מס' נבחן

חלק א': (30 נקודות)

בחלק זה 6 שאלות רב ברירה. יש לבחור את התשובה הנכונה מבין התשובות האפשריות. משקל כל שאלה 5 נקודות. **יש לסמן באופן ברור ב- X, בטבלה שלפניכם, את התשובה הנכונה**

הערה: **יש לסמן רק אפשרות אחת לכל שאלה**! ייבדקו רק הסימונים בטבלה. כל רישום ליד השאלה עצמה הינו בחזקת טיוטה, ולא ייבדק!!!

שאלה	א	ב	ג	ד	ה
1		X			
2		X			
3	X				
4		X			
5	X				
6				X	

שאלות 1 ו- 2 מתייחסות לתמונת סגמנט הקוד שלהלן, שנלקחה מה MARS:

Address	Code	Basic	Source
0x4040003c	0x200803e8	addi $8,$0,0x000003e8	34: addi $t0, $zero, 1000
0x40400040	0x8d1007d4	lw $16,0x000007d4($8)	35: lw $s0, 2004($t0)
0x40400044	0x11100007	beq $8,$16,0x00000007	36: beq $t0, $s0, exit
0x40400048	0x08100008	j next	37: j next

שאלה 1

לפי הפקודה בשורה 36 בקוד, מהי כתובת התווית (לייבל) **next** בהקסה?

א. 0x00100008

ב. 0x40400020

ג. 0x00400020

ד. 0x40100008

ה. 0x4040004C

שאלה 2

בהתייחסות **לתרשים 4.51** של מעבד הצנרת בנספח תרשימי החומרה, בשילוב תוספת יחידת ה HDU **(תרשים 4.60)** – כיצד ייפתר סיכון הנתונים הקיים בפקודת BEQ (שורה 36) עם אוגר $s0 – המתעדכן בפקודת LW (שורה 35)?

א. כאשר פקודת LW תהיה בשלב MEM, ופקודת BEQ תהיה בשלב EXE ביחידת ה Forwarding יזוהה **Ex/MEM.RegisterRD = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ב. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **MEM/WB.RegisterRd = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ג. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **MEM/WB.RegisterRd = ID/EX.RegisterRs** ותבוצע העברה קדימה לאוגר $s0

ד. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **EX/MEM.RegisterRd = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ה. כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב MEM ביחידת ה Forwarding יזוהה **WB/MEM.RegisterRD = EX/MEM.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

שאלה 3

נתון ערך הקסה 0x41BF0000 המייצג מספר רציונאלי Float בהתאם ל IEEE 754.

כאשר נמיר ערך זה לעשרוני – איזה ערך הוא מייצג?

א. 23.875

ב. 7.875

ג. 3.875

ד. 23.578

ה. 7.578

שאלה 4

נתון מעבד עם 3 סוגי פקודות. בטבלה משמאל יש את ה CPI ואחוז השימוש בכל סוג פקודה.

סוג פקודה	CPI	אחוז שימוש
A	1	30%
B	4	40%
C	3	30%

נתון כי קצב השעון הינו 5.6 Ghz (Giga = 10^9)

נתונה לולאה, שידוע לנו שעובדת 300 מיליון פעמים ($3*10^8$).

ידוע לנו כי לולאה זו עובדת במשך 15 שניות.

כמה פקודות יש בלולאה?

א. 50

ב. 100

ג. $30*10^9$

ד. 17.857

ה. לא ניתן לדעת לפי נתוני השאלה

לפי נתוני השאלה:

$CPI = 0.3*1 + 0.4 * 4 + 0.3* 3 = 2.8$

$15 = (X * 3*10^8) * 2.8 / (5.6*10^9)$

$X = 15 * (5.6*10^9) / (3 * 10^8 * 2.8) = 100$

שאלה 5

נתון זיכרון מטמון בהתאם לנתונים הבאים:

- גודל בלוק – 64 words
- 2 ways set associative
- יש 2048 שורות במטמון

נתונה הכתובת (32 סיביות) הבאה: 0x4F512345. מה צריך להיות הערך הבינארי של ה TAG בכדי שיזוהה HIT?

א. 0100111101010

ב. 010011110101000

ג. 00100100011

ד. 001101000101

ה. אין אפשרות לדעת מהנתונים של השאלה

לפי נתוני השאלה: k = 2, m = 6, n = 11, tag = 13

הכתובת בתרגום לסיביות:

0100 1111 0101 0001 0010 0011 0100 0101

שאלה 6

שיטת חיזוי דינאמי של Brach (Dynamic Branch Prediction) באופן עבודה בצנרת היא:

א. הטכניקה בה נבצע את סידור פקודות branch בקוד (scheduling) בצורה דינאמית, באופן המצמצם את מספר ההשהיות (stalls)

ב. הטכניקה בה נבצע שינוי דינאמי ונעביר את ביצוע הפקודה Branch לשלב שני בצנרת על מנת לצמצם את מספר השטיפות (flush) שיתבצעו אם פקודת ה Branch תילקח (Branch Taken).

ג. הטכניקה בה נחזרה את התנהגות הפקודה Branch על סמך חישוב דינאמי של ה Branch Target, במהלך ביצוע הפקודה וניתוח ושמירת ערך זה בתוך המעבד

ד. הטכניקה בה נחזה את התנהגות פקודת ה- Branch על סמך המידע של ההתנהגות הקודמת של ה Branch הנשמרת ומנותחת בתוך המעבד

ה. תשובות (א), (ב) ו- (ד) נכונות

חלק ב' – שאלות פתוחות (70 נקודות)

בחלק זה 2 שאלות. יש לענות על שאלות אלה בטופס המבחן בהתאם להנחיות בסעיפים השונים

שאלה 7 (45 נקודות):

Address	Code	Basic	Source
0x0040000c	0x02004020	add $8,$16,$0	16: add $t0, $s0, $zero
0x00400010	0x1100fffb	beq $8,$0,0xfffffffb	17: beq $t0, $zero, cont
0x00400014	0x8d090010	lw $9,0x00000010($8)	18: lw $t1, 16($t0)
0x00400018	0x21290004	addi $9,$9,0x00000004	19: addi $t1, $t1, 4
0x0040001c	0xad2a00a0	sw $10,0x000000a0($9)	20: sw $t2, 160($t1)

נתונה תמונת הקוד שלהלן, הלקוחה מה- MARS, באסמבלי של ה MIPS-32:

נתונים הערכים ההתחלתיים של האוגרים:

- $s0 = 0x00000050
- $t2 = 0xABCD1234
- כל שאר האוגרים, ערכם שווה לערך מספר האוגר בהקסה ($0 = 0, $1 =1, ... $10 = 0xa,... $20 = 0x140,... $31 = 0x1F0)

ניתן לגשת לכל מרחב הזיכרון בפקודות LW ו- SW (כל עוד הכתובת מתחלקת ב- 4).

