Preface LAB1

***יהונתן ארמא 207938903***

***יובל סעיד 206921892***

**חלק E**

1. ההבדל בין משתנים גלובליים ללוקאליים:

בשפת c, משתנים ניתן להגדיר בתוך פונקציות ומחוץ לפונקציות.

משתנים המוגדרים מחוץ לפונקציות נקראים 'משתנים גלובליים' ומשתנים המוגדרים בתוך פונקציות נקראים 'משתנים לוקאליים'.

ה-“scope” של המשתנים הגלובליים הוא כל התוכנית, כלומר כל פונקציה בתוכנית (יותר נכון ב-file כל עוד לא הגדרנו את המשתנה ב-extern) יכולה 'לראות' את המשתנה ולהשתמש בו.

ה-“scope” של המשתנים הלוקאליים הוא הפונקציה או "הסוגריים המסלוסלים" בהם הם הוגדרו, כלומר אם משתנה הוגדר בפונקציה מסוימת אז לא ניתן 'לראות' אותו בפונקציה אחרת, ואם משתנה הוגדר בתוך בלוק מסוים (לדוגמה “if”) אז לא ניתן 'לראות' אותו מחוץ לבלוק.

עוד הבדל בין השניים – משתנים לוקאליים שמורים במחסים (Stack frame) בעוד שמשתנים גלובליים שמורים מחוץ ל-Stack. (BSS או Data)

בתוכנית שהוצגה לנו:

דוגמה למשתנה גלובלי:

maxTrace (שורה 14) – חוץ מזה שסומן לנו בגדול שזה החלק של המשתנים הגלובליים, ניתן להבין שזה משתנה גלובלי מכיוון שהוא לא בתוך אף פונקציה (הוא לא חלק משום “scope”) ולכן ה-“scope” שלו הוא כל התוכנית.

דוגמה למשתנה לוקאלי:

Trace (שורה 57) – משתנה זה הוגדר לנו בתוך פונקציה (ComputeTrace) ולכן ה-“scope” שלו הוא רק בתוך הפונקציה הזאת.

1. A screenshot of a computer program

   Description automatically generatedIAR:

הכתובת היא – 0x26C

אפשר לראות שהמטריצה מכסה מכתובת

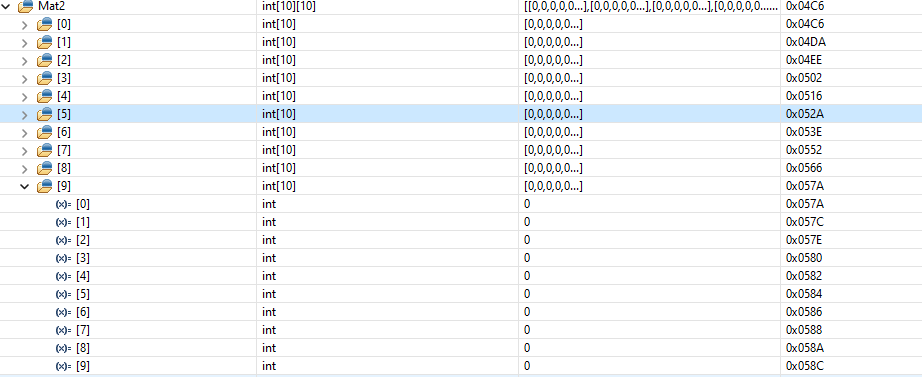
0x26C עד לכתובת 0x332 כולל.

סה"כ C6 כתובות בהקסא שזה 198 . ונוסיף 2 בייטים עבור המילה האחרונה במערך.

סה"כ המטריצה לוקחת 200 בייטים של כתובות. נבין שזה 100 תאים במערך וכל אחד דורש שני בייטים כי במעבד זה int הוא 2 בייטים.

כתובת זו היא של ה-Stack frame כי מטריצה זו מוגדרת בפונקציה של ה-main.

CCS:



גם פה נשים לב כי טווח הכתובות הוא בין 0x4C6 ל-0x58C שזה C6 וכאמור זה 198.

