

תוכן העניינים:

| | |
|----|------------------------------|
| 2 | מבנה המחשב ותכן לוגי |
| 2 | המעבד החד-מחזורי |
| 2 | מבנה מעבד חד-מחזורי: |
| 2 | סיכום כללי: |
| 7 | שאלות: |
| 21 | תשובות סופיות: |
| 23 | הערכת ביצועי מעבד חד-מחזורי: |
| 23 | סיכום כללי: |
| 24 | שאלות: |
| 26 | תשובות סופיות: |

מבנה המחשב ותכן לוגי

המעבד החד-מחזורי

מבנה מעבד חד-מחזורי:

סיכום כללי:

המעבד החד-מחזורי:

לאחר הכרנו את הבלוקים המרכזיים שלנו, ולמדנו על פורמט הפקודות של מחשבי ה-MIPS32 אנחנו מוכנים לחבר את הכל לכדי יחידת עיבוד המסוגלת לבצע מגוון רחב של פקודות.

נתמקד במודל מופשט שכל מטרתו היא לימודית והוא אינו מיושם באופן מסחרי

נתיב הנתונים ויחידת הבקרה:

במימוש של יחידת עיבוד מרכזית (CPU – Central Processing Unit או בעברית: יע"מ) יש להתמקד בשני היבטים:

- **נתיב הנתונים** – אופן החיבור בין הבלוקים השונים לכדי יצירת המעבד. נתיב הנתונים קובע אלו כניסות ואלו יציאות יחוברו לכל אחד ואחד מרכיבי המעבד. קווים אלו הם קווי המידע (Data lines).
- **יחידת הבקרה** – יחידה צירופית המקבלת רצף כניסה בהתאם ל-ISA של פקודה מסוימת ושולטת בסיביות האֵיפְשור (ה- enable, set, reset, preset) של כל רכיבי המעבד. כל הקווים שיוצאים מיחידת הבקרה לכל אחד מרכיבי המעבד נקראים קווי בקרה (Control lines).

עקרונות בסיסיים במימוש נתיב הנתונים של מעבד חד-מחזורי:

- מחזור שעון אחד לכל פקודה (כלומר, במעבד זה: $CPI=1$).
- זמן ההתייצבות של כל אחד מהרכיבים הצירופיים מתכנס במחזור שעון אחד. (בהתאם לכך מחזור השעון נקבע לפי זמן ההתייצבות הארוך ביותר!)
- כל כתיבה המתבצעת לאוגרים (ה- Registers File) תתבצע פעם אחת והיא תהיה בסוף מחזור השעון (edged triggered).

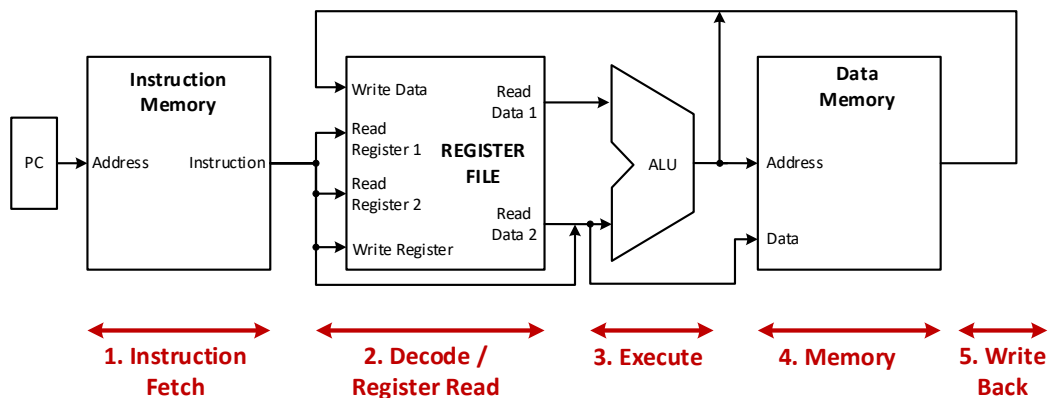
מימוש מעבד חד-מחזורי:

נתמקד במימוש של מעבד חד-מחזורי המסוגל לבצע רק חלק מפקודות ה-MIPS והן:

- פעולות מסוג R-Type שהן: add, sub, and, or, slt.
- גישה לזיכרון עבור קריאה וכתיבה: lw, sw.
- פקודות קפיצה לניתוב הבקרה: beq, jump.

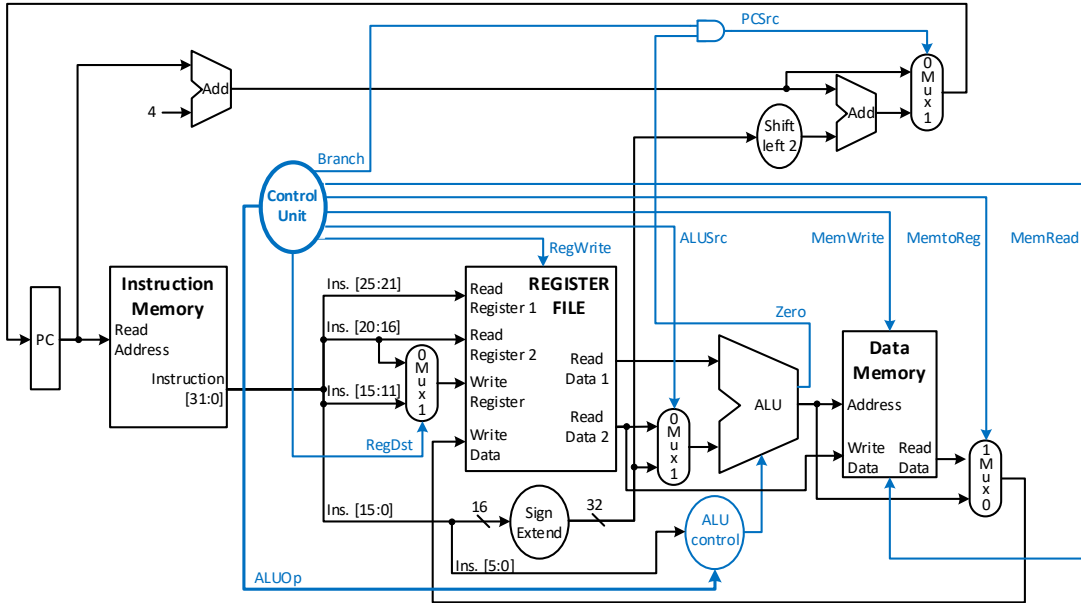
שלבי ביצוע ההוראה ע"י המעבד:

- Fetch – הבאת הפקודה מזיכרון הפקודות באמצעות הערך אליו ה-PC מצביע.
- Decode – פיענוח הפקודה וקריאת המידע הנדרש לביצועה.
- Execute – ביצוע הפקודה בעזרת ה-ALU או רכיבים לוגים אחרים.
- Memory – שימוש בתוצאה או בערך הפקודה עבור גישה לזיכרון (טעינה או כתיבה).
- Write Back – בחלק מן הפקודות יש צורך לכתוב ערך מהזיכרון חזרה למקבץ האוגרים.



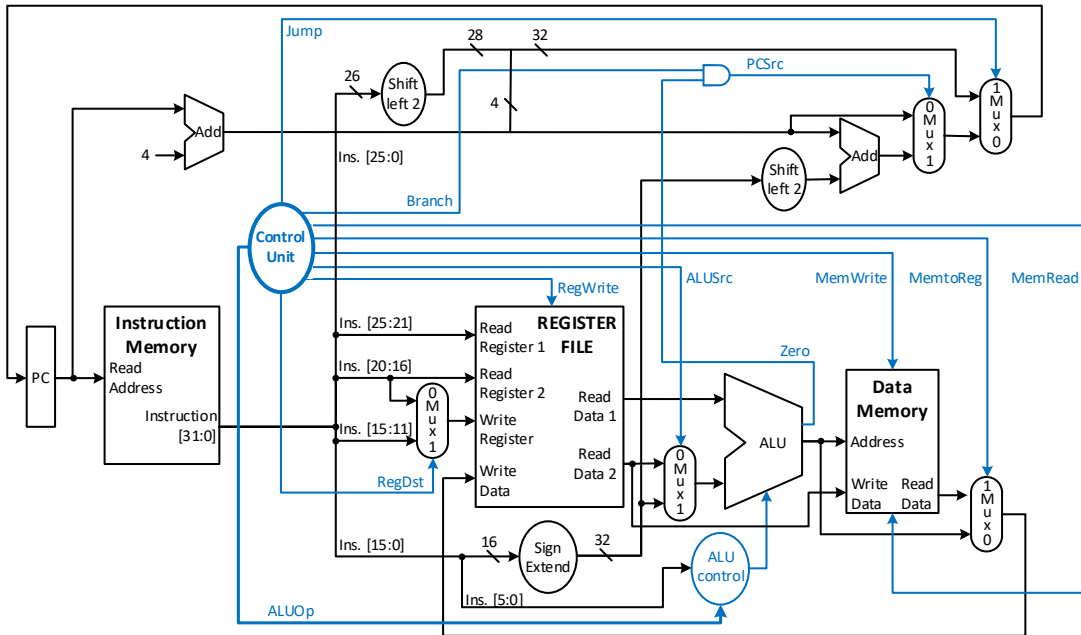
מבנה כללי של מעבד חד-מחזורי - ללא פקודת jump:

