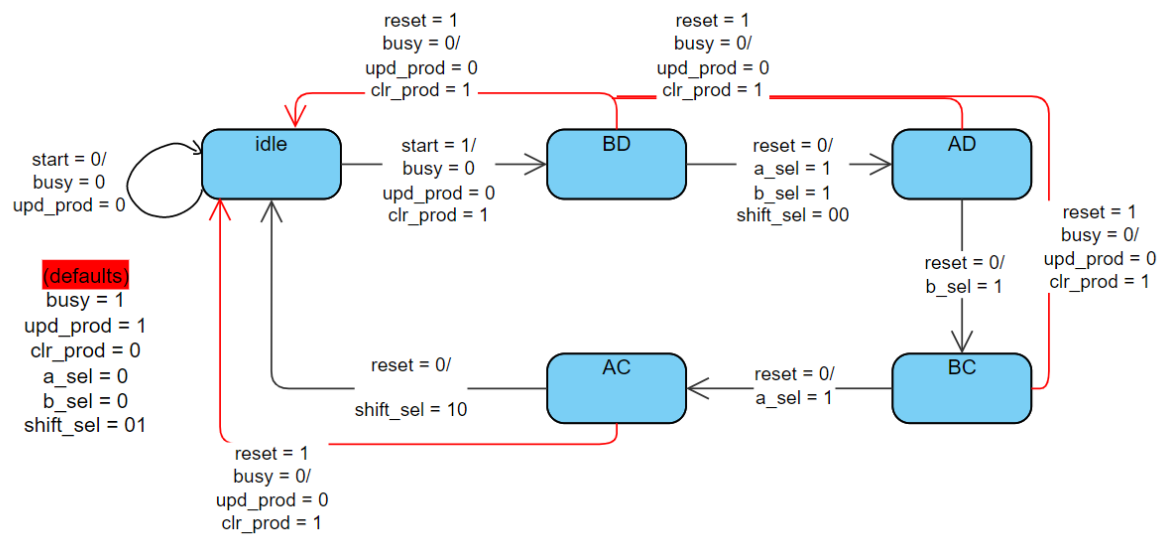


209146471	אלון הרטמן
207923566	תומר אברשקין

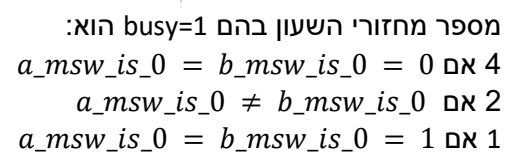
## חלק יבש

2.1

דיאגרמת המצבים של מכונת המצבים:



החישוב יקח 4 מחזורי שעון, כלומר במשך 4 מחזורי שעון היציאה busy תהיה על ערך '1'.



### 2.3

נתאר את האלגוריתם באמצעות pseudo code:

נסמן את הקלטים בתור  $a = a_{N-1}a_{N-2} \dots a_1a_0$ ,  $b = b_{N-1}b_{N-2} \dots b_1b_0$ , כאשר גודל  $a, b$  הוא  $8N$ , וגודל כל  $a_i, b_i$  הוא 8 סיביות.

כמו כן נסמן  $shift\_left(input, bits)$  בתור פונקציה המקבלת קלט  $input$  ועושה לו הזזה  $bits$  סיביות שמאלה.

לבסוף נסמן  $a_{2j+1}a_{2j}$  בתור המספר הבינארי המיוצג על ידי שרשרתם.

```
int result, i, j;
for(i = 0; i < N; ++i) {
    for(j = 0; j < N/2; ++j) {
        int temp = shift_left(b_i * (a_{2j+1}a_{2j}), 8i + 8 * 2j);
        result += temp;
    }
}
```

הסיבוכיות של האלגוריתם היא  $O(N^2)$  כיוון שרצים על לולאה שמתבצעת  $O(N)$  פעמים ובתוכה רצים על לולאה שמתבצעת  $O(N)$  פעמים גם היא. בסך הכל -  $O(N^2)$ .

Run Step Prev Reset Dump

Machine Code	Basic Code	Original Code
0x10000e17	auipc x28 65536	la t3, a
0x000e0e13	addi x28 x28 0	la t3, a
0x000e2e03	lw x28 0(x28)	lw t3, 0(t3)
0x10000e97	auipc x29 65536	la t4, b
0xff9e8e93	addi x29 x29 -8	la t4, b
0x000eae83	lw x29 0(x29)	lw t4, 0(t4)
0x0000fb3	add x31 x0 x0	add t6, x0, x0
0x0ff06293	ori x5 x0 255	ori t0, x0, 0xff
0x00829293	slli x5 x5 8	slli t0, t0, 8
0x0ff2e293	ori x5 x5 255	ori t0, t0, 0xff
0x00829293	slli x5 x5 8	slli t0, t0, 8
0x0ff2e293	ori x5 x5 255	ori t0, t0, 0xff
0x00000333	add x6 x0 x0	add t1, x0, x0 # t1 = i

