מערכות ספרתיות ומבנה המחשב (044252)

סמסטר אביב + קיץ תשע"ט

בחינה סופית – מועד ב

7 באוקטובר 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 20 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)

נתונים שני ה- modules הבאים:

```
module mymodule (
   input logic clk,
   input logic rst,
   output logic done
);
   parameter N = 3;
   logic [31:0] w;

   always_ff @(posedge clk, posedge rst) begin
      if (rst == 1'b1) begin
         w <= 32'd0;
      end
      else begin
         if (w < N - 1) begin
            w <= w + 1;
         end
         else if (w == N - 1) begin
            w <= 32'd0;
         end
      end
   end

   always_comb begin
      if (w == N - 1) begin
         done = 1'b1;
      end
      else begin
         done = 1'b0;
      end
   end

endmodule
```

```
module mymodule2 (
   input logic clk,
   input logic rst,
   output logic out
);
   logic w1;
   logic w2;
   mymodule uut1 (clk, rst, w1);
   mymodule #(.N(2)) uut2 (clk, rst, w2);

   always_comb begin
      if ((w1 == 1'b1) && (w2 == 1'b1)) begin
         out = 1'b1;
      end
      else begin
         out = 1'b0;
      end
   end

endmodule
```

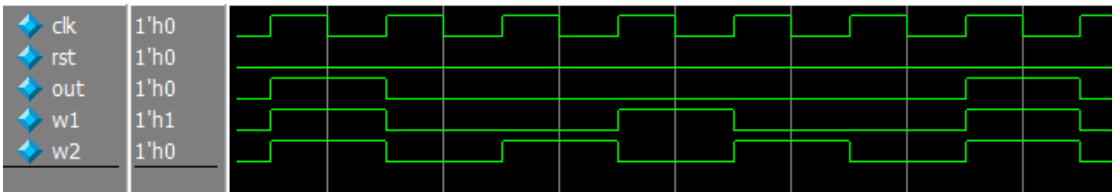
אל כניסת ה-clk של mymodule2 מחברים שעון ואל כניסת ה-rst שלו מחברים אות reset.

לאחר reset, כל כמה מחזורי שעון יקבל הסיגנל out במודול mymodule2 את הערך 1'b1?

א- 2
ב- 3
ג- 4
ד- 5
ה- 6

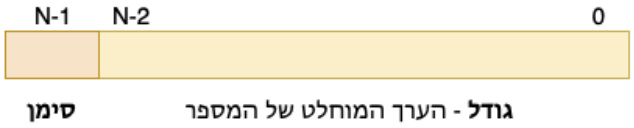
התשובה הנכונה היא ה'

ה-module הראשון שנתון בשאלה, mymodule, מממש counter שכאשר הוא מסיים לספור עד N, מוציא ביציאה done את הערך הלוגי 1. ב-mymodule2 יוצרים שני instances של ה-counter, כאשר הראשון מביניהם סופר עד 3 והשני סופר עד 2 (כי בראשון משתמשים בערך ברירת-המחדל של N, המוגדר בתוך mymodule, ובשני מגדירים ערך שונה ל-N כאשר מבצעים module instantiation). היציאה out תכיל ערך לוגי 1 כאשר סיגנלי ה-done של שני ה-instances יהיו בעלי ערך לוגי 1. זה יקרה כל 6 מחזורי שעון (המכפלה המשותפת המינימלית של שני המספרים), כפי שניתן לראות בסימולציה המצורפת:



שאלה 2 (5 נקודות)

נגדיר שיטת ייצוג חדשה בשם 'גודל וסימן' $(sign\ \&\ magnitude)$ לייצוג מספרים שלמים בעלי סימן. בשיטה זו, עבור מספר בן N סיביות, משתמש ב- $N-1$ הסיביות התחתונות כדי לייצג את הערך המוחלט של המספר בייצוג בינארי, וב- MSB כדי לקבוע את הסימן, באופן הבא:



$$sign = \begin{cases} 0 - \textbf{\textit{positive}} \\ 1 - \textbf{\textit{negative}} \end{cases}$$

ה- MSB השווה ל- '0' משמעו מספר חיובי, וה- MSB השווה ל- '1' משמעו מספר שלילי.

דוגמאות:

1- המספר 1011 בייצוג $sign\ \&\ magnitude$ שווה ל- (-3) בבסיס 10.
הסבר: המספר מורכב מ- 4 סיביות, הערך המיוצג ב- 3 הסיביות התחתונות הוא 3, וה- MSB השווה ל- '1' מסמן שהמספר הוא שלילי ולכן הערך בבסיס 10 הוא (-3).
2- המספר 0011 בייצוג $sign\ \&\ magnitude$ שווה ל- 3 בבסיס 10.
3- שימו לב ש- $10\ldots0$ ו- $00\ldots0$ שניהם מייצגים את הערך 0.

בשאלה זו, מעוניינים לממש רכיב המכפיל בין שני מספרים **בני 2 סיביות** כל אחד המיוצגים בשיטת $sign\ \&\ magnitude$, ומוציא מספר בייצוג זה בעל מספר סיביות מינימאלי הדרוש לייצוג התוצאה.

ברשותכם מספר בלתי מוגבל של השערים הלוגיים:

$$AND, \quad OR, \quad NAND, \quad NOR, \quad XOR, \quad XNOR, \quad NOT$$

לכל אחד מהשערים יש 2 כניסות בלבד (למעט שער NOT יש לו רק כניסה אחת).

מבין התשובות הבאות, מהו מספר השערים המינימלי הנדרש על מנת לממש את המכפל המבוקש?

א- 1
ב- 2
ג- 3
ד- 4
ה- 5

התשובה הנכונה היא ב'
ערכי כניסה האפשריים של כל מספר הן : 1,0,-1, ולכן מוצא המכפל יכול להיות רק 1,0,-1.
מכאן שכדי לייצג את מוצא המכפל יש להשתמש במספר בייצוג sign & magnitude בן 2 סיביות. נכתוב את טבלת האמת של המכפל (הכניסות הן a_1a_0 ו-b_1b_0, היציאה היא c_1c_0).
נזכור שייצוג של 0 יכול להיות 00 או 10.

b_1	b_0	a_1	a_0	c_1	c_0
0	0	0	0	ϕ	0
0	0	0	1	ϕ	0
0	0	1	0	ϕ	0
0	0	1	1	ϕ	0
0	1	0	0	ϕ	0
0	1	0	1	0	1
0	1	1	0	ϕ	0
0	1	1	1	1	1
1	0	0	0	ϕ	0
1	0	0	1	ϕ	0
1	0	1	0	ϕ	0
1	0	1	1	ϕ	0
1	1	0	0	ϕ	0
1	1	0	1	1	1
1	1	1	0	ϕ	0
1	1	1	1	0	1

לכן מפות קרנו של המוצאים זה:
<u>:C0</u>

$a_1a_0\backslash b_1b_0$	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	0	1	1	0
10	0	0	0	0

$$c_0 = a_0b_0$$

<u>:C1</u>

$a_1a_0\backslash b_1b_0$	00	01	11	10
00	ϕ	ϕ	ϕ	ϕ
01	ϕ	0	1	ϕ
11	ϕ	1	0	ϕ
10	ϕ	ϕ	ϕ	ϕ

$$c_1 = a_1\overline{b_1} + \overline{a_1}b_1 = a_1 \oplus b_1$$

לכן, כדי לממש את המכפל צריך להשתמש רק בשניים מהשערים הנתונים: 2-input-AND ו-2-input-XOR.

