הטכניון - מכון טכנולוגי לישראל

מבנה מחשבים ספרתיים (234267)

מבחן מסכם מועד א'

10 פברואר 2013

<u>מרצים</u>: **ליהוא רפופורט, עדי יועז.**

<u>מתרגלים</u>: **גיל איינציגר, פרנק סלה.**

שם :	_____
מס. ת.ז. :	_____

- משך הבחינה: שלוש שעות.
- מותר כל חומר עזר.
- יש לכתוב את התשובות בטופס הבחינה בלבד ובמקום המיועד לתשובה.
- כתוב בקיצור ככל האפשר. <u>יש לנמק כל תשובה</u>.
- בדוק שבטופס שבידך יש 9 עמודים כולל עמוד זה.
- המבחן כולל ארבע שאלות, יש לענות על כולן.

שאלה 1	20 /
שאלה 2	30 /
שאלה 3	40 /
שאלה 4	10 /
ציון סופי	/100

בהצלחה !

שאלה 1 – זיכרון וירטואלי (20 נק')

נתון מעבד דמוי x86 העובד במוד של 64 ביט ומבנה הכתובת הבא:

63 48	47 40	39 28	27 16	15 8	7 0
Sign ext	PML4	PDP	DIR	PTE	offset

המעבד תומך הן בדפים בגודל קטן (המוצבעים ע"י כניסות ב-PTE) והן בדפים בגודל גדול (המוצבעים ע"י כניסות ב-DIR). גודל כל כניסה בטבלאות הדפים בכל אחת מהרמות היא 8 bytes.

במעבד קיימים TLBs עבור כל אחת מהרמות, שבכל אחד מהם 4 כניסות, direct mapped.

א. (2 נק') מהו גודלו של דף גדול ? _____

הסבר: _____

_____

ב. (2 נק') לכל אחת מהרמות רשמו את מספר הכניסות בטבלת דפים ברמה זו:

PTE ? _____ DIR ? _____ PDP ? _____ PML4 ? _____

הסבר: _____

_____

ג. (16 נק') נתונה סידרת פניות לכתובות וירטואליות (בבסיס 16). רשמו לכל כתובת, האם יש miss או hit בכל אחת מרמות ה-TLBs. הניחו כי נעשה שימוש בדפים קטנים בלבד, ושבתחילת הסידרה ה-TLBs ריקים.

1) FFFF 2345 6789 ABCD

TLB	PML4	PDP	DIR	PTE
Hit / Miss				

הסבר _____

_____

2) FFFF **12**45 6789 ABCD

TLB	PML4	PDP	DIR	PTE
Hit / Miss				

הסבר _____

_____

3) FFFF **23**45 6789 ABCD

TLB	PML4	PDP	DIR	PTE
Hit / Miss				

הסבר _____

_____

_____

_____

4) FFFF 2345 678**E** ABCD

TLB	PML4	PDP	DIR	PTE
Hit / Miss				

הסבר _____

_____

_____

שאלה 2 – זיכרון Cache (30 נק')

נתון מעבד עם היראכית זיכרון בעלת שתי רמות מטמון:

L1 data cache: 4-Way, 64 Byte בשורה, גודל 64K Byte, מדיניות LRU.

L2 cache: 2-Way, 64 Byte בשורה, גודל 2M Byte, מדיניות LRU.

נתונה התוכנית הבאה, הפועלת על אברי מערך חד-מימדי בזיכרון המתחיל בכתובת 44444H.
כל איבר במערך הוא בגודל **4 בתים**. לשם פשטות הניחו כי הכתובות בתוכנית הן פיזיות.

```
For j=0 to 16420₁₀ {S = S + (A[j]^2);}
For j=0 to 16420₁₀ {S = S + (A[j]^3);}
```

הניחו כי בתחילת הביצוע זיכרונות המטמון ריקים.

א. (5 נק') לאיזה set ב-L1 יוכנס A[20] ? הסבר _____

_____

_____

_____

_____

_____

ב. (6 נק') מה הם אברי המערך A שנמצאים ב-way 0 שב-set שחושב בסעיף א' בסוף ביצוע הלולאה הראשונה ? הסבר.

_____

_____

_____

_____

_____

ג. (3 נק') מהו ה-hit rate ב-L1 עבור הלולאה הראשונה (אחוז הפניות לאברי המערך שפוגע) ? הסבר

_____

_____

_____

ד. (6 נק') מהו ה-hit rate ב-L1 עבור הלולאה השנייה (אחוז הפניות לאברי המערך שפוגע) ? הסבר

ה. (6 נק') ללא קשר לסעיפים הקודמים ולקוד שתחילת השאלה, הציעו סידרת פניות לזיכרון כך שעבור הכתובת האחרונה בסדרה יש hit ב-L1, אך הכתובת לא תמצא ב-L2. הניחו כי L1 אינו מוכל ב-L2, וכן כי בתחילת הסדרה ה-cache-ים ריקים. יש להסביר.

