**יישומי זמן אמת**

**יישום זמן אמת – יישום שצריך לעמוד בלוחות זמנים אנושיים מוגדרים בדייקנות.**

**חישוב, אפילו נכון, במידה והוא נעשה תוך אי עמידה בלוח הזמנים הוא חסר משמעות.**

**העמידה בלוח הזמנים הוא חלק מהנכונות של התוכנית.**

**מה שמבדיל בין יישומים כאלו ליישומים "רגילים" שלוחות הזמנים הם ספציפיים.**

**במערכות זמן אמת – שיפור ביצועים עשוי להיות מיותר.**

**זמן אמת אינו בהכרח חישוב סופר יעיל.**

**לפעמים יישום זמן אמת צריך להמתין.**

**מה שנכון – לפעמים עמידה בלוחות זמנים כרוכה בייעול.**

**לרוב מערכות זמן אמת יש את המאפיינים הבאים (חלקם צפויים וחלקם לא):**

* **הם בדרך כלל יישומים מחזוריים. זמן המחזור בדרך כלל קבוע (לפחות על פני שלבים). ליישומים הללו בדרך כלל אין להם זמן סיום ברור.**
* **בדרך כלל הם פותרות את משוואת הבקרה: X = F(X,I).**
* **לעיתים קרובות יישומים די גולמיים: מנהלים פסיקות, ק/פ בעצמם. רצים על ציוד ייעודי.**
* **(לא בהכרח אינטואיטיבי) הם בדרך כלל יישומים שהם מולטי תכנות multi-programming: קוד מרובה תהליכים או מרובה thread –ים. הסיבה העיקרית לכך היא הצורך במערכות כאלו לעבור בפתאומיות מנקודה אחת בקוד לנקודה שרירותית אחרת בקוד.**

**המנגנון הסטנדרטי של החלפת הקשר של תהליכים/thread-ים די מקל על העניין.**

* **היישום מכיל תהליכי מחזור ותהליכיים אסינכרוניים. קלט מהעולם החיצון מטופל בדרך כלל ע"י תהליכים אסינכרוניים.**
* **נקודת ההשקפה של ניתוח אלגוריתמים מיושמים במערכת היא יותר עניין של דטרמניזם מאשר יעילות. המקרה הגרוע ביותר הוא השיקול העיקרי.**
* **יישומי זמן אמת רגישים רצים מעל מערכת הפעלה מיוחדת ליישומי זמן אמת RTOS – Real Time Operating System.**
* **יישומי זמן אמת רגישים הם בדרך כלל היישום היחיד שרץ על המחשב שמריץ אותם.**

**דטרמינזם – התוכנה עשויה לרוץ ברציפות למשך שנים. זמן הבדיקה של התוכנה הרבה יותר קצרה. התכנון צריך להיות אי תלות באורך של הריצה של התוכנית – תסריטים כאלו או אחרים אינם משנים.**

**מתי אפשר לבדוק תוכנה בחודש ולהניח שהיא תעבוד במשך שנים?**

* **זמני תגובה שניתן לחזות בוודאות (אחידים).**
* **אי הצוברות נזקים בזמן.**