שאלה

נתונה המשוואה הבאה: $(0101)_{a+b}-(0101)_{a-b}=(0100)_a$. כמו כן נתון כי a,b שלמים ומקיימים: $b>1,\ a>b$. בחר את המשפט הנכון:

א. המשוואה מתקיימת לכל a,b.

ב. המשוואה מתקיימת לזוג אחד בלבד של a,b.

ג. המשוואה מתקיימת עבור אינספור צירופים אפשריים של a,b, המקיימים את התנאי a=2b.

ד. המשוואה מתקיימת עבור אינספור צירופים אפשריים של a,b, המקיימים את התנאי a=4b.

ה. המשוואה לא מתקיימת עבור אף צירוף של a,b.

תשובה: **ד'**.

נמיר את כל המשוואה לבסיס עשרוני:

$(a+b)^2+1-(a-b)^2-1=a^2$

$a^2+2ab+b^2+1-a^2+2ab-b^2-1=a^2$

$4ab=a^2$

$a=4b$

לכן התשובה הנכונה היא תשובה ד'.

שאלה

נתונות 3 פונקציות ב-4 משתנים $f(x,y,z,w),g(x,y,z,w),h(x,y,z,w)$. ידוע כי:

- $f(x,y,z,w)$ מהווה מערכת פעולות שלמה
- $g(x,y,z,w)$ לא מהווה מערכת פעולות שלמה אך מהווה מערכת פעולות חצי שלמה
- $h(x,y,z,w)$ לא מהווה מערכת פעולות שלמה וגם לא מערכת פעולות חצי שלמה

מעוניינים להרכיב כל אחת מהפונקציות הנתונות בעזרת שתי הפונקציות האחרות, ללא הגבלה על מספר הפונקציות שניתן להשתמש בהרכבה, וללא שימוש בקבועים. מה ניתן לומר <u>בוודאות תמיד</u>?

א. ניתן להרכיב את f באמצעות מספר בלתי מוגבל של הפונקציות g
ב. ניתן להרכיב את g באמצעות מספר בלתי מוגבל של הפונקציות h
ג. ניתן להרכיב את h באמצעות מספר בלתי מוגבל של הפונקציות f
ד. לא ניתן לומר דבר ללא ידיעת הפונקציות הספציפיות

פתרון

הפונקציה f מהווה מערכת פעולות שלמה ולכן היא מממשת את כל הפונקציות בפרט את h

שאלה

נתון קטע הקוד הבא בשפת אסמבלי:

```
0x1000          addi s1, x0, 0x100
0x1004   loop:  add s0, s1, x0
0x1008          lw s1, 0(s0)
0x100C          bne s1, x0, loop
0x1010   exit:
```

נתון תוכן הזיכרון:

address	value
0x100	0x2010
0x104	0x2008
0x108	0x0
0x10C	0x2000
0x110	0x2004

address	value
0x2000	0x110
0x2004	0x104
0x2008	0x0
0x200C	0x0
0x2010	0x10C

מה יהיה תוכן הרגיסטר $S0$ לאחר סיום ריצת התוכנית (מגיעים לכתובת 0x1010) ?

א- 0x108
ב- 0x2008
ג- 0x200C
ד- התוכנית לא תסתיים (לולאה אינסופית)

פתרון: תשובה ב
הקוד בשפת C:

```
int* s1 = 0x100, *s0;
do {
        s0 = s1;
        s1 = *s0;
} while( s1 != 0 );
```

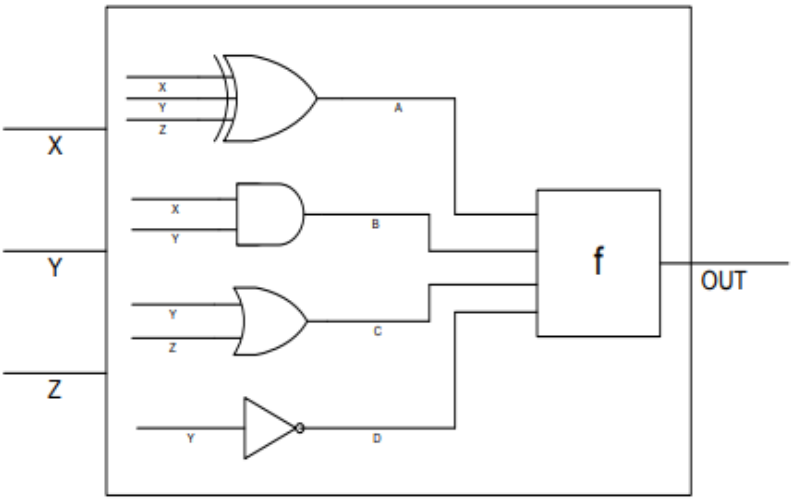
בכל איטרציה אנחנו ניגשים לזיכרון כדי לקרוא את הכתובת אליה אנחנו ניגשים באיטרציה הבאה.
הלולאה מסתיימת ברגע שקוראים את הערך 0. ולכן התוכן של s0 בסוף ריצת התוכנית יהיה הכתובת שמכילה את ה-0 הראשון שניגשים אליו:

Iteration	S0	S1
init - 0	0xdeadbeef	0x100
1	0x100	0x2010
2	0x2010	0x10C
3	0x10C	0x2000
4	0x2000	0x110
5	0x110	0x2004
6	0x2004	0x104
7	0x104	0x2008
8	0x2008	0

שאלה

ברצוננו לתכנן מערכת המזהה אם מספר שלם אי שלילי בייצוג unsigend קטן ממש מ-5. למערכת 3 כניסות y,x ו-z, כאשר x הינו ה-MSB.
אם המספר המיוצג על ידי הכניסות קטן ממש מ-5, היציאה צריכה להיות '1', אחרת היציאה צריכה להיות '0'.

חלק מהמערכת נתון אך יחידה אחת הממשת את הפונקציה f לא ידועה:



מהו הייצוג המינימלי של הפונקציה f בצורה של סכום מכפלות?

א. f = C'+AB'+B'D'

ב. f = C'+AB'+B'D'+BD

ג. f = B'C'D+ B'CD' +AB'D

ד. f = A'B'+A'C'

ה. f = C'+AB'+B'CD'

פתרון : א'

טבלת אמת של OUT ביחס לכניסות X,Y,Z :

X	Y	Z	A	B	C	D	OUT
0	0	0	0	0	0	1	1
0	0	1	1	0	1	1	1
0	1	0	1	0	1	0	1
0	1	1	0	0	1	0	1
1	0	0	1	0	0	1	1
1	0	1	0	0	1	1	0
1	1	0	0	1	1	0	0
1	1	1	1	1	1	0	0

טבלת אמת של OUT ביחס לכניסות A,B,C,D :

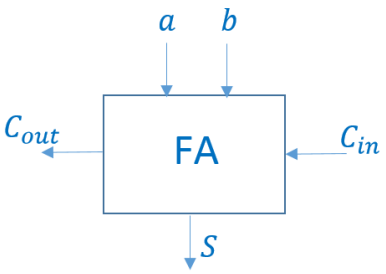
A	B	C	D	OUT
0	0	0	0	Ø
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	Ø
0	1	0	1	Ø
0	1	1	0	0
0	1	1	1	Ø
1	0	0	0	Ø
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	Ø
1	1	0	1	Ø
1	1	1	0	0
1	1	1	1	Ø

מפת קרנו :

cd \ ab	00	01	11	10
00	Ø	Ø	Ø	Ø
01	1	Ø	Ø	1
11		Ø	Ø	1
10	1			1

שאלה

ברשותכם כמות אינסוית של יחידות FA שראיתם בתרגול:



נתונים זמני ההשהיה של ה-FA:

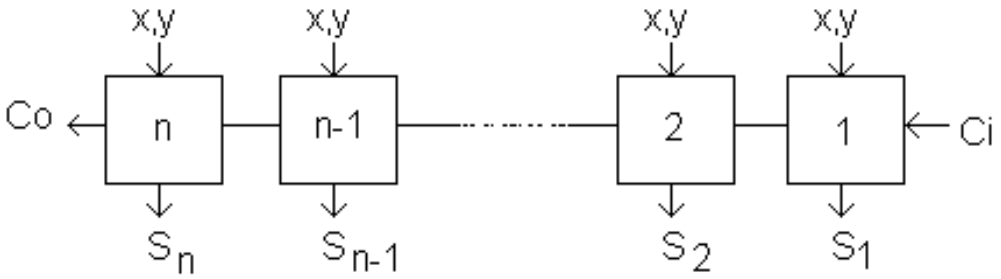
$path$	t_{pd}
$a, b \rightarrow S$	20 ns
$a, b \rightarrow C_{out}$	5 ns
$C_{in} \rightarrow S$	4 ns
$C_{in} \rightarrow C_{out}$	2 ns

מהו זמן ההשהיה של סוכם ברוחב 7 ביטים הממומש ע"י שרשור יחידות ה-FA הנתונות?