ו. (4 נק'). בהנחה שגודל דף וירטואלי הוא 1KB ושכתובת פיזית היא בגודל 48 סיביות, כמה סיביות מתוך ה-set צריכות להיות וירטואליות ומהו מספר סיביות ה-tag, על מנת לאפשר פנייה ל-set במקביל לתרגום הכתובות מוירטואוליות לפיסיות ?

שאלה 3 – Out-Of-Order Execution (40 נק')

בשאלה זו נתייחס למעבד עם OOOE ו- Speculative Execution . נתון קטע הקוד הבא:

```
1000    load   R1,R2, 0      ; R1=m[R2+0]
1004    load   R3,R2, 100    ; R3=m[R2+100]
1008    add    R2,R2,10      ; R2=R2+10
100C    store  R1,100,R3     ; m[R1+100]=R3
1010    blt    R2,30,1000    ; if (R2<30) PC=1000
1014    sub    R2,R2,20      ; R2=R2-20
```

הנחות:

- פקודת הקפיצה נחזית כנלקחת.
- בתחילת הביצוע בכל כתובת N בזיכרון קיים הערך 2×N, וכן R1=R2=R3=10.
- למען פשטות נניח כי הכתובות בתוכנית הן פיזיות (בבסיס 16) ואין צורך בתרגום.
- L1 data cache מחזיר data תוך מחזור שעון אחד, אך הוא ריק בתחילת הביצוע.
- גודל שורה ב-L1 cache היא 64B ($64 = 40_{16}$).
- L2 data cache מחזיר data תוך 9 מחזורי שעון, והוא מכיל את כל הכתובות המבוקשות כבר בתחילת הביצוע.

<u>אלוקציה של פקודות</u>:

- בכל מחזור מתבצעת אלוקציה **לפקודה אחת** (למעט במחזורים שלאחר flush).
- ה-ROB, MOB, וה- RS הם גדולים ואינם מתמלאים.

<u>ביצוע של פקודות</u>:

- ישנן אינסוף יחידות ביצוע.
- פקודה יכולה להיכנס לביצוע במחזור שלאחר האלוקציה, בתנאי שכל הנתונים להם היא זקוקה כבר מוכנים. פקודה שממתינה לנתון יכולה להיכנס לביצוע מייד במחזור שלאחריו הוא מוכן.
- ביצוע פקודת ALU אורך מחזור שעון אחד.
- ביצוע פקודת branch אורך מחזור אחד.
  - אם החיזוי מתגלה כשגוי, **במחזור הנוכחי מיד מבוצע flush**.
  - הפקודות מהמסלול הנכון מבצעות אלוקציה 5 מחזורים לאחר flush (בזמן t+5).
- פקודת load נשלחת לביצוע בזמן t כאשר הנתונים לחישוב הכתובת מוכנים.
  - במחזור הראשון (t) מחושבת הכתובת.
  - במחזור השני (t+1) נבדק התנאי הבא: עבור כל פקודת store הקודמת ל-load, הכתובת של ה-store ידועה ומתקיים: או שהכתובת של ה-load שונה מהכתובת של ה-store, או ששתי הכתובות שוות, וה-data של ה-store כבר ידועה.
  - במחזור השלישי (t+2), במידה והבדיקה מצליחה, הנתון מתקבל מה-L1 cache (אם יש hit), או ישירות מה-MOB ע"י store to load forwarding.
  - במידה והבדיקה מצליחה אך יש L1 cache miss וכן אין store to load forwarding, הנתון מתקבל בזמן t+11 מה- L2 cache.
  - במידה והבדיקה נכשלת, ה-load הוא חסום (blocked). כאשר מוסר תנאי החסימה, ה-load נשלח שוב לביצוע, ומדלגים על המחזור הראשון (מתחילים בבדיקת התנאי).
  - במידה וה-load נחסם יותר מפעם אחת, יש לרשום את כל קודי החסימה.

- פקודת store נשלחת לביצוע כאשר הנתונים לחישוב הכתובת מוכנים.
  - חישוב הכתובת אורך מחזור שעון אחד, ובסופו נכתבת הכתובת ל-MOB.
  - באופן בלתי תלוי, כאשר הנתון לכתיבה לזיכרון מוכן, במחזור הבא הוא נכתב ל-MOB.

