מערכות ספרתיות ומבנה המחשב (044252)

סמסטר אביב תשע"ט

בחינה סופית – מועד א

4 ביולי 2019

פתרון

טור 1

מספר סטודנט

משך המבחן: 3 שעות (180 דקות). **תכננו את זמנכם היטב.**

חומר עזר: אין להשתמש בכל חומר עזר בכתב, מודפס או אלקטרוני, פרט לדפי העזר שיחולקו במהלך הבחינה.

הנחיות והוראות:

- הבחינה כתובה על גבי 18 עמודים כולל עמוד זה (**בדקו בתחילת הבחינה שלא חסרים לכם עמודים**).
- בתחילת הבחינה תקבלו חוברת בחינה, מחברת טיוטה, דפי עזר וטופס תשובות ממוחשב. בסיום הבחינה, החזירו את חוברת הבחינה וטופס התשובות הממוחשב בלבד.
- יש לענות על כל השאלות בגוף המבחן.
- אין **לתלוש או להפריד** דפים מחוברת הבחינה, ממחברות הטיוטה ומדפי העזר.
- רשמו את מספר הסטודנט שלכם על חוברת הבחינה (בראש עמוד זה), על דפי העזר, ועל כל מחברות הטיוטה.
- לא מורדות נקודות (אין "קנס") בגין תשובה שגויה. לכן, כדאי לסמן תשובה כלשהי לכל שאלה.
- ציון השאלות רב הברירה ייקבע על סמך סריקה ממוחשבת של טופס התשובות בלבד. **לא לשכוח לסמן בטופס התשובות הממוחשב את מספר הטור שלכם (מופיע בראש עמוד זה).**
- אסור שימוש בכל חומר חיצוני. אסורה העברת חומר כלשהו בין הנבחנים, ואסורה כל תקשורת עם אנשים אחרים או כל מקור מידע. האיסור חל על כל צורות התקשורת – מילולית, חזותית, כתובה, אלקטרונית, אלחוטית, טלפתית, או אחרת. בפרט, אין להחזיק בטלפון סלולארי.

בהצלחה!

שאלה 1 (5 נקודות)

נתון קוד ה-System Verilog הבא:

```
module test(
   input logic clk,
   input logic [3:0] a,
   output logic [3:0] z
);
   always_comb begin
      z = (a << 2) + 1;
   end
endmodule
```

איזה קטע קוד ייצור חומרה זהה לחומרה שתיווצר כתוצאה מהקוד הנ"ל?

א-

```
module test(
   input logic clk,
   input logic [3:0] a,
   output logic [3:0] z
);
   always_ff @(posedge clk) begin
      z <= (a * 4) + 1;
   end
endmodule
```

ב-

```
module test(
   input logic [3:0] clk,
   input logic [3:0] a,
   output logic [3:0] z
);
   assign z = {a[1:0], {2{1'b1}}};
endmodule
```

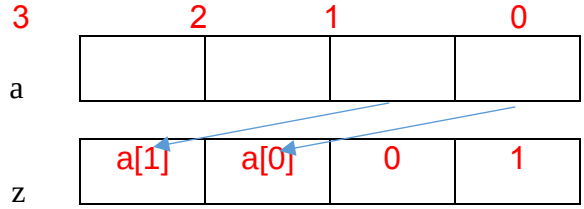
ג-

```
module test(
   input logic clk,
   input logic [3:0] a,
   output logic [3:0] z
);
   assign z[3] = a[1];
   assign z[2] = a[0];
   assign z[1:0] = 2'b01;
endmodule
```

ד- תשובות א' ו-ב' נכונות.

ה- תשובות ב' ו-ג' נכונות.

התשובה הנכונה היא תשובה ג'.
הסבר:
קטע הקוד הנתון מבצע shift left בשני מקומות ומוסיף 1, ואת התוצאה מוציא ל-output.
כלומר:



לכן תשובה ג' נכונה.
נשים לב כי אחד מה-input-ים הוא שעון, אבל מאחר והמימוש הפנימי של ה-module כולל רק לוגיקה צירופית, אין לו משמעות. קטע הקוד בתשובה א' מבצע לוגיקה דומה (כי shift left בשני מקומות שקול להכפלה ב-4), אך מאחר והלוגיקה נמצאת בתוך משפט always_ff, היא מחושבת רק בעליית שעון ולכן יוצרת גם Flip flops, בניגוד ללוגיקה המקורית. בתשובה ב', שני '1' משורשרים אל הביטים התחתונים של a במקום '01', כדרוש.

<u>מה שאלה 1 בודקת:</u>

- ניתוח והבנה של קוד
- תרגום של קוד לחומרה
- ידע של syntax מתקדם כגון pack ו-repeated signals
- ידע של ההבדל בין procedural block סינכרוני ואסינכרוני

שאלה 2 (5 נקודות)

שני סטודנטים להנדסת חשמל מעוניינים לתקשר ביניהם באמצעות מילים השייכות לקוד 2 out of 4 עליו למדו בקורס. בקוד זה, המילים הקיימות הן מילים בעלות 4 ביטים כך שמספר הסיביות במילה שערכן הוא '1' הינו 2 בדיוק.
למשל, המילה: 0101, נמצאת בקוד. שכן, 0101 הינה מילה בעלת 4 ביטים, ושני ביטים הם '1'.

סטודנט א' שולח מילים לסטודנט ב' מתוך הקוד.
סטודנט ב' בודק האם המילים שקיבל אכן שייכות לקוד.
סטודנט ב' יודע שקו התקשורת בין הסטודנטים רועש מעט באופן שבו כל מילה שקיבל עלולה להכיל **עד היפוך סיבית אחת לכל היותר**. לכן, סטודנט ב' מעוניין לממש מערכת צירופית, שמקבלת מילה אחת (4 ביטים) ובודקת האם היא שייכת לקוד או לא. במידה וכן, המערכת תוציא '1', אחרת '0'.

לסטודנט נתונים שערי AND, OR, XOR בעלי **שתי כניסות**, ושערי NOT.

איך ניתן לממש את המערכת הצירופית הנדרשת?

א- ניתן לממש את הפונקציה בעזרת שערי XOR בלבד.
ב- ניתן לממש את הפונקציה בעזרת שערי OR בלבד.
ג- ניתן לממש את הפונקציה בעזרת שערי AND בלבד.
ד- כדי לממש את הפונקציה צריך גם שערי AND וגם- OR, אך אין צורך בשערים נוספים.
ה- תשובות א' – ד' אינן נכונות מכיוון שהשערים הנתונים בהן אינם מהווים מערכת פעולות שלמה.

