 **Lab3 preparation report**

Digital System Design

Multi-Cycle CPU Design

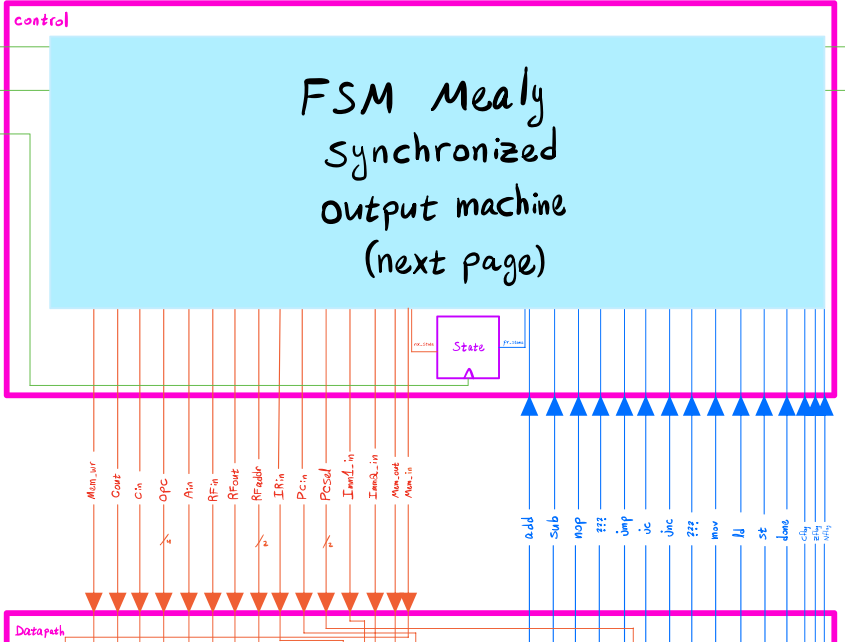
הפקולטה למדעי ההנדסה –   
מעבדת ארכיטקטורת מעבדים מתקדמת ומאיצי חומרה

עבור: רבוא חנניה

מגישים:  
אורן שור – 316365352  
טל שוורצברג – 316581537

**הסבר על המערכת**

1. **Control Unit**

מודול זה מממש מכונת מצבים סופית מסוג Mealy, כאשר כל מצב נוכחי (וקוי הבקרה האדומים הנשלחים ל-Datapath ) נקבעים לפי פונקציה של המצב הקודם וקוי הבקרה הכחולים  
  
  
אל מודול זה נכנסים בנוסף קוי בקרה מה-Testbench (ירוק), לצורכי סימולציה, וסיגנל שעון חיצוני.

**Control Unit - סימולציה:**  
בסימולציה הנ"ל הוגדרו מספר תהליכים –

* Gen\_clk – תהליך המייצר שעון מערכת.
* StartTb – תהליך המדמה רצף פקודות לביצוע, זו אחר זו ולפי הסדר   
  (משמאל לימין):  
  add, sub, nop, unused1, jmp, jc, jnc, unused2, mov, ld, st, done  
  בתהליך זה קוי הבקרה (הכחולים) מופעלים ידנית ומדמים את אופן פעולת ה-Datapath. כל קו בקרה פועל לפרק זמן התואם את מספר מחזורי השעון של הפעולה המתאימה לו.

להלן תוצאות הסימולציה, המתוארות על פני תרשים ה-wave :

