



TEL AVIV UNIVERSITY

# מערכות הפעלה

ערן טרומר

סמסטר א' תשע"ב

## הרצאה 4

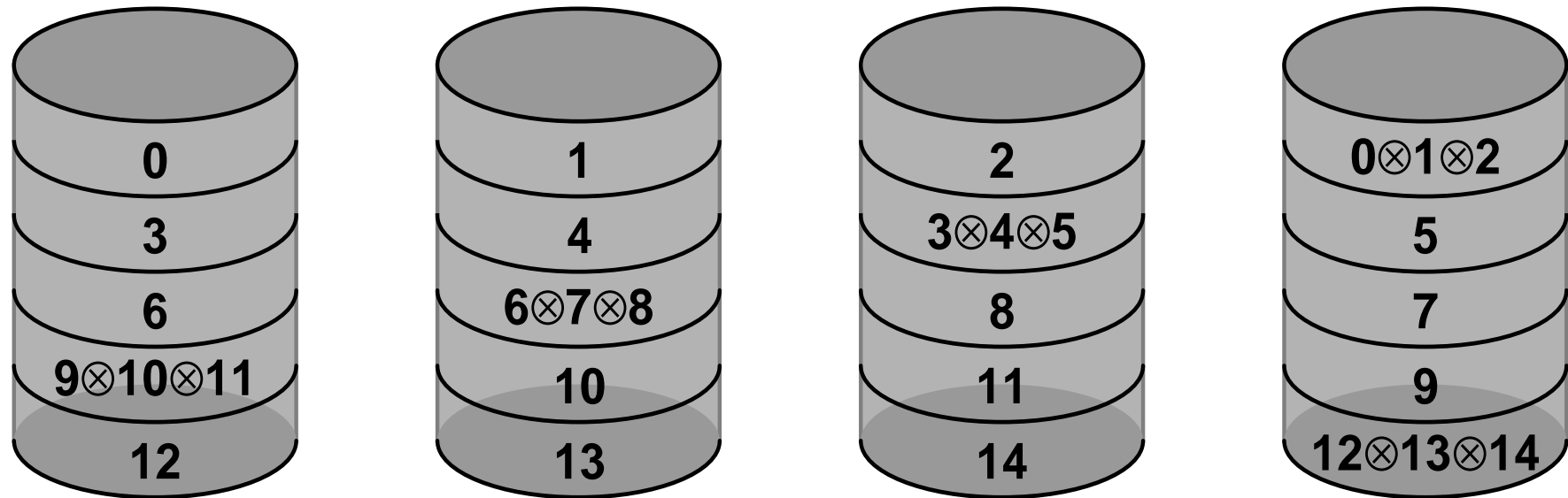
קלט/פלט (המשך)

ניהול זיכרון

# מערכי דיסקים (RAID)

- ❖ Redundant Array of Inexpensive Disks
- ❖ מספר דיסקים משמשים כהתקן אחסון אחד
- ❖ מבוסס על מיפוי בלוקים של נתונים בהתקן הוירטואלי (מערך הדיסקים) לדיסק+היסט, כלומר למיקום של בלוק פיזי של נתונים
- ❖ מספר ואריאציות שנקראות רמות שמספקות קומבינציות שונות של שירותים; רמה גבוהה אינה בהכרח טובה יותר לכל שימוש

# שחזור נתונים עם RAID



- ❖ המערך מחולק לרצועות
- ❖ רצועה שומרת  $n-1$  בלוקים של נתונים ובלוק זוגיות
- ❖ אפשר לקרוא בלוק בודד; או לקרוא רצועה ולבדוק שגיאות
- ❖ אפשר לשחזר כל בלוק חסר, ואפשר גם לשחזר דיסק שלם שהתקלקל
- ❖ כתיבה של בלוק על ידי קריאה של הבלוק ובלוק הזוגיות ועדכון, או על ידי כתיבה של רצועה שלמה בלי לקרוא קודם כלום
- ❖ שיפור ביצועים ע"י גישה מקבילה

