מערכות ספרתיות ומבנה המחשב (044252)

סמסטר חורף תשפ"א

בחינה סופית – מועד ב' – פתרון

7 במרץ 2021

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 ולמחשבון.

הנחיות והוראות:

- יש לענות על הבחינה במערכת המודל בשני חלקים נפרדים - חלק אמריקאי וחלק לשאלות הפתוחות. בנוסף יש לענות על הבחינה כגיבוי המערכת הטפסים של מייקרוסופט - קישור באתר המודל. יש להעלות קובץ המכיל את טיוטת הפתרון לאתר המודל.
- במבחן זה ישנן 15 שאלות. 10 שאלות אמריקאיות, ו- 5 שאלות פתוחות מרובות סעיפים.
- לא מורדות נקודות (אין "קנס") בגין תשובה שגויה. לכן, בשאלות האמריקאיות, כדאי לסמן תשובה כלשהי לכל שאלה.
- בסיומו של המבחן יינתנו 15 דקות לצורך ביצוע סריקה של טיוטת הבוחן.
- אסור שימוש בכל חומר חיצוני מלבד מחשבון ודפי העזר. 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, או אחרת. בפרט, אין להשתמש בטלפון הסלולארי לכל שימוש שאינו צילום מסך המחשב, סריקת המבחנים או התנהלות מול הסגל והמשגיחים.
- עליכם להשאיר את המצלמה אשר מצלמת את פניכם במצב פעיל לאורך כל שלבי הבחינה.
- עליכם להשאיר את המצלמה אשר מצלמת את מסך המחשב במצב פעיל לאורך כל שלבי הבחינה.
- עליכם להשאיר את השמע של המחשב פעיל לכל אורך הבחינה.
- שימוש בטלפון הנייד יתאפשר לצורך וידוא נהלים/פתרון בעיות על ידי משגיח/איש סגל, לצורך ביצוע סריקות של מחברת הבחינה/טיוטה ולצורך צילום מסך המחשב בלבד. כל שימוש אחר בטלפון נייד בזמן הבחינה הוא אסור. בזמן הבחינה מכשיר הטלפון הנייד צריך להיות על מצב רטט.

בהצלחה!

שאלה 1 (5 נקודות)

נתונים שני הרכיבים הבאים:

```
module my_module1 (

      input logic clk,
      input logic rst,
      output logic out1,
      output logic out2

);
      nand(out2, out1, out1);

      always_ff @(posedge clk, posedge rst) begin
            if (rst == 1'b1) begin
                  out1 <= 1'b0;
            end
            else begin
                  out1 <= out2;
            end
      end

endmodule
```

```
module my_module2 (

      input logic clk,
      input logic rst,
      output logic o1,
      output logic o2,
      output logic o3

);
      logic sig;

      my_module1 inst1 (
            .clk(clk),
            .rst(rst),
            .out1(o1),
            .out2(sig)
      );

      my_module1 inst2 (
            .clk(sig),
            .rst(rst),
            .out1(o2),
            .out2(o3)
      );

endmodule
```

בהנחה שהמערכת עברה reset, סמנו את התשובה המתארת בצורה הנכונה ביותר את המוצאים של **my_module2:**

א. o1 – שעון בעל תדר קטן פי 2 מ-clk ; o2 – שעון בעל תדר קטן פי 4 מ-clk.

ב. o1 – שעון בעל תדר קטן פי 4 מ-clk ; o2 – שעון בעל תדר קטן פי 2 מ-clk.

ג. o1 – שעון בעל תדר קטן פי 2 מ-clk ; o3 – שעון בעל תדר קטן פי 8 מ-clk.

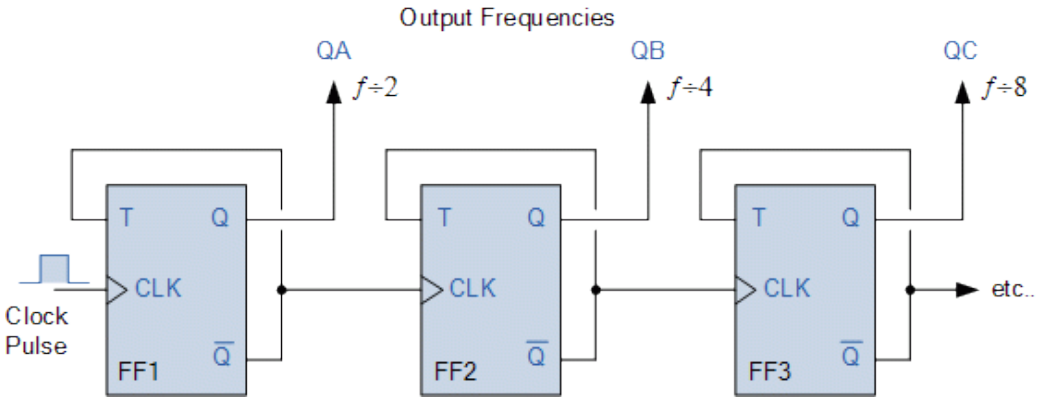
ד. תשובות א' ו-ג' נכונות.

ה. אף תשובה אינה נכונה.

פתרון:

my_module1 ממש FF רגיל בעל מוצא out1 ומוצא out2 שהינו השלילה של out1 (nand(out2,ou1,ou1)) שקול ל-(not(out2,out1). בכל עליית שעון, ה-FF דוגם את המוצא out2 ל-out1 .

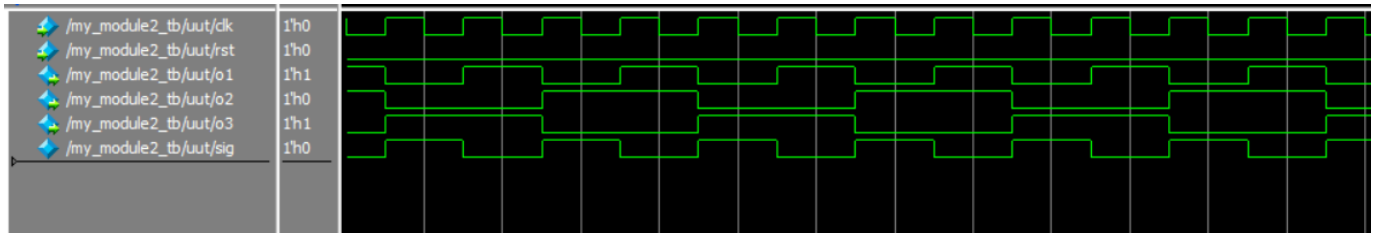
ב- my_module2, ישנם שני מופעים של ה-FF המחוברים כפי שמתואר בשרטוט:



out1 של המופע הראשון הוא שעון בעל תדר שהינו קטן פי 2 מתדר השעון.
out1 של המופע השני הוא שעון בעל תדר שהינו קטן פי 4 מתדר השעון. out2 של המופע השני הוא שעון בעל תדר שהינו קטן פי 4 מתדר השעון, רק שהוא בהפרש פאזה של חצי מחזור שעון.

