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.

Prosedür Tanımlama

```
name PROC
    ; Prosedür kodu buraya yazılır.
    RET
name ENDP
```

name: Prosedürün adıdır. Başta ve sonda aynı olmalıdır.

Derleyici Direktifleri

- 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.

Örnek Kod Parçası

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

Prosedürlere Parametre Geçirme

- 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.

Prosedürlere Parametre Geçirme

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

Merhaba Dünya Mesajı Yazdırma

- 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.
print_me   PROC
; .....................
print_me   ENDP
msg     DB  'Hello World!', 0   ; null terminated string.
END
```

Merhaba Dünya Mesajı Yazdırma

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

SON