



פרק 5 FLAGS, IP

- ▶ Special Purpose Registers
- ▶ הזכרנו אותם כשדיברנו על מבנה המחשב
- ▶ IP – Instruction Pointer
 - קפיצות
 - פרוצדורות, פסיקות
- ▶ FLAGS – דגלים
 - תנאים לוגיים
 - קפיצות מותנות

IP– Instruction Pointer

- ▶ IP מחזיק את הכתובת של הפקודה הבאה לביצוע
- ▶ העתיקו והריצו את הקוד הבא

IDEAL

MODEL small

STACK 100h

DATASEG

CODESEG

start:

mov ax, @data

mov ds, ax

mov ax, 1234h

mov bx, 0

mov bl, 34h

mov cx, 0

mov ch, 12h

quit:

mov ax, 4c00h

int 21h

END start

```

File Edit View Run Breakpoints Data Options Window Help READY
[ ]-CPU 80486- 2- [ ]
#base#start: mov ax, @data
cs:0000 B87B08 mov ax,087B
#base#8: mov ds, ax
cs:0003 8ED8 mov ds,ax
#base#9: mov ax, 1234h
cs:0005 B83412 mov ax,1234
#base#10: mov bx, 0
cs:0008 BB0000 mov bx,0000
#base#11: mov bl, 34h
cs:000B B334 mov bl,34
#base#12: mov cx, 0
cs:000D B90000 mov cx,0000
#base#13: mov ch, 12h
cs:0010 B512 mov ch,12
#base#quit: mov ax, 4c00h

ds:0000 00 00 00 00 00 00 00 00
ds:0008 00 00 00 00 00 00 00 00
ds:0010 00 00 00 00 00 00 00 00
ds:0018 00 00 00 00 00 00 00 00
ds:0020 00 00 00 00 00 00 00 00

ax 087B c=0
bx 0000 z=0
cx 0000 s=0
dx 0000 o=0
si 0000 p=0
di 0000 a=0
bp 0000 i=1
sp 0100 d=0
ds 087B
es 0869
ss 087B
cs 0879
ip 0005

ss:0108 0004
ss:0106 0000
ss:0104 0019
ss:0102 0403
ss:0100 52FB
  
```

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

IP - דוגמה (המשך)

```

File Edit View Run Breakpoints Data Options Window Help READY
[CPU 80486]
#base#start: mov ax, @data
cs:0000 B87B08 mov ax,087B
#base#8: mov ds, ax
cs:0003 8ED8 mov ds,ax
#base#9: mov ax, 1234h
cs:0005 B83412 mov ax,1234
#base#10: mov bx, 0
cs:0008 BB0000 mov bx,0000
#base#11: mov bl, 34h
cs:000B B334 mov bl,34
#base#12: mov cx, 0
cs:000D B90000 mov cx,0000
#base#13: mov ch, 12h
cs:0010 B512 mov ch,12
#base#quit: mov ax, 4c00h

ds:0000 00 00 00 00 00 00 00 00
ds:0008 00 00 00 00 00 00 00 00
ds:0010 00 00 00 00 00 00 00 00
ds:0018 00 00 00 00 00 00 00 00
ds:0020 00 00 00 00 00 00 00 00

ax 1234 c=0
bx 0000 z=0
cx 0000 s=0
dx 0000 o=0
si 0000 p=0
di 0000 a=0
bp 0000 i=1
sp 0100 d=0
ds 087B
es 0869
ss 087B
cs 0879
ip 0008

ss:0108 0004
ss:0106 0000
ss:0104 0019
ss:0102 0403
ss:0100 52FB
  
```

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

שנו את IP באופן ידני כך שבשום שלב ax לא יהיה שווה 1234h

The screenshot shows a DOS-based assembly debugger (likely DOSBox) with the following components:

- Menu Bar:** File, Edit, View, Run, Breakpoints, Data, Options, Window, Help.
- Register Window (Right):**
 - ax: 087B, c: 0
 - bx: 0000, z: 0
 - cx: 0000, s: 0
 - dx: 0000, o: 0
 - si: 0000, p: 0
 - di: 0000, a: 0
 - bp: 0000, i: 1
 - sp: 0100, d: 0
 - ds: 087B
 - es: 0869
 - ss: 087B
 - cs: 0879
 - ip: 0005 (highlighted)
- Code Window (Left):**

```

[CPU 80486]
#base#start: mov ax, @data
cs:0000 B87B08      mov ax,087B
#base#8:  mov ds, ax
cs:0003 8ED8        mov ds,ax
#base#9:  mov ax, 1234h
cs:0005 B83412      mov ax,1234
#base#10: mov bx, 0
cs:0008 BB0000      mov bx,0000
#base#11: mov bl, 34h
cs:000B B334        mov bl,34
#base#12: mov cx, 0
cs:000D B90000      mov cx,0000
#base#13: mov ch, 12h
cs:0010 B512        mov ch,12
#base#quit: mov ax, 4c00h

es:0000 CD 20 7D 9D 00 EA FF FF =
es:0008 AD DE 32 0B C3 05 6B 07 ;
es:0010 14 03 28 08 14 03 92 01 ;
es:0018 01 01 01 00 02 04 FF FF ;
es:0020 FF FF FF FF FF FF FF FF