שאלה זו בוטלה בגלל הניסוח הלא חד-משמעי של התשובה הנכונה

התשובה הנכונה היא ה'
יכולה להיות שגיאה יחידה בקו, לכן לא נוכל לקבל את הקלטים 0000 ו- 1111, שכן נצטרך שתי שגיאות לפחות על מנת לקבל קלטים אלה. למעשה, אנחנו צריכים לבדוק האם למילה מספר זוגי של '1'. זאת משום שמילים בעלות שני '1' נמצאות בקוד ומילים ללא '1' או בעלות ארבעה '1' לא יכולות להתקבל (Don't Care). במפת קרנו זה יראה כך:

yz \ wx	00	01	11	10
00	ϕ		1	
01		1		1
11	1		ϕ	
10		1		1

$$f = \overline{(w \oplus x \oplus y \oplus z)}$$

ולא נוכל לממש את הפונקציה בעזרת אחד מהאופציות א' - ד' מכיוון שלא ניתן לממש שער NOT באף אחת מהן. אם אחת מהאופציות הייתה מערכת פעולות שלמה, אז היינו יכולים לממש בעזרתה את הפונקציה.

שאלה 3 (5 נקודות)

בהמשך לשאלה הקודמת, סטודנט א' שולח לסטודנט ב' מילים על הקו באופן סדרתי.
סטודנט ב' מעוניין לבדוק האם קו השידור בין הסטודנטים אמין. לשם כך הוא משתמש ביציאת המערכת מהסעיף הקודם שמדווחת האם המילה הנוכחית נמצאת בקוד. אם הקלט הינו '0' אז המילה איננה בקוד ואם '1' אז המילה אכן בקוד.
אם **מתוך 3 המילים** האחרונות שהתקבלו, **לפחות 2 לא בקוד**, הקו מוגדר כלא תקין.

סטודנט ב' מעוניין לבנות מכונת מצבים מסוג **MOORE** שמקבלת ככניסה סיבית המסמלת האם המילה הנוכחית תקינה או לא. במידה והקו נמצא כלא תקין, המכונה תוציא '1', ללא תלות בקלט עתידי.

למשל:

Cycle	1	2	3	4	5	6	7	8	9	10	11
IN	1	1	0	1	1	0	1	0	1	0	1
OUT	X	0	0	0	0	0	0	0	**1**	**1**	**1**

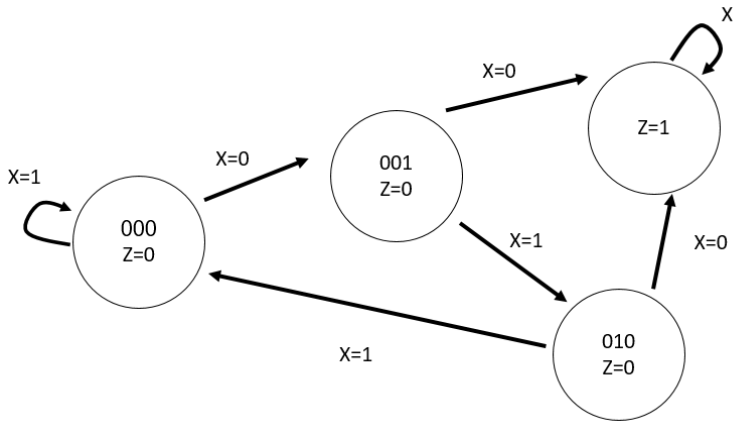
במהלך מחזורים מספר 6 ו- 8 התקבל בכניסה '0'. כלומר במשך שלושה מחזורים קיבלנו שתי מילים שאינן בקוד, ולכן החל מהמחזור הבא, המערכת תוציא את הפלט '1', מבלי להתחשב בכניסות.

יש להניח כי המערכת מתחילה את פעולתה במצב כאילו קיבלה כמות גדולה של מילים חוקיות.

כמה מצבים יהיו במכונת **המצבים המצומצמת ביותר** המממשת את המערכת הנדרשת?

א- 4 מצבים או פחות.
ב- 5 מצבים.
ג- 6 מצבים.
ד- 7-8 מצבים.
ה- 9 ומעלה

התשובה הינה א' – 4 מצבים. מכונת המצבים המצומצמת ביותר הינה:



שימו לב שמהמצב של הסדרה 010, ניתן לעבור ישירות למצב 000 בהינתן קלט 0, ולא נצטרך מצב נוסף ביניהם.

שאלה 4 (5 נקודות)

יהי X מספר בינארי ברוחב $N > 0$ ביטים, ויהי ψ_X קידוד גריי שלו.
נתונות הטענות הבאות:

1) $X \oplus \psi_X \neq 0$, לכל X
2) $X \leq Y$ **אמ"ם** $\psi_X \leq \psi_Y$
3) $X \oplus \psi_X = Y \oplus \psi_Y$ **אמ"ם** $X = Y$

הערות:

- $\psi_X \leq \psi_Y$ משמעותו שהערך המיוצג בבסיס בינארי (ללא סימן) כ- ψ_X קטן או שווה לערך המיוצג בבסיס בינארי (ללא סימן) כ- ψ_Y.
  למשל, עבור $\psi_Y = 1000$, $\psi_X = 0111$ מתקיים ש- $\psi_X < \psi_Y$ מכיוון ש- $7 \leq 8$.
  כנ"ל לגבי $X \leq Y$.
- $Z = X \oplus Y$ מוגדרת להיות פעולת bitwise XOR בין כל שני ביטים מ-X ומ-Y באופן הבא:

$$Z_i = X_i \oplus Y_i\,, \quad \forall i: 0 \leq i < N$$

מבין התשובות הבאות, בחרו את התשובה הנכונה:

א- **רק** טענה 1 נכונה
ב- **רק** טענה 2 נכונה
ג- **רק** טענה 3 נכונה
ד- מבין הטענות 1-3 יש **רק** 2 טענות נכונות
ה- **כל** הטענות 1-3 **לא** נכונות

התשובה הנכונה היא ה'

טענה 1 לא נכונה:
למשל עבור $N = 2$, דוגמא נגדית:

$X = 00 \rightarrow \psi_X = 00 \qquad X \oplus \psi_X = 00 \oplus 00 = 00$
$X = 01 \rightarrow \psi_X = 01 \qquad X \oplus \psi_X = 01 \oplus 01 = 00$

טענה 2 לא נכונה:
למשל עבור $N = 2$, דוגמא נגדית:

$X = 10 \rightarrow \psi_X = 11$
$Y = 11 \rightarrow \psi_X = 10$

כלומר $X \leq Y$ אבל $\psi_X > \psi_Y$.