א- $18\ ns$
ב- $19\ ns$
ג- $20\ ns$
ד- $21\ ns$

פתרון: תשובה ג

צריך להשתמש ב-7 רכיבים של FA ולשרשר אותם כך שה-COUT של ה-FA ה-i יחובר ל-CIN של ה-FA ה- i+1 באופן הבא:



בציור לעיל n=7

$$t_{pd} = \max\{t_{pd}^{*}, t_{pd}(x, y \to S)\}$$

$$\begin{aligned} t_{pd}^{*} = \max\{&t_{pd}(x, y \to C_{out}), t_{pd}(C_{in} \to C_{out})\} \\ &+(n-2) \cdot t_{pd}(C_{in} \to C_{out}) \\ &+ \max\{t_{pd}(C_{in} \to C_{out}), t_{pd}(C_{in} \to S)\} \end{aligned}$$

$$t_{pd}^{*} = \max\{5, 2\} + (7-2) \cdot 2 + \max\{2, 4\} = 19\ ns$$

$$t_{pd} = \max\{t_{pd}^{*}, t_{pd}(x, y \to S)\} = \max\{19, 20\} = 20\ ns$$

שאלה

נתונות מערכות העקיבה הבאות:

1. מערכת המקבלת כניסה אחת ומוציאה 1 אם"ם היא מזהה את הסדרה "10101".
2. מערכת המקבלת מספר החל מה-MSB ומוציאה 1 אם"ם המספר שהתקבל עד כה הוא כפולה של 8.
3. מערכת המקבלת כניסה אחת ומוציאה 1 אם"ם הערך של הכניסה במחזור הקודם היה שווה לערך שלה לפני 2 מחזורים.
4. מערכת המקבלת כניסה אחת ומוציאה 1 אם"ם החצי השמאלי של המספר שהתקבל עד כה שווה לחציו הימני.

אילו מהמערכות ניתן לממש **כמכונת מילי** עם שני FF-ים **לכל היותר**?:

א. 3
ב. 2 ו-3
ג. 1, 2 ו-3
ד. 1 ו-2

הערה: התשובה לא משתנה גם אם היו מבקשים לממש את המערכת כמכונת מור.
פתרון: תשובה ב

1- ניתן לממש את המכונה כמכונת מילי עם 5 מצבים (כמו שראינו בתרגול), במכונת מור יהיו לפחות 5 מצבים (במקרה הזה יהיו 6 מצבים). ולכן צריך לפחות 3 פליפ-פלופים כדי לממש את המכונה.
2- בתרגול ראינו שאלה דומה, צריך לזהות את הרצף 000, במילי צריך 3 מצבים וכדי לעבור למור צריך להוסיף עוד מצב, סה"כ 4 מצבים במכונת מור ולכן נצטרך 2 פליפ-פלופים.
3- צריך לזכור את כל הקומבינציות של שני הביטים האחרונים שנכנבו למערכת (ללא תלות בכניסה הנוכחית) ויש לנו סה"כ 4 קומבינציות שונות ולכן מספיק 2 פליפ-פלופים.
4- לא ניתן לממש מערכת כזו כמכונת מצבים סופית (לא כמילי וגם לא כמור) – צריך אינסוף מצבים. (מצורפת הוכחה בעמוד הבא להעשרה – בקורס אין הוכחות).

הגדרה: נאמר שמערכת עקיבה סינכרונית מקבלת סדרה, אם כאשר היא מופעלת מן המצב ההתחלתי שלה ומוזנת בסדרה זו, היא מייצרת פלט 1 עם קבלת התו האחרון של הסדרה בקלט.

דוגמא: הוכח שלא קיימת מכונה עם מספר מצבים סופי, המקבלת את כל הפלינדרומים, אך לא מקבלת אף סדרה שאינה פלינדרום. (פלינדרום הוא סדרה שהיא שיקוף של עצמה. לדוגמא : 01110, 1001).

