

ALT SEVİYE PROGRAMLAMA

Hafta 2

Dr. Öğr. Üyesi Erkan USLU

80x86 KOMUT YAPISI

80x86 Komut Yapısı

{Etiket :} **Mnemonic**

{{Operand1,} Operand2}

{; Açıklamalar}

80x86 Komut Yapısı

- 80x86'da komutlar:
 - Operand almayan
 - Tek operand alan
 - İki operand alan
 - Üç operand (80x86 ailesi üst serilerinde)
 - $\text{Operand1} \leftarrow \text{Operand1} \text{ **işlem** } \text{Operand2}$
 - Operandlar aynı tipte olmalı veya uygun sözde komut dönüşümü yapılmalı

80x86 VERİ AKTARIM KOMUTLARI

Komutlardaki Kısaltmalar

- acc : akümülatör
- reg : 8/16 bitlik yazmaç
- regb : 8 bitlik yazmaç
- regw : 16 bitlik yazmaç
- sreg : segment (kesim) yazmacı
- mem : bellek adresi
- idata : 8/16 bitlik sabit değer
- disp8/disp16 : [-128...0...127]/[-32768...0...32767]
- dest/scr : hedef/kaynak

Veri Aktarım Komutları

- MOV
 - LEA
 - LDS
 - LES
 - XCHG
 - XLAT/XLATB
-
- Bayraklar etkilenmez

Veri Aktarım Komutları

- MOV : move data
- MOV dest, src
- $\text{dest} \leftarrow \text{src}$
- dest=mem ve src=mem olamaz
- dest=sreg ve src=sreg olamaz
- dest=sreg ve src=idata olamaz
- dest ve src aynı tipte olmalıdır

Veri Aktarım Komutları

- MOV reg, idata
- MOV mem, idata
- MOV reg, reg
- MOV reg, mem
- MOV mem, reg
- MOV sreg, reg
- MOV sreg, mem
- MOV reg, sreg
- MOV mem, sreg

Veri Aktarım Komutları

- LEA : load effective address
- LEA regw, mem
- $\text{regw} \leftarrow \text{offset}(\text{mem})$

Veri Aktarım Komutları

- LDS : load data segment register
- LDS regw, mem
- regw \leftarrow [mem]
- DS \leftarrow [mem+2]

Veri Aktarım Komutları

- LES : load extra segment register
- LES regw, mem
- regw \leftarrow [mem]
- ES \leftarrow [mem+2]

Veri Aktarım Komutları

- XCHG : exchange
- XCHG dest, src
- dest <-> src
- XCHG reg,reg
- XCHG reg, mem
- XCHG mem, reg

Veri Aktarım Komutları

- XLAT/XLATB : translate byte (XLAT = XLATB)
- Doğrudan operandı olmayan bir komut, AL gizli operand
- $AL \leftarrow DS:[BX+AL]$
- DS:[BX] adresindeki tablonun AL numaralı indisindeki değeri AL yazmacına kopyalar

ARİTMETİK KOMUTLAR

Aritmetik Komutlar

- ADD
- ADC
- SUB
- SBB
- INC
- DEC
- NEG
- CMP
- MUL
- IMUL
- DIV
- IDIV

Aritmetik Komutlar

- ADD : addition
- ADD dest, src
- $\text{dest} \leftarrow \text{dest} + \text{src}$
- ADD reg, idata
- ADD mem, idata; PTR gerekli
- ADD reg, reg
- ADD reg, mem
- ADD mem, reg

Aritmetik Komutlar

- ADC : add with carry
- ADC dest, src
- $\text{dest} \leftarrow \text{dest} + \text{src} + \text{CF}$
- ADC reg, idata
- ADC mem, idata; PTR gerekli
- ADC reg, reg
- ADC reg, mem
- ADC mem, reg

Aritmetik Komutlar

- SUB : subtraction
- SUB dest, src
- $\text{dest} \leftarrow \text{dest} - \text{src}$
- SUB reg, idata
- SUB mem, idata; PTR gerekli
- SUB reg, reg
- SUB reg, mem
- SUB mem, reg