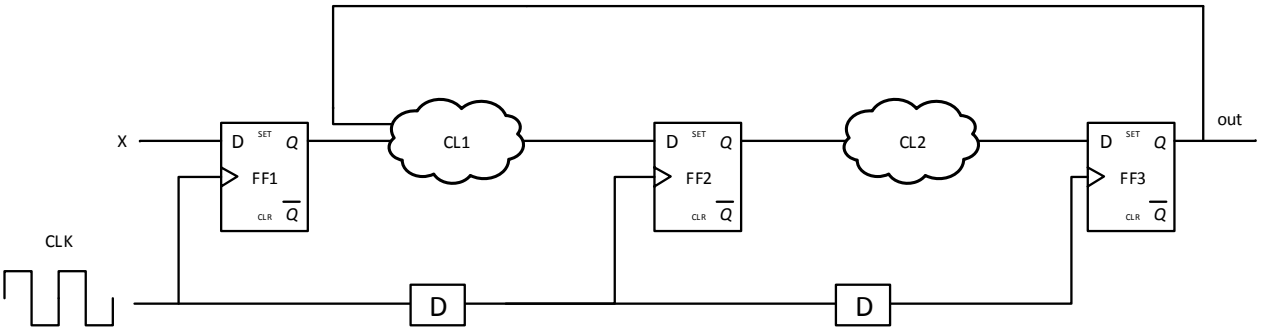
טענה 3 לא נכונה: (הכיוון משמאל לימין נכון, אבל מימין לשמאל לא נכון)
למשל עבור $N = 2$, דוגמא נגדית, הדוגמה מטענה 1:

$X = 00 \rightarrow \psi_X = 00 \qquad X \oplus \psi_X = 00 \oplus 00 = 00$
$Y = 01 \rightarrow \psi_Y = 01 \qquad Y \oplus \psi_Y = 01 \oplus 01 = 00$

כלומר עבור $X = 00 \neq 01 = Y$ קיבלנו ש- $X \oplus \psi_X = 0 = Y \oplus \psi_Y$.

שאלה 5 (5 נקודות)

נתון המעגל הבא:



נתון כי הדלגלגים דוגמים בעליית שעון. בנוסף, נתון כי כניסת המערכת x מתוזמנת בהתאם לתנאי ה- *hold* וה-*setup* של הדלגלג *FF1*.

להלן זמני ההשהיה:

$t_{pd}(CL1) = 5\ ns$
$t_{pd}(CL2) = 6\ ns$
$t(D) = 2\ ns$
$t_{pCQ}(FF) = 7\ ns$
$t_{setup}(FF) = 3\ ns$

בשאלה זו ניתן להניח שתנאי *hold* מתקיים (לא צריך לבדוק אותו).

מבין התשובות הבאות, מהו אורך מחזור השעון **המינימלי** שיאפשר למעגל לפעול בצורה תקינה?

א- $14\ ns$
ב- $15\ ns$
ג- $16\ ns$
ד- $17\ ns$
ה- גדול מ- $17ns$

התשובה הנכונה היא ד'

$FF1 \rightarrow FF2$:
$t_{pd}(FF1) + t_{pd}(CL1) + t_{su}(FF2) \leq T_{clk} + t(D) \rightarrow T_{clk} \geq 13\ ns$

$FF2 \rightarrow FF3$:
$t_{pd}(FF2) + t_{pd}(CL2) + t_{su}(FF3) \leq T_{clk} + t(D) \rightarrow T_{clk} \geq 14\ ns$

$FF2 \rightarrow FF3$:
$t_{pd}(FF3) + t_{pd}(CL1) + t_{su}(FF2) \leq T_{clk} + t(D) - 2 \cdot t(D) \rightarrow T_{clk} \geq 17\ ns$

$T_{clk,min} = \max(13, 14, 17) = 17\ ns$

שאלה 6 (5 נקודות)

מעוניינים לממש מערכת צירופית המכפילה מספר בינארי X בן n סיביות פי 5, אך ממומשת בעזרת רכיבי FullAdder וקבועים ('0' ו-'1') **בלבד**.

הערות:
1- המספר X מיוצג בשיטת ה- unsigned.
2- שימו לב שייתכן שצריך יותר מ-n ביטים כדי לייצג את התוצאה.

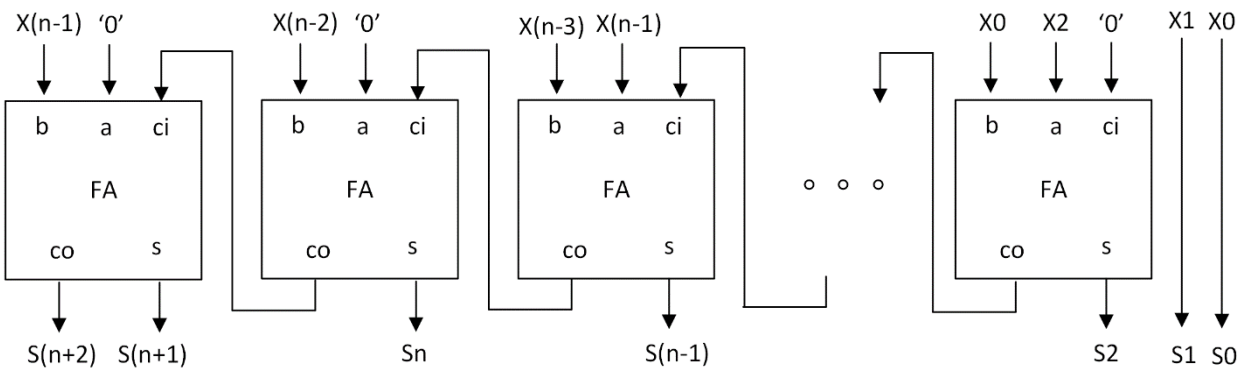
מבין התשובות הבאות, מהי הכמות **המינימלית** של רכיבי FullAdder בה אפשר לממש את המערכת?

א- $2n$
ב- $4n$
ג- n^2
ד- $5n$
ה- n

התשובה הנכונה היא תשובה ה'
נסמן את המספר בן n הסיביות ב-$x = x_{n-1} \dots x_0$. נשים לב ש- $5x = 4x + x$, ו-$4x =$ $x_{n-1} \dots x_0 00$, ולכן

$$
\begin{array}{cccccccccc}
\times & & & & & & & & & 5 \\
 & & & & x_{n-1} & \dots & x_2 & x_1 & x_0 & \\
\hline
+ & & 0 & 0 & x_{n-1} & \dots & x_2 & x_1 & x_0 & \\
 & & x_{n-1} & x_{n-2} & x_{n-3} & \dots & x_0 & 0 & 0 & \\
\hline
 & s_{n+2} & s_{n+1} & s_n & s_{n-1} & \dots & s_2 & s_1 & s_0 &
\end{array}
$$

ולכן, המעגל שיקיים זאת הוא



זה מעגל המשתמש ב-n רכיבי FA. זו האפשרות הנמוכה ביותר מבין התשובות, ולכן היא המינימאלית מביניהם.