שאלה 3 (5 נקודות)

שני מהנדסים מתקשרים ביניהם באמצעות "קוד חזרות" באורך $r > 2$. בקוד זה, כל סיבית משודרת r פעמים ברצף.
ידוע שלפעמים, על קו התקשורת נוצרת שגיאה בצורת התהפכות ביט אחד בלבד.
אחד המהנדסים מעוניין לתכנן מערכת צירופית שמטרתה לשחזר את הביט המקורי שנשלח, מתוך המילה שהתקבלה. כלומר, המערכת מקבלת כקלט מילה באורך r ומייצרת פלט בן סיבית אחת שערכה כערך הסיבית שהופיעה יותר פעמים במילה.

דוגמה:
עבור $r = 4$ והקלט 1011, הפלט של המערכת יהיה 1, מכיוון שיש יותר '1' מ- '0' בקלט המערכת.

תכננו את המערכת עבור $r = 4$ וצמצמו אותה ככל שניתן כסכום מכפלות.
נתונים השערים הבאים:

	tpd_{LH}	tpd_{HL}
AND	$10ns$	$25ns$
OR	$15ns$	$5ns$
NOT	$5ns$	$6ns$

הערה: לכל אחד מהשערים יש 2 כניסות בלבד (למעט שער NOT יש לו רק כניסה אחת).

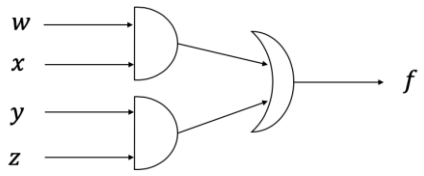
מבין התשובות הבאות, מהו ה- t_{pd} ההדוק ביותר של המערכת?

א- $25\ ns$
ב- $30\ ns$
ג- $35\ ns$
ד- $36\ ns$
ה- $40\ ns$

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

התשובה הנכונה היא ב'

לכן הפונקציה הינה: $f(w,x,y,z) = yz + wx$
שימו לב שאין צורך בשימוש במכפלות נוספות, שכן כל ה- '1' כבר מסומנים.



ולכן חישוב ה- tpd הינו:

$$tpd_{LH} = tpd_{LH}(AND) + tpd_{LH}(OR) = 25ns$$
$$tpd_{HL} = tpd_{HL}(AND) + tpd_{HL}(OR) = 30ns$$
$$\rightarrow tpd = 30ns$$

שאלה 4 (5 נקודות)

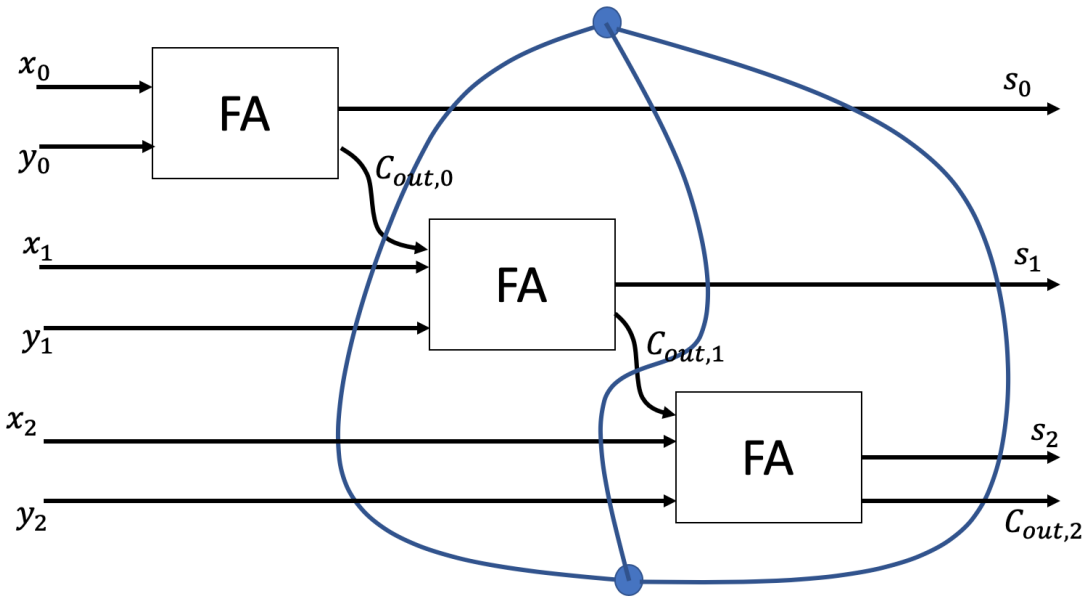
מהנדסת חשמל מעוניינת לחשב סכום של מספרים המאוחסנים בשני רגיסטרים בעלי **3** ביטים כל אחד. תוצאת החישוב בעלת 4 ביטים (הביט הנוסף הוא ה- Carry Out של הרכיב האחרון).
בידי המהנדסת ישנם 3 רכיבי Full Adder צירופיים, בעלי סיביות כניסה a, b, C_{in} וסיביות מוצא s, C_{out} (כל אחת מהכניסות והיציאות הינה סיבית בודדת).

המהנדסת מעוניינת לבנות מערכת מצונרת (Pipeline) כפי שלמדה בקורס, על מנת להשיג תפוקה (Throughput) גבוהה ככל האפשר בעדיפות ראשונה, ושימוש במינימום פליפ-פלופים בעדיפות שניה.

מבין התשובות הבאות, מהי כמות הפליפ-פלופים המינימלית הנדרשת למימוש המערכת?

א- 13
ב- 14
ג- 15
ד- 16
ה- 17

התשובה הנכונה היא ג'



שאלה 5 (5 נקודות)

מהנדסת פישטה פונקציה כלשהי $f(x,y,z)$ לצורה **המצומצמת ביותר כמכפלת סכומים**, וקיבלה את הביטוי הבא:

$$x \cdot (z' + y)$$

בהתייחסות לצורה של **סכום מכפלות**, איזו מהטענות הבאות נכונה?

א- ייתכן שהביטוי המצומצם ביותר בצורה של סכום מכפלות הוא: x, רק במקרה שקיים לפחות $don't\ care$ אחד במפה.

ב- ייתכן שהביטוי המצומצם ביותר בצורה של סכום מכפלות הוא: x, גם אם לא קיימים $don't\ care$ במפה.

ג- ייתכן שהביטוי המצומצם ביותר בצורה של סכום מכפלות הוא: xy, רק במקרה שקיים לפחות $don't\ care$ אחד במפה.

ד- ייתכן שהביטוי המצומצם ביותר בצורה של סכום מכפלות הוא: xy, גם אם לא קיימים $don't\ care$ במפה.

ה- בהכרח מתקיים ש- xyz שווה ל- 1.