התשובה הנכונה הינה א'.

סימולציה להדגמה:



שאלה 2 (5 נקודות)

נתונה המשוואה הבאה:

$$(31)_\beta + (\alpha\beta)_9 = (4B)_{\alpha+\beta} + (10)_{10}$$

כאשר הספרות A, B, ... הן הספרות שמגיעות אחרי הספרה 9, כמו בבסיס 16. המספר שרשום בתחתית הסוגריים מציין את הבסיס של המספר שבתוך הסוגריים, למשל המספר $(31)_\beta$ מייצג את הספרות 3,1 בבסיס β.

מבין התשובות הבאות, מה **יכולים** להיות ערכם של α, β?

א. $\alpha = 7, \beta = 8$
ב. $\alpha = 4, \beta = 11$
ג. $\alpha = 7, \beta = 5$
ד. $\alpha = 4, \beta = 8$
ה. יש יותר מתשובה אחת נכונה.

פתרון:

ראשית האילוצים שנובעים מהמשוואה הם:

$$\beta > 3, \qquad \alpha, \beta < 9, \qquad \alpha + \beta > 11$$

פתרון המשוואה הוא:

$$(31)_\beta = 3\beta + 1$$

$$(\alpha\beta)_9 = 9\alpha + \beta$$

$$(4B)_{\alpha+\beta} = 4(\alpha + \beta) + 11$$

$$3\beta + 1 + 9\alpha + \beta = 4(\alpha + \beta) + 11 + 10$$

$$5\alpha = 20 \rightarrow \alpha = 4$$

ולכן קיבלנו את האילוץ הנוסף $\beta > 7$.

ולכן $\alpha = 4, \beta = 8$

שאלה 3 (5 נקודות)

נתונה הפונקציה $\boldsymbol{f(w,x,y,z)}$ אשר מוציאה 1 במידה ושניים או יותר מהקלטים שלה הם 1.

כמו כן נתונה הפונקציה $\boldsymbol{g(w,x,y,z) = \Pi(0,1,3,4,5,9,12,13)}$.

נגדיר את הפונקציה $\boldsymbol{h(w,x,y,z) = nor\big(f(w,x,y,z), g(w,x,y,z)\big) + xz}$

תזכורת: $\boldsymbol{nor(x,y) = \overline{(x+y)}}$.

כאשר מצמצמים את $h(w,x,y,z)$ כסכום מכפלות, אילו מהביטויים הבאים יכולים לתאר את הפונקציה המצומצמת?

א. $h(w,x,y,z) = w'x' + z$

ב. $h(w,x,y,z) = w'x'y' + z$

ג. $h(w,x,y,z) = w'y' + xz$

ד. $h(w,x,y,z) = xz$

ה. $h(w,x,y,z) = w'y' + x$

פתרון:

אנו יודעים שהביטוי $\boldsymbol{nor(x,y) = 0}$ במידה ואחד ממשתני הכניסה שלה (או שניהם) שווה 1.
נשים לב שהפונקציה $\boldsymbol{f(w,x,y,z)}$ תוציא 0 עבור 5 קלטים בלבד.
(0000,0001,0010,0100,1000), כלומר עבור כל כניסה אחרת היא תוציא 1.
נבדוק עבור אילו מהכניסות הללו (עבורן $\mathbf{f(w,x,y,z) = 0}$) נקבל כי

$\mathbf{g(w,x,y,z) = 1}$ (עבור כל כניסה אחרת אנו יודעים כי תוצאת ה $\boldsymbol{nor}$ היא 0.
נשים לב שהכניסות עבורן $\boldsymbol{f(w,x,y,z) = \Pi(0,1,2,4,8)}$. נשים לב שהכניסות היחידות עבורן $nor = 0$ הן (0000,0001,0100). נשים את הביטויים במפת קרנו.

yz \ wx	00	01	11	10
00	1	1	0	0
01	1	1	1	0
11	0	1	1	0
10	0	0	0	0

כלומר קיבלנו $h(w,x,y,z) = w'y' + xz$.

שאלה 4 (5 נקודות)

נתונים שני מספרים בינאריים $a = a_1a_2,\ b = b_1b_2$. המספרים בני 2 סיביות כאשר סיבית ה- MSB היא סיבית סימן בשיטת המשלים ל- 2. בנוסף נתונה פונקציה $f(a_1, a_2, b_1, b_2)$ המוציאה '1' אם a < b אחרת '0'.

בחר בתשובה הנכונה ביותר:

א. הפונקציה f הינה מערכת פעולות שלמה.
ב. הפונקציה f אינה מערכת פעולות שלמה, אבל בתוספת הקבוע '1' היא מערכת פעולות שלמה.
ג. הפונקציה f אינה מערכת פעולות שלמה, אבל בתוספת הקבוע '0' היא מערכת פעולות שלמה.
ד. התשובות ב' ו-ג' נכונות.
ה. התשובות א' – ד' אינן נכונות.

פתרון:

a1a2 b1b2	00 (0)	01 (1)	11 (-1)	10 (-2)
00 (0)			1	1
01 (1)	1		1	1
11 (-1)				1
10 (-2)				

והתשובה המתקבלת היא:

$$f(a1, a2, b1, b2) = a_1\overline{b_1} + a_1\overline{a_2}b_2 + \overline{b_1}b_2a_2$$

הפונקציה אינה מקיימת את התנאי מההרצאה:

$$f(x, x, x, x) = 0$$

לכן איננה מערכת פעולות שלמה, גם לא בתוספת הקבוע 0.

אם נקבל את הקבוע '1' לעומת זאת אז:

$$f(1,1,x,1) = 1 \cdot \bar{x} + 0 + \bar{x} \cdot 1 = \bar{x}$$

נממש שער and:

$$f(x,1,\bar{y},0) = xy + 0 + 0 = xy$$

כשאת הקבוע '0' קיבלנו מ- $f(x,x,x,x) = 0$.

שאלה 5 (5 נקודות)

נתון קטע הקוד הבא שמטרתו חישוב עצרת של מספר. קטע הקוד מתחיל לרוץ מ-*main*:

הערה: שימו לב שמימוש זה שונה מהמימוש אשר הוצג בתרגול.

```
0x0CCD 0000    main:     addi sp, sp, -4
0x0CCD 0004              sw ra, 0(sp)
0x0CCD 0008              addi a0, x0, 10
0x0CCD 000C              addi a1, a0, -1
0x0CCD 0010              jal factorial
0x0CCD 0014              …(more code)

0x1AA0 0014    factorial: mul a0, a0, a1
0x1AA0 0018              addi a1, a1, -1
0x1AA0 001C              beq a1, x0, exit
0x1AA0 0020              jal factorial
0x1AA0 0024    exit:     lw ra, 0(sp)
0x1AA0 0028              jr ra
```

הפקודה mul rd,rs1,rs2 מבצעת כפל בין שני הרגיסטרים, rs1,rs2 ומאחסנת את התוצאה ברגיסטר rd.

