מערכות ספרתיות ומבנה המחשב (044252)

סמסטר אביב תש"פ

בחינה סופית – מועד א

פתרון

2 באוגוסט 2020

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 ולמחשבון.

הנחיות והוראות:

- הבחינה כתובה על גבי **22** עמודים כולל עמוד זה והעמוד הקודם (**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 ובנוסף **להעתיק את תשובותיכם הסופיות אל דפי התשובות**.
- אין **לתלוש או להפריד** דפים מחוברת הבחינה, ממחברות הטיוטה ומדפי העזר.
- יש לכתוב את התשובות באמצעות עט שחור או כחול בלבד. **אין לכתוב או לצייר בעט אדום**.
- רשמו את מספר הסטודנט שלכם על חוברת הבחינה (בראש עמוד זה), על דפי העזר, ועל כל מחברות הטיוטה. **ודאו כי על מחברת הבחינה ועל טופס התשובות האמריקאי מודבקת מדבקת הנבחן שלכם.**
- לא מורדות נקודות (אין "קנס") בגין תשובה שגויה. לכן, כדאי לסמן תשובה כלשהי לכל שאלה.
- ציון שאלות 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).**
- **את התשובות לשאלות הפתוחות יש לכתוב בדף אשר מצורף בתחילת מחברת הבחינה**. לנוחיותכם, בכל שאלה פתוחה ישנו איזור לכתיבת הפתרון, אך תשובות אשר ייכתבו באיזור זה לא יבדקו.
- אסור שימוש בכל חומר חיצוני מלבד מחשבון. 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 13:

Pcsource	*PCwrite*	*Regwrite*	*PCCen*	*IntExcep*	*CauseWrite*	*SEPCWrite*	*IntCause*

א.

	6	*7*	*8*	*10*
IntExcep				

ב.

$t2$	
$t3$	

שאלה 14:

```
0x1AA0 000C   OuterLoop:     sub s3, s1, s0                      // s3 = N-i-1
0x1AA0 0010                  addi s3, s3, -1
0x1AA0 0014   InternalLoop:  slli t0, s2, _________
0x1AA0 0018                  add t0, t0, _________
0x1AA0 001C                  addi t1, _________, 4               // access a[j]
0x1AA0 0020                  lw a0, 0(t0)                        // access a[j+1]
0x1AA0 0024                  lw a1, 0(t1)
0x1AA0 0028                  bge _________, _________, _________ // swap cells
0x1AA0 002C                  _________ a1, 0(_________)
0x1AA0 0030                  _________ a0, 0(_________)
0x1AA0 0034   AfterSwap:     addi s2, s2, 1
0x1AA0 0038                  bne s3, s2, InternalLoop
0x1AA0 003C                  addi _________, x0, _________      // j = 0
0x1AA0 0040                  addi s0, s0, 1
0x1AA0 0044                  bne s0, s7, OuterLoop
0x1AA0 0048   Exit:                                              // done
```

שאלה 15:

א.	ב.	ג.	ד.
_____ _____ _____ _____			

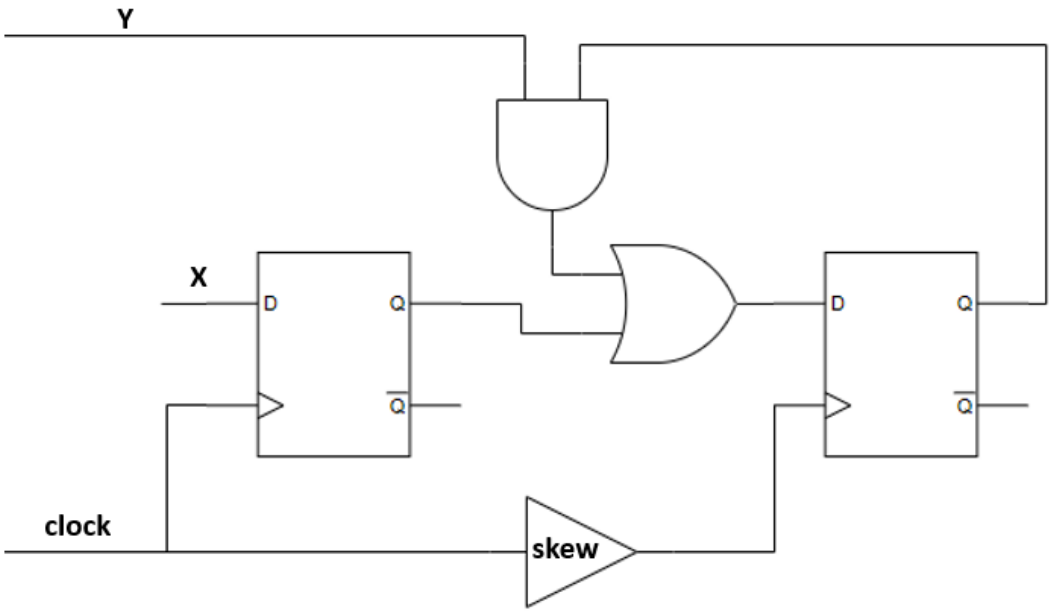
שאלה 61:

ZW \ XY	00	01	11	10
00				
01				
11				
10				

$f(x,y,z,w) =$

שאלה 17:

א. פועל בצורה תקינה	כן / לא
ב. מבין הערכים הבאים, מהו הערך של t_{skew} עבורו המעגל עומד במשטר הזמנים הדינאמי? 1. $-2ns$ 2. $3ns$ 3. $0ns$ 4. אף ערך 5. $1ns$	
ג.	$\leq t_{buffer} \leq$



שאלה 1 (5 נקודות)

נתון קוד ה-SystemVerilog הבא:

```
module my_module(
    input logic clk,
    input logic [1:0] a,
    input logic b,
    output logic o1,
    output logic o2,
    output logic o3,
    output logic o4
);

    always_comb begin
        casex(a)
            2'b00: o1 = 1'b0;
            2'b1?: o1 = 1'b1;
        endcase
    end

    always_ff @(posedge clk) begin
        o2 = b;
        o3 <= a[0];
        o4 = o3;
    end
endmodule
```

כמה רכיבי זיכרון ייווצרו בתהליך הסינתזה של קוד זה?

א. 1
ב. 2
ג. 3
ד. 4
ה. 5

פתרון:

התשובה הנכונה היא תשובה ד': ייווצרו ארבעה רכיבים.
הקוד מורכב משני procedural blocks. הבלוק הראשון הוא בלוק מסוג always_comb ומכיל בתוכו משפט case. כפי שנלמד בכיתה, במשפטי conditional בהם מתבצע פיצול של ה-flow, יש לדאוג להשמה בכל המקרים האפשריים. תווים z,x או ? הם תווים מיוחדים שיכולים להיות 0 או 1. במידה ולא מתבצעת השמה בכל המקרים האפשריים, נוצר/ים רכיב/י זיכרון (מסוג latch). במקרה זה, o1 הוא בעל ביט אחד, ולכן יווצר latch אחד.
הבלוק השני הוא בלוק מסוג always_ff. אנחנו יודעים שכאשר משתמשים ב-always_ff נוצר לפחות רכיב זיכרון (מסוג FF) אחד, אבל יש לקרוא את הקוד באופן מעמיק כדי להבין כמה רכיבי זיכרון ייווצרו, כאשר סוג ההשמה (blocking/non-blocking) יכול להשפיע. במקרה זה, o2 מקבל את הערך הקודם (לפני עליית השעון) של הכניסה b, ולכן ייווצר רכיב זיכרון אחד כתוצאה מההשמה הראשונה (ללא תלות בסוג ההשמה). O3 מקבל את הערך הקודם של הכניסה a[0], ולכן ייווצר עוד רכיב זיכרון (ללא תלות בסוג ההשמה הקודמת או ההשמה הזו, משום שהן לא תלויות אחת בשנייה). אם ההשמה של o3 היא מסוג non-blocking, היא תתבצע רק בסוף ה-procedural block ותסומן בטבלת המעקב בסוגריים. אם ההשמה של o3 היא מסוג blocking, היא תתבצע בסיום ההשמה ולא תסומן בסוגריים. כתלות בסוג ההשמה, O4 יקבל את הערך הישן או החדש של o3. אם ההשמה של o3 היא מסוג blocking לא יווצר רכיב זיכרון, ואם ההשמה היא מסוג non-blocking יווצר רכיב זיכרון. טבלת המעקב המתארת את הבלוק השני עבור גירסה 1 של השאלה:

	לפני עליית השעון	אחרי עליית השעון
a[0]	0	
b	0	
O2		0
O3	0	(0)
O4		0

כאשר כל חץ מהעמודה הראשונה אל העמודה השנייה מצביע על יצירה של FF.