ערך 200 **הבתים** (**Bytes**) הראשונים בזיכרון הינו כערך 2 ספרות ההקסה הנמוכות של כתובת הבית

לדוגמה: בית בכתובת 0x00000054 ערכו 0x54

בית בכתובת 0x000000A8 ערכו 0xA8

נתונים אלה תקפים לכל סעיפי שאלה 7

סעיף 7.1: מעקב (4 נקודות):

מה ערך אוגר $t1, **בבסיס הקסה**, לאחר פקודת LW (פקודה שלישית, שורה 18)?

0x63626160

אחרי פקודה 16: $t0 = 0x00000050

בפקודה 18, הכתובת אליה ניגשים הינה: $t0 + 0x10 = 0x00000060

פונים ל- 4 הבתים המסומנים, ולפי little endian order נקבל את הערך

מבנה הזיכרון הוא:

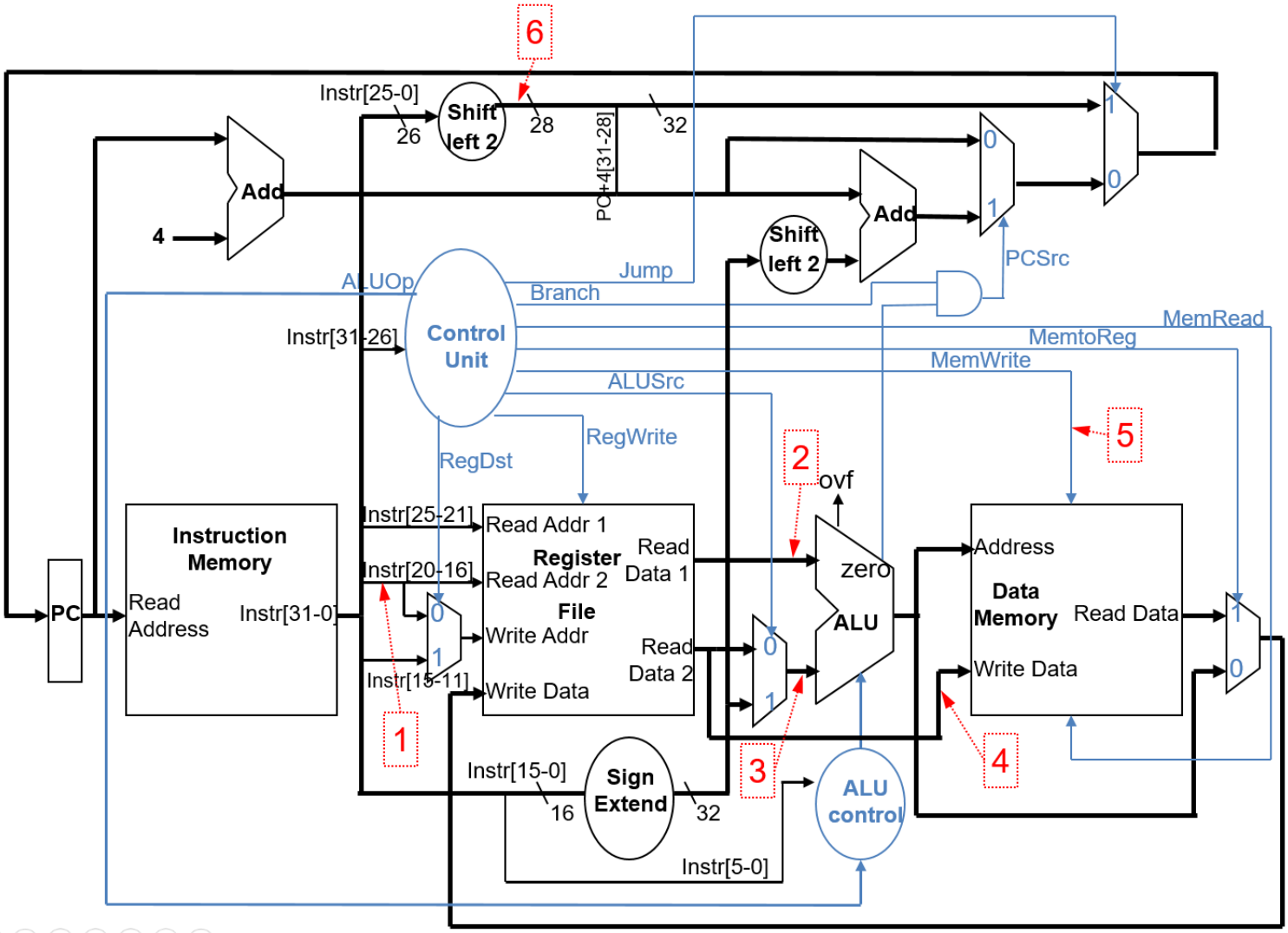
כתובת	0x65	0x64	0x63	0x62	0x61	0x60	0x5f	0x5e	0x5d	0x5c	0x5b	0x5a
ערך הבית	0x65	0x64	0x63	0x62	0x61	0x60	0x5f	0x5e	0x5d	0x5c	0x5b	0x5a

סעיף 7.2: מעבד חד מחזורי (18 נקודות):

בתרשים שלהלן מעבד חד מחזורי, ובו סימונים 1-6 על קווים מסוימים של המעבד. עליכם לחשב את הערכים העוברים על קווים אלה, בהנחה שהמידע נבדק לקראת סוף פעימת השעון בביצוע הפקודה **האחרונה** המתוארת בעמוד הקודם: **sw $t2, 160($t1)** (הפקודה בשורה 20 של הקוד)

את תשובתכם יש למלא בטבלה שבתחתית העמוד – יש למלא את הנתונים בערכים בבסיס הקסה באמצעות הסימון 0x.... (3 נקודות לסימון)

- ערך לא ידוע – יש לסמן ב- X
- ניתן להיעזר בנתונים מטבלאות 4.12 ו- 4.18 בנספח התרשימים של המבחן.