התשובה הנכונה היא ה'

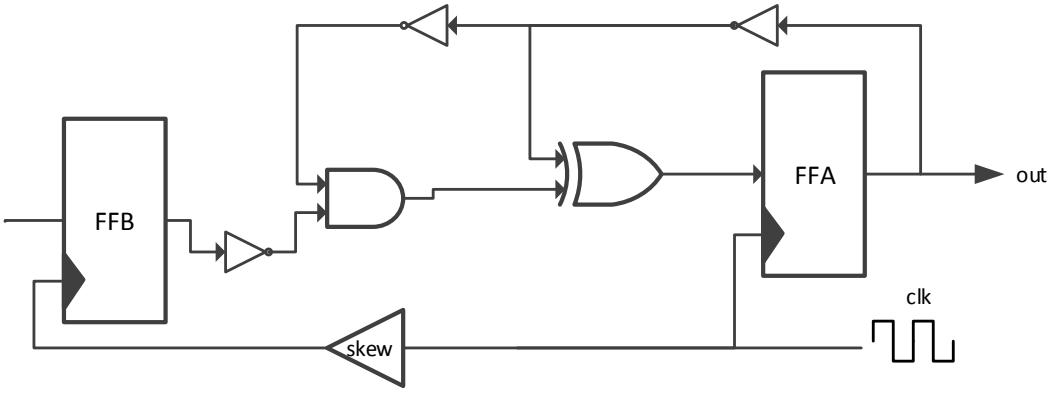
הפונקציה $x \cdot (z' + y)$ כמכפלת סכומים ללא $don't\ care$ נראית כך:

z \ xy	00	01	11	10
0	0	0		
1	0	0		0

תשובה א' – אם נוסיף צירופי ברירה כדי לקבל את x כסכום מכפלות נקבל ש $x \cdot y' \cdot z = \emptyset$, ואז יהיה ניתן לצמצם את הפונקציה כמכפלת סכומים בסתירה לנתון שהפונקציה מצומצמת.
תשובה ב' – לא אפשרי, כיוון שיש חפיפה במפה בין הייצוג כמכפלת סכומים והייצוג כסכום מכפלות, וזה יתכן רק במקרה שיש *don't care*.
תשובה ג' – לא יתכן, כיוון שבמקרה זה $xy'z'$ הוא *don't care*, אבל אז ניתן לצמצם את $x(z' + y)$ ל- $x \cdot y$ בסתירה לנתון שהפונקציה מצומצמת.
תשובה ד' – לא יתכן, כיוון שבמקרה זה בהכרח יש '1' ב- $xyz', xyz, xy'z'$, ככה שהפונקציה כסכום מכפלות היא: $xz' + xy$
תשובה ה' – אם נניח בשלילה שב- xyz יש 0 או צירוף ברירה, נקבל שיהיה ניתן לצמצם את הפונקציה כמכפלת סכומים בסתירה לנתון, ולכן בהכרח יש שם '1'.

שאלה 6 (5 נקודות)

נתון המעגל הבא:



ונתונים מאפייני רכיבי המעגל:

	NOT	AND	XOR	FF
T_{pd}/T_{pCQ}	$10\ ns$	$20\ ns$	$30\ ns$	$40\ ns$
T_{cd}/T_{cCQ}	$5\ ns$	$10\ ns$	$15\ ns$	$20\ ns$
T_{su}				$10\ ns$

שני ה- FFים חוברו לשעון המעגל clk, אך נוצר t_{skew} ביניהם, כך שהשעון שמזין את FFA מגיע t_{skew} לפני השעון שמזין את FFB. בנוסף, נתוני FFB זהים לאלו של FFA.

בנוסף, נתון ש $t_{skew} = 30ns$.

בהנחה שתנאי $hold$ מתקיים, מהו תדר העבודה המרבי בו ניתן להפעיל את המעגל?

א- $9.09\ MHz$
ב- $8.33\ MHz$
ג- $7.14\ MHz$
ד- $6.67\ MHz$
ה- $6.25\ MHz$

התשובה הנכונה היא ג'
המסלול הארוך ביותר מיציאת FFb לכניסת FFa:

$$Tmin \leq TpCQ(FFb) + Tpd(NOT) + Tpd(AND) + Tpd(XOR) + Tsu(FFa) + skew = 140ns$$

המסלול מיציאת FFa לכניסתו:

$$Tmin \leq TpCQ(FFa) + 2Tpd(NOT) + Tpd(AND) + Tpd(XOR) + Tsu(FFa) = 120ns$$

בעקבות ההשהיה של השעון, המסלול בין ה FFים ארוך יותר ולכן קובע את T_{min}:
1/140ns = 7.14MHz.

שאלה 7 (5 נקודות)

נתון קטע הקוד הבא:

$$
\begin{aligned}
&slli \quad t0, s0, 3\\
&add \;\; t0, t0, s1\\
&mul\; s3, t0, s2\\
&mul\; s3, s3, s3\\
&mul\; s3, s3, t0
\end{aligned}
$$

נתון שלפני הרצת הקוד:

$$
\begin{aligned}
s0 &= x\\
s1 &= y\\
s2 &= z
\end{aligned}
$$

מה יהיה הערך ברגיסטר $s3$ בסיום ריצת התוכנית?

א- $[z^2(\mathrm{x}+8\mathrm{y})^2]\ mod\ 2^{32}$
ב- $[z^2(\mathrm{x}+8\mathrm{y})^3]\ mod\ 2^{32}$
ג- $[z^3(\mathrm{x}+8\mathrm{y})^3]\ mod\ 2^{32}$
ד- $[z^2(8\mathrm{x}+\mathrm{y})^3]\ mod\ 2^{32}$
ה- $[z^3(8\mathrm{x}+\mathrm{y})^3]\ mod\ 2^{32}$

התשובה הנכונה היא ד'

```
sll $t0, $s0, 3          // $t0 = 8x
add $t0, $t0, $s1        // $t0 = 8x+y
mult $s3, $t0, $s2       // $s3 = z(8x+y)
mult $s3, $s3, $s3       // $s3 = z^2(8x+y)^2
mult $s3, $s3, $t0       // $s3 = z^2(8x+y)^3
```

שאלה 8 (5 נקודות)

נתונה הפונקציה הבאה:

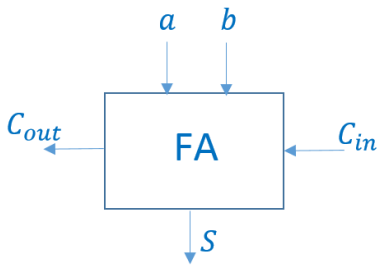
$$f(w,x,y,z) = \sum(0,2,5,8,10,13,15) + \sum_{\emptyset}(3,7,11)$$

מעוניינים לממש את הפונקציה $f(w,x,y,z)$ בעזרת רכיבי FA והקבועים '0' ו-'1' **בלבד**.

מבין התשובות הבאות, מהו מספר רכיבי ה- FA **המינימלי** הנדרש כדי לממש את הפונקציה $f(w,x,y,z)$?

