

Bölüm 7: Prosedürler

Mikroişlemciler

Prosedürler



- Belirli bir görevi gerçekleştiren kod parçacığı.
- Programın içinden çağrılır.
- Görevi tamamladığında genellikle çağrıldığı noktaya geri döner.
- Programı daha yapılandırılmış (structured) ve anlaşılır hale getirir.





```
name PROC
```

; Prosedür kodu buraya yazılır.

RET

name ENDP

name: Prosedürün adıdır. Başta ve sonda aynı olmalıdır.





- RET:
 - İşletim sistemine geri dönmek için kullanılır.
 - Aynı zamanda prosedürden dönmek için de kullanılır.
- PROC ve ENDP:
 - Derleyiciye prosedürün adresini hatırlatır.
 - Gerçek makine koduna çevrilmezler.
- CALL:
 - Bir prosedürü çağırmak için kullanılır.

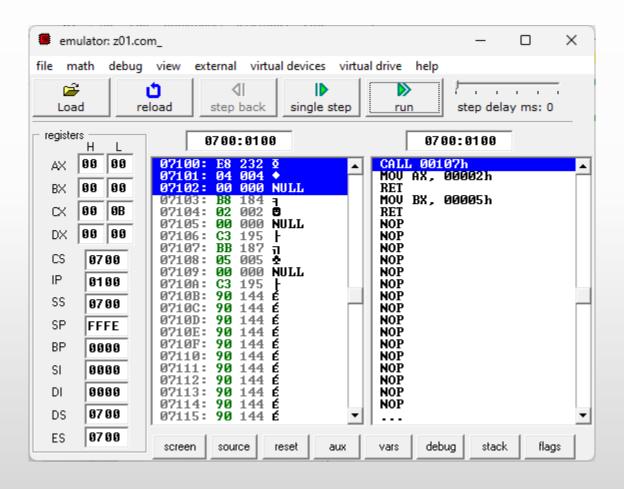




```
ORG
       100h
  CALL
         m1
        AX, 2
  MOV
                       ; return to operating system.
RET
       PROC
m1
  MOV
         BX, 5
                       ; return to caller.
RET
m1
       ENDP
END
```

