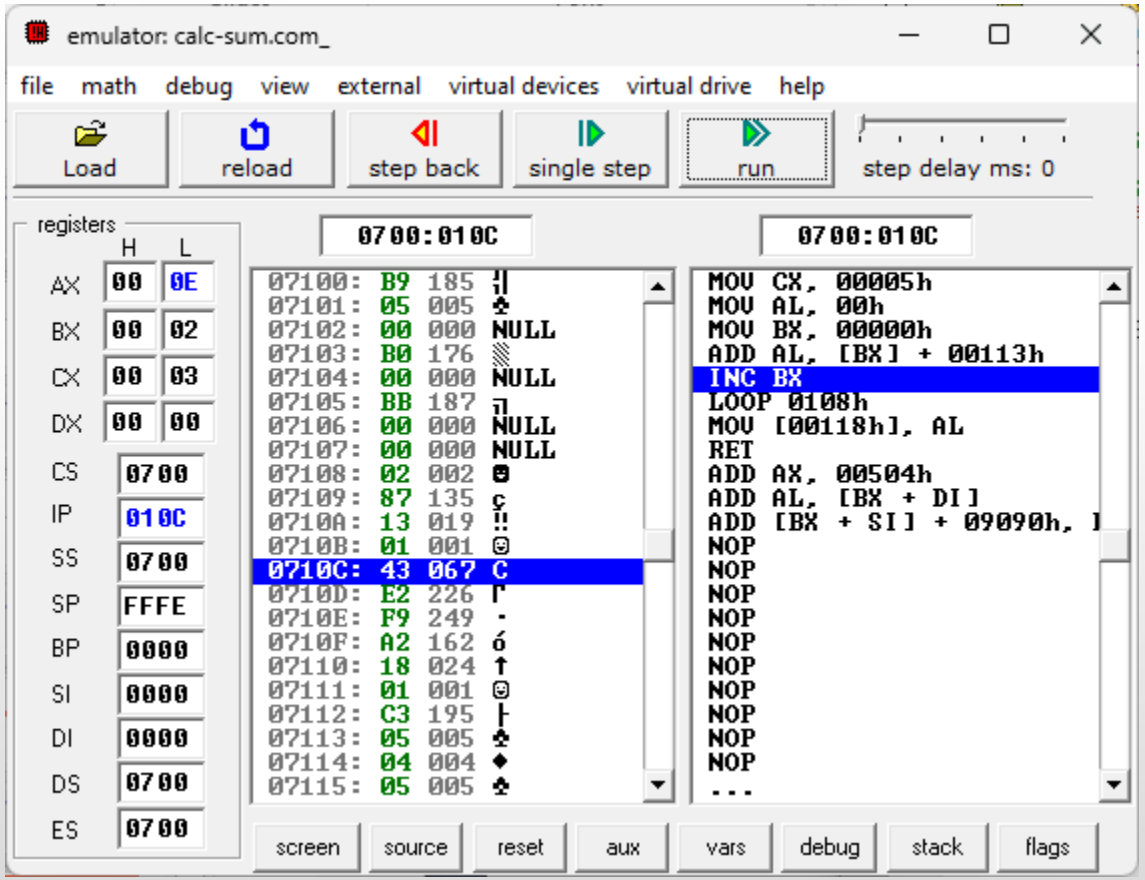
Dizi Elemanları Toplamı



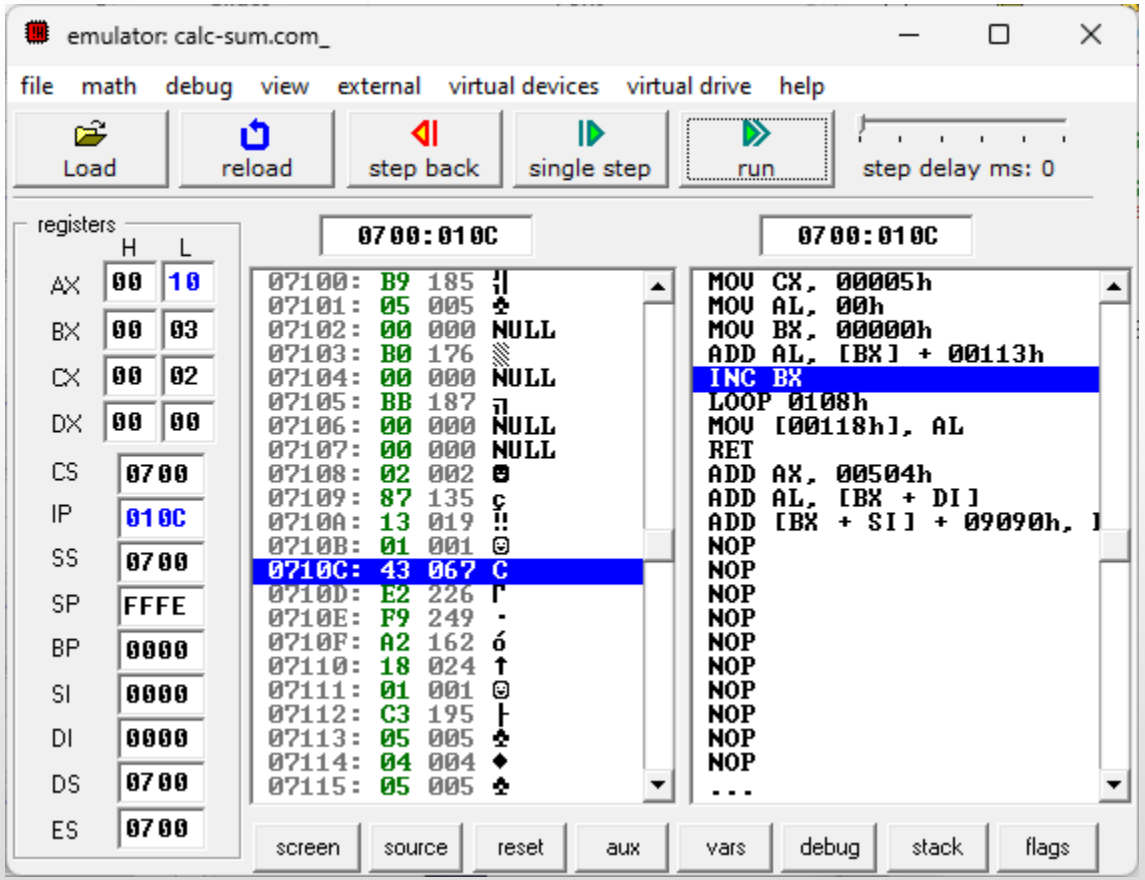
