



## מושגי יסוד במערכות הפעלה ניר אדר

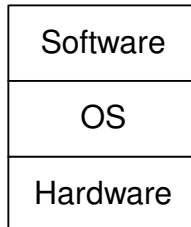
מסמך זה הורד מהאתר [www.underwar.co.il](http://www.underwar.co.il).

מסמך זה מופץ תחת רשיון [CC-by-sa](http://creativecommons.org/licenses/by-sa/3.0/) גרסה 3.0 (<http://creativecommons.org/licenses/by-sa/3.0/>). הוא חובר ע"י ניר אדר בהתבסס, בין השאר, על חומרי הרצאה מאת פרופ' חגית עטיה, ד"ר טל כהן וד"ר ארז חזד. תודתי נתונה למרצים ולמתרגלים שאפשרו את השימוש ביצירתם לשם יצירת מסמך זה.

## הקדמה

### מהי מערכת הפעלה?

- א. ממשק בין החומרה לתוכניות המשתמש.
- ב. הפשטה של החוברה עבור כל תהליכי המשתמש – החבאת המבנה המסובך של החומרה, והצגה מופשטת של המחשב אל המשתמש.



### מטרות מערכת ההפעלה:

- א. לאפשר למשתמש להריץ תוכנות.
- ב. לחסוך למתכנת את הצורך להתייחס לנושאים בחומרה, כגון גישה להתקנים שונים. סיפוק ממשק אחיד למשתמש בעזרת system calls.
- ג. תמיכה ב-Multi-tasking.
- ד. חלוקת משאבים בין תהליכים שונים ובין מספר משתמשים שונים.
- ה. לתת לכל משתמש אשלייה כאילו כל משאבי המערכת עומדים לרשותו.

### קריאות מערכת (System Calls)

- תהליכי המשתמש מקבלים שירותים ממערכת ההפעלה בעזרת **קריאות מערכת**.
- קריאות מערכת הן פונקציות שניתן לקרוא להן מכל תוכנות המשתמש.
- המשתמש יכול לגשת או לשנות את המבנים הפנימיים של מערכת ההפעלה רק בעזרת קריאות המערכת.
- קריאות מערכת בודקות את תקינות הפרמטרים המועברים אליהן (בניגוד לפונקציות פנימיות של מערכת ההפעלה).

## תהליכים

### מהו תהליך

תהליך זוהי תוכנית שכרגע רצה.  
רכיבי התהליך:

- התוכנה שרצה.
- הנתונים עליהם התוכנה רצה.
- המשאבים שהתליך צורך – זיכרון, קבצים וכו'.
- מצב התהליך (פעיל, מושהה וכו').

### ריבוי תהליכים / ריבוי משימות

נרצה לאפשר למספר תהליכים לרוץ בו זמנית.  
אם למערכת יש רק מעבד אחד, הרי שהתהליכים לא ירוצו ממש בו זמנית, אלא שזמן המעבד יתחלק בין התהליכים השונים.  
מערכת ההפעלה היא זו שמטפלת בחלוקת הזמנים:  
חלוקת זמנים - כל תהליך מקבל ממערכת ההפעלה הזדמנות לרוץ.  
הגנה – תהליך לא יכול לשנות את המצב של תהליכים אחרים.

מערכת ההפעלה מנהלת את זיכרון התהליכים, את חלוקת זמן המעבד ואת המעבר בין התהליכים השונים.  
בכל רגע ישנו תהליך רץ אחד – "התהליך הנוכחי".  
התהליכים האחרים ממתנינים לביצוע או ממתנינים למשאב מערכת כלשהו שיתפנה.

### Virtual Continuity

תהליך יכול להיות "מופעל" או "מופסק" – כלומר, להתחיל להשתמש במעבד בצרכיו, או לשחרר את המעבד כך שתוכנות אחרות יוכלו להשתמש בו.  
מערכת ההפעלה נותנת לכל תהליך את האשליה כאילו הוא רץ בצורה מתמשכת, ללא כל הפסקות, ומנהלת באופן שקוף את החלפה – ההחלפה נקראת גם Context Switching.

## Context Switching

במהלך Context Switching, מערכת ההפעלה שומרת את המצב של התהליך הפעיל, וטוענת את המצב של התהליך החדש.

איזה נתונים מערכת ההפעלה שומרת? כל נתון שהתהליך הבא עלול לשנות:

- א. Program Counter (PC).
- ב. Program Status Word (PSW).
- ג. CPU Registers.
- ד. File access pointers.
- ה. זיכרון המשותף לתהליכים השונים.

תהליך מוותר על המעבד כאשר:

- א. מתרחשת פעולת קלט/פלט – התהליך ממתין לסיום הפעולה, והריצה בנתיים יכולה לעבור לתהליך אחר.
- ב. כאשר התהליך נכנס למצב המתנה (semaphore).
- ג. כאשר התהליך נכנס למצב השהייה (suspend).

הוויתור על התהליך כולל שמירת המידע של התהליך הנוכחי, ומעבר לתהליך אחר.