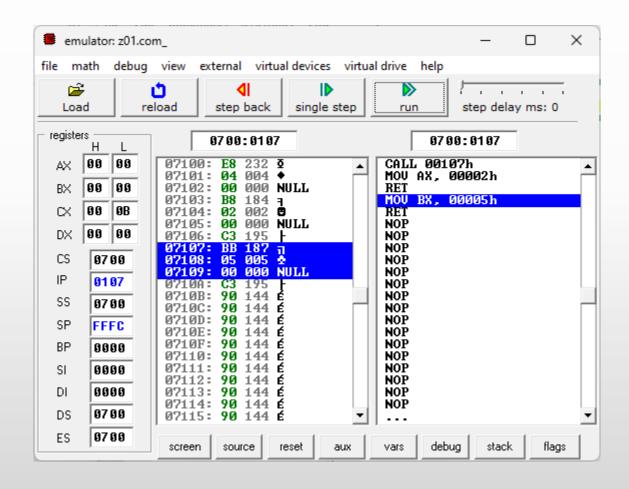






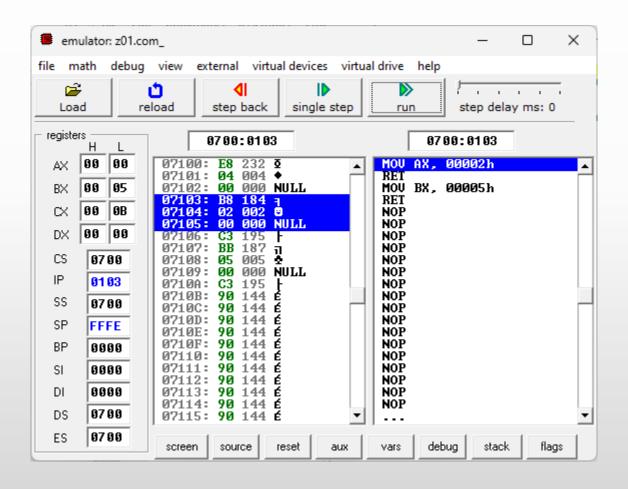






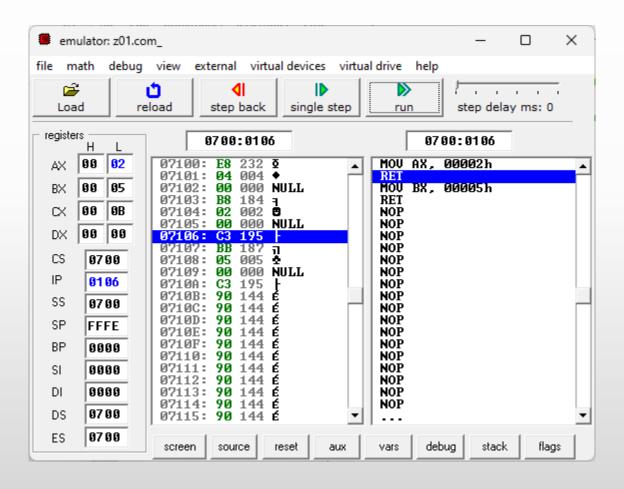
















- En kolay yolu, yazmaçları kullanmaktır.
- Örneğin, AL ve BL yazmaçları iki parametreyi temsil eder.
- m2 prosedürü,
 - AL ve BL yazmaçlarını kullanarak iki parametre alır,
 - Çarpar ve sonucu AX yazmacına saklar.
- CALL m2 prosedürünü çağırır.
- RET: Çağrıyı yapan yere döner.

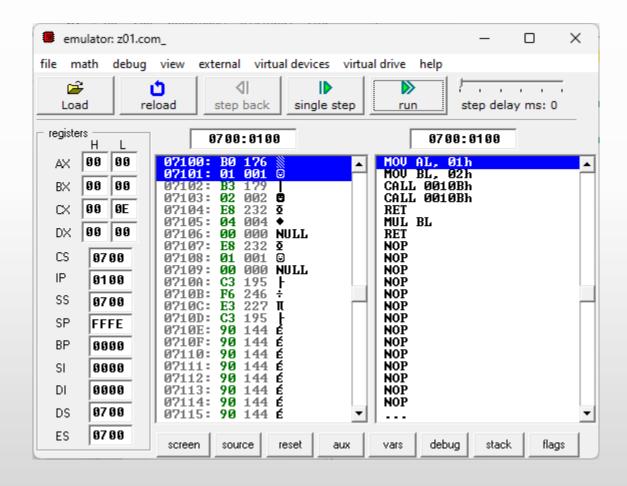




```
ORG
       100h
 MOV
      AL, 1
 MOV BL, 2
 CALL
         m2
  CALL
        m2
                      ; return to operating system.
RET
m2
       PROC
        BL
                        ; AX = AL * BL.
 MUL
                      ; return to caller.
RET
m2
       ENDP
END
```