שאלה 7 (5 נקודות)

נתון קטע הקוד הבא, המתחיל לרוץ מ-main (כתובת 0x0001 0000).

```
[0x0001 0000] main:   addi sp, sp, -4
[0x0001 0004]         addi a0, x0, 3
[0x0001 0008]         addi a1, x0, 2
[0x0001 000c]         sw ra, 0(sp)
[0x0001 0010]         jal ra, func
[0x0001 0014]         lw ra, 0(sp)   <=
[0x0001 0018]         addi sp, sp, 4
[0x0001 001c]         jalr x0, ra

[0x0001 0100] func:   addi sp, sp, -4
[0x0001 0104]         beq a1, x0, done
[0x0001 0108]         addi a1, a1, -1
[0x0001 010c]         sw ra, 0(sp)
[0x0001 0110]         jal ra, func
[0x0001 0114]         add a0, a0, a0
[0x0001 0118]         lw ra, 0(sp)
[0x0001 011c] done:   addi sp, sp, 4
[0x0001 0120]         jalr x0, ra
```

מה יהיה הערך ברגיסטר a0 מיד לפני שמריצים את הפקודה בכתובת [0x0001 0014]
(הפקודה המסומנת בחץ)?
בחרו את התשובה המתאימה:

א- התוכנית לעולם לא תגיע לפקודה בכתובת [0x0001 0014]
ב- 6
ג- 9
ד- 12
ה- 15

<u>פתרון:</u> התשובה היא ד'.
הפונקציה func מבצעת רקורסיה בעומק a1+1 ובכל איטרציה שלה היא מבצעת $a0 = a0 + a0 = 2 \cdot a0$, פרט לאיטרציה האחרונה (a1=0) בה הפונקציה לא מבצעת כלום.
באינדוקציה נקבל כי התוצאה ש-func מחזירה היא $a0 \cdot 2^{a1}$ (כאשר הערכים של a0 ו-a1 הם אלו שהפונקציה נקרא איתם בתחילת הרקורסיה), במקרה של השאלה ערך זה יוצא $3 \cdot 2^2 = 3 \cdot 4 = 12$.
הפונקציה והקוד כתובים ללא טעויות ולכן הקוד יגיע לפקודה בכתובת [0x0001 0014] מייד עם חזרת הפונקציה func, ולכן תוצאת func תהיה ב-a0, כלומר a0=12.

שאלה 8 (5 נקודות)

בפרוטוקול תקשורת חדש שנקרא XUART, מועברים X סיביות מידע בכל שידור, כאשר X הינו מספר ידוע וקבוע (אינו משתנה משידור לשידור).
שאר המאפיינים של פרוטוקול זה זהים ל- UART עליו למדנו.

נתונים:

$T_{Cycle}(Tx) = 10nsec$
$T_{bit} = 50nsec$
$T_{cycle}(Rx) = 13nsec$

בנוסף נתון שהמקלט מזהה את ה- Start Bit באופן מידי.

יש להניח שלאחר שהמקלט קבע מהו ה- N_R שאיתו יעבוד, הוא קבוע עד לסוף קליטת השידור ולא ניתן לשינוי.

מבין התשובות הבאות, מה המספר **המקסימלי** של **סביות מידע** שניתן לשדר ולקלוט בלי שגיאה?
הערה: סביות Start-bit ו- Stop-bit אינן סביות מידע.

א- 8
ב- 9
ג- 10
ד- 13
ה- 14

פתרון:

$$N_{Tx} = \frac{T_{bit}}{T_{Cycle}(Tx)} = 5$$

$$N_{Rx} = \left\lceil \frac{T_{bit}}{T_{Cycle}(Tx)} \right\rceil = \left\lceil \frac{50}{13} \right\rceil \rightarrow N_{Rx} = 4$$

כלומר, המקלט ימתין 4 מחזורי שעון, בין דגימה לדגימה.
בתחילת השידור, המקלט ירצה להמתין $1.5T_{bit}$, כלומר 6 מחזורי שעון.
המקלט ידגום הביט הראשון לאחר $T_{Cycle}(Rx) \cdot 6 = 78ns$ בעוד שהוא אמור לקלוט את הביט הראשון באמצע השידור שלו, כלומר לאחר $1.5T_{bit} = 75ns$, ונוצרה הסטה של $3ns$.
מכאן, אנחנו יודעים שהמקלט ממתין 4 מחזורי שעון בין כל דגימה לדגימה, כלומר $T_{cycle}(Rx) \cdot 4 = 52ns$. וההסטה רק תגדל ב- $2ns$ בין דגימה לדגימה.
אנחנו יודעים ש- $T_{bit} = 50ns$ ולכן, כשההסטה תגיע ל- $25ns$, המקלט ידגום ביט שגוי.
למעשה פיתחנו נוסחה להיסט כפונקציה של מספר סביות המידע: $3ns + 2ns \cdot x$.
נדרוש שהיא תהיה קטנה מ- $25ns$:

$$3ns + 2ns \cdot x < 25ns$$
$$x < 11$$

כאשר x הינו מספר סביות המידע שניתן לדגום ולכן בסה"כ נקבל שניתן לדגום $x = 10$ סביות מידע.
שימו לב בשאלה הזו, צריך שגם ה- Stop-bit ידגם כראוי, ושדגימה שנמצאת בדיוק "על התפר" שבין שידורי ביטים אינה חוקית. אם לא היינו מבצעים את ההנחות האלה, היינו מקבלים $x < 13$, כך שעדיין $x = 10$ היא התשובה היחידה שנכונה.

שאלות 9 (5 נקודות)

נתונה טבלת המעברים הבאה:

PS	NS(x=0), z	NS(x=1), z
A	A,0	C,0
B	E,1	A,0
C	B,0	E,1
D	B,1	C,0
E	F,1	D,0
F	D,0	F,1

מבין הסדרות הבאות, לאיזו סדרת קלט היציאה תהיה '1' **במהלך המחזור החמישי (האחרון)**, כאשר נתון שמתחילים ממצב A והסיביות נקלטות משמאל לימין בכל אחת מהסדרות?

א-

cycle	1	2	3	4	5
x	1	0	1	0	0

ב-

cycle	1	2	3	4	5
x	0	1	1	1	1

ג-

cycle	1	2	3	4	5
x	1	1	0	1	0

ד-

cycle	1	2	3	4	5
x	1	1	1	1	0

ה-

cycle	1	2	3	4	5
x	0	0	1	0	0

תשובה הנכונה היא ה'
כאשר תמיד מתחילים ממצב A והסיביות נקלטות משמאל לימין כנתון:

א. 10100:

NS	C	B	A	A	A
x	1	0	1	0	0
z	0	0	0	0	0

ב. 01111:

NS	A	C	E	D	C
x	0	1	1	1	1
z	0	0	1	0	0

ג. 11010:

NS	C	E	F	F	D
x	1	1	0	1	0
z	0	1	1	1	0

ד. 11110:

NS	C	E	D	C	B
x	1	1	1	1	0
z	0	1	0	0	0

ה. 00100:

