

ALT SEVİYE PROGRAMLAMA

Hafta 3

Dr. Öğr. Üyesi Erkan USLU

YIĞIN KOMUTLARI

Yığın Komutları

- POP
- POPF
- PUSH
- PUSHF

Yığın Komutları

- POP: pop word from stack
- POP dest
- $\text{dest} \leftarrow \text{SS}:[\text{SP}]$
- POP regw
- POP mem

Yığın Komutları

- POPF: pop flag from stack
- POPF
- FLAG \leftarrow SS:[SP]

Yığın Komutları

- PUSH: push word to stack
- PUSH src
- $SS:[SP] \leftarrow src$
- PUSH idata
- PUSH regw
- PUSH mem
- PUSH sreg

Yığın Komutları

- PUSH: push flag to stack
- PUSHF
- $SS:[SP] \leftarrow FLAG$

DÖNGÜ
KOMUTLARI

Döngü Komutları

- LOOP
- LOOPZ/LOOPE
- LOOPNZ/LOOPNE
- JCXZ

Döngü Komutları

- LOOP: loop until complete
- LOOP disp
- Her loop geçişinde CX azaltılır, sonrasında disp ile gösterilen adrese gidilir
- CX≠0 olduğu sürece tekrarlanır

Döngü Komutları

	toplam	CX	SI	
MOV toplam, 0;	0	?	?	
MOV CX, 5;	0	5	?	
MOV SI, 1;	0	5	1	
L1: ADD toplam, SI;	1	15	1	5
INC SI;	1	15	1	6
LOOP L1;	1	15	0	6

Döngü Komutları

- LOOPZ/LOOPE: loop zero/loop equal
- LOOPZ disp
- Her loop geçişinde CX azaltılır, sonrasında disp ile gösterilen adrese gidilir
- $CX \neq 0$ ve $ZF=1$ olduğu sürece tekrarlanır

Döngü Komutları

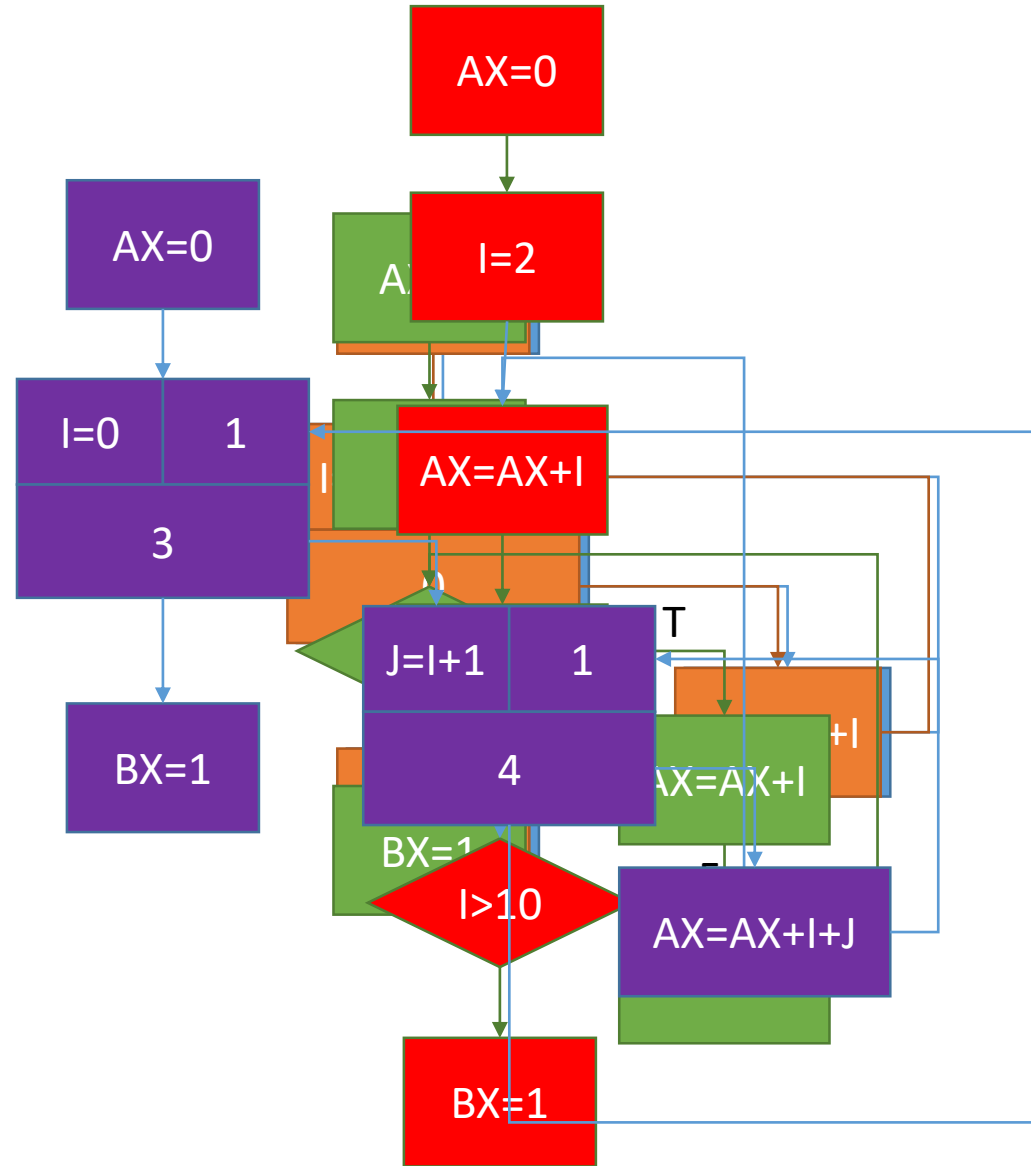
- LOOPNZ/LOOPNE: loop not zero/loop not equal
- LOOPNZ disp
- Her loop geçişinde CX azaltılır, sonrasında disp ile gösterilen adrese gidilir
- $CX \neq 0$ ve $ZF=0$ olduğu sürece tekrarlanır

