

BİL 362 Mikroişlemciler: Aritmetik İşlem Komutları - Ek

Ahmet Burak Can
abc@hacettepe.edu.tr

1

Çarpma ve Bölme Komutları

- İşaretsiz tamsayılarda çarpma ve bölme
 - MUL komutu
 - DIV komutu
- İşaretlı tamsayılarda çarpma ve bölme
 - CBW, CWD komutları
 - IMUL komutu
 - IDIV komutu
- Aritmetik ifadeleri gerçekleştirme: Örnekler

2

MUL Komutu

- MUL (“unsigned multiply”) komutu AL, AX veya EAX yazmaçlarının değerlerini; 8-, 16- veya 32-bit işlenenlerle çarpar.

- Geçerli işlenenler:

Multiplicand	Multiplier	Product
AL	<i>r/m8</i>	AX
AX	<i>r/m16</i>	DX:AX
EAX	<i>r/m32</i>	EDX:EAX

↓
“register” / “memory”

3

MUL Komutu: Örnekler

100h * 2000h (16-bit işlenenler):

```
.data
val1 WORD 2000h
val2 WORD 100h
.code
mov ax, val1
mul val2 ; DX:AX = 00200000h, CF=1
```

Elde bayrağı (CF),
çarpımın üst kısmında
1'in varlığını gösterir.

12345h * 1000h (32-bit işlenenler):

```
mov eax, 12345h
mov ebx, 1000h
mul ebx ; EDX:EAX = 0000000012345000h, CF=0
```

4

Alıştırma (16-bit MUL)

Aşağıdaki komutlar işletildikten sonra AX ve DX yazmaçları ile Elde bayrağının (CF) değerlerini gösterin.

```
mov ax,1234h
mov bx,100h
mul bx
```

DX = 0012h, AX = 3400h, CF = 1

5

Alıştırma (32-bit MUL)

Aşağıdaki komutlar işletildikten sonra EAX ve EDX yazmaçları ile Elde bayrağının (CF) değerlerini gösterin.

```
mov eax,00128765h
mov ecx,10000h
mul ecx
```

EDX = 00000012h, EAX = 87650000h, CF = 1

6

DIV Komutu

- DIV (“unsigned divide”) komutu işaretsiz işlenenler üzerinde; 8-, 16- veya 32-bit bölme yapar.
- Geçerli işlenenler:

Dividend	Divisor	Quotient	Remainder
AX	<i>r/m8</i>	AL	AH
DX:AX	<i>r/m16</i>	AX	DX
EDX:EAX	<i>r/m32</i>	EAX	EDX

7

DIV Komutu: Örnekler

8003h / 100h (16-bit işlenenler):

```
mov dx,0                ; bölüneni temizle (üst kısım)
mov ax,8003h            ; bolunen (alt kısım)
mov cx,100h             ; bolen
div cx                  ; AX = 0080h, DX = 3
```

8003h / 100h (32-bit işlenenler):

```
mov edx,0               ; bolunen temizle (üst kısım)
mov eax,8003h           ; bolunen (alt kısım)
mov ecx,100h            ; bolen
div ecx                 ; EAX = 00000080h, EDX = 3
```

8

Alıştırma

Aşağıdaki komutlar işletildikten sonra AX ve DX yazmaçlarının değerlerini gösterin.

```
mov dx,0087h
mov ax,6000h
mov bx,100h
div bx
```

DX = 0000h, AX = 8760h

9

Alıştırma

Aşağıdaki komutlar işletildikten sonra AX ve DX yazmaçlarının değerlerini gösterin.

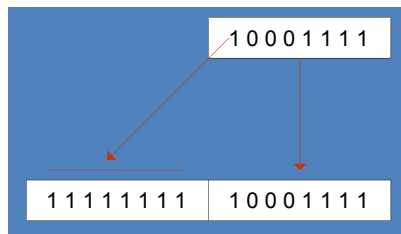
```
mov dx,0087h
mov ax,6002h
mov bx,10h
div bx
```

Divide Overflow

10

İşaretli Tamsayı Bölme

- İşaretli sayılar bölme gerçekleşmeden önce, işaretleriyle genişletilmelidir.
 - Üst kısmın bitleri, alt kısmın işaret biti ile doldurulur.
- Örnek:



11

CBW, CWD Komutları

- The CBW, CWD ve CDQ komutları, işaret genişletme işlemlerini gerçekleştirmeye yarar.
 - CBW ("convert byte to word"), AL'yi, AH'ye genişletir.
 - CWD ("convert word to doubleword"), AX'i, DX ve AX'e genişletir.
- Örnek:

```
mov ax,0FF9Bh      ; (-101)
cwd                ; EDX:EAX = 0FFFFFF9Bh
```

12

IMUL Komutu

- IMUL (“signed integer multiply”) komutu AL, AX veya EAX yazmaçlarının değerlerini; 8-, 16- veya 32-bit işlenenlerle çarpar.
- Çarpımın işaretini korur.
 - Çarpımın üst yarısı alt yarısının işaretiyle genişlememişse (“sign extension”) Elde (CF) ve Taşma (OF) bayrakları 1 olur.

Örnek: $48 * 4$ (8-bit işlenenler):

```
mov al,48
mov bl,4
imul bl      ; AX = 00C0h, OF=1
```

OF=1, çünkü AH yazmacı AL'nin işaretiyle genişlememiş.

13

IMUL Komutu: Örnek

$4,823,424 * -423$:

```
mov eax,4823424
mov ebx,-423
imul ebx      ; EDX:EAX = FFFFFFFF86635D80h, OF=0
```

OF=0, çünkü EDX yazmacı EAX'in işaretiyle genişlemiş.

14

Alıştırma

Aşağıdaki komutlar işletildikten sonra AX ve DX yazmaçları ile Taşma bayrağının (OF) değerlerini gösterin.

```
mov ax,8760h
mov bx,100h
imul bx
```

DX = FF87h, AX = 6000h, OF = 1

15

IDIV Komutu

- IDIV (“signed divide”) komutu, işaretli tamsayı bölme gerçekleştirir.
 - DIV komutuyla aynı sözdizimine sahiptir.

- Geçerli işlenenler:

Dividend	Divisor	Quotient	Remainder
AX	<i>r/m8</i>	AL	AH
DX:AX	<i>r/m16</i>	AX	DX
EDX:EAX	<i>r/m32</i>	EAX	EDX

Örnek: $-48 / 5$ (8-bit işlenenler):

```
mov al,-48
cbw      ; AL'yi AH'ye genişlet
mov bl,+5
idiv bl   ; AL = -9, AH = -3
```

16

IDIV Komutu: Örnek

Örnek: $-48 / 5$ (16-bit işlenenler):

```
mov ax,-48
cwd      ; AX'i DX'e genişlet
mov bx,+5
idiv bx   ; AX = -9, DX = -3
```

Örnek: $-48 / 5$ (32-bit işlenenler):

```
mov eax,-48
cdq      ; EAX'i EDX'e genişlet
mov ebx,+5
idiv ebx  ; EAX = -9, EDX = -3
```

17

Alıştırma

Aşağıdaki komutlar işletildikten sonra AX ve DX yazmaçlarının değerlerini gösterin.

```
mov ax,0FDFFh ; -513
cwd
mov bx,100h
idiv bx
```

DX = FFFFh (-1), AX = FFFEh (-2)

18

İşaretsiz Aritmetik İfadeler

Örnek: $\text{var4} = (\text{var1} + \text{var2}) * \text{var3}$

```
; isaretsiz islenenler olduklarini varsayalim
mov eax,var1
add eax,var2 ; EAX = var1 + var2
mul var3     ; EAX = EAX * var3
jc TooBig   ; Elde'yi kontrol et
mov var4,eax ; carpimi sakla
TooBig:
```

19

Örnek: $\text{eax} = (-\text{var1} * \text{var2}) + \text{var3}$

```
mov eax,var1
neg eax
imul var2
jo TooBig ; Tasma'yi kontrol et
add eax,var3
jo TooBig ; Tasma'yi kontrol et
TooBig:
```

Örnek: $\text{var4} = (\text{var1} * 5) / (\text{var2} - 3)$

```
mov eax,var1 ; sol kisim
mov ebx,5
imul ebx     ; EDX:EAX = carpim
mov ebx,var2 ; sag kisim
sub ebx,3
idiv ebx     ; EAX = bolum
mov var4,eax
```

20

İşaretili Aritmetik İfadeler - 2

Örnek: `var4 = (var1 * -5) / (-var2 % var3);`

```
mov ax,var2          ; sag kisma basla
neg ax
cwd                  ; isaretle genisletilmis bolunen
idiv var3            ; EDX = kalan
mov bx,dx            ; EBX = sag kisim
mov ax,-5            ; sol kisma basla
imul var1            ; EDX:EAX = sol kisim
idiv bx              ; son bolme
mov var4,ax          ; bolum
```

Bazen bir ifadenin önce sağ tarafını hesaplamak kolaylık sağlar.

