ארכיטקטורת מחשב – תרגיל בית 02

אסמבלי – פקודות בסיסיות ומשתנים

כללי:

נושאי התרגול:

- חישוב כתובת פיזית
- כתיבת תוכניות בסיסיות

לאורך כל הסמסטר העבודה היא אישית - אסור לשבת לעבוד ביחד. עם זאת, מותר ואפילו מומלץ להתייעץ אחד עם השני.
במידה ואתם מתקשים נסו להעזר בגוגל ובמידה ועדיין אתם מתקשים פנו למדריך!

הוראות להגשת התרגיל:

1. עבור כל התכניות הבאות שתכתבו באסמבלי, אנא השתמשו בתבנית הבאה:

```
org 100h
[your code here]
mov ah, 0
int 16h
ret
```

2. כל סעיף מהווה תוכנית נפרדת, ולכן יש לכתוב אותה בקובץ מפרד. יש לשמור את הקבצים עם סיומת asm, כאשר שם הקובץ כולל את מספר השאלה ואת מספר הסעיף. לדוגמא, תוכנית של סעיף הרביעי מתוך שאלה מס' 2, ישמר בקובץ שנקרא 2_4.asm.
3. לאחר שכתבתם את הקוד, הריצו את תכניתכם ועקבו אחר פעולתה באמצעות כפתור emulate. וודאו כי תכניתכם פועלת כראוי ומבצעת את הנדרש.
4. יש להגיש קובץ zip (בלבד!) המכיל את כל התוכניות. יש להקפיד על השם של הקובץ לפי התבנית הבאה:
   02_israel_israeli.zip (כאשר 02 מציין את מס' העבודה, וישראל ישראלי הוא שם פרטי ומשפחה).

המשך בעמוד הבא

שאלה 1

כפי שלמדנו, במעבד 8086 כתובת יחסית וכתובת בסיס הן בנות 16 סיביות. אולם שמירה על עקרון זה גורמת לבעיה, אין מספיק כתובות: כיצד ניתן לתרגם כתובות בנות 16 סיביות לכתובת פיזית שהיא בת 20 סיביות? כדי לפתור בעיה זו בחישוב הכתובת הפיזית, המעבד מבצע שתי פעולות:

תחילה הוא מוסיף לסוף כתובת **הבסיס (segment)** ביטים 0000, או בהקסדצימאלי את הספרה 0h, שנרשמות כסיביות הפחות משמעותיות. תוספת זו שקולה להכפלת כתובת הבסיס ב-16.

לאחר מכן הוא מוסיף לתוצאה שהתקבלה את הכתובת היחסית.

א. חשבו את הכתובת הפיזית עבור כתובת היסט (הכתובת היחסית) 100h וכתובת הבסיס 2000h.

ב. פתחו את תכנית הדוגמא בשם HelloWorld.asm הנמצאת בתיקיית הדוגמאות של האמולטור. הריצו אותה וקפצו ע"י "single step" לשורה הבאה - start: mov dx, msg.

חשבו את הכתובת הפיסית של השורה הנוכחית שאותה אתם מריצים (בסיס –CS, היסט - IP).

שאלה 2 – תכניות בסיסיות

תזכורת: כל סעיף מהווה תוכנית נפרדת.

1. כתבו תכנית אשר מכניסה את הערך 15 (בבסיס 10) לאוגר al.
2. כתבו תכנית אשר מכניסה את הערך 11010011 (בבסיס 2) לאוגר bl.
3. כתבו תכנית אשר מכניסה ערכים לאוגרים כפי שצוין ב-2 השאלות האחרונות. לאחר מכן התכנית תחליף את הערכים בין האוגרים al ו-bl, זאת ללא שימוש בפקודה xchg.
4. בהמשך לסעיף 3, כתבו תכנית דומה אשר משתמשת בפקודה xchg.
5. כתבו תכנית שבקוד שלה נמצאות השורות הבאה:

   mov al, 0f2h

   mov bx, al

   הבינו מה הבעיה וכיצד פותרים אותה.
6. כתבו תכנית אשר מחברת את הערכים 1011 בבינארי ו-4f1 ב-Hex (הציבו את הערכים לתוך אוגרים) ומציבה את התוצאה באוגר DX.

המשך בעמוד הבא

7. קראו בספר על פעולות כפל וחילוק. כתבו תכנית שמבצעת את הפעולות הבאות בזו אחר זו:
   a. עושה השמה של הערך 10 לתוך האוגר bx.
   b. עושה השמה של הערך 4 לתוך האוגר cx.
   c. מכפילה ביניהם (רמז: לא תוכלו לבצע זאת ישירות ותצטרכו להשתמש באוגר עזר).
   d. שומרת את התוצאה באוגר dx.

הריצו את התכנית בסימולטור ועקבו אחר תוכן האוגרים בזמן ריצת התכנית.

8. קראו בספר/באינטרנט על הפקודות shl, shr. השתמשו בהן כדי לכתוב תכנית אשר תחשב את 2 בחזקת 13.

שאלה 3 – קצת לחשוב

בתרגיל זה נראה שיש יותר מדרך אחת לעשות את אותן הפעולות באסמבלי. יש כמה סיבות בגללן כדאי להכיר מספר דרכים להגיע לתוצאה זהה באסמבלי:

- לעתים דווקא הדרכים היותר "מוזרות" הן יעילות יותר מבחינת זמן ריצה/גודל הקוד.
- כאשר קוד אסמבלי (או ליתר דיוק, שפת המכונה) שנוצר ע"י קומפיילר (למשל בתהליך הקומפילציה של שפת C), נראה שונה מאשר אם היינו כותבים אותו בעצמנו. כאשר אנחנו נרצה לעבוד עם קוד שיוצר ע"י קומפיילר (למשל בעת Debugging או Reverse Engineering), כדאי שנכיר צורות שונות ש"אהובות" על קומפיילרים.
- זה משעשע ונחמד (:

1. איפוס משתנה – כתבו שורת קוד אחת עבור כל אחת מהדרכים הבאות לאפס את האוגר ax
   א. שימוש בפקודת mov
   ב. שימוש בפקודת and
   ג. שימוש בפקודת xor
   ד. שימוש בפקודת sub

2. כתבו קוד המבצע את הפעולה neg (פקודת neg מבצעת הכפלת האופרנד ב- **-1**) לאוגר bx אך ללא שימוש בפקודות neg או mul (רמז: היזכרו במה שלמדנו על שיטת המשלים ל-2).

3. כפל וחילוק – כתבו את הפעולות הבאות ללא שימוש בפקודות mul ו-div
   א. $bx \leftarrow bx \times 32$ (בשורת קוד אחת)
   ב. $bx \leftarrow bx \times 41$ (בחמש שורות קוד)

בהצלחה!