הטכניון – מכון טכנולוגי לישראל
הפקולטה למדעי המחשב

מבנה מחשבים ספרתיים - תרגיל בית 1 : Pipeline

מתרגל שאחראי על התרגיל : אנדריי.

הגשה אלקטרונית בלבד. הגשה בזוגות.

עליכם לבנות Emulator עבור המכונה LC-3 , על כל אחד משלבי העיבוד שלה. כקלט הסימולציה תקבל קוד אסמבלי לביצוע וכן מס' מחזורי שעון של הסימולציה שיש לבצע. על ה-Emulator להיות Cycle-Accurate, כלומר - על הסימולציה לשקף את מצב המכונה במדויק בכל מחזור שעון.

ארכיטקטורת המכונה :

ה-LC3 הוא מעבד pipeline בן 4 שלבים. במעבד אין חיזוי קפיצות או forwarding במקרה של data hazards. כמו כן נתון כי :

- אורך כל פקודה 2 בתים.
- גודל הזיכרון 512 בתים.
- במכונה 8 רגיסטרים בגודל 2 בתים.
- סט הפקודות מצורף בקובץ LC3.pdf.
- במכונה 3 דגלים – N – שלילי, Z – אפס, P – חיובי.
- כל ה-Offsets הינם מהכתובת שלאחר הפקודה הרלוונטית.
- הייצוג של המספרים הינו כמו הייצוג של המכונה המבצעת את האמולציה (כלומר אין צורך לבצע המרות Little-Endian→Big-Endian וכו').
- במכונה 4 שלבים :
  - Fetch
  - Decode
  - Execute
  - Memory + WB (כתיבה לזיכרון ולרגיסטרים).
- המעבד משתמש ב-stall בלבד כאשר מתגלה hazard. בפרט, אין חציה של קובץ הרגיסטרים.
- פרט לפעולת LD, הדגלים NZP זמינים מיד לאחר שלב ה-execute כך שאין צורך ב-stall בגללם. בפעולת LD הדגלים זמינים לאחר שלב ה-Memory , אך מצב שבו מתרחש stall בעקבות הדגלים לא ייבדק.
- הרגיסטרים ו-PC מתעדכנים רק לאחר שלב ה-wb/memory.
- הכרעת הקפיצה (branch resolution) מתרחש בשלב ה-wb/mem.
- הזיכרון מתעדכן אך ורק בשלב ה-memory.
- זיהוי ה-Control Hazard מתבצע בשלב ה-fetch.
- זיהוי ה-Data Hazard מתבצע בשלב ה-decode.

הנחיות:

- מצורף מסמך PDF המתאר את סט הפקודות של ארכיטקטורת LC3
- לצורך פישוט התרגיל, האמולציה תתמוך אך ורק בפקודות הבאות:
  - ADD – חיבור רגיסטר עם רגיסטר נוסף או קבוע והשמת התוצאה ברגיסטר התוצאה.
  - AND – ביצוע AND לוגי בין רגיסטר לרגיסטר נוסף או קבוע והשמת התוצאה ברגיסטר התוצאה.
  - LD – טעינת רגיסטר מהזיכרון.
  - ST – כתיבת רגיסטר לזיכרון.
  - BR – ביצוע קפיצה. בקפיצה שאינה מותנית <u>הקפיצה מתבצעת ללא תלות באיזה דגל NZP דלוק במערכת</u>. בקידוד קפיצה שאינה מותנית כל הדגלים דלוקים. אם רק דגל אחד דולק בקידוד פקודת ה-br מדובר בקפיצה מותנית ותתבצע קפיצה רק אם הדגל התואם לאחד הדגלים בשדה NZP של הפקודה דולק.
- פקודות LD, ADD ו-AND ישפיעו על ערכי הדגלים, בעוד יתר הפקודות לא ישפיעו על ערכי הדגלים.
- ניתן לכם מהדר לאסמבלר של LC-3 ומחלקה בשם LC3. עליכם לממש את המתודות הבאות, המייצגות את השלבים של עיבוד הפקודה:
  - Fetch
  - Decode
  - Execute
  - MemWB
- כל הסיגנלים וערכי ה-Latches השונים של פקודה צריכים להימצא בתוך ה-struct הקרוי Signals. בהתאם לנלמד, יש להימנע מלשבור את השלבים השונים. לדוגמא, אין לגשת לערך רגיסטר של המכונה ישירות בשלב ה-Execute, אלא לקרוא אותו בשלב ה-Decode ולהציב את ערכו בתוך Signals.
- פירוש ערכי השדות הארכיטקטוניים – SRx הינו רגיסטר מקור, DR – הינו רגיסטר יעד. שימו לב כי כתובות הקפיצה של BR הינן יחסיות.
- יש לשמור על לוגיקה המפרידה בין שלבי ה-pipeline.
- לימדו את דרך הפעולה של LC3 לפי המימוש של lc3.cpp ו-lc3.h. מומלץ להיעזר במתודות שהוגדרו.