NS	A	A	C	B	E
x	0	0	1	0	0
z	0	0	0	0	1

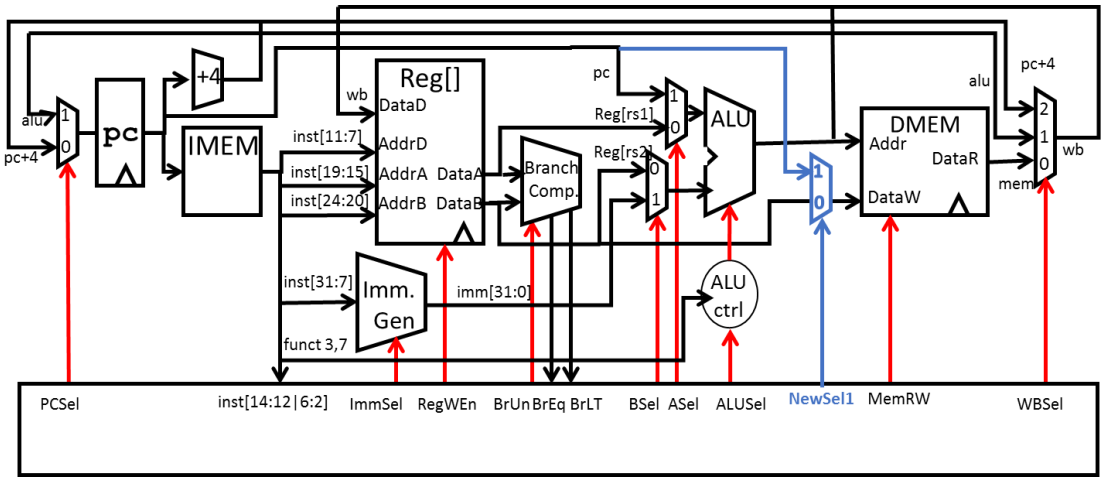
שאלה 100 (8 נקודות)

במעבד מסוג Single Cycle RISC-V, רוצים להוסיף תמיכה בפקודה החדשה המיוצגת בפורמט S-type:

sPC rs1, rs2

הפקודה sPC (store Program Counter), שומרת את ערך ה- PC בזיכרון בכתובת Reg[rs1]+Reg[rs2]. כלומר ערך PC יכתב ל- Mem[Reg[rs1]+Reg[rs2]].
אין לשנות את הזיכרון או את ה- Register File, אך ניתן להוסיף בוררים.

בצעו את השינויים הנדרשים במסלול הנתונים של המעבד כך שיתמוך בפקודה בשרטוט הבא (ציירו על השרטוט):



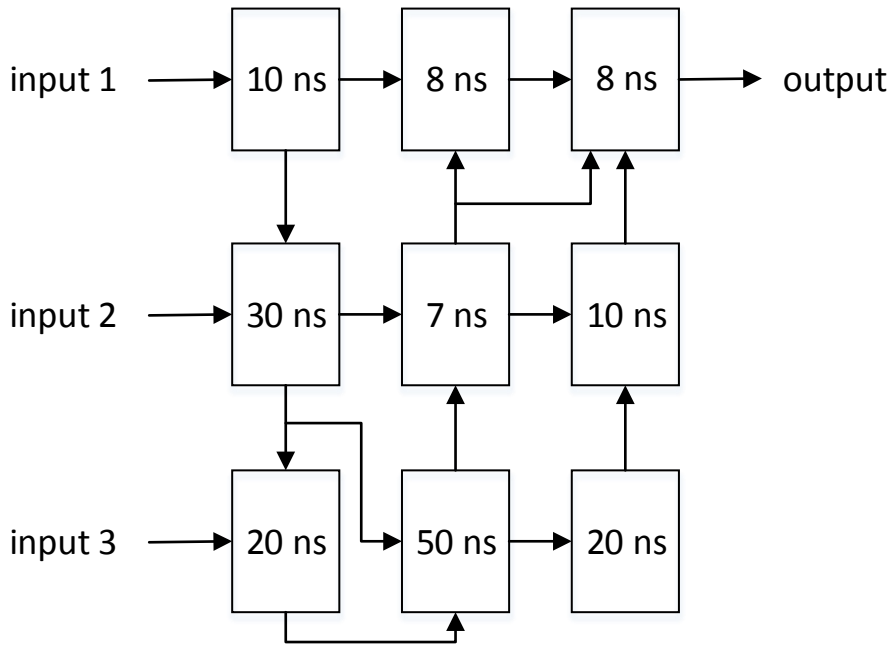
כתבו מהם קווי הבקרה לביצוע הפקודה (אם הוספתם בורר/ים, הגדירו גם מהם קווי הבקרה שלו/שלהם ב- NewSel1 ו/או NewSel2 **וסמנו** אותם בשרטוט):

ImmSel =	S	BSel =	0	WBSel =	Ø
RegWEn =	0	ALUSel =	S	NewSel1 =	1
ASel =	0	MemRW =	W	NewSel2 =	Ø
PCSel =	0				

קיבלנו גם תשובות שבהן ALUSel = add

שאלה 101 (7 נקודות)

נתונה המערכת הצירופית הבאה



המספרים בתוך המלבנים מסמנים את זמני ההשהיה של הרכיבים השונים במערכת.

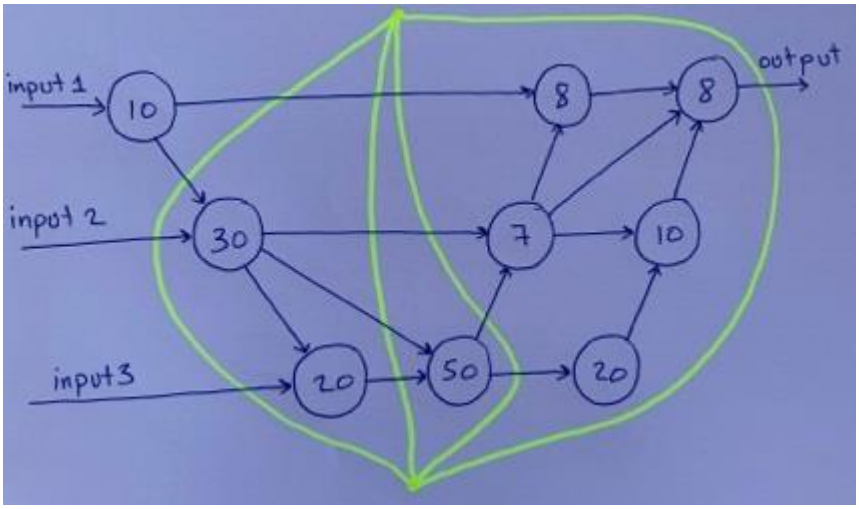
להלן זמני ההשהיה של הרגיסטרים:

$t_{pC-Q}(FF) = 4\ ns$
$t_{cC-Q}(FF) = 1\ ns$
$t_{setup}(FF) = 2\ ns$
$t_{hold}(FF) = 1\ ns$

צנרו את המעגל לקבלת Throughput מקסימלי. לשם כך נדרש להשתמש במספר רגיסטרים **מינימלי** אפשרי. מהו מספר הרגיסטרים, ה-Latency וה- Throughput של המעגל המצונר?

