מטלת מנחה (ממ"ן) 11

הקורס: "מערכות הפעלה"

חומר הלימוד למטלה: ראו פירוט בסעיף "רקע"

מספר השאלות: 6 **משקל המטלה:** 12

סמסטר: 2019א **מועד אחרון להגשה:** 15.11.2018

הגשת המטלה: שליחה באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.
הסבר מפורט ב"נוהל הגשת מטלות המנחה".

החלק המעשי (70%)

כללי

בממ"ן זה עליכם לממש שתי ספריות לעבודה עם תהליכונים (threads) ברמת המשתמש (user-level). אחת הספריות תממש סמפורים בינאריים לעבודה עם קטעים קריטיים וספריה שנייה תממש מספר פונקציות המאפשרות יצירה והרצה של תהליכונים ברמת המשתמש ומדידת זמן הריצה ל profiling של תוכניות המשתמשות בספרייה זו.

מטרה

- הכרת ההיבטים המעשיים של מימוש תהליכונים ברמת המשתמש
- שימוש בסיגנלים
- שימוש ב-non-local branching
- timers
- profiling
- קטעים קריטיים

רקע

א) פרקים 2.3.5, 2.5.1, 2.2.1, 2.2.2, 2.2.3 בספר של Tanenbaum, "Modern operating systems".

ב) פרק 24.3 של The GNU C library

ג) פרק 23.4 של The GNU C library

ד) פרק "Libraries" מחוברת " Ubuntu 12.04 programming environment, making first steps"

ה) man pages של Linux - מידע על קריאות מערכת ופונקציות הבאות:

alarm, sigfillset, sigaction, swapcontext, getcontext, makecontext, steitimer, kill,getpid

תיאור המשימה

בממ״ן זה עליכם לממש שתי ספריות סטטיות:

1) libut.a - ספרייה פשוטה לעבודה עם תהליכונים ברמת המשתמש, שה-API שלה מוגדר בקובץ ut.h. קובץ זה מכיל תיאור מפורט לגבי תפקידה של כל פונקציה שעליכם לממש (אין לשנות קובץ זה, אך כמובן שבמידת הצורך ניתן להגדיר פונקציות עזר בקובץ C). הספרייה תתמוך רק בפעולות הבסיסיות ביותר, שהן יצירת התהליכונים, הרצתן ותזמונן. על מנת שלא להפוך את המשימה למסובכת מדי, הספרייה תממש רק מודל פשוט של שימוש בתהליכונים המבוסס על ההנחות הבאות:

1. כל תהליכון מריץ פונקציה אינסופית שמקבלת פרמטר יחיד מטיפוס int ומחזירה void.
   לא נטפל בסיום תהליכונים ובבדיקת סטטוס היציאה.
2. אין הוספה דינאמית של תהליכונים. המשתמש קודם יצור את כל התהליכונים, וא״כ יקרא ל-ut_start כדי להריץ את כל התהליכונים.
3. כל התהליכונים הם בעלי אותה עדיפות. תזמון התהליכונים יהיה בשיטת round-robin, כאשר גודל ה-quantum הוא שנייה אחת.
4. שימו לב שלא הגדרנו מצב blocked לתהליכונים. זאת מפני שבמודל שלנו ההנחה היא שתהליכונים לא מבצעים פעולות הגורמות לחסימה (blocking calles). לאחר הביצוע של ut_start, כל תהליכון יכול להיות באחד משני המצבים - רץ או מוכן לריצה. וודאו שאתם מבינים כי בהנחה כזאת כלל לא נצטרך לשמור את מצב ההליכונים מכוון שמנגנון התזמון שלנו תמיד יבחר את התהליכון הבא בתור ויריץ אותו.

בשלב ראשון של הכנת הממ״ן קראו את הסעיפים א), ב) וג) מחומר רקע והריצו והבינו את התוכניות demo1.c, demo2.c, demo3.c שסיפקנו לכם. התוכנית הראשונה מדגמה כיצד מתאפשר לשים ״שעון מעורר״ לתהליך ב Linux. התוכנית השנייה מרחיבה את הראשונה ומדגימה כיצד אפשר ליצור 2 ניבים של ריצה בתוכנית באמצעות המנגנון המכונה non-local jumping. התוכנית השלישית מדגימה כיצד אפשר לבצע רישום של זמן ריצה של תוכנית לצורך ה profiling.