נגדיר פונקציה כתקינה במידה והיא עומדת בכללי הקריאה לפונקציה אשר נלמדו בקורס.

האם הפונקציה factorial תקינה, ומה הערך של a0 לאחר ביצוע הקפיצה בשורה 0x1AA0 0028 בפעם האחרונה?

א. הפונקציה factorial תקינה, ו- $a0 = 10!$.

ב. הפונקציה factorial תקינה, ו- $a0 \neq 10!$.

ג. הפונקציה factorial אינה תקינה, ו- $a0 = 10!$.

ד. הפונקציה factorial אינה תקינה, ו- $a0 \neq 10!$.

ה. הפונקציה factorial לעולם לא תפסיק את ריצתה.

פתרון:

תשובה ג'.

הפונקציה אכן מחשבת עצרת. בכל כניסה לפונקציה אנו מבצעים $\boldsymbol{a0} = \boldsymbol{a0} * \boldsymbol{a1}$.

ולאחר מכן מבצעים $\boldsymbol{a1} = \boldsymbol{a1} - \mathbf{1}$. לאחר שורה זו אנו בודקים את תנאי העצירה $\boldsymbol{a1} == \mathbf{0}$. במידה והתנאי מתקיים אנו מסיימים את ריצתנו. נשים לב שבעצם כפלנו את **a0** בכל המספרים שקטנים ממנו (פעולת עצרת), כאשר בפעם בה ביצענו את הכפל עם הערך 1 סיימנו את ריצת הפונקציה (קפצנו ל ***exit***).
עם זאת נשים לב שהפונקציה אינה עומדת בקונבנציית הקריאה לפונקציה אשר נלמדה בכיתה (אינה שומרת את ***ra*** למרות שהיא משנה אותו). ובסיום ריצתה הפונקציה לא תחזור למקום ממנו היא נקראה.

שאלה 6 (5 נקודות)

בתהליך הצמצום של טבלת המצבים של מערכת עקיבה סינכרונית שיש לה יציאה של ביט יחיד, התקבלה השורה הבאה:

$$P2 = (A)(B)(C)(DEFG)(HIJK)$$

נתונות הטענות הבאות:

A. אם למערכת כניסה של ביט יחיד אז המערכת היא בהכרח מסוג Moore.

B. אם למערכת כניסה של 2 ביטים אז המערכת היא בהכרח מסוג Mealy.

C. אם $P_1 = P_2$, אז למערכת כניסה של 2 ביטים או יותר.

D. אם $P_2 \neq P_3$, אז למערכת כניסה של 2 ביטים או יותר.

בחרו את התשובה הנכונה:

א. טענה A נכונה

ב. טענה B נכונה

ג. טענה C נכונה

ד. טענות A,D נכונות

ה. טענות B,C נכונות

פתרון:

התשובה הנכונה היא ג'.

טענה A: המערכת יכולה להיות Mealy.
עבור מכונת mealy עם ביט יציאה יחיד ב p1 יכולים להיות עד 4 מחלקות, וב-p2 מספיקה חלוקה נוספת.

טענה B: ניתן עבור mealy ו moore

טענה C: אין אפשרות להגיע ל p1 עם 5 מחלקות באמצעות כניסה אחת.

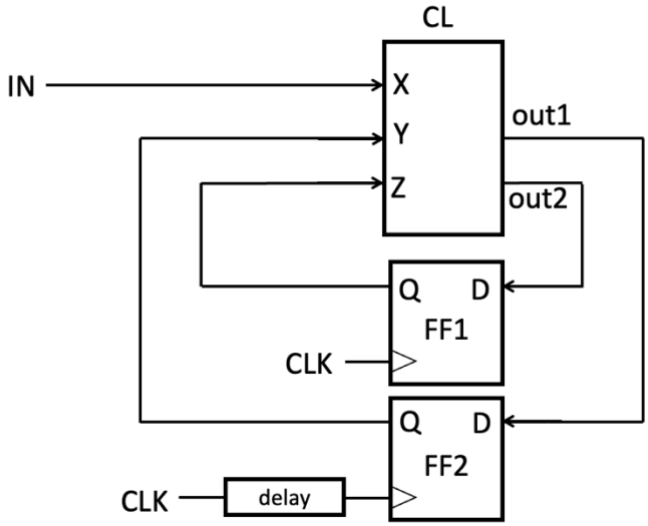
טענה D: אם p2, p3 שונים, אז ב p1 יש לכל היותר 4 מחלקות. אפשר להגיע ל p1 עם כניסה אחת mealy.

למשל

	x=0		x = 1	
	NS	out	NS	out
A		1		0
B		1		1
C		0		1
D		0		0
E		0		0
F		0		0
G		0		0
H		0		0
I		0		0
J		0		0
K		0		0

שאלה 7 (5 נקודות)

סטודנט בקורס הרכיב את המעגל הבא שמממש מערכת עקיבה עם קלט יחיד, וללא פלט מסוג *MOORE*. שימו לב למיקום הקלט (D) והפלט (Q) של רכיבי ה- *FF*. בנוסף, הרכיב *CL* הינו רכיב צירופי.



לרכיב *CL* יש 3 כניסות (2 עבור המצב הנוכחי, ואחד עבור הקלט) ו- 2 פלטים (עבור המצב הבא).

הסטודנט גילה שיש תקלה במעגל משום שהשעון שמוזן ל- *FF2* מגיע עם $\boldsymbol{delay = 3ns}$.

נתון שהתקלה לא פוגעת בתנאי *hold* של שני הרגיסטרים.

נתונים:

	tpd/tpcq	tsetup
FF1/FF2	10ns	3ns
$CL(X-> out_i)$	4ns	
$CL(Y-> out_i)$	5ns	
$CL(Z-> out_i)$	6ns	

לסטודנט יש שליטה אך ורק על זמן המחזור של המעגל ועל הזמן שבו קלט המעגל משתנה. כאשר זמן השינוי של הקלט t_{IN} נמדד מעליית השעון *CLK*.

<u>מבין התשובות הבאות</u>, מה התנאי ההדוק ביותר על t_{IN} שבו נוכל לשנות את הקלט, כך שבוודאות לא נפגע בתפקוד המעגל?

א. $t_{IN} \leq 4ns$
ב. $t_{IN} \leq 9ns$
ג. $t_{IN} \leq 12ns$
ד. $t_{IN} \leq 14ns$

ה. $t_{IN} \leq 17ns$

פתרון:

תשובה ד'.