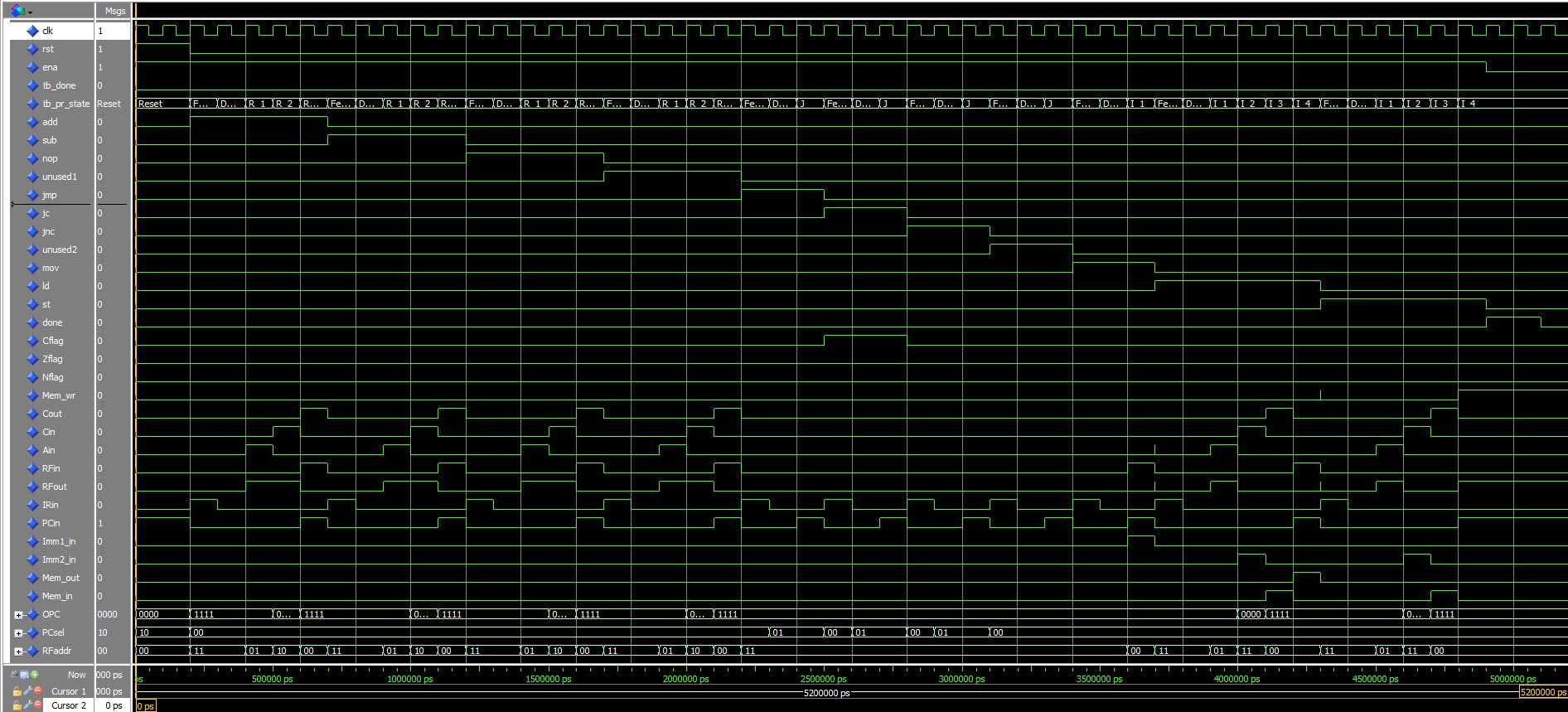
הסימולציה מתחילה במצב של ריסט,  
ניתן לראות ש:  
PC🡸 0..0

הדלקה ידנית של קו הבקרה   
done  
לסיום הסימולציה

הוראה מסוג   
J-Type  
לוקחת 3 סייקלים  
Fetch --> J

הוראה מסוג   
R-Type  
לוקחת 5 סייקלים  
Fetch -> R\_WB

שעון המערכת בזמן מחזור של 100 ns



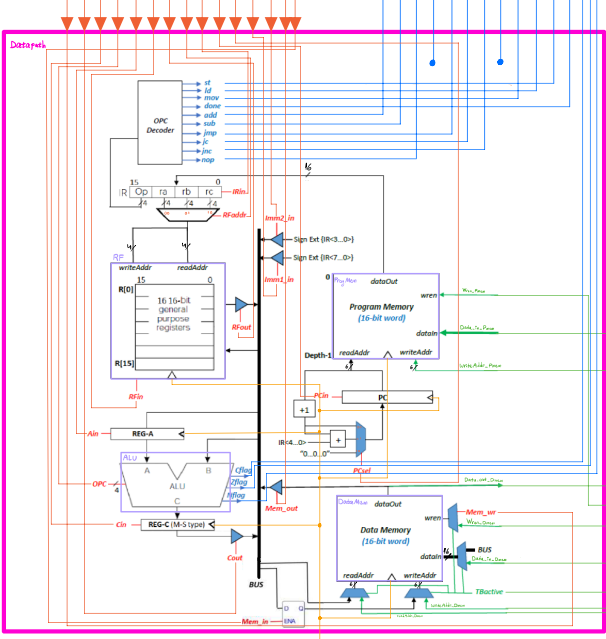
בהוראות   
J-Type  
שינוי ערך ה-  
PC  
לפי הקבוע הנתון במבנה הפקודה