שאלה 2 (5 נקודות)

נתון אוסף מילים באורך 4 ביט מהצורה $abcd$. אוסף המילים מהווה קוד, כאשר a הוא ה-MSB ו-d הוא ה-LSB. בכדי לשפר את יכולת גילוי השגיאות, הוחלט להרחיב את מילות הקוד המקורי כך שכל מילה תהיה מהצורה $abcdxyz$, כאשר מתקיים:

x – סיבית הזוגיות של המילה $abcd$ אשר שייכת לקוד המקורי

y – סיבית אי-הזוגיות של המילה $abcd$ אשר שייכת לקוד המקורי, כלומר סיבית אשר גורמת למספר ה-1-ים במילה להיות אי זוגי.

z – סיביות הזוגיות של המילה $abcdxy$.

הניחו כי הקוד מכיל לפחות שתי מילים, ובחרו את הטענה הנכונה:

א. עבור כל קוד מהצורה $\{abcd\}$, הקוד המורחב, $\{abcdxyz\}$, מגדיל את מרחק הקוד ב-3

ב. בהינתן כי מרחק הקוד של הקוד המקורי, $\{abcd\}$, הוא 2, מרחק הקוד של הקוד החדש, $\{abcdxyz\}$, הינו בהכרח 5

ג. קיים קוד מקורי, $\{abcd\}$, בעל מרחק קוד השווה ל-1, אשר הקוד המורחב שנוצר על בסיסו, $\{abcdxyz\}$, הוא בעל מרחק קוד השווה ל-4

ד. קיים קוד מקורי, $\{abcd\}$, בעל מרחק קוד השווה ל-1, אשר הקוד המורחב שנוצר על בסיסו, $\{abcdxyz\}$, הוא בעל מרחק קוד השווה ל-1

ה. תשובות ב' ו-ג' נכונות

פתרון:

תשובה ג

תחילה ננתח את מבנה הקוד החדש. אנו יודעים כי סיבית הזוגיות x מתקבלת על ידי החישוב הבא:

$x = a \oplus b \oplus c \oplus d$

סיבית אי הזוגיות מתקבלת על ידי היפוך ערכה של p, כלומר סיבית זו מקיימת:

$y = x'$

בכדי לחשב את סיבית z נבצע את הפעולה הבאה:

$z = a \oplus b \oplus c \oplus d \oplus x \oplus y = x \oplus x \oplus x' = x'$

כלומר צורתו של הקוד החדש הינה:

$abcdxx'x'$

א. **לא נכון**. נתבונן בקוד הבא: $\{1001,1010\}$ מרחק הקוד במקרה זה הוא 2. לאחר הוספת הסיביות, הקוד החדש אשר נקבל הוא $\{1001011,1010011\}$. מרחק הקוד החדש נותר 2 ולכן התשובה אינה נכונה.

ב. **לא נכון**. ראו תשובה א'.

ג. **נכון**. על פי המשפט אשר נלמד בכיתה, עבור קוד בעל מרחק קוד השווה ל-1, הוספה של סיביות זוגיות/אי-זוגיות מגדילה את מרחק הקוד ב-1. על פי הניתוח אשר בוצע בתחילת הפתרון אנו רואים כי הסיביות אשר נוספות לכל מילה הן סיביות זוגיות/אי-זוגיות בלבד. כל אחת מן הסיביות אשר נוספות מגדילה את מרחק הקוד ב-1. כיוון שמרחק הקוד המקורי הינו 1, מרחק הקוד החדש הינו 4. למעשה, משפט זה נכון עבור כל קוד בעל מן הצורה $(abcd)$ אשר מרחק הקוד שלו שווה ל-1.

ד. **לא נכון**. ראו תשובה ג'.

ה. **לא נכון**. ראו תשובות קודמות.

שאלה 3 (5 נקודות)

בשאלה זו התעלמו מקיום תנאי ה-hold במערכת.

במהלך התכנון של מעבד Pipelined RISC-V, בעל מנגנון forwarding מלא ($WB \rightarrow DEC, MEM \rightarrow EXE, WB \rightarrow EXE$) ו-hazard detection unit כפי שנלמדו בכיתה. קבלת ההחלטה על branch מתקבלת בשלב ה-execute. זמני ה-setup וה-t_{pcq} של רגיסטר PC זהים לאלו של הרגיסטרים אשר מפרידים בין השלבים.

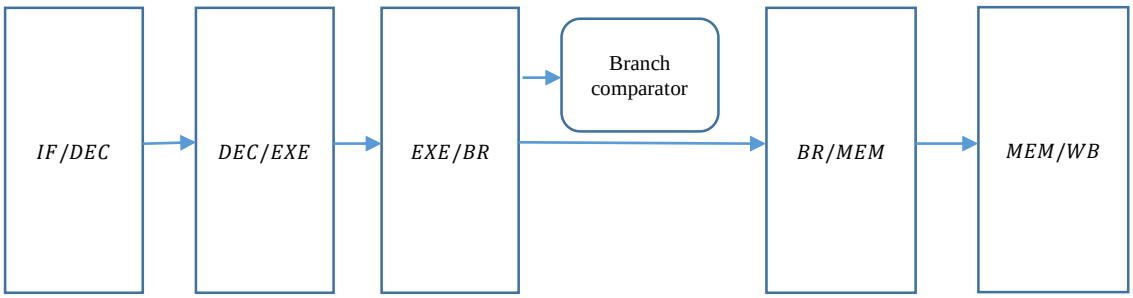
במעבד נפלה תקלה אשר גרמה להפרה של תנאי ה-setup של משטר הזמנים הדינאמי בשלב ה-execute. התקלה נגרמת מכיוון שזמן ה-t_{pd} של ה-Branch comparator הוא גדול מידי. בנוסף, התגלה כי תהליך הייצור גורם להיווצרותו של skew ברגיסטר mem/wb (ביחס לרגיסטרים האחרים אשר מפרידים בין השלבים השונים), וכי לא ניתן למנוע סטייה זו בשום צורה. בטבלה מתוארים פרמטרי המערכת. רכיבים אשר זמן ה-t_{pd} שלהם לא צוין הם בעלי זמן t_{pd} זניח.

Timing	
$5ns$	Memory access (data or instruction)
$4ns$	Read/write a value from/to the register file
$5ns$	ALU operation
$7ns$	branch comparator
$10ns$	T (זמן המחזור של המעבד המקורי)
$3ns$	Skew
$3ns$	$t_{pcq}(IF/DEC), (DEC/EXE), (EXE/MEM), (MEM/WB), (BR/MEM)$
$2ns$	$t_{su}(IF/DEC), (DEC/EXE), (EXE/MEM), (MEM/WB), (BR/MEM)$
$1ns$	$t_{pcq}(EXE/BR)$
$1ns$	$t_{su}(EXE/BR)$

בכדי לפתור את הבעיה הוצעו שלושה רעיונות:

1. הגדלת זמן המחזור של המעבד: $T_{new} = 12ns$
2. הוספת שלב חדש ל-Pipeline בין שלב ה-execute לשלב ה-memory, אשר יקרא br. רכיב ה-branch comparator יעבור לשלב זה. במידה וישנו צורך ב-forwarding, הוא מבוצע בשלב ה-execute והשלב החדש מקבל את הערכים העדכניים ביותר. התמיכה הרלוונטית לצורך תפקוד תקין של מנגנון ה-forwarding הכללי נוספת גם היא. הרגיסטר exe/br, יפריד בין שלב ה-exe לשלב ה-br ויהיה בעל הפרמטרים אשר צוינו בטבלה. פרמטרי הרגיסטר אשר יפריד בין שלב ה-br לשלב ה-mem, br/mem, זהים לרגיסטרים המקוריים ומופיעים בטבלה. זמן המחזור של המעבד נקבע על פי השינויים. מצורף איור של הצעה זו.
3. השארת המעבד המקורי ללא שינוי, למעט העברה של רכיב ה-branch comparator לשלב ה-memory.

הצעה 2:



העדיפות העיקרית של מתכנני המערכת הוא **latency קצר ורק לאחר מכן throughput גבוה (מקסימלי).**
מבין מהמשפטים הבאים, מהו המשפט הנכון על בסיס שיקולי המתכננים?