נקודה	ערך (בהקסה)
1	0xA (מס' אוגר $t2, 10)
2	0x63626164 (ערך אוגר $t1 לאחר פקודה בשורה 19)
3	0xA0 (ערך מיידי – 160)

נקודה	ערך (בהקסה)
4	0Xabcd1234 (אוגר $t2)
5	0x1 (דגל כתיבה בזיכרון)
6	0x4A80280 (28 סיביות: 0-25 אחרי הזזה 2 מקומות)

סעיף 7.3: טיפול בסיכוני נתונים (5 נקודות):

א. (2 נקודות) רשמו את סיכוני הנתונים הקיימים בקטע הקוד הנתון. כל סיכון נתונים יש לרשום באופן הבא: אוגר X בין פקודה בשורה Y לבין שורה Z

בין פקודה בשורה 17 לשורה 16 – אוגר $t0

בין פקודה בשורה 18 לשורה 16 – אוגר $t0

בין פקודה בשורה 19 לשורה 18 – אוגר $t1

בין פקודה בשורה 20 לשורה 19 – אוגר $t1

ב. (3 נקודות) כיצד ייפתר סיכון הנתונים הקיים בין הפקודה בשורה 18 לבין הפקודה בשורה 16. התייחסות בתשובתכם לאפשרויות של טיפול על ידי יחידת העברה קדימה (forwarding unit), ו/או Hazzard Detection Unit (HDU) ו/או חציית מקבץ האוגרים.

יש להתייחס למיקום הפקודות (כל פקודה, באיזה שלב של המעבד נמצאת (IF, ID, EX, MEM, WB).

במידה ויש שימוש ביחידת העברה קדימה, יש להתייחס לאופן הזיהוי באופן מפורט גם עם התייחסות למקרי הבדיקה של יחידת ההעברה (מצ"ב), וכן לדגלים היוצאים אל ה MUS לביצוע ההעברה קדימה.

מקרים לבדיקה של יחידת העברה קדימה 1a,1b,2a,2b

1a. EX/MEM.RegisterRd = ID/EX.RegisterRs
1b. EX/MEM.RegisterRd = ID/EX.RegisterRt
2a. MEM/WB.RegisterRd = ID/EX.RegisterRs
2b. MEM/WB.RegisterRd = ID/EX.RegisterRt

יש להתייחס בפתרון לתרשים 4.60 המופיע בנספח החומרה של המבחן.

סיכון הנתונים בין 2 פקודות אלה הינו עם אוגר $t0.

כאשר פקודת ADD בשורה 16 תהיה בשלב WB, ופקודת LW בשורה 18 תהיה בשלב EX

תזהה יחידת ההעברה קדימה את הצורך/אפשרות לביצוע העברה קדימה.

הזיהוי יהיה לפי מקרה 2a (אוגר $t0 בפקודת LW הינו אוגר Rs)

יחידת ההעברה קדימה תעביר את הערך 1 – אל ה MUX "העליון" שלפני ה ALU (המקבל את

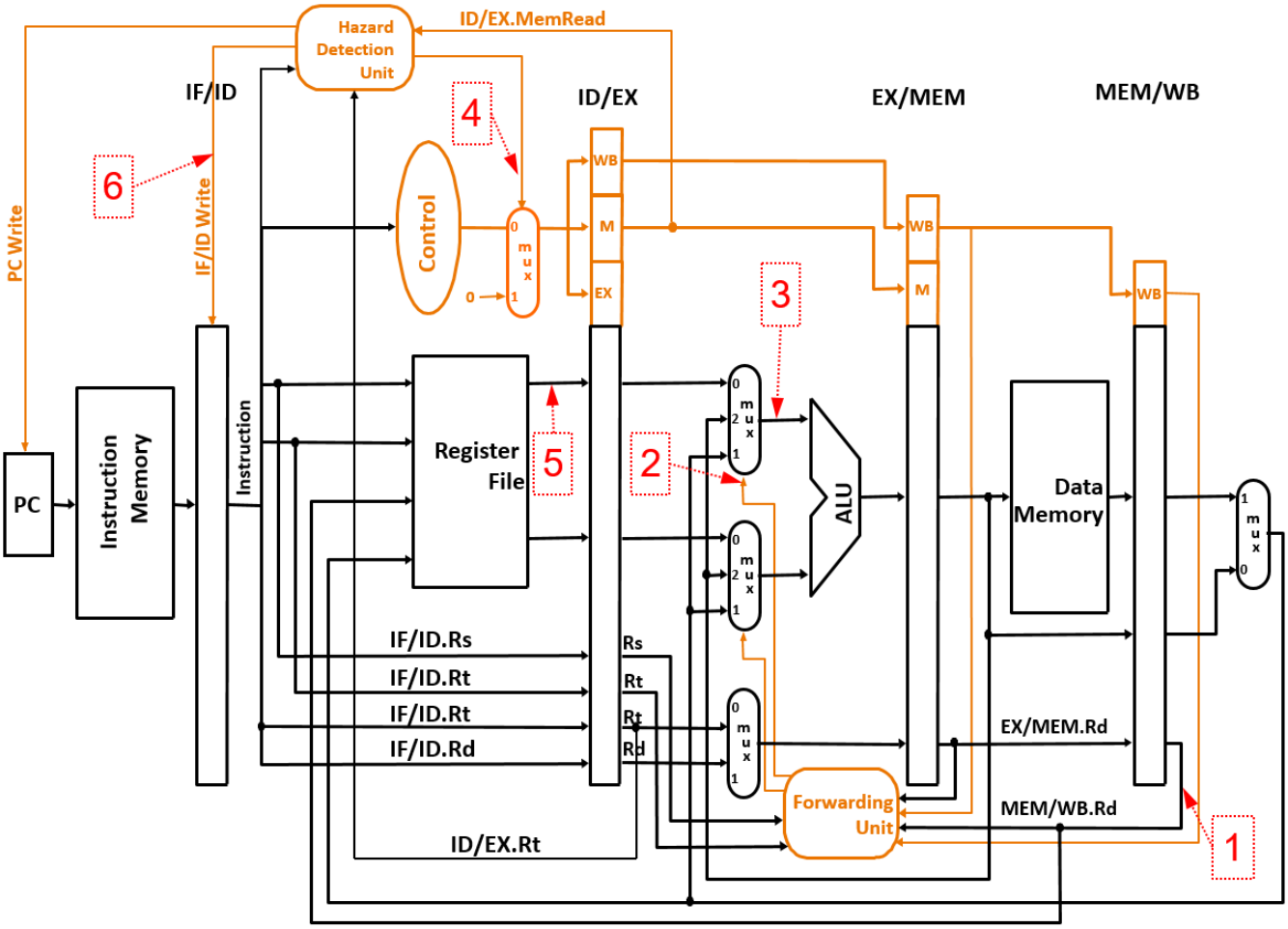
נתוני אוגר Rs) כדי שיבחר הערך המקודם משלב WB אל שלב EX

סעיף 7.4: מעבד צנרת (18 נקודות):

בתרשים שלפניכם, של מעבד MIPS העובד בטכנולוגיית **צנרת** (המבוסס על תרשים 4.60 בנספח) מתוארת סוף פעימת השעון **השלישית** בביצוע הקוד שבשאלה, בהתאם לנתון בתחילת השאלה.