בשלב שני עליכם לממש את הממשק המוגדר הקובץ ut.h. הממשק מגדיר פונקציות לאתחול הספריה, ליצירת תהליכון חדש ולהרצת התהליכונים שנוצרו. ut.h מממשת את מודל התהליכונים הפשוט שתיארנו לעיל. שימו לב ש demo2.c מדגימה כיצד ליצור 2 תהליכונים. אתם מתבקשים להכליל את הפתרון למספר תהליכונים. לכן, לאחר שהשלמתם את שני השלבים הקודמים כל שנותר לעשות הוא להעביר חלקים של הקוד מ demo2.c ל ut.c עם שינויים מינוריים.

ב ut.h עליכם לממש את ut_get_vtime המשמשת למדידת זמן הריצה של תהליכון. השתמשו בקוד של demo3.c שמשתמשת בשעון מסוג ITIMER_VIRTUALשישלח סיגנל SIGVTALRM כל 10) 100msec פעמים לשנייה). בכל פעם שהסיגנל מתקבל, יש להוסיף 100msec לשדה הזמן הווירטואלי של התהליכון האקטיבי בזמן קבלת הסיגנל.

2) libbinsem.a - ספרייה של סמפורים בינאריים שנועדו לשימוש ע"י התהליכונים מהסעיף הראשון. הקובץ binsem.h מגדיר את הטיפוס של סמפור בינארי ומתאר את הפונקציות הרלוונטיות (אין לשנות קובץ זה). עליכם לממש את הפונקציות שמוצהרות בקובץ זה, תוך כדי שימוש במקרו xchg() המוגדר בקובץ atomic.h. כמו כן, תסתמכו על העובדה שהחלפת התהליכונים מתבצעת כתוצאה מקבלת הסיגנל SIGALRM כדי לממש את ההמתנה ב- binsem_down() (כפי שפורט בסעיף הקודם, לתהליכונים שעליכם לממש לא מוגדר מצב blocked. יש לדמות את המצב ע"י כך שתהליכון ה"ממתין" בסמפור מייד לאחר קבלת ה-CPU ישלח סיגנל SIGALRM שיגרום להפעלת המתזמן ומעבר לתהליכון הבא).

לצורך הבדיקה של שתי הספריות סיפקנו לכם פתרון של בעיית הפילוסופים הסועדים בקובץ ph.c. בעיית הפילוסופים הסועדים מתוארת בפרק 2.5.1 בספר של Tanenbaum. כל פילוסוף רץ כתהליכון נפרד (לצורך זה משתמשים בספריית התהליכונים שהממשק שלה הוגדר ב ut.h. התהליכונים משתמשים בסמפורים שהוגדרו ב binsem.h). התוכנית תופעל ע"י הפקודה "ph<N>", כאשר N (בטווח מ-2 עד 32) הוא מספר התהליכונים(פילוסופים). התוכנית תופסק ע"י הקשת "Ctrl-C", לפני היציאה יודפסו זמני השימוש ב-CPU של כ"א מהתהליכונים.

כדי לקמפל את תוכנית הפילוסופים עם הספריות שתכתבו, תשתמשו ב Makefile שסיפקנו. שימו לב שעליכם לשנות את ה Makefile לפני ההגשה (ראו סעיף "הגשה" בהמשך).

טיפול בשגיאות

יש תמיד לבדוק את ערכי החזרה של קריאות מערכת ופונקציות סטנדרטיות של C. במקרה של כשלון, יש לפעול כפי שמוגדר בקבצים ut.h ו-binsem.h. בנוסף, במקרה של כשלון המערכת תוך כדי ביצוע של signal handler בספריית התהליכונים, יש להודיע על השגיאה באמצעות perror ולהפסיק את הביצוע ע"י exit(1).

