|  |  |  |
| --- | --- | --- |
| **BAR-ILAN UNIVERSITY**  School of Engineering |  | אוניברסיטת בר-אילן בית הספר להנדסה |

מבוא להנדסת מחשבים

# תשע"ד סמס' א' מועד א'

**83-252**

* מרצה : פרופ' שמואל וימר
* מתרגלת : אילת חיימוביץ
* **חומר עזר מותר:** מחברות/ דפים מהרצאות, תרגולים ותרגילי בית, ספר computer organization and design, מילון עברי צרפתי.
* **משך המבחן שלוש שעות**
* **סך כל הנקודות הוא 111**.
* משקל השאלות השונות נתון בגוף השאלות.
* **יש לנמק את כל תשובותיכם**. אין צורך לפתח מחדש תוצאות שהוכחו בכיתה, אלא אם כן נאמר מפורשות לעשות כן.
* **יש לשרטט דיאגראמות באופן ברור !**

**בהצלחה**

**שאלה מס' 1 (20 נק'):**

הטבלה שלהלן מתארת את זמני הבצוע של שלש תכניות א', ב' ו ג', המבוצעות על שלשה מחשבים שוניםA, B ו C, ואת מספר הפעולות האריתמטיות בכל תכנית. מניחים שלכל פעולה אריתמטית זמן בצוע שווה.

1. בהנחת עומס ((WORKLOAD שבו שלשת התכניות רצות אותו מספר פעמים, איזה מחשב מהיר ביותר, ובאיזה יחס לשני האחרים?

זמן הביצוע הכולל עבור המחשבים הינו כדלהלן:

מחשב A: 2+20+200=222 seconds

מחשב B: 5+20+50=75 seconds

מחשב C: 10+20+15=45 seconds

על כן בהנחת עומס שבו שלושת התוכניות רצות אותו מספר פעמים מחשב C הוא המהיר ביותר. הוא מהיר פי 222/45=75/15 ממחשב A, ומהיר פי 75/45=5/3 פעמים ממחשב B.

1. חזור על סעיף א עבור עומס בו תכנית א' רצה פי 8 פעמים מתכנית ג' ותכנית ב' רצה פי 2 פעמים מתכנית ג'.

מחשב A: 8\*2+2\*20+1\*200=256 seconds

מחשב B: 8\*5+2\*20+1\*50=130 seconds

מחשב C: 8\*10+2\*20+1\*15=135 seconds

על כן עבור העומס הנ"ל מחשב B הוא המהיר ביותר, הוא מהיר פי 135/130=1.04 ממחשב C, ומהיר פי 256/130=1.97 ממחשב A.

1. עבור עומס אחר מתקיים ששלשת התכניות רצות זמן שווה על מחשב A, חשב את זמני הבצוע של העומס על כל אחד מהמחשבים. איזה מחשב הכי מהיר? (רמז: מצא את השכיחות היחסית של כל תכנית בעומס).

מהשוואת זמני ריצה של 3 התוכניות על מחשב A נמצא כי כדי להשיג זמני ריצה שווים נצטרך להריץ את תוכנית A 100 פעמים, את תוכנית B 10 פעמים ואת תוכנית C פעם אחת. מכאן נשיג את זמני הביצוע הבאים:

מחשב A: 100\*2+10\*20+1\*200=600 seconds

מחשב B: 100\*5+10\*20+1\*50=750 seconds

מחשב C: 100\*10+10\*20+1\*15=1215 seconds

כלומר עבור העומס הנ"ל מחשב A הוא המהיר ביותר.

1. כנ"ל בהנחה ששלשת התכניות רצות זמן שווה על מחשב B.

נשתמש בזמני הריצה של מחשב B כדי לקבוע משקל, ונמצא כי כדי להשיג זמני ריצה שווים נצטרך להריץ את תוכנית A 20 פעמים, את תוכנית B 5 פעמים ואת תוכנית C פעמיים. מכאן נשיג את זמני הביצוע הבאים:

מחשב A: 20\*2+5\*20+2\*200=540 seconds

מחשב B: 20\*5+5\*20+2\*50=300 seconds

מחשב C: 20\*10+5\*20+2\*15=330 seconds

כלומר עבור העומס הנ"ל מחשב B הוא המהיר ביותר.

1. כנ"ל בהנחה ששלשת התכניות רצות זמן שווה על מחשב C.