בתרשים מסומנות 6 נקודות (1-6) על קווים מסוימים של המעבד. יש למלא בטבלה שבתחתית העמוד את הערכים על קווים אלו בהתייחסות להנחות הבאות: סיכוני נתונים בהרצת הקוד נפתרים על ידי יחידת ההעברה קדימה (Forwarding Unit), יחידת איתור סיכונים (HDU) וחציית מקבץ האוגרים.

IF	ID	EX	MEM	WB
sw $t2, 160($t1)	addi $t1, $t1, 4	lw $t1, 16($t0)	beq $t0, $zero, cont	add $t0, $s0, $zero



נקודה	ערך (בהקסה)
1	0x8 (מס' אוגר $t0)
2	0x1 (בורר את הערך המקודם משלב WB)
3	0x63626160 (4 בתים מהזיכרון, החל מכתובת 0x60)

נקודה	ערך (בהקסה)
4	0x1 (בורר כניסה עם 0 – לאיפוס דגלי הבקרה)
5	0x00000001 (ערך מקורי של אוגר $t1 בתחילת הביצוע לפני קידום)
6	0x0 (מניעת כתיבה על IF/ID כדי לייצר מרווח פקודה עם LW)

שאלה 8: (25 נקודות)

עליכם לכתוב פרוצדורה בשם **sumRevDiagonalFrom**.

הפרוצדורה מקבלת 4 פרמטרים:

- $a0 – כתובת של מערך דו-ממדי ריבועי. הערכים במערך הינם ערכים עם סימן (signed) וגודל כל איבר במערך הינו **מילה (WORD)**.
- $a1 – מספר האיברים בכל שורה/עמודה (תזכורת: זה מערך דו-ממדי ריבועי)
- $a2 – מספר שורת התחלה (ערך בין 0 למספר השורות פחות אחד)
- $a3 – מספר עמודת התחלה (ערך בין 0 למספר העמודות פחות אחד)

הערכים באוגרים $a2 ו- $a3 הם אינדקס של שורה/עמודה המציינים את נקודת ההתחלה של משימת הפרוצדורה.

הפרוצדורה צריכה לחשב את סכום "**האלכסון ההפוך**" מנקודת ההתחלה (לפי אוגרים $a2 ו- $a3) עד לקצה המערך (מצד שמאל, או מלמטה).

דוגמה להמחשה:

המערך הדו-ממדי משמאל הינו בגודל 7x7

עבור נקודת התחלה [4][1] – "האלכסון ההפוך" מסומן ברקע אפור מנקודת התחלה זו – שמאלה ולמטה – עד לנקודה [0][5]. המשך מעבר לנקודה זו – מהווה גלישה מהמערך לעמודה 1- (שכמובן אינה קיימת...). הסכום במקרה זה הוא: 20

עבור נקודת התחלה [6][4] – "האלכסון ההפוך" מסומן ברקע אפור מנקודת התחלה זו – שמאלה ולמטה – עד לנקודה [4][6]. המשך מעבר לנקודה זו – מהווה גלישה מהמערך לשורה 7 (שכמובן אינה קיימת...). הסכום במקרה זה הוא: 12

	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	2	0	0
2	0	0	0	3	0	0	0
3	0	0	4	0	0	0	0
4	0	5	0	0	0	0	3
5	6	0	0	0	0	4	0
6	0	0	0	0	5	0	0

הנחיות לפתרון:

- יש לכתוב את הפרוצדורה, בלבד, לפי כללי העבודה שלמדנו בכתיבת פרוצדורות.
- ניתן להשתמש בכל הפקודות כפי ש MARS מקבל (כלומר...עובר קומפילציה ב MARS). הקפידו על כך, והקפידו "שלא להמציא" פקודות שאינן קיימות.
- הפתרון חייב להשתמש אך ורק בפרמטרים שהתקבלו, **ללא** כל שימוש/תוספת של משתנים במקטע הנתונים (.data).
- הפתרון צריך להתאים לכל גודל מערך דו ממדי ריבועי (ולא רק למערך של 7x7 שניתן בדוגמה).
- ניתן להניח שהערכים בפרמטרים (אוגרים $a0.... $a3) תקינים – כלומר, **אין צורך** לבצע בדיקות תקינות.
- הפרוצדורה מחזירה ערך שהוא סכום "האכלסון ההפוך" מנקודת ההתחלה שניתנה באוגרים $a2, $a3.

בעמוד הבא נתונה לכם תוכנית ראשית המשתמשת בפרוצדורה שעליכם לכתוב. היעזרו בקוד התוכנית הראשית בכדי להבין כיצד מופעלת הפרוצדורה.

את הפתרון יש לרשום על גבי טופס המבחן, בהמשך, במקום המיועד לכך.