קידום ה-  
PC  
מתבצע בסייקל האחרון של כל פקודה

ביצוע פעולת   
store:  
הכנסת המידע אל ה-  
DataMem  
ע"י הדלקת קו הבקרה המתאים

הוראת   
R-Type  
הכנסת הערך  
A 🡨 R[rb]  
שני קוי הבקרה המתאימים דולקים

1. **Datapath -**

מודול זה אחראי על מימוש המערכת, וכולל את האינטרקציה של כלל המודולים ב-CPU, כגון ALU, רכיבי זיכרון, באס משותף ועוד (מתוארים בהרחבה בהמשך).  
  
אל מודול זה נכנסים קוי בקרה (באדום) המגיעים מה-Control Unit שתיארנו קודם. במודול זה ישנם רכיבים אסינכרוניים (מעגלים צירופיים, Mux), ורכיבים סינכרוניים המתוזמנים לעבודה בעליית שעון (רגיסטרים, זכרונות).  
בכל שלב נתון, ובהינתן סטטוס קוי בקרה, מתבצעת פעולה אחת ביחידת ה-datapath, כחלק מסדרת פעולות בארכיטקטורת ה-Multi-Cycle.  
  
ביחידה זו קיים מפענח (Decoder) אשר מפיק אותות בקרה (כחולים) ע"פ שדה ה- Opcode של הפקודה הנוכחית לביצוע. אותם קוי הבקרה יוצאים ממודול זה ונכנסים למודול הcontrol שתיארנו קודם.  
  