נשתמש בזמני הריצה של מחשב C כדי לקבוע משקל, ונמצא כי כדי להשיג זמני ריצה שווים נצטרך להריץ את תוכנית A 6 פעמים, את תוכנית B 3 פעמים ואת תוכנית C 4 פעמים. מכאן נשיג את זמני הביצוע הבאים:

מחשב A: 6\*2+3\*20+4\*200=872 seconds

מחשב B: 6\*5+3\*20+4\*50=290 seconds

מחשב C: 6\*10+3\*20+4\*15=180 seconds

כלומר עבור העומס הנ"ל מחשב C הוא המהיר ביותר.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| תוכנית | פעולות אריתמטיות | זמן ריצה בשניות | | |
| מחשב A | מחשב B | מחשב C |
| א' | 5x109 | 2 | 5 | 10 |
| ב' | 20x109 | 20 | 20 | 20 |
| ג' | 40x109 | 200 | 50 | 15 |

**שאלה מס' 2 (35 נק'):**

במכונת MIPS בעלת מחזור שעון אחד רוצים לממש בפקודה אחת הנקראת **l\_inc** שתפקידה לטעון מילה מהזכרון ולהגדיל ב 1 את אוגר האינדקס (הבסיס). **l\_inc** מממשת בפועל את שתי הפקודות שלהלן:

lw $rs, L($rt)

addi $rt, $rt, 4

נתונה סכמת ה MIPS המצורפת.

1. מה עשוי להיות תפקידה ויתרונה של פקודה כזאת?

לביצוע פעולות כלשהן על בלוק שלם בזיכרון.

1. מה היתרון של הפקודה לעומת הקוד הנ"ל?

ב- single cycle MIPS ביצוע כל פקודה אורכת מחזור שעון אחד. על כן הפקודה הנ"ל תארך מחזור שעון אחד בניגוד לקוד שידרוש 2 מחזורי שעון.

1. האם נדרשים שינויים ב REGISTER FILE? אם כן הסבר אילו שינויים ומדוע, וציירם בסכמה.

כן. בפקודה זו נדרשות כתיבה ל-2 רגיסטרים ב- register file באותו מחזור שעון. על כן נצטרך להוסיף עוד פורט כתיבה ל- register file "Write Register 2", פורט זה ישלט על ידי כניסת בקרה חדשה "Write 2", עוד כניסת נתונים "Write data 2". כאשר Write2=1 המידע ב- Write data 2 יכתב לרגיסטר שכתובתו נכנסת ל- Write Register 2.

הערה: ישנה גם אפשרות לא להוסיף כניסת Write Register 2, היות וכתובת 2 הרגיסטרים אליהם נכתוב חייבים להיכנס כקלט ל- Read Register 1 או Read Register 2.

1. הוסף אותות בקרה ומעבר נתונים נדרשים וציירם בסכמה.

למען הנוחות, נניח שלפורט הכניסה הישן נכנס $rs (זו אינה התשובה היחידה האפשרית, אך זו התשובה הדורשת מספר קטן ביותר של שינויים ל- data path). על כן $rt נכנס ל- Read Register 1. נדאג שהקלט ל- Read Register 1 יכנס גם ל- Write Register 2.

כאשר Write 2=1 תהיה כתיבה לרגיסטר $rt של המידע שנכנס ל- Write Data 2. על כן נוסיף מחבר חדש שיקבל כקלט את הקבוע 4 ואת מוצא Read Data 1 של ה- Register file. מוצא המחבר יהיה כניסת Read Data 2.

1. עדכן את הטבלת אותות הבקרה המצורפת עבור הפקודה החדשה.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Write2 | ALUOp0 | ALUOp1 | branch | MemWrite | MemRead | RegWrite | MemtoReg | ALUSrc | RegDst | Instruction |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | R-format |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | lw |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | X | 1 | X | sw |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | X | 0 | X | beq |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | l\_inc |

1. במידה ובסעיף 3 נדרשים שינויים, האם השינויים הללו נחוצים גם ב MIPS רב מחזורי? נמק במדויק.

לא. ב- multicycle MIPS ניתן לכתוב לאותו פורט כתיבה ב- Register file מספר פעמים במחזורי שעון שונים, על כן אין צורך לשנות את מבנה ה- register file.

**שאלה מס' 3 (20 נק'):**

השווה את ביצועיהם של מכונות MIPS בנות מחזור אחד, רב מחזוריות ומצונרות (PIPELINED). לבצוע מצונר הנח שבמחצית מהמקרים הפקודה המיידית אחרי **lw** משתמשת בתוצאת הטעינה, שההשהיה כתוצאה מניחוש לא נכון של קפיצה מותנית הינה מחזור אחד, ושרבע מניחושי הקפיצה שגויים. הנח שקפיצה (בלתי מותנית) תמיד דורשת השהייה של מחזור אחד. התעלם מכל HAZARD אחר.

* הנח זמן גישה לזכרון 200 פיקו שניות, **ALU**100 פיקו שניות, וכתיבה וקריאה מרגיסטר 50 פיקו שניות.
* העומס הינו 25% פקודות **lw**, 10% **sw**, 11% קפיצות מותנות, 2% קפיצות לא מותנות ו52% פקודות **ALU**.