להלן המבנה כללי של המעבד כפי שפותח בסרטוני הוידאו:



מבנה כללי של מעבד חד-מחזורי - כולל פקודת jump:

להלן המבנה כללי של המעבד כפי שפותח בסרטוני הוידאו:



פורמט קווי הבקרה של מעבד חד-מחזורי:

| Ins. | Opcode | RegDst | ALUSrc | MemtoReg | RegWrite | MemRead | MemWrite | Branch | Jump | ALUOp |
|----------|--------|--------|--------|----------|----------|---------|----------|--------|------|-------|
| R-format | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 10 |
| lw | 35 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 00 |
| sw | 43 | X | 1 | X | 0 | 0 | 1 | 0 | 0 | 00 |
| beq | 4 | X | 0 | X | 0 | 0 | 0 | 1 | 0 | 01 |
| jump | 2 | X | X | X | 0 | 0 | 0 | X | 1 | XX |

מפרט קווי הבקרה של מעבד חד-מחזורי:

| Signal Name | Line Width (bits) | Reset (0) | Set (1) |
|---------------------------|-------------------|---|--|
| RegWrite | 1 | No write is allowed | The content in <i>Write Register</i> (\$rd) is replaced with the new value (from ALU or Data Memory according to command). |
| MemWrite | 1 | No write is allowed | The cell in Data Memory at <i>Address</i> opens for writing (replacing the current content with the new one). |
| MemRead | 1 | No read is allowed | The cell in Data Memory at <i>Address</i> is output through <i>Read Data</i> data-line. |
| RegDst | 1 | The register destination is <i>rt</i> (Ins.[20:16]). | The register destination is <i>rd</i> (Ins.[15:11]). |
| ALUSrc | 1 | Second ALU operand is the content of <i>rt</i> (\$rt). | Second ALU operand is the 32-bits signed extended <i>Address</i> value. |
| MemtoReg | 1 | Content to write back to Register File is from ALU. | Content to write back to Register File is from Data Memory. |
| PCSrc | 1 | Next instruction is at: PC + 4. | Next Instruction is at: PC + 4 + 4*Branch target address. |
| Jump | 1 | Next Instruction is determined by PCSrc. | Next Instruction is at: PC + 4 + 4 * jump target Address. |
| ALUOp (Secondary Control) | 2 | For: ALUOp[1:0] = 00: ALU performs addition. For: ALUOp[1:0] = 01: ALU performs subtraction. For: ALUOp[1:0] = 10: ALU performs the <i>funct</i> operation. | |

קווי הבקרה משנית - ALU control lines:

| Instruction Opcode | ALUOp | Instruction Operation | Funct Field | Desired ALU action | ALU control input |
|--------------------|-------|-----------------------|-------------|--------------------|-------------------|
| lw | 00 | Load word | XXXXXX | Add | 0010 |
| sw | 00 | Store word | XXXXXX | Add | 0010 |
| Branch equal | 01 | Branch equal | XXXXXX | Subtract | 0110 |
| R-Type | 10 | Add | 100000 | Add | 0010 |
| R-Type | 10 | Subtract | 100010 | Subtract | 0110 |
| R-Type | 10 | AND | 100100 | AND | 0000 |
| R-Type | 10 | OR | 100101 | OR | 0001 |
| R-Type | 10 | Set on less than | 101010 | Set on less than | 0111 |

יתרונות וחסרונות של מעבד חד-מחזורי:

יתרונות:

- מודל פשוט וקל להבנה.

חסרונות:

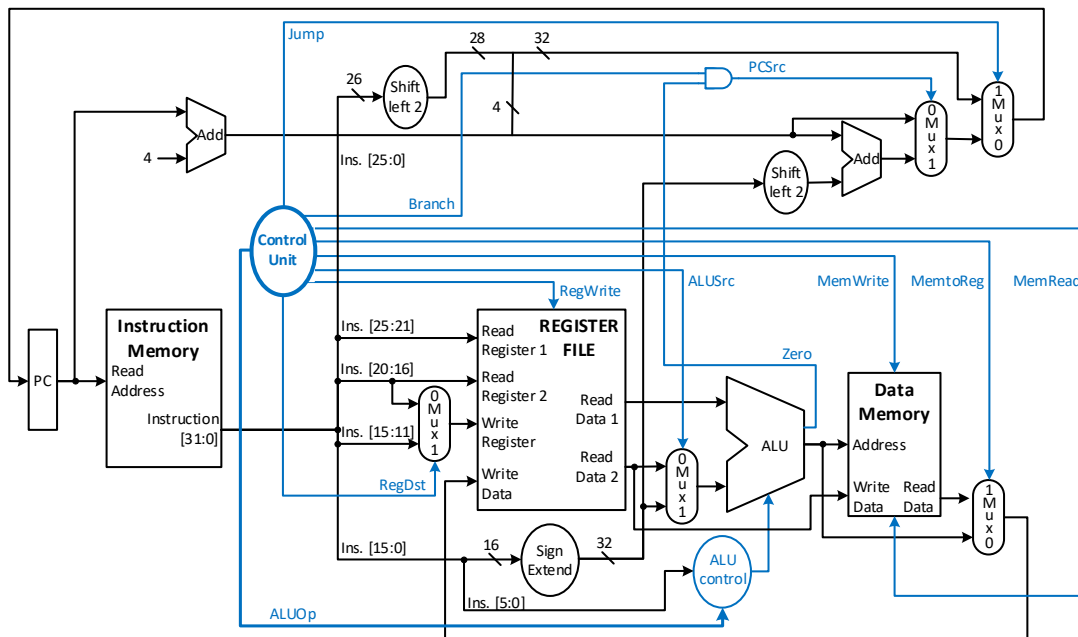
- זמן ביצוע פקודה הוא איטי מכיוון שיש להתאים את מחזור השעון לפקודה הארוכה ביותר. עקב כך ישנן פקודות מהירות יותר אשר יסיימו את מלאכתן בטרם מחזור השעון יתחלף – דבר אשר יפגע ביעילות הפעילות של המעבד.
- ניתן להשתמש בכל אחד מן הרכיבים החומרתיים פעם אחת במשך מחזור שעון. לכן פקודות הדורשות שימוש חוזר בבלוקים מסוימים (כגון פסאודו פקודות) יצטרכו להתבצע במשך מספר מחזורי שעון – דבר שיוביל לזמן ביצוע ארוך יותר.
- שימוש בחומרה עודפת – במודל שלנו יש שימוש ברכיבי add כאשר ה-ALU כבר יודע לבצע את פעולה זו.
- חלוקת הזיכרון ל-Instruction Memory ול-Data Memory כאשר בפועל מדובר באותו מערך זיכרון (רכיב אחד). נרצה אפשרות לגשת לזיכרון במחזור אחד גם לקריאה וגם לכתובה.

שאלות:

שאלות יסודיות בסימון ערכי קווים:

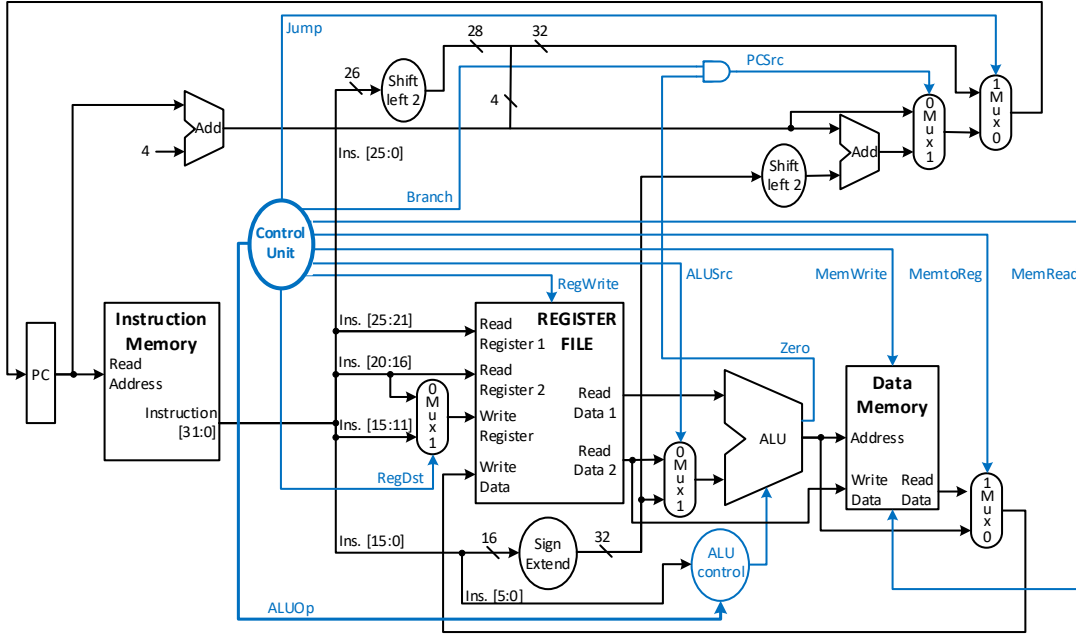
- 1) בכל אחד מסעיפי השאלה מופיעה פקודה מסוימת האורכת מחזור שעון יחיד. יש לסמן בסרטוט הסמוך את מסלול הנתונים וערכי קווי הבקרה שמבצע המעבד בכל מקרה. במקומות בהם הערך אינו ידוע יש לסמן X.
א. פקודת האסמבלי הבאה מתקבלת כאשר $PC = 480$:

add \$17, \$18, \$19



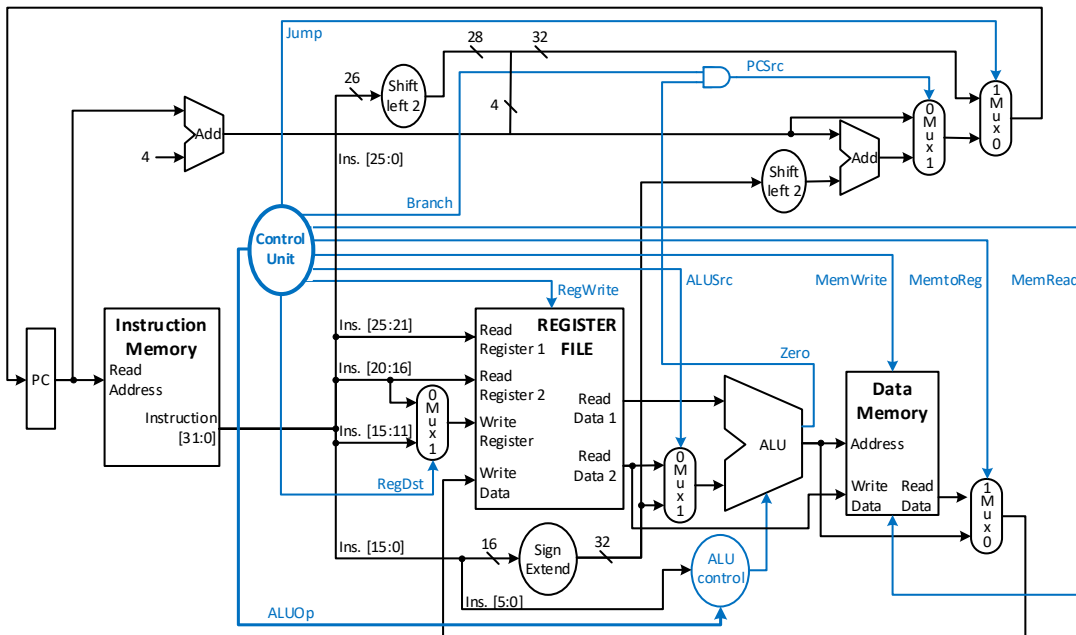
ב. פקודת האסמבלי הבאה כאשר בתחילה $\$20 = 15$:

`addi $20, $20, 84`



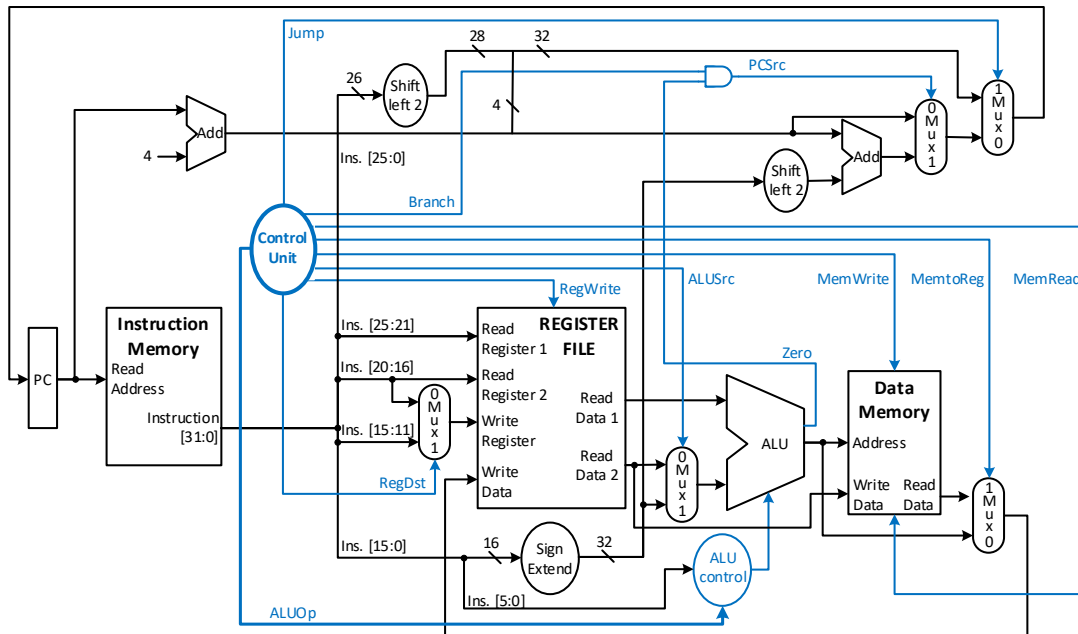
ג. פקודת האסמבלי הבאה מתקבלת כאשר $PC = 0x00608004$:

`beq $8, $9, 0x7B`



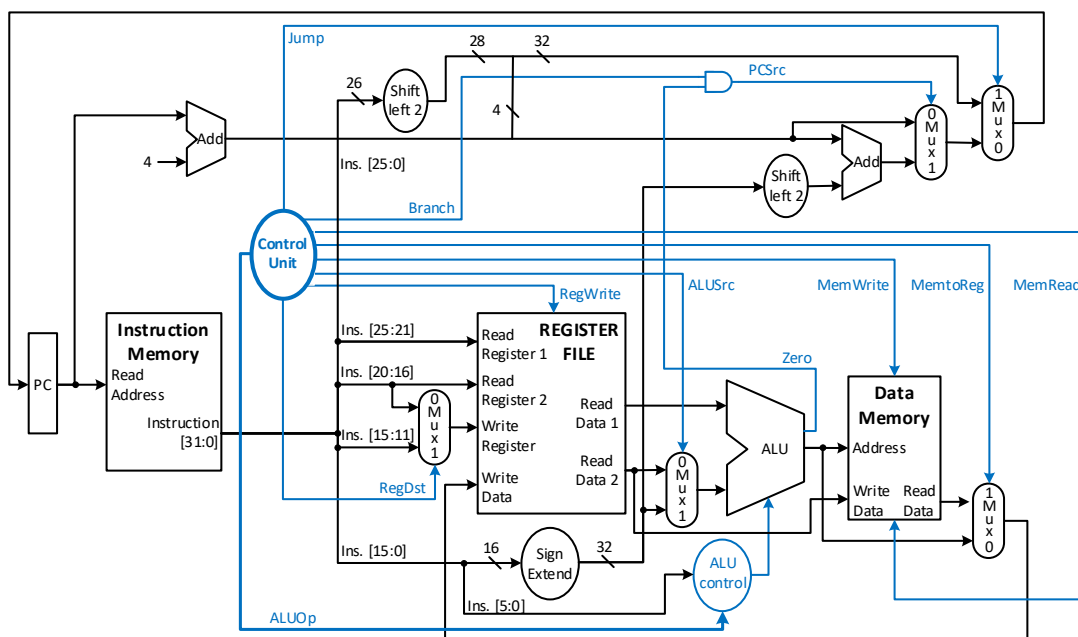
ד. פקודת האסמבלי הבאה מתקבלת כאשר $PC = 0x216072EC$:

j 0x3A50C0



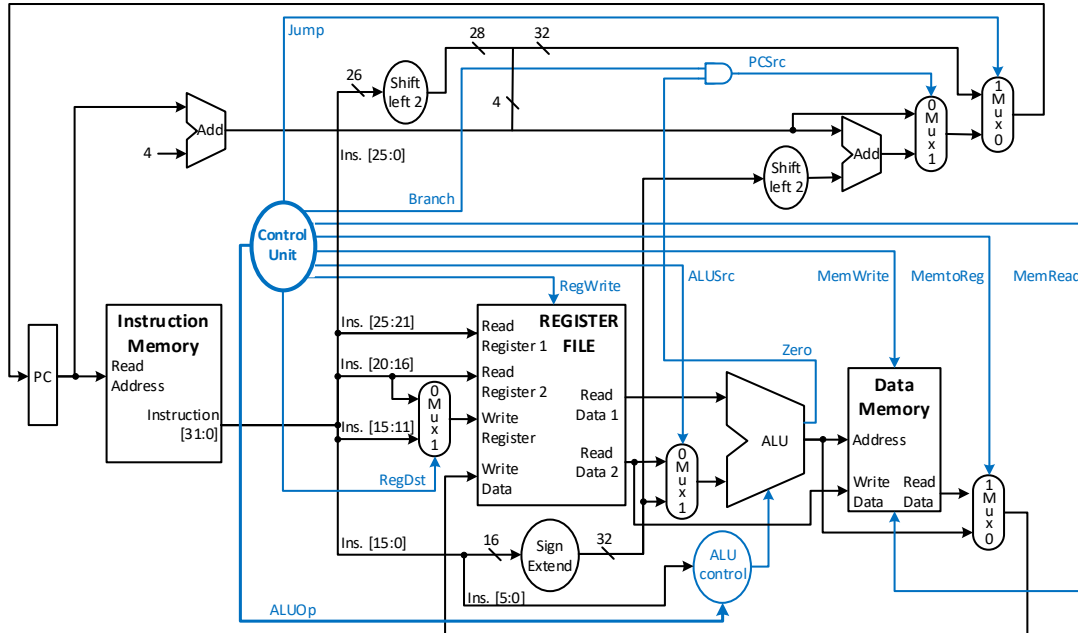
ה. פקודת האסמבלי הבאה כאשר בתחילה $\$16 = 0xABCD1234$:

ori \$16, \$16, 0xF0F0



1. פקודת אסמבלי:

slt \$15, \$10, \$5

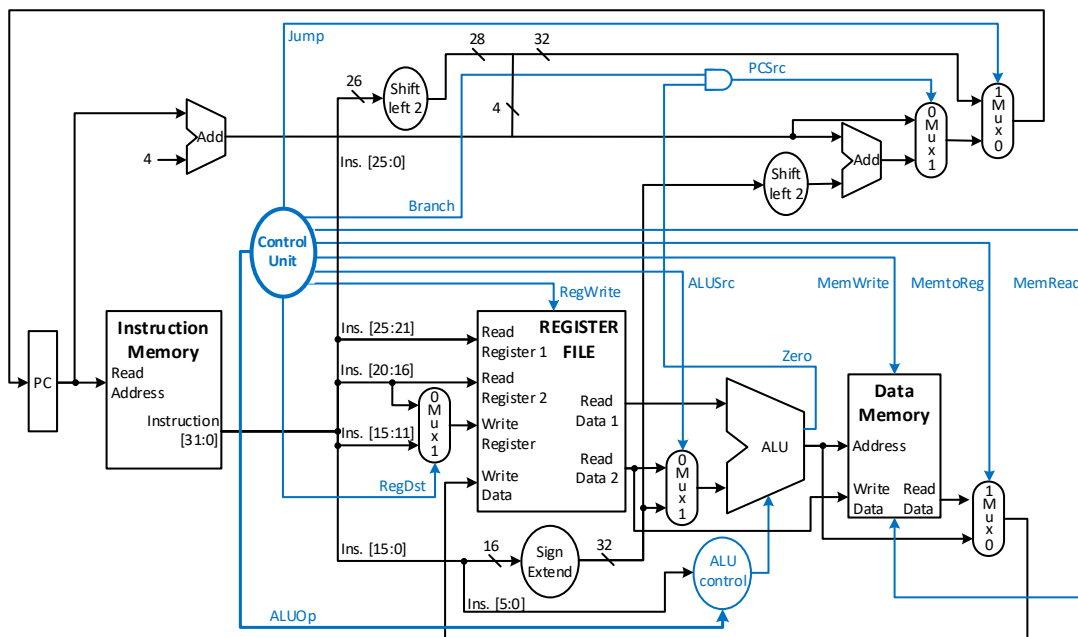


2) בשאלה זו נתמקד בהוספת חומרה וכתיבת ערכי קווי הבקרה הנדרשים למימוש פקודות שאינן ניתנות לביצוע במודל הנוכחי.

א. פקודת lui (Load Upper Immediate).

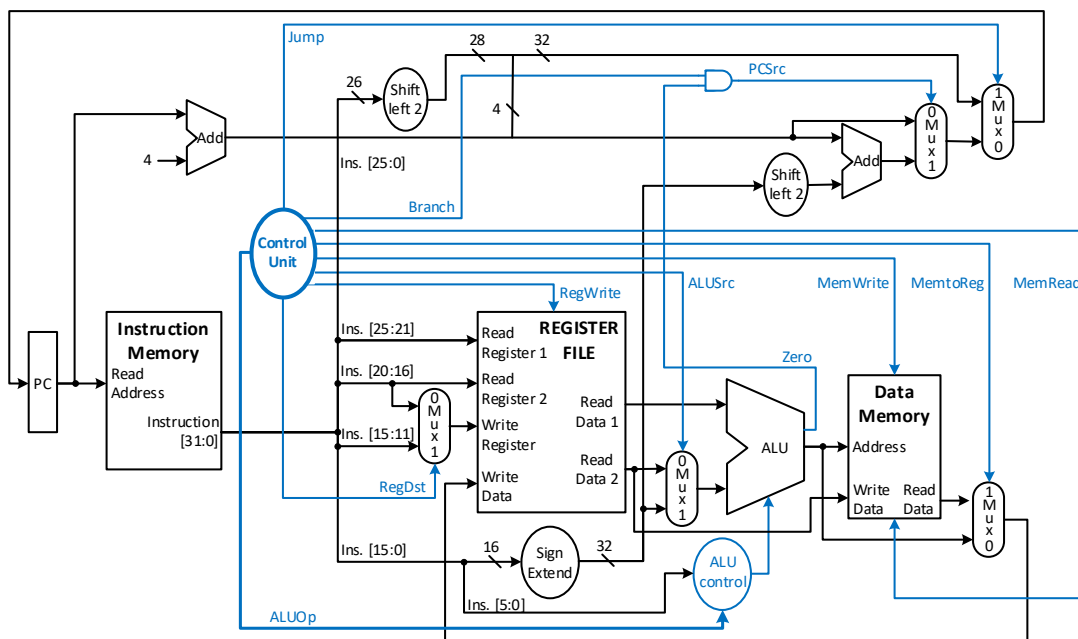
כזכור, פקודת lui לוקחת את הערך בשדה ה-immediate ומזיזה אותו ב-16 מקומות קדימה. את הערך, בן ה-32 ביטים המתקבל, כותבים לאוגר היעד rt (כאשר $rs = 0x0$).
הוסיפו חומרה וקו בקרה מתאים לתרשים המעבד למטה, וציינו את ערכי קווי הבקרה ומסלול הנתונים עבור הפקודה הבאה:

`lui $15, 0xAA00 # The number 0xAA000000 is loaded into $15`



ב. פקודת bne (Branch if Not Equal).
 כזכור, פקודת bne משווה בין הערכים של שני אוגרים ומבצעת קפיצה
 לכתובת יעד אחרת רק אם הערכים לא שווים זה לזה.
 שימו לב! זו היא הפקודה המשלימה ל-beq.
 הוסיפו חומרה וקו בקרה מתאים לתרשים המעבד למטה, וציינו את
 ערכי קווי הבקרה ומסלול הנתונים עבור הפקודה הבאה.
 הניחו כי $PC = 0x43000208$.

```
bne $15, $14, 0xB013 # if: ($15 != $14)
# then: PC = PC + 4 + 4 * 0x0000B013
```



שאלות עם זיהוי פקודות וניתוח פעילות המעבד:

3) נתונים קווי הבקרה הבאים עבור ביצוע פקודה שנקראת LoadSum במעבד MIPS העובד באופן עבודה חד מחזורי (single cycle).

| RegDst | ALUSrc | RegWrite | MemWrite | MemRead | MemtoReg |
|--------|--------|----------|----------|---------|----------|
| 1 | 0 | 1 | 0 | 1 | 1 |

| Branch | Jump | ALUOp1 | ALUOp0 |
|--------|------|--------|--------|
| 0 | 0 | 0 | 0 |

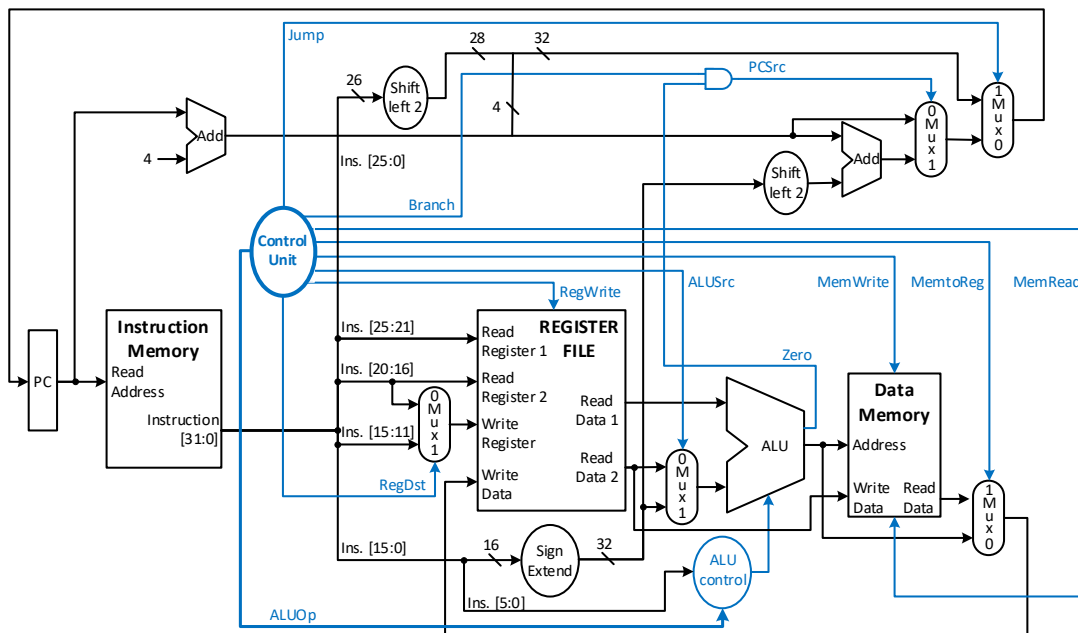
התייחסו לתרשים המעבד החד-מחזורי שבסוף השאלה וענו על הסעיפים הבאים:

- א. (1) מה מבצעת הפקודה LoadSum? אילו ערכים יכתבו ולאן?
 (2) לאיזה פורמט מתאימה הפקודה LoadSum? נמקו.