195065129

Run Step Prev Reset Dump

Machine Code	Basic Code	Original Code
0x10000e17	auipc x28 65536	la t3, a
0x000e0e13	addi x28 x28 0	la t3, a
0x000e2e03	lw x28 0(x28)	lw t3, 0(t3)
0x10000e97	auipc x29 65536	la t4, b
0xff9e8e93	addi x29 x29 -8	la t4, b
0x000eae83	lw x29 0(x29)	lw t4, 0(t4)
0x0000fb3	add x31 x0 x0	add t6, x0, x0
0x0ff06293	ori x5 x0 255	ori t0, x0, 0xff
0x00829293	slli x5 x5 8	slli t0, t0, 8
0x0ff2e293	ori x5 x5 255	ori t0, t0, 0xff
0x00829293	slli x5 x5 8	slli t0, t0, 8
0x0ff2e293	ori x5 x5 255	ori t0, t0, 0xff
0x00000333	add x6 x0 x0	add t1, x0, x0 # t1 = i

195065129

Registers Memory

zero	0x00000000
ra (x1)	0x00000000
sp (x2)	0x7ffff000
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0xffffffff
t1 (x6)	0x00000002
t2 (x7)	0x00000000
a0 (x8)	0x00000002
a1 (x9)	0x00000000
a0 (x10)	0x0000000e
a1 (x11)	0xb8e07829
a2 (x12)	0x00000000
a3 (x13)	0x00000000
a4 (x14)	0x00000000
a5 (x15)	0x00000000
a5 (x15)	0x00000000
a6 (x16)	0x00000000
a7 (x17)	0x00000000
s2 (x18)	0x00000000
s3 (x19)	0x0000000e
s4 (x20)	0x00000008
s5 (x21)	0xb8e07829
s6 (x22)	0x00000000
s7 (x23)	0x00000000
s8 (x24)	0x00000000
s9 (x25)	0x00000000
s10 (x26)	0x00000000
s11 (x27)	0x00000000
t3 (x28)	0x000000ad
t4 (x29)	0x000000ad
t5 (x30)	0x00000000
t6 (x31)	0xb8e07829

פעולת הכפל לוקחת 25 מחזורי שעות, כלומר 25 פקודות מתבצעות.

## 2.5

נסמן  $a=xy$  – כאשר  $x$  אלו 8 הסיביות השמאליות של  $a$  ו- $y$  אלו הימניות ובאופן דומה  $b=wz$ .

נפריד למקרים:

- אם  $x=0, z=0$ : נקבל ש- $a, b$  הם באורך 8 ביטים. נחשב את תוצאת  $mul$  על  $a$  ו- $z$ , נכניס ל- $t_6$  ונסיים.
- אם  $x=0, z \neq 0$ : נקבל ש- $a$  באורך 8 ו- $b$  באורך 16. נחשב את תוצאת  $mul$  על  $b$  ו- $y$ , נכניס את התוצאה ל- $t_6$  ונסיים.
- אם  $x \neq 0, z = 0$ , נקבל ש- $a$  באורך 16 ו- $b$  באורך 8. נעשה כמו במקרה הראשון.

אחרת, נעשה את התוכנית המקורית מהסעיף הקודם.

קיבלנו שצריך להוסיף 3 בדיקות ולטפל בהם.

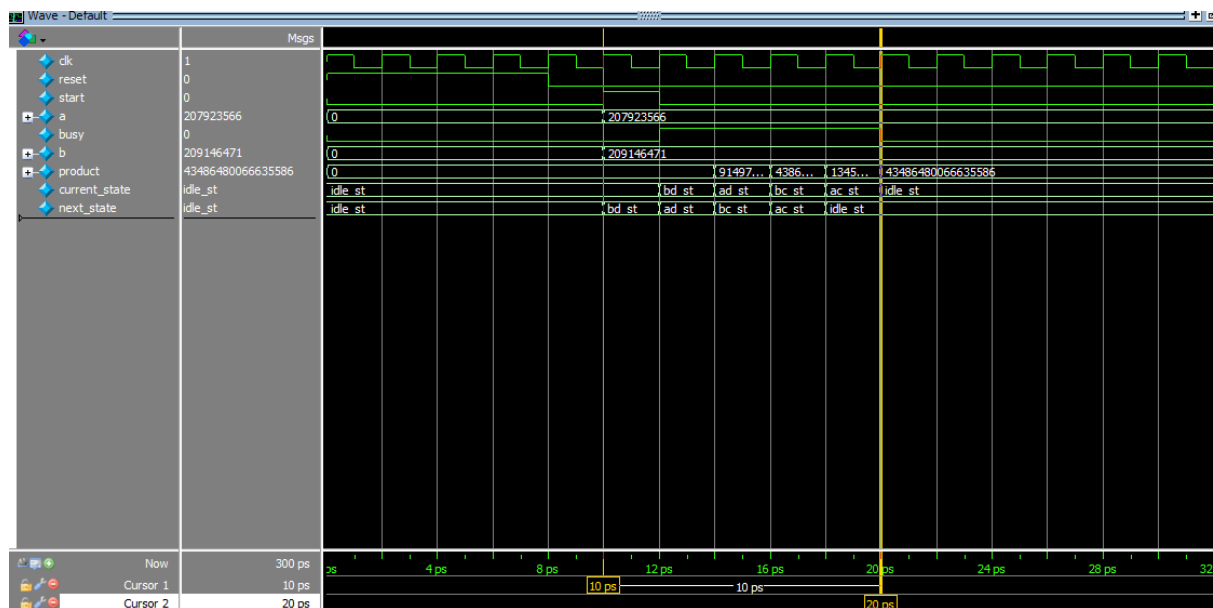
בכל אחד מהמקרים יהיה חישוב יותר פשוט ומהיר מאשר החישוב מהסעיף הקודם, אך אם אף אחד מהמקרים לא מתקיים, הוספנו בדיקות "מיותרות" לחישוב המקורי.