א. ההצעה הטובה ביותר היא הצעה 2, לאחריה הצעה 3 ולבסוף הצעה 1
ב. ההצעה הטובה ביותר היא הצעה 2, לאחריה הצעה 1 ולבסוף הצעה 3
ג. ההצעה הטובה ביותר היא הצעה 3, לאחריה הצעה 2 ולבסוף הצעה 1
ד. ההצעה הטובה ביותר היא הצעה 3, לאחריה הצעה 1 ולבסוף הצעה 2
ה. הצעות 1 ו-3 שקולות, וטובות יותר מהצעה 2

פתרון :

תשובה ג'.
ראשית נשים לב כי במעבד בשלב ה-mem מתקיים:
$$t_{pd}(EXE/MEM) + t_{pd}(logic) + t_{su}(MEM/WB) \leq T + skew$$
כלומר ניתן להוסיף לוגיקה אשר אינה מקיימת:
$$t_{pd}(EXE/MEM) + t_{pd}(logic) + t_{su}(MEM/WB) \leq T$$
אך עומדת בתנאי הראשון.

הצעה 1:
עבור זמן המחזור החדש, בשלב ה-exe יתקיים:
$$t_{pd}(DEC/EXE) + t_{pd}(br\ comp) + t_{su}(EXE/MEM) = 12ns \leq 12ns = T_{new}$$
כלומר אנו עומדים בתנאי ה-su.
המעבד זהה למעבד ה- pipelined RISC-V כפי שנלמד בכיתה, אך עם זמן מחזור שונה. לכן מתקיים:
$$Throughptut = \frac{1}{T_{new}} = \frac{1}{12ns} = \frac{1}{12}GHz = 83.33Mhz$$
$$Latency = 5 \cdot T_{new} = 12ns \cdot 5 = 60ns$$

הצעה 2:
נשים לב כי הוספת השלב החדש מאפשרת עמידה בתנאי ה-su יחד עם זמן המחזור הישן:
$$t_{pd}(EXE/BR) + t_{pd}(br\ comp) + t_{su}(BR/MEM) = 1 + 7 + 2 = 10ns \leq 10ns = T$$
$$t_{pd}(DEC/EXE) + t_{pd}(ALU) + t_{su}(EXE/BR) = 3 + 5 + 1 = 9ns \leq 10ns = T$$
עבור המעבד בעל השלב הנוסף אין שינוי ב-throughput (נשארנו עם אותו זמן המחזור), אך הוספנו שלב ל-Pipe ולכן זמן ה-Latency גדל.
$$Throughptut = \frac{1}{T} = \frac{1}{10ns} = \frac{1}{10}GHz = 100MHz$$
$$Latency = 6 \cdot T = 6 \cdot 10ns = 60ns$$
הצעה 3:
בהצעה זו מזיזים את ה-branch comparator לשלב ה-mem, לכן צריך לבדוק את האם התנאי הבא מתקיים:
$$t_{pd}(EXE/MEM) + t_{pd}(br\ comp) + t_{su}(MEM/WB) \leq T + skew$$
וניתן לראות כי אכן
$$3 + 7 + 2 = 12ns \leq 13ns = T + skew$$
כלומר, במידה ומבצעים את הזזת ה-branch comparator לשלב ה-mem, ניתן להותיר את זמן המחזור ללא שינוי, וכך קורה גם עם מספר שלבי ה-pipe, לכן:
$$Throughptut = \frac{1}{T} = \frac{1}{10ns} = \frac{1}{10}GHz = 100MHz$$
$$Latency = 5 \cdot T = 5 \cdot 10ns = 50ns$$
לסיכום:

הצעה 1	הצעה 2	הצעה 3
$Latency = 60ns$	$Latency = 60ns$	$Latency = 50ns$
$Throughptut = 83.33Mhz$	$Throughptut = 100MHz$	$Throughptut = 100MHz$

לכן, על פי סדר העדיפויות שהוגדר, ההצעה הטובה ביותר היא הצעה 3, לאחריה הצעה 2 ולבסוף הצעה 1, כלומר התשובה הנכונה היא ג'.

שאלה 4 (5 נקודות):

נתוני שאלה זו זהים לאלו של השאלה הקודמת. כל שלוש ההצעות עובדות תחת ההנחה שפקודות קפיצה אינן מתבצעות, ובמידה ומתגלה כי קפיצה כן צריכה להתבצע מבוצע שימוש במגנון flush בדומה לנלמד בכיתה.

מהו מספר הפקודות אשר עליהן מתבצע ה-flush במידה ויש בו צורך?

א. הצעה 1 – 2 פקודות, הצעה 2 – 2 פקודות, הצעה 3 – 2 פקודות
ב. הצעה 1 – 2 פקודות, הצעה 2 – 2 פקודות, הצעה 3 – 3 פקודות
ג. הצעה 1 – 2 פקודות, הצעה 2 – 3 פקודות, הצעה 3 – 3 פקודות
ד. הצעה 1 – 3 פקודות, הצעה 2 – 3 פקודות, הצעה 3 – 3 פקודות

פתרון:
תשובה ג.
נשים לב כי הצעה 1 זהה לנלמד בכיתה. בהצעה 2 אנו מוסיפים שלב ומזיזים את שלב ההחלטה, לשלב הרביעי כלומר יזרקו 3 פקודות. בהצעה 3 ההחלטה זזה לשלב הרביעי ולכן גם במקרה זה יזרקו 3 פקודות.

<u>שאלה 5 (5 נקודות)</u>

נתונות שתי פונקציות: f ו-g המקבלות כקלט מספר בינארי בן 4 סיביות $wxyz$.
הפונקציה f מוציאה 1 אם ורק אם המספר מתחלק ב-'3' ללא שארית.
הפונקציה g מוציאה 1 אם ורק אם המספר מתחלק ב-'2' ללא שארית.
שימו לב: המספר 0 אינו יכול להתקבל כקלט.

הפונקציה h ממומשת באופן הבא:

$$h(w,x,y,z) = \big(f(w,x,y,z) \oplus g(w,x,y,z)\big) \cdot \overline{wxyz}$$

כאשר הסימן $\oplus$ מציין את הפונקציה XOR.

כאשר מצמצמים את h כסכום מכפלות, אילו מהביטויים הבאים מתאר את הפונקציה המצומצמת ביותר?

א. $h(w,x,y,z) = w'y'z' + wx'y' + wyz' + w'x'y$
ב. $h(w,x,y,z) = w'y'z' + wx'y' + wyz' + w'x'y + x'z'$
ג. $h(w,x,y,z) = x'z'$
ד. $h(w,x,y,z) = x'z' + wxyz$
ה. התשובות א' – ד' אינן נכונות

פתרון:

נתבונן במפת קרנו של שתי הפונקציות:

הפונקציה g:

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

הפונקציה f:

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

ולכן h תהיה XOR בין שתי הפונקציות, מוכפלת ב- $\overline{wxyz}$:

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

מצמצום מפת הקרנו נקבל:

$$h(wxyz) = w'y'z' + wx'y' + wyz' + w'x'y$$

שימו לב שהביטוי $x'z'$ הינו מיותר ומסומן כבר ע"י הגוררים האחרים. תשובה א'.

<u>שאלה 6 (5 נקודות)</u>

נתונה הפונקציה הבאה: $f(w,x,y,z) = \sum(3,4,5,11,12,13) + \sum_{\phi}(6,7,15)$
במעבדה ישנם בוררים (סלקטורים) בגדלים שונים, ושערי AND ו- OR **(לא ניתן להשתמש ב-NOT)**.
מהנדס מעוניין לממש את הפונקציה בעזרת מספר מינימלי של בוררים בעדיפות ראשונה, ובוררים קטנים ככל האפשר בעדיפות שניה. ניתן להשתמש בשערים הלוגיים הנתונים באופן חופשי.

איך ניתן לממש את הפונקציה באופן המיטבית על פי סדר העדיפויות אשר הוגדר?

א. אין צורך בבוררים במימוש הפונקציה.
ב. בעזרת בורר $1 \rightarrow 2$ יחיד.
ג. בעזרת בורר $1 \rightarrow 4$ יחיד.
ד. בעזרת בורר $1 \rightarrow 8$ יחיד.
ה. בעזרת שני בוררים בגודל $1 \rightarrow 4$.

פתרון :

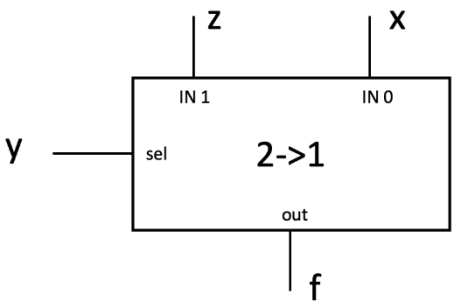
מפת קרנו של הפונקציה:

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

לכן, הפונקציה המצומצמת תראה כך:

$$f(w,x,y,z) = y'x + yz$$

הפונקציה המצומצמת ניתנת למימוש בעזרת בורר $2 \to 1$, באופן הבא:



כמובן ניתן לפתור את השאלה בעזרת טבלת אמת כפי שראינו בתרגול, ובדיקת כל אפשרויות הקלטים בכניסות הבקרה של הבוררים.