**Datapath - סימולציה:**  
בסימולציה הנ"ל הוגדרו מספר תהליכים –

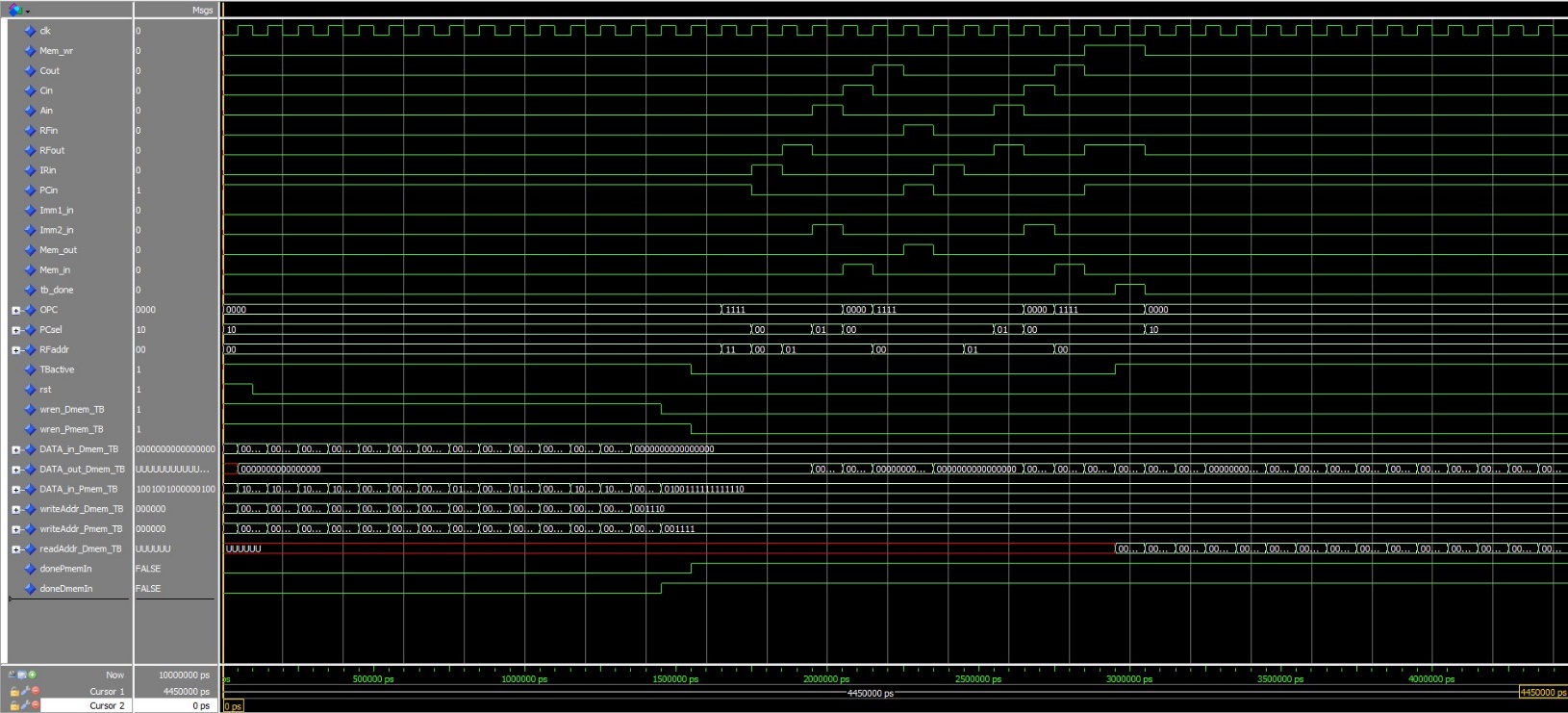
* Gen\_clk – תהליך המייצר שעון מערכת.
* Gen\_rst – תהליך המייצר אות אתחול למערכת.
* Gen\_TB – תהליך המשנה את קו הבקרה Tbactive בהתאם לשלב בתכנית (קריאה מקבצים, עבודה, כתיבה לקובץ)
* LoadDataMem– תהליך קריאת מידע מקובץ חיצוני ((DTCMinit וכתיבתו   
  בזכרון הפנימי DataMem.
* LoadProgramMem– תהליך קריאת מידע מקובץ חיצוני ((ITCMinit וכתיבתו   
  בזכרון הפנימי ProgMem.
* StartTb – תהליך המדמה שתי פקודות בלבד לביצוע: Load 🡪 Store   
  זו אחר זו (התהליך טוען את הקבוע '4' לרגיסטר R2 ולאחר מכן מאחסן אותו בזכרון בתא מספר 3 (יופיע בשורה רביעית בקובץ DTCMcontent)  
  בתהליך זה קוי הבקרה (האדומים) מופעלים ידנית ומדמים את אופן פעולת הControl Unit.
* WriteToDataMem – תהליך קריאת מידע מה-DataMem וכתיבתו בקובץ חיצוני ((DTCMcontent

