

## 1. מה זה ארכיטקטורת מחשב?

- תכנון המבנה, התפקוד והאינטראקציה בין רכיבי המחשב, כולל המעבד, הזיכרון, והתקני קלט/פלט.

## 2. ארכיטקטורת Von Neumann לעומת Harvard - יתרונות וחסרונות:

- Von Neumann: זיכרון משותף לתוכנה ולנתונים → גמישות גבוהה אך צוואר בקבוק בגישה לזיכרון.
- Harvard: זיכרון נפרד לתוכנה ולנתונים → ביצועים מהירים יותר עקב גישה מקבילית, אך מורכב יותר ליישום.

## 3. באיזו ארכיטקטורה בנוי 8086?

- Von Neumann.

## 4. ייעוד של זיכרון ראשי (RAM):

- אחסון זמני של נתונים ותוכניות בזמן ריצה כדי לאפשר גישה מהירה למעבד.

## 5. מה זה BIOS? על איזה סוג של זיכרון הוא נשמר?

- מערכת בסיסית לאתחול המחשב ולבדיקות חומרה. נשמר בזיכרון ROM או Flash Memory.

## 6. תפקידים של פס כתובות, פס נתונים וקווי בקרה:

- פס כתובות: קובע כתובת בזיכרון או בהתקן קלט/פלט.
- פס נתונים: מעביר נתונים בין המעבד לרכיבים אחרים.
- קווי בקרה: שולטים בפעולות המעבד וכתחמום הערכת נתונים.

## 7. למה מתכוונים הרגליים של 8086 בפרט ורכיב אלקטרוני בכלל?

- הפינים שמאפשרים תקשורת בין הרכיב לרכיבים אחרים.

## 8. מה המרכיבים העיקריים של 8086?

- יחידת בקרה, יחידת ALU, רגיסטרים, זיכרון מטמון, פסי נתונים וכתובות, ויחידת קדם-שליפה (Prefetch Unit).

## 9. מה זה Multiplexed בהקשר של 8086?

- שילוב של פסי כתובות ונתונים על אותם פינים לחיסכון בכמות החיבורים.

## 10. מה מאפשר פס נתונים של Byte, Word, Double Word?

- גישה ליחידת מידע בגודל 8 ביטים (Byte), 16 ביטים (Word), או 32 ביטים (Double Word).

## 11. מה מאפשר פס כתובות של Byte, Word, Double Word?

- קובע את כמות הזיכרון הנגיש; כ-8086 פס הכתובות הוא 20 ביטים ומאפשר גישה עד 1MB.

## 12. למה מחשב צריך CLOCK?

- לתזמון פעולות המעבד, קביעת מהירות העיבוד וסנכרון הרכיבים.

## 13. מה תפקידים של אוגרים במעבד?

- אחסון זמני של נתונים, כתובות, תוצאות חישובים וניהול זרימת התוכנית.

## 14. מה זה שפת אסמבלי? מה ההבדל בין אסמבלר ל-C או ל-C#?

- שפת תכנות נמוכה עם פקודות ישירות למעבד. אסמבלר מתרגם קוד אסמבלי לשפת מכונה, בעוד ש-C ו-C# הן שפות עיליות המאפשרות תכנות מתקדם יותר.

## 15. אילו סוגי רגיסטרים יש ב-8086? מה תפקידם?

- רגיסטרים כלליים (AX, BX, CX, DX): משמשים לאחסון נתונים וחישובים.
- רגיסטרים מיוחדים (IP, FLAGS): מנהלים את זרימת התוכנית ומאחסנים דגלי מצב.
- רגיסטרים סגמנטיים (CS, DS, SS, ES): אחראים לחלוקת הזיכרון.
- רגיסטרים של המחסנית (SP, BP): משמשים לניהול קריאות פונקציות.
- 16. מה זה מחסנית בתוכנות? למה היא משמשת? איך היא עובדת?
- מבנה נתונים מסוג LIFO (האחרון שנכנס – הראשון שיוצא), משמש לניהול קריאות פונקציות, אחסון פרמטרים זמניים ושמידת כתובות חזרה, פועל עם PUSH (הכנסה) ו-POP (שליפה).

המרה מ base 16

base 10 -

$$\begin{array}{r} 205 \mid 5 \\ 41 \mid 0 \end{array}$$



$$xyz_{Base 8} = 321_8$$

$$x \cdot 8 + y = \square$$

$$\square \cdot 8 + z = Answer$$

205	5
41	0
8	1
1	3
0	1



5. (נק') בצעו המרה של מספר בינארי 11001101  
 Base 2 = 11001101  
 Base 4 =  
 Base 8 =  
 Base 16 =  
 Base 10 = 205  
 Base 3 =  
 Base 5 = 1310

6. (נק' 10) בעזרת השיטה "משלים ל-2" בצעו המרה של מספר דצימלי -115  
 Base 10 = -105  
 Base 2 =

105	2
52	1
26	0
13	0
6	1
3	0
1	1
0	1

01101001  
 10010111

← Add up to 8 bit!  
 Flip then +1

7. (נק') מצאו סכום ומכפלה של זוג מספרים חיוביים בינאריים.