הגשה

יש להגיש **כל** קבצי הקוד Makefile המייצר שתי ספריות סטטיות: libut.a ו- libbinsem.a. אין להגיש קבצים מקומפלים. ראה הוראות הגשה כלליות בחוברת הקורס.

את הקבצים המוגשים יש לשים בקובץ ארכיון בשם exYZ.zip (כאשר YZ הנו מספר המטלה). הכנת קובץ ארכיון מתבצעת ע"י הרצת הפקודה הבאה משורת הפקודה של Ubuntu:

zip exYZ.zip <ExYZ files>

הערה חשובה: בכל קובץ קוד שאתם מגישים יש לכלול כותרת הכוללת תיאור הקובץ, שם הסטודנט ומספר ת.ז.

פתרון ביה"ס

קיבלתם את שתי הספריות, libut.a ו- libbinsem.a , כפי שמומשו על ידינו. תוכלו להיעזר בהן בהכנת הממ"ן. למשל, לקמפל את תוכנית הבדיקה ph עם ספרייה אחת משלכם (שאותה אתם רוצים לבדוק) וספרייה השנייה של פתרון ביה"ס.

הערה: תוך כדי העבודה על הממ"ן תצטרכו להכיר ולהבין מספר נושאים שאינם פשוטים - זהו הקושי של ממ"ן זה. יחד עם זאת, הממ"ן לא ידרוש מכם הרבה עבודת תכנות. ניתן לממש את שתי הספריות בכ-100 שורות קוד בסה"כ.

החלק העיוני (30%)

שאלה 2 (10%)

א) מהי פעולת ה TRAP instruction (TRAP). תארו מתי היא מתבצעת ומה קורא בעת ביצועה.

ב) הסבירו מה קורה בעת הקריאה לפונקציית write של ה C library. בפרט הסבירו כיצד עוברים הפרמטרים של ה write למערכת הפעלה Linux וכיצד המערכת מטפלת ב write. יש התייחס הן למקרה של legacy system calls והן ל fast system calls.

ג) מה ההבדל בין write ל printf? תוכלו להעזר בקבצי מקור של C library מ http://www.gnu.org/software/libc/

שאלה 3 (5%)

הסבר את מדוע פתרון התור (strict alternation), איננו מהווה פתרון סביר. איזה תנאי/ים הוא מפר.

שאלה 4 (10%)

תקראו פרק 3 של המאמר שדן בנושא הוספת תהליכונים כספריה לשפה שלא תמכה בהם מלכתחילה והסברו מדוע תקן של Pthreads אינו מתאר באופן פורמאלי את מודל הזיכרון ואת הסמנטיקה של המקביליות הממומשות ב Pthreads. כיצד מפתחי התקן מסבירים מהו מודל הזיכרון בכל זאת?

שאלה 6 (5%)

הוכיחו כי בפתרון של Peterson תהליכים אינם ממתינים זמן אינסופי על מנת להיכנס לקטע קריטי. בפרט הוכיחו כי תהליך שרוצה להיכנס לקטע קריטי לא ממתין יותר ממה שלוקח מתהליך אחר להיכנס ולעזוב את הקטע הקריטי.

הגשת החלק העיוני

החלק העיוני יוגש כקובץ Word או כקובץ pdf. שם הקובץ צריך להיות exYZ.pdf או exYZ.doc (כאשר YZ הנו מספר המטלה).

בדיקה לאחר ההגשה

לאחר ההגשה יש להוריד את המטלה (חלק מעשי/עיוני) משרת האו"פ למחשב האישי ולבדוק שהקבצים אכן הוגשו באופן תקין ושניתן לקרוא אותם. בנוסף, הבדיקה של החלק המעשי תכלול את הצעדים הבאים:

- פתיחת ארכיון *exXY.zip* בספרייה חדשה (*new folder*).
- וידוא שכל הקבצים הדרושים נוצרו בספריה בה פתחתם את הארכיון.
- הרצת *make* ווידוא שכל ה *targets* נוצרו ללא שגיאות וללא *warnings*
- הרצת בדיקות רלונטיות לווידוא תקינות הריצה של החלק המעשי