



ארכיטקטורה אסמבלי



הדליקו מצלמות
כבו מיקרופונים
השתיקו טלפונים
ארגנו מחברת וכלי כתיבה

שיעור 6

פסיקות וקלט- פלט



ארכיטקטורה
אסמבלי



ארכיטקטורה
**מערכות
הפעלה**

מבוא לאסמבלי

אסמבלי – משתנים ופקודות

תנאים ולולאות

מחסנית ופונקציות

פרמטרים ופרוצדורות

פסיקות

סיכום אסמבלי

תרגיל בספות

מבוא למערכות הפעלה

כלי דיאגנוסטיקה ל-Windows

Processes and Threads

Memory

Linux shell

Shell המשך

מערכות קבצים

Bootstrapping

פרויקט סיכום מערכות הפעלה

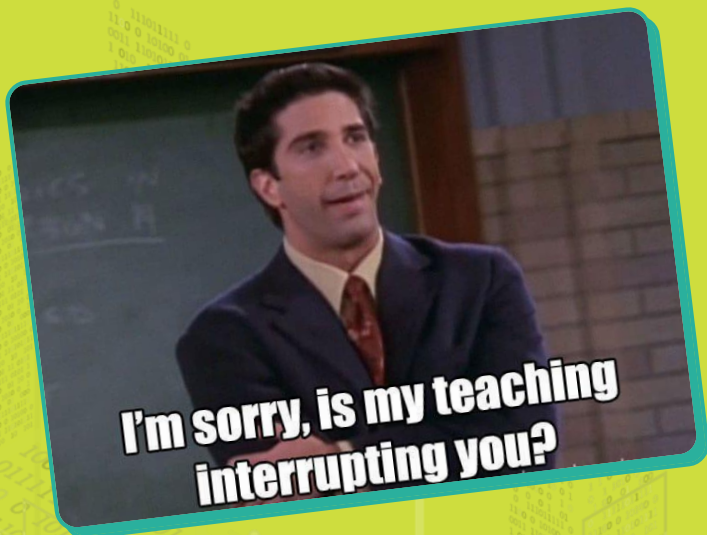


בשיעור הקודם

למדנו על תנאים ולולאות באסמבלי

- למדנו כמעט
- וגם נהנו
- תרגלנו את כתיבת





שיעור 6

פסיקות וקלט-פלט

קלט \ פלט

כתיבת שגרת
פסיקה

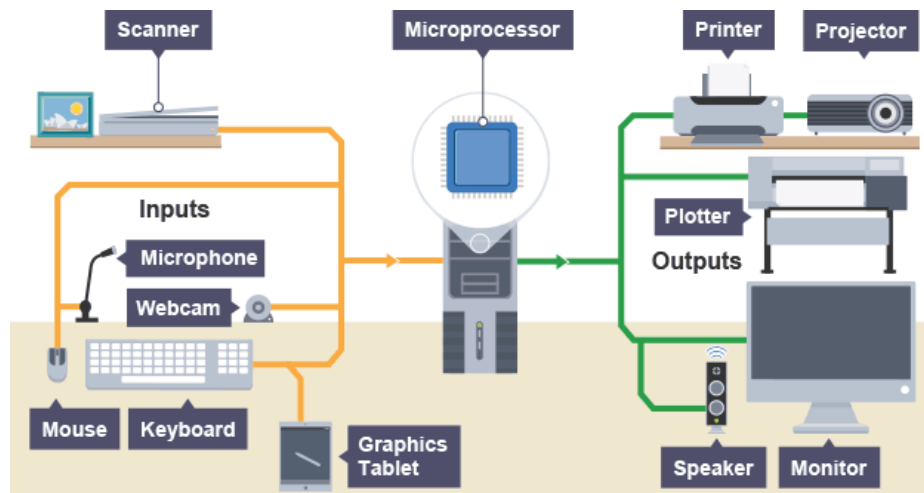
פסיקת תכונה

פסיקות

חזרה

עבודה עם חומרה

בכדי ליצור חוויית משתמש עשירה, תוכנת מחשב יכולה לעשות שימוש ברכיבי חומרה שונים – מקלדת, מסך, כרטיס קול, כרטיס רשת ורכיבים נוספים המחוברים למחשב



טיפול באירועים

אתגר משמעותי: אירועי חומרה יכולים לקרות ברגע בלתי צפוי – כלומר בכל שלב במהלך ריצת התכנית

למשל: הגיעה חבילת רשת, המשתמש הזיז את העכבר..