<u>commit של פקודות</u>:

- פקודה יכולה לבצע commit החל מהמחזור שלאחר סיום הביצוע, ובתנאי שהפקודה שלפניה ביצעה/מבצעת commit. אין מגבלה על כמות הפקודות שמבצעות commit בכל מחזור.
- פקודת store מבצעת את הכתיבה אל ה-cache בזמן post-commit.

<u>יש למלא את הטבלה שבעמוד הבא. לכל פקודה יש לרשום</u>:

- R3, R2, R1 – ערכי הרגיסטרים הארכיטקטוניים לאחר commit של הפקודה.
  יש להקיף בעיגול את הערך של הרגיסטר הארכיטקטוני שאליו הפקודה כותבת.
  במידה והפקודה אינה מגיעה ל-commit יש להשאיר שדות אלה ריקים.
- addr – כתובת הגישה לזיכרון – עבור פקודות load ו-store בלבד.
- data – ערך זיכרון שנקרא או נכתב – עבור פקודות load ו-store בלבד.
- T alloc: הזמן בו מבוצעת אלוקציה לפקודה (פקודה אחת בכל מחזור, החל מ- t =1)
- src1, src2: מספרי הרגיסטרים המשמשים כ-sources לפקודה:
  Pi עבור רגיסטר פיזי, ו-Ri במידה וקוראים ישירות את הרגיסטר הארכיטקטוני.

  עבור store: src1 – הרגיסטר המשמש לחישוב הכתובת. src2 – הרגיסטר המכיל את הנתון.
- Imm – במידה ולפקודה יש Imm, ערך ה- Imm.
- T src1 ready , T src2 ready: הזמן בו מוכן כל אחד ערכי ה-sources לפקודה.
  אם ה-src מוכן בזמן האלוקציה, אז זמן זה יהיה שווה לזמן האלוקציה.
  אם הפקודה שמחשבת את הערך של src מסיימת ביצוע בזמן t, ה-src מוכן בזמן t.
- T exe: הזמן בו הפקודה נשלחת לביצוע.
  אם כל ה-src-ים של פקודה (עבור store: ה-src-ים לחישוב הכתובת) מוכנים בזמן t, ניתן לשלוח את הפקודה לביצוע בזמן t +1.
- Load block code (רלוונטי רק בפקודות load): קוד החסימה של ה-load.
  0 – אין חסימה.
  1 – חסימה כתוצאה מ- unresolved store address
  2 – חסימה כתוצאה מ- waiting for store data
- T data ready:
  - עבור store: הזמן בו ה-data לכתיבה לזיכרון מוכן.
  - עבור load: הזמן בו מתקבל ה-data (מה-cache או ישירות מה-MOB).
- T commit: הזמן בו הפקודה מבצעת commit
- שימו לב כי ניתן למלא את 5 העמודות השמאליות (המתארות את ריצת התוכנית) תחילה. לאחר מכן מומלץ למלא את 3 העמודות הבאות (src1,src2,imm) ולבסוף את שאר העמודות.
- **יש למלא גם את פקודות 10 ו-11 בטבלה**

Pdst	instruction	R1	R2	R3	addr	data	T alloc	src1	src2	Imm	T src1 ready	T src2 ready	T exe	Load block code	T data ready	T commit
0	load R1=m[R2+0]															
1	load R3=m[R2+100]															
2	R2=R2+10															
3	store m[R1+100]=R3															
4	if (R2<30) PC=1000															
5	load R1=m[R2+0]															
6	load R3=m[R2+100]															
7	R2=R2+10															
8	store m[R1+100]=R3															
9	if (R2<30) PC=1000															
10																
11																

שאלה 4 (10 נק')

א. (6 נק') מהו אורך ההיסטוריה המינימלי הנדרש לחיזוי מושלם של הסידרה הבאה במצב היציב ? הסבר

1010111 1010111 1010111 ...

ב. (2 נק') במערכת שבה מעבד צפוי לעבוד רוב הזמן מעל Vmin, האם מנגנון המשפר את הביצועים ב-1% ומעלה את ההספק (power) ב-2% הוא power efficient ? הסבר

האם מנגנון זה הוא energy efficient ? הסבר

ג. (2 נק') יש לרשום על כל אחד מהמשפטים הבאים אם הוא נכון או לא נכון, ולתת הסבר קצר

1. במערכת מבוססת Ring זמן הגישה של כל מעבד לכל cache slice הוא שווה _____

2. במערכת מבוססת Ring כל cache slice מכסה תחום כתובות שונה _____