# מימוש RAID

- ❖ במנהל ההתקן במערכת ההפעלה
- ❖ בספרייה כללית של מערכת ההפעלה
  - כמו דיסקים לוגיים ומחיצות
  - דוגמה: Linux dm-raid
  - תאימות וגמישות
- ❖ בבקר הדיסקים (חומרה/קושחה) כשדרושים ביצועים מירביים

# הפשטות נוספות

- ❖ בקר RAID או מנהל התקן RAID גורם לכמה של דיסקים להיראות כיחידת אחסון אחת (גדולה ואמינה יותר)
- ❖ מחיצות: חלוקה של דיסק פיזי לדיסקים לוגיים; טבלה בתחילת הדיסק מתארת את החלוקה, כל מערכות ההפעלה מבינות אותה
- ❖ logical volumes: מנגנון יותר מתוחכם ממחיצות (partitions) שמחלק את הדיסק לרצפים (extents) ומצרף כמה רצפים ליחידת אחסון לוגית אחת; למשל, אפשר לאחד את החלקים המהירים של 4 דיסקים שונים ליחידת אחסון אחת

## דיסקים אופטיים

- ❖ תקליטורים מאחסנים 650-700 MB, תקליטי DVD עד 4.7 GB, Blu-ray רגיל עד 50 GB
- ❖ מידע מיוצג על ידי שקעים או צריבות במשטח
- ❖ תהליך דפוס מהיר להפצה של מידע ספרתי, אפשרות צריבה (איטית יחסית) בכוננים מתאימים
- ❖ אורך חיים גבוה מתאים לשמירת מידע ארכיונית
- ❖ שימשו לגיבויים אבל היום דיסקים מגנטיים משמשים לגיבויים (בעבר גם סרטים מגנטיים שימשו לגיבויים)

## סרטים מגנטיים

- ❖ עד עשרות GB בכל קלטת
- ❖ מחיר נמוך ל-GB
- ❖ כוננים יקרים (טכנולוגיה שונה מזו של קלטות וידאו למניעת שחיקה של הראש והסרט)
- ❖ שימוש נרחב לגיבוי ושמירה ארכיונית של כמויות נתונים גדולות, תוך שימוש בספריות קלטות רובוטיות

# תצוגות גרפיות

- ❖ הבקר מכיל יחידת זיכרון (frame buffer) שכל מילה בה ממופה לפיקסל (לפעמים נעשה שימוש בזיכרון הראשי של המחשב, דרך הפס)
- ❖ בבקרים פשוטים המעבד צובע פיקסלים על ידי כתיבה לזיכרון הזה
- ❖ המבנה נגזר מהצורך להעביר לתצוגה נתונים בקצב גבוה, בעיקר עבור אנימציה ווידאו.  
 דוגמה:  $30 \text{ frame/s} \times 1 \text{ Mpixel/frame} \times 3 \text{ B/pixel} = 90 \text{ MB/s}$
- ❖ בבקרים מתוחכמים המעבד יכול להורות לבקר לבצע פעולות מורכבות:
  - 2D: קווים, מלבנים, שכפול
  - 3D: הטלת מודלים תלת-מימדיים, תאורה, הסתרות של עצמים במרחב.
  - Graphical Processing Unit (GPU) - מעבד עצמאי חזק מאד, ייעודי לחישובים גרפיים
- ❖ אפשרות לתצוגה וירטואלית מרוחקת (X Window, VNC)