יחסית לכל המקרים האפשריים (יותר מ-2 מיליארד תוצאות אפשריות של קומבינציות של שני מספרים בינאריים באורך 16), המקרים לעיל, שחוסכים לנו זמן, הם בערך 33 מיליון מקרים. כלומר ברוב המקרים אנו ניתקל בחישוב ארוך יותר ולא נחסוך זמן. לכן השינוי הזה לא בהכרח משתלם.

## חלק רטוב

3.4

תוצאות הסימולציה:



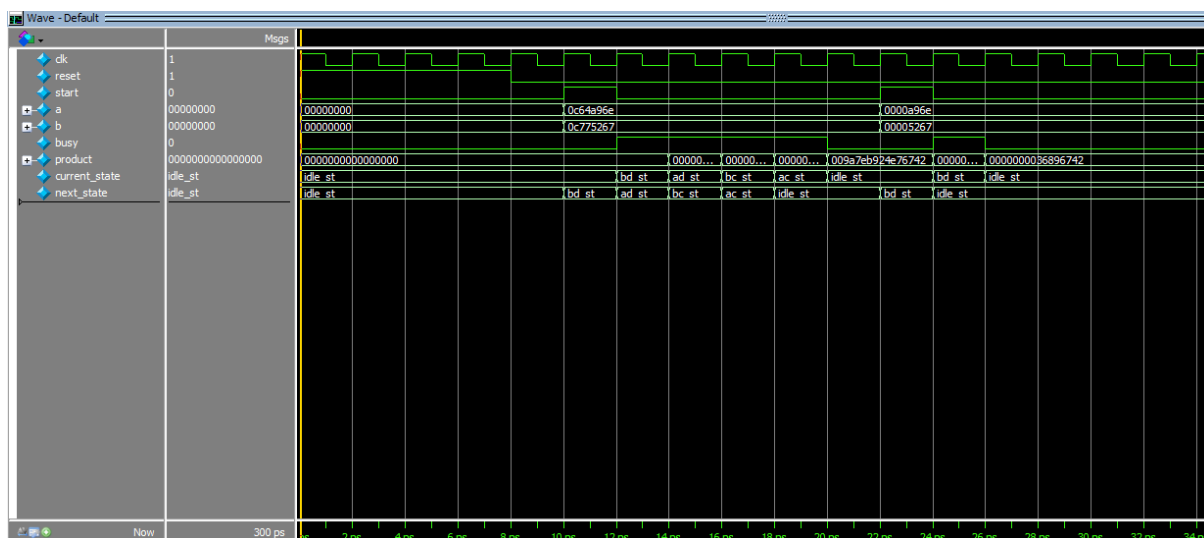
הסבר התוצאה:

כפי שניתן לראות, בתחילת הסימולציה הערכים מאותחל כמתבקש, לאחר 4 מחזורי שעון הסיגנל reset יורד ל-0 ולאחר מחזור שעון נוסף הסיגנל start עולה, הכניסות a ו-b מקבלות את ערכי תעודות הזהות שלנו והחישוב מתחיל.

ניתן לראות שהמכונה עוברת בכל המצבים כפי שתיארנו בסעיף 2.1 ובסוף החישוב, כלומר לאחר 4 מחזורי שעון מעליית busy ל-1, מתקבלת התוצאה הרצויה ונשמרת עד לעליית start או reset.

### 3.7

#### תוצאות הרצת הסימולציה:



#### הסבר התוצאה:

כפי שניתן לראות, בתחילת הסימולציה הערכים מאותחל כמתבקש, לאחר 4 מחזורי שעון הסיגנל reset יורד ל-0 ולאחר מחזור שעון נוסף הסיגנל start עולה, הכניסות a ו-b מקבלות את ערכי תעודות הזהות שלנו והחישוב מתחיל.

לאחר כ-5 מחזורי שעון המכונה מתייצבת על הפלט הרצוי, ולאחר מחזור שעון אחד מכניסים לה חישוב חדש בו שני חצאי המילה העליונות של תעודות הזהות מאופסים ולאחר מחזור שעון אחד התוצאה הרצויה מתקבלת בפלט. זה תואם את ניתוח הזמן שעשינו למערכת בסעיף 2.2.

בסוף כל חישוב, לאחר ש-busy יורד ל-0, מתקבלת התוצאה הרצויה ונשמרת עד לעליית start או reset.