נתחיל מהמסלול *FF1->FF2*:

$$t_{pCQ}(FF1) + t_{pd}(CL: Z \rightarrow out_1) + t_{setup}(FF2) \leq T + t_{delay}$$

$$10ns + 6ns + 3ns \leq T + 3ns$$

$$16ns \leq T$$

מסלול *FF2->FF2*:

$$t_{pCQ}(FF2) + t_{pd}(CL: Y \rightarrow out_1) + t_{setup}(FF2) \leq T$$

$$10ns + 5ns + 3ns = 18ns \leq T$$

מסלול *FF2->FF1*:

$$t_{delay} + t_{pCQ}(FF2) + t_{pd}(CL: Y \rightarrow out_2) + t_{setup}(FF1) \leq T$$

$$3ns + 10ns + 5ns + 3ns = 21ns \leq T$$

מסלול *FF1->FF1*:

$$t_{pCQ}(FF1) + t_{pd}(CL: Z \rightarrow out_2) + t_{setup}(FF1) \leq T$$

$$10ns + 6ns + 3ns = 19ns \leq T$$

מסלול *IN->FF2*:

$$t_{IN} + tpd(CL: X \rightarrow out_1) + t_{setup}(FF2) \leq T + t_{delay}$$

$$t_{IN} + 4ns + 3ns \leq T + 3ns$$

$$t_{IN} \leq T - 4ns$$

מסלול *IN->FF2*:

$$t_{IN} + tpd(CL: X \rightarrow out_2) + t_{setup}(FF1) \leq T$$

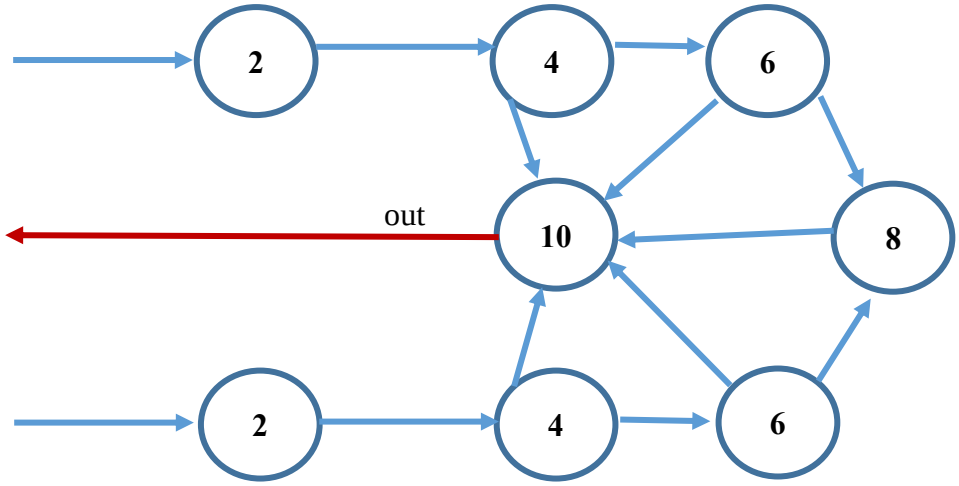
$$t_{IN} + 4ns + 3ns \leq T$$

$$t_{IN} \leq T - 7ns$$

קיבלנו את התנאי $t_{IN} \leq 14ns$.

שאלה 8 (5 נקודות):

נתונה המערכת הבאה:



זמני ההשהיות של הרכיבים נתונים ב- *ns* ורשומים בתוך הרכיבים.
הניחו רגיסטרים אידיאליים.

נרצה לצנר את המערכת כדי לקבל *throughput* מקסימלי בעדיפות גבוהה, *latency* מינימלי בעדיפות שניה, ומספר רגיסטרים מינימלי בעדיפות שלישית.

לאחר הצינור, מהו ה- *latency* שיתקבל, ומהו מספר הרגיסטרים שנצטרך לצינור המערכת?

א. $latency = 40ns, regs = 14$

ב. $latency = 40ns, regs = 16$

ג. $latency = 50ns, regs = 15$

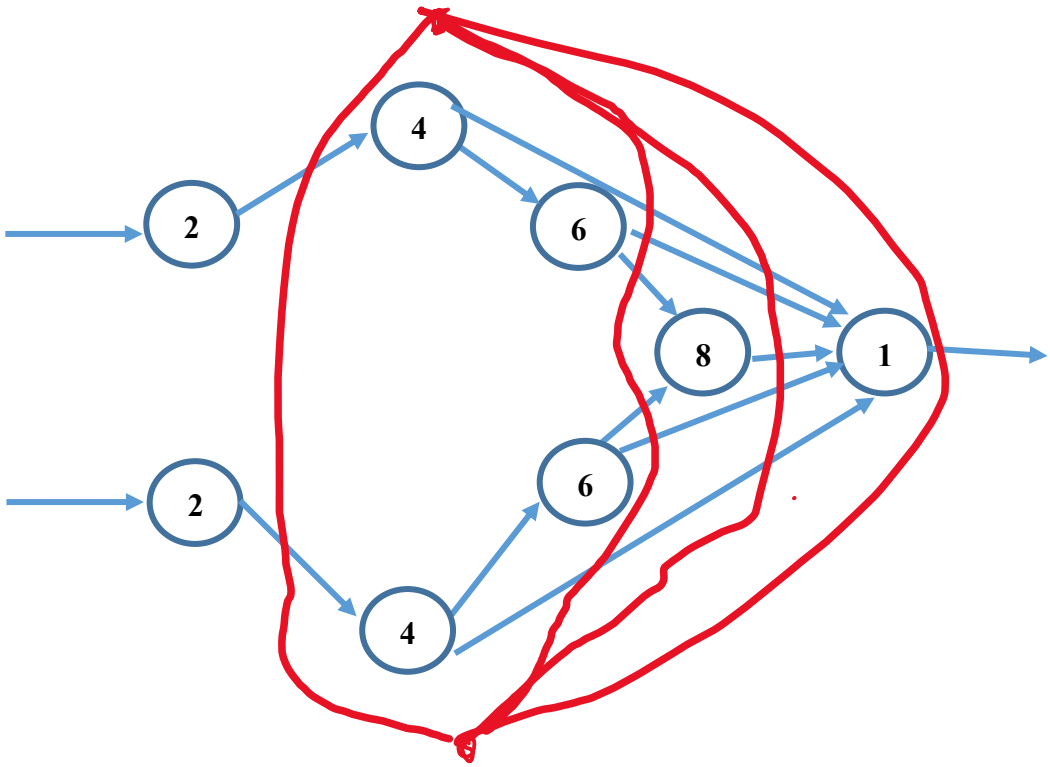
ד. $latency = 50ns, regs = 18$

ה. $latency = 60ns, regs = 19$

פתרון:

תשובה א'.

נצנר את המערכת:



שאלה 9 (5 נקודות):

נתון מעבד *Single-cycle RISC-V* כפי שנלמד בקורס.

בנוסף נתונים תזמוני חלקי המעבד באופן הבא:

	Timing
Memory access (instruction or data)	$3ns$
Read a value from the register file	$1ns$
ALU operation	$2ns$
Write a value to the register file	$2ns$

רכיבי המעבד אשר אינם מוזכרים בטבלה הינם בעלי זמני השהייה זניחים.

נתונה התוכנית הבאה, שמתחילה בפקודה שבכתובת *0x00001000*. הניחו כי התוכנית רצה כהלכה וללא חריגות או פסיקות.