תוכנית ראשית, לדוגמה, המשתמשת בפרוצדורה שעליכם לכתוב:

```
################## Data segment #######################
.data
matrix: .word        0 , 0 , 0 , 0 , 0 , 0 , 0 ,
                     0 , 0 , 1 , 1 , 2 , 0 , 0 ,
                     0 , 1 , 0 , 3 , 0 , 1 , 0 ,
                     0 , 1 , 4 , 0 , 0 , 1 , 0 ,
                     0 , 5 , 0 , 0 , 0 , 1 , 3 ,
                     6 , 0 , 1 , 1 , 1 , 4 , 0 ,
                     0 , 0 , 0 , 0 , 5 , 0 , 0
length:        .byte  7
startRow:      .byte 1
startCol:      .byte 4
################## Code segment #######################
.text
.globl main
main:  # main program entry
        la $a0, matrix
        lbu $a1, length
        lbu $a2, startRow
        lbu $a3, startCol
        jal sumRevDiagonalFrom

        move $a0, $v0
        li $v0, 1
        syscall

        li $a0, '\n'
        li $v0, 11
        syscall                         # new line

        la $a0, matrix
        lbu $a1, length
        li $a2, 4
        li $a3, 6
        jal sumRevDiagonalFrom

        move $a0, $v0
        li $v0, 1
        syscall                         # print substruction of nodes' value

exit:
        li $v0, 10      # Exit program
        syscall
```

פתרון שאלה 8:

```
##################### the solution ############################
sumRevDiagonalFrom:
        addi $t0, $a1, -1           # $t0 - increment size for reverse diagonal
        sll $t0, $t0, 2             # multiply by 4 - for WORD

        li $v0, 0                   # $v0, is the sum - initialize with 0

        mul $t1, $a1, $a2           # $t1 - number of ELEMENTS to skip to row
                                    # beginning
        add $t1, $t1, $a3           #     - number of ELEMENTS to skip to column
                                    #       beginning at starting point
        sll $t1, $t1, 2             #     - multiply by 4 - for WORD
        add $t1, $t1, $a0           #     - accurate address of the starting [row][col]

sumLoop:
        lw $t2, 0($t1)          # get current value
        add $v0, $v0, $t2       # accumulate the sum
        add $t1, $t1, $t0       # next position on the reversed diagonal
        addi $a2, $a2, 1            # increment the row to next row
        bge $a2, $a1, sumFinish     # in case of reaching last row (bottom row) –
                                    #stop calculate
        addi $a3, $a3, -1       # decrement the col to next col
        bge $a3, $zero, sumLoop         # in case of not reaching the left side end –
                                        # continue the sum
sumFinish:
                jr $ra
```

חלק א': (30 נקודות)

בחלק זה 6 שאלות רב ברירה. יש לבחור את התשובה הנכונה מבין התשובות האפשריות. משקל כל שאלה 5 נקודות. **יש לסמן באופן ברור ב- X, בטבלה שלפניכם, את התשובה הנכונה**

הערה: **יש לסמן רק אפשרות אחת לכל שאלה**! ייבדקו רק הסימונים בטבלה. כל רישום ליד השאלה עצמה הינו בחזקת טיוטה, ולא ייבדק!!!

שאלה	א	ב	ג	ד	ה
1					
2					
3					
4					
5					
6					

שאלות 1 ו- 2 מתייחסות לתמונת סגמנט הקוד שלהלן, שנלקחה מה MARS:

Address	Code	Basic	Source
0x4040003c	0x200803e8	addi $8,$0,0x000003e8	34: addi $t0, $zero, 1000
0x40400040	0x8d1007d4	lw $16,0x000007d4($8)	35: lw $s0, 2004($t0)
0x40400044	0x11100007	beq $8,$16,0x00000007	36: beq $t0, $s0, exit
0x40400048	0x08100008	j next	37: j next

שאלה 1

לפי הפקודה בשורה 36 בקוד, מהי כתובת התווית (לייבל) **next** בהקסה?

א. 0x00100008

ב. 0x40400020

ג. 0x00400020

ד. 0x40100008

ה. 0x4040004C

שאלה 2

בהתייחסות **לתרשים 4.51** של מעבד הצנרת בנספח תרשימי החומרה, בשילוב תוספת יחידת ה HDU **(תרשים 4.60)** – כיצד ייפתר סיכון הנתונים הקיים בפקודת BEQ (שורה 36) עם אוגר $s0 – המתעדכן בפקודת LW (שורה 35)?

א. כאשר פקודת LW תהיה בשלב MEM, ופקודת BEQ תהיה בשלב EXE ביחידת ה Forwarding יזוהה **Ex/MEM.RegisterRD = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ב. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **MEM/WB.RegisterRd = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ג. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **MEM/WB.RegisterRd = ID/EX.RegisterRs** ותבוצע העברה קדימה לאוגר $s0

ד. במעבר של פקודת LW משלב EXE לשלב MEM, יוכנס Bubble לשלב EXE ופקודת BEQ תישאר בשלב ID.

כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב EXE ביחידת Forwarding יזוהה **EX/MEM.RegisterRd = ID/EX.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

ה. כאשר פקודת LW תהיה בשלב WB, ופקודת BEQ תהיה בשלב MEM ביחידת ה Forwarding יזוהה **WB/MEM.RegisterRD = EX/MEM.RegisterRt** ותבוצע העברה קדימה לאוגר $s0

שאלה 3

נתון ערך הקסה 0x41BF0000 המייצג מספר רציונאלי Float בהתאם ל IEEE 754.

כאשר נמיר ערך זה לעשרוני – איזה ערך הוא מייצג?

א. 23.875

ב. 7.875

ג. 3.875

ד. 23.578

ה. 7.578

שאלה 4

נתון מעבד עם 3 סוגי פקודות. בטבלה משמאל יש את ה CPI ואחוז השימוש בכל סוג פקודה.

סוג פקודה	CPI	אחוז שימוש
A	1	30%
B	4	40%
C	3	30%

נתון כי קצב השעון הינו 5.6 Ghz ($Giga = 10^9$)

נתונה לולאה, שידוע לנו שעובדת 300 מיליון פעמים ($3*10^8$).

ידוע לנו כי לולאה זו עובדת במשך 15 שניות.

כמה פקודות יש בלולאה?

א. 50

ב. 100

ג. $30*10^9$

ד. 17.857

ה. לא ניתן לדעת לפי נתוני השאלה

שאלה 5

נתון זיכרון מטמון בהתאם לנתונים הבאים:

- גודל בלוק – 64 words
- 2 ways set associative
- יש 2048 שורות במטמון

נתונה הכתובת (32 סיביות) הבאה: 0x4F512345. מה צריך להיות הערך הבינארי של ה TAG בכדי שיזוהה HIT?

א. 0100111101010

ב. 010011110101000

ג. 00100100011

ד. 001101000101

ה. אין אפשרות לדעת מהנתונים של השאלה

שאלה 6

שיטת חיזוי דינאמי של Brach (Dynamic Branch Prediction) באופן עבודה בצנרת היא:

א. הטכניקה בה נבצע את סידור פקודות branch בקוד (scheduling) בצורה דינאמית, באופן המצמצם את מספר ההשהיות (stalls)

ב. הטכניקה בה נבצע שינוי דינאמי ונעביר את ביצוע הפקודה Branch לשלב שני בצנרת על מנת לצמצם את מספר השטיפות (flush) שיתבצעו אם פקודת ה Branch תילקח (Branch Taken).