Döngü Komutları

- JCXZ: jump cx zero
- JCXZ disp
- CX=0 ise dallan, CX değerinin kontrolü kullanıcıda

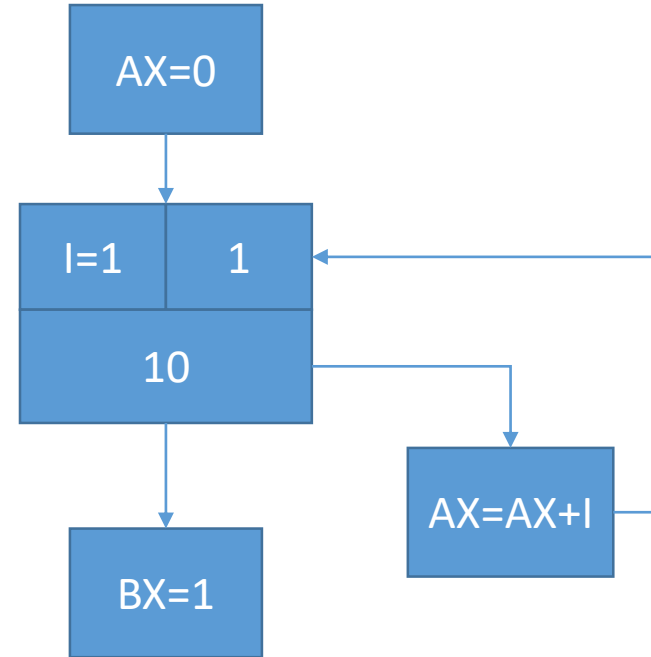
Döngü Şekilleri

- For to
- For downto
- While
- Repeat until
- For for



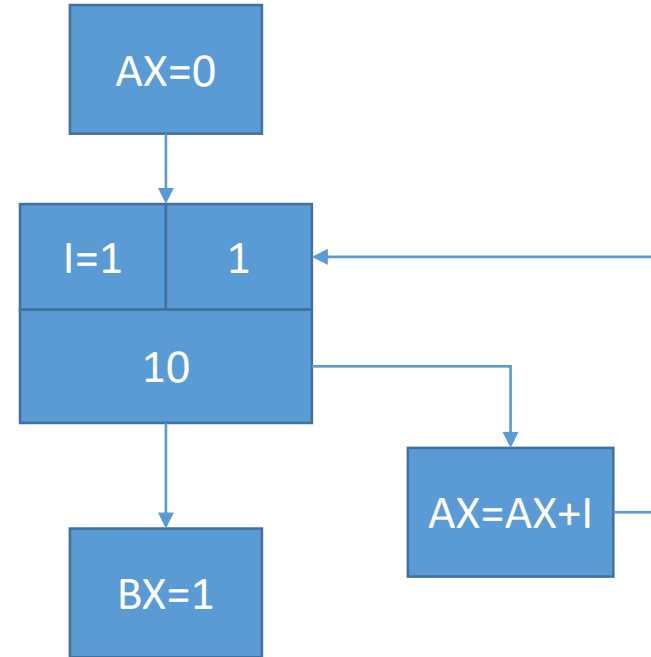
Döngü Şekilleri

- XOR AX, AX
- MOV SI, 1
- MOV CX, 10