א- 1
ב- 2
ג- 3
ד- 4
ה- לא ניתן לממש את הפונקציה בעזרת רכיבי FA וקבועים בלבד.

תזכורת:



$$S(a,b,C_{in}) = a \oplus b \oplus C_{in}$$
$$C_{out}(a,b,C_{in}) = a \cdot b + a \cdot C_{in} + b \cdot C_{in}$$

התשובה הנכונה היא א'

פתרון

$\frac{wx}{yz}$	00	01	11	10
00	1			1
01		1	1	
11	∅	∅	1	∅
10	1			1

$$f(w,x,y,z) = x'z' + xz = \overline{x \oplus y} = x \oplus y \oplus '1' = S(x,y,'1')$$

שאלות 9 (5 נקודות)

בחברת "אין יותר הזדמנויות" מעוניינים לתכנן מנעול סדרתי בן 3 ספרות עשרוניות 0-9 באופן הבא:
בעת הכנסת הסיסמה, נותנים למשתמש 3 הזדמנויות כדי להכניס את רצף הספרות הנכון.
עבור כל ספרה נכונה שהמשתמש מכניס, המנעול מדליק נורה ירוקה ועוברים לספרה הבאה,
אחרת המנעול מדליק נורה אדומה ונותן למשתמש עוד הזדמנות להכניס את הספרה מחדש.
שימו לב שיש חשיבות לסדר הכנסת הספרות, וכשהמשתמש מכניס ספרה שגויה הוא מקבל עוד הזדמנות להכניס את הספרה הנכונה ואינו מתחיל מחדש.
אם המשתמש מכניס ספרה שגויה **3 פעמים** (לאו דווקא ברצף), המנעול יינעל לנצח ולא יהיה ניתן לפתוח אותו שוב. אחרת, כל פעם שמכניסים את הסיסמה הנכונה, המנעול נפתח ונותנים למשתמש 3 הזדמנויות חדשות לפעם הבאה שבה הוא ינסה לפתוח את המנעול.

דוגמה
בהנחה שהסיסמה הנכונה היא 123

$\#cycle$	1	2	3	4	5	6	7	8	9
$input$	0	1	6	2	3	1	7	2	9
$output$	red	$green$	red	$green$	$green$	$green$	red	$green$	red

$\#cycle$	10	11	12
$input$	9	3	3
$output$	red	red	red

הסבר: במחזור שעון 5 המנעול נפתח בפעם הראשונה ולכן מספר ההזדמנויות חוזר להיות 3 (למרות שבמחזור שעון 1 ו-3 הוכנסו ספרות שגויות). במחזור שעון 10 הוכנסה ספרה שגויה בפעם השלישית ולכן המנעול ננעל לנצח והנורה תראה צבע אדום כל הזמן למרות שבמחזור שעון 11 הוכנסה הספרה האחרונה הנכונה.

מהו מספר המצבים במכונת מילי המצומצמת?

א- 8
ב- 9
ג- 10
ד- 11
ה- 12 או יותר

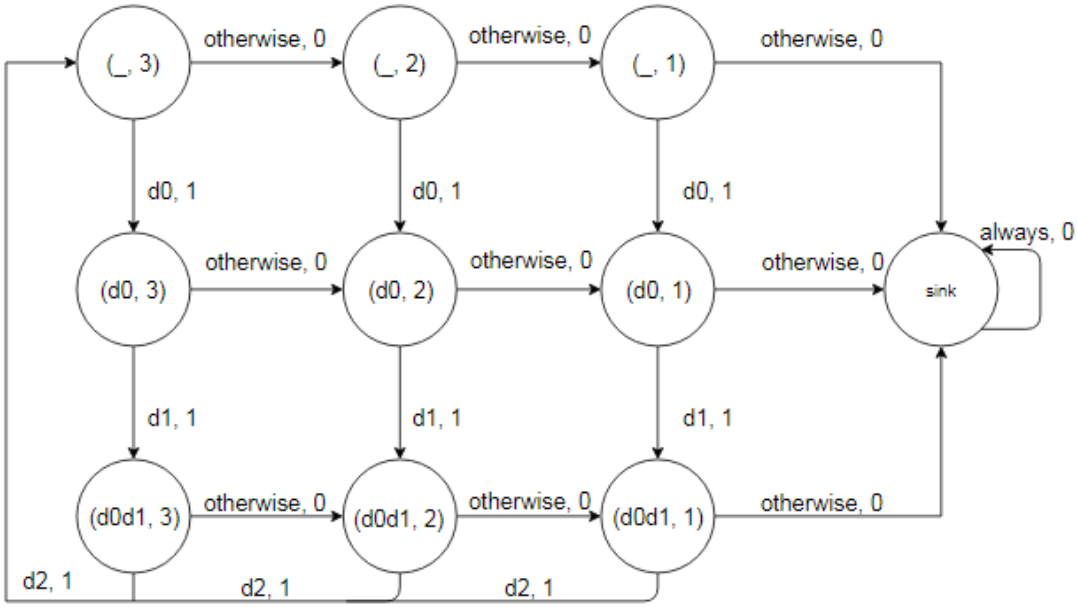
התשובה הנכונה היא ג'

דרך א':
מה צריך לזכור?
1- כמה שגיאות התקבלו עד עכשיו: 0,1,2
2- לאיזה ספרה אנחנו מחכים בקלט: הראשונה, השנייה או השלישית
3- מצב בור שכל הזמן מוציא צבע אדום

מאחר והמאורעות 1 ו-2 בלתי תלויים, כדי למצוא את כל הקומבינציות האפשריות צריך להכפיל בין מספר האפשרויות של כל מאורע ולכן סה"כ מספר המצבים יהיה:

$$\#states = 3 \cdot 3 + 1 = 10$$

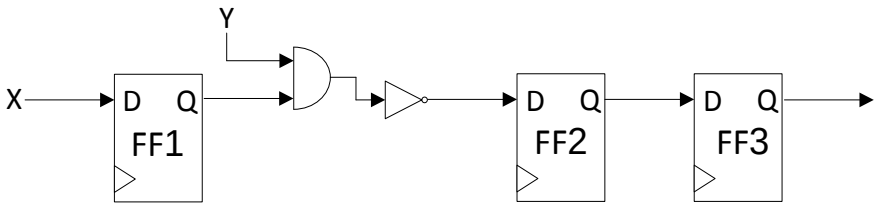
דרך ב' - לצייר את מכונת המצבים:



בתוך כל מצב רשום את רצף הספרות החוקיות שהמצב זוכר (בהנחה שהסיסמה הנכונה היא $d_0d_1d_2$), ומספר ההזדמנויות שנותרו.
על החצים רשום הקלט הנוכחי, והמוצא של המערכת (0 אדום, 1 ירוק).