מערכת ההפעלה יכולה לכפות על תהליך לפנות את המעבד. נאמר על מערכות ההפעלה המבצעות זאת כי הן מבצעות את מדיניות "preemptive scheduling". אחרי שתהליך קיבל "מספיק" זמן עיבוד, מערכת ההפעלה מעבירה את המעבד לשימוש תהליך אחר.

### יתרונות וחסרונות החלפת תהליכים יזומה על ידי מערכת ההפעלה

יתרונות:

- ניצול מרבי של המעבד ושל ההתקנים.
- תהליך בודד אינו חוסים את שאר התהליכים.
- תמיכה יעילה במשתמשים אינטראקטיביים.
- תהליכים יכולים להתקשר זה עם זה ולהסתנכרן.

חסרונות:

- תהליך איננו מקבל את כל משאבי המערכת.
- זמן ביצוע תהליך משתנה מריצה לריצה (תלוי בעומס הנוכחי של המערכת).
- תהליך אינו יכול להסתמך על אי יציאתו מהמעבד במהלך זמן ריצתו.
- זמן החלפת התהליכים נושא תקורה – זהו זמן מבוזבז.

### מנגנון החלפת ההקשר

- על מנת להחליף תהליך P שרץ כרגע בתהליך אחר המוכן לריצה:
- מערכת ההפעלה שומרת את מצבו של P על מנת שתוכל לשחזר את המצב ולהחזירו לריצה בעתיד.
  - מערכת ההפעלה בוחרת תהליך Q מתוך כל התהליכים המוכנים לריצה.
  - מערכת ההפעלה מפסיקה את פעולת P.
  - מערכת ההפעלה משחזרת את Q ומתחילה להפעיל אותו.

### רכיבי מנגנון החלפת ההקשר

- מצבי תהליכים – המועמדים לריצה הם רק תהליכים במצב "מוכן לריצה".
- טבלת התהליכים – מסד נתונים השומר, לכל תהליך, את כל המידע הנדרש על ידי מערכת ההפעלה. נקרה גם Process Control Block (PCB).
- מבני נתונים לניהול תהליכים.
- מדיניות זימון – הרכיב שמחליט מיהו התהליך הבא שייכנס לריצה.

#### מצבי תהליכים:

- New – התהליך נוצר כרגע.
- Ready – התהליך מוכן לריצה כאשר מערכת ההפעלה תאפשר לו זאת.
- Current – התהליך הוא התהליך הנוכחי שרץ.
- Waiting – התהליך מחכה – לסיום פעולת קלט/פלט, לפינוי משאב וכו'.
- Ready – כאשר המשאב המבוקש יתפנה, התהליך יעבור למצב Ready.
- Terminated – ריצת התהליך הסתיימה.

טבלת התהליכים מכילה רשומה עבור כל תהליך. רשומת התהליך מכילה מידע על מצב התהליך ומעל ההקשר הדרוש להרצתו, וכן מידע נוספת שאינו רלוונטי להחלפת ההקשר. אינדקס הכניסה בטבלה של תהליך הוא המזהה שלו (pid).

#### מבנה הנתונים:

אפשרות אחת למבנה הנתונים, הינה תור הממתינים לריצה, למשל: תור דו כיווני הממוין לפי עדיפויות.

#### מדיניות התזמון:

התהליך הנוכחי הבא נבחר מבין אלו הממתינים לריצה. קיימים שני אלמנטים מרכזיים אשר שילובם מהווה את הבסיס למדיניות התזמון:

1. פרק זמן קצוב – לאחר לכל היותר פרק זמן, הנקרא בדרך כלל Quantum, בו רץ התהליך הנוכחי הוא מוחלף.
2. עדיפויות – התהליך הנוכחי הוא התהליך בעל העדיפות הגבוהה ביותר. עדיפויות יכולות להשתנות באופן דינמי.

שיקולים בקביעת עדיפויות:

- תהליכים של משתמשים "חשובים" יקבלו עדיפות גבוהה.

- עדיפות גבוהה לתהליכים שסיומם דחוף.
- תהליכים אינטראקטיביים יקבלו עדיפות גבוהה.

אלגוריתמי תזמון :

1. First Come First Serve (FCFS) – התהליך שהגיע ראשון לתור הממתנים יכנס ראשון.  
אלגוריתם זה נותן עדיפות לתהליכים צורכי מעבד, ממזער את ניצול ההתקנים ואינו מספק דרישות שיתוף.
2. Shortest Job First (SJF) – התהליך בעל יתרת זמן ריצה מינימלית ייכנס ראשון.
3. מדיניות Round Robin :
  - a. מחלקים את הזמן למרווחי זמן קבועים הנקראים Quantum.
  - b. כל התהליכים המתחרים מאורגנים בתור מעגלי.
  - c. התהליכים נכנסים לעיבוד לפי סדר הופעתם בתור.
  - d. כל תהליך מקבל לכל היותר Quantum זמן בכל סיבוב.