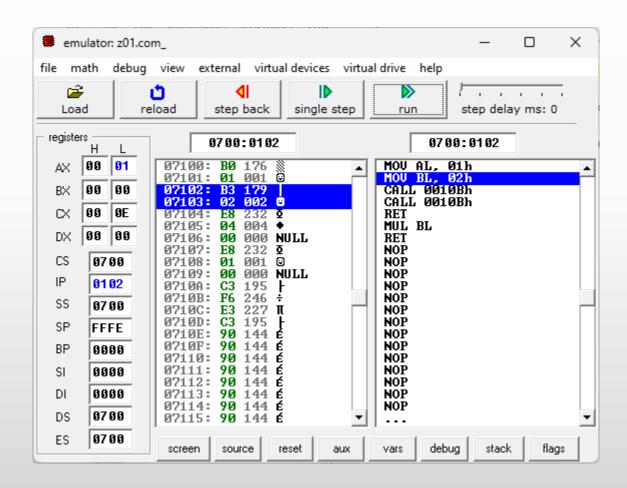






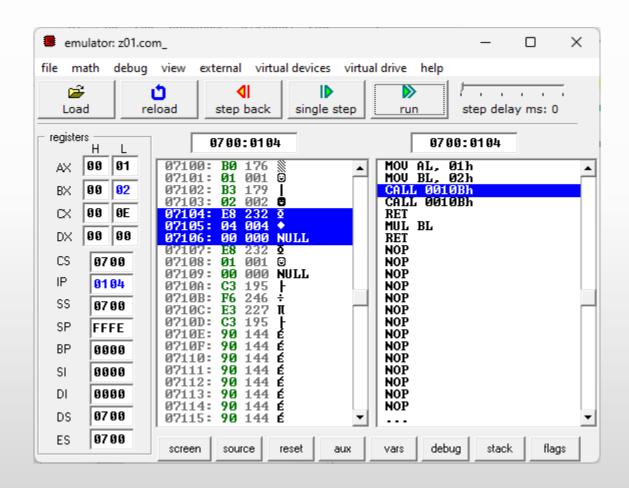






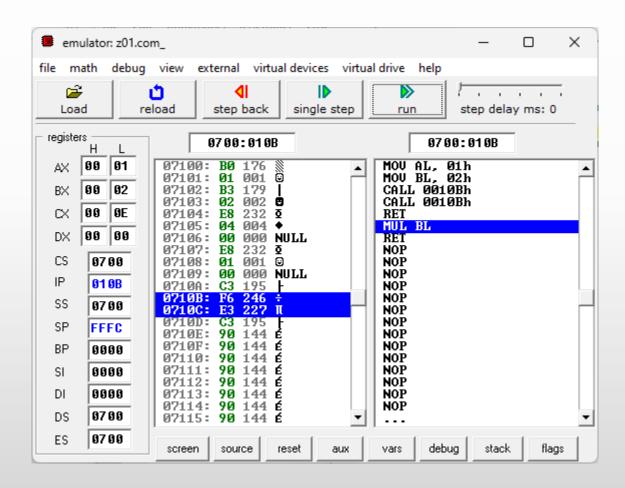






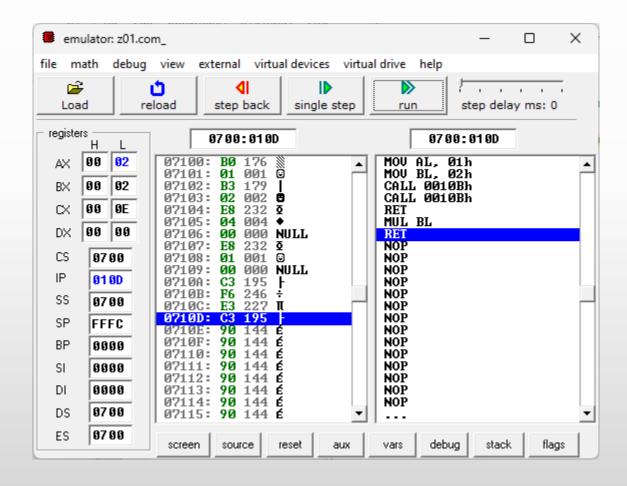






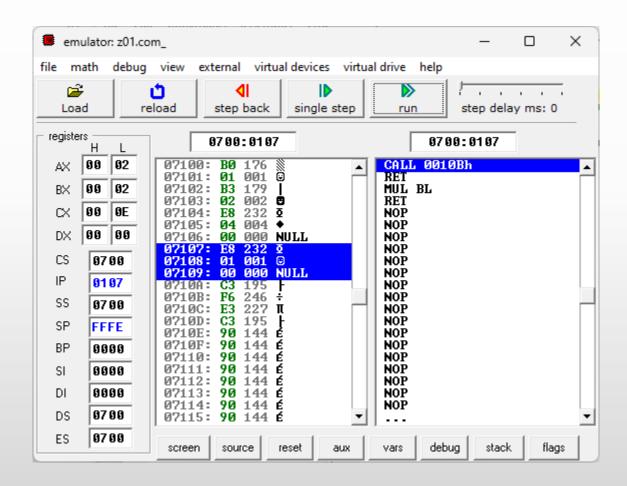






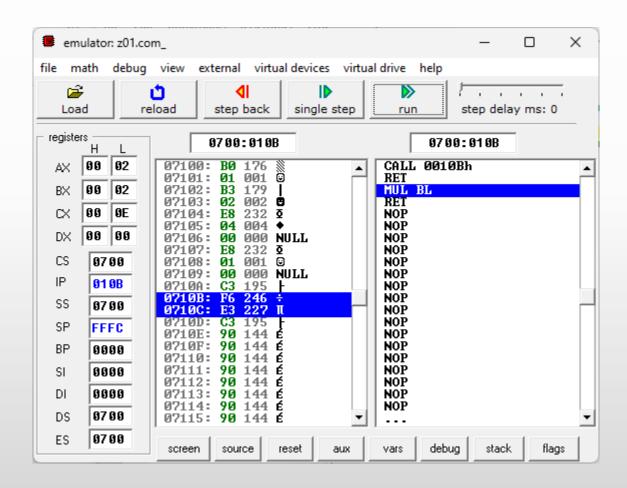






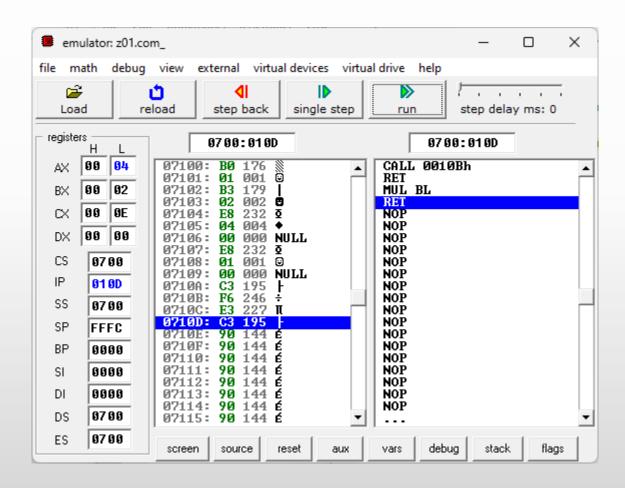
















- Prosedür kullanarak "Merhaba Dünya!" mesajı yazdırma.
- LEA SI, msg:
 - msg adlı dizgenin adresini SI yazmacına yükler.
- CALL print_me:
 - print_me prosedürünü çağırır.
- RET:
 - İşletim sistemine geri döner.
- print_me prosedürü,
 - null ile sona eren bir dizgeyi yazdırır.



Merhaba Dünya Mesajı Yazdırma

```
ORG
      100h
LEA SI, msg
                     ; load address of msg to SI.
CALL print me
RET
                     ; return to operating system.
        PROC
print_me
print_me ENDP
msg DB 'Hello World!', 0; null terminated string.
END
```



Merhaba Dünya Mesajı Yazdırma