שאלה 10 (7 נקודות)

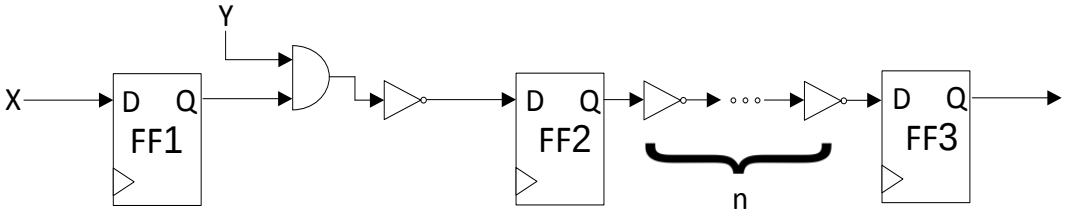
מהנדס מעוניין לממש את המעגל הבא, עבור זמן מחזור של $T_{clk} = 15\ ns$



נתונים הרכיבים (נתוני רכיב ה-NOT חסרים):

$[ns]$	$t_{cd} \backslash t_{cCQ}$	$t_{pd} \backslash t_{pCQ}$	t_{setup}	t_{hold}
AND	2	6		
FF	1	4	3	4

המהנדס שם לב שתנאי hold של FF3 בוודאות לא מתקיים. כדי לתקן זאת, המהנדס מעוניין להכניס n שערי NOT בין המוצא של FF2 לכניסה המידע של FF3, כמתואר באיור הבא:



נתון שהמעגל עומד בכל תנאי ה-setup ו-hold <u>וסיגנל המוצא (מוצא FF3) מתנהג כפי שהמהנדס התכוון במעגל המקורי</u>.

רשמו את תנאי setup של המעגל בין FF1 ל-FF2:

רשמו את תנאי setup של המעגל בין FF2 ל-FF3 כתלות ב-n:

רשמו את תנאי hold של המעגל בין FF1 ל-FF2:

רשמו את תנאי hold של המעגל בין FF2 ל-FF3 כתלות ב-n:

מה הערכים האפשריים ל- n שיבטיחו פעולה תקינה של המעגל?

פתרון

בהינתן שתנאי hold של רכיבי FF2 ו-FF3 מתקיימים נקבל את האי-שיוויונים הבאים:

$$t_{hold} \le t_{cCQ} + t_{cd}(AND) + t_{cd}(NOT) \Rightarrow 4 \le 1 + 2 + t_{cd}(NOT) \Rightarrow 1 \le t_{cd}(NOT)$$

$$t_{hold} \le t_{cCQ} + n \cdot t_{cd}(NOT) \Rightarrow 4 \le 1 + n \cdot t_{cd}(NOT) \Rightarrow 3 \le n \cdot t_{cd}(NOT)$$

כדי שהמעגל יתפקד לכל ערכי $t_{cd}(NOT)$ האפשריים צריך להתקיים האי-שיוויון הבא:

$$3 \le n \cdot 1 \Rightarrow 3 \le n$$

בהינתן שתנאי setup של רכיבי FF2 ו-FF3 מתקיימים נקבל את האי-שיוויונים הבאים:

$$T - t_{setup} \ge t_{pCQ} + t_{pd}(AND) + t_{pd}(NOT) \Rightarrow 15 - 3 \ge 4 + 6 + t_{pd}(NOT)$$
$$\Rightarrow t_{pd}(NOT) \le 2$$

$$T - t_{setup} \ge t_{pCQ} + n \cdot t_{pd}(NOT) \Rightarrow 15 - 3 \ge 4 + n \cdot t_{pd}(NOT)$$
$$\Rightarrow n \cdot t_{pd}(NOT) \le 8$$

כדי שהמעגל יתפקד לכל ערכי $t_{pd}(NOT)$ האפשריים צריך להתקיים האי-שיוויון הבא:

$$n \cdot 2 \le 8 \Rightarrow n \le 4$$

מכאן שקיבלנו את התנאים הבאים לתקינות המעגל מבחינה חשמלית:

$$3 \le n \le 4$$

כדי שהמעגל החדש יתפקד לוגית כמו המעגל המקורי צריך ש-n יהיה זוגי (כדי שפעולות הלוגיות של שערי ה-NOT יבטלו אחד את השני), ולכן נקבל שהתשובה הסופית צריכה להיות:

$$n = 4$$

שאלה 11 (5 נקודות)

מהנדס מתכנן ערוץ תקשורת בין שתי מערכות המבוסס על פרוטוקול UART המעביר 8 סיביות מידע בכל שליחה.
נתון שהמערכות צריכות להעביר ביניהן מילים בנות 32 סיביות בכל העברת מידע, כאשר קצב העברת המילים הללו צריך להיות 1000 מילים בשניה.

מה צריך להיות ה-T_{bit} של פרוטוקול ה-UART כדי שהתקשורת בין המערכות תפעל כרצוי?

פתרון

פרוטוקול UART מעביר בכל שליחה רק 8 סיביות, ולכן צריך 4 העברות ב-UART כדי לשלוח מילה יחידה בת 32 סיביות.
זמן העברת 8 סיביות ב-UART הוא $10 \cdot T_{bit}$ בגלל שיש מחזור T_{bit} ל-start-bit ול--stop bit בנוסף לזמן מחזור לכל סיבית מידע.
לכן, זמן העברת מילה בת 32 סיביות הוא $4 \cdot 10 \cdot T_{bit} = 40 \cdot T_{bit}$, וקצב העברת המילים הללו הוא $\frac{1}{40 \cdot T_{bit}}$. מהנתון שקצב זה צריך להיות 1000 מילים לשנייה, נקבל:

$$\frac{1}{40 \cdot T_{bit}} = 1000$$

$$T_{bit} = \frac{1}{40 \cdot 1000} = \frac{1\ mSec}{40} = 0.025\ mSec = 25\ \mu Sec$$

שאלה 12 (8 נקודות)

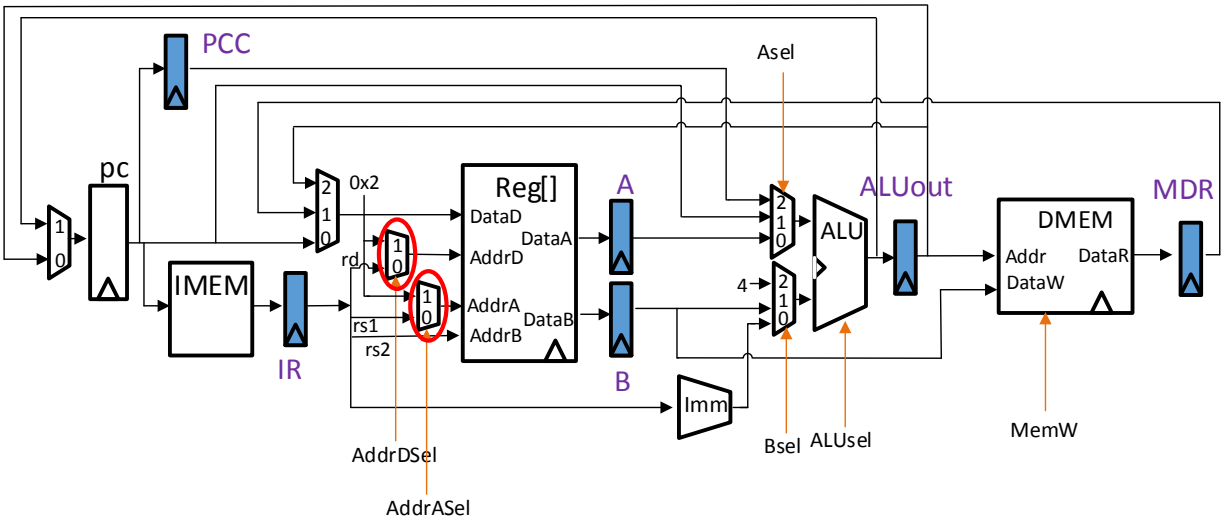
כדי לייעל את כתיבת הקוד של תכנית רוצים להוסיף פקודת $push$ למעבד $MultiCycle\ RISCV$. פקודת push נכתבת כך:

push rs2

כלומר, הפקודה מקבלת רק רגיסטר יחיד כאופרנד. פעולת הפקודה היא להגדיל את המחסנית (ה- stack) במילה אחת ולהכניס את ערך הרגיסטר rs2 למיקום זה.