- tekrar: ADD AX, SI
- INC SI
- LOOP tekrar



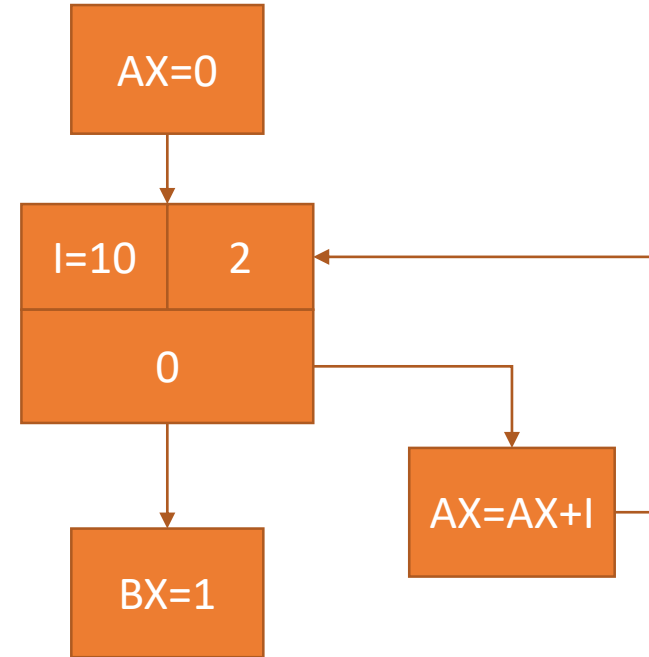
Döngü Şekilleri

- XOR AX, AX
- MOV CX, 10
- tekrar: ADD AX, CX
- LOOP tekrar



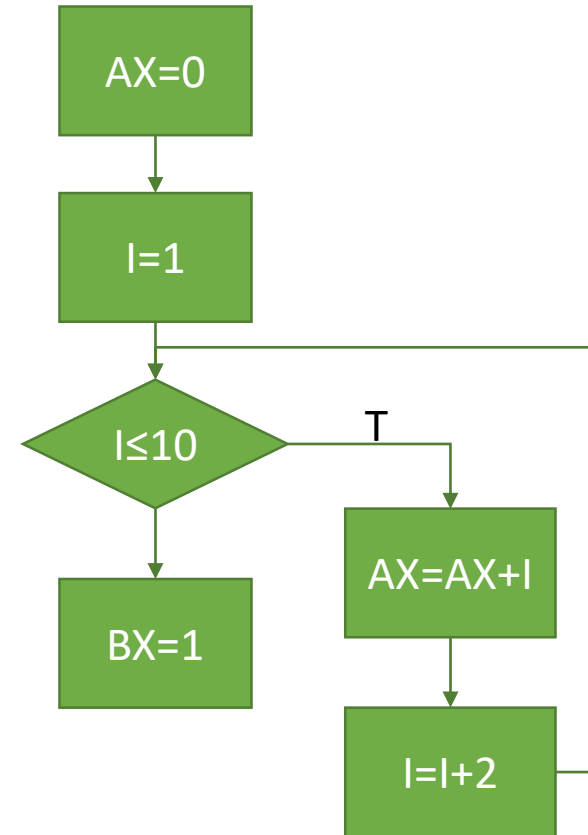
Döngü Şekilleri

- XOR AX, AX
- MOV CX, 10
- tekrar: ADD AX, CX
- SUB CX, 2
- JCXZ devam
- JMP tekrar
- devam:



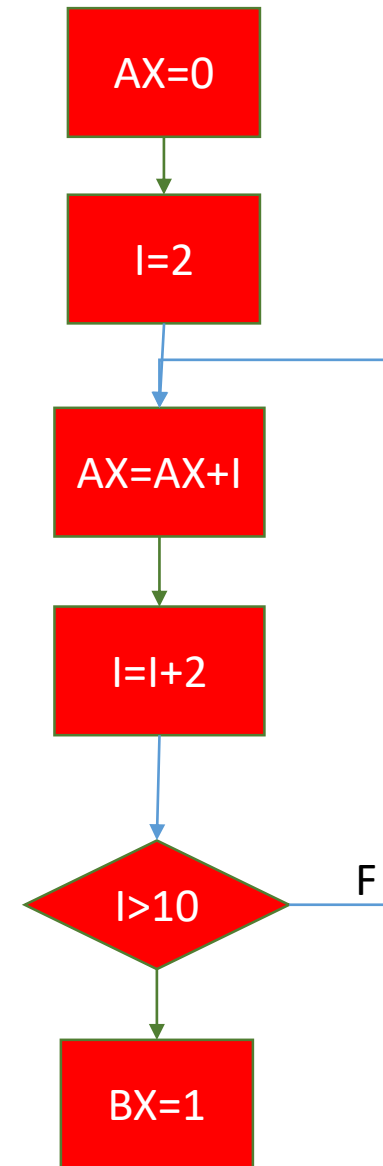
Döngü Şekilleri

- XOR AX, AX
- MOV SI, 1
- tekrar: CMP SI, 10
- JA devam
- ADD AX, SI
- ADD SI, 2
- JMP tekrar
- devam: MOV BX, 1



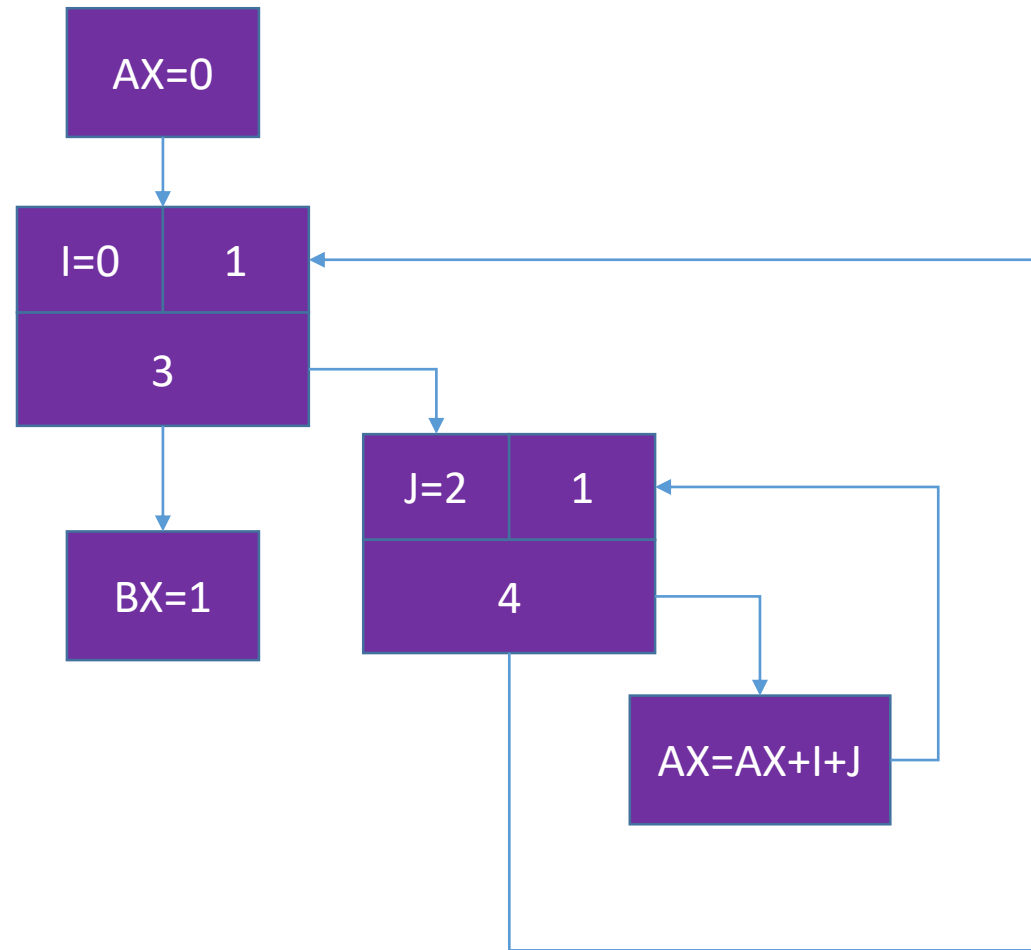
Döngü Şekilleri

- XOR AX, AX
- MOV SI, 2
- tekrar: ADD AX, SI
- ADD SI, 2
- CMP SI, 10
- JBE tekrar
- MOV BX, 1