Aritmetik Komutlar

- SBB : subtraction with borrow
- SBB dest, src
- $\text{dest} \leftarrow \text{dest} - \text{src} - \text{CF}$
- SBB reg, idata
- SBB mem, idata; PTR gerekli
- SBB reg, reg
- SBB reg, mem
- SBB mem, reg

Aritmetik Komutlar

- INC : increment
- INC dest
- $\text{dest} \leftarrow \text{dest} + 1$
- INC reg
- INC mem; PTR gerekli

Aritmetik Komutlar

- DEC : decrement
- DEC dest
- $\text{dest} \leftarrow \text{dest} - 1$
- DEC reg
- DEC mem; PTR gerekli

Aritmetik Komutlar

- NEG : negate / two's complement
- NEG dest
- $\text{dest} \leftarrow 0 - \text{dest}$
- NEG reg
- NEG mem; PTR gerekli

Aritmetik Komutlar

- CMP : compare
- CMP dest, src
- $\text{dest} - \text{src}$
- Sonuç saklanmaz, bayraklar etkilenir
- Komut sonrasında koşullu dallanma komutları kullanılabilir
 - $\text{dest} > \text{src}$, $\text{dest} \geq \text{src}$,
 - $\text{dest} = \text{src}$,
 - $\text{dest} \leq \text{src}$, $\text{dest} < \text{src}$

Aritmetik Komutlar

- CMP reg, idata
- CMP mem, idata
- CMP reg, reg
- CMP reg, mem
- CMP mem, reg

Aritmetik Komutlar

- MUL : unsigned multiplication
- MUL src
- Eğer src 8 bit ise : $AX \leftarrow AL \times src$
- Eğer src 16 bit ise : $DX:AX \leftarrow AX \times src$
- MUL reg
- MUL mem; PTR gerekli

Aritmetik Komutlar

- IMUL : integer signed multiplication
- IMUL src
- Eğer src 8 bit ise : $AX \leftarrow AL \times src$
- Eğer src 16 bit ise : $DX:AX \leftarrow AX \times src$
- Sonuç ve operandlar işaretli sayı olarak değerlendirilir
- IMUL reg
- IMUL mem; PTR gerekli

Aritmetik Komutlar

- DIV : unsigned divide
- DIV src
- Eğer src 8 bit ise : $AL \leftarrow AX / src, AH \leftarrow AX \% src$
- Eğer src 16 bit ise :

$AX \leftarrow DX:AX / src, DX \leftarrow DX:AX \% src$

- DIV reg
- DIV mem; PTR gerekli

Aritmetik Komutlar

- IDIV : integer signed divide
- IDIV src
- Eğer src 8 bit ise : $AL \leftarrow AX / src, AH \leftarrow AX \% src$
- Eğer src 16 bit ise :

$AX \leftarrow DX:AX / src, DX \leftarrow DX:AX \% src$

Sonuç ve operandlar işaretli sayı olarak değerlendirilir

- IDIV reg
- IDIV mem; PTR gerekli

DALLANMA
KOMUTLARI

Dallanma Komutları

- Koşullu Dallanma
 - Basit Koşullu Dallanma
 - İşaretsiz Sayı İşlemlerinde Dallanma
 - İşaretli Sayı İşlemlerinde Dallanma
- Koşulsuz Dallanma
- Çağırma Komutları
- Dönüş Komutları

Dallanma Komutları (1)

Koşullu Dallanma

- Basit Koşullu Dallanma

- JE/JZ
- JNZ/JNE
- JS
- JNS
- JO
- JNO
- JP/JPE
- JNP/JPO

- İşaretsiz Sayı İşlemlerinde Dallanma

- JB/JNAE/JC
- JA/JNBE
- JAE/JNB/JNC
- JBE/JNA

- İşaretli Sayı İşlemlerinde Dallanma

- JL/JNGE
- JNL/JGE
- JLE/JNG
- JG/JNLE

Basit Koşullu Dallanma

- JE/JZ: jump zero/equal, ZF=1 ise dallan
- JNZ/JNE: jump not zero/not equal, ZF=0 ise dallan
- JS: jump if sign, SF=1 ise dallan
- JNS: jump no sign, SF=0 ise dallan
- JO: jump if overflow, OF=1 ise dallan
- JNO: jump if no overflow, OF=0 ise dallan
- JP/JPE: jump on parity/even parity, PF=1 ise dallan
- JNP/JPO: jump no parity/odd parity, PF=0 ise dallan