הוכחה: בדרך השלילה:

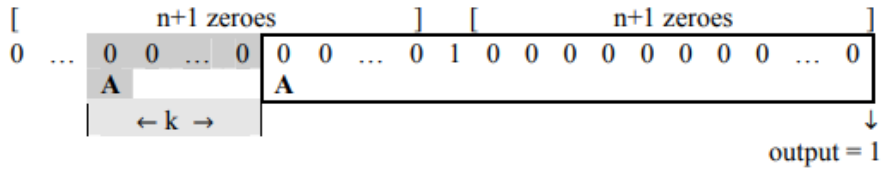
נניח שקיימת מכונה כזו בעלת n מצבים.

מאחר והמכונה מקבלת את כל הפלינדרומים, אזי היא תקבל גם את הפלינדרום :

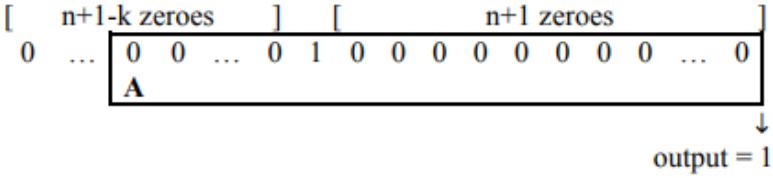
$$\underbrace{00\ldots0}_{n+1}1\underbrace{00\ldots0}_{n+1}$$

עבור $n+1$ האפסים הראשונים המכונה תעבור בהכרח דרך מצב מסוים לפחות פעמיים.

נסמן מצב זה ב-A ונניח שהמרחק בין שתי ההופעות של מצב A הוא k מחזורי שעון.



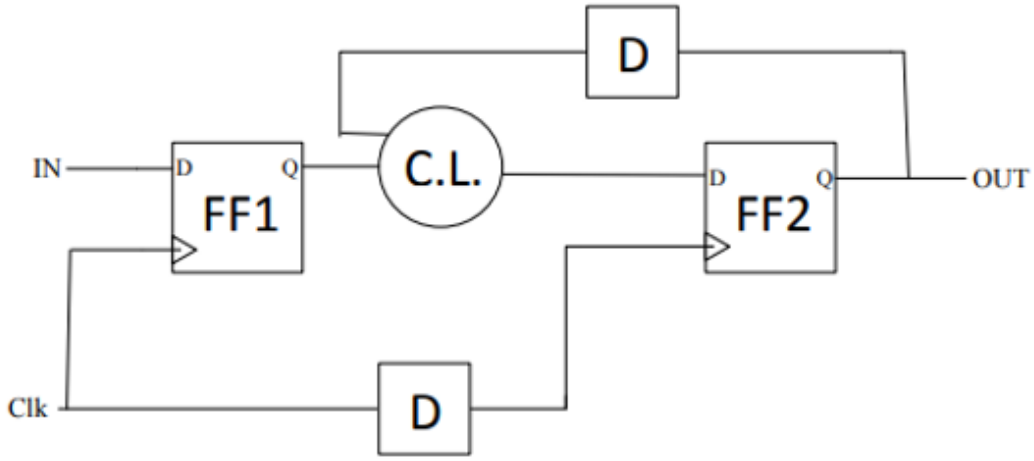
כעת נמחק ממילת הקלט את k האפסים המסומנים באפור, ונזין את המילה המתקבלת למכונה (החל מהמצב ההתחלתי). המילה שקבלנו (**[n+1 אפסים] 1 [n+1-k אפסים]**) אינה פלינדרום ולכן פלט המכונה עבור מילה זו אמור להיות 0. כעת נראה שפלט המכונה שיתקבל יהיה 1 ולכן המכונה שהנחנו את קיומה לא עובדת כראוי, ומכאן שלא קיימת מכונה כזו.



נשים לב (מסומן בריבוע) שבשלב מסוים עבור הקלט השני נגיע למצב A ולאחריו סדרת קלט זהה למסומן עבור מלת הקלט הראשונה, לכן משלב זה ואילך, עבור מילת הקלט השניה, המכונה תנהג בדיוק באותה דרך (אותם מעברי מצבים ואותן היציאות) כפי שנהגה עבור מילת הקלט הראשונה, ולכן מוצא המערכת בסיום קליטת מילת הקלט יהיה 1 אע"פ שמילת הקלט אינה פלינדרום.