	$0x00001000$	$addi\ t2, x0, 4$
	$0x00001004$	$addi\ t1, t3, 6$
	$0x00001008$	$lw\ \ s3, 0(s1)$
$Loop$:	$0x0000100C$	$addi\ t2, t2, -1$
	$0x00001010$	$add\ \ t1, t2, x0$
	$0x00001014$	$sub\ \ s4, s3, s4$
	$0x00001018$	$bne\ t2, x0, Loop$
$Exit$:	$0x0000101C$	NOP

מהו משך הזמן אשר יידרש להרצת התוכנית (עד לסיום הפקודה בכתובת $0x00001018$ בפעם האחרונה)?

א. $152ns$

ב. $190ns$

ג. $209ns$

ד. $228ns$

ה. $264ns$

פתרון:

תחילה נמצא את זמן המחזור של המעבד:

$$T_{cycle} = Mem_{fetch} + RegRead_{dec} + ALU + Mem_{data} + RegWrite_{wb}$$

$$T_{cycle} = 3 + 1 + 2 + 3 + 2 = 11ns$$

כיוון שזהו מעבד *Single-cycle*, כל על מה שנותר הוא לספור את מספר הפקודות שמריצים

ולהכפיל מספר זה בזמן המחזור. הלולאה רצה 4 פעמים, כאשר בכל איטרציה ישנן 4 פקודות. מחוץ לולאה ישנן 3 פקודות נוספות. בסה"כ קיבלנו כי ישנן $4 \cdot 4 + 3 = 19$ פקודות אשר מריצים. לכן זמן הריצה הכולל יהיה:

$$19 \cdot 11 = 209ns$$

<u>שאלה 10 (5 נקודות):</u>

שירי החליטה להתנדב לקופת חולים כדי לעזור לחסן את האוכלוסייה על מנת שסמסטר אביב יהיה פרונטלי.

כידוע, כל אדם צריך לקבל 2 מנות חיסון בהפרש של **<u>21 ימים</u>**.
מערכת קביעת התורים ממודלת במעבד *Multicycle RISCV* כך שכל שורה בזיכרון מתארת יום והמידע בזיכרון הוא ת"ז המתחסן.

על מנת לייעל את תהליך קביעת התורים שירי הציעה להוסיף פקודה למעבד שמקבלת את ת"ז המתחסן וקובעת לו באופן אוטומטי 2 תורים בהפרש של **21 ימים**.

הנחות:

- הניחו כי בכל יום ניתן לחסן רק אדם אחד (בשורה אחת בזיכרון יש מספר ת"ז אחד בלבד).
- הניחו כי שורות היעד בזיכרון תמיד פנויות (תאריך החיסון הראשון והשני).
- הניחו כי מספר ת"ז של אדם מורכב מ- 11 סיביות.
- הניחו כי כל שורה בזיכרון מורכבת מ- 4 בתים.

פורמט הפקודה:

$$Vacc\ rs, ID$$

כאשר rs זהו רגיסטר המכיל את הכתובת בזיכרון שמתארת את תאריך המנה הראשונה.

ID - ערך *imm* המתאר את ת"ז המתחסן.

למשל: בהינתן שב- *t0* שמור הערך 0. כאשר תרוץ הפקודה *vacc t0, 123*, ירשם בזיכרון בשורות 0 ו- 21 הערך 123.

מה מהבאים נכון לגבי הוספת הפקודה? (כאשר מדובר על הוספת רכיבים, הכוונה היא ב- *Datapath* בלבד).

א. ניתן לממש את הפקודה ע"י הוספת 4 מצבים לבקר בלבד ללא שינוי ה- *Datapath*.

ב. ניתן לממש את הפקודה ע"י הוספת *mux*, ו2 מצבים לבקר בלבד.

ג. ניתן לממש את הפקודה ע"י הוספת 2 מצבים לבקר, הוספת ערך קבוע לאחד ה- *mux* ב- *Datapath*, והוספת *mux*.

ד. ניתן לממש את הפקודה ע"י הוספת מצב חדש לבקר, הוספת *mux* והוספת סיגנל.

ה. ניתן לממש את הפקודה ע"י הוספת 3 מצבים לבקר, הוספת ערך קבוע לאחד ה- *mux* ב- *Datapath*, והוספת *mux*.

פתרון:

תשובה ה.

השינויים הנדרשים ב*DP*:

- הוספת ערך קבוע "14" על מנת שנוכל לקדם את הכתובת בזיכרון.
- הוספת *mux* בכניסה ל*dataW* שיבחר בין ערך ה*imm* לבין *regB*.

השינויים הנדרשים בבקר:

- הוספת 3 מצבים חדשים
$EXE \rightarrow MEM\_EXE \rightarrow MEM$

הסבר:

נשים לב שמצב ה*EXE* של *SW/LW* אנחנו מחברים את ערך ה*REG* הנתון ל*imm*. במקרה שלנו אנחנו לא רוצים לעשות את זה ולקחת את ערך הרגיסטר ככתובת לכן אנחנו לא יכולים "לרכב" על מצב 2.

במצב הבא, אנחנו ניגשים לכתוב בזיכרון את תז המתחסן ותוך כדי משתמשים ב*ALU* כדי לחשב את הכתובת הבאה.

במצב האחרון, אנחנו כותבים את תז המתחסן בכתובת שחישבנו במחזור הקודם.

שאלה 11 (5 נקודות):

נתונים 2 מעבדי pipeline RISC-V, ללא forwarding וללא hazard detection unit. המעבדים מניחים כי פקודות branch לא קופצות ובמידה וכן מבצעים flush:

- מעבד *A*: זמן מחזור *T*, ההחלטה על הקפיצה מתבצעת בשלב *Execute*.
- מעבד *B*: זמן מחזור *0.8T*, ההחלטה על הקפיצה מתבצעת בשלב *Memory*.

נתונה תוכנית ארוכה מאוד אשר יחס פקודות ה-*branch* הנכנסות למעבד בפועל בהרצתה הוא *X*.
כלומר, עבור תוכנית שבהרצתה מבוצעות 1000 פקודות ועבור $X = 0.5$, 500 פקודות הינן פקודות *branch*. כמו כן נתון כי בתוכנית הנ"ל לא קיימים *load hazards*, ואין *data hazards*.
עבור התוכנית 70% מכל פקודות ה-*branch* שנכנסות למעבד קופצות.