İşaretsiz Sayı İşlemlerinde Dallanma

- İşaretsiz sayılarda CMP işlemi sonrasında koşullu dallanma oluşturmak için kullanılır
- JB/JNAE/JC: jump if below
- JA/JNBE: jump if above
- JAE/JNB/JNC: jump if above or equal
- JBE/JNA: jump if below or equal

İşaretli Sayı İşlemlerinde Dallanma

- İşaretli sayılarda CMP işlemi sonrasında koşullu dallanma oluşturmak için kullanılır
- JL/JNGE: jump if less
- JNL/JGE: jump if greater or equal
- JLE/JNG: jump if less or equal
- JG/JNLE: jump if greater

Aritmetik Komutlar- İşlemci koşullara nasıl karar verir

	İşaretsiz Sayılarda	İşaretli Sayılarda
$\text{dest} > \text{src}$	$\text{CF}=0$ VE $\text{ZF}=0$	$\text{ZF}=0$ VE $\text{SF}=\text{OF}$
$\text{dest} \geq \text{src}$	$\text{CF}=0$	$\text{SF}=\text{OF}$
$\text{dest} = \text{src}$	$\text{ZF}=1$	$\text{ZF}=1$
$\text{dest} \leq \text{src}$	$\text{CF}=1$ VEYA $\text{ZF}=1$	$\text{ZF}=1$ VEYA $\text{SF} \neq \text{OF}$
$\text{dest} < \text{src}$	$\text{CF}=1$	$\text{SF} \neq \text{OF}$

Dallanma Komutları (2)

- Koşulsuz Dallanma

- JMP
- JMP FAR PTR

- Çağırma Komutları

- CALL
- CALL FAR PTR
- INT
- INTO

- Dönüş Komutları

- RET
- RETF
- IRET

Koşulsuz Dallanma

- JMP: near jump
- JMP disp
- $IP \leftarrow IP + disp$
- JMP reg
- $IP \leftarrow reg$
- JMP mem
- $IP \leftarrow [mem]$

Koşulsuz Dallanma

- JMP FAR PTR: far jump
- JMP FAR PTR idata1:idata2
- $CS \leftarrow idata1, IP \leftarrow idata2$