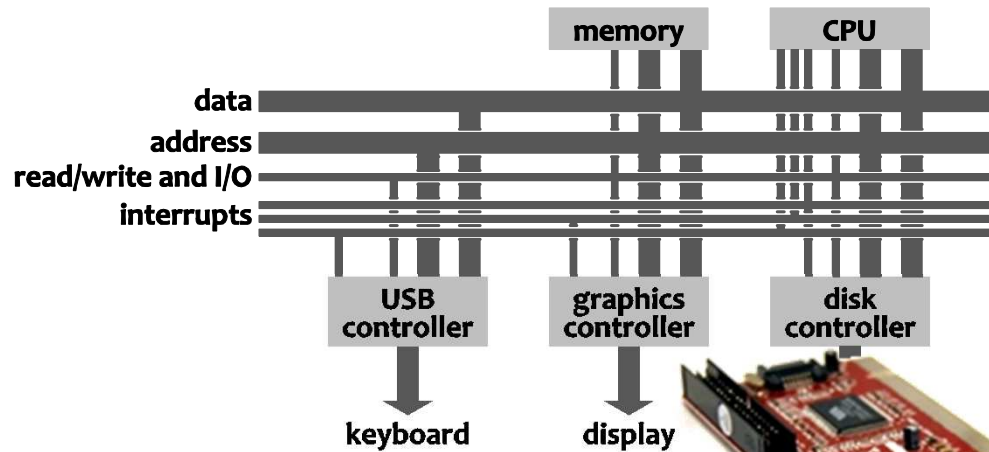
# כרטיסי קול

- ❖ קצב הנתונים נמוך יותר
- ❖ דורשים השהייה נמוכה (low latency)
- ❖ שירותים נדרשים: mixer, ניתוב, ווליום
- ❖ אפשרות לכרטיסי קול וירטואליים מרוחקים

# ממשקים נוספים ברמת החומרה

❖ מטרה: להקל על כתיבה והפצה של מנהל ההתקן, ובניית

החומרה.



ממשק בקר

• "IDE"

• AHCI

• ייחודי לדגם

• ...

ממשק התקן

• PATA

• SATA

• SCSI

• ...