1. מה מחזור השעון למכונה בת מחזור שעון אחד ומה זמן בצוע פקודה ממוצע?

נחשב את אורך מחזור השעון הדרוש עבור כל סוג פקודה:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Instruction Memory | Register Read | ALU Operation | Data Memory | Register Write | Total |
| R-type | 200ps | 50ps | 100ps |  | 50ps | 400ps |
| Lw | 200ps | 50ps | 100ps | 200ps | 50ps | 600ps |
| sw | 200ps | 50ps | 100ps | 200ps |  | 550ps |
| Branch | 200ps | 50ps | 100ps |  |  | 350ps |
| jump | 200ps |  |  |  |  | 200ps |

מחזור השעון עבור מכונה עם מחזור שעון יחיד לכל פקודה יקבע על ידי הפקודה הארוכה ביותר, כלומר 600ps. לכן מחזור שעון הוא 600ps. מכיוון שמדובר ב- single MIPS מספר מחזורי שעון לפקודה הוא 1, ועל כן זמן ממוצע לביצוע פקודה הוא 600ps.

1. מה מספר מחזורי שעון ממוצע לפקודה בממוש רב מחזורי? מה זמן בצוע פקודה ממוצע?

ידוע כי ב- multi cycle MIPS, פקודת lw אורכת 5 מחזורי שעון, פקודת sw ו- r-type אורכות 4 מחזורי שעון ופקודת branch ו- jump אורכות 2 מחזורי שעון. על כן:

אורך מחזור שעון הוא 200ps, ועל כן זמן בצוע פקודה ממוצע הוא 4.12\*200ps=824ps.

1. מה מספר מחזורי השעון הממוצע לפקודה במכונה מצונרת?מה זמן בצוע פקודה ממוצע?

על פי הנתון, בחצי מפקודות lw ישנו load hazard הגורר שיהוי של מחזור שעון אחד. רבע מפקודות ה- branch גוררות שיהוי של מחזור שעון אחד, וכל פקודות jump גוררות שיהוי של מחזור 1. על כן, השיהוי הממוצע הנגרם כתוצאה מ- hazards הינו:

0.25\*0.5+0.11\*0.25+0.02=0.1725.

קיבלנו 2 פתרונות לשאלה זו. פתרון ראשון: מספר מחזורי השעון הממוצע לפקודה במכונה מצונרת הוא 1.1725.

פתרון שני: מספר מחזורי השעון הממוצע לפקודה במכונה מצונרת הוא 5.1725.

אורך מחזור שעון הוא 200ps, על כן עבור זמן בצוע פקודה ממוצע קיבלנו את התשובה 1.1725\*200ps או 5.1725\*200ps.

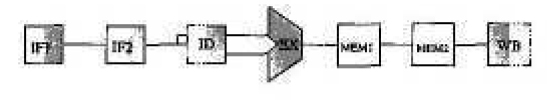
**שאלה מס' 4 (36 נק'):**

על מנת להאיץ את ביצועי המכונה המצונרת, מציעים להקדיש שני מחזורי שעון לגישה לזכרון.

1. מה תדר השעון החדש?

כעת זמן מחזור הוא 100ps ועל כן תדר השעון הוא 1/100ps.

1. שרטט באופן סכמתי את ה PIPE החדש (רק את השלבים השונים, ללא אותות בקרה).



1. רשום את כל מצבי ה HAZARDS וה FORWARD ואת אורכם.

לפקודה מסוג r-type יש סכנה ש-3 הפקודות שאחריה יקראו מה- register file ערך שגוי מכיוון שהפקודה עוד לא הגיע לשלב ה- write back. הטיפול בכך יעשה באמצעות forwarding ולא יהיה השהיה בכלל.

לפקודה מסוג lw יש סכנה ש-3 הפקודות שאחריה יקראו מה- register file ערך שגוי מכיוון שהפקודה עוד לא הגיע לשלב ה- write back. אם הפקודה שאחרי lw ניגשת לרגיסטר ל- lw אמור לכתוב אליו יש לעשות stall של 2 מחזורים ולאחר מכן forwarding. אם הפקודה שנמצאת 2 פקודות לאחר ה- lw ניגשת לרגיסטר ל- lw אמור לכתוב אליו יש לעשות stall של מחזור אחד ולאחר מכן forwarding. אם הפקודה שנמצאת 3 פקודות לאחר ה- lw ניגשת לרגיסטר ל- lw אמור לכתוב אליו יש לעשות forwarding, ואין צורך ב- stall.

ישנו גם branch hazard. אין מקרים חדשים של hazard זה והטיפול לא ישתנה.

