# תכן לוגי ממוחשב של שבבים – ש.ב 2


**מגישים:**

**לידור זנו ת.ז: 313551186**

**גל קנר ת.ז: 207910738**


**תאריך: 17/01/2025**

1. נשמור את הכניסות, היציאות והצמתים (nodes) במפות על מנת שיהיה קל לגשת אליהם. במעגל שהוא flat יש לכל node שם ייחודי משלו, ולכן קל לגשת אליהם.
2. בדומה לתרגול 3 – נשתמש בתור משימות לevents, כאשר event הוא השמת חדש לצומת, ונשתמש בתור לשערים, אשר לפחות אחת הכניסות שלו השתנתה.

```
Event_Processor:
  For each Event E in Event_Queue do
    N:=Net Identifier of E
    Copy New value from E to the Net Table entry for N.
    For  each gate G in the fanout of N do
      If G is not already in the Gate_Queue Then
        Add G to Gate_Queue
      End If
    End For
    Remove E from Event_Queue
  EndFor
```

```
Gate_Processor:
  For Each Gate G in Gate_Queue do
    N:= The Output of G
    Simulate G, put the result in New_N
    If New_N is different from the current value of N then
      Create a new event E
      Net Identifier of E := N
      New value of E := New_N
      Add E to Event_Queue
    End If
    Remove G from Gate_Queue
  EndFor
```

3. הזמן הנתון הוא שניה 1, ניתן לראות זאת בקוד הvcd. $timescale 1s. בנוסף, הדיליי ממודל כשווה ל-0, אך יחד עם זאת הפליפ-פלופים מושפעים מהערך הקודם בכניסה, ולא מהערך החדש, למרות שזמן החישוב של הערך החדש מוזנח.
4. יש לנו הגבלה בסוג הרכיבים שכן הם נקבעים ע"י קובץ הווריילוג. סימולציה של מעגל עם רכיבים אחרים לא תעבוד.
5.

```
Simulation:
  GateInput();
  While Event_Queue is not empty do
```

```
        Event_Processor();
        If Gate_Queue is not empty then
          PrintIntermediateOutput();
          Gate_Processor();
        End If
      End While
    PrintFinalOutputToVCD();
```

לאחר שינוי קוד הווריילוג, שינינו את הקוד כך שנאתחל את הפליפ-פלופים למצב הגיוני ולא שרירותי. לקחנו בחשבון שהכניסות והיציאות שלו הן 0, וקבענו שכל פליפ פלופ מאותחל כך:

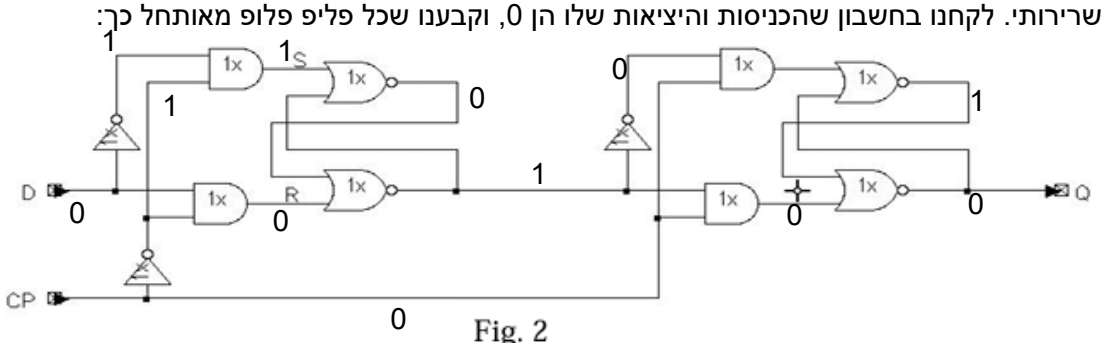

Fig. 2

בנוסף, שמנו דגש אל מניעת כפילויות, כלומר בתורים לא יהיה פעמיים את אותה צומת/אותו שער, ואם הערך החדש של הצומת זהה לקודם – לא יוצר אירוע חדש.

6.1

Event_process:

$$o(vectorSize \cdot nodes \cdot gates)$$

הסבר: במקרה הגרוע ביותר נקבל מעגל בו כל צמתים נכנסים לכל השערים, ולכן נצטרך לעבור על כל הצמתים, ומתוך כל צומת נעבור על כל השערים, וזו היא סיבוכיות של מספר הצמתים כפול מספר השערים. בנוסף, התהליך קורה מספר פעמים לפי אורך וקטור הכניסה. בנוסף, בדקנו, ואין פעולה שהסיבוכיות שלה גבוהה מזו, לכן בסה"כ הסיבוכיות שלנו היא $o(vectorSize \cdot nodes \cdot gates)$.

ביטוי מדוייק יותר, המתחשב יותר בשערים יהיה:
$o(gates \cdot vectorSize \cdot averageFanOut + gates)$.

6.2

*event_exist*, פונקציות ספרייה של *hcm*, וספריות אחרות. שאר הסימולציה כתובה באופן מפורש.

6.3

*Gate_exist*, פונקציות ספרייה של *hcm*, וספריות אחרות. שאר הסימולציה כתובה באופן מפורש.