ג. הטכניקה בה נחזרה את התנהגות הפקודה Branch על סמך חישוב דינאמי של ה Branch Target, במהלך ביצוע הפקודה וניתוח ושמירת ערך זה בתוך המעבד

ד. הטכניקה בה נחזה את התנהגות פקודת ה- Branch על סמך המידע של ההתנהגות הקודמת של ה Branch הנשמרת ומנותחת בתוך המעבד

ה. תשובות (א), (ב) ו- (ד) נכונות

חלק ב' – שאלות פתוחות (70 נקודות)

בחלק זה 2 שאלות. יש לענות על שאלות אלה בטופס המבחן בהתאם להנחיות בסעיפים השונים

שאלה 7 (45 נקודות):

Address	Code	Basic	Source
0x0040000c	0x02004020	add $8,$16,$0	16: add $t0, $s0, $zero
0x00400010	0x1100fffb	beq $8,$0,0xfffffffb	17: beq $t0, $zero, cont
0x00400014	0x8d090010	lw $9,0x00000010($8)	18: lw $t1, 16($t0)
0x00400018	0x21290004	addi $9,$9,0x00000004	19: addi $t1, $t1, 4
0x0040001c	0xad2a00a0	sw $10,0x000000a0($9)	20: sw $t2, 160($t1)

נתונה תמונת הקוד שלהלן, הלקוחה מה- MARS, באסמבלי של ה MIPS-32:

נתונים הערכים ההתחלתיים של האוגרים:

- $s0 = 0x00000050
- $t2 = 0xABCD1234
- כל שאר האוגרים, ערכם שווה לערך מספר האוגר בהקסה ($0 = 0, $1 =1, ... $10 = 0xa,... $20 = 0x140,... $31 = 0x1F0)

ניתן לגשת לכל מרחב הזיכרון בפקודות LW ו- SW (כל עוד הכתובת מתחלקת ב- 4).

ערך 200 **הבתים** (**Bytes**) הראשונים בזיכרון הינו כערך 2 ספרות ההקסה הנמוכות של כתובת הבית

לדוגמה: בית בכתובת 0x00000054 ערכו 0x54

בית בכתובת 0x000000A8 ערכו 0xA8

נתונים אלה תקפים לכל סעיפי שאלה 7

סעיף 7.1: מעקב (4 נקודות):

מה ערך אוגר $t1, **בבסיס הקסה**, לאחר פקודת LW (פקודה שלישית, שורה 18)?

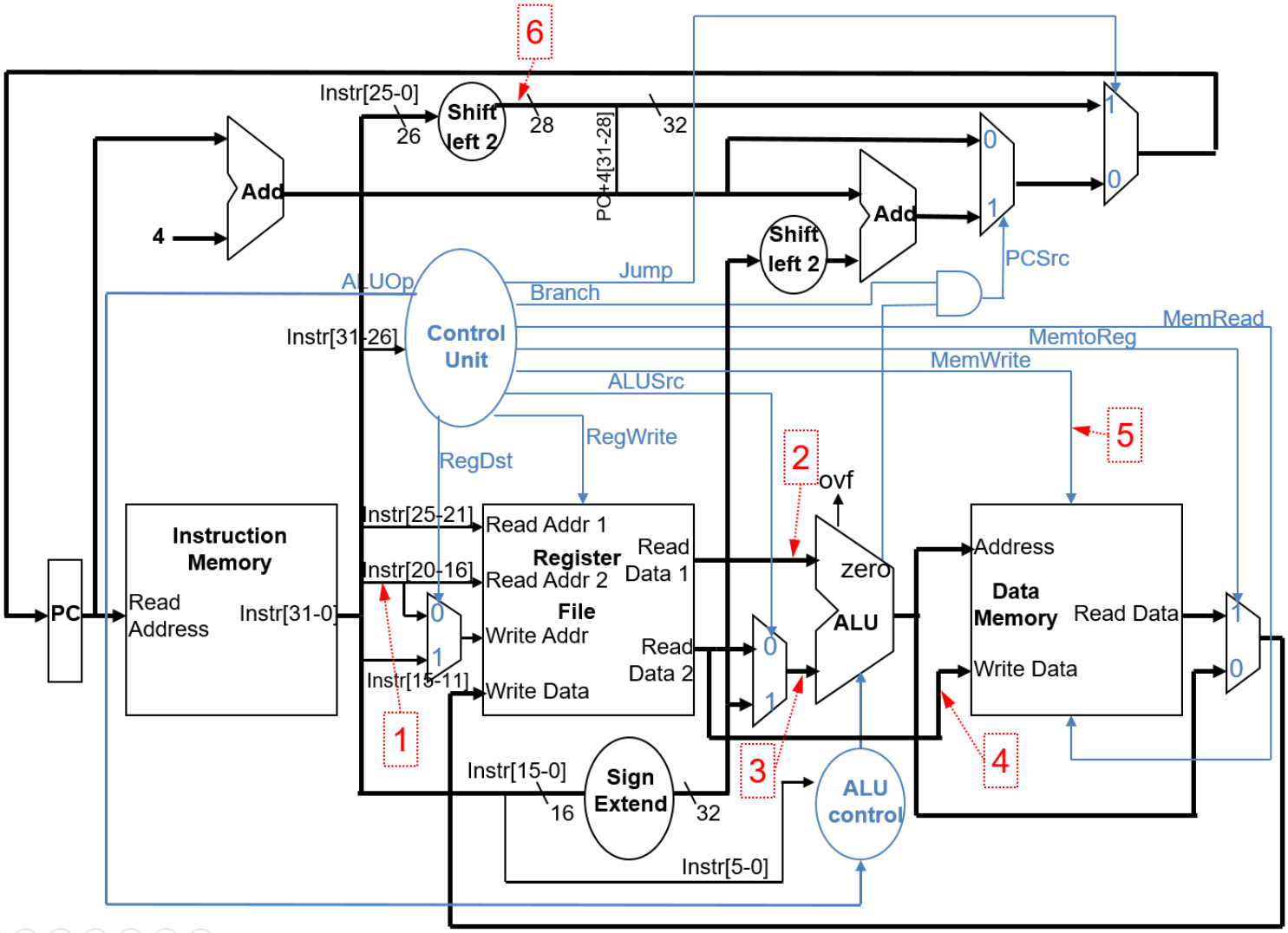
_____

סעיף 7.2: מעבד חד מחזורי (18 נקודות):