בחיבור אין אנסוף כסף

$$10111 * 10011 =$$

10111  
 101110  
 1011100  
 10111000  
 101110000  
 110110101

8. (נק') ביצעו חילוק של מספרים בינאריים. התוצאה ומה השארית?

a.  $11001 / 101 = 101$   $11001 \% 101 = 0$

$\geq$   
 $11001 \div 101 = 101$

$$\begin{array}{r}
 1001 \div 101 \mid 101 \\
 \hline
 101 \downarrow \downarrow \\
 \hline
 101 \\
 - 101 \\
 \hline
 0
 \end{array}$$

השלמת חומר:

Syntax for a variable declaration:

name **DB** value

name **DW** value

**DB** - stays for Define Byte.

**DW** - stays for Define Word.

name - can be any letter or digit combination, though it should start with a letter. It's possible to declare unnamed variables by not specifying the name (this variable will have an address but no name).

value - can be any numeric value in any supported numbering system (hexadecimal, binary, or decimal), or "?" symbol for variables that are not initialized.

```

1.  msg: DB "ABBA abba"      ; store string
2.  OPR_BYTE: DB 16
3.  RESULT1: DW 0            ; declare place to store result
4.  RESULT2: DW 0
5.  OPR1: DW 0x0102          ; declare first number
6.  OPR2: DW 0x090A          ; declare second number
7.  start:
8.  MOV AX, word OPR1         ; move first number to AX
9.  MOV BX, word OPR2         ; move second number to BX
10. CLC                      ; clear the carry flag
11. ADD AX, BX                ; add BX to AX
12. MOV DI, OFFSET RESULT1    ; move offset of result1 to DI
13. MOV word [DI], AX         ; store result1
14. SUB AX, BX                ; sub BX from AX
15. INC AX                   ; add 1 to AX
16. DEC BX                   ; dec 1 from BX
17. SUB BX, AX                ; sub ax from bx
18. MOV DI, OFFSET RESULT2    ; move offset of result2 to DI
19. MOV word [DI], BX         ; store result2
20. print reg                 ; print reg

```



$$\begin{array}{r} 0102 \\ + 090A \\ \hline 0A0C \end{array}$$
  
 ↓
   
 AX

9. (נק') מה תהיה פריסת הזיכרון לאחר קימפול של התוכנית?

(טבלת ה-ASCII נמצאת בנספח בסוף טופס המבחן)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
41	42	42	41	20	61	62	62	61	20	0	0	0	0	02	01
-----															
16	17	18	19	20	21	22	23	24	25	...					
0A	09														
-----															

0A0C  
-090A  
0102 AX

+  
1  
0103 AX  
0909 BX

0909  
-0103  
0806

א' (נק') מה יהיה תוכן הרגיסטרים אחרי ביצוע של שורה 11 ?

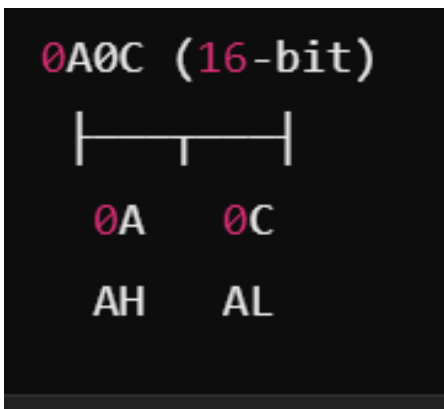
AH = 0A | AL = 0C  
BH = 09 | BL = 0A

ב' (נק') מה יהיה תוכן הרגיסטרים אחרי ביצוע של שורה 17 ?

AH = 01 | AL = 03  
BH = 08 | BL = 06

ג' (נק') מה תהיה פריסת הזיכרון לאחר סיום של התוכנית?

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
41	42	43	44	20	61	62	63	64	20	0C	0A	06	08	02	01
-----															
16	17	18	19	20	21	22	23	24	25	...					
0A	09														
-----															



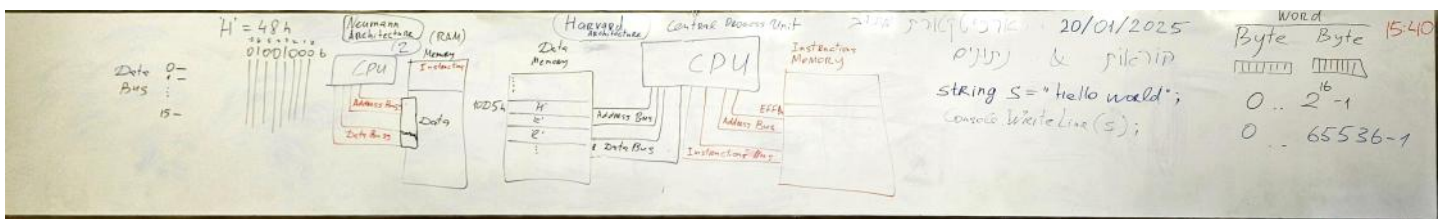
האסטר ש 16 ב' ,

AL ש ה 8 ב' ,

ה'נ' ,

AH ש ה 8 ב' ,

ה'נ' ,



11. (נק') כתבו תוכנית אסמבלר שתדפיס את השם שלכם יחד עם הגיל.

שימו לב לאורך המחרוזת שצריך להדפיס.

לדוגמה:

My name is John Blues

I am 25 years old

```
1  name: DB "ido kahlon" ; store string
2  age: DB "age: 24" ;
3
4  start:
5  MOV AH, 0x13          ; move BIOS interrupt number in AH
6  MOV CX, 10            ; move length of string in cx
7  MOV BX, 0             ; mov 0 to bx, so we can move it to es
8  MOV ES, BX            ; move segment start of string to es, 0
9  MOV BP, OFFSET name   ; move start offset of string in bp
10 MOV DL, 0             ; start writing from col 0
11 int 0x10              ; BIOS interrupt
12
13 MOV CX, 7
14 MOV BP, OFFSET age
15 int 0x10
```

אורך?  
כיוון  
הקצאה