לצורך ביצוע הפקודה הוספו שני בוררים למעבד $MultiCycle\ RISCV$. שני הבוררים מסומנים בעיגול באיור מטה.

מלאו את הטבלה מטה בערכי הסיגנלים המצוינים, כך שפעולת הסיגנלים בכל מחזור שעון תתאים למימוש פקודת ה-push.
יש למלא את הטבלה כך שמספר מחזורי השעון הממשים את הפקודה הוא מינימאלי (אין הכרח למלא את כל העמודות הריקות בטבלה).
הסיגנלים $AddrAsel$ $AddrDsel$, יכולים לקבל את הערכים 0,1 או ϕ.
הסיגנלים $Asel, Bsel$ יכולים לקבל את הערכים 0,1,2 או ϕ.
הסיגנל $ALUsel$ מקבל שמות של פעולות (למשל: add, sub, xor וכו') לפי הפעולה שעליו לבצע, או את הערך ϕ אם סוג הפעולה אינו משנה.
הסיגנל $MemW$ יכול לקבל את הערכים 0 (עבור קריאה מה-DMEM) או 1 (עבור כתיבה ל-DMEM).



מס' מחזור שעון	1	2	3	4	5	6	7	8	9
AddrAsel									
AddrDsel									
Asel									
Bsel									
ALUsel									
MemW									

פתרון

נשים לב כי הבוררים הנוספים מאפשרים לנו לבחור באופן קבוע את רגיסטר 2, שהוא ה-stack-pointer. אז כדי לבצע את הפקודה נצטרך לבחור ברגיסטר sp בעזרת הבורר AddrAsel, לבצע בשלב ה-EX את sp-4 ולשמור את התוצאה ברגיסטר ALUout. במחזור שעון לאחר מכן להשתמש ב-ALUout ככתובת ל-Mem וכמידע חדש ל-register-file, כך נכתוב לזכרון ונכתוב לרגיסטר sp באותו מחזור שעון (כלומר נעשה Memory access ו-Write back באותו מחזור שעון).
נשים לב אבל ש-2 מחזורי השעון הראשונים צריכים להישאר אותו הדבר כמו שנלמד בכתה. כלומר ששלב ה-decode לוקח את שדה ה-rs1 מהפקודה ככתובת לפורט A ב- register file. לכן, בשלב ה-decode נצטרך לקבוע את AddrAsel = 0 והערך של sp לא יהיה זמין במחזור שעון השלישי. לכן, צריך לבצע עוד מחזור שעון של decode בא רק נקרא את הערך של sp מה-register file עם AddrAsel = 1.
סה"כ נצטרך 5 מחזורי שעון כדי לבצע את הפקודה push.

הטבלה הבאה מסכמת את ערך הסיגנאלים בכל אחד ממחזורי השעון של הפקודה:

	IF	ID	ID2	EX	Mem+WB
AddrDsel	ϕ	ϕ	ϕ	ϕ	1
AddrAsel	ϕ	0	1	ϕ	ϕ
Bsel	2	0	ϕ	2	ϕ
Asel	1	2	ϕ	0	ϕ
ALUsel	add	add	ϕ	sub	ϕ
MemW	0	0	0	0	1

שאלה 13 (12 נקודות)

מהנדס חשמל שעובד בחברה המתמחה בראייה ממוחשבת מעוניין לבצע מכפלה סקלרית בין וקטורים.
הווקטורים שמורים בתור שני מערכים בזיכרון, כאשר כל איבר בווקטור שמור כמילה בזיכרון.

- הרגיסטר S0 מאותחל לראשית המערך הראשון.
- הרגיסטר S1 מאותחל לראשית המערך השני.
- הרגיסטר S2 מאותחל לערך 10, שהינו גודל המערכים.
- עבור כל פעולת הכפלה שנבצע, נגדיל את S5 ב-1 (רגיסטר זה מודד את סיבוכיות האלגוריתם). רגיסטר S5 מאותחל לערך 0 לפני ריצת הקוד להלן.

לבסוף, תוצאת המכפלה נשמרת בכתובת שמאותחלת ברגיסטר S3.

להזכירכם, מכפלה סקלרית בין ווקטורים מוגדרת באופן הבא:
$< v, u > = v_1 u_1 + v_2 u_2 + \cdots + v_n u_n$.

נתון הקוד הבא, שמבצע את הנדרש:

```
1          addi  t3,  x0,  0
2   loop:  lw    t0,  0(s0)
3          lw    t1,  0(s1)
4          mul   t2,  t1,  t0
5          add   t3,  t3,  t2
6          addi  s5   s5,  1
7          addi  s0,  s0,  4
8          addi  s1,  s1,  4
9          addi  s2,  s2,  -1
10         bne   s2,  x0,  loop
11  end:   sw    t3,  0(s3)
```

החברה מעוניינת להשוות בין שני סוגי מעבדים: $Single\ Cycle\ RISCV$ ו- $Pipeline\ RISCV$, ולבסוף להחליט איפה להריץ את הקוד.

זמני ריצת השלבים השונים במעבד נתונים בטבלה הבאה (זהים לשני סוגי המעבדים):

Fetch	Decode	Execute	Memory	Writeback
300ns	150ns	200ns	300ns	50ns

א- מה יהיה זמן הריצה של הקוד כולו על גבי מעבד מסוג $Single\ Cycle\ RISCV$?

$$T = T_{fetch} + T_{decode} + T_{exe} + T_{mem} + T_{wb} = 1000ns$$

כלומר, זמן ביצוע פקודה הינו 1000ns. אנחנו יודעים שהלולאה תרוץ 10 פעמים, וישנן 9 פקודות לבצע.
בנוסף יש עוד 2 פקודות לבצע (Addi בהתחלה ו- SW בסוף). מספר הפקודות הכולל שירוצו הינן:

$$\#ops = 9 \cdot 10 + 2$$

בסה"כ, נקבל שזמן הריצה הינו:

$$T_{total} = \#ops \cdot T = 92 \cdot 1000ns = 92{,}000ns = 92\mu s$$

כעת החברה מעוניינת להשתמש במעבד מסוג $Pipeline\ RISCV$. נתוני המעבד:

- מכיל Bypass בתוך ה- Register File (כלומר, Forwarding בין שלב ה- WB לשלב ה- Decode).
- **אינו** מכיל Forwarding נוסף.
- **אינו** מכיל Load Hazard Detection Unit.
- מניח ש- Branch אינו נלקח, ומבצע Flush ל- Pipe במידה שכן. החלטות על Branch נלקחות בשלב ה- Execute.
- זמני חלקי המעבד זהים למעבד ה- $Single\ Cycle\ RISCV$.