Dizi Elemanları Toplamı



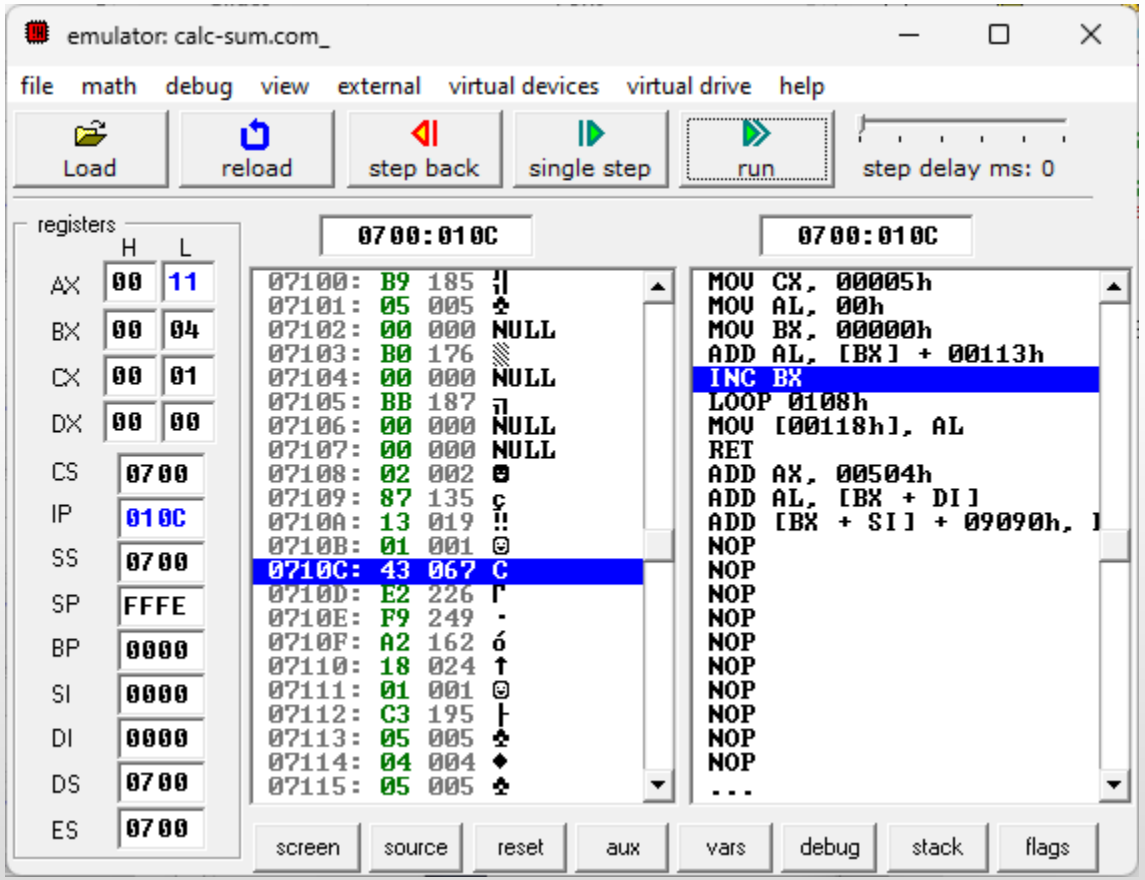
Dizi Elemanları Toplamı