ב. בהנחה שהפקודה LoadSum מתבצעת עם:

- ערך הסיביות [25:21] (\$rs) הוא 17.
- ערך הסיביות [20:16] (\$rt) הוא 18.
- ערך הסיביות [15:11] (\$rd) הוא 19.
- בכתובת הזיכרון 88000 נמצאת המילה 2224.

כאשר ערך האוגר \$17 הוא 36000 וערך האוגר \$18 במקבץ האוגרים הוא 52000, מה יהיה הערך הסופי והיכן עבור הפקודה LoadSum בסיום מחזור השעון?



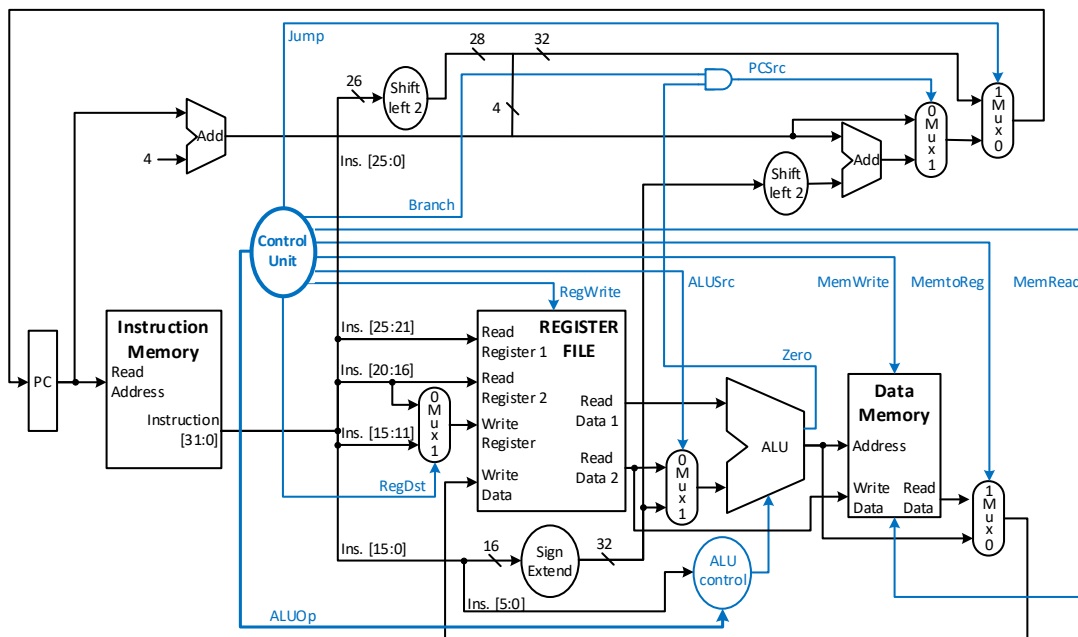
4) נתונים קווי הבקרה הבאים עבור ביצוע פקודה שנקראת Save2Mem במעבד MIPS העובד באופן עבודה חד מחזורי (single cycle).

| RegDst | ALUSrc | RegWrite | MemWrite | MemRead | MemtoReg |
|--------|--------|----------|----------|---------|----------|
| 0 | 1 | 0 | 1 | 0 | 0 |

| Branch | Jump | ALUOp1 | ALUOp0 |
|--------|------|--------|--------|
| 0 | 0 | 0 | 1 |

התייחסו לתרשים המעבד החד-מחזורי שבסוף השאלה וענו על הסעיפים הבאים:

- א. (1) מה מבצעת הפקודה Save2Mem?
(2) לאיזה פורמט ניתן לשייך את הפקודה?
- ב. משנים את אחד מן הערכים של קווי הבקרה כמתואר בכל מקרה. קבעו האם הפקודה תשתנה או לא. נמקו.
 - (1) RegDst = 1
 - (2) RegWrite = 1
 - (3) MemWrite = 0
- ג. ערכי הסיביות [15:0] בפקודה הם 0xFFFF. האם הפקודה חוקית? אם כן אז מתי? נמקו.



שאלות אמריקאיות קצרות:

- (5) עקב שריפה פנימית, הסיביות ALUOp1 ו-ALUOp0 של מעבד MIPS חד מחזורי התקצרו וכעת מתקיים: $ALUOp1 = ALUOp0$. אלו מן הפקודות הבאות לא תעבודנה כשורה?
 א. R-Type בלבד.
 ב. Beq בלבד.
 ג. פקודות lw ו-sw בלבד.
 ד. פקודות R-Type ופקודת beq בלבד.
 ה. כל הפקודות תעבודנה כרגיל.
- (6) במעבד חד מחזורי מסוים נוצר דפקט בייצור והמוצאים מה-Register File התחלפו להם. המוצא Read Data1 התחלף עם המוצא Read Data2. אלו מן הפקודות הבאות תתבצענה כרגיל?
 א. פקודת lw.
 ב. פקודת sw.
 ג. רק פקודות מסוג R-Type תתבצענה כרגיל.
 ד. הדבר לא ישפיע כלל על כל הפקודות שמעבד חד מחזורי מבצע.
 ה. רק פקודות jump ו-beq.
 ו. פקודות lw, sw ו-slt בלבד.
- (7) בנתיב הנתונים החד-מחזורי, פקודת ה-branch נעזרת ביחידת ה-ALU לביצוע חישוב החיסור והדלקת הקו zero במידה וערכי האוגרים שבכניסתה שווים. במידה ו- $zero = 1$, הקפיצה של ה-PC לכתובת היעד תתבצע בוודאות מכיוון ש:
 א. פעולת החיסור המבוצעת ע"י ה-ALU היא מהירה ויעילה מאוד.
 ב. לכן קו הבקרה zero יידלק בטרם ערך ה-PC יעודכן ל- $PC+4$ ולכן הוא יצביע לכתובת היעד הנכונה.
 ג. קו הבקרה Branch נדלק רק לאחר שהערך על zero מתייצב.
 ד. הכתיבה של הערך לתוך ה-PC, בין אם $zero = 0$ ובין אם $zero = 1$, מתבצעת רק בסיום פעימת השעון.
 ה. תשובות ב' ו-ג' בלבד נכונות.
 ו. כל התשובות נכונות.
 ז. אף אחת מן התשובות הנ"ל אינה נכונה ואין ביטחון שהקפיצה לכתובת היעד תתבצע בוודאות מוחלטת.

8) תכנית מסוימת כוללת פקודות המבוצעות ע"י מעבד חד-מחזורי ומחולקות לפי היחסים הבאים:

- 45% פקודות R-Type.

- 5% פקודות lw.

- 15% פקודות sw.

- 30% פקודות beq.

- 15% פקודות jump.

מהו אחוז מחזורי השעון שבהם ערך קו הבקרה ALUSrc מוחזק על 0?

א. 50%

ב. 75%

ג. 45%

ד. 65%

9) יחידת ה-sign extend של מעבד חד-מחזורי התקלקלה וכעת היא מרפדת

באחדים בלבד ב-16 הסיביות הגבוהות. ידוע כי: $\$16 = 40$, $\$17 = -12$.

איזו פקודה לא תתבצע בצורה תקינה?

א. or \$17, \$16, \$17

ב. lw \$16, -12(\$17)

ג. add \$16, \$17, \$16

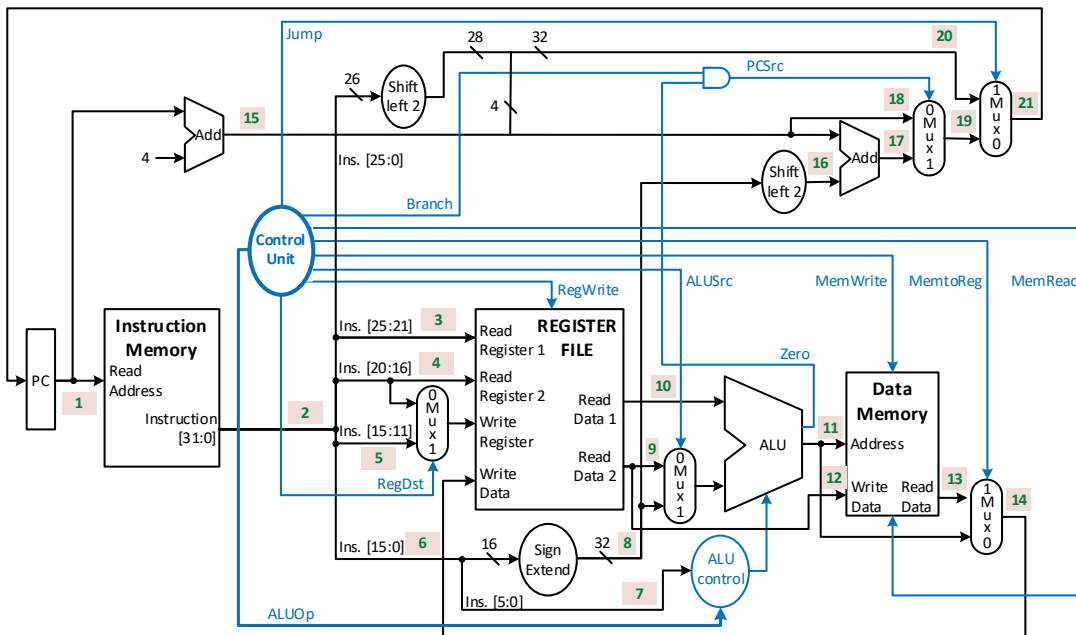
ד. sw \$16, -12(\$17)