שאלה 7 (5 נקודות)

עבור הפקודות הבאות, איזו פקודה לא ניתן לממש כפקודה אמיתית (לא פסאודו-פקודה) במעבד ה-Multicycle RISC-V ?

ניתן לבצע שינויים בבקר והוספת בוררים וחיווטים במסלול הנתונים של המעבד, אך אסור לבצע שינויים ביחידות ה-Register file, Memory, ALU.

א. פקודת swap rd, rs1,rs2 אשר מחליפה בין התוכן של שלושת הרגיסטרים כך שמתקיים:

.rs1->rs2, rs2->rd, rd->rs1

ב. פקודת mv rd, rs1,rs2 אשר טוענת את הערך השמור ברגיסטר rd לרגיסטרים rs1 ו-rs2.

ג. פקודת addi24 rd, rs, imm אשר מוסיפה ערך immediate בגודל 24 ביט לערך אשר שמור ברגיסטר rs ושומרת את התוצאה ברגיסטר rd.

ד. פקודת add3 rd, rs1, rs2 אשר מבצעת את הפעולה rd=rd+rs1+rs2 (שומרת את התוצאה לרגיסטר rd).

ה. ניתן לממש את כל הפקודות הנ"ל.

פתרון:

תשובה ג' נכונה:

לא ניתן לממש addi24 מכיוון שלא ניתן לקודד בפקודת אסמבלי (בגודל 32 ביט) גם opcode, גם מספרי רגיסטרים, וגם ערך imm של 24 ביטים.

שאלה 8 (5 נקודות)

נתונה טבלת המעברים של מערכת עקיבה בעלת כניסה אחת, X, ויציאה אחת, Z:

	X=0		X=1	
Present State	Next state	Z	Next state	Z
A	A	0	B	0
B	E	0	C	0
C	A	0	D	0
D	A	0	D	1
E	E	0	F	0
F	E	0	C	0

מהן מחלקות השקילות אשר מתקבלות מצמצום מכונת המצבים הנתונה?

א. (ABCDEF)
ב. (AB)(CD)(EF)
ג. (AE)(BF)(CD)
ד. (A)(B)(C)(D)(E)(F)
ה. (AE)(BF)(C)(D)

פתרון:

תשובה ה'.

$P_0 = (ABCDEF)$

X=0 → $(ABCDEF)$

X=1 → $(D)(ABCEF)$

$P_1 = (D)(ABCDE)$

(D) $(ABCEF)$

X=0 → $(AEAEE)$

X=1 → $(BCDFC)$

$P_2 = (D)(ABDE)(C)$

(D) (C) $(ABEF)$

X=0 → $(AEEE)$

X=1 → $(BCFC)$

$P_3 = (D)(AE)(BF)(C)$

(D) (C) (AE) (BF)

X=0 → (AE)

X=1 → (BF)

X=0 → (EE)

X=1 → (CC)

$P_4 = (D)(AE)(BF)(C)$

שאלה 9 (5 נקודות)

התקשורת בין חיפה לתל אביב מתבססת על פרוטוקול ה-UART הבסיסי כפי שנלמד בכיתה (בכל שידור נשלחות 8 סיביות מידע, סיבית start וסיבית stop). על הקו נשלחות מילים (words) **באורך 64 סיביות**. קצב שידור המילים הוא $f_{word} = 2000 \left[\frac{words}{sec}\right]$.
לאור השיבושים הרבים בקו התקשורת, החליטו מתכנני הקו לשלוח, בנוסף למילה המקורית, סיבית זוגיות עבור כל בית במילה, ללא ביצוע שינויים בפרוטוקול ה-UART הבסיסי כפי שנלמד בכיתה. השינוי היחיד אשר ניתן לבצע הוא שינוי משך השידור של ביט בודד (T_{bit}).

מהו משך השידור $T_{bit-new}$ אשר יאפשר שמירה על קצב שידור המילים המקורי?

א. $5.56\mu sec$
ב. $5.68\mu sec$
ג. $6.25\mu sec$
ד. $6.94\mu sec$
ה. $7.81\mu sec$

פתרון:

לפני השינוי, כדי לשלוח מילה היינו צריכים לשדר 8 פעמים על בסיס הפרוטוקול, כיוון שהיו 64 סיביות מידע לשדר וכל שידור מעביר 8 סיביות. כיוון שלכל שידור מתווספות סיבית התחלה וסיבית סיום היינו משדרים סה"כ 80 סיביות לכל מילה. קצב השידור של מילה אשר היה מתקבל הינו: $f_{word-old} = \frac{1}{\text{bits-per-word} \cdot T_{bit-old}}$. כאשר $\text{bits\_per\_word} = 80$

לאחר השינוי, מכיוון שבכל מילה יש 8 בתים, ולכל בית אנחנו מוסיפים ביט זוגיות, אנחנו נוסיף עוד בית שלם/שני בתים של סיביות מידע, ובכך נצטרך לשדר שידור נוסף. מה שגורר שידור של סה"כ 90 סיביות עבור מילה אחת, ונותן קצב של: $f_{word-new} = \frac{1}{new\_\text{bits\_per\_word} \cdot T_{bit-new}}$.
כאשר $new\_\text{bits\_per\_word} = 90/180$

מהדרישה לערכו נקבל:

$$f_{word_{old}} = \frac{1}{new_{\text{bits}_{\text{per}_{\text{word}}}} \cdot T_{bit-new}} \Rightarrow$$

$$T_{bit-new} = \frac{1}{f_{word_{old}} \cdot new_{\text{bits}_{\text{per}_{\text{word}}}}} = 5.56 \cdot \mu sec$$

$$f_{word_{old}} = 2000; new\ bits\ per\ word = 90 \rightarrow T_{bit-new} = 5.56 \mu sec$$

שאלה 10 (5 נקודות)

במפעל לייצור מעבדים התגלתה תקלה במעבדי ה-Multicycle RISC-V (מכונת המצבים זהה לזו אשר נתונה בדף העזר). התקלה מתרחשת בעת ביצוע הכתיבה ל-register file. בכל כתיבה שנייה אל ה-register file הכתיבה נכשלת.
על מנת לפתור את התקלה, הוצע לבצע את שלב ה-Write Back פעמיים בכל פעם שבה נדרש לבצע כתיבה אל ה-register file.

בהנתן כי ניתן לבצע שינויים במכונת המצבים בבקר, אך לא ניתן לשנות את מסלול הנתונים, מהו מספר המצבים המינימלי שיש להוסיף למכונת המצבים בבקר, על מנת לתמוך בפתרון זה?

א. אין צורך בהוספת מצבים חדשים
ב. הוספת מצב חדש אחד
ג. הוספת שני מצבים חדשים
ד. הוספת שלושה מצבים חדשים
ה. הוספת ארבעה מצבים חדשים
ו. אין אפשרות לתמוך בפתרון זה

פתרון:

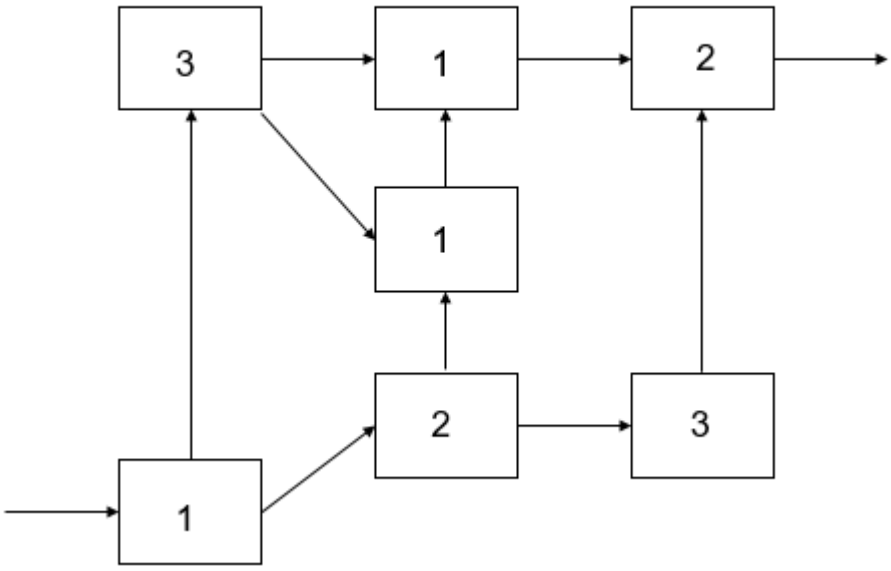
למעשה, עלינו לשים לב כי השלב שהמעבד לא מצליח לבצע הוא שלב ה*WB*.
כלומר עלינו בסה"כ לבצע פעמיים רק את שלב ה-*WB* ולא את כל מהלך הפקודה, זאת בשל העובדה שזה מעבד *Multicycle* והפקודה הבאה לא תתחיל עד לסיום הפקודה הנוכחית.
מכיוון שאסור לנו לשנות שום דבר ב*data path*, הפתרון שלנו יהיה שינוי מכונת המצבים. לכן נוסיף מצב נוסף לאחר שלב ה*WB* המקורי של הפקודות המבצעות שלב *WB*, שיבצע בדיוק אותו הדבר. כך נבטיח כתיבה מוצלחת בכל פקודה.