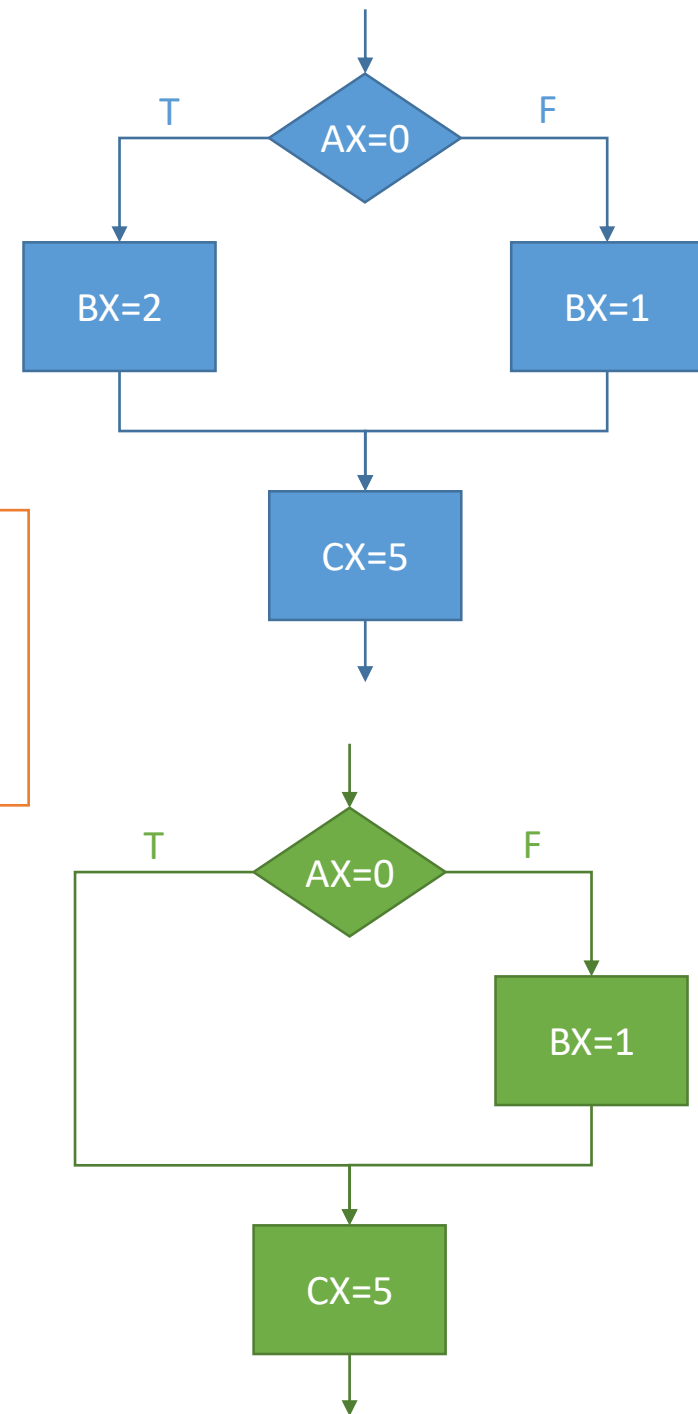
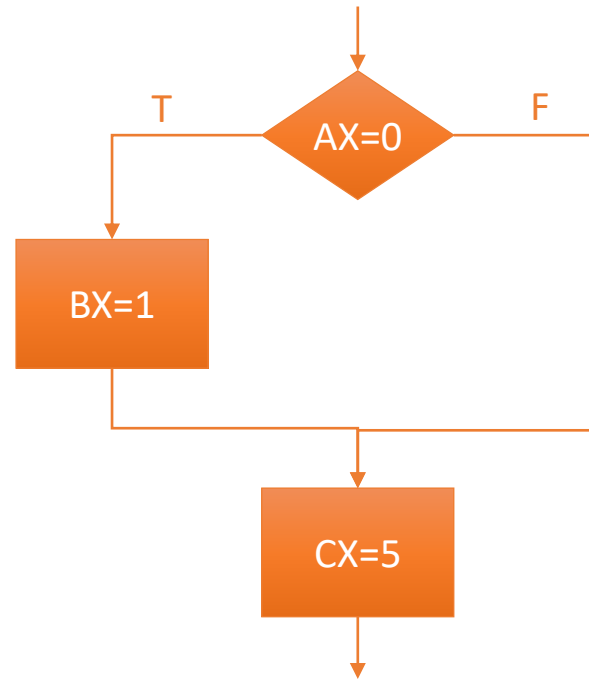
- JMP FAR PTR mem
- $IP \leftarrow [mem], CS \leftarrow [mem+2]$