אפשרות ראשונה:

התכנית תחכה לאירוע (בדומה ל scanf ב – c)

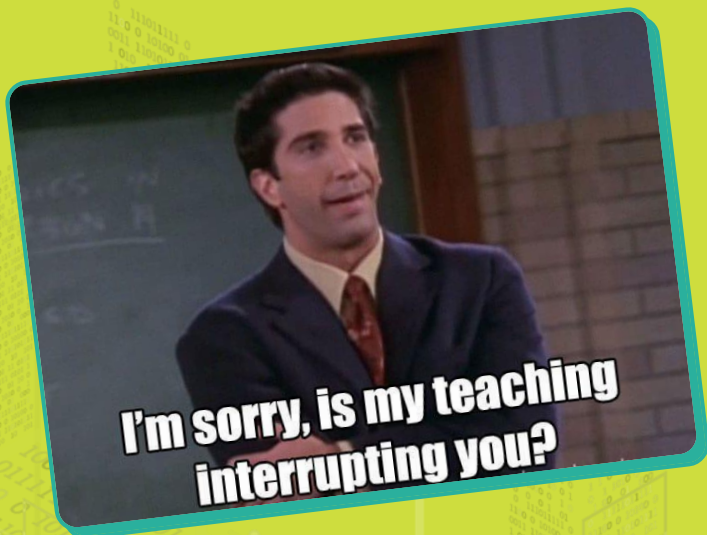
האם זה רעיון טוב?

אפשרות שנייה: נגיב לאירועים

יש לכם רעיון איך?



מה דעתכם?
דברו



שיעור 6

פסיקות וקלט-פלט

קלט \ פלט

כתיבת שגרת
פסיקה

פסיקת תכונה

פסיקות

חזרה

תיאור כללי



איך זה עובד?

IP	INT 0 – Divide error
CS	
IP	INT 1 – Single stepping
CS	
IP	INT 2 – NMI
CS	
IP	INT 3 – Breakpoint
CS	
IP	INT 4 – Interrupt on overflow
CS	
IP	INT 5
CS	
...	
IP	INT 31
CS	
...	
IP	INT 255
CS	