בסימולציה זו אנו טוענים את זכרונות ה-CPU במידע ובפקודות לביצוע, מבצעים את הפעולות הנדרשות, ולבסוף כותבים את תוכן הזכרון לקובץ חיצוני.

להלן תוצאות הסימולציה, המתוארות על פני תרשים ה-wave :

שלב 2:  
ביצוע הפקודות.  
ניתן לראות את קוי הבקרה הפועלים בכל אחד מחמשת מחזורי השעון

שלב 1:  
כתיבה לזכרונות מקבצים חיצוניים

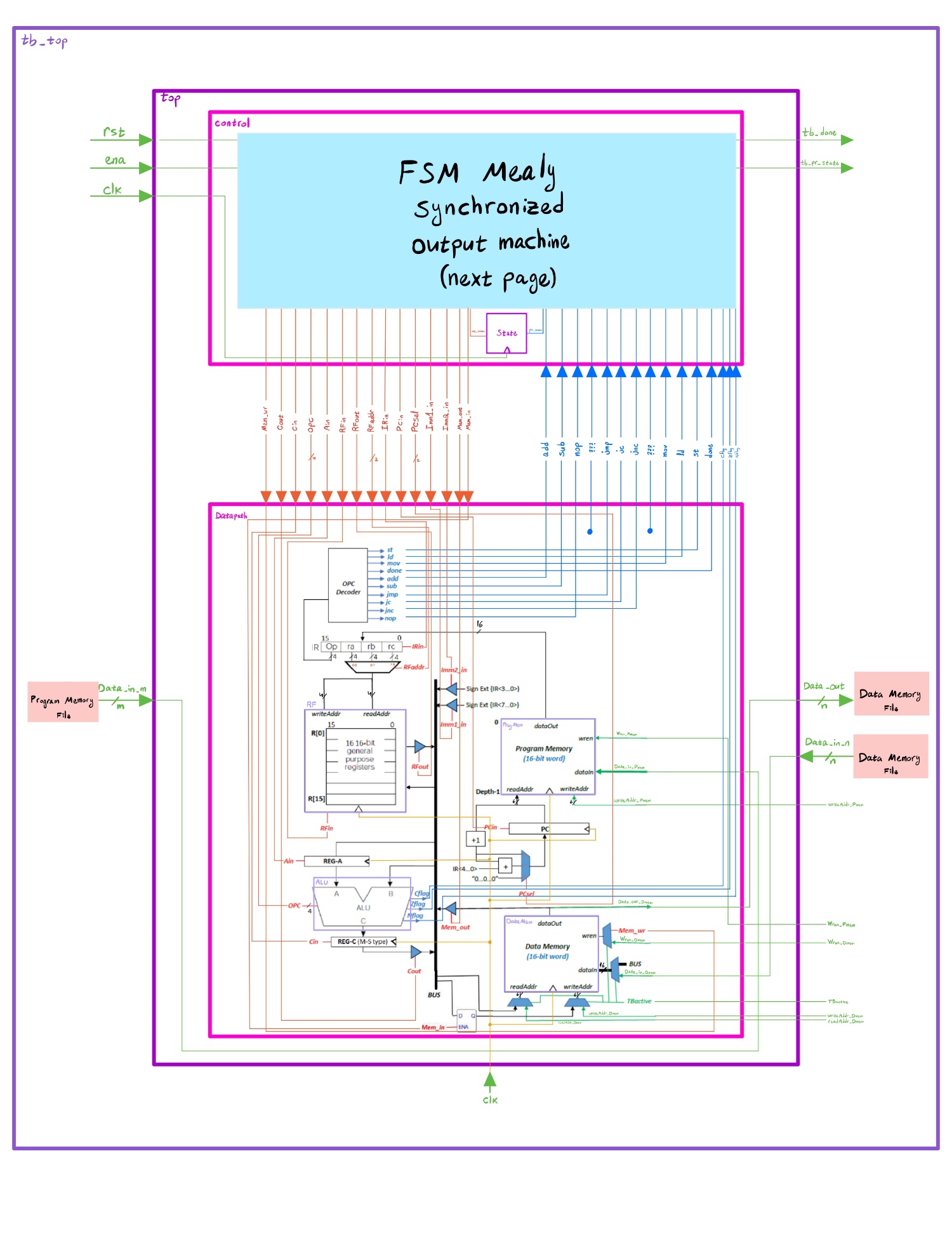


שלב 3:  
כתיבת תוכן הזכרון לקובץ חיצוני

סוף שלב 2:  
הדלקה ידנית של הסיגנל  
tb\_done

סוף שלב 1:  
עם סיום קריאת הקבצים מופעלים שני דגלים

1. **Top -**

מודול זה מכיל בתוכו את שני המודולים שתיארנו לעיל, ואחראי על הקישור ושיתוף הפעולה ביניהם.  
להלן איור המפרט את המודול המלא:

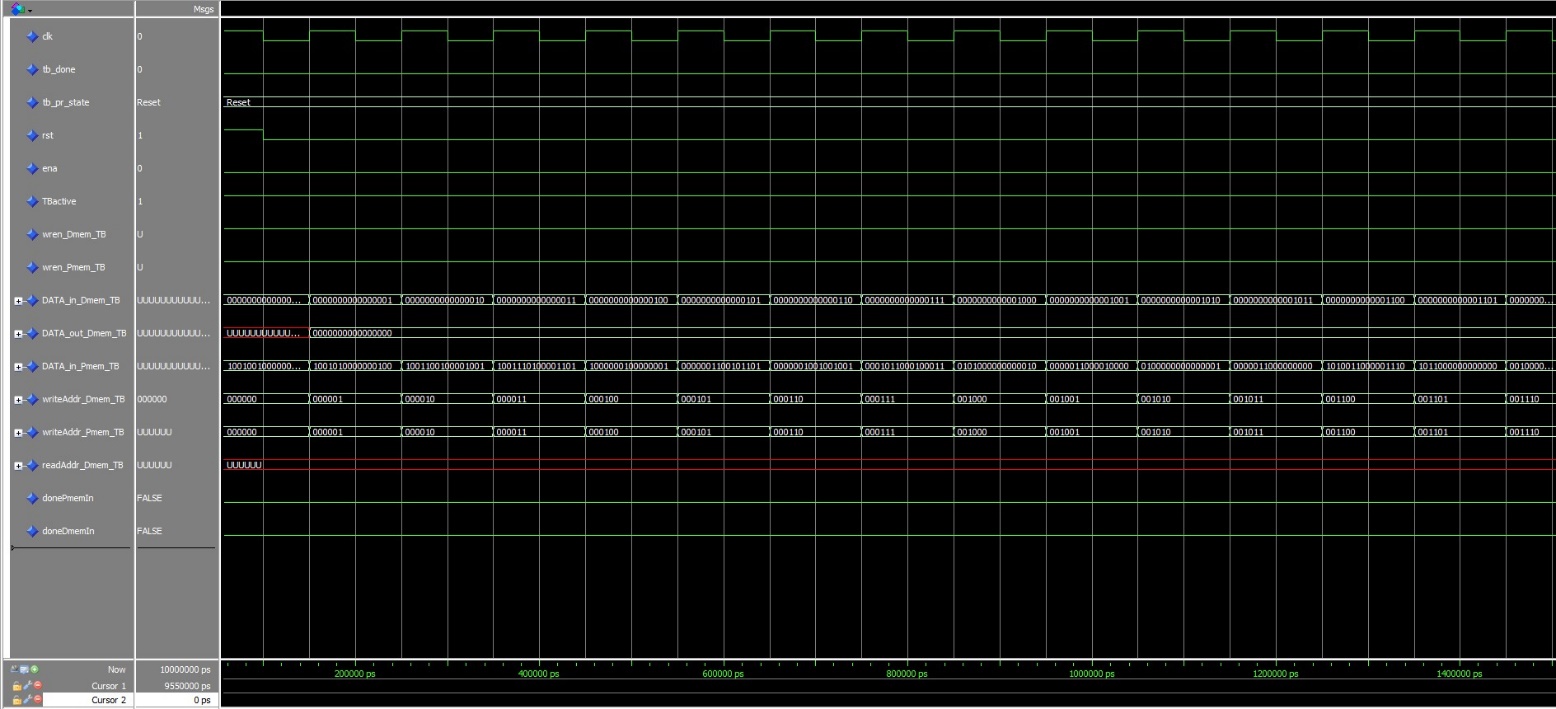
**Top - סימולציה:**

בסימולציה הנ"ל הוגדרו מספר תהליכים –

* Gen\_clk – תהליך המייצר שעון מערכת.
* Gen\_rst – תהליך המייצר אות אתחול למערכת.
* Gen\_TB – תהליך המשנה את קו הבקרה Tbactive בהתאם לשלב בתכנית (קריאה מקבצים, עבודה, כתיבה לקובץ)
* LoadDataMem– תהליך קריאת מידע מקובץ חיצוני ((DTCMinit וכתיבתו   
  בזכרון הפנימי DataMem.
* LoadProgramMem– תהליך קריאת מידע מקובץ חיצוני ((ITCMinit וכתיבתו   
  בזכרון הפנימי ProgMem.
* WriteToDataMem – תהליך קריאת מידע מה-DataMem וכתיבתו בקובץ חיצוני ((DTCMcontent
* בסיום שני תהליכי הקריאה מקבצים חיצוניים, יופעל אות ה-En ובכך תתחיל פעולת ה-CPU עבור רצף הפקודות שהוגדרו. עם פקודת ה-tb\_done ובסיום פעולת ה-CPU יחל שלב הכתיבה לקובץ החיצוני.
* הCPU מבצע את התכנית שפורסמה בקובץ המטלה, אשר משווה בין ערכם של שני רגיסטרים ובהתאם לכך משנה את תוכנו של תא מספר 14 (יופיע בשורה האחרונה בקובץ ה-DTCMcontent).

שלב 1:  
כתיבה לזכרונות מקבצים חיצוניים

להלן תוצאות הסימולציה, המתוארות על פני תרשים ה-wave.



