תוכניות הבדיקה

Summat.asm

מטרת התכנית- לבצע חיבור של שתי מטריצות בגודל 4X4.
נסביר בקצרה על אופן פעולתה:
איברי המטריצה הראשונה נשמרו במקומות 0x100 עד 0x10F, ואיברי המטריצה השניה מ0x110 עד 0x11F. מטריצת התוצאה תיכתב ב0x120 ועד 0x12F.
נשמור בזיכרון את הערכים השמורים ברגיסטרים $ra, $S1, $S0, כדי להימנע מאיבוד מידע. נקבע את $S0 להיות אינדקס לכתובת ההתחלה של המטריצה הראשונה ואת $S1 עבור המטריצה השניה.
נקבע את $t0 להיות counter ללולאה, ונאתחל ל1.
בתוך הלולאה נחלץ בכל פעם שני האיברים בזיכרון המייצגים מקומות זהים בשתי המטריצות ונחבר ביניהם. את תוצאת החיבור נשמור בכתובת המתאימה לאיבר זה במטריצה השלישית (מטריצת התוצאה). נגדיל את $t0 ב1 ונמשיך בפעולת הלולאה כל עוד הוא קטן או שווה ל16 (מספר האיברים במטריצה, שמור ברגיסטר $t1).
בסיום נחזיר את המידע המקורי שהיה שמור ברגיסטרים $ra, $S1, $S0 ונסיים את התכנית עם שימוש בפקודת halt.

bubble.asm

מטרת התכנית- לבצע מיון מערך מספרים בסדר עולה בעזרת שימוש באלגוריתם bubble sort.
נסביר בקצרה על אופן פעולתה:
איברי המערך נשמרו במקומות שהוקצו להם בזיכרון- מ0x100 ל0x10F. נשמור בזיכרון את הערכים השמורים ברגיסטרים $S2, $S1, $S0, כדי להימנע מאיבוד מידע.
נרוץ על איברי המערך בעזרת שתי לולאות ונשווה בין הערכים, וכנעשה באלגוריתם bubble- נבצע החלפה בין שני איברים אם נידרש לכך ונקפוץ לקטע הקוד המתואר על ידי skipped, אשר מטרתו לקדם את אינדקס הלולאה הפנימית ב1.
במידה ואין צורך בביצוע החלפה נקפוץ ישירות לקטע ה- skipped.
לאחר ביצוע מעבר מלא ראשון על איברי המערך נקבל את האיבר הגדול ביותר במקום האחרון בזיכרון המוקצה, ונוכל לרוץ שוב על המערך בלעדיו.
כאשר התנאי עבור הלולאה הראשונה לא יתקיים ניתן להסיק שהמערך שלנו ממוין. בסיום נחזיר את המידע המקורי שהיה שמור ברגיסטרים $S2, $S1, $S0 ונסיים את התכנית עם שימוש בפקודת halt.

Binom.asm

מטרת התכנית- לחשב את מקדם הבינום של ניוטון באופן רקורסיבי, לפי האלגוריתם הנתון בהוראות הפרויקט.
נשמור את n בכתובה 0x100 ואת k בכתובת 0x101.
לאחר שליפת ערכי n וk מהזיכרון והשמתם ברגיסטרים $a1, $a0 בהתאמה, נעבור ישר על פונקציית הבינום הרקורסיבית. נפנה מקום במחסנית ונשמור את הערכים השמורים ברגיסטרים $ra, $S0, $a1, $a0.
במידה ומתקיים k == 0 או n == k, אנחנו נמצאים בתנאי העצירה- ולכן נקפוץ לRETURN: שם נגדיל את ערך $v0 ב1 ונקפוץ לEND.
אחרת- נבצע את הפעולה הרקורסיבית: נקפוץ לBINOM עבור n-1, k-1 ונשמור את ערך ה$v0 המתקבל בסיום הריצה הרקורסיבית לערכים אלו ב$S0. נקפוץ לBINOM עבור n-1, k, ואת ערך ה$v0 שקיבלנו בסיום הריצה הרקורסיבית נחבר ל$S0, ונשמור מחדש ב$v0.

בחלק האחרון של התכנית, END, נחזיר את המידע המקורי שהיה שמור ברגיסטרים $a0, $a1, $S0, $ra, נקפוץ חזרה ל$ra, נשמור את התוצאה הסופית $v0 בזיכרון במיקום הדרוש: 0x120 ונסיים את התכנית עם שימוש בפקודת halt.