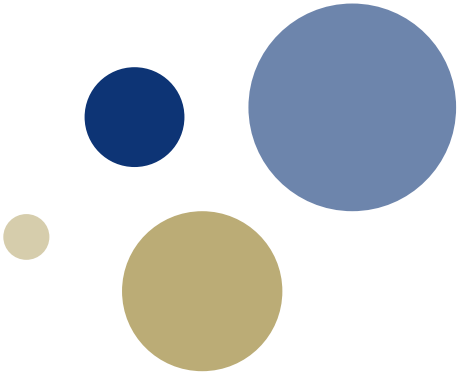
NTNU – Trondheim
Norwegian University of
Science and Technology



TTK4145 – Real-time Programming

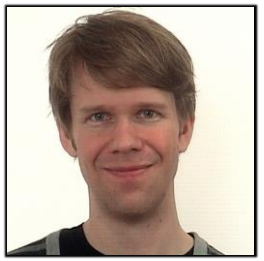
Lecture 1 - Introduction

Teaching staff

- Lecturer: Torleif Anstensrud (me)
  - Office hours: e-mail me
  - torleif.anstensrud@itk.ntnu.no

- Research Assistant: Kjetil Kjeka
  - Project
  - Exercises

- Guest lecturers
  - Anders Rønning Petersen
  - Øyvind Teig
  - Kristoffer Gregertsen



Practical Information

- Lectures
  - Thursday 10:15 – 13:00 (here)
  - Friday 14:15 – 16:00 (EL3, only as needed)
- Exercises
  - 10 in total
  - 1,2,3,6,7 and 8 requires approval
  - Kjetil has more details later
- Grading
  - Exam (75 %)
    - Date: 8th of June
    - Digital exam (as in on a computer)
  - Project (25 %)
    - Again, Kjetil has more details later

Motivation

Why real-time programming?

- Surrounded by software
  - Embedded systems are everywhere (Internet of Things)
  - Commercial and industrial (different requirements)

- Challenges for software development
  - Timing (things happen in real-time)
    - Controllers need measurements at specific intervals
    - Tasks can't wait for other tasks indefinitely
  - Distributed systems/multi thread
    - Subsystems cooperating (multicore processors, large-scale plants)
    - Shared variables (ensure consistent data)
  - Safety/Reliability/Availability
    - Rebooting (laptop versus respirator)
    - If (when) things go wrong, software must detect and handle it

Not only embedded systems

- Bugs can’t be completely removed
  - A test can only show the presence of a fault, not an absence
  - Software must account for bugs (within reason)
- Tasks can be interrupted unexpectedly
  - Pulling the plug
  - Remote server crashing
  - External damage to system
- Data can be changed unexpectedly
  - Cosmic rays flipping bits
  - Implicit type casting
  - Overflow

Curriculum?

Course Learning Goals, TTK4145

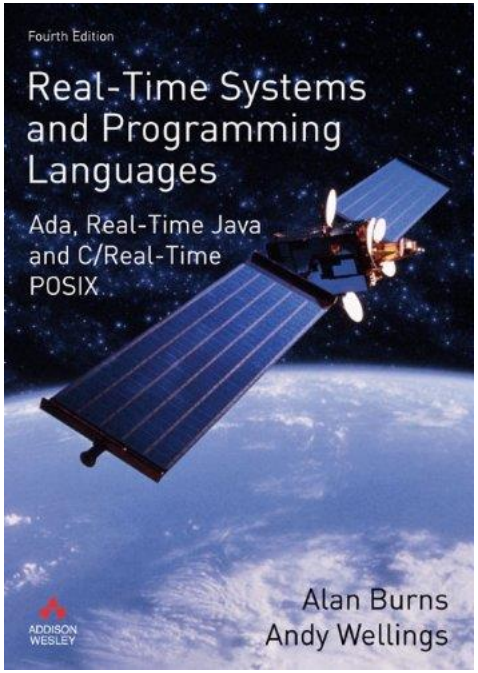
- General maturation in software engineering/computer programming.
- Ability to use (correctly) and evaluate mechanisms for shared variable synchronization.
- Understanding how a deterministic scheduler lays the foundation for making real-time systems.
- Insight into principles, patterns and techniques for error handling and consistency in multi thread / distributed systems.
- Knowledge of the theoretical foundation of concurrency, and ability to see how this can influence design and implementation of real-time systems