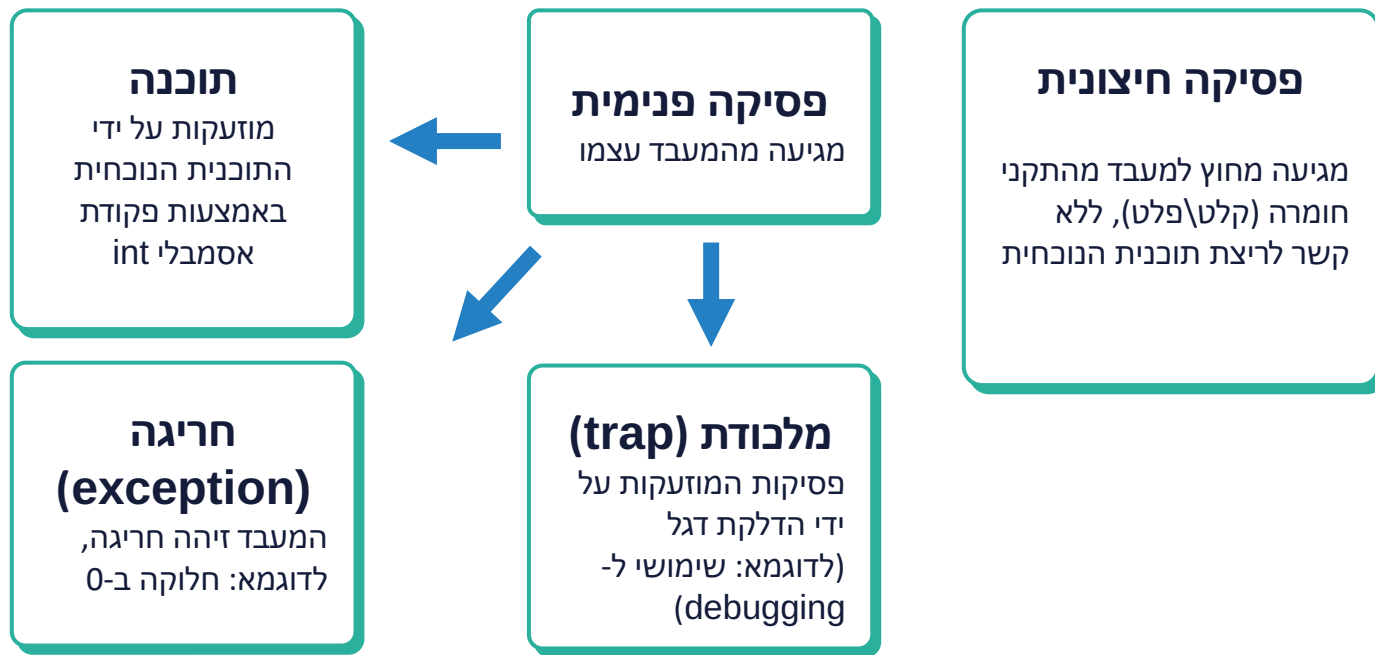
פסיקות ייעודיות

שמורות

בשימוש משתמש

המעבד מחזיק טבלה של כתובות לפונקציות
מגיעה הודעה/בקשה
תוכנית אשר רצה כרגע במעבד נעצרת
נקראת הפונקציה שיודעת לטפל בהודעה
הפונקציה מטפלת בבקשה
זמן הריצה מוחזר לתוכנית שנעצרה

סוגי הפסיקות



טיפול בפסיקה

המעבד מקבל בקשות פסיקה ממקורות שונים, וכל מקור דורש
טיפול שונה

על מקור הפסיקה להודיע למעבד מהו סוג הפסיקה המבוקש
המעבד יפעיל את הפונקציה ע"י מציאת כתובתה בטבלת וקטור
הפסיקות (Interrupt Vector Table)



איך יודעים לחזור לנקודה המדויקת בתכנית בה היא הופסקה?

המחסנית כמובן

בעת ביצוע פסיקה, המעבד דוחף למחסנית את CS, IP של התכנית שהופסקה

כאשר הטיפול בפסיקה מסתיים, המעבד מוציא מהמחסנית את CS, IP אליהם הוא קופץ חזרה

למה צריך גם את CS ?

זאת מכיוון שפונקציית המטפלת בפסיקה יכולה להימצא מחוץ למקטע הקוד הנוכחי של התוכנית

	00F8
	00FA
	00FC
IP	00FE
CS	0100

טיפול בפסיקה

```
mov ax,0  
cmp bx,ax  
je bigger
```

מה יקרה אם תופעל פסיקה
בין שתי הפקודות האלה,
ובזמן הטיפול בה, ישתנה
הערך של ax?

מה יקרה אם תופעל פסיקה
בין שתי הפקודות האלה,
ובזמן הטיפול בה, ישתנה
ZF של ?



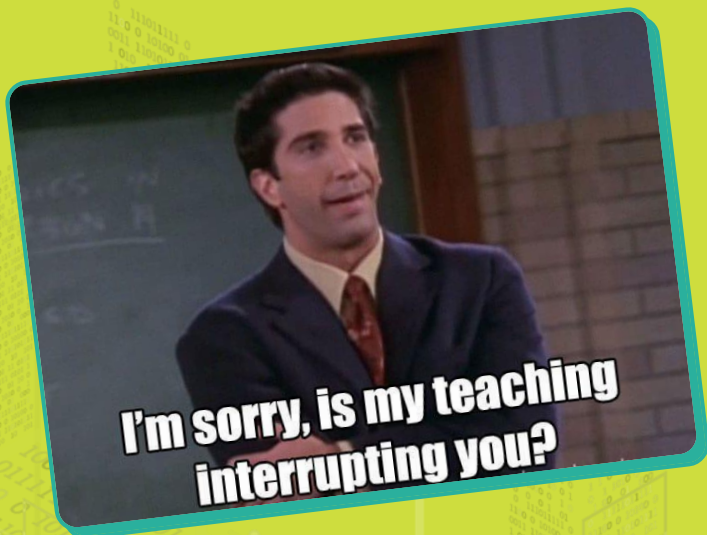
תהליך הטיפול בפסיקה חייב לשמור ערכי כל האוגרים, איפה נשמור?