שאלה

נתון המעגל הבא :



הזמנים של הלוגיקה וכן של רכיבי הזיכרון נתונים בטבלה הבאה (ב-ns) :

Thold	Tsetup	Tcd	Tpd	
2	4	5	16	FF1
1	3	3	14	FF2
-	-	7	15	C.L.

היחידה .C.L מהווה יחידת לוגיקה צירופית כלשהי (ללא רכיבי זיכרון).
היחידה D מהווה יחידת השהייה המורכבת **ממספר זוגי של מהפכים** . עבור m מהפכים בתוך היחידה D מתקיים :

$$T_{cd}(D) = 2m, T_{pd}(D) = 3m$$

כמו כן, נתון כי השעון בעל זמן מחזור של 42ns . מהו המספר **המקסימלי** של מהפכים שניתן להרכיב בתוך היחידה D **מבלי לפגוע** בפעולתו התקינה של המעגל ? :

א- 0
ב- 1
ג- 2
ד- 3

פתרון: תשובה ב

מספר המהפכים חייב להיות זוגי כדי לא לפגוע בפעולתו התקינה של המעגל (כדי שלא נשנה את הפונקציונליות של הרכיב במסלול מפליפ-פלופ 2 לעצמו. ולכן תשובות ב' ו- ד' לא נכונות.

בהנחה שהכניסה מקיימת את תנאי setup & hold. המסלולים הרלוונטיים שצריך לבדוק הם:

$$FF1 \rightarrow FF2 \quad \& \quad FF2 \rightarrow FF2$$

FF1 → FF2:

$setup$: $t_{pC\rightarrow Q}(FF1) + t_{pd}(CL) + t_{su}(FF2) \leq T_{clk} + t_{cd}(D)$

$16 + 15 + 3 \leq 42 + 2m$

$-8 \leq m \quad \rightarrow \quad m \geq 0$

$hold$: $t_{cd}(FF1) + t_{cd}(CL) \geq t_{hold}(FF2) + t_{pd}(D)$

$5 + 7 \geq 1 + 3m$

$$\left\lfloor \frac{11}{3} \right\rfloor \geq m \quad \rightarrow \quad m \leq 3$$

FF2 → FF2:

$setup$: $t_{pC\rightarrow Q}(FF2) + t_{pd}(D) + t_{pd}(CL) + t_{su}(FF2) \leq T_{clk}$

$$14 + 3m + 15 + 3 \leq 42$$

$$m \leq \left\lfloor \frac{10}{3} \right\rfloor \quad \rightarrow \quad m \leq 3$$

$hold$: $t_{cd}(FF2) + t_{cd}(D) + t_{cd}(CL) \geq t_{hold}(FF2)$

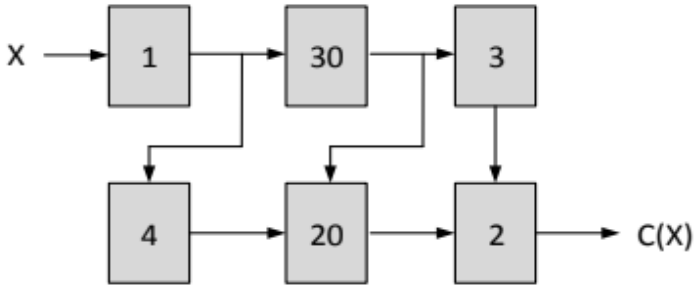
$$3 + 2m + 7 \geq 1$$

$$m \geq -3 \quad \rightarrow \quad m \geq 0$$

כלומר קיבלנו ש- $0 \leq m \leq 3$, המספר **<u>הזוגי</u>** הגדול ביותר בטווח זה הוא 2.

