פרויקט במבנה המחשב – 313133324 318218625

בפרויקט זה ממומשת סימולציה של מעבד SIMP בשפת C, המעבד סט מסוים של פעולות בשפת אסמבלי, הכולל תמיכה בהתקני קלט ופלט חיצוניים – דיסק קשיח, מוניטור, תצוגת לדים וכו'.

הפרויקט מחולק לשני חלקים – אסמבלר וסימולטור.

אסמבלר

בחלק זה מתבצעת המרה של קוד אסמבלי לקוד מכונה שאותו הסימולטור מריץ ומבצע את הפקודות בהתאם.

פעולת האסמבלר מתבצעת באופן הבא:

- מתקבל קלט בקובץ asm. ובקובץ מפרסרים שורה לאחר שורה, תוך התעלמות מהערות, הזחות ומתוך הנחת קלט של מבנה קבוע של פקודות.
- מפני שישנה תמיכה בתוויות (labels) תחילה התוכנית עובדת על הקובץ פעם ראשונה כדי לזהות באילו שורות ישנן תוויות (גם אם שורות בהן יש פקודת עוקבת אחר התווית: `L1: out $t0 ....`
- ריצה שנייה מתבצעת כאשר מחליפים כל תווית שנקראה בכתובת המתאימה לה (כלומר הpc המתאים), תוך כדי כתיבה לקובץ imemin.txt שורה לאחר שורה בפורמט ידוע מראש של שפת המכונה:

47:40	39:36	35:32	31:28	27:24	23:12	11:0
opcode	rd	rs	rt	rm	immediate1	immediate2

- כתיבת כל הערכים המתאימים לזיכרון dmemout.txt שהתקבלו מפקודת word.

תהליך זה התבצע באמצעות שתי פונקציות עיקריות – אפיון השורה בפונקציה אחת (תווית, פקודה, word.) ובשנייה פרסור כל פקודה לערכים המתאימים.

בנוסף נעזרנו בפונקציות עזר – המרת הפקודה ל opcode המתאים או המרת הרגיסטר למספר המתאים לו, החזרת הכתובת המתאימה לפי שם התווית, הסרת תווי רווח מהמחרוזת וכו'.

מבנה הקבצים: בחלק זה נכתב הקוד בארבעה קבצים –

Main.c - בקובץ הראשי ישנו רק זימון לפונקציה העיקרית לפרסור הפקודות.

Parser.c – בקובץ זה קיימות כל הפעולות הדרושות לפרסור הפקודה.

Parser.h – קובץ בו מוגדרת חתימת הפונקציה העיקרית

Consts.h – קובץ בו קיימים מבני הנתונים והקבועים אשר ישמשו אותנו לפרסור – ערכי רגיסטרים, מבנה נתונים של פקודה ושל תווית וכו'.

סימולטור

מקבלים את קובץ הקלט בשפת מכונה לאחר שעבר את האסמבלר, ערך הזיכרון ההתחלתי ואת קבצים חיצוניים הנוגעים בפסיקות ובהתקני הקלט והפלט, ומבצעים את הפקודות המתאימות.

התהליך מתבצע באופן הבא:

- קוראים את הזיכרון ההתחלתי של התוכנית ומפרשים כל פקודה לפי הפורמט הנתון, ולאחר מכן באמצעות הפעולה הראשית בתוכנית מחלקים למקרים (switch:case) ומבצעים את הפעולה המתאימה על ערכי הקלט הנוספים מהשורה.
- תוך כדי ביצוע הפעולות ישנה תמיכה בפסיקות המתקבלות במספר דרכים – דיסק קשיח, טיימר וקובץ חיצוני, ובמקביל כתיבה לתוך קבצי פלט שונים.
- לאחר קבלת פקודת HALT נעצרת התוכנית וכל ערכי הזיכרון נשמרים בקבצי פלט ומסתיים ביצוע הפעולות.

מבנה הקוד של הסימולטור מורכב מהפעולה העיקרית בה מתבצע פרסור הפקודות וביצוען, ובעיקר פונקציות עזר לכתיבת הפלט המתקבל מהמעבד.

Main.c - בקובץ הראשי ישנו רק זימון לפונקציה העיקרית לפרסור הפקודות.

sim.c – בקובץ זה קיימות כל הפעולות הדרושות לפרסור הפקודה.

sim.h – קובץ בו מוגדרת חתימת הפונקציה העיקרית

Consts.h – קובץ בו קיימים מבני הנתונים והקבועים אשר ישמשו אותנו לפרסור – ערכי רגיסטרים ורגיסטרי חומרה, מאקרו שישמשו אותנו לאינדיקציה פשוטה יותר של הערכים איתם נעבוד בתוכנית, מבנה נתונים של זיכרון ושל דיסק קשיח וכו'.

האתגר העיקרי הוא האופן המקבילי בו מתבצעות הפעולות – פסיקה יכולה להתרחש במקביל לביצוע פעולות אחרות למעבד, שיכולה להתבצע במקביל לקריאה וכתיבה לדיסק הקשיח וכו'.
אופן פעולה זה שונה מהאופן הטורי בו אנו רגילים שתוכנה עובדת בו ולכן נדרש לשנות את דפוס החשיבה ולהתאימו לאתגרים אלו.

תוכניות הבדיקה

עבור וידוא תקינות הסימולטור נתבקשנו לכתוב תוכניות בדיקה מתאימות:

- ציור עיגול על המוניטור: בתוכנית זו אנו מקבלים את רדיוס העיגול מהזיכרון, והמעבד צריך לצייר עיגול ברדיוס מתאים על המוניטור. ביצענו זאת באמצעות ריצה על הבאפר, חישוב מיקום כל פיקסל, והסקה האם הוא בתוך המעגל או מחוצה לו. הערך המתאים ייכתב בבאפר באם הוא בתוך המעגל או מחוצה לו.
  נעזרנו בנוסחה $\left(y_{pix} - y_{center}\right)^2 + \left(x_{pix} - x_{center}\right)^2 < R^2$
- כפל מטריצות בגודל 4X4: מקבלים מהזיכרון ערכי שתי מטריצות בגודל 4X4, וכותבים לזיכרון את מטריצת הפלט המתאימה.
- חישוב מקדם בינומי של ניוטון: זוהי פעולה רקורסיבית בה מחשבים את המקדם הבינומי המתאים לשני ערכים נתונים מהזיכרון, וכתיבה במיקום אחר את תוצאת החישוב.
- הזזת סקטורים בדיסק הקשיח: בפעולה זו הועתקו הסקטורים 0 עד 7 בדיסק הקשיח לסקטורים 1 עד 8.
  מפני שזמן העבודה של כל פעולת דיסק קשיח הוא 1024 מחזורי שעון, נעזרנו בפסיקות כך שנצא לפסיקה בכל פעם שבה הדיסק הקשיח מסיים קריאה או כתיבה ונדע שהסתיימה פעולה זו.
  אופן ההעתקה היה מהסוף להתחלה – תחילה מעתיקים את סקטור 7 לסקטור 8, ואז את 6 ל 7 וחוזר חלילה.