21

Alıştırma

Aşağıdaki ifadeyi hesaplayın.

`ax = (bx * 20) / cx`

```
mov ax,20
imul bx
idiv cx
```

22

Alıştırma

Aşağıdaki ifadeyi 32-bit işaretili tamsayı kullanarak gerçekleştirin. ECX ve EDX yazmaçlarını saklayın ve geri alın.

`eax = (ecx * edx) / eax`

```
push edx
push eax             ; EAX sonra kullanılacak
mov eax,ecx
imul edx             ; sol kisim: EDX:EAX
pop ebx              ; EAX'in saklanan degeri
idiv ebx             ; EAX = bolum
pop edx              ; EDX'i geri al
```

23

Alıştırma

Aşağıdaki ifadeyi 32-bit işaretili tamsayı kullanarak gerçekleştirin. “var3”den başka değeri değiştirmeyin.

`var3 = (var1 * -var2) / (var3 - ebx)`

```
mov eax,var1
mov edx,var2
neg edx
imul edx             ; sol kisim: EDX:EAX
mov ecx,var3
sub ecx,ebx
idiv ecx             ; EAX = bolum
mov var3,eax
```

24

Genişletilmiş Ekleme ve Çıkarma

25

Genişletilmiş Ekleme ve Çıkarma

- Genişletilmiş Ekleme
 - Genişletilmiş Duyarlıklı Ekleme (“Extended Precision Addition”)
 - ADC komutu
- Genişletilmiş Çıkarma
 - Genişletilmiş Duyarlıklı Çıkarma (“Extended Precision Subtraction”)
 - SBB komutu

26

Genişletilmiş Duyarlıklı Ekleme

- Bilgisayarın sözcük uzunluğundan daha uzun iki işleneni eklemek
 - İşlenen boyutu sanal olarak limitsiz
- Aritmetik 2 adımda gerçekleştirilir:
 - Her adımın Elde (CF) değeri bir sonraki adıma geçirilir.

27

ADC Komutu

- ADC (“add with carry”) komutu, kaynak işlenenle birlikte Elde bayrağının (CF) değerini hedef işlenene ekler.
- İşlenenler yine ikili değerlerdir
 - Sözdizimi ADD, SUB, vb. komutlarla aynıdır.
- Örnek:
 - İki adet 32-bit tamsayıyı ekle (FFFFFFFFh + FFFFFFFFh) ve sonucu 64-bit EDX:EAX yazmaç çiftinde tut.

```
mov edx,0
mov eax,FFFFFFFFh
add eax,FFFFFFFFh
adc edx,0 ; EDX:EAX = 00000001FFFFFFFFh
```

28

Genişletilmiş Ekleme: Örnek

- Görev: EDX:EAX çiftine 1 ekle
 - EDX:EAX başlangıç değeri : 00000000FFFFFFFFh
 - Önce alt yarıdaki 32 biti ekle (CF=1)
 - Ardından üst yarıdaki 32 biti, Elde (CF) değeri ile birlikte ekle

```
mov edx,0           ; üst yariyi ilklendir
mov eax,0FFFFFFFFh   ; alt yariyi illendir
add eax,1           ; alt yariyi ekle
adc edx,0           ; üst yariyi ekle
```

EDX:EAX = 00000001 00000000

29

SBB Komutu

- SBB (“subtract with borrow”) komutu, kaynak işlenenle birlikte Elde bayrağının (CF) değerini hedef işlenenden çıkarır.
- İşlenenler yine ikili değerlerdir.
 - Sözdizimi ADC komutuyla aynıdır.

30

Genişletilmiş Çıkarma: Örnek

- Task: Subtract 1 from EDX:EAX
 - EDX:EAX başlangıç değeri : 0000000100000000h
 - Önce alt yarıdaki 32 biti çıkar (CF=1)
 - Ardından üst yarıdaki 32 biti, Elde (CF) değeri ile birlikte çıkar

```
mov edx,1           ; üst yariyi ilklendir
mov eax,0           ; alt yariyi ilklendir
sub eax,1           ; alt yariyi cikar
sbb edx,0           ; üs yariyi cikar
```

EDX:EAX = 00000000 FFFFFFFF

31