```
- Dialog Box:** "Enter new value" with a text input field and buttons: OK, Clip..., Cancel, Help.
- Status Bar:** "Enter item prompted for in dialog title"

שיעורי בית

...

Start:

```
mov ax, @data
mov ds, ax
mov ax, 14
add ax, 1
add ax, 2
add ax, 4
```

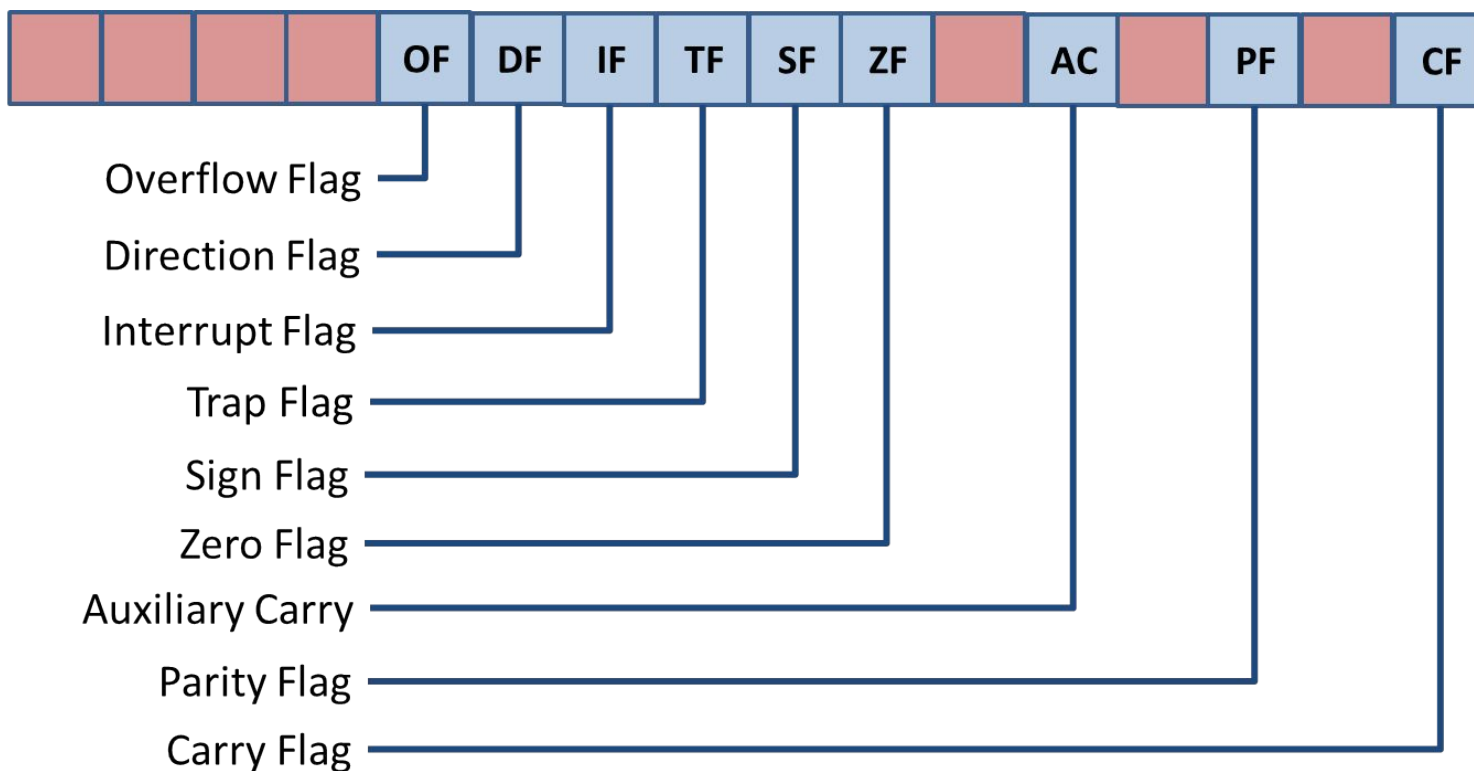
quit:

...