בתרשים שלהלן מעבד חד מחזורי, ובו סימונים 1-6 על קווים מסוימים של המעבד. עליכם לחשב את הערכים העוברים על קווים אלה, בהנחה שהמידע נבדק לקראת סוף פעימת השעון בביצוע הפקודה **האחרונה** המתוארת בעמוד הקודם: **sw $t2, 160($t1)** (הפקודה בשורה 20 של הקוד)

את תשובתכם יש למלא בטבלה שבתחתית העמוד – יש למלא את הנתונים בערכים בבסיס הקסה באמצעות הסימון 0x.... (3 נקודות לסימון)

- ערך לא ידוע – יש לסמן ב- X
- ניתן להיעזר בנתונים מטבלאות 4.12 ו- 4.18 בנספח התרשימים של המבחן.



נקודה	ערך (בהקסה)
1	
2	
3	

נקודה	ערך (בהקסה)
4	
5	
6	

סעיף 7.3: טיפול בסיכוני נתונים (5 נקודות):

א. (2 נקודות) רשמו את סיכוני הנתונים הקיימים בקטע הקוד הנתון. כל סיכון נתונים יש לרשום באופן הבא: אוגר X בין פקודה בשורה Y לבין שורה Z

ב. (3 נקודות) כיצד ייפתר סיכון הנתונים הקיים בין הפקודה בשורה 18 לבין הפקודה בשורה 16. התייחסות בתשובתכם לאפשרויות של טיפול על ידי יחידת העברה קדימה (forwarding unit), ו/או Hazzard Detection Unit (HDU) ו/או חציית מקבץ האוגרים.

יש להתייחס למיקום הפקודות (כל פקודה, באיזה שלב של המעבד נמצאת (IF, ID, EX, MEM, WB).

במידה ויש שימוש ביחידת העברה קדימה, יש להתייחס לאופן הזיהוי באופן מפורט גם עם התייחסות למקרי הבדיקה של יחידת ההעברה (מצ"ב), וכן לדגלים היוצאים אל ה MUS לביצוע ההעברה קדימה.

מקרים לבדיקה של יחידת העברה קדימה 1a,1b,2a,2b

1a. EX/MEM.RegisterRd = ID/EX.RegisterRs
1b. EX/MEM.RegisterRd = ID/EX.RegisterRt
2a. MEM/WB.RegisterRd = ID/EX.RegisterRs
2b. MEM/WB.RegisterRd = ID/EX.RegisterRt

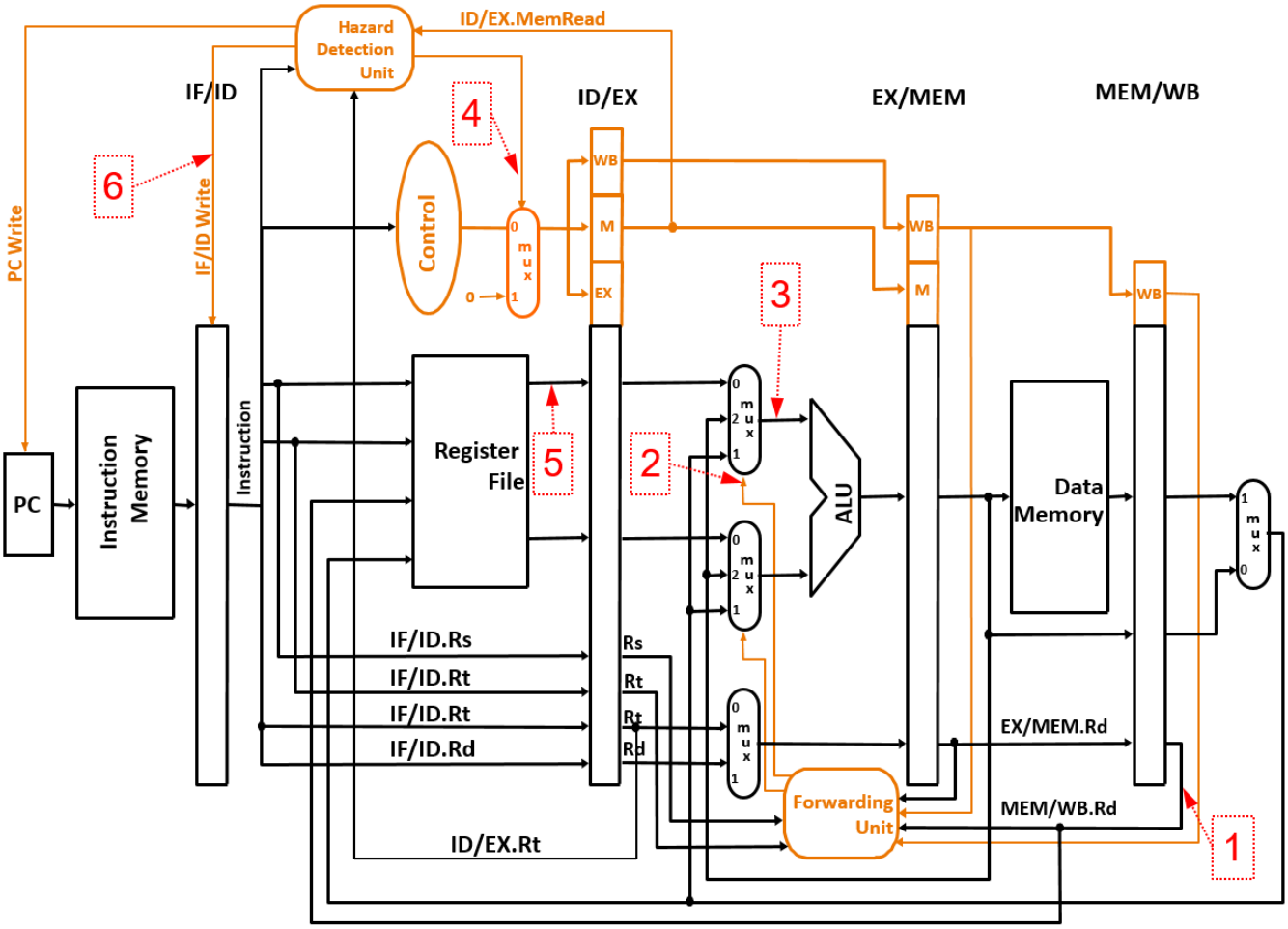
יש להתייחס בפתרון לתרשים 4.60 המופיע בנספח החומרה של המבחן.