הפקודות שמבצעות שלב *WB* הן, *LW*, פקודות *Rtype*, ופקודת *JAL*.
לכן נצטרך להוסיף 3 מצבים חדשים סך הכל.

- נשים לב שפתרון הכולל הוספת מצב חדש בלבד ל*WB* אינו אפשרי מכיוון שאת כל אחד מהמצבים האלו אנחנו מבצעים באופן ספציפי לאותה פקודה.

שאלה 11 (5 נקודות)

נתונה המערכת הבאה:



זמן ההשהייה של כל רכיב כתוב בתוכו ונתון ב-ns.

נרצה לצנר את המערכת בעזרת רגיסטרים אידיאליים על מנת לקבל $throughput$ מקסימלי בעדיפות ראשונה, ומספר רגיסטרים מינימלי בעדיפות שניה.

מהו מספר הרגיסטרים המינימלי אשר דרוש לצורך צינור המערכת על פי סדר עדיפויות זה?

א. 5
ב. 8
ג. 9
ד. 10
ה. 12

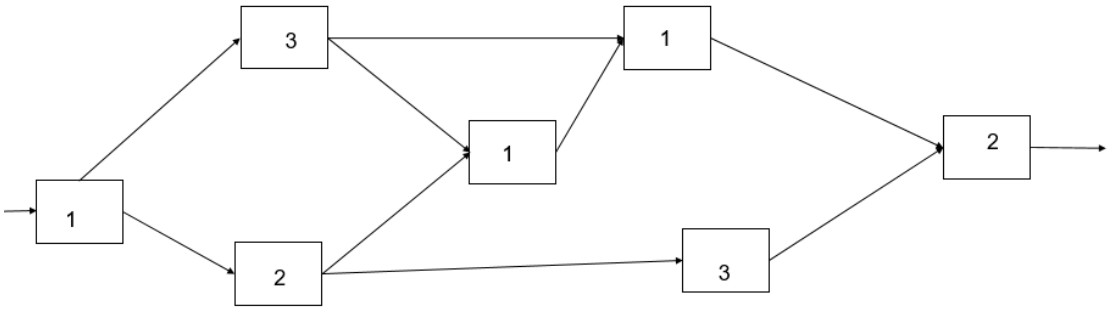
פתרון:

ניזכר כי:

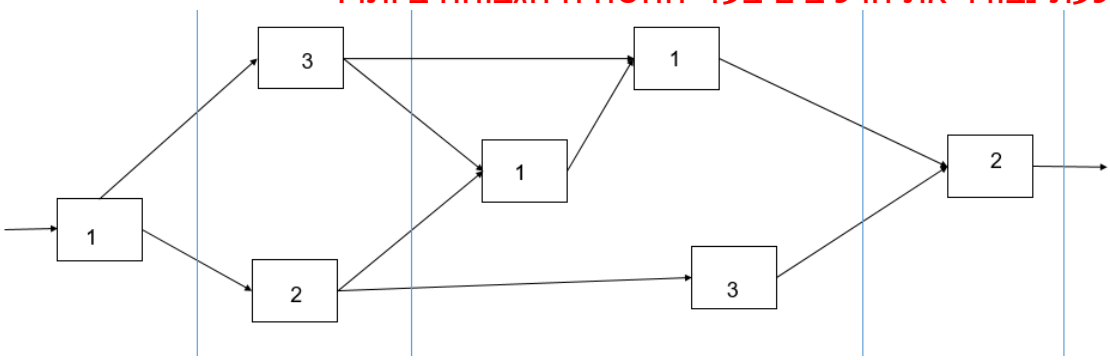
$$throughput = \frac{1}{T_{min}}$$

זמן ההשהיה הארוך ביותר ליחידה הוא $3ns$ והוא מגדיר לנו את זמן המחזור המינימלי, נקבל לכן $throughput = \frac{1}{3ns} = \frac{1}{3}GHz$.

ראשית, נשרטט את המעגל בצורה שונה (נוחה יותר):



כעת נבודד את הרכיבים בעלי ההשהיה הגבוהה ביותר:



נבדוק האם יש מסלול מרגיסטר לרגיסטר עם זמן השהייה גדול מ-3, מכיוון שאין סיימנו, וקיבלנו כי דרושים כ-9 רגיסטרים.

שאלה 12 (5 נקודות)

נתון מעבד Multicycle RISCV התומך בטיפול בחריגות, כך שהוא מפסיק את ריצת התוכנית במידה והתקבלה חריגה. אחד הסטודנטים בקורס "מערכות ספרתיות ומבנה המחשב" כתב את הקוד הבא:

```
0x1AA0 0000        Main:   addi x2, x0, 4
0x1AA0 0004                addi x4, x0, 1
0x1AA0 0008                mult x1, x2, x2
0x1AA0 000C                add x1, x1, x1
0x1AA0 0010                add x0, x1, x2

0x1AA0 0014        Loop:   addi x1, x1, -1
0x1AA0 0018                beq x1, x0, EXIT
0x1AA0 001C                add x4, x4, x4
0x1AA0 0020                div x0, x1, x1
0x1AA0 0024                j Loop
0x1AA0 0028        EXIT:
0x1AA0 002C                sw x4, 0(x4)
```

שימו לב:
הפקודה $mult\ rd, rs1, rs2$ מבצעת כפל בין שני הרגיסטרים, $rs1, rs2$ כך שמתקיים:

$rd = rs1 \cdot rs2$

באופן דומה הפקודה $div\ rd, rs1, rs2$ מבצעת חלוקה כך שמתקיים:
$rd = rs1/rs2$

הסטודנט בדק את רכיב הזיכרון וגילה כי נפחו הוא 1GB ($1G = 2^{30}B$).
שותפו של הסטודנט בחן את הקוד וקבע בהחלטיות כי הרצת הקוד תגרום לחריגה. עזרו לסטודנט להבין מה היא החריגה אשר שותפו זיהה. סמנו את התשובה הנכונה ביותר:

א. תתרחש חריגה מסוג "כתיבה לרגיסטר x0"
ב. תתרחש חריגה מסוג "חלוקה ב 0"
ג. התוכנית תקינה ותרוץ כהלכה (ללא חריגות)
ד. תתרחש חריגה מסוג גישה לכתובת לא חוקית
ה. תתרחש חריגה מסוג גלישה (Overflow)

פתרון:

תשובה א' נפסלת מכיוון שהפעולה חוקית ואין חריגה כזו.
תשובה ב' נפסלת מכיוון שהחלוקה ב – 0 לא תתבצע לעולם, אנו נקפוץ בפקודה beq לפני שנגיע לפקודת div ברגע שיתקיים השוויון.
נקבל Overflow בפעם האחרונה שנבצע add x4, x4, x4 בגלל שנבצע בעצם את החישוב 2^30+2^30, התשובה היא כמובן 2^31. אך add זו פעולה עם סימן (signed) והמספר עם סימן הגדול ביותר שניתן לייצג הוא 2^31-1, לכן יהיה overflow. ה- overflow יקרה לפני שניגש לזיכרון לכתובת לא חוקית ולכן תשובה ה' היא התשובה הנכונה. אם היה נתון שהפעולה הינה unsigned אז ד' הייתה הנכונה שכן גודל הזיכרון הינו 2^30B.