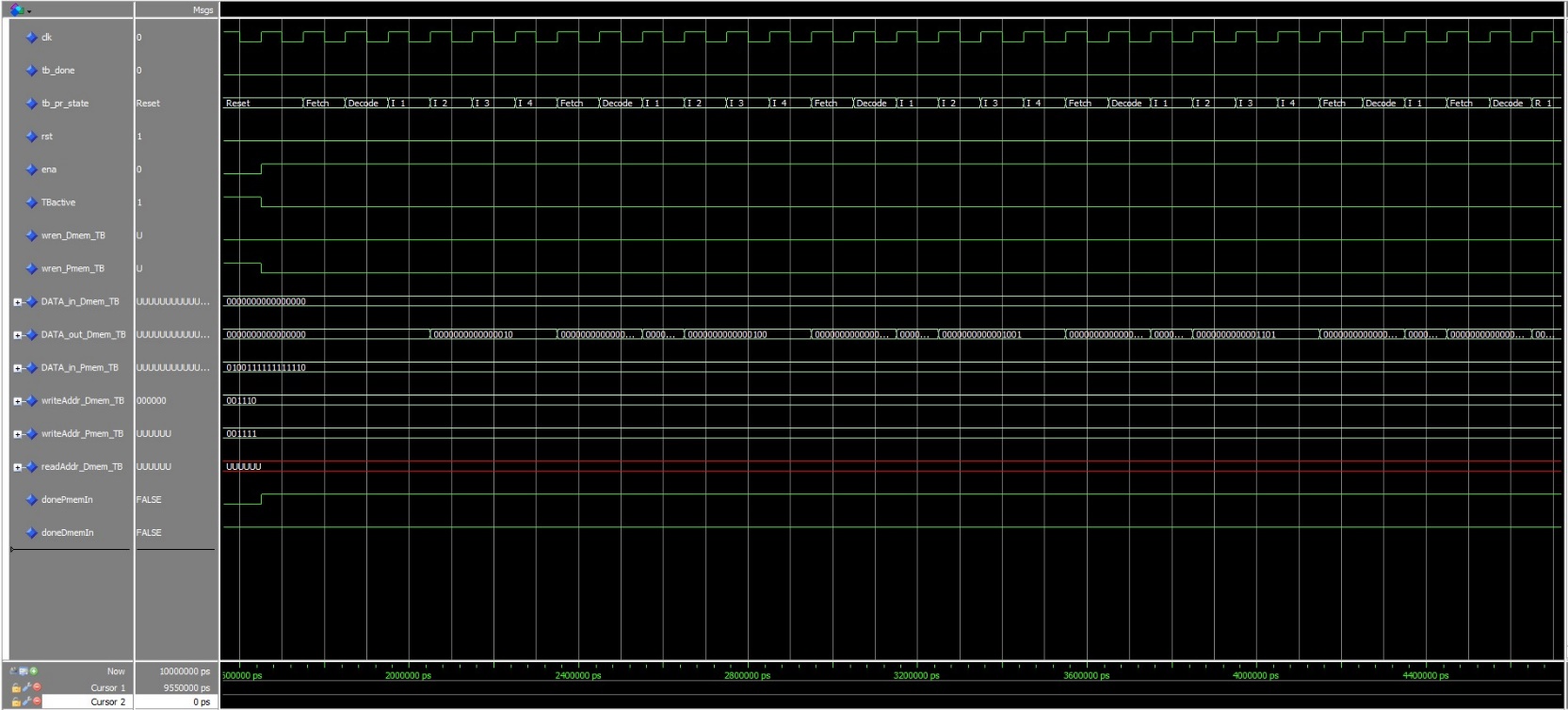
הכתובות בזכרון בהן נרצה לאחסן את המידע

המידע המוזן ל-  
DataMem

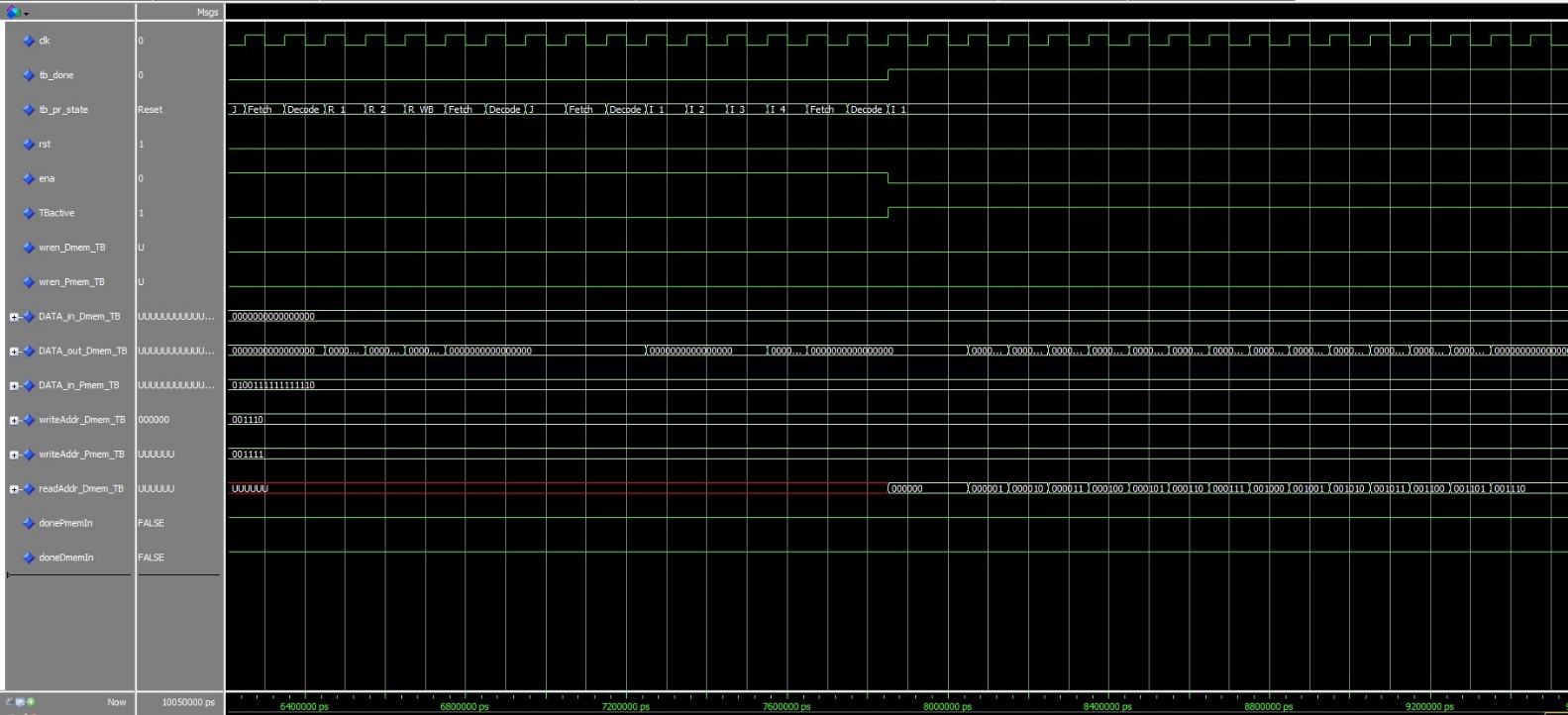
הפקודות לביצוע-  
זוהי פקודת  
ld R9, R0, 9

ביצוע 5 הסייקלים של פקודת   
load  
שתוארה בעמוד הקודם

שלב 2 -   
בעת הפעלת קו ה-  
en  
מתחילות להתבצע הפקודות



שלב 3-  
עם עליית קו   
tb\_done  
מסיים ה-  
CPU  
את עבודתו ומתחילה הכתיבה לקובץ החיצוני



הפעלת קו   
Tbactive  
לצורך כתיבה לקובץ חיצוני

המידע הנקרא ממודול ה-  
DataMem

כתובות לקריאה

1. **רכיבים נוספים:**כפי שציינו קודם, בתוך מודול ה-Datapath מופיעים מספר מודולים נוספים האחראיים על ביצוע הפעולות הנדרשות ב-CPU:

ALU/FA:  
מודול זה מסייע לביצוע פעולות אריטמטיות ולוגיות בין שתי כניסותיו: A,B למוצאו C. המודול ממומש באמצעות מחבר/מחסר אדווה (Carry Ripple Adder/sub) המורכב משרשור רכיבי FA.

DataMem/ProgMem:  
מודולים אלו משמשים כרכיבי זיכרון RAM, ע"י מתן כתובת באורך 6 ביטים  
(מרחב כתובות של ), כאשר בכל כתובת מאוחסן מידע באורך 16 ביטים.  
מודול אחד משמש לאחסון מידע הקשור לתכנית, בעוד השני מאחסן את סט הפקודות לביצוע.

RF:  
בדומה לזכרונות ה-RAM, מודול זה מממש קובץ רגיסטרים בעל 16 רגיסטרים המכילים כל אחד 16 ביטים של מידע.

BidirPin:  
מודול זה מדמה BUS דו כיווני, אשר מתחבר לכלל המודולים השונים המתוארים לעיל.  
על מנת למנוע מצב של Over driven, ועל מנת לשמור על הקוד סינטזבילי, קיים צורך במודל זה הכולל בתוכו רכיבי Latch, קוי אפשור, קוי כניסה ויציאה.