דוקומנטציה לטסטים באסמבלי

באופן גס, ניתן לחלק את הטסטים לשני סוגים:

סוג ראשון אשר בודק את אופן ההתמודדות של האסמבלר והסימולטור עם פקודות הקשורות לרגיסטרים שהם לא רגיסטרי חומרה. מדובר בטסטים: summit,bubble ו-qsort. באופן כללי בתוכניות האלה עשוי להיות שימוש ב-16 רגיסטרים בלבד, כלומר ללא רגיסטרי החומרה.

סוג שני אשר בודק את אופן ההתמודדות של האסמבלר והסימולטור עם פקודות הקשורות גם לרגיסטרי החומרה, כלומר הן כוללות גם את פקודות האסמבלי in ו-out. בתוכניות אלה עשוי להיות שימוש ב-16 הרגיסטרים בהם נעשה שימוש בסוג הראשון של הטסטים וגם ב-18 רגיסטרי החומרה.

<u>התוכניות:</u>

- **<u>summat-</u>** בטסט זה מתבצעת סכימת מטריצות בהתאם להוראות המפורטות בתדריך הפרויקט. את התוכנית ממשנו בעזרת לולאה המיוצגת בעזרת for : label עם counter שגדל בכל איטרציה. נעזרנו בעובדה שכתובת האיבר במטריצת הסכום הוא במרחק 32 כתובות מזה של האיבר המתאים במטריצה הראשונה שנסכמת ובמרחק 16 כתובות מהאיבר במטריצה השנייה שנסכמת. כך, בכל איטרציה בלולאה התוכנית נגשת לכתוב לאיבר הבא במטריצת הסכום.
- **bubble-** בטסט זה מתבצע סידור של רשימת מספרים לפי אלגוריתם bubble sort. בתוכנית זו התבססנו על הפסאודו קוד הבא של תוכנית bubble sort (בהתאם לכך גם נרשמו ההערות ליד כל שורה בקוד האסמבלי):

```
VOID BUBBLE(INT A [], INT N){
FOR (I=0; I<N-1; I++)
    FOR(J=0;J<N-I-1;J++)
      IF(A[J]>A[J+1])
        SWAP(A[J], A[J+1])}
```

- **<u>qsort-</u>** בטסט זה מתבצע סידור של רשימת מספרים לפי אלגוריתם Quick sort. בתוכנית זו נעשה שימוש במבנה נתונים של מחסנית כאשר רגיסטר $sp מייצג את כתובת ראש המחסנית. התבססנו על הפסאודו קוד הבא של תוכנית Quick sort (בהתאם לכך גם נרשמו ההערות ליד כל שורה בקוד האסמבלי):

```
QSORT(A,p,r){
  If(p>=r)
    Return
  Else{
    j←Partition(A,p,r)
    QSORT(A,p,j)
    QSORT(A,j+1,r)
   }
}
```

כמו כן מימוש פונקציית Partition , מיוצגת בעזרת הלייבלים swap ו-partition בטסט האסמבלי, התבססה על הפסאודו-קוד הבא:

```
Partition (A,p,r) {

  Pivot=A[p], i=p-1, j=r+1, done=FALSE

  While (done=FALSE){

      repeat j=j-1 until A[j]≤pivot

      repeat i=i-1 until A[i] ≥pivot

      if i<j then swap(A[i], A[j])

    }

    return (j)

}
```

- **leds:** בתוכנית זו הדלקנו וכיבינו את רגיסטר החומרה leds בהתאם להוראות בתדריך. על מנת להדליק כל פעם נורה אחת, כך שמספר הנורה יהיה באינדקס 1 גבוה יותר מהנורה שנדלקה באיטרציה האחרונה, התוכנית מעדכנת בכל פעם רגיסטר בעזרת הוספת 0 מימין ולאחר מכן התוכנית מבצעת out לרגיסטר leds. כמו כן, תוכנית האסמבלי מזהה כי יש כעת להדליק את הנורה הבאה בתור אחרי שניה, דרך מעקב אחרי רגיסטר החומרה irq0status אשר יידלק אחרי שהסימולטור רץ במשך timermax מחזורי שעון, אותו הגדרנו ל-255, על מנת להגיע ל-256 פקודות המתבצעות בין הדלקה להדלקה. כמו כן הוספנו הוראה שלא מבצעת כלום. למעשה זוהי שורת סרק בשביל לשמור על זוגיות סדר הופעת הפקודות כדי לשמור על 256 מחזורי שעון קבועים בגלל לולאת ההמתנה שמכילה 2 שורות פקודה.
  בתוכנית בחרנו להשתמש בשיטת polling לכן אחרי שרגיסטר הסטטוס נדלק כיבינו אותו בשביל להכין אותו לקראת ההדלקה הבאה שתבשר לתוכנית כי יש להליק את הנורה הבאה ללא צורך בשימוש בשגרת פסיקה.
- **clock:** בתוכנית זו סמלצנו תצוגה של שעון במשך מספר שניות. האופן שבו ידענו כי יש לעדכן את השעון לאחר שעברה שניה נעשה באופן זהה כמו בתוכנית leds.
  זיהינו את השעה 20:00:00 כשעה "בעייתית" להכניס לשעון מכיוון שהיא לא שווה לשעה האחרונה שעודכנה ועוד 1 כפי שהיה עד אותו רגע בעבור כל שעה שיוצגה בהקסא.
  לצורך פתרון "בעיה" זו הכנסנו תנאי בתוכנית שתזהה כי השעה הבאה שיש לעדכן היא 20:00:00. בנוסף התוכנית מכניסה מבעוד מועד לזכרון בעזרת פקודת word. את המספר הבא בייצוג הקסאדצימלי `0x1fffff`. כעת כאשר התוכנית תוסיף למספר זה 1 השעה 20:00:00 תוכל להתעדכן בשעון. הערך הנ"ל נשמר מראש בזכרון מכיוון ששדה immediate מחזיק 12 ביטים בלבד.
- **disktest:** בתוכנית זו מועתקים 4 סקטורים הצמודים אחד לשני מהדיסק לסקטורים אחרים בדיסק שגם הם צמודים אחד לשני בהתאם להוראות בפרויקט. בחרנו ליישם את התוכנית כך שתחילה התוכנית קוראת את סקטור n ולאחר מכן תוכן סקטור n מועתק לסקטור מספר n+4, לאחר מכן נקרא את סקטור n+4-3=n+1 ונכתוב את תוכן סקטור זה לסקטור מספר n+1+4=n+5 וכן הלאה. כאשר התוכנית תזהה שהיא צריכה לקרוא את סקטור 4, שלתוכו כבר התוכנית כתבה את סקטור 0, התוכנית תסתיים. גם בטסט זה בחרנו בשיטת polling.
  התוכנית מזהה כי המעבד סיים לקרוא/לכתוב לסקטור אחרי 1024 מחזורי שעון דרך לולאה (המסומנת ב-L2 :label) כאשר רגיסטר חומרה diskstatus מכובה. כלומר פעולת הקריאה/כתיבה נגמרה והדיסק פנוי לביצוע פעולה חדשה.