סעיף 7.4: מעבד צנרת (18 נקודות):

בתרשים שלפניכם, של מעבד MIPS העובד בטכנולוגיית **צנרת** (המבוסס על תרשים 4.60 בנספח) מתוארת סוף פעימת השעון **השלישית** בביצוע הקוד שבשאלה, בהתאם לנתון בתחילת השאלה.

בתרשים מסומנות 6 נקודות (1-6) על קווים מסוימים של המעבד. יש למלא בטבלה שבתחתית העמוד את הערכים על קווים אלו בהתייחסות להנחות הבאות: סיכוני נתונים בהרצת הקוד נפתרים על ידי יחידת ההעברה קדימה (Forwarding Unit), יחידת איתור סיכונים (HDU) וחציית מקבץ האוגרים.

IF	ID	EX	MEM	WB
sw $t2, 160($t1)	addi $t1, $t1, 4	lw $t1, 16($t0)	beq $t0, $zero, cont	add $t0, $s0, $zero



נקודה	ערך (בהקסה)
1	
2	
3	

נקודה	ערך (בהקסה)
4	
5	
6	

שאלה 8: (25 נקודות)

עליכם לכתוב פרוצדורה בשם **sumRevDiagonalFrom**.

הפרוצדורה מקבלת 4 פרמטרים:

- $a0 – כתובת של מערך דו-ממדי ריבועי. הערכים במערך הינם ערכים עם סימן (signed) וגודל כל איבר במערך הינו **מילה (WORD)**.
- $a1 – מספר האיברים בכל שורה/עמודה (תזכורת: זה מערך דו-ממדי ריבועי)
- $a2 – מספר שורת התחלה (ערך בין 0 למספר השורות פחות אחד)
- $a3 – מספר עמודת התחלה (ערך בין 0 למספר העמודות פחות אחד)

הערכים באוגרים $a2 ו- $a3 הם אינדקס של שורה/עמודה המציינים את נקודת ההתחלה של משימת הפרוצדורה.

הפרוצדורה צריכה לחשב את סכום "**האלכסון ההפוך**" מנקודת ההתחלה (לפי אוגרים $a2 ו- $a3) עד לקצה המערך (מצד שמאל, או מלמטה).

דוגמה להמחשה:

המערך הדו-ממדי משמאל הינו בגודל 7x7

עבור נקודת התחלה [4][1] – "האלכסון ההפוך" מסומן ברקע אפור מנקודת התחלה זו – שמאלה ולמטה – עד לנקודה [0][5]. המשך מעבר לנקודה זו – מהווה גלישה מהמערך לעמודה 1- (שכמובן אינה קיימת...). הסכום במקרה זה הוא: 20

עבור נקודת התחלה [6][4] – "האלכסון ההפוך" מסומן ברקע אפור מנקודת התחלה זו – שמאלה ולמטה – עד לנקודה [4][6]. המשך מעבר לנקודה זו – מהווה גלישה מהמערך לשורה 7 (שכמובן אינה קיימת...). הסכום במקרה זה הוא: 12

	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	2	0	0
2	0	0	0	3	0	0	0
3	0	0	4	0	0	0	0
4	0	5	0	0	0	0	3
5	6	0	0	0	0	4	0
6	0	0	0	0	5	0	0

הנחיות לפתרון:

- יש לכתוב את הפרוצדורה, בלבד, לפי כללי העבודה שלמדנו בכתיבת פרוצדורות.
- ניתן להשתמש בכל הפקודות כפי ש MARS מקבל (כלומר...עובר קומפילציה ב MARS). הקפידו על כך, והקפידו "שלא להמציא" פקודות שאינן קיימות.
- הפתרון חייב להשתמש אך ורק בפרמטרים שהתקבלו, **ללא** כל שימוש/תוספת של משתנים במקטע הנתונים (data.)
- הפתרון צריך להתאים לכל גודל מערך דו ממדי ריבועי (ולא רק למערך של 7x7 שניתן בדוגמה).
- ניתן להניח שהערכים בפרמטרים (אוגרים $a0.... $a3) תקינים – כלומר, **אין צורך** לבצע בדיקות תקינות.
- הפרוצדורה מחזירה ערך שהוא סכום "האכלסון ההפוך" מנקודת ההתחלה שניתנה באוגרים $a2, $a3.

בעמוד הבא נתונה לכם תוכנית ראשית המשתמשת בפרוצדורה שעליכם לכתוב. היעזרו בקוד התוכנית הראשית בכדי להבין כיצד מופעלת הפרוצדורה.

את הפתרון יש לרשום על גבי טופס המבחן, בהמשך, במקום המיועד לכך.

תוכנית ראשית, לדוגמה, המשתמשת בפרוצדורה שעליכם לכתוב:

```
################## Data segment #######################
.data
matrix: .word       0 , 0 , 0 , 0 , 0 , 0 , 0 ,
                    0 , 0 , 1 , 1 , 2 , 0 , 0 ,
                    0 , 1 , 0 , 3 , 0 , 1 , 0 ,
                    0 , 1 , 4 , 0 , 0 , 1 , 0 ,
                    0 , 5 , 0 , 0 , 0 , 1 , 3 ,
                    6 , 0 , 1 , 1 , 1 , 4 , 0 ,
                    0 , 0 , 0 , 0 , 5 , 0 , 0
length:     .byte  7
startRow:   .byte 1
startCol:   .byte 4
################## Code segment #######################
.text
.globl main
main:  # main program entry
        la $a0, matrix
        lbu $a1, length
        lbu $a2, startRow
        lbu $a3, startCol
        jal sumRevDiagonalFrom

        move $a0, $v0
        li $v0, 1
        syscall

        li $a0, '\n'
        li $v0, 11
        syscall                     # new line

        la $a0, matrix
        lbu $a1, length
        li $a2, 4
        li $a3, 6
        jal sumRevDiagonalFrom

        move $a0, $v0
        li $v0, 1
        syscall                     # print substruction of nodes' value

exit:
        li $v0, 10      # Exit program
        syscall
```

פתרון שאלה 8:

################### the solution ###########################