ה. lw \$16, 12(\$17)

שאלות מסכמות בניתוח של פעילות מעבד:

10) נתון מעבד חד-מחזורי כמתואר בתרשים שבסוף בשאלה.
 - בשורה 0x8400 בזיכרון נמצאת הפקודה and \$8, \$9, \$18.
 - נתון כי התוכן של כל אוגר במקבץ האוגרים (Register File) החל מהאוגר \$1 הוא: $5^2 - (5 - 1)^2 - 1$ כאשר NumReg הוא מספר האוגר.
לדוגמה: באוגר \$5 נמצא הערך: $5^2 - (5 - 1)^2 = 9$.
 ענו על הסעיפים הבאים:

- א. מה הוא הקידוד של הפקודה and \$8, \$9, \$18 ו-8 ספרות הקסדצימליות?
 ב. במהלך ביצוע הפקודה לעיל, מה הוא ערך הקו המסומן ב-16 והאם יש שימוש בערך זה?
 ג. (1) במהלך ביצוע הפקודה לעיל, מה הוא ערך הקו המסומן ב-17 והאם יש שימוש בערך זה?
 (2) איזה ערך נצפה לקבל על הקו 19?
 ד. מה יהיה הערך שבקו 7?
 ה. מה יהיה הערך שבקו 11?
 ו. עקב תקלה מסוימת הקידוד של הפקודה and \$8, \$9, \$18 השתבש והקידוד שנכנס למעבד הוא 0x01323024 במקום זה שחושב בסעיף א. אלו מקווי הנתונים הבאים ישתנו ולאיזה ערך?
 (1) קו 16 (2) קו 7 (3) קו 11



11) נתון מעבד חד-מחזורי כמתואר בתרשים בסוף השאלה.

- בשורה 0x5408 בזיכרון נמצאת הפקודה `lw $6, 400($18)`.
 - התוכן שנמצא באוגר \$18 הוא 0x1B0C.
 - הניחו כי ניתן לגשת לכל מערך הזיכרון וכי הערך בכל תא שווה לכתובתו מוזתת ב-3 סיביות ימינה וריפוד באפסים עבור 3 סיביות ה-MSB.
- לדוגמה:**

התא שכתובתו 0x98A3 יכיל את הכתובת הבאה:

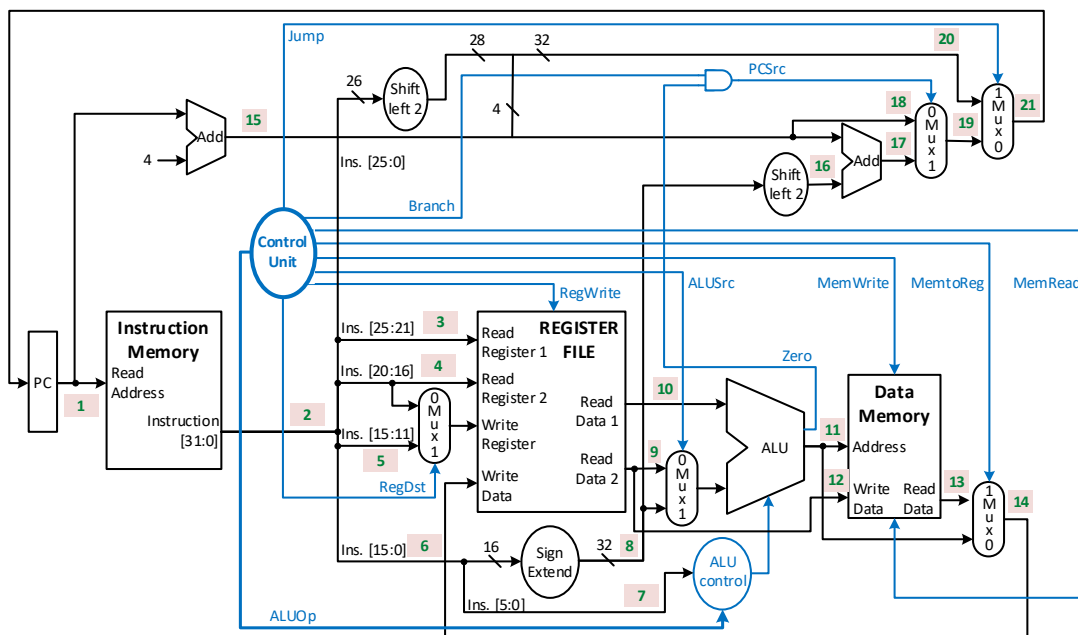
Hex to Bin Representation: 1001 1000 1010 0011

Shifted Left Value by 3 bits: 0001 0011 0001 0100

כלומר התוכן יהיה: 0001 0011 0001 0100 = 0x1314.

ענו על הסעיפים הבאים:

- א. מה הוא הקידוד של הפקודה `lw $6, 400($18)` ב-8 ספרות הקסדצימליות?
- ב. במהלך ביצוע הפקודה לעיל, מה הוא ערך הקו המסומן ב-16? והאם יש שימוש בערך זה?
- ג. האם ניתן לדעת מה הוא ערך הקו המסומן ב-9? והאם יש שימוש בקו זה?
- ד. איזה ערך נצפה לקבל על הקו 11? ומה משמעותו?
- ה. מה יהיה הערך שבקו 14? מה הוא מייצג?
- ו. עקב תקלה, בלוק ה-`sign extend` מבצע ריפוד ב-1 במקום ריפוד לפי סימן. כיצד הדבר ישפיע על ביצוע הפקודה? אלו קווים מבין הקווים שנשאלו בסעיפים הקודמים ישנו את ערכם? ואיזה ערך הם יקבלו?
- ז. עקב תקלה אחרת, סיבית ה-LSB של שדה ה-`Instruction` היא 1, כלומר: `Instruction[0] = 1`. כיצד הדבר ישפיע על ביצוע הפקודה? הסבירו באופן כללי, אין צורך לחשב את ערכי הקווים.

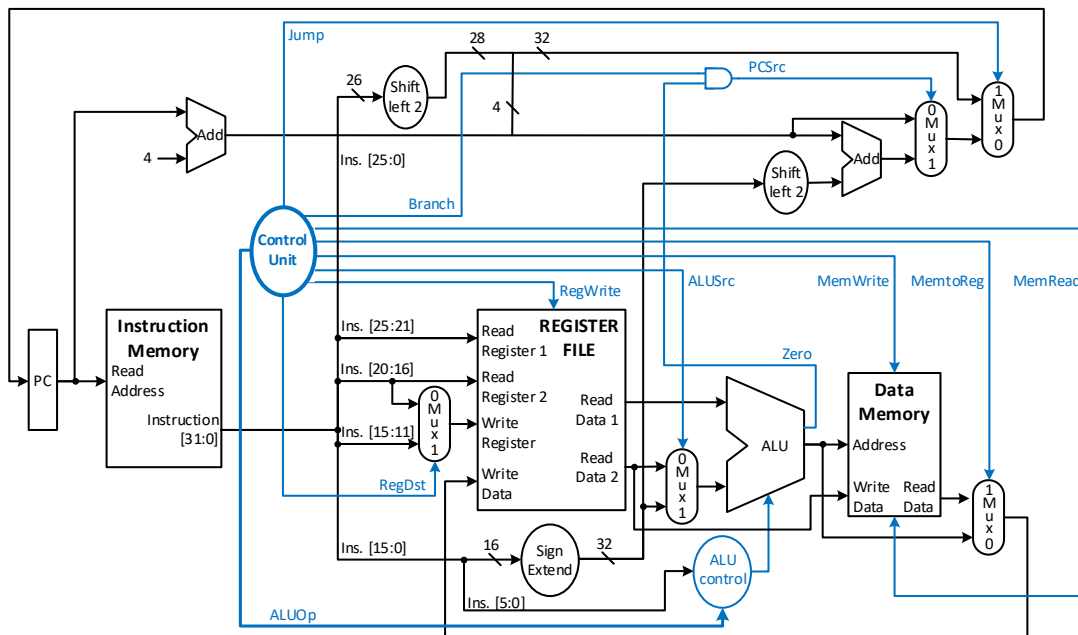


12) מעבד חד-מחזורי מבצע את הפקודה הבאה : $sw \$4, -92(\$8)$.
התוכן של אוגר 8 הוא $0x20200$.