Dallanma Şekilleri

- If then else

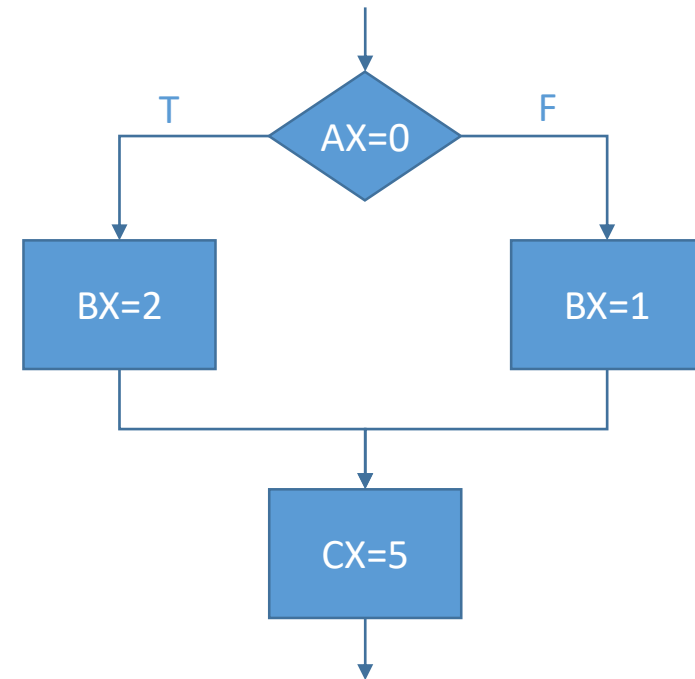
- If null else

If null then



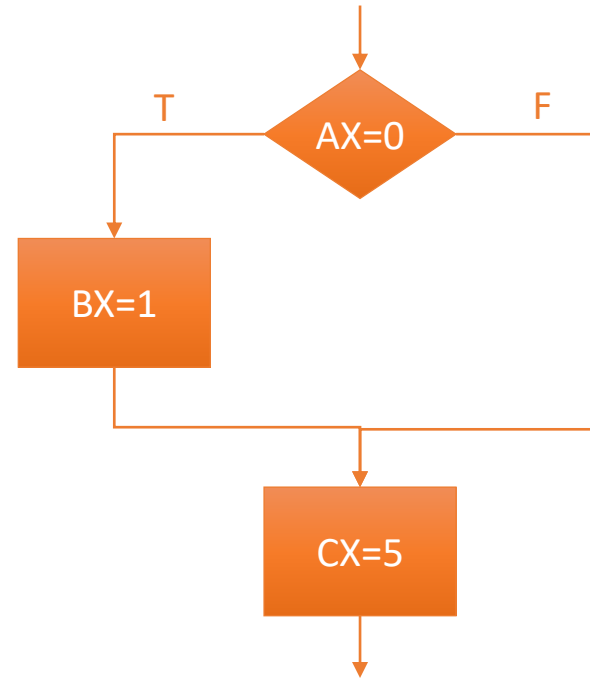
Dallanma Şekilleri

- `CMP AX, 0`
- `JZ true`
- `MOV BX, 1`
- `JMP next`
- `true: MOV BX, 2`
- `next: MOV CX, 5`