מספר רגיסטרים	13
זמן ההשהיה (Latency)	224 ns
ספיקה (Throughput)	$\frac{1}{56\ ns}$

פתרון
נצייר את המעגל כך שכל החצים יהיו באותו כיוון ונשתמש בשיטת הלולאות כדי לצנר אותו



מספר הרגיסטרים המינימלי: 13
(עבור הלולאה הכי שמאלית, יש שתי אפשרויות להעביר את הקו, בכניסות של ה-30 כמו בציור או ביציאות שלו, אם בוחרים להעביר את הקו ביציאות של ה-30 נקבל מספר רגיסטרים גדול יותר = 14 ואבל זה לא המספר המינימלי).

$$T_{min} = t_{pd}(FF) + t_{pd,\max}(CL) + t_{su}(FF) = 4 + 50 + 2 = 56$$

$$Throughput = \frac{1}{T_{min}} = \frac{1}{56\,ns}$$

$$K = 4$$

$$Latency = K \cdot T_{min} = 4 \cdot 56 = 224\,ns$$

טעויות נפוצות:

- צינור לא נכון של המעגל (הועברו גומיות איפה שלא צריך).
- הרגיסטר במוצא המעגל לא הוסף.
- ספירת מספר הרגיסטרים כמספר הגומיות (ולא כמספר החיתוכים של הגומיות עם חיצים).
- חישוב latency כמחזור שעון כפול מספר הרגיסטרים (ולא כמספר השלבים ב-pipeline, שהוא מספר הגומיות).
- חישוב latency ככפולה של 1 חלקי זמן מחזור.
- חישוב latency כזמן מחזור יחיד.
- חישוב זמן מחזור לא נכון לפי תנאי setup או התעלמות מתנאי setup לחלוטין.

שאלה 102 (8 נקודות)

לאחר ייצור מעבד *Single Cycle RISCV* כפי שנלמד בכיתה. התגלתה תקלה בסיגנל הזיכרון *MemRW* מסוג *stuck at '1'*. כלומר *MemRW = Write* **כל הזמן** ולא רק עבור פקודות כתיבה לזיכרון.
שימו לב שלמעבד יש סיגנל בקרה אחד לזיכרון, אם הערך שלו הוא 0 מתבצעת קריאה מתוך הזיכרון, אחרת מתבצעת כתיבה.
המצב ההתחלתי של הרגיסטרים הוא $RegFile[reg\_addr] = reg\_addr$ והמצב ההתחלתי של הזיכרון הוא $Mem[mem\_addr] = mem\_addr$. כאשר *reg_addr* הוא מספר הרגיסטר, ו- *mem_addr* הוא כתובת תחילת המילה בזיכרון.

לדוגמא:
התוכן של הרגיסטר $x2$ הוא $0x2$, התוכן של הרגיסטר $x31$ הוא $0x31$ וכו'.
התוכן של הזיכרון:

mem_addr	$value$
$0x0$	$0x0$
$0x4$	$0x4$
...	...
$0x2000$	$0x2000$
$0x2004$	$0x2004$
...	...

מעוניינים להריץ את הפקודה הבאה על המעבד

$addi\ x5, x8, 0x104$

לאחר ביצוע הפקודה לעיל, האם יהיה שינוי במצב הרגיסטרים ו/או מצב הזיכרון? אם כן, כיתבו את **כל** השינויים, אחרת כיתבו בטבלה "אין שינוי".

הערה: הכוונה בשינוי במצב הרגיסטרים/זיכרון היא שהתוכן של אחד או יותר מהרגיסטרים/כתובות הזיכרון השתנו לאחר ביצוע הפקודה. בתשובה שלכם ציינו את כל הרגיסטרים/כתובות שהשתנו, את הערך הישן שלהם (לפני ביצוע הפקודה) ואת הערך החדש שלהם (לאחר סיום ביצוע הפקודה).
אין צורך לרשום את הרגיסטרים/כתובות שהערך שלהם לא השתנה.

רגיסטרים

$Register\ number$	$Old\ value$	$New\ value$
$x5$	$0x5$	$0x10C$

זיכרון

$Memory\ address$	$Old\ value$	$New\ value$
$0x10C$	$0x10C$	$0x4$

פתרון:

רגיסטרים:
הפקודה מבצעת פעולת חיבור בין תוכן הרגיסטר *s0* וערך ה-*imm* *0x104* ומעדכנת את הרגיסטר *t0* להכיל את תוצאת החיבור.
לפי טבלת הרגיסטרים בדפי העזר, הרגיסטר *s0* הוא רגיסטר מספר *x8* ולכן לפי הנתון בשאלה, התוכן שלו הוא גם *0x8* כלומר $0x8 + 0x104 = 0x10C$. את התוצאה צריך לכתוב לתוך הרגיסטר *t0* שהמספר שלו הוא *x5*
רגיסטרים:

$Register\ number$	$Old\ value$	$New\ value$
$x5$	$0x5$	$0x10C$

זיכרון:
בפקודת *addi* אין גישה לזיכרון אבל מאחר ויש תקלה במעבד, תתבצע כתיבה לכתובת שה-*ALU* חישב (תוצאת החיבור), כלומר לכתובת *0x10C* ייכתב תוכן הרגיסטר *sr2*.
בפקודת *Itype* אין *src2* אבל המעבד מבצע קריאה ספקולטיבית בשלב ה-*decode* ולוקח את מספר הרגיסטר מתוך 5 הביטים התחתונים של ה-*imm* (לפי הפורמט של הפקודות):

	31 27	26 25	24 20	19 15	14 12	11 7	6 0
R	funct7		rs2	rs1	funct3	rd	Opcode
I	imm[11:0]			rs1	funct3	rd	Opcode

כלומר:
$0x104 = 0b\ 0001\ 0000\ 0100$
ולכן הערך של ה-src השני שהמעבד יקרא הוא התוכן של הרגיסטר מספר x4, לפי הנתון התוכן של הרגיסטר הוא 0x4.
ולכן לתוך הכתובת 0x10C ייכתב הערך 0x4
זיכרון:

$memory\ address$	$Old\ value$	$New\ value$
$0x10C$	$0x10C$	$0x4$

טעויות נפוצות:
- $0x104 + 0x8 = 0x112$
- לא ניתן לדעת מה הערך שייכתב לתוך הזיכרון

שאלה 103 (7 נקודות)

נתונים מעבדי *Single Cycle RISCV* ו-*Multi Cycle RISCV* כפי שנלמדו בכתה.

מעבד ה-single-cycle פועל עם מחזור שעון T_{SC}.
מעבד ה-multi-cycle פועל עם מחזור שעון T_{MC}.