# ניהול זיכרון

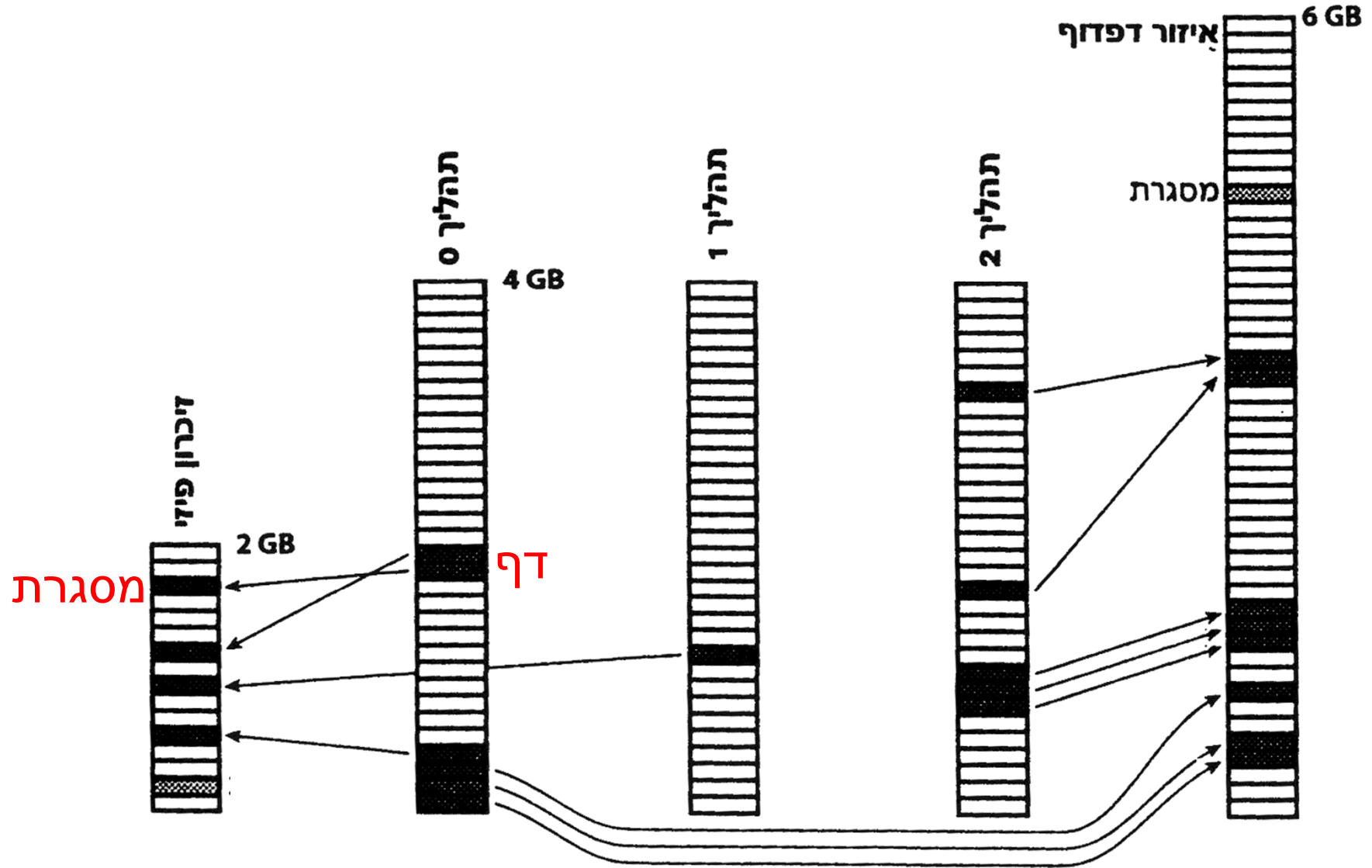
# מטרות ניהול הזיכרון

- ❖ יכולת להריץ מספר תוכניות בו-זמנית תוך כדי הגנה על הזיכרון של כל תוכנית מפני האחרות
- ❖ יכולת להריץ תוכניות שסך גודל הזיכרון המוקצה שלהן גדול מהזיכרון המותקן במחשב
- ❖ יכולת להשתמש בדיסקים כהרחבה זולה אך איטית של הזיכרון (המעבד אינו יכול להתייחס ישירות למידע בדיסק)
- ❖ יכולת להזיז את מבני הנתונים של תוכנית במהלך ריצתה בלי שהתוכנית מודעת להזזות הללו; כך ננצל "חורים" קטנים בזיכרון, או לאחד חורים לזיכרון פנוי רצוף, או להעביר מבנה נתונים בין מהזיכרון לדיסק וחזרה למקום אחר בזיכרון

# הפרדה בין כתובות בתוכנית וכתובות על הפס

- ❖ תוכנית מציגה למעבד כתובות של תאי זיכרון שיש לקרוא או לכתוב מהם נתונים
- ❖ הכתובות מאוחסנות ברגיסטרים או בזיכרון
- ❖ מצביע התוכנית מציג למעבד כתובות שיש לקרוא מהן פקודות
- ❖ המעבד אינו מציג את הכתובות הללו על ערוץ הכתובות בפס, אלא מתרגם אותם לכתובות אחרות תוך שימוש במבנה נתונים שמערכת ההפעלה מקימה ומתחזקת
- ❖ כתובות וירטואליות בתוכנית, כתובות פיזיות על הפס

# זיכרון וירטואלי



## תרגום כתובות וירטואליות לפיזיות

- ❖ מערכת ההפעלה מקימה **טבלת דפים**, מבנה נתונים שממפה את הדפים של תהליך למסגרות פיזיות
- ❖ טבלת הדפים שמורה בזיכרון ומערכת ההפעלה מודיעה למעבד היכן היא בעזרת אוגר מיוחד
- ❖ המעבד מסוגל לחפש מיפוי של דף למסגרת בטבלה
- ❖ על מנת לחסוך את הצורך בחיפוש בכל גישה לזיכרון, המעבד שומר בהתקן חומרה מיוחד (חלק מהמעבד) בשם TLB מיפויים שנעשה בהם שימוש לאחרונה
  - הנחה: מקומיות גישה (מה שהיה לאחרונה הוא שיהיה בקרוב)
- ❖ במעבדים מסוימים מערכת ההפעלה אחראית לביצוע חיפוש בטבלה והכנסת מיפויים ל-TLB; לא נפוץ