עבור איזה טווח ערכים של *X* **מבין הערכים הבאים** ביצועיו של מעבד *A* טובים יותר (ביצוע של פקודות ליחידת זמן) מאלה של *B*? (שימו לב לכיוון האי-שוויון)

א. *X>0.2*

ב. *X<0.2*

ג. *X>0.5*

ד. *X<0.5*

ה. *X>0.8*

פתרון:

סעיף ה'

ביצועי *A* טובים יותר, *N* פקודות:

$$T * N * (1 + 2 * 0.7X) < 0.8 * T * N * (1 + 3 * 0.7X)$$

$$1 + 1.4X < 0.8 * (1 + 2.1X)$$

$$0.2 < 0.28X$$

$$0.714 < X$$

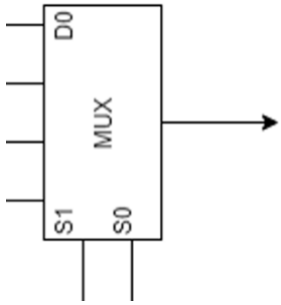
<u>שאלה 12 (8 נקודות):</u>

נתונות הפונקציות הבאות:

$$h(w,x,y,z) = \bar{x}\bar{z} + \bar{x}\bar{y} + wz$$
$$g(w,x,y,z) = \Pi(0,4,6,8,9,12,14)$$

כמו כן, נכון כי $f(w,x,y,z) = g \cdot h$

האם ניתן לממש את f באמצעות בורר <u>יחיד</u> $4 \rightarrow 1$ ושער xor אחד בלבד?
אם לא, רשום "לא ניתן". אם ניתן, צייר מימוש של f באמצעות הנתון בציור מטה (כאשר S0,D0 הן סיביות ה- LSB של כניסות הנתונים והבקרה בהתאמה).



פתרון:

g=

y, z \ w, x			
	1	1	1
	1	1	1
	1		1

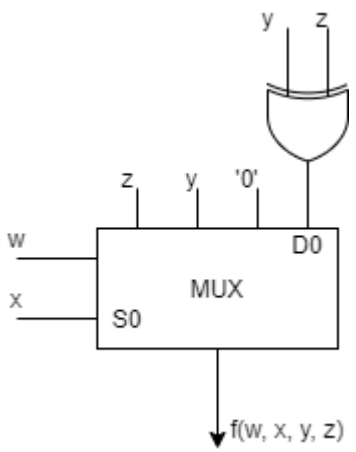
h=

y, z \ w, x			
1			1
1		1	1
		1	1
1			1

f=

y, z \ w, x			
1		1	
		1	1
1			1

ס\ניתן. מימוש:



שאלה 13 (12 נקודות)

נגדיר פרוטוקול תקשורת חדש נסמנו B, אשר משדר שידורים ברצף.

בפרוטוקול B נשלחות 16 סיביות מידע, סיבית start ו- stop, ובנוסף בפרוטוקול B גילו כי שגיאות יכולות להתרחש (רק בסיביות המידע), לכן החליטו לשפר את אמינות הפרוטוקול ע"י הוספת סיבית זוגיות לפני ה- stop bit. סיבית הזוגיות בודקת את זוגיות 16 סיביות המידע באותו שידור (סה"כ 19 סיביות בשידור כולל start ו stop).
בכל שידור בפרוטוקול B יכולה להתרחש עד שגיאה אחת בביט יחיד מבין 16 הביטים של המידע. אם המקלט זיהה שהתרחשה שגיאה, הוא יכול לבקש מהמשדר לשלוח שוב את השידור הלא תקין (לאחר שהמשדר יסיים את השידור שהוא כרגע שולח). בשידור השני של אותו מידע מובטח שלא תתרחש תקלה.

E מוגדר כסיכוי לכישלון שידור ($E \in [0,1]$) בפרוטוקול B (כאשר E=0 אומר שהשידור משודר תמיד בהצלחה).

<u>סעיף א'</u>

מלאו בטבלה הבאה מה יהיה היחס בין מספר סיביות **<u>המידע</u>** המשודרות בפרוטוקול B מבין כלל הסיביות המשודרות בפרוטוקול זה. שימו לב שלאחר שידור כושל, השידור הבא בהכרח מוצלח.

	$E = 0$	$E = 1$	$E = \alpha$
יחס			

עבור סעיפים ב' וג':

כעת נתון פרוטוקול C שמבוסס על פרוטוקול B, אך כעת שולחים את 16 סיביות המידע ואת סיבית הזוגיות פעמיים. סה"כ 36 סיביות בשידור כולל start ו- stop. בכל שידור יכולה להתרחש עד שגיאה אחת בביט יחיד מבין כל סיביות המידע או זוגיות. בנוסף אם הייתה תקלה, שולחים את השידור שוב רק אם בלתי אפשרי לשחזרו.

סעיף ב'

מהו מרחק הקוד של סיביות המידע והזוגיות בפרוטוקול C (לא נתון כל פרט על ערכי סיביות מידע)?

סעיף ג'

מהם הערכים של E שעבורם קצב השידור של מידע תקין בפרוטוקול C גבוה יותר מפרוטוקול B?

פתרון:

סעיף א':

	$E=0$	$E=1$	$E=\alpha$
יחס	$16/19$	$16/38$	$\frac{16}{19(1+\alpha)}$

א. קצב שידור מידע פרוטוקול A:

8 סיביות מידע בכל 10 מחזורים. $\frac{8}{10}$

קצב שידור מידע פרוטוקול B:

בשידור תקין: 16 סיביות מידע בכל 19 מחזורים

$$\frac{16}{19+19E}$$

$$\frac{16}{19+19E} > \frac{8}{10}$$

$$20 > 19+19E$$

$$\frac{1}{19} > E$$

ב. מרחק קוד 4.

ג. כעת ניתן לשחזר את השגיאה ואין צורך לשלוח את המידע שוב.