	00F8
IP	00FA
CS	00FC
FLAGS	00FE
	0100

המחסנית כמובן

בעת ביצוע פסיקה, המעבד דוחף למחסנית את CS, IP של התכנית שהופסקה ואת אוגר הדגלים

שגרת הפסיקה תשמור את כל האוגרים בעזרת **PUSHA** בסיומה, שגרת הפסיקה תשחזר את כל האוגרים בעזרת **POPA** כאשר הטיפול בפסיקה מסתיים, המעבד מוציא מהמחסנית את אוגר הדגלים, ואת CS, IP אליהם הוא קופץ חזרה



שיעור 6

פסיקות וקלט-פלט

קלט \ פלט

כתיבת שגרת
פסיקה

פסיקת תכונה

פסיקות

חזרה

פסיקת תוכנה

הפעלת פסיקות תוכנה מתבצעת באמצעות האופקוד INT (interrupt)
אופקוד זה מקבל אופרנד יחיד, המייצג את סוג הפסיקה המבוקש



```
int 13h
```

רשימת כל סוגי הפסיקות הממומשים ע"י emu8086 ניתן למצוא בקובץ
[interrupts for 8086 emulator.html](http://interrupts.for8086.com/html) (לינק)

עבודה עם חומרה

בשביל לתקשר עם חומרה צריך:

- לדעת איך ניתן לגשת אליה
- לתאם עם החומרה מתח, זרם, קצב עבודה
- ועוד

אז איך עשינו את כל זה עד עכשיו?

השתמשנו בקובץ `magshimim.inc`

קוד בקובץ הזה הכיל קריאה לשגרות השירות של מערכות הפעלה שנועדו לשחרר את המתכנת מהצורך לחשוב על איך לתקשר עם החומרה

עבודה עם חומרה

האמולטור emu8086 מכיל תמיכה בהפעלת שירותים של מערכת ההפעלה MS-DOS

- לצורך כך ניתן להשתמש בפסיקה מיוחדת

❖ **21h** משמשת להעברת מידע למערכת ההפעלה

- הפונקציה שמופיעה בטבלה לטיפול בפסיקה זו, היא פונקציה של מערכת ההפעלה וכך יכולה תוכנה לתקשר עם מערכת ההפעלה בשביל שזו תקשר בשמה עם החומרה
- העברת פרמטרים לפסיקה **21h** מתבצעת באמצעות אוגר AX



קריאה לפסיקה

פסיקת 21h

לפסיקה זו נעביר את הפרמטרים בצורה הבאה:
AH יכיל את המספר של השירות המבוקש ממערכת ההפעלה
AL יכיל את הפרמטר עבור השירות המבוקש

```
mov al,0    ; מייצג סיום בהצלחה  
mov ah,4ch  ; שירות המבצע סיום התוכנית  
int 21h     ; זימון פסיקה המייצגת בקשה של שירות מערכת הפעלה
```