כתיבת הקוד

עליכם לממש את המתודה RUN אשר מקבלת את כקלט כמה מחזורי שעון תתבצע הסימולציה. כמו כן עליכם לממש מתודות עבור שלבי הביצוע השונים ב-pipeline .
חשוב: את הפתרון שלכם כתבו בקבצים lc3-hw.cpp ו-lc3-hw.h. כל הפתרון צריך להיכלל בקבצים אלו. אתם רשאים במידת הצורך לשנות את הקובץ lc3.h כדי לשנות את הגדרות המחלקה LC3 . הינכם רשאים לשנות את החתימה של הפונקציות של שלבי הביצוע, אך את החתימה של המתודה run אסור לכם לשנות . אין לשנות חלקים אחרים בתוכנית.

מצורף קובץ Makefile וקובץ הדגמה multiply.asm המכפיל שני מספרים. בסיום התוכנית מודפס תוכן הזיכרון. הקובץ res.txt מכיל את הפלט של האמולטור בסוף ריצת התוכנית (הריצו את האמולציה כ-100 מ״ש , זה יהיה יותר ממספיק).
ההרצה מתבצעת על-ידי הרצת
lc3 [program name] [number of cycles].

טיפים:

- עליכם להגדיר משתנים שיחזיקו את ערכי החוצצים בין השלבים השונים של ה-Pipeline
- אחד הקשיים שתיתקלו בהם הינו ההבדל בין מעבד, המבצע את כל השלבים במקביל לאמולטור המריץ את השלבים סדרתית. אי-לכך, שינויים שהאמולטור מבצע בחוצצים או במצב המכונה בזמן אמולציה של שלב אחד יכולים להשפיע על שלב אחר. שני פתרונות אפשריים הינם:
  - האמולטור יחזיק עותק של מצב המכונה ועותק של החוצצים בתחילת כל מחזור השעון שישמשו כקלט. שינויים שהאמולטור יבצע ייכתבו לחוצצים ולמצב המכונה המקוריים.
  - האמולטור יבצע את שלבי ה-Pipeline מהאחרון לראשון, כך ששינויים בחוצצים לא ישפיעו על שלבים מאוחרים יותר המשתמשים בנתוני החוצצים כקלט. שימו לב שפתרון כזה עדיין מצריך יצירת עותק של מצב המכונה במידה ושלבים מאוחרים ב-Pipeline מעדכנים חלקים במצב המכונה ששלבים מוקדמים יותר משתמשים בהם.
- שימו לב מהו ה-opcode השמור ב-lc3 והשתמשו בו לצורך stall במידת הצורך.
- אין צורך להסתכל בקבצים compiler.cpp וכן compiler.h .

הוראות הגשה:

- יש להגיש אלקטרונית את כל קבצי המקור. אין לשנות חלקים אחרים בתוכנית פרט לאלו שצוינו. הסיבה שאנו מאשרים הגשת כל הקבצים היא למנוע מקרה שבו יורדו כל הנקודות בעקבות שינוי שביצעתם בטעות באחד הקבצים האחרים .
- יש להגיש את הפתרון בתוך zip ואין לכנן ספריות בתוך קובץ זה.
- וודאו שהתרגיל שלכם מתקמפל עם קובץ ה-makefile בסביבת לינוקס(t2).
- שם ה- zip צריך להיות <id1>_<id2>.zip כאשר <id1>,<id2> מספרי הסטודנט של המגישים.