מה התחום שבו צריך להיות היחס $\frac{T_{SC}}{T_{MC}}$ כדי שמעבד ה-multi-cycle יריץ מהר יותר ממעבד ה-single-cycle את התוכנית הבאה:

```
        addi   x1, x0, 0xA
loop:   lw     x2, 0(x1)
        sw     x2, 4(x1)
        subi   x1, x1, 0x1
        beq    x1, x0, loop
exit:
```

רשמו את תשובתכם במרובעים מטה, כך שבמרובע הימיני (הגדול) מופיע **רק** מספר ובריבוע השמאלי (הקטן) מופיע **רק** סימן יחס (סימן היחס יכול להיות רק אחד מהסימנים הבאים: $<, >, \leq, \geq, =, \neq$).
רשמו את המספר וסימן היחס כך שהביטוי המתקבל עם השבר $\frac{T_{SC}}{T_{MC}}$ יהווה את התשובה לשאלה.

$$\frac{T_{SC}}{T_{MC}} \quad \square \quad \square$$

פתרון
ניתן לחלק את הקוד לשני חלקים: לפקודות בתוך הלולאה ולפקודות מחוץ לולאה.
הפקודות מחוץ לולאה הן רק addi, הלוקחת מחזור שעון יחיד במעבד single cycle ו-4 מחזורי שעון ב-multi cycle.
הפקודות בתוך הלולאה הן חזרה אחת של lw,sw,subi ו-beq. במעבד single cycle, איטרציה יחידה של הלולאה תרוץ במשך 4 מחזורי עשון. במעבד multi cycle, מספר מחזורי השעון שדרוש לפקודת lw הוא 5, לפקודת sw הוא 4, לפקודת subi הוא 4 ולפקודת beq הוא 3, סך-הכל 5+4+4+3=16 מחזורי שעון לאיטרציה אחת של הלולאה.
נסמן את מספר מחזורי השעון של הלולאה כ-X, אז זמן ריצת מעבד ה-single cycle הוא $T_{SC} \cdot (1 + 4X)$, וזמן ריצת מעבד ה-multi cycle הוא $T_{MC} \cdot (4 + 16X)$.
כדי שמעבד ה-multi cycle ירוץ מהר יותר צריך להתקיים

$$T_{SC} \cdot (1 + 4X) > T_{MC} \cdot (4 + 16X)$$

$$\frac{T_{SC}}{T_{MC}} > \frac{4 + 16X}{1 + 4X} = \frac{4 \cdot (1 + 4X)}{1 + 4X} = 4$$

נשים לב שמספר האיטרציות שמבצעת הלולאה איננו משנה את התשובה. לכן, למרות שהפקודה האחרונה בלולאה היא beq ומתבצעת איטרציה יחידה, לעומת 10 איטרציות אם הייתה פקודת bne, בשני המקרים נקבל את אותה תשובה.

ניקוד מלא ניתן גם לתשובות שרשמו $\geq$ במקום רק $>$ ולתשובות שרשמו שבר לא מצומצם שערכו שווה ל-4 (למשל $\frac{20}{5}$ או $\frac{164}{41}$).

<u>טעויות נפוצות:</u>

- ספירת מחזורי השעון של sw ב-multi cycle כ-3 או 5 (במקום 4).
- ספירת מחזורי השעון של פקודת addi ב-Multi cycle כ-1 (במקום 4).
- ספירת מחזורי השעון של כל אחת מהפקודות ב-single cycle כ-5 (במקום 1).
- ביצוע איטרציה יחידה של הלולאה ב-single cycle בזמן מחזור שעון יחיד.
- לקיחת מספר שונה של איטרציות עבור הלולאה בין המעבדים.
- בנייה לא נכונה של האי-שוויון הנותנת את היחסים $\leq$ או $>$, או את המספר $\frac{1}{4}$.
- טעויות חישוב.

שאלה 104 (8 נקודות)

מהנדס משתמש במעבד מסוג MultiCycle RISCV. זמן המחזור של המעבד הינו 100ns ובסה"כ ירוצו Nפקודות על המעבד. מתוך כל הפקודות שירוצו, אחוז הפקודות מכל סוג מפורט בטבלה הבאה:

סוג הפקודה	אחוז מכלל הפקודות
Load	25%
Store	15%
R-type	35%
Branch	15%
Jump	10%

המהנדס שוקל 3 סוגי שיפורים למעבד. השיפורים יקרים ולכן יוכל לבחור רק אחד.

1) ייעול הקוד והפחתת מספר הפקודות הכולל ל- 0.75N. ייעול זה אינו משנה את אחוז ביצוע סוגי הפקודות.
2) פיצול חלק ה- Execute שהינו צוואר הבקבוק ב- Datapath של מעבד זה. הפיצול יגרום לשיפור קצב השעון והפחתת זמן המחזור מ- $100ns$ ל- $\boldsymbol{80ns}$.
ייעול זה יגרור הגדלת שלב ה- Execute ממחזור אחד לשניים <u>עבור כל הפקודות</u>.
3) שיפור פקודת Load, והפחתת מספר המחזורים לביצוע הפקודה מ- 5 ל-4, ללא שינוי זמן המחזור.

א- מה יהיה זמן הריצה הכולל כתלות ב- N, על המעבד **<u>ללא שיפורים כלל</u>**?

Total runtime =

ב- מה יהיה זמן הריצה הכולל כתלות ב- N, על המעבד **<u>עם שיפור 1</u>**?

Total runtime =

ג- מה יהיה זמן הריצה הכולל כתלות ב- N, על המעבד **<u>עם שיפור 2</u>**?

Total runtime =

ד- מה יהיה זמן הריצה הכולל כתלות ב- N, על המעבד **<u>עם שיפור 3</u>**?

Total runtime =

פתרון

סה"כ זמן ממוצע לביצוע פקודה כלשהי הינו:

$$T_{eff} = 100[5 \cdot 0.25 + 4 \cdot 0.5 + 3 \cdot 0.25] = 100[1.25 + 2 + 0.75] = 400ns$$

מכאן, שזמן הריצה הכולל הינו (ללא שיפורים כלל):

$$T_{total} = 400N[ns]$$

<u>אופציה א':</u>

$$T_{total} = 0.75N \cdot 400ns = 300N[ns]$$

<u>אופציה ב':</u>

$$T_{eff} = 80 \cdot [6 \cdot 0.25 + 5 \cdot 0.5 + 4 \cdot 0.25] = 80[1.5 + 2.5 + 1] = 5 \cdot 80 = 400[ns]$$

$$T_{total} = 400N[ns]$$

<u>אופציה ג':</u>

$$T_{eff} = 100[4 \cdot 0.25 + 4 \cdot 0.5 + 3 \cdot 0.25] = 100 \cdot [1 + 2 + 0.75] = 375ns$$