Döngü Şekilleri

- MOV AX, 0
- MOV CX, 4
- MOV DI, 0
- L2: PUSH CX
- MOV CX, 3
- MOV SI, 2
- L1: ADD AX, DI
- ADD AX, SI
- INC SI
- LOOP L1
- POP CX
- INC DI
- LOOP L2
- MOV BX, 1



BAYRAKLARLA İLGİLİ KOMUTLARI

Bayraklarla İlgili Komutlar

- CLC: clear carry f.
- CMC: complement carry f.
- STC: set carry f.
- CLD: clear direction f.
- STD: set direction f.
- STI: set interrupt f.
- CLI: clear intrerrupt f.
- LAHF: load AH with flag: $AH \leftarrow SF, ZF, ?, AF, ?, PF, ?CF$
- SAHF: store AH to flag: $SF, ZF, ?, AF, ?, PF, ?CF \leftarrow AH$

MANTIKSAL KOMUTLAR

Mantıksal Komutlar

- NOT
- OR
- AND
- XOR
- TEST

Mantıksal Komutlar

- NOT : bitwise not
- NOT dest
- $\text{dest} \leftarrow (\text{dest})'$
- NOT reg
- NOT mem; PTR gerekli

Mantıksal Komutlar

- OR : bitwise or
- OR dest, src
- $\text{dest} \leftarrow \text{dest OR src}$
- OR reg, idata
- OR mem, idata; PTR gerekli
- OR reg, reg
- OR reg, mem
- OR mem, reg

Mantıksal Komutlar

- AND : bitwise and
- AND dest, src
- $\text{dest} \leftarrow \text{dest AND src}$
- AND reg, idata
- AND mem, idata; PTR gerekli
- AND reg, reg
- AND reg, mem
- AND mem, reg

Mantıksal Komutlar

- XOR : bitwise exclusive or
- XOR dest, src
- $\text{dest} \leftarrow \text{dest XOR src}$
- XOR reg, idata
- XOR mem, idata; PTR gerekli
- XOR reg, reg
- XOR reg, mem
- XOR mem, reg

Mantıksal Komutlar

- XOR AL, 0FFH ; AL'de 1'ler 0, 0'lar 1 yapılır
- XOR AX, AX ; $AX \leftarrow 0$, 4 clock cycle, 3 byte
- MOV AX, 0 ; 3 clock cycle, 2 byte

Mantıksal Komutlar

- TEST : test bits
- TEST dest, src
- dest AND src
- Sadece bayraklar değişir, sonuç saklanmaz
- TEST reg, idata
- TEST mem, idata; PTR gerekli
- TEST reg, reg
- TEST reg, mem
- TEST mem, reg