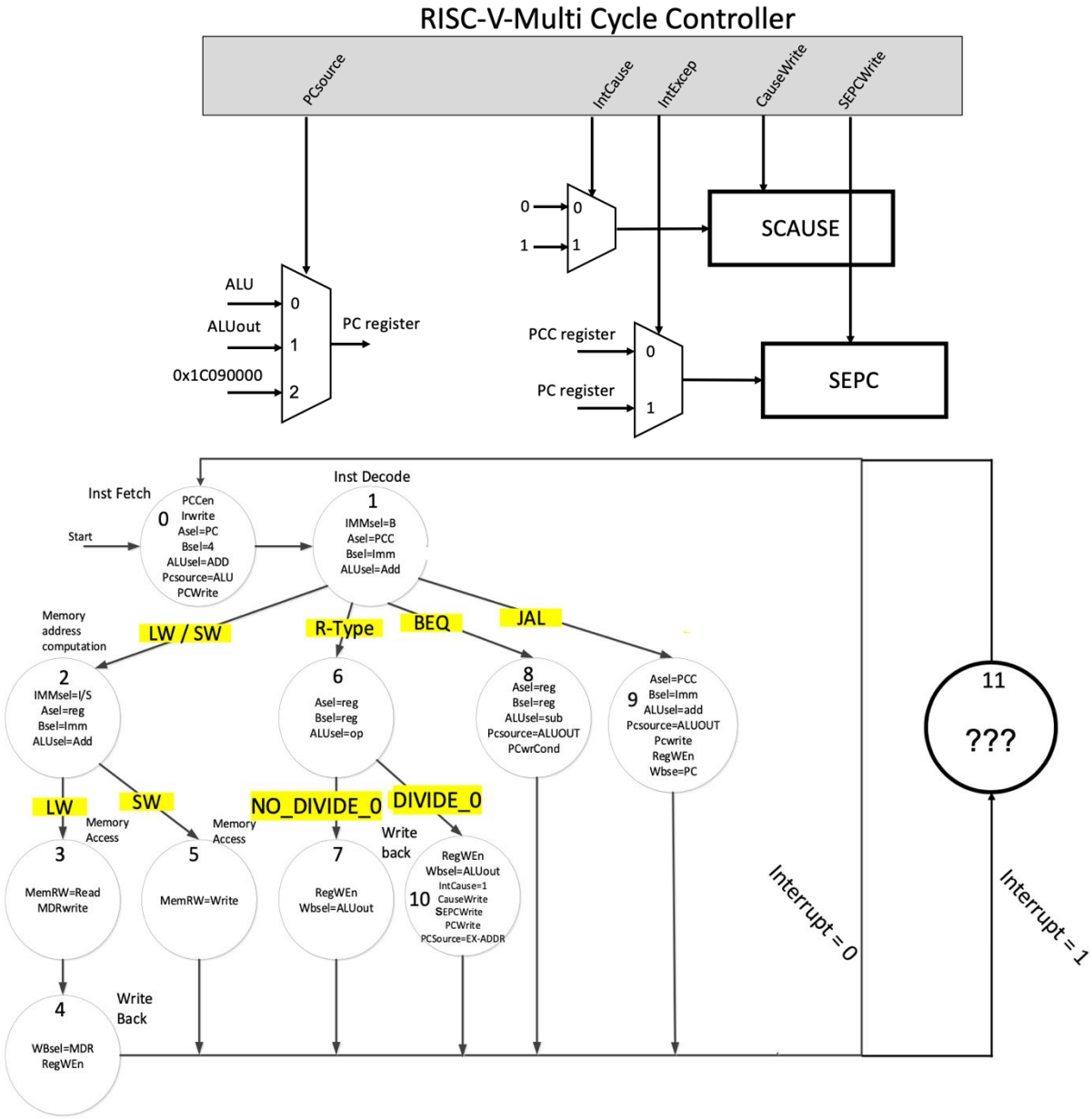
שאלה 13 (8 נקודות)

נתון מעבד מסוג Multicycle RISC-V שיכול לטפל בחריגות (Exceptions). בשאלה זו נתמקד בחריגות אשר נגרמו ע"י חלוקה ב-0 (על ידי הפרדה למצבים שונים במכונת המצבים – DIVIDE_0 ו-NO_DIVIDE_0). בסיום הטיפול בחריגה, הפקודה בה התרחשה החריגה מתבצעת מחדש.
מהנדס מעוניין להוסיף למעבד אפשרות לטפל גם בחריגות הנגרמות ע"י אות חיצוני (פסיקות – Interrupts). כאשר מגיעה פסיקה חיצונית, הבקר מקבל אות Interrupt=1 מאות חיצוני (ונשאר '1' עד לסיום הטיפול בפסיקה). המהנדס מעוניין שהמעבד יסיים את ביצוע שלבי הפקודה הנוכחית, ורק לאחר מכן יעביר את השליטה למערכת ההפעלה. לאחר סיום הטיפול בפסיקה, מערכת ההפעלה תחזור לפקודה הבאה בתכנית המקורית ולא תריץ שוב את הפקודה שבמהלכה התקבלה הפסיקה.
בנוסף, המהנדס החליט שעבור Interrupt חיצוני ישמר ב- SCAUSE הערך '0', ועבור חלוקה ב- 0, ישמר ב- SCAUSE הערך '1'.

למשל, אם המעבד נמצא במהלך שלב ה-Decode של LW, אז רק לאחר ביצוע שלב ה-WriteBack של הפקודה, השליטה תעבור למערכת ההפעלה. לאחר סיום הטיפול, המעבד יחזיר את השליטה לפקודה שלאחר ה-LW.

להלן השינויים במסלול הנתונים עבור טיפול בחריגות ופסיקות, ודיאגרמת המצבים של הבקר (שאר מסלול הנתונים של המעבד ללא שינוי כפי שנתון בדפי העזר).
שימו לב שנוספה כניסה חדשה בשם Interrupt לבקר, ויציאה חדשה בשם IntExcep הקובעת האם החריגה היא בעקבות אות חיצוני או לא.

<u>הערה</u>: ניתן להניח כי בכל רגע נתון קיימת לכל היותר חריגה אחת או פסיקה אחת (לא יכולות להתקבל חריגה ופסיקה בו זמנית).



א.

מלאו את קווי הבקרה עבור מצב 11 המטפל בפסיקה בטבלה להלן (אין צורך למלא קווי הבקרה שלא נמצאים בטבלה). במידה וקו בקרה יכול להיות ערך שרירותי כלשהו יש לסמן ϕ עבור Don't Care.

PCsource	*PCwrite*	*Regwrite*	*PCCen*	*IntExcep*	*CauseWrite*	*SEPCWrite*	*IntCause*

מלאו בטבלה הבאה את ערך הסיגנל IntExcep, עבור המצבים הנתונים (כל מספר מייצג מצב מתאים במכונת המצבים):

	6	*7*	*8*	*10*
IntExcep				

ב.

המהנדס הוסיף את המימוש הדרוש והוא כעת תומך בחריגות עבור "חלוקה ב-0" ובפסיקות חיצוניות ע"י קו הבקרה *Interrupt*.
מהנדס אחר מעוניין לבדוק את תקינות המעבד ע"י הרצת הקוד להלן.
הרגיסטרים *t0, t2, t3* מאותחלים לערך 0, והרגיסטר *t1* מאותחל לערך *10*.
שימו לב שלצורך השאלה, נתון הקוד שמערכת ההפעלה מריצה עבור כל טיפול בפסיקה / חריגה החל מכתובת *0x1c090000*, כפי שניתן לראות להלן.

```
                                0x10000000    div     t2, t1, t0
                                0x10000004    addi    t3, t2, 5
                                0x10000008    add     t3, t3, t3
                                              ...
Interrupt/exception handler:    0x1c090000    addi    t0, x0, 2
                                0x1c090004    add     t3, t3, t3
                                0x1c090008    jr      SEPC
```

כדי לבדוק את תקינות הקוד, המהנדס יוזם סיגנל Interrupt חיצוני במהלך שלב ה- **Decode** של הפקודה בכתובת 0x10000004.
מה יהיו ערכי הרגיסטרים t2 ו- t3 לאחר סיום ביצוע הפקודה שבכתובת 0x10000008?

פתרון:

א.

Pcsource	PCwrite	Regwrite	PCCen	IntExcep	CauseWrite	SEPCWrite	IntCause
2	1	0	0	1	1	1	0

	6	7	8	10
IntExcep	ϕ	ϕ	ϕ	0

הורדו למי

לא נקודות שמילא ערך אחר עבור PCCen

ב.

גרסה 1:

במהלך ביצוע פקודת ה- *div*, יחול *Exception* בעקבות החלוקה ב-0. מערכת ההפעלה תגדיל את *t0* ב- 2, ותחזיר את השליטה לקוד שיריץ שוב את הפקודה *div*, הפעם בהצלחה, **וישמור את התוצאה 5 ברגיסטר *t2*** (10 לחלק ל- 2). הפקודה הבאה תסתיים לרוץ ותעדכן את *t3* להיות 10 (5+5), למרות שקיבלה *Interrupt* ורק לאחר מכן תעביר את השליטה למערכת ההפעלה. מערכת ההפעלה תעדכן את *t3* להיות 20 (10 כפול 2). כעת מערכת ההפעלה תחזיר שליטה לפקודה שבכתובת *0x10000008*, משום שב- *Interrupt* אנחנו לא חוזרים לפקודה שגרמה לחריגה, אלא לפקודה הבאה. לאחר ביצוע הפקודה שבכתובת *0x10000008*, **ערך הרגיסטר *t3* יהיה 40** (20 כפול 2).