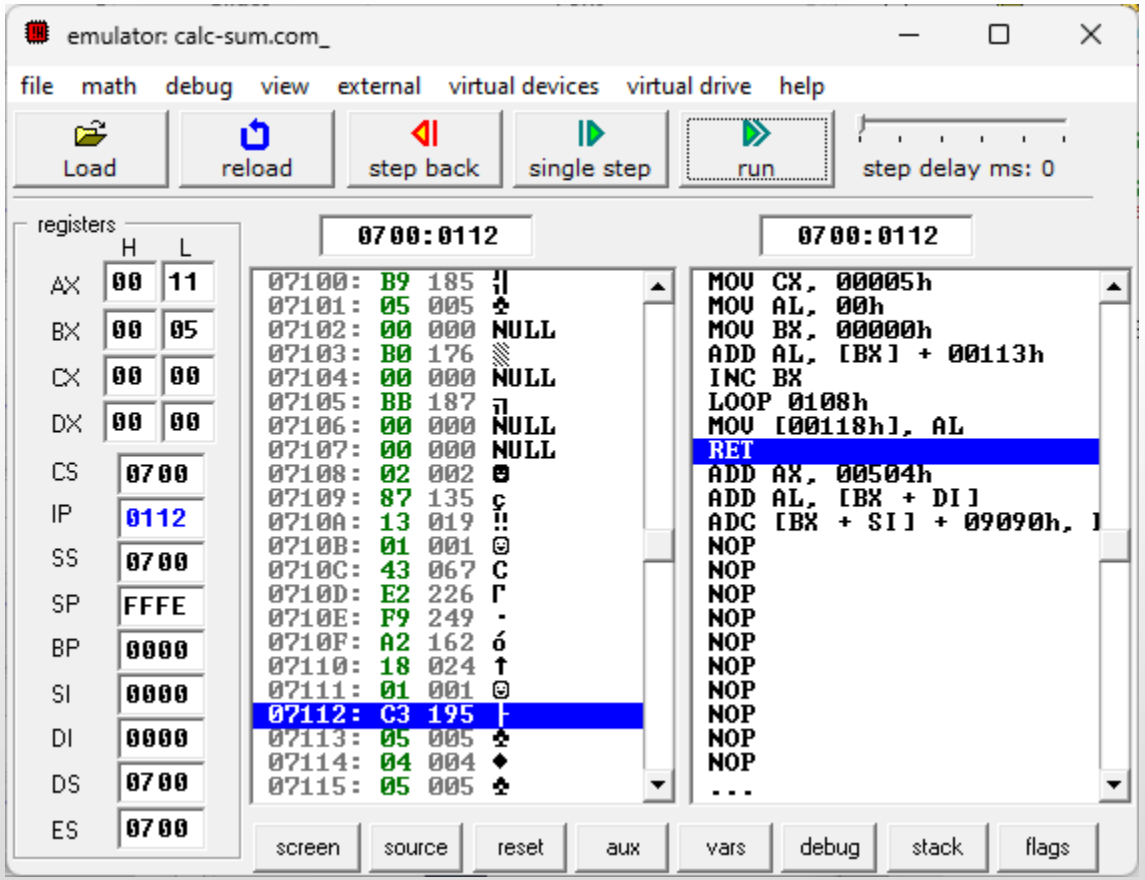
Dizi Elemanları Toplamı



Dizi Elemanları Toplamı



Dizi Elemanları Toplamı



SON