# תרגום כתובות (המשך)

❖ בהינתן כתובת פיזית, המעבד מפרק אותה ל**מספר דף** (סיביות בכירות) ו**להיסט** בתוך הדף (סיביות זוטרות)

❖ לדוגמה, אם גודל דף 4096 בתים, אזי 12 סיביות מציינות את ההיסט והשאר את הדף

❖ המעבד ממפה את הדף למסגרת פיזית ומשרשר את ההיסט למספר המסגרת על מנת ליצור כתובת פיזית שניתן להציג על הפס, למשל

0000 0000 0000 0000 0011 0000 0000 1001

❖ הכתובת הוירטואלית מתייחסת לבית ה-9 בדף מספר 3; אם הדף הזה ממופה למסגרת מספר 255, הכתובת שתוצג על הפס היא

0000 0000 0000 1111 1111 0000 0000 1001



# טבלאות דפים שטוחות

עבור כך תהליך:

❖ מערך בגודל מספר הדפים במרחב זיכרון וירטואלי

❖ בכל איבר במערך מצוין:

▪ מספר המסגרת

▪ מספר סיביות לתיאור מצבים שונים, כמו דף שאין מוקצה כלל לתהליך  
([in]valid) או שהמסגרת שמכילה אותו אינה בזיכרון ([not]present)