שאלה

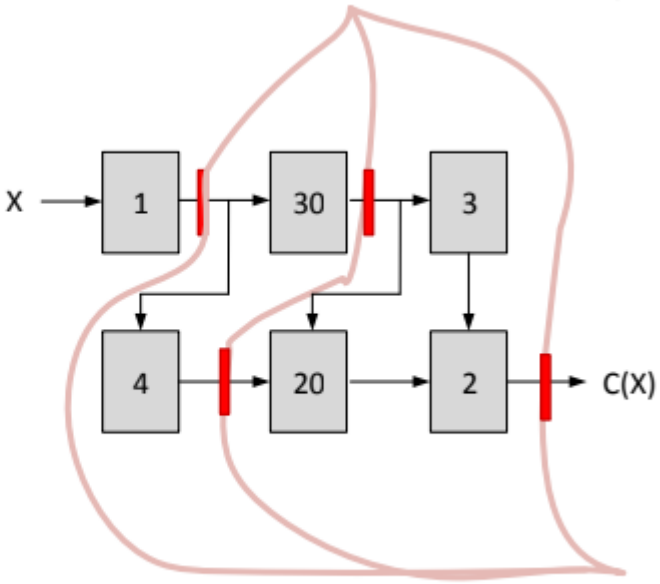
נתונה המערכת הצירופית הבאה:



צנרו את המעגל לקבלת Throughput מכסימלי. הניחו רגיסטרים אידיאליים חסרי השהייה (Tsetup=Thold=TpCQ=0). לשם כך נדרש להשתמש במספר רגיסטרים מינימלי אפשרי.
מהם ה – Latency וה- Throughput של המעגל המצונר?

א.	1/30	Throughput=	90	Latency=
ב.	1/30	Throughput=	120	Latency=
ג.	1/31	Throughput=	62	Latency=
ד.	1/53	Throughput=	53	Latency=
ה.	1/50	Throughput=	100	Latency=

פתרון:



Latency = 3*30 = 90.
Throughput = 1/30.

שאלה

נתון קטע הקוד הבא:

```
main:     addi sp, sp, -4
          addi a0, x0, 0x200
          addi a1, x0, 0x5
          sw ra, 0(sp)
          jal foo
          lw ra, 0(sp)
          addi sp, sp, 4
          ret

foo:      addi sp, sp, -8

          bne a1, x0, recall
          add a0, x0, x0
          j done

recall:   lw t0, 0(a0)
          addi a0, a0, 4
          addi a1, a1, -1
          sw ra, 0(sp)
          sw t0, 4(sp)
          jal foo
          lw ra, 0(sp)
          lw t0, 4(sp)
          add a0, a0, t0

done:     addi sp, sp, 8
          ret
```

ונתון מערך של מספרים שלמים arr (כל מספר תופס 4 בתים בזיכרון)

$$arr = \{1, \quad 2, \quad 3, \quad 4, \quad 5\}$$

כתובת תחילת המערך היא 0x200.

מהו תוכן הרגיסטר a0 לפני ביצוע הפקודה ret בפונקציה main?

א- 10

ב- 15

ג- 20

ד- לא ניתן לדעת כי התוכנית לא מקיימת את קונבנציית הקריאה לפונקציות

פתרון:

הקוד בשפת C:

```
int foo(int* arr, int n) {
	if(n == 0) {
		return 0;
	}
	return arr[0] + foo(arr + 1, n-1);   // (compiler: arr + 1 * sizeof(int) = arr+4)
}


int main() {

	Int arr[5] = {1, 2, 3, 4, 5};
	foo(arr, 5);
	return;
}
```

הפונקציה FOO עוברת על המערך בצורה רקורסיבית ומחזירה את הסכום של איברי המערך.

שאלה

נתון מעבד מסוג single cycle RISCV מעבד נוסף מסוג multi cycle RISCV.
בדיקה העלתה שבשניהם יש תקלה:
כאשר ALUSel=sub, רכית ה- ALU מבצע דווקא add.

מבין התשובות הבאות, ובהינתן תקלה זו, בחרו את התשובה הנכונה:

א- **ניתן** לבצע פקודת BEQ על single cycle אך **לא ניתן** לבצע BEQ על multi cycle
ב- **לא ניתן** לבצע פקודת BEQ על single cycle אך **ניתן** לבצע BEQ על multi cycle
ג- **ניתן** לבצע פקודת BEQ על single cycle **וגם ניתן** לבצע BEQ על multi cycle
ד- **לא ניתן** לבצע פקודת BEQ על single cycle **וגם לא ניתן** לבצע BEQ על multi cycle

פתרון: תשובה א'

ב- single cycle, בפקודת BEQ ה- ALU צריך לחשב את $PC = PC + imm$ ולכן יעבוד.

ב- multi cycle יש צורך לחשב גם את ההשוואה בין הרגיסטרים ע"י ביצוע פעולת חיסור, ולכן הפקודה לא תוכל לרוץ כשורה.