```
PROC
print_me
next_char:
        b.[SI], 0 ; check for zero to stop
   CMP
   JE stop
   MOV AL, [SI]
                    ; next get ASCII char.
   MOV AH, ØEh
                     ; teletype function number.
   INT
        10h
                     ; using interrupt to print a char in AL.
   ADD SI, 1
                     ; advance index of string array.
       next_char
                     ; go back, and type another char.
   JMP
stop:
                     ; return to caller.
RET
print_me
            ENDP
```

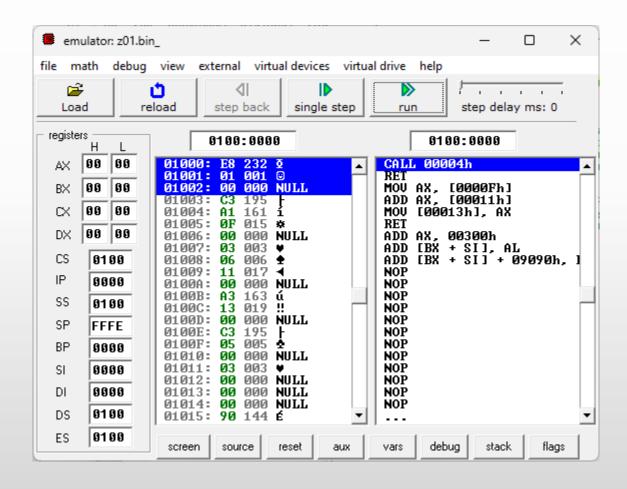




```
start:
  CALL AddNumbers ; prosedürü çağır
RET
AddNumbers PROC
  MOV AX, number1 ; İlk sayıyı AX yazmacına yükle
  ADD AX, number2 ; İkinci sayıyı AX yazmacına ekle
  MOV result, AX ; Sonucu result değişkenine taşı
  RET
              ; Prosedürü bitir
AddNumbers ENDP
number1 DW 5 ; İlk sayı 5
number2 DW 3 ; İkinci sayı 3
result DW?; Sonucu tutar
```