ב- עזרו למהנדס שתכנן את הקוד והוסיפו פקודות NOP בטבלה להלן במידת הצורך.

לדוגמה: נניח שאנו מעוניינים להוסיף 3 פקודות NOP בין שורה 4 ל- 5, נרשום זאת כך:

אחרי שורה מס'	לפני שורה מס'	כמות פקודות ה- NOP שנדרש להוסיף
4	5	3

מלאו את הטבלה הבאה (אין הכרח למלא את כולה):

אחרי שורה מס'	לפני שורה מס'	כמות פקודות ה- NOP שנדרש להוסיף
3	4	2
4	5	2
9	10	2

שימו לב: סעיפים ג' ו-ד' בעמוד הבא.

ג- המהנדס שאינו מרוצה מהפתרון שהצעתם, ביקש שתשנו את הקוד כך שנצטרך להוסיף פחות פקודות NOP. הציעו דוגמה אחת לשיפור הקוד, כך שיכיל פחות פקודות NOP.

הפתרון האופטימלי

```
1          addi  t3,  x0,    0
2   Loop:  lw    t0,  0(s0)
3          lw    t1,  0(s1)
6          addi  s5   s5     1
7          addi  s0,  s0,    4
4          mul   t2,  t1,    t0
9          addi  s2,  s2,    -1
8          addi  s1,  s1,    4
5          add   t3,  t3,    t2
10         bne   s2,  x0,    loop
11  End:   sw    t3,  0(s3)
```

כל פתרון שבו הוצע סידור מחדש של שורות הקוד ללא פגיעה בפעילות הקוד קיבל את מלוא הנקודות (פתרון שגוי למשל הינו - החלפת שורות 4 ו- 5 אשר יפחיתו NOP אחד, אך פקודה 4 כותבת לתוך רגיסטר שפקודה 5 קוראת ממנו, דבר שלבסוף ישים ערך לא נכון ברגיסטרים).
הפתרון של הזזת פקודת ה- bne לתחילת הלולאה לא התקבל משום שעדיין נצטרך להוסיף פעולת קפיצה כלשהי בסוף הלולאה. שימוש ב- jal או ב branch ע"מ לקפוץ לתחילת הלולאה עדיין ידרוש ריקון ה- pipe.

ד- המהנדס שכעת מעוניין לבחור האם להשתמש במעבד Single Cycle או – Pipeline, מעוניין להשוות ביניהם מבחינת זמן ריצה כולל.
בהנחה שבסעיף ג' הצעתם פתרון שמאפשר קוד ללא פקודות NOP בכלל, מה יהיה זמן הריצה הכולל על מעבד Pipeline?

זמן המחזור של מעבד ה- PP-RV הינו 300ns.
חישוב מספר המחזורים שנדרשים עד להרצת פקודת ה- SW האחרונה:
ראשית – 5 מחזורים לביצוע פקודת ה- Addi הראשונה), ומילוי הצינור.
כעת במהלך 9 המחזורים הבאים תבוצענה הפקודות הבאות (בגוף הלולאה).
במהלך שני המחזורים הבאים יטענו פקודות מספר 11 ו- 12, ונצטרך לעשות להן Flush (כלומר יתווספו שתי פקודות NOP) סה"כ, 11 מחזורים לביצוע הלולאה פעם אחת.
נחזור על הלולאה 9 פעמים. בריצה ה- 10 והאחרונה של הלולאה, נבצע את 9 הפקודות + פקודות ה- SW (מבלי לעשות Flush), סה"כ 10 מחזורים. כלומר:

$$\#cycles = 5 + 9 \cdot 11 + 10 = 114$$

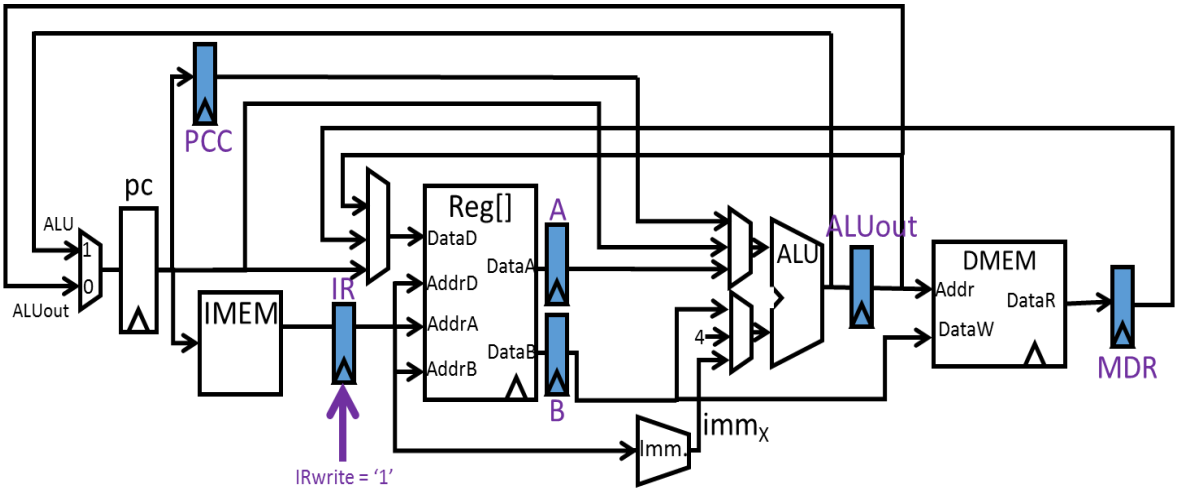
וזמן הריצה יהיה:

$$T_{PP} = 114 \cdot 300ns = 34{,}200ns$$

לעומת זמן הריצה של מעבד ה- SC: $92{,}000ns$.
לכן, המהנדס יבחר להשתמש במעבד ה- PP.
הרבה מאוד סטודנטים קיבלו שזמן הריצה ע"ג מעבד ה- Pipeline ארוך יותר מה Single Cycle. זהו מצב שיכול לקרות רק במקרים נדירים, ואתם צריכים לשים לב שהפתרון שקיבלתם הגיוני! סטודנטים נוספים התייחסו ל- Flush במקרים של קפיצה בתור הוספת פקודות NOP, ולכן לא הכניסו אותם לחישוב (התשובה שהתקבלה להם היא 28,880ns). תשובה זו אינה נכונה והתקבל עליה ניקוד חלקי.

שאלה 14 (8 נקודות)

נתון מעבד $MultiCycle\ RISCV$ כפי שנלמד בכיתה. בעת ייצור בקר המעבד קרתה תקלה והקו $IRwrite$ חובר קבוע ל '1' לוגי:



האם קיימת פקודה אחת או יותר שיעבדו כהלכה למרות התקלה (הקף בעיגול)?