- א. מה הוא הקידוד של הפקודה?
 ב. (1) לאיזו כתובת במערך הזיכרון הפקודה מצביעה?
 (2) האם סיביות הבקרה היוצאות מה-ALU : $carryout[31]$ ו- $arithmatic\ overflow$ תידלקנה? אם כן כיצד יש להתייחס אליהן?
 ג. עקב תקלה מסוימת רכיב ה- $sign\ extend$ מרפד אך ורק באפסים.
 האם הדבר ישפיע על כתובת התא בזיכרון אליו תצביע הפקודה?
 ד. עקב תקלה אחרת, קווי הבקרה שהתקבלו הם :
 מה תבצע הפקודה כעת? נמקו.

| RegDst | ALUSrc | RegWrite | MemWrite | MemRead | MemtoReg |
|--------|--------|----------|----------|---------|----------|
| 1 | 1 | 1 | 0 | 1 | 1 |

| Branch | Jump | ALUOp1 | ALUOp0 |
|--------|------|--------|--------|
| 0 | 0 | 1 | 0 |



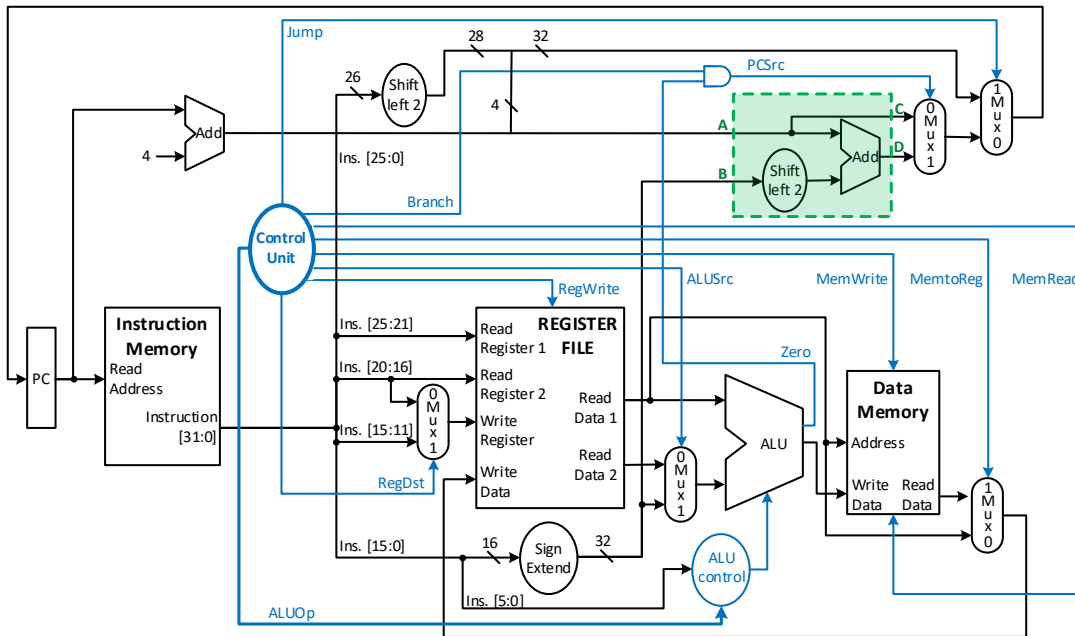
13 באיור אי נתון מעבד חד-מחזורי שעבר התאמה ללקוח מסוים ובו מבצעים את השינוי הבא: מחליפים את חיווט הכניסות לרכיב הזיכרון (ה-Data Memory) שכעת הכניסות ל-Write Data ול-Address שונות ביחס למבנה היסודי שהכרנו.

- איך החיווט הנ"ל משפיע על פעולת המעבד? ועל אילו פקודות?
- בנוסף, מחליפים גם בין רכיב ה-Shift left 2 ורכיב ה-add המסומנים באיור אי במסגרת הירוקה המקווקות. הכניסות מסומנות ב-A,B,C,D והן מופיעות באיור ב'. איך הדבר ישפיע על פעילות המעבד?
- מבצעים פקודה מסוימת עם ערכי קווי הבקרה הבאים. קבעו מה הפקודה מבצעת: (הניחו כי כל השינויים שתוארו בתחילת השאלה ובסעיף ב' תקפים).

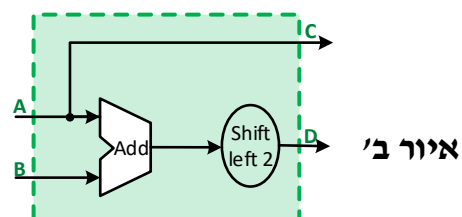
| RegDst | ALUSrc | RegWrite | MemWrite | MemRead | MemtoReg |
|--------|--------|----------|----------|---------|----------|
| 0 | 0 | 0 | 1 | 0 | 0 |

| Branch | Jump | ALUOp1 | ALUOp0 |
|--------|------|--------|--------|
| 1 | 0 | 0 | 1 |

ד. הניחו כי קווי הבקרה שמופיעים בסעיף ג' מתאימים לפקודה הבאה: 0xAD100111. התוכן שבאוגר \$rs = 0x0 והתוכן שבאוגר \$rt = 0xF1F1. הפקודה הנ"ל נמצאת בכתובת 0x101000 שבמרחב הזיכרון. הניחו כי ניתן לגשת לכל מרחב הזיכרון והסבירו מה הפקודה מבצעת. נמקו.



איור א'



איור ב'

תשובות סופיות:

- 1) ראו סימוני ערכים בסרטוני הוידאו.
- 2) ראו בסרטוני הוידאו את החומרה שיש להוסיף בכל מקרה.
- 3) א. (1) הפקודה מבצעת: $\$rd = Mem[\$rs + \$rt]$.
א. (2) הפקודה היא מסוג R-Type מכיוון שקיימת כתיבה למקבץ האוגרים ויש שימוש ב-3 אוגרים.
ב. באוגר שבכתובת המתאימה (19) לסיביות [15:11] תהיה המילה 2224.
- 4) א. (1) הפקודה מבצעת: $Mem[\$rs - Immediate] = \rt
א. (2) הפקודה מתאימה לפורמט I-Type מכיוון שיש לה ערך Immediate ושימוש בשני אוגרים וכן אין כתיבה למקבץ האוגרים.
ב. (1) אין שוני כי לא כותבים למקבץ האוגרים.
ב. (2) יש שוני - הערך של החיסור $\$rs - \rt ייכתב בנוסף לתוך האוגר $\$rt$.
ב. (3) יש שוני - לא תבצע כתיבה כלל (לא לאוגר $\$rt$ ולא לזיכרון).
ג. החיסור המתבצע הוא: $\$rs - Immediate$. אם התוכן של $\$rs$ קטן מהערך 0xFFFF נקבל מספר שלילי המיוצג במשלים ל-2. זה עשוי להוביל לאזורים בזיכרון שלא ניתן לכתוב אליהם. בפועל, ניתן לכתוב רק לכתובות בתחום: [0x7FFFFFFC:0x10000000].
לכן הערכים ש- $\$rs$ יכול להכיל על מנת להימצא באזור הזה הם: [0x8000FFFB:0x1000FFFF].
- 5) ד.
- 6) ה.
- 7) ג.
- 8) ב.
- 9) ה.
- 10) א. 0x01324024 ב. 0x00010090 (אין שימוש בו)
ג. (1) 0x00018494 (אין שימוש בו). ג. (2) $PC + 4 = 0x8404$.
ד. 0x24 ה. 0x1.
ו. בקו 16: 0xC090, בקו 7: 0x24, בקו 11: 0x1.
- 11) א. 0x8E460190 ב. 0x00000640 (אין שימוש בו).
ג. אין אפשרות לדעת ואין שימוש בו.
ד. 0x00001C9C (הכתובת של התא בזיכרון הנתונים ממנו נרצה למשוך את התוכן).
ה. 0x0393 (תוכן התא אליו פנינו בזיכרון הנתונים אשר נכתב לאוגר 6).
ו. בקו 16: 0xFFFF0640 (אין שימוש), בקו 9: אין אפשרות לדעת ואין שימוש בו.
ז. בקו 11: 0xFFFF1C9C (הכתובת החדשה בזיכרון הנתונים), בקו 14: 0xFFFE394.
ז. ערך ה-address לא מתחלק ב-4 ולכן הפקודה לא תקנית.

- (12)** א. $0xAD04FFA4$ ב. $0x000201A4$ (1) ב. (2) הסיביות תידלקנה אך אין להתייחס אליהן.
 סיבית Carry[31] מעידה על הזליגה אך לא בהכרח שתוצאת החישוב שגויה.
 סיבית ה-overflow רלוונטית רק כאשר מחברים שני מספרים שווי סימן,
 וכאן אנו מחברים מספרים שוני סימן.
 ג. כן. נפנה לכתוב : $0x000301A4$.
 ד. הפקודה תבצע AND בין הערכים $0x20200$ ו- $0xFFFFFA4$ ותכתוב את ערך זה לאוגר 31, כלומר : $Mem(0x20200) = \$31$.
- (13)** א. פקודות R-Type : נקבל תמיד : $\$rd = \rs .
 פקודת lw : נצביע לתא שכתובתו היא $\$rs$ והתכן ייכתב ל-rt.
 פקודת sw : נצביע לתא שכתובתו היא $\$rs$ והערך שייכתב הוא : $\$rs + \rt עבור $ALUSrc = 0$ או $\$rs + offset$ עבור $ALUSrc = 1$.
 פקודות beq ו-jump לא ישתנו.
 ב. פקודת beq (וכל פקודות branch) נקפוץ לפקודה הבאה בזיכרון הפקודות :
 $PC (new) = (PC + 4 + offset) \times 4$ (כאשר : $Jump = 0$, $Branch = 1$).
 ג. קווי הבקרה מתאימים לפקודת sw. נכתוב לתא שכתובתו $\$rs$ והתוכן שייכתב הוא $\$rs - \rt .
 בנוסף, אם $\$rs = \rt אז נקפוץ לפקודה : $PC (new) = (PC + 4 + offset) \times 4$.
 ד. הפקודה מבצעת כתיבה לזיכרון הנתונים וכותבת את התוכן : $0xFFFF0E0F$.
 אם התוכן של $\$rs$ ו- $\$rt$ יהיה שווה אז נקפוץ לפקודה בכתובת $0x00404454$ בזיכרון הפקודות.