$$T_{total} = 375N[ns]$$

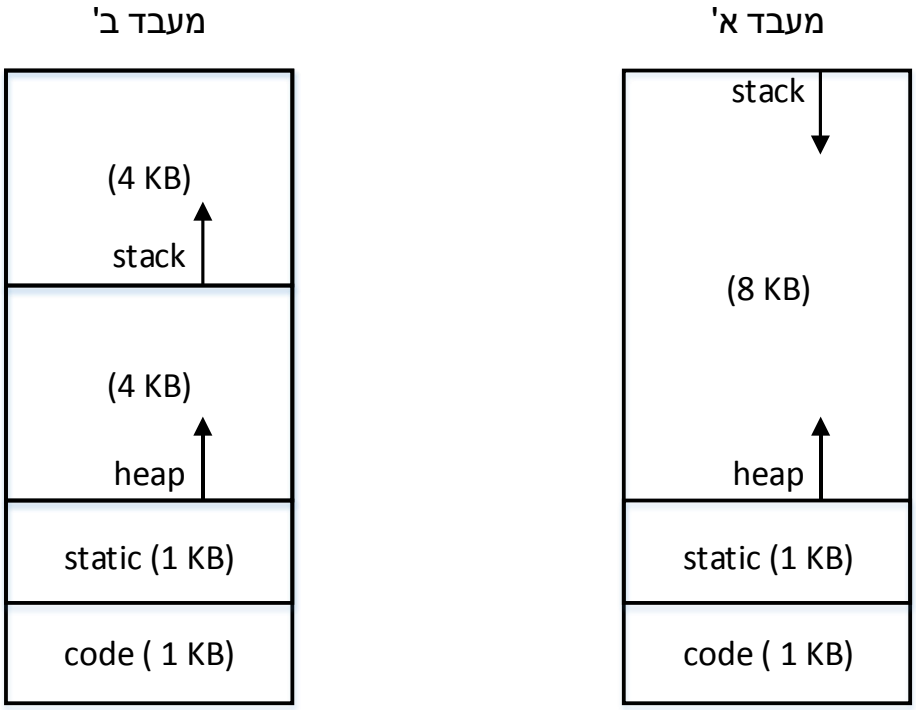
שאלה 105 (10 נקודות)

בהינתן הקוד הבא הכתוב בשפת C (שימו לב שהקוד דומה לקוד שראיתם בתרגול עם שינוי בפונקציה main):

```
int fact (unsigned int n) {
    if (n == 0)
        return 1;
    return n * fact(n-1);
}

int main() {
    int x = 1;
    return x + fact(x);
}
```

מעוניינים להריץ את הקוד על שני מעבדים מסוג RISCV. שני המעבדים זהים לחלוטין חוץ מחלוקת הזיכרון (Memory Layout), כפי שמתואר בציור:



הערה: $(x\ KB)$ מסמן את גודל כל segment ב-KB כאשר $1\ KB = 2^{10}\ Bytes$.

נגדיר את החריגה **stack overflow**: חריגה המתרחשת כאשר תוכנית קוד קריאה לשגרה מנסה לייצר frame חדש במחסנית, אבל חורג מגודל המקסימלי המותר למחסנית.

ענו על השאלות א'-ד' תחת **ההנחות** הבאות:

i. אין הקצאות דינאמיות (לא משתמשים ב- heap).
ii. קוד ה- assembly מקיים את קונבנציית הקריאה לפונקציות ומשתמש אך ורק ברגיסטרים: $ra,\ sp,\ a0,\ a1$.

א- סטודנט בקורס כתב קוד אסמבלי שמממש את הפונקציות הנתונות בשאלה על מעבד א' בצורה תקינה. המתרגל הריץ את הקוד של הסטודנט ללא שינוי על מעבד ב' וקיבל stack overflow. מדוע קרתה החריגה?

במעבד א' המחסנית גדלה כלפי מטה ולכן בהקצאת המחסנית צריך להקטין את ה-SP.
ברגע שמריצים את הקוד על מעבד ב', בפעם הראשונה שהפונקציה תנסה להקצות מחסנית היא תגלוש לאזור ה-HEAP ונקבל חריגה מסוג stack overflow
טעות נפוצה: הרבה סטודנטים כתבו שהסיבה היא שגודל המחסנית במעבד ב' הוא יותר קטן. שימו לב שסיבה זו אינה רלוונטית לסעיף זה כי נתון ש- $X = 1$.

כעת, נניח שהקוד שרץ על שני המעבדים הוא תקין ורץ כהלכה. בנוסף, X הוא משתנה המתקבל כקלט בפונקציה main (ולא מוגבל לערך 1).

ב- מהו גודל ה-frame (בבתים) של הפונקציה main?

8 Bytes

ג- מהו גודל ה-frame (בבתים) של הפונקציה fact?

8 Bytes

בשני הסעיפים ב' ו-ג' קיבלנו גם את התשובה $12\ Bytes$

ד- תנו דוגמה לערך של המשתנה X (המוגדר בפונקציה main) **שלא** גורם לחריגה מסוג stack overflow אם מריצים את התוכנית על מעבד א' אבל **גורם** לחריגה מסוג stack overflow אם מריצים את התוכנית על מעבד ב' (בהנחה שאין לנו את הבעיה מסעיף א').

$X = 2^9$

עבר ערך X כלשהו המעבד יקצה $X + 2\ frames$ כל אחד בגודל של 8 בתים.
כדי לגרום לחריגה על מעבד ב' צריך להשתמש ביותר מ- 4KB כמחסנית אבל פחות מ- 8KB
כדי לא לגרום לחריגה במעבד א':

$$8 \cdot (X+2)\ B > 4KB$$
$$8 \cdot (X+2)\ B > 4 \cdot 2^{10}\ B$$
$$8 \cdot (X+2) > 8 \cdot 2^9$$
$$(X+2) > 2^9$$
$$X > 2^9 - 2$$

למשל עבור $X = 2^9$

שאלה 106 (7 נקודות)

במחשב עם מעבד מסוג Pipeline RISC-V הורץ הקוד הבא:

```
lw s1, 12(s0)
addi s0, s1, 0
sub s1, s0, s1
sw s1, 4(s0)
```

נתון שהמעבד בעל forwarding **מלא** ויחידת hazard detection.
(בין השלבים: $MEM \rightarrow EX, WB \rightarrow EX, WB \rightarrow ID$)

נתון שהפקודה הראשונה נמצאת בשלב ה- IF במחזור שעון מספר 1.
מלאו את הטבלה הבאה, ע"פ המעקפים שיתבצעו במהלך ריצת התוכנית (יתכן שיש פחות מעקפים ממספר השורות):

מס' מעקף	מחזור שעון	שם רגיסטר	מאיזה שלב	לאיזה שלב
1				
2				
3				
4				
5				
6				

פתרון

מס' מעקף	מחזור שעון	שם רגיסטר	מאיזה שלב	לאיזה שלב
1	5	S1	WB	EX
2	5	S1	WB	ID
3	6	S0	MEM	EX
4	7	S0	WB	EX
5	7	S1	MEM	EX

קיבלנו גם את הפתרון שבו צריך הוכנסו שני stalls לפני ה- sw (למרות שלא צריך).