שנו את IP באופן ►
ידני כך שבסוף
התוכנית ערכו של
ax יהיה 16

Processor Status Register – FLAGS

- ▶ שונה מיתר הרגיסטרים
- ▶ 16 ביט – כל ביט הוא אות חשמלי נפרד



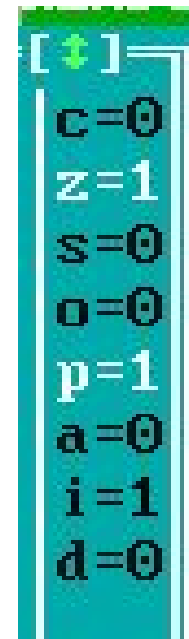
דגלי הבקרה Control Flags

- ▶ חלק מהדגלים הם Control Flags
 - משמשים לבדיקת תנאים לוגיים ופעולות בקרה ("אם תנאי מתקיים אז בצע...")
 - דגל האפס Zero Flag
 - דגל הגלישה Overflow Flag
 - דגל הנשא Carry flag
 - דגל הסימן Sign Flag
- ▶ נראה את השימוש בהם כשנלמד פקודות בקרה וקפיצה

דגל האפס Zero Flag

- ▶ דגל האפס מקבל 1 אם לאחר הרצת הפקודה האחרונה אופרנד היעד התאפס
 - אופרנד היעד – המקום בו נשמרת התוצאה. רגיסטר או מקום בזיכרון.
- ▶ דוגמה:

- ▶ `mov al, 4Bh ; 75 decimal`
- ▶ `mov ah, 4Bh ; 75 decimal`
- ▶ `sub al, ah ; subtract al minus ah, result is 0`



דגל האפס Zero Flag

דוגמה נוספת: ▶

```
mov    al, 0FFh    ; 255 decimal
mov    ah, 01h    ; 1 decimal
add    al, ah      ; add al and ah, result is 256
```

הריצו ובידקו את מצב הדגלים ▶

תרגילי כיתה

► צרו דוגמה שמדליקה את דגל האפס בעזרת פעולת **חיסור**. השתמשו ברגיסטרים של 16 ביט.

► צרו דוגמה שמדליקה את דגל האפס בעזרת פעולת **חיבור**. השתמשו ברגיסטרים של 16 ביט.

דגל הנשא – Carry Flag

- מקבל את הערך 1 אם תוצאת הפעולה חורגת מתחום של מספרים unsigned. המעבד מתייחס אל כל הערכים בחישוב בתור unsigned. תחום החריגה:
 - עבור 8 ביט: חריגה מהתחום 0-255
 - עבור 16 ביט: חריגה מהתחום 0-65,535

```
mov    al, 0C8h    ; 200 decimal
mov    ah, 64h; 100 decimal
add    al, ah      ; result is 300,
                  ; out of 8 bit range
```

[+]
c=1
z=0
s=0
o=0
p=0
a=0
i=1
d=0

► צרו דוגמה שמדליקה את דגל הנשא.
השתמשו ברגיסטרים של 16 ביט.

דגל הגלישה – Overflow Flag

► דומה לדגל הנשא, אך מקבל את הערך 1 אם תוצאת הפעולה חורגת מתחום של מספרים signed. התחום תלוי בגודלו של אופרנד היעד:

- עבור 8 ביט: חריגה מהתחום -128 to $+127$
- עבור 16 ביט: חריגה מהתחום: ? (חשבו בעצמכם!)
- דוגמה (הריצו ובידקו את מצב הדגלים):

```
mov    al, 64h ; 100 decimal
mov    ah, 28h ; 40 decimal
add    al, ah   ; result is 140, out of
               ; SIGNED 8 bit range
```

► צרו דוגמה שמדליקה את דגל הגלישה.
השתמשו ברגיסטרים של 16 ביט.

שאלה למחשבה

▶ האם הפקודות הבאות ידליקו את דגל הגלישה?

```
mov    ax, 0
```

```
mov    bx, 8888h    ; same as:  
                        ; mov bx, 34952
```

```
sub     ax, bx
```

▶ ומה תהיה השפעתן על דגל הנשא?