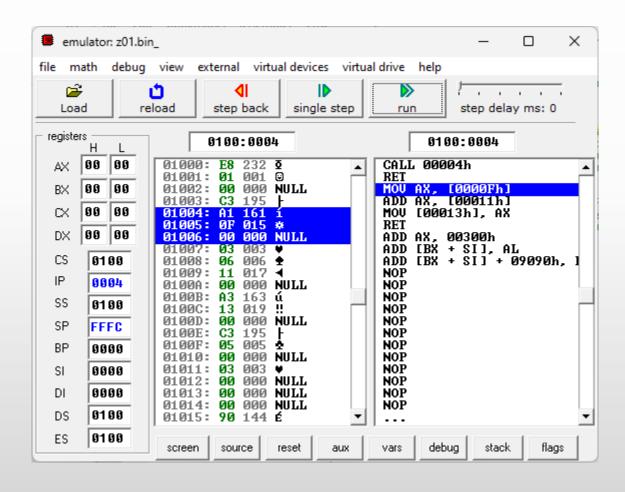






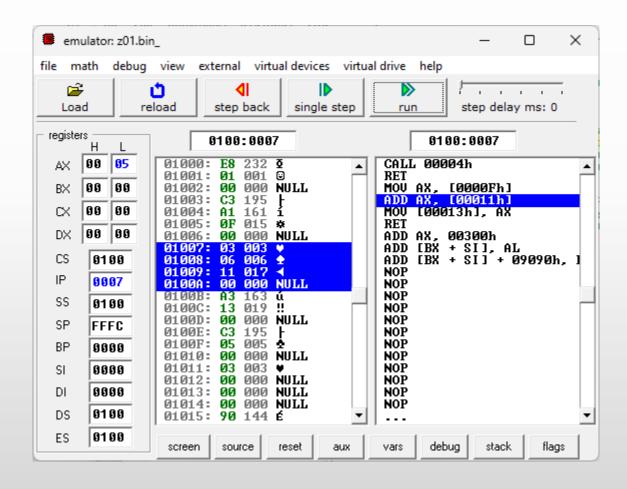






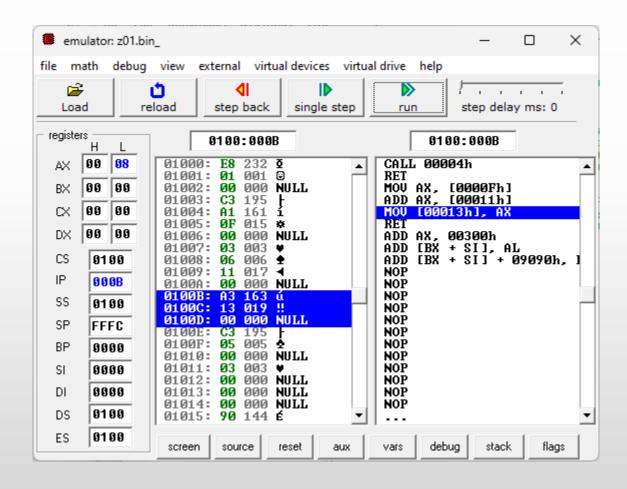






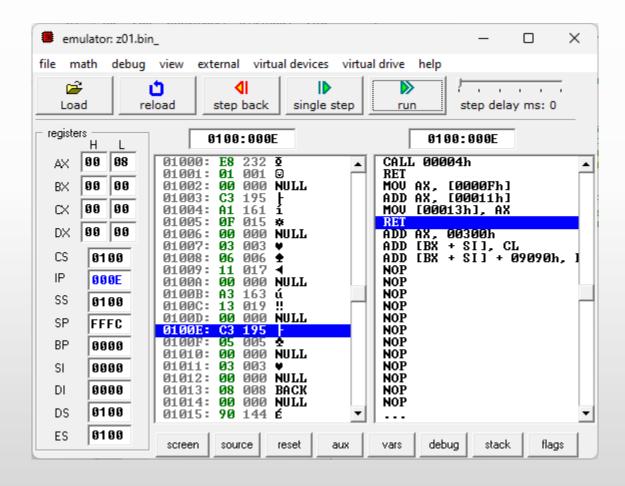






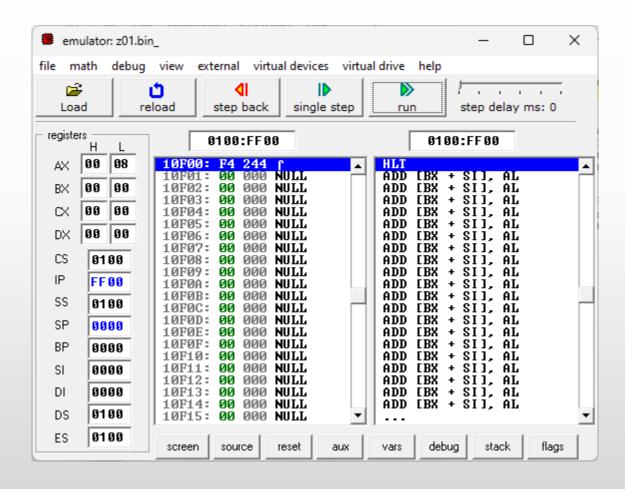














SON