הערכת ביצועי מעבד חד-מחזורי:

סיכום כללי:

$$\text{CPU Time} = \text{IC} \times \text{CPI} \times \text{CCT} \left[\frac{\text{sec}}{\text{program}} \right]$$

הקשר להערכת ביצועי מעבד:

במעבד חד-מחזורי מתקיים:

- לכל פקודה $\text{CPI} = 1$.
- סט ההוראות אחיד לכל המימושים של מעבד MIPS (IC const.).
- מחזור השעון תלוי בפקודה הארוכה ביותר.

הערות:

- (1) כדי להעריך ביצועי מעבד חד-מחזורי נבחר את משך זמן פקודת lw.
- (2) משכי הזמנים מיוחדים ל-5 שלבי ביצוע פקודה: (I-fetch, Decode, Execute, Memory, Write Back) ולמשכי הזמנים של הבלוקים הצירופיים (2 Shift Left, Sign Extend, add, Mux) את כל אלו נקבל בשאלות.

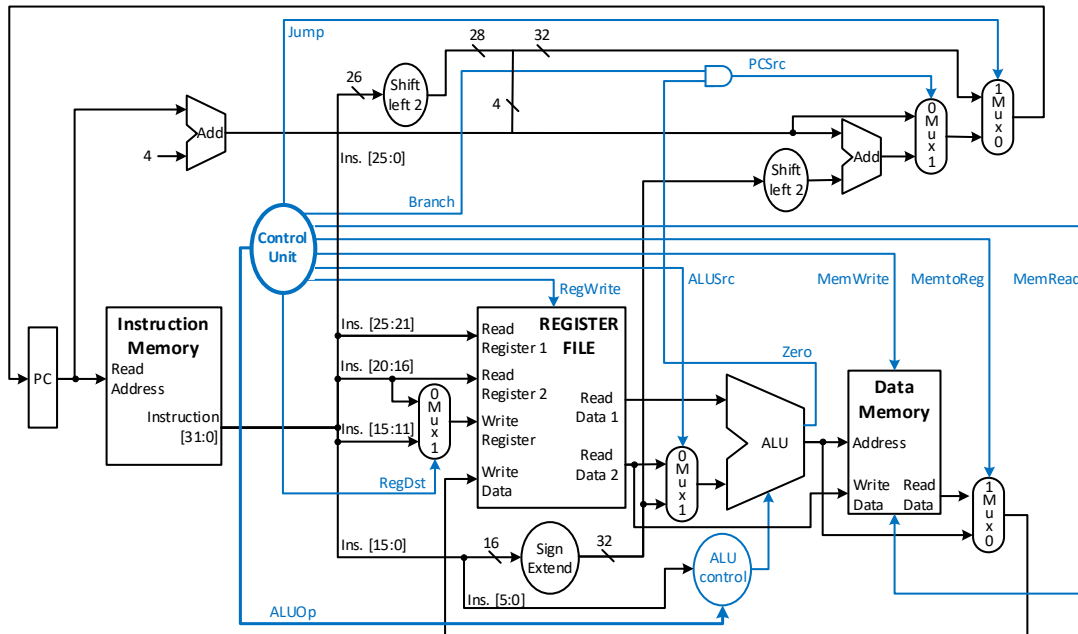
טבלת משכי זמן:

נזכור את השלבים הרלוונטים לכל פקודה במעבד שלנו ונעבוד עם טבלה מהסגנון הזה, כאשר ברוב השאלות נתעניין במציאת משך הזמן הכללי (Total) שלוקח לבצע כל פקודה:

| Ins. | I-fetch | Decode | Execute | Memory | Write Back | Total |
|--------|-------------|-----------|---------|-------------|-------------|-------|
| | Ins. Memory | Read Regs | ALU | Data Memory | Write Regs. | |
| R-Type | V | V | V | | V | |
| lw | V | V | V | V | V | |
| sw | V | V | V | V | | |
| branch | V | V | V | | | |
| jump | V | | | | | |

שאלות:

1) לפניכם תרשים של מעבד חד-מחזורי טיפוסי וטבלת זמני השהיה לכל אחד מן הבלוקים הצירופיים והסדרתיים של המעבד.



| Instruction Memory | Register File | ALU | Data Memory | Add | Mux | Sign-Extend | Shift Left 2 |
|--------------------|---------------|-------|-------------|-------|------|-------------|--------------|
| 350ps | 250ps | 200ps | 800ps | 150ps | 60ps | 100ps | 50ps |

- א. מצאו את משך זמן המעבד הנדרש על מנת לבצע כל אחת מן הפקודות הבאות. בפקודות בהן קיים נתיב קריטי ציינו מהו.

| | | | |
|---------|--------|---------|----------|
| add (1) | lw (2) | beq (3) | jump (4) |
|---------|--------|---------|----------|
- ב. בהנחה כי המעבד נועד לבצע אך ורק את פקודות מסוג R-Type ואת פקודת beq מה יהיה זמן המחזור שלו?
- ג. מה יהיה זמן המחזור המינימלי שיבטיח כי המעבד מסוגל לבצע באופן תקין את כל הפקודות שצוינו בסעיף א'?

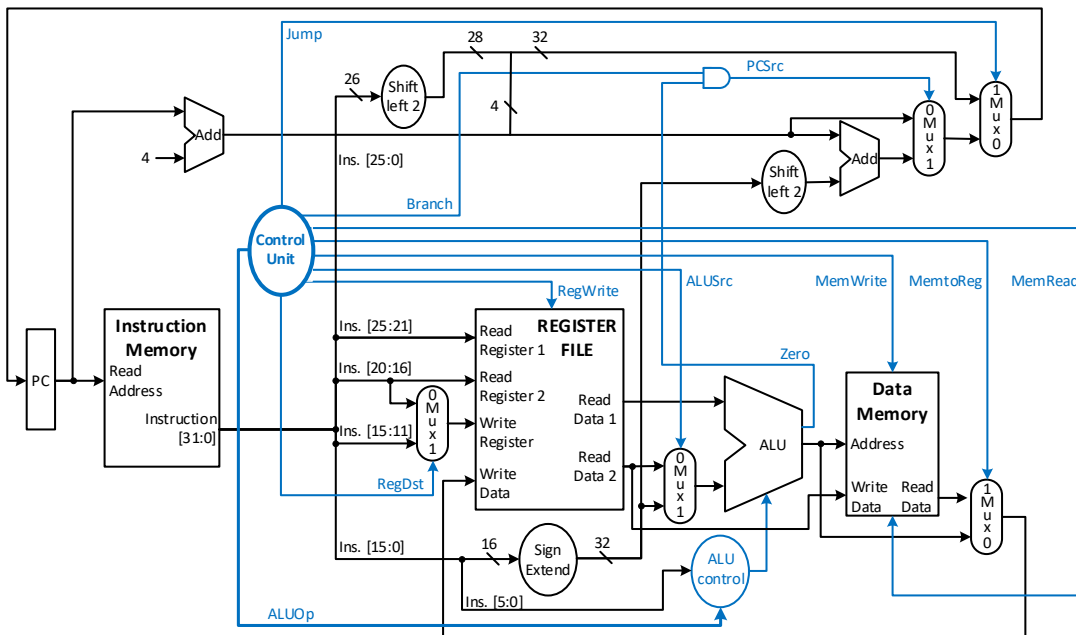
2) מציעים שינוי של פורמט ה-ISA לפקודות טעינה ושמירה של מילה מהזיכרון (lw ו-sw) כך שבמקום גישה לזיכרון כחיבור של ערך האוגר rs עם שדה ה-offset, ניקח את ערך השדה offset לאחר בלוק ה-sign extend והוא יצביע על כתובת היעד בזיכרון. כלומר הפקודה:

```
lw $t0, 830($t1)
```

תהפוך לפסאודו-קוד של סט הפקודות הבא:

```
and $t1, 0
add $t1, $t1, 830
lw $t0, $t1
```

א. איזה שינוי יש לעשות למסלול הנתונים ולקווי הבקרה בעקבות השינוי הנ"ל? סרטטו את השינוי בתרשים הבא:



ב. נבדוק את היעילות של השינוי שהוצע. היות ופקודות טעינה ושמירה של מידע מ-ול- זיכרון הן ארוכות, ריבוי של פקודות קצרות עשוי להצדיק את השינוי. יש להתייחס לזמני ההשהיה הבאים ולמצוא את החלק היחסי המירבי של פקודות הגישה לזיכרון עבורו השינוי יעיל את זמן ההרצה של תכנית נתונה. ניתן להניח בשאלה זו כי הלוגיקה הצירופית אינה מוסיפה להשהיה.

| Instruction memory | Register Read | ALU | Data memory | Register write |
|--------------------|---------------|-----|-------------|----------------|
| 3ns | 2ns | 5ns | 3ns | 2ns |

תשובות סופיות:

- (1) א. 1230ps (1) 1910ps (2) 980ps (3) 460ps (4)
ב. CCT = 1230ps
ג. CCT = 1910ps
- (2) א. ראו שינוי בסרטוט מבנה המעבד בסרטון הוידאו.
ב. כל עוד פחות מ-7.7% מהפקודות הן lw, השינוי ישתלם.