קצב שידור מידע פרוטוקול B: $\frac{16}{19+19E}$

קצב שידור מידע פרוטוקול C: $\frac{16}{36}$

$$\frac{16}{19+19E} < \frac{16}{36}$$

$$36 < 19+19E$$

$$\frac{17}{19} < E$$

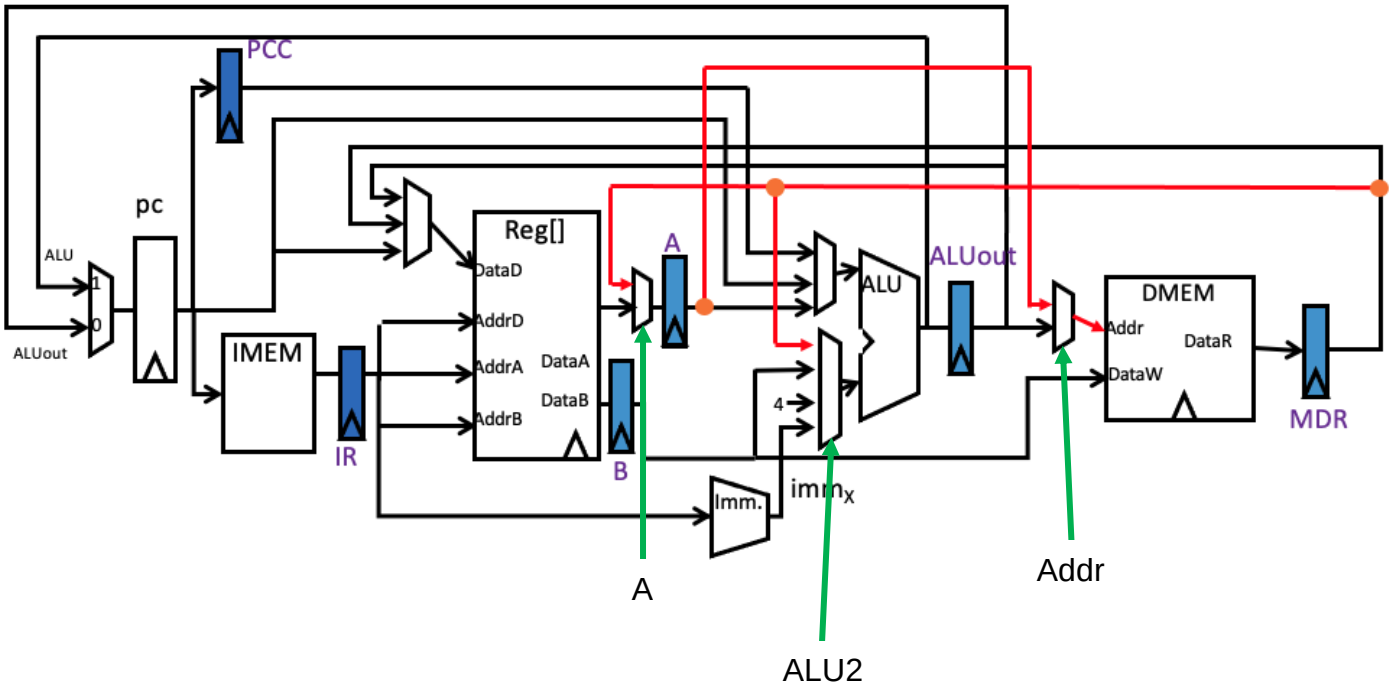
שאלה 14 (10 נקודות):

נתון מעבד מסוג *Multi-Cycle RISC-V* כפי שנלמד בקורס. סטודנט בקורס מעוניין לממש פקודה שתבצע את הפעולות הבאות:

AddMem rd, rs1, imm

Reg[rd] = Mem[Reg[rs1]] + Mem[Reg[rs1]+imm]

לצורך ביצוע הפקודה, הסטודנט שינה את ה- *Datapath* של המעבד באופן הבא:



שימו לב לשינויים המודגשים באדום:

1. נוסף בורר הכניסה לרגיסטר *A*, המקבל בנוסף את המידע שיוצא מרגיסטר *MDR*.
2. נוסף בורר בכניסת ה-*addr* של ה- *DMEM*, המקבל בנוסף את יציאת רגיסטר *A*.
3. נוספה כניסה לבורר שמחובר לקלט התחתון של ה- *ALU*. כניסה זו מקבלת את פלט ה- *MDR*.

מלאו את ערכי הבקרה של הבוררים המסומנים (הניחו כי כניסת המידע התחתונה של כל בורר היא כניסה מספר 0 וכי מספרי הכניסות העליונות עולים בהתאמה). שימו לב כי כל עמודה מייצגת מצב וכי לאו דווקא ישנו צורך לבצע שימוש בכל העמודות. במידה וערך קו הבקרה יכול להיות 0 או 1 יש לסמן *dc*.

Name	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
A	*0*		*dc*					
ALU2	*1*				*dc*			
Addr		*dc*		*0*				

פתרון:

שני המחזורים הראשונים יבצעו *Fetch, Decode* ללא שינוי.

במחזור השלישי רגיסטר *rs1* יעבור לכניסת ה- *addr* של ה- *DMEM*, ויקרא את תוכן הזיכרון שישמר ברגיסטר *MDR*. במקביל יחושב *Reg[rs1]+imm* ב- *ALU* והמידע ישמר ב- *ALUout*.

במחזור הרביעי המידע שנקרא מהזיכרון *Mem[Reg[rs1]]* ישמר ברגיסטר *A*, ובמקביל יקרא המידע *Mem[Reg[rs1+imm]]* וישמר ברגיסטר *MDR*. כלומר בסוף מחזור זה יהיה לנו את המידע הנדרש משתי כתובות הזיכרון.

במחזור החמישי נחשב את סכום הערכים שמגיעים מ- *MDR*, ומ- *A* ונשמור ב- *ALUout*.

במחזור השישי נשמור את המידע ב- *rd*.

Name	*1*	*2*	*3*	*4*	*5*	*6*
A	*0*	*0*	*dc*	1	*dc*	*dc*
ALU2	*1*	*0*	0	*dc*	3	*dc*
Addr	*dc*	*dc*	1	0	*dc*	*dc*

שאלה 15 (10 נקודות)

נתון מעבד Multi Cycle RISC-V אשר תומך במנגנון טיפול בחריגות כפי שנלמד בכיתה (שיטת קוד הגורם לחריגה). כמו כן נתון כי בשאלה זו ניתן לגשת לרגיסטרים *SCAUSE, SEPC*.

סטודנט חרוץ החליט לבדוק את מנגנון הטיפול בחריגות, לשם כך הוא כתב את קטע הקוד הבא אשר מתחיל לרוץ מ-*main* :

0x0CCD 0000	main:	addi x0, x0, 1
0x0CCD 0004		slli s1, 32(x0)
0x0CCD 0008		addi s2, x0, 2
0x0CCD 000C		slli s2, 31(s2)
0x0CCD 0010		div s3, s2, s1
0x0CCD 0014		…(more code)

0x1C09 0000	Interrupt handler:	addi t0, x0, 1
0x1C09 0004		slli t1, t0, 1
0x1C09 0008		slli t2, t1, 1
0x1C09 000C		beq *SCAUSE* , t0, exception1
0x1C09 0010		beq *SCAUSE* , t1, exception2
0x1C09 0014		beq *SCAUSE* , t2, exception3
0x1C09 0018	Exit:	jr SEPC
0x1C09 001C		…(more code)

הפקודה div rd,rs1,rs2 מבצעת חלוקה כך שמתקיים:

$$\mathrm{rd} = \mathrm{rs1}/\mathrm{rs2}$$

כמו כן ידוע כי רגיסטר SCAUSE לא יכול לקבל את הערך 3 לעולם.

<u>סעיף א':</u>

ציינו את שיטת קידוד קוד הפסיקות הנהוגה במעבד הנתון.

להזכירכם השיטות שנלמדו הן: קידוד חזקות של 2, ו- מספר רץ.

<u>סעיף ב':</u>

ציינו יתרון אחד וחיסרון אחד של השיטה:

<u>סעיף ג':</u>

ציינו את החריגות אשר מתרחשות בקוד ובאילו כתובות הן מתרחשות:

<u>פתרון:</u>

השיטה הנהוגה היא שיטת קידוד חזקות.
יתרונות:

-מאפשר אינדיקציה על פסיקות במקביל.

-חוסך בחומרה (פחות בוררים).

חסרונות:

-מוגבל לכמות פסיקות כגודל רגיסטר הפסיקות.