פתרון חלק א'

- ▶ כשהמעבד בודק אם להדליק את דגל הגלישה overflow, הוא מתייחס לכל הערכים בפעולה בתור signed.
- ▶ לכן 8888h הוא למעשה מספר שלילי (איך יודעים? הביט השמאלי הוא 1)
- ▶ על פי שיטת המשלים ל-2, נמצא את ערכו של 8888h על ידי מציאת ערכו של הנגדי החיובי:

1000 1000 1000 1000 →

0111 0111 0111 1000 = 30564

כלומר 8888h הינו -30564, ולכן תוצאת החיסור שלו מאפס היא +30564, ערך זה אינו חורג מטווח הערכים שמדליקים את דגל הגלישה.

פתרון חלק ב'

- ▶ כשהמעבד בודק אם להדליק את דגל הנשא carry, הוא מתייחס לכל הערכים בפעולה בתור unsigned.
- ▶ לכן 8888h, שווה ל-34952+.
- ▶ כאשר המחסר (במקרה זה 8888h) גדול מהמחוסר (במקרה זה 0) ישנה חריגה מהתחום שמוגדר לדגל הנשא.

[■]—CPU 80486			1	[▲]
cs:0000	B87B08	mov ax,087B	ax 7778	c=1
cs:0003	8ED8	mov ds,ax	bx 8888	z=0
cs:0005	B80000	mov ax,0000	cx 0000	s=0
cs:0008	BB8888	mov bx,8888	dx 0000	o=0
cs:000B	2BC3	sub ax,bx	si 0000	p=1
cs:000D	B8004C	mov ax,4C00	di 0000	a=1
cs:0010	CD21	int 21	bp 0000	i=1

דגל הסימן Sign Flag

- ▶ יקבל את הערך 1 אם הביט המשמעותי (השמאלי) של התוצאה הוא 1.
- ▶ דרכים לראות שמספר signed הוא שלילי:
 - בבינארי- הביט השמאלי הוא 1
 - בהקסדצימלי- ה-Nibble השמאלי הוא בין 8 ל-F
- ▶ צרו תוכנית שמדליקה את דגל הסימן

► צרו דוגמה שמדליקה את דגל הסימן.

שיעורי בית א'

▶ האם ניתן לכתוב קוד שעל ידי פעולה אחת של חיבור או חיסור, מדליק בו זמנית את דגל הגלישה, את דגל הנשא ואת דגל הסימן? (אם כן – תנו דוגמה)

פתרון שיעורי הבית א'

▶ נתבונן בקוד הבא:

```
mov    ax, 0
mov    bx, 8000h
sub    ax, bx
```

▶ דגל הנשא carry יידלק כיוון שתוצאת החיסור של מספר חיובי מאפס היא שלילית – חורגת מהתחום

▶ דגל הגלישה overflow יידלק משום שהתרגום של 8000h למספר signed הוא -32768. אפס פחות 8000h שווה +32768, חורג (באחד...) מהתחום של דגל הגלישה.

▶ דגל הסימן sign יידלק משום שהביט השמאלי של התוצאה הוא 1

c	=1
z	=0
s	=1
o	=1
p	=1
a	=0
i	=1
d	=0

שיעורי בית ב'

▶ האם ניתן לכתוב קוד שעל ידי פעולה אחת של חיבור או חיסור, מדליק בו זמנית את דגל הגלישה, את דגל הנשא ואת דגל האפס? (אם כן – תנו דוגמה)

פתרון שיעורי הבית ב'

▶ נתבונן בקוד הבא:

```
mov    ax, 8000h
mov    bx, 8000h
add    ax, bx
```

```
c=1
z=1
s=0
o=1
p=1
a=0
i=1
d=0
```

▶ דגל הנשא carry יידלק כיוון שתוצאת החיבור של המספרים – 10000h או 65536d, חורגת מהתחום (כלל לא ניתנת לייצוג על ידי 16 ביט).

▶ דגל הגלישה overflow יידלק משום שהתרגום של 8000h למספר signed הוא -32768d. תוצאת החיבור היא -65536d, חורגת מהתחום (כנ"ל, חורגת מתחום הייצוג של 16 ביט)

▶ דגל האפס יידלק משום שכל 16 הביטים הימניים בתוצאה הינם 0, כלומר הרגיסטר ax יתאפס.

- ▶ דגל ה-IP - מצביע על ה-Opcode הבא
- ▶ FLAGS - סטטוס המעבד
 - Control Flags נקבעים על סמך החישוב האחרון שבוצע
 - דגל האפס Zero Flag
 - דגל הגלישה Overflow Flag
 - דגל הנשא Carry flag
 - דגל הסימן Sign Flag
 - בהמשך נראה שימושים ל(רוב) הדגלים