1. IAR:

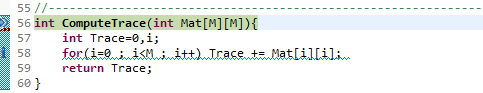


CCS:



1. CCS:

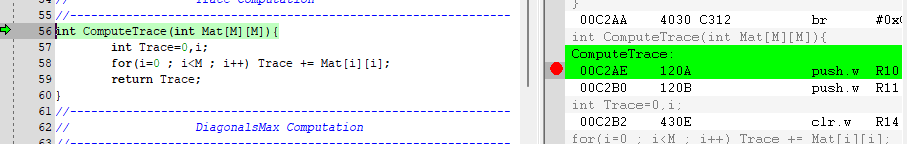
נשים לב ראשית ששמנו ברייק פוינט בנקודה המתאימה :



וערך ה-SP הינו:

0x026A

IAR:

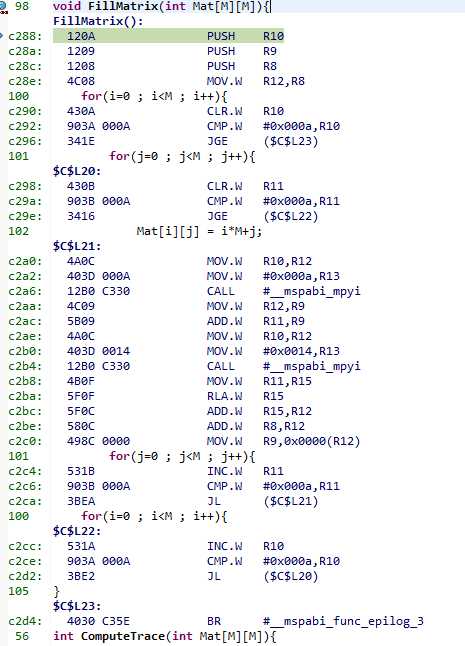
שוב שמנו ברייק פוינט במקום המתאים:

והערך הינו:



1. CCS:

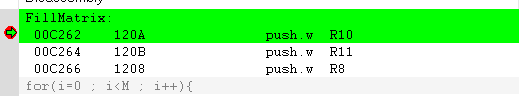
נשים לב ראשית שעשינו Disassembly בשביל לראות את הכתובות:



נבחין שהפונקציה כתובה מהשורה – 0xC288 עד 0xC2D4 וזה 0x4C וזה 76 בייטים.

זיכרון זה הוא זיכרון Flash או מה שנקרא Code segment. האזור בזיכרון בו שמורות הפקודות של התוכנית.

IAR:

תחילת הפונקציה:

סוף הפונקציה:

A screenshot of a phone

Description automatically generated

וטווח הכתובות הוא מ-0xC262 ועד 0xC2AA וסה"כ 0x48 ובדצימלי 72 בייט.

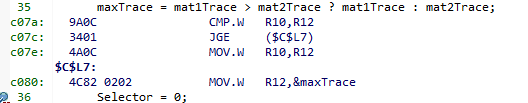
1. ב-IAR ביצענו עצירה של התוכנית בעזרת BreakPoint בפקודה הראשונה של הפונקציה ועוד BreakPoint בפקודה האחרונה של הפונקציה. זמן הריצה הוא ההפרש בין השתיים. כלומר:

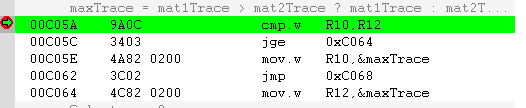
END-START = 3013-77 = 2936

1. נבחין כי משתנה זה מוגדר בפונקציה של ה-main. ולכן ה-scope של המשתנה הוא בפונקציית ה-main בלבד. כלומר הוא משתנה לוקאלי.

מיקומו בזמן ה-scope הוא על המחסנית (כיאה למשתנה לוקאלי).

1. CCS:

נבחין שעצרנו ב-Disassembly וקיבלנו את הקוד באסמבלי:

IAR:

תודה!