פסיקת 21h

מספר דוגמאות חשובות לשימוש
בפסיקה

הדפסת תו

```
mov dl,'a'  
mov ah,2  
int 21h
```

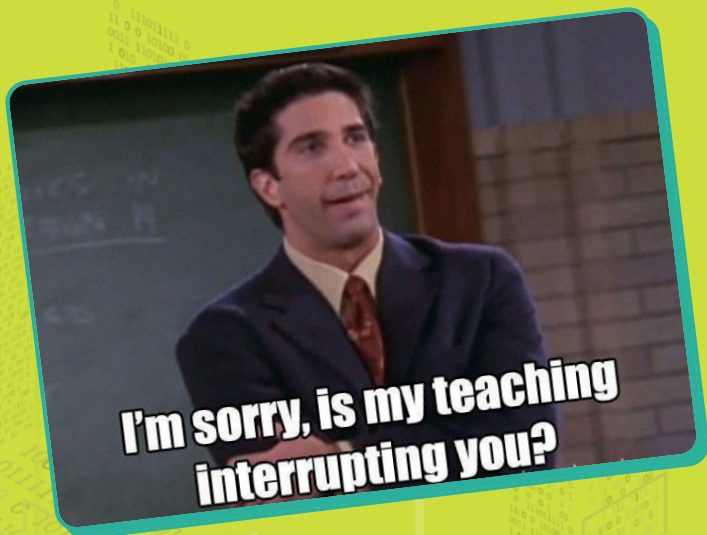
הדפסת מחרוזת

```
mov dx, offset msg  
mov ah,9  
int 21h
```

קלט תו

```
mov ah,1  
int 21h
```





שיעור 6

פסיקות וקלט-פלט

קלט \ פלט

כתיבת שגרת
פסיקה

פסיקת תכונה

פסיקות

חזרה

כתיבת שגרת פסיקה

```
proc my_interrupt  
pusha  
...  
popa  
iret  
endp my_interrupt
```

נגדיר את שגרת פסיקה (ISR) כפונקציה רגילה
להקפיד על שמירה ושחזור אוגרים
בסיום, לחזור בעזרת IRET (Interrupt Return)
דומה ל- ret אך בנוסף ל- IP משחזרת גם את ערכם של CS ו-FLAGS

עדכון טבלת וקטור הפסיקות (IVP)

התכנית שלנו נמצאת במקטע כלשהו (שאיננו 0)
כל אוגרי המקטע (DS, CS) מצביעים למקטע זה
טבלת הפסיקות נמצאת במקטע שבכתובת 0
כדי לכתוב לתוך מקטע אחר, נוכל להשתמש באוגר המקטע ES בתור
"מקטע עזר" באופן הבא:

```
mov ax,0h
mov es,ax
mov es:[12h],55h ;update ip (offset)
mov es:[14h],102h ;update cs (base)
```


קריאות מקוננת לפסיקה, בעיה?

בעת ביצוע של שגרת פסיקה עלולה להגיע פסיקה נוספת
במצב זה שגרת השרות עצמה תופסק ותקרא שגרת שרות לטיפול בפסיקה האחרונה
תופעה זו יכולה לחזור שוב ושוב

my_interrupt
my_interrupt
my_interrupt



מה דעתכם?
דברו

מיסוך פסיקות

כדי לטפל בפסיקה ללא הפרעה, בעת הפעלת שגרת השרות, המעבד יכבה את הדגל IF (Interrupt Flag).
דגל IF קובע האם המעבד יענה לבקשת פסיקת חומרה.

דלוק = יענה

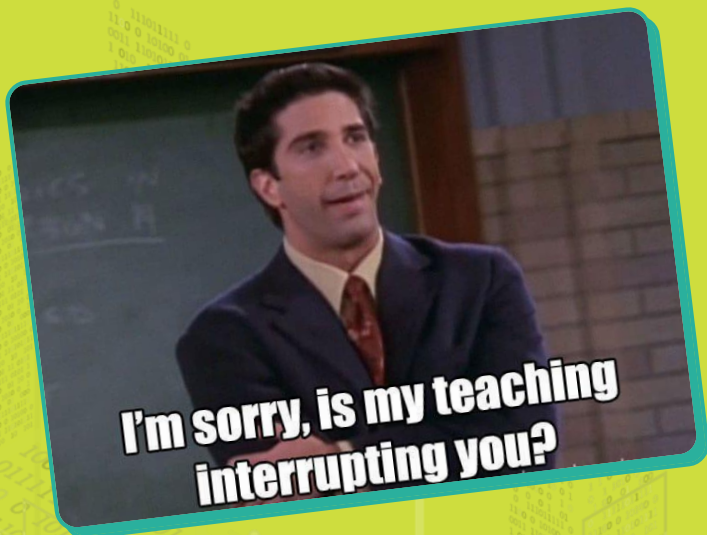
כבוי = יתעלם

באילו עוד מקרים טיפול בפסיקה יכול להפריע?

בזמן החלפת שגרת הפסיקה בחדשה

עדכון טבלת וקטור הפסיקות (IVP)

מיסוך פסיקות בזמן העדכון ע"י STI/CLI
עדכון נכון הן של כתובת המקטע והן של ההיסט
במידת הצורך: שמירת כתובת השגרה המקורית



שיעור 6

פסיקות וקלט-פלט

קלט \ פלט

כתיבת שגרת פסיקה

פסיקת תכונה

פסיקות

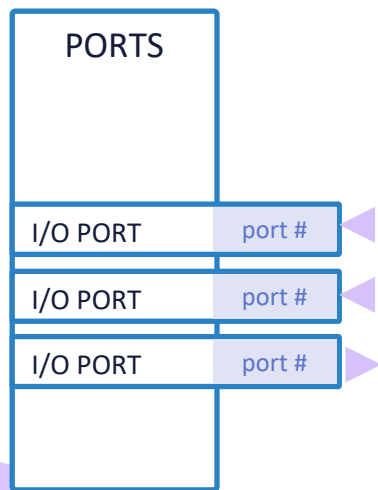
חזרה

קלט\פלט

בדומה לזיכרון הראשי, גם התקני קלט\פלט מחוברים אל המעבד באמצעות הפסים (buses)

לכל התקן מוקצה כתובת ייחודית הנקראת port
גודל מרחב הכתובות של הקלט\פלט במעבד 8086 הוא 64 kilobytes

מרחב הכתובות קלט\פלט לא חופף למרחב כתובות של הזיכרון הראשי
גישה לכתובות קלט\פלט (ports) מתבצעת באמצעות הפקודות IN ו- OUT (בלבד!)



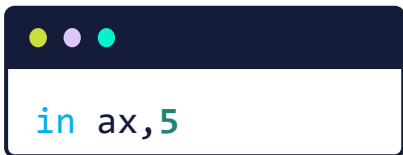


פקודה IN

משמשת להעברת מידע מהתקן קלט למעבד
מבנה הפקודה:

IN <אופרנד מקור>, <אופרנד יעד>

אופרנד מקור מכיל את מספר הכניסה (port)
אופרנד יעד הוא אוגר AL או AX שיאחסן את הנתון



```
in ax,5
```



פקודה OUT

משמשת להעברת מידע מהמעבד להתקן הפלט

OUT <אופרנד יעד>, <אופרנד מקור>

אופרנד מקור הוא אוגר AL או AX שיאחסן את הנתון

אופרנד יעד מכיל את מספר הכניסה (port)

כתיבת הערך 1234h לכניסה מספר 13

```
MOV ax, 1234h
```

```
OUT 13, ax
```

```
mov ax, 1234h
out 13, ax
```

תרגול כיתה 1

כתבו תכנית שתציג על הצג את התוכן של מחרוזת המכילה את שמכם
הגדירו משתנה שיכיל את שמכם ובסיום רשמו "\$"
כתבו שגרה בשם `printString` שתקבל כפרמטר מצביע לתחילת מחרוזת ותציג את המחרוזת על הצג



תרגול כיתה 2

כתבו תכנית שתקלוט מהמשתמש תו ותציג אותו על הצג.
התכנית תשתמש בשגרה `getChar` הקולטת ומחזירה תו
ובשגרה `printChar` שהוצגה קודם לכן להצגת התו





ביצוע Hands-On
*נמצא בתיקיית החניכים



סיכום שיעור



תרגיל הבית



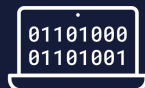
מעבר נוסף
על המצגת

איך כדאי לחזור
על החומר?

שאלות? תהיות?

תכתבו עכשיו שאלות בצ'ט

ארכיטקטורה
אסמבלי



מגזר ימים
תכנית הסייבר הלאומית

המרכז לחינוך סייבר
CYBER EDUCATION CENTER



שיעור 6 סיימנו!!!

קלט \ פלט

כתיבת שגרת
פסיקה

פסיקת תכונה

פסיקות

חזרה