כן / לא

אם סימנתם כן, מי מהפקודות הבאות <u>בהכרח</u> תעבודנה כהלכה למרות התקלה (הקף בעיגול)?

R-type **LW** **SW** **BEQ** **Jump**

פתרון

רק פקודת BEQ תעבוד כהלכה.
במעבד ללא תקלה, הפקודה הנוכחית (שנמצאת בכתובת PC) תטען לרגיסטר IR בעליית השעון בסוף שלב ה Fetch ואז תישאר ללא שינוי עד סוף ריצת הפקודה. התקלה תגרום לכך שהפקודה הבאה (שנמצאת בכתובת PC+4) תטען לרגיסטר IR בעליית שעון בסוף שלב ה Decode. במקרה זה, הרגיסטרים הנכונים rs1,rs2 יקראו בשלב ה Decode (ויטענו לרגיסטרים A ו- B בסופו), כך שניתן להשתמש בהם נכונה בשלב ה Execute.
במקרה של BEQ, בשלב ה- Execute משווים בין הערכים השמורים ברגיסטרים A ו- B וקופצים לכתובת ששמורה ברגיסטר ALUout רק אם הם שווים. זה מתבצע באופן תקין.
פקודות R-type לא יתבצעו כהלכה, כיוון שבשלב ה WB צריך לכתוב את הערך שחושב ב- ALU לרגיסטר rd, ובשלב זה כבר לא ידוע מיהו ה- rd של הפקודה הנוכחית.
באותו האופן, פקודות LW לא יתבצעו כהלכה כיוון שבשלב ה WB צריך לכתוב את הערך שנקרא מהזיכרון לרגיסטר rd שאינו ידוע מיהו.
פקודת SW לא תתבצע כהלכה כיוון שבשלב ה Memory כבר יהיה ברגיסטר B הערך מרגיסטר rs2 של הפקודה הבאה, כך שיכתב הערך הלא נכון (לכתובת הנכונה).
פקודת jump לא תעבוד כהלכה כיוון שבשלב ה execute ה immediate נקרא מרגיסטר IR ומחובר ע"י ה- ALU ל- PC הנוכחי (מרגיסטר PCC). כאמור, בשלב זה, הפקודה ששמורה ב IR זו כבר הפקודה הבאה ולכן ה- immediate יהיה שגוי.

שאלה 15 (8 נקודות)

במעבד מסוג RISCV מעוניינים להוסיף תמיכה בפקודה חדשה בשם dam (double access memory):

$$dam\ rd, rs, imm \quad \#\ RegFile[rd] = Mem[\ Mem[rs + imm]\ ]$$

הפקודה ניגשת **לזיכרון המידע** פעמיים: בפעם הראשונה ניגשים לכתובת שמחושבת ע"י חיבור תוכן הרגיסטר rs וערך ה-imm. בפעם השנייה ניגשים לכתובת שנקראת מתוך הזיכרון בגישה הראשונה, ואת הערך שקוראים מתוך הזיכרון כותבים לתוך הרגיסטר rd. שימו לב, זיכרון המידע זהה לנלמד בקורס.

ענו על שני הסעיפים הבאים ב- **נכון / לא נכון** (הקיפו בעיגול) עם הסבר קצר – **לכל היותר** שורה אחת.
(בכל הטענות אנחנו מתייחסים למימוש של פקודת dam אמיתית ולא פסאודו-פקודה):

א- ניתן לממש את הפקודה על מעבד מסוג Single Cycle RISCV לאחר הכנסת שינויים למסלול הנתונים ו/או לבקר ו/או הגדלת זמן המחזור של השעון.

הסבר: לא נכון, לא ניתן לגשת לזיכרון פעמיים במחזור שעון אחד.

ב- ניתן לממש את הפקודה על מעבד מסוג Pipelined RISCV לאחר הכנסת שינויים למסלול הנתונים ו/או לבקר ו/או הגדלת זמן המחזור של השעון.

הסבר: לא נכון, מאותה סיבה כמו בסעיף א'.
קיבלנו גם את התשובה נכון (עם ההסבר הבא **בלבד**): ניתן לממש את הפקודה ע"י הכנסת NOP ע"י המעבד כדי לעקב את הפקודות הבאות, והוספת loop back עבור פקודת dam בשלב ה-MEM.
שימו לב שלמרות שפתרון זה מאפשר לממש את הפקודה dam, הוא **איננו נכון** כי הוא סותר את העקרונות של pipeline מכיוון שבמעבד מצונר כל פקודה צריכה להשתמש בכל stage פעם אחת בלבד, בפתרון זה פקודת dam משתמשת בשלב הזיכרון פעמיים.
(פתרון זה קיבל את כל הנקודות מכיוון שהוא פתרון יצירתי).

כעת נתמקד במעבד $MultiCycle\ RISCV$ שבו **ניתן** לממש את הפקודה.
מבין הטענות הבאות, הקף בעיגול את הטענות הנכונות (**שימו לב, ייתכן שיש יותר מטענה אחת נכונה**).

i. כדי לממש את הפקודה צריך לבצע שינויים במסלול הנתונים (שינוי חיבור חוטים / הוספת חומרה כמו MUX או REGISTERS וכו').
ii. **ניתן** לממש את הפקודה ע"י ביצוע שינויים בבקר כמו הוספת סיגנלי בקרה חדשים, אבל **אין צורך** בהוספת מצבים חדשים לבקר.
iii. כדי לממש את הפקודה חייבים לבצע שינויים בבקר כמו הוספת סיגנלי בקרה חדשים, בנוסף **יש צורך** בהוספת מצבים חדשים לבקר.

צריך להוסיף MUX בכניסה של ה- memory address ולכן גם סיגנלי בקרה, בנוסף צריך להוסיף עוד מצב חדש memory access כי יש שתי גישות לזיכרון שכל אחת מתבצעת במחזור שעון נפרד.
ולכן הטענה הראשונה והשלישית נכונות והטענה השנייה לא נכונה.

שאלה 16 (7 נקודות)

ענו על הסעיפים הבאים לגבי חריגות ופסיקות.

א- תנו דוגמה (אחת) לפסיקה הנגרמת ע"י גורם חיצוני:

שעון, קלט מהמשתמש דרך עכבר, מקלדת, כרטיס רשת ועוד..
טעויות נפוצות: שימו לב שנפילת חשמל אינה פסיקה, זו תקלה ☺

ב- תנו דוגמה (אחת) לחריגה:

חילוק באפס, גלישה ב-ALU, גישה לא חוקית לזיכרון, stack over flow (ממועד א׳ ☺) ועוד..

ג- הסבירו במשפט אחד מהו ההבדל באופן הטיפול בחריגה לעומת הטיפול בפסיקה במעבד מסוג $MultiCycle\ RISCV$.

בחריגה אנחנו עוצרים את הטיפול בפקודה וישר עוברים לטפל בחריגה (להריץ קוד של מערכת ההפעלה). לעומת זאת פסיקות נענות בסיום ביצוע פקודה, קודם כל אנחנו מסיימים להריץ את כל הפקודה ורק אז עוברים למערכת ההפעלה כדי לטפל בפסיקה.