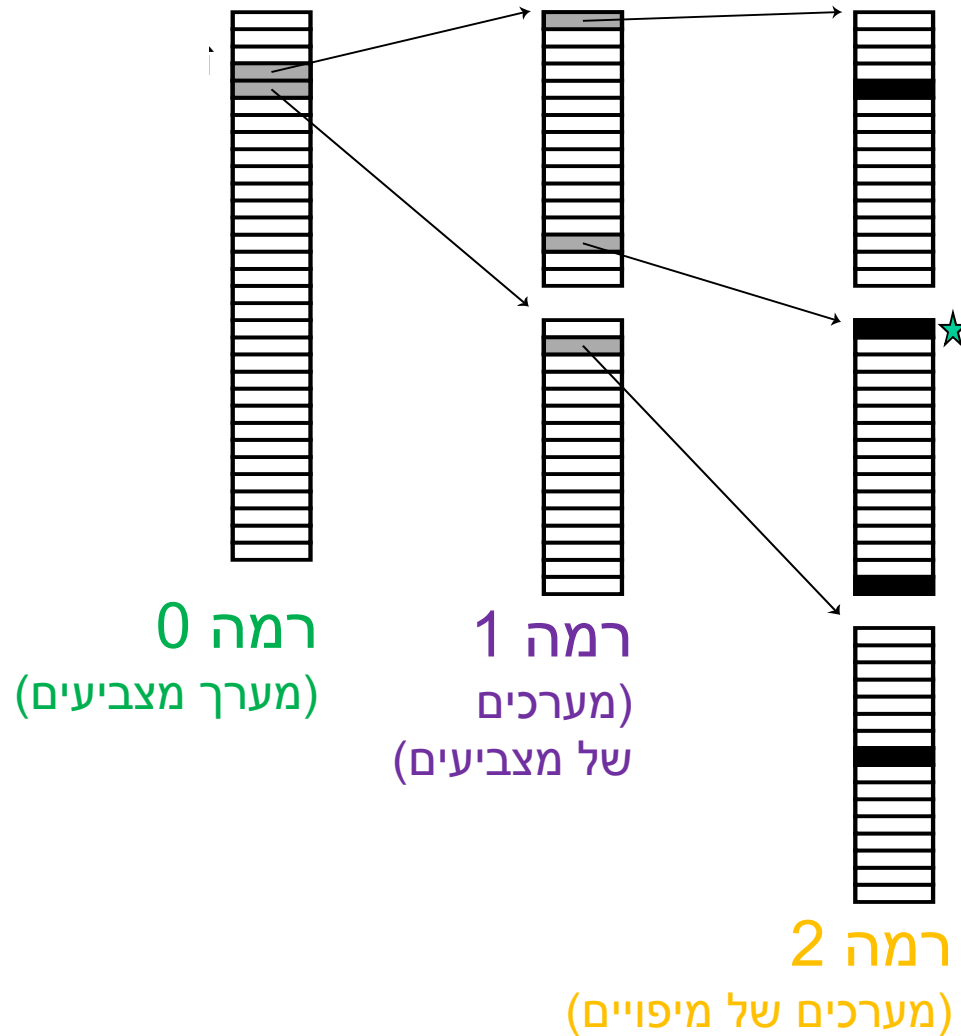
❖ חיפוש פשוט, טבלה גדולה

❖ למשל, מרחב וירטואלי של 2 GB (מצביעים של 32 סיביות) עם

דפים בגודל 4 KB דורש טבלה עם 512 איברים. אם גודל כל

איבר 4 בתים הטבלה של כל תהליך צורכת 2 MB

# טבלאות דפים היררכיות דלילות



- ❖ מבנה נתונים עבור כל תהליך
- ❖ המיפויים שמורים בעץ חיפוש
- ❖ תתי עצים שאין בהם מיפויים תקפים אינם מיוצגים כלל (המצביע אליהם הוא null)
- ❖ מספר הדף מפורק לקבוצות סיביות
- ❖ הסיביות הבכורות מצביעות על הבן של השורש וכך הלאה
- ❖ המיפויים בעלים מיוצגים על ידי מערכים של עלים

היסט  
00010000110100000000000000001001

כתובת  
וירטואלית:

## טבלאות דפים היררכיות

- ❖ חיפוש מורכב יותר מאשר בטבלה שטוחה (ממומש בחומר!)
- ❖ יותר גישות לזיכרון בזמן חיפוש (אחת לכל רמה בעץ)
- ❖ חיסכון עצום בזיכרון עבור מרחבי זיכרון שרק חלק קטן מהם מוקצה (או בשימוש)

## טבלאות דפים הפוכות

- ❖ מטרה: ייצוג חסכוני כאשר סך הזיכרון הוירטואלי של התהליכים גדול בהרבה מהזיכרון הפיזי
- ❖ טבלה אחת לכל התהליכים
- ❖ שומרת רק מיפויים של דפים בזיכרון; מיפויים לאזור הדפדוף שמורים במבנה נתונים אחר
- ❖ ממומשת כטבלת גיבוב (hash) שמפתח החיפוש שלה הוא מספר הדף (ואולי מספר התהליך)
- ❖ כניסה צריכה לציין גם מספר מסגרת וגם מספר תהליך
- ❖ אלגוריתם חיפוש מורכב יחסית
- ❖ גודל הטבלה תלוי רק בגודל הזיכרון הפיזי ואינו תלוי בגודל מרחבי הזיכרון הוירטואליים או במספר התהליכים

# תחזוקת טבלאות הדפים וה-TLB

- ❖ מערכת ההפעלה מחליטה היכן לשכן דפים
- ❖ אי לכך, מערכת ההפעלה היא שמתחזקת את טבלאות הדפים
- ❖ המעבד מחפש בטבלה ומכניס מיפויים ל-TLB
- ❖ שינוי כניסה בטבלת דפים דורש מחיקת המיפוי מה-TLB אם הוא נמצא שם, בעזרת פקודת מכונה מיוחדת
- ❖ מערכת ההפעלה צריכה להודיע למעבד מה הכתובת של טבלת הדפים של התהליך שרץ (באוגר מיוחד)

# הגנה על זיכרון

- ❖ כל הפקודות שעוסקות ב-TLB ובכתובת של טבלת הדפים מותרות רק במצב מיוחס
- ❖ אם טבלת הדפים הפוכה, מערכת ההפעלה מודיעה למעבד מה המזהה של התהליך שרץ כרגע על מנת להשתמש רק במיפויים שלו
- ❖ אי לכך, המעבד משתמש רק במיפויים שמערכת ההפעלה יצרה עבור התהליך שרץ כרגע; תהליך לא יכול להתייחס לכתובות פיזיות כלל, ולא יכול להתייחס למסגרות שאינן ממופות לדפים שלו