-מקשה על שגרת הטיפול בפסיקה (ברירה של סיביות קשה יותר מהשוואת ערך - הקוד ברגיסטר השלם).
קיימות שתי חריגות המופיעות בקוד:
חריגת גלישה המופיעה בכתובת ***0x0CCD 000C***

slli s2, 31(s2)

חריגה נוספת המופיעה בקוד היא חלוקה באפס המופיעה בכתובת

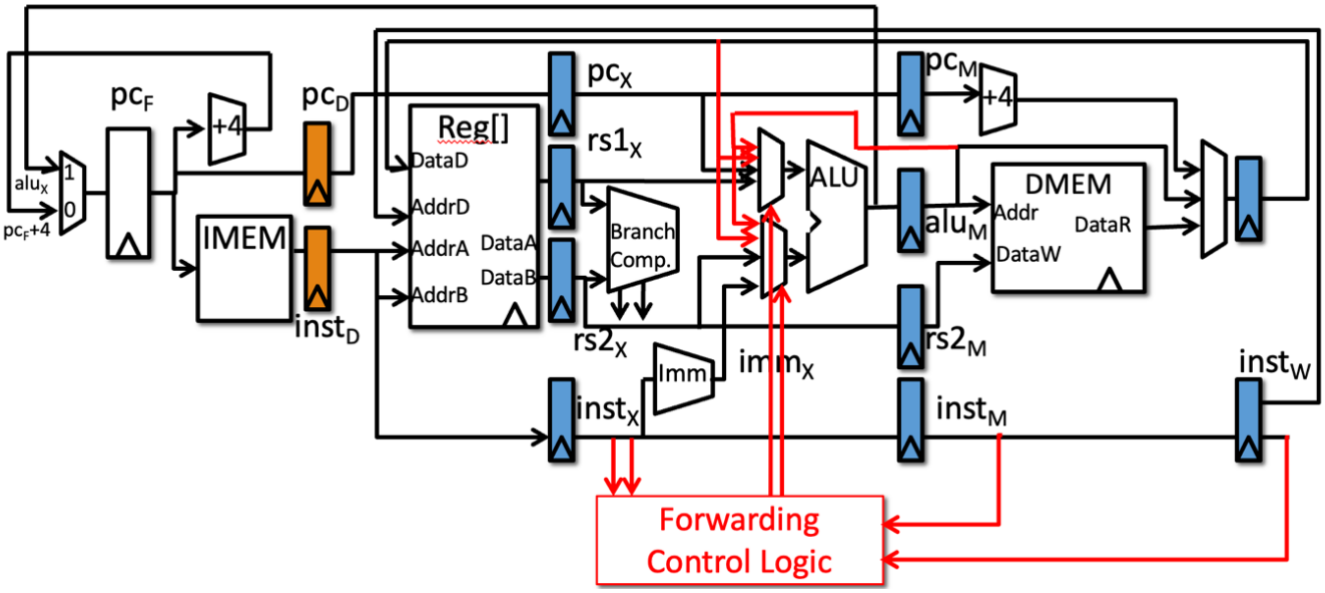
0x0CCD 0010

div s3, s2, s1

שאלה 16 (10 נקודות)

נתון מעבד RISC-V Pipeline, בעל התכונות הבאות:

- מכיל Forwarding ברכיב ה- Register File (Decode <- WB).
- מכיל רכיב Hazard Detection Unit.
- מכיל רכיב Forwarding Unit **לפי האיור להלן** בין שלבי ה- WB וה- MEM לשלב ה- Exe.
- מניח ש- Branch אינו נלקח, ומבצע Flush במידה וכן. חישוב הקפיצה מתבצע בשלב ה- Exe



סטודנט מעוניין להריץ את הקוד הבא, המתחיל מ- main:
<u>הערה:</u> הערכים הכתובים בכל הרגיסטרים לפני ריצת הקוד אינם ידועים.

```
Main:   add t0, s0, x0
        add t1, s1, x0
        beq t0, t1, label
        lw t0, 0(s0)
        addi t3, t0, 0
label:  sw t0 0(t4)
```

<u>סעיף א':</u>
מלאו בטבלה הבאה כמה פקודות NOP הסטודנט שכתב את הקוד יצטרך להוסיף, ובאיזה מיקום (אחרי איזה שורה, ולפני איזו שורה).
למשל, אם לדעתכם צריך להוסיף 2 פקודות NOP בין שורה 1 ל- 2, אז מלאו בטבלה הבאה: כמות NOP: 2, אחרי שורה: 1, לפני שורה: 2.

כמות NOP	אחרי שורה	לפני שורה

<u>סעיף ב':</u>
כעת כדי לצמצם את מספר פקודות ה- NOP שצריך להוסיף, הסטודנט החליף את המעבד במעבד זהה, **<u>אך עם Forwarding מלא</u>** (בין השלבים WB -> EXE, EXE<-MEM ו- Decode<-WB).
להזכירכם Forwarding מלא בין שלב ה- WB לשלב ה- Mem למשל אומר שניתן להעביר ערך רצוי מתחילת שלב ה- WB לכל מקום דרוש בשלב ה- EXE.
בסעיף זה נתונים לנו גם זמני ריצת שלבי המעבד:

Fetch	Decode	Execute	Memory	WB
200ns	100ns	200ns	250ns	50ns

נתון בנוסף שערכי הרגיסטרים לפני ריצת הקוד הינם 0.
מה יהיה זמן ריצת הקוד כולו?

פתרון:

סעיף א'-

אין Forwarding לכניסות ה- Branch Compare, ולכן נצטרך להוסיף NOP כך שה- Forwarding יקרה בין שלב ה- WB לשלב ה- Decode.

```
1    Main:   add t0, s0, x0
2            add t1, s1, x0
             2 x NOP
3            beq t0, t1, label
4            lw t0, 0(s0)
5            addi t3, t0, 0
6    label:  sw t0 0(t4)
```

אין צורך להוסיף NOP בין פקודות 4 ל- 6 בגלל שה- Hazard Detection Unit יוסיף שם NOP ולמעשה כאשר ה- LW יהיה ב- WB, פקודת ה- SW תהיה ב- Decode וה- Forwarding הזה נתון לנו.

סעיף ב' –

הפעם בגלל שנתון לנו Forwarding מלא לא צריך להוסיף את פקודות ה- NOP כמו בסעיף א'. זמן המחזור הינו 250ns.

מספר המחזורים הינו: 4 עבור מילוי ראשוני של הצינור, לאחר מכן 3 פקודות עד פקודת BEQ, 2 פקודות נוספות יטענו לצינור וימחקו (שימו לב שהצינור עדיין לא יוסיף bubble אחרי ה-lw משום שזה קורה כשה- lw מגיע לשלב ה- exe, אך כאן זה לא מתרחש). לבסוף נריץ את פקודת ה- SW ללא שום פקודות NOP נוספות. סה"כ

$$T_{total} = \#cycles * T = 250ns(4+6) = 2500ns$$