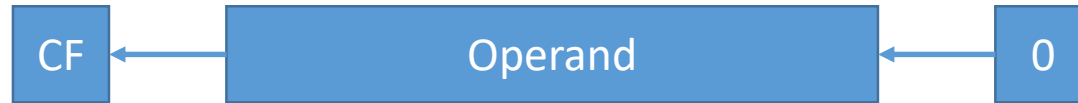
ÖTELEME VE DÖNDÜRME KOMUTLARI

Öteleme ve Döndürme Komutları

- SHL
- SAL
- SHR
- SAR
- RCL
- RCR
- ROL
- ROR

Öteleme ve Döndürme Komutları

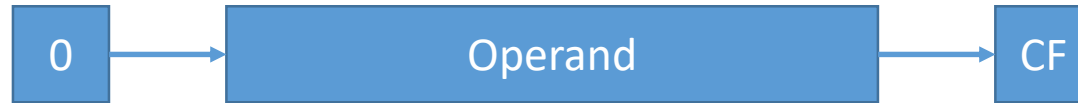
- SHL : shift left logical



- SHL reg, 1
- SHL mem, 1
- SHL reg, CL
- SHL mem, CL
- **SAL = SHL**

Öteleme ve Döndürme Komutları

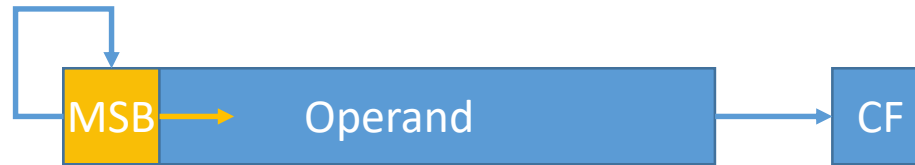
- SHR : shift right logical



- SHR reg, 1
- SHR mem, 1
- SHR reg, CL
- SHR mem, CL

Öteleme ve Döndürme Komutları

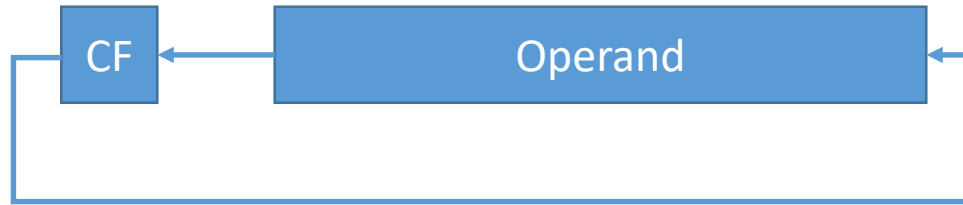
- SAR : shift arithmetic right



- Öteleme işleminde sayının işaret bitini dikkate alır
- SAR reg, 1
- SAR mem, 1
- SAR reg, CL
- SAR mem, CL

Öteleme ve Döndürme Komutları

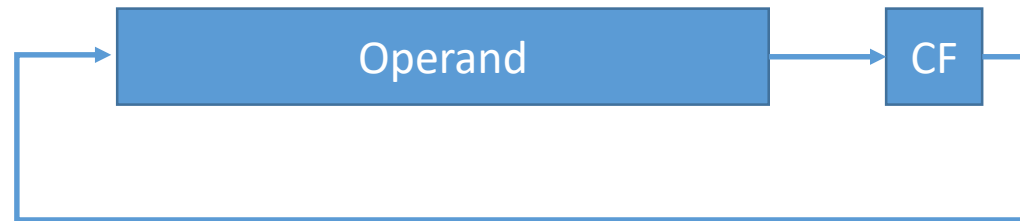
- RCL : rotate through carry left



- RCL reg, 1
- RCL mem, 1
- RCL reg, CL
- RCL mem, CL

Öteleme ve Döndürme Komutları

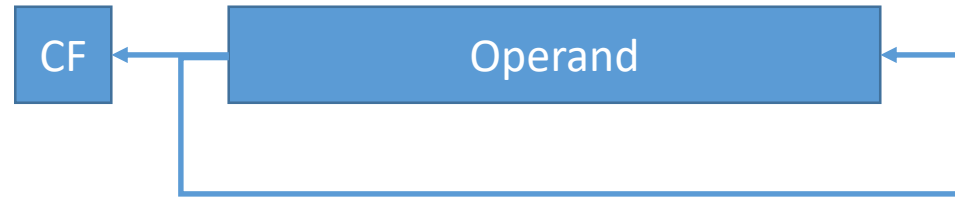
- RCR : rotate through carry right



- RCR reg, 1
- RCR mem, 1
- RCR reg, CL
- RCR mem, CL

Öteleme ve Döndürme Komutları

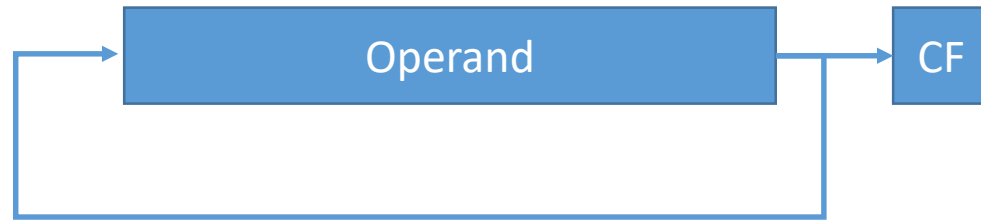
- ROL : rotate left



- ROL reg, 1
- ROL mem, 1
- ROL reg, CL
- ROL mem, CL

Öteleme ve Döndürme Komutları

- ROR : rotate right



- ROR reg, 1
- ROR mem, 1
- ROR reg, CL
- ROR mem, CL