

נושא הפרויקט: **Microkernel**

שם התלמיד: **עדן פרנקל**

ת.ז. של התלמיד: **206964124**

שם בית הספר והעיר: **אורט גוטמן, נתניה**

שמות המנחים והמורים המלווים: **מיכאל צ'רנובילסקי וגד רוזנטל**

מועד הגשת המסמך: **03.11.2015**

**מבוא**

העבודה עוסקת בתחום מערכות ההפעלה, ובמימוש של מערכת הפעלה בסיסית על חומרה פיזית. מערכות ההפעלה פותחו על מנת לנהל את משאבי החומרה והתוכנה של המחשב, ועל מנת ליצור אבסטרקציה של החומרה הפיזית על מנת שמשתמש פשוט לא יצטרך להתמודד עם גישה ישירה לחומרה ויוכל ליצור תוכנות כלליות שירוצו על מערכות בעל רכיבי חומרה שונים.

אני, באופן אישי, רציתי להבין יותר לעומק כיצד פועלות מערכות ההפעלה ומבצעות את פעולותיהן השונות, וכיצד עובד המחשב ברמה הבסיסית ביותר. כבר כמה שנים שנחשפתי לעולם התכנות ולמדתי שפות Scripting (Python), שפות עיליות (C#, Java) ואף שפות סף (Assembly), אבל בכל אלו השתמשתי בפעולות מוכנות של מערכת ההפעלה שריחקו ממני את החומרה ויצרו בשבילי מן אשליה של סביבה וירטואלית שהייתי אמור לקחת כדבר מובן מאליו, אבל עדיין הטריד אותי שלא ידעתי כיצד זה באמת פועל, ביסוד.

המערכת הבסיסית שאני מתכנן ליצור תנהל באופן בסיסי את משאבי החומרה של המערכת (מסך, מקלדת, זיכרון ועוד) וכן את משאבי התוכנה של המערכת (תהליכים, מערכת קבצים ועוד). היא תציג למשתמש ממשק טקסטואלי בו הוא יוכל להזין פקודות ולהשתמש במערכת ההפעלה הבסיסית לביצוע פעולות שונות.

**תיאור המוצר**

המטרה שלי בעבודה זו היא ליצור מערכת עובדת שתוכל לנהל ברמה בסיסית את משאבי החומרה והתוכנה הבסיסיים של המחשב, ותוכל לספק למשתמש ממשק טקסטואלי שבו הוא יוכל להזין פקודות ולעשות שימוש בפעולות המערכת.

מערכת ההפעלה שלי תסייע למשתמש להשתמש במשאבי המחשב שלו, לעבוד בסביבה וירטואלית שתספק לו שירותים בסיסיים באמצעות פקודות פשוטות.

תיאור פשוט של תהליך ה-Boot ושל טעינת ה-Kernel:

תיאור כללי של התקשורת בין חלקי המערכת העיקריים (שיתוף הפעולה ביניהם):

User Interface

Interrupts Handling

File System

Task Management

Device Management

Memory Management

**אילוצים ודרישות**

המערכת תיאלץ להתמודד עם שגיאות חומרה, עם קלט לא תקין מהמשתמש, עם טיפול ב-Exceptions ו-Interrupts מהמערכת ועוד. לכל אחד מאלה היא תיתן את הטיפול המתאים: שגיאות חומרה יטופלו בצורה המתאימה (ניסיון חוזר, ויתור על הפעולה ופליטת הודעה מתאימה ועוד), ב-Exceptions ו-Interrupts היא תטפל בצורה שגרתית באמצעות טבלת IDT[[1]](#footnote-1), קלט מהמשתמש ייבדק ועוד.

מבחינת לוח הזמן יהיה אילוץ של להספיק כמה שיותר בזמן הנתון, משום שהפרויקט הזה הוא רחב ביותר, ואין סוף לדברים שניתן להוסיף לו ולפתוח אותו איתם.

אילוץ נוסף הוא רכישת הידע למימוש הפרויקט. גם שפת התכנות C וגם מרבית החומר שאני נדרש לדעת על מנת לממש את מערכת ההפעלה אינם נלמדים בכיתה, ועלי להיעזר במקורות חיצוניים (ספרים, אתרים שונים) על מנת לרכוש את הידע הדרוש למימוש מערכת הפעלה, לתקשורת עם החומרה, וכמובן את שפת התכנות.

**תיחום הפרויקט**

הפרויקט עוסק בעיקר בתחום מערכות ההפעלה ובמימוש שלהן. המערכת תעסוק בתחום התהליכים ותהיה מימוש של מערכת Multitasking, היא תאפשר תמיכה במערכת קבצים בסיסית והיא תציג ממשק טקסטואלי ותקבל פקודות דרך המקלדת. בנוסף, היא תטפל בצורה בסיסית במשאבי המערכת – תנהל את הזיכרון (תקצה אותו למטרות שונות), תציג פלט על המסך, תקבל קלט מהמקלדת, תטפל ב-Interrupts ו-Exceptions (לדוגמא – תטפל ב-Interrupt של ה-RTC[[2]](#footnote-2) וכך תקבל מידע על הזמן החולף) ועוד.

**סביבת העבודה**

שפות התכנות: C, Assembly.

סביבת העבודה: Editor (כמו Notepad++), GCC Compiler ו-GCC Linker, Cygwin – שורת פקודה המדמה Terminal של Linux ב-Windows.

הכלים הנדרשים לבדיקות: מחשב פיזי ודיסק-און-קי כדי לבצע ממנו את ה-Boot או מכונה וירטואלית.

**ניהול פרויקט עתידי**

|  |  |
| --- | --- |
| **יעד (מה יש לבצע)** | **זמן ביצוע (מתאריך עד תאריך)** |
| Boot, פלט למסך, קלט מהמקלדת, טיפול ב-Interrupts, הרצת קוד בסיסי של Command-Line עם מספר פקודות בסיסיות. | *בוצע* |
| ניהול הזיכרון, Multitasking | *עד 01.01.2016* |
| ניהול החומרה ומימוש מערכת קבצים | *01.01.2016-01.03.2016* |
| תוספות נוספות (אם יהיו), תיקוני באגים ויצירת התוצר הסופי | *01.03.2016-01.05.2016* |

1. Interrupt Descriptor Table – טבלה המכילה את הכתובות בזיכרון אליהם יש לקפוץ במידה ויש Exception או Interrupt מסוים. [↑](#footnote-ref-1)
2. Real Time Clock – חלק חומרתי המפעיל, בין השאר, Interrupt כל פרק זמן קבוע. [↑](#footnote-ref-2)