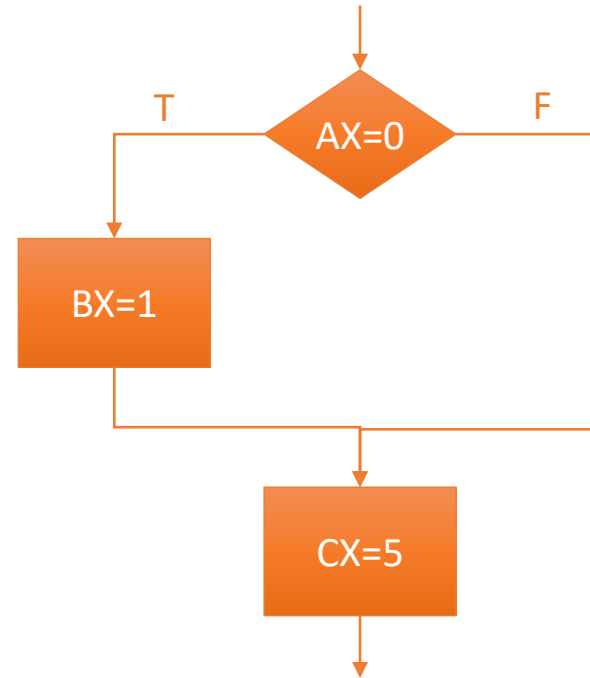
Dallanma Şekilleri

- `CMP AX, 0`
- `JZ true`
- `JMP next`
- `true: MOV BX, 1`
- `next: MOV CX, 5`



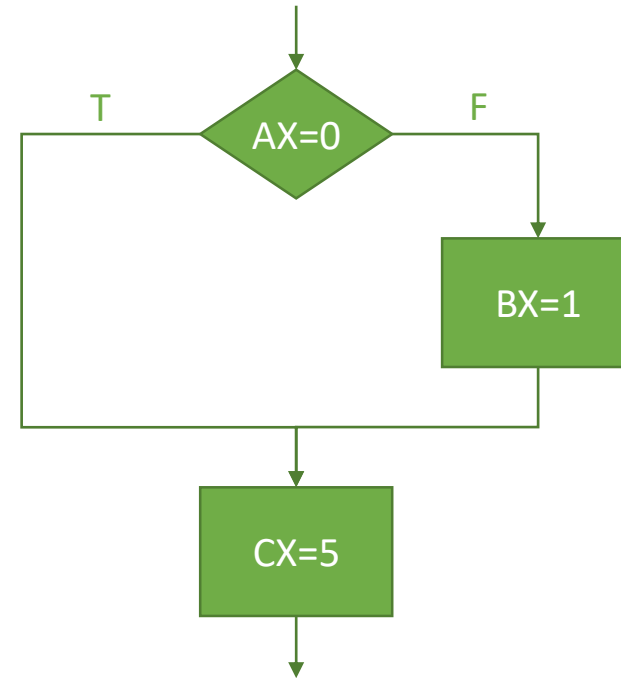
Dallanma Şekilleri

- `CMP AX, 0`
- `JNZ false`
- `MOV BX, 1`
- `false: MOV CX, 5`



Dallanma Şekilleri

- `CMP AX, 0`
- `JZ true`
- `MOV BX, 1`
- `true: MOV CX, 5`



Çağırma Komutları

- CALL: near procedure call, (prosedür dönüşü RET ile)
- Yığına sadece IP atılır
- CALL disp
- $IP \leftarrow IP + disp$
- CALL mem
- $IP \leftarrow reg$
- CALL reg
- $IP \leftarrow reg$

Çağırma Komutları

- CALL FAR PTR: far procedure call, (prosedür dönüşü RETF ile)
- Yığına CS ve IP atılır
- CALL FAR PTR idata1:idata2
- $CS \leftarrow idata1, IP \leftarrow idata2$
- CALL FAR PTR mem
- $IP \leftarrow [mem], CS \leftarrow [mem+2]$

Çağırma Komutları

- INT: software interrupt
- INT idata
- İşlemci aşağıdaki işlemleri otomatik yapar
 - PUSHF
 - PUSH CS
 - PUSH IP
 - $TF \leftarrow 0$
 - $IF \leftarrow 0$
 - $IP \leftarrow 0000:[idata*4]$
 - $CS \leftarrow 0000:[idata*4+2]$

Çağırma Komutları

- INTO: interrupt on overflow
- INTO
- INTO = INT 04H

Dönüş Komutları

- RET: near return from procedure
 - Yığından sadece IP çekilir
- RETF: far return from procedure
 - Yığından IP ve CS çekilir
- IRET: return from interrupt
 - Yığından IP, CS ve FLAG çekilir

EK - Tüm dallanma komutları ve
bayrak koşulları

Opcode	Description	CPU Flags
JA	Above	CF = 0 and ZF = 0
JAE	Above or equal	CF = 0
JB	Bellow	CF
JBE	Bellow or equal	CF or ZF
JC	Carry	CF
JE	Equality	ZF
JG	Greater ^(s)	ZF = 0 and SF = OF
JGE	Greater of equal ^(s)	SF = OF
JL	Less ^(s)	SF ≠ OF
JLE	Less equal ^(s)	ZF or SF ≠ OF
JNA	Not above	CF or ZF
JNAE	Neither above nor equal	CF
JNB	Not bellow	CF = 0
JNBE	Neither bellow nor equal	CF = 0 and ZF = 0
JNC	Not carry	CF = 0
JNE	Not equal	ZF = 0
JNG	Not greater	ZF or SF ≠ OF
JNGE	Neither greater nor equal	SF ≠ OF
JNL	Not less	SF = OF
JNLE	Not less nor equal	ZF = 0 and SF = OF

Opcode	Description	CPU Flags
JNO	Not overflow	OF = 0
JNP	Not parity	PF = 0
JNS	Not negative	SF = 0
JNZ	Not zero	ZF = 0
JO	Overflow ^(s)	OF
JP	Parity	PF
JPE	Parity	PF
JPO	Not parity	PF = 0
JS	Negative ^(s)	SF
JZ	Null	ZF

(s): signed numbers

EK - Dallanma komutlarının
eşdeğerliliği

JB \equiv JC \equiv JNAE

JAЕ \equiv JNB \equiv JNC

JE \equiv JZ

JNE \equiv JNZ

JBE \equiv JNA

JA \equiv JNBE

JP \equiv JPE

JNP \equiv JPO

JL \equiv JNGE

JGE \equiv JNL

JLE \equiv JNG

JG \equiv JNLE