בסה"כ: *t2=5*, ו- *t3=40*.

עבור גרסה 2, החישוב זהה אך המספרים שונים.

נקבל – *t2=8/4=2*, ועבור *t3* נקבל: *t3=(8/4+4)*2*2=24*.

שאלה 14 (8 נקודות)

סטודנט חרוץ החליט לממש את אלגוריתם המיון *bubble sort* בעזרת קוד אסמבלי. האלגוריתם יבצע מיון על מערך של מספרים, כאשר **כל מספר הוא בגודל של 4 בתים**. עקב תקלה, חלקים מן המימוש נמחקו. עליכם להשלים את חלקי הקוד החסרים (מסומנים בקו תחתון) בכדי שהמימוש יפעל כנדרש.
לנוחיותכם מצורף מימוש אלגוריתם המיון בקוד *c*:

```
void bubbleSort(int arr[], int N)
{
   int i, j;
   for (i = 0; i < N-1; i++)

       // Last i elements are already in place
       for (j = 0; j < N-i-1; j++)
           if (arr[j] > arr[j+1])
              swap(&arr[j], &arr[j+1]);
}
```

הפונקציה *swap* מבצעת החלפה במיקומם של שני איברים במערך.
במהלך המימוש הניחו כי המיפוי בין רגיסטרים למשתנים הוא:

$s0 \rightarrow i, s1 \rightarrow N, s2 \rightarrow j, s10 \rightarrow arr$

שימו לב כי **רגיסטר $s1$ מכיל את גודל המערך (N)**.

<u>יש להשלים את הקוד אשר נתון להלן במקומות הנדרשים</u>:

```
0x1AA0 0000          Main:  addi s0, x0, 0                  // s0 = i = 0
0x1AA0 0004                 addi s7, s1, -1                 // s7 = N-1
0x1AA0 0008                 addi s2, x0, 0                  // s2 = j = 0

0x1AA0 000C     OuterLoop:  sub s3, s1, s0
0x1AA0 0010                 addi s3, s3, -1                 // s3 = N-i-1
0x1AA0 0014 InternalLoop:   slli t0, s2, _________
0x1AA0 0018                 add t0, t0, _________
0x1AA0 001C                 addi t1, _________, 4
0x1AA0 0020                 lw a0, 0(t0)                    // access a[j]
0x1AA0 0024                 lw a1, 0(t1)                    // access a[j+1]
0x1AA0 0028                 bge _________, _________, _________
0x1AA0 002C                 _________ a1, 0(_________)     // swap cells
0x1AA0 0030                 _________ a0, 0(_________)
0x1AA0 0034      AfterSwap: addi s2, s2, 1
0x1AA0 0038                 bne s3, s2, InternalLoop
0x1AA0 003C                 addi _________, x0, _________  // j = 0
0x1AA0 0040                 addi s0, s0, 1
0x1AA0 0044                 bne s0, s7, OuterLoop
0x1AA0 0048           Exit:                                 // done
```

פתרון:

```
0x1AA0 0000          Main:  addi s0, x0, 0          // s0 = i = 0
0x1AA0 0004                 addi s7, s1, -1         // s7 = N-1
0x1AA0 0008                 addi s2, x0, 0          // s2 = j = 0

0x1AA0 000C     OuterLoop:  sub s3, s1, s0
0x1AA0 0010                 addi s3, s3, -1         // s3 = N-i-1
0x1AA0 0014 InternalLoop:   slli t0, s2, 2
0x1AA0 0018                 add t0, t0, s10
0x1AA0 001C                 addi t1, t0, 4
0x1AA0 0020                 lw a0, 0(t0)            // access a[j]
0x1AA0 0024                 lw a1, 0(t1)            // access a[j+1]
0x1AA0 0028                 bge a1, a0, AfterSwap
0x1AA0 002C                 sw a1, 0(t0)            // swap cells
0x1AA0 0030                 sw a0, 0(t1)
0x1AA0 0034      AfterSwap: addi s2, s2, 1
0x1AA0 0038                 bne s3, s2, InternalLoop
0x1AA0 003C                 addi s2, x0, 0          // j = 0
0x1AA0 0040                 addi s0, s0, 1
0x1AA0 0044                 bne s0, s7, OuterLoop
0x1AA0 0048           Exit:                         // done
```

שאלה 15 (8 נקודות)

מעוניינים להוסיף מימוש של הפקודה dlw **כפסאודו פקודה** תוך שימוש בפקודות קיימות. פקודה זו מביאה מילה מהזיכרון לפי כתובת המחושבת באופן הבא: כתובת המילה מובאת מהזיכרון מהכתובת ששמורה ברגיסטר rs ועוד ערך ה-imm, ושומרת את המילה שהובאה מהזכרון ברגיסטר rd.
פקודה זו בעלת הפורמט:

dlw rd, rs, imm

המבצעת את הפעולה הבאה:

reg[rd] ← Mem[Mem[reg[rs]+imm]]

א. כתבו את המימוש המינימלי של הפקודה כרצף של פקודות אמתיות (ניתן להשתמש ברגיסטרים t0,t1 במידת הצורך).

_____

_____

_____

_____

ב. מה מספר המחזורים המינימלי הנדרש לביצוע פסאודו פקודה זו במעבד single cycle RISC-V?

ג. מה מספר המחזורים המינימלי הנדרש לביצוע פסאודו פקודה זו במעבד Multicycle RISC-V?

ד. כעת ניתן לבצע שינויים במעבד הכוללים הוספת/הרחבת בוררים, והוספת חיווטים. מה מספר המחזורים **המינימלי** הנדרש לביצוע פקודה זו **כפקודה אמיתית** במעבד Multicycle RISC-V?

פתרון:

א. כתבו את המימוש המינימלי של הפקודה כרצף של פקודות אמתיות (ניתן להשתמש ברגיסטרים t0,t1 במידת הצורך).

lw t0, imm(rs)
lw rd, 0(t0)

ב. מה מספר המחזורים המינימלי הנדרש לביצוע פסאודו פקודה זו במעבד single cycle RISC-V?

_____2____

ג. מה מספר המחזורים המינימלי הנדרש לביצוע פסאודו פקודה זו במעבד Multicycle RISC-V?

_____10_____

ד. כעת ניתן לבצע שינויים במעבד הכוללים הוספת/הרחבת בוררים, והוספת חיווטים. מה מספר המחזורים **המינימלי** הנדרש לביצוע פקודה זו **כפקודה אמיתית** במעבד Multicycle RISC-V?

_____6_____

שאלה 16 (8 נקודות):

שירי אוהבת לאפות ולכן החליטה להגשים את חלומה ולפתוח קונדיטוריה. לרוע מזלה, דווקא בתקופה ההצלחה, התפרצה הקורונה וכעת עליה להישמע להנחיות משרד הבריאות. הקונדיטוריה פועלת משעה $02:00$ לפנות בוקר ועד $15:00$. לפי ההנחיות, על שירי לחטא את הקונדיטוריה בשעת הפתיחה ומידי 4 שעות החל מרגע זה. בנוסף, עליה לחטא גם בשעת העומס, $08:00$.
שירי פנתה אליכם בבקשה לעזרה. בנו מערכת צירופית בעלת 4 כניסות $f(x,y,z,w)$ המפיקה '1' כאשר על שירי לחטא את המטבח.

רשמו את הפונקציה המצומצמת ביותר אשר מתארת את התנהגות המערכת והשלימו את מפת הקרנו הבאה:

ZW \ XY	00	01	11	10
00				
01				
11				
10				

$f(x,y,z,w) =$

פתרון:

נשים לב כי בשעות 2,6,10,14 שירי חייבת לחטא את המטבח לפי ההנחיות.
מכיוון שהמאפייה נפתחת רק ב-2, ב00 ו-01 נשים *dc* בטבלת האמת.
בשעה 08:00 גם כן, שירי צריכה לחטא, ולכן גם שם שמנו '1' בטבלת האמת.

נקבל:

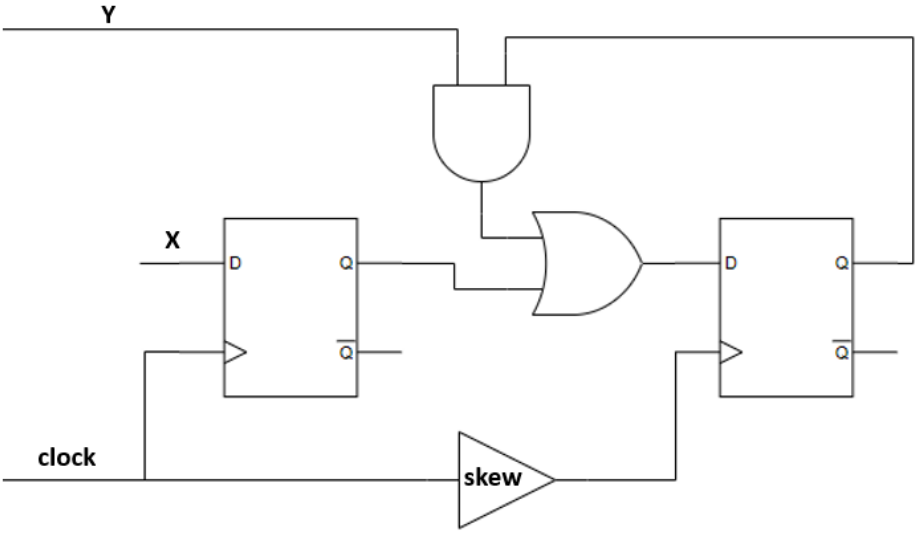
	X	y	z	w	f
0	0	0	0	0	*F*
1	0	0	0	1	*F*
2	0	0	1	0	1
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	1
7	0	1	1	1	
8	1	0	0	0	1
9	1	0	0	1	
10	1	0	1	0	1
11	1	0	1	1	
12	1	1	0	0	
13	1	1	0	1	
14	1	1	1	0	1
15	1	1	1	1	

zw\xy	00	01	11	10
00	*F*			1
01	*F*			
11				
10	1	1	1	1

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

תרגיל 17 (8 נקודות)

נתון המעגל הבא:



זמן המחזור של השעון הוא:

$T_{cycle} = 25ns$

בנוסף נתונים זמני ההשהיה הבאים:

	FF	OR	AND
T_{cCQ}/T_{cd}	$1ns$	$1ns$	$1ns$
T_{pCQ}/T_{pd}	$4ns$	$3ns$	$5ns$
T_{su}	$7ns$		
T_{hold}	$4ns$		

נתון כי הכניסות עומדות בתנאי $hold$ ו- $setup$.

בין שני ה-FF קיים $SKEW$ בשעון שערכו הוא t_{skew}.

א. עבור $t_{skew} = 0$, האם המעגל עומד במשטר הזמנים הדינאמי?

ב. מבין הערכים הבאים, מהו הערך של t_{skew} עבורו המעגל עומד במשטר הזמנים הדינאמי?

1. $-2ns$
2. $3ns$
3. $0ns$
4. אף ערך
5. $1ns$

ג. כעת נתון כי $t_{skew} = 1ns$.

על מנת לאפשר פעילות תקינה של המעגל, הוחלט לבצע שימוש בחוצץ (*buffer*) בעל הפרמטרים הבאים:

$$t_{buffer} = t_{cd}(buffer) = t_{pd}(buffer)$$

הוסיפו את החוצץ במקום המתאים בשרטוט וקבעו את זמן ההשהיה המינימלי והמקסימלי של החוצץ, המאפשרים עמידה במשטר הזמנים הדינאמי?

א. פועל בצורה תקינה	כן / לא
ב. מבין הערכים הבאים, מהו הערך של t_{skew} עבורו המעגל עומד במשטר הזמנים הדינאמי? 1. $-2ns$ 2. $3ns$ 3. $0ns$ 4. אף ערך 5. $1ns$	
ג.	$\leq t_{buffer} \leq$

פתרון:

נבדוק את המסלול מ*FF1* ל *FF2*.

$FF1 \rightarrow FF2$

תנאי *hold*:

$$T_{hold}(FF) + T_{skew} \leq T_{cd}(FF) + T_{cd}(OR)$$
$$T_{skew} \leq 1 + 1 - 4 = -2$$

נבדוק את המסלול מ*FF2* ל *FF2*.

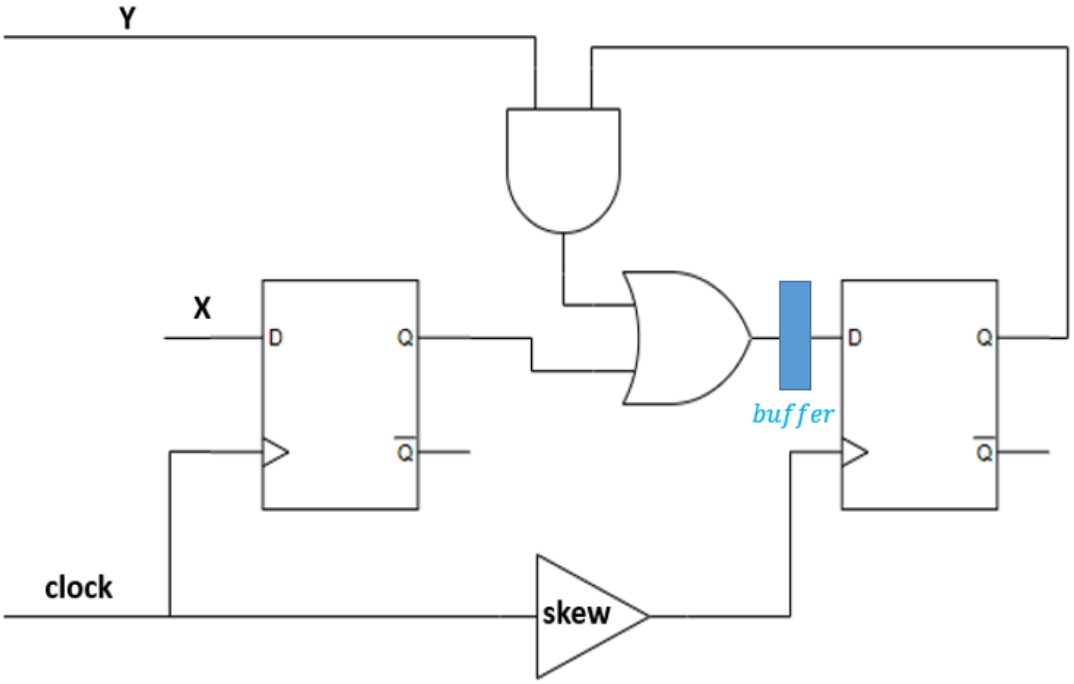
$FF2 \rightarrow FF2$

תנאי *hold*:

$$T_{hold} \leq T_{cd}(FF) + T_{cd}(OR) + T_{cd}(AND)$$
$$4 \leq 1 + 1 + 1 = 3$$

כלומר התנאי לא מתקיים (ללא תלות ב skew).

נוסיף את הבאפר לשני המסלולים (בהינתן כי שניהם לא עומדים בתנאי *hold*), נשים לב שהדבר אפשרי רק אם נוסיף את הבאפר לכניסת ה *FF* השני. נבדוק האם המעגל פועל בצורה תקינה.



נבדוק את המסלול מ*FF1* ל *FF2*.

$FF1 \rightarrow FF2$

תנאי *hold*:

$$\mathrm{T}_{hold}(FF) + T_{skew} \leq T_{cd}(FF) + T_{cd}(OR) + T_{buffer}$$
$$4 + 1 \leq 1 + 1 + T_{buffer} \rightarrow T_{buffer} \geq 3$$

תנאי *setup*:

$$T_{cycle} + T_{skew} \geq T_{pd}(FF) + T_{pd}(OR) + T_{buffer} + T_{su}(FF)$$
$$25 + 1 \geq 4 + 3 + 7 + T_{buffer} \rightarrow \ T_{buffer} \leq 12ns$$

נבדוק את המסלול מ*FF2* ל *FF2*.

$FF2 \rightarrow FF2$

תנאי *hold*:

$$\mathrm{T}_{hold} \leq T_{cd}(FF) + T_{cd}(OR) + T_{cd}(AND) + T_{buffer}$$
$$4 \leq 1 + 1 + 1 + T_{buffer} \rightarrow T_{buffer} \geq 1$$

תנאי *setup*:

$$T_{cycle} \geq T_{pd}(FF) + T_{pd}(AND) + T_{pd}(OR) + T_{su}(FF) + T_{buffer}$$
$$25 \geq 4 + 5 + 3 + 7 + T_{buffer} \rightarrow T_{buffer} \leq 6$$

נאחד את כל התנאים ונקבל:

$$3\mathrm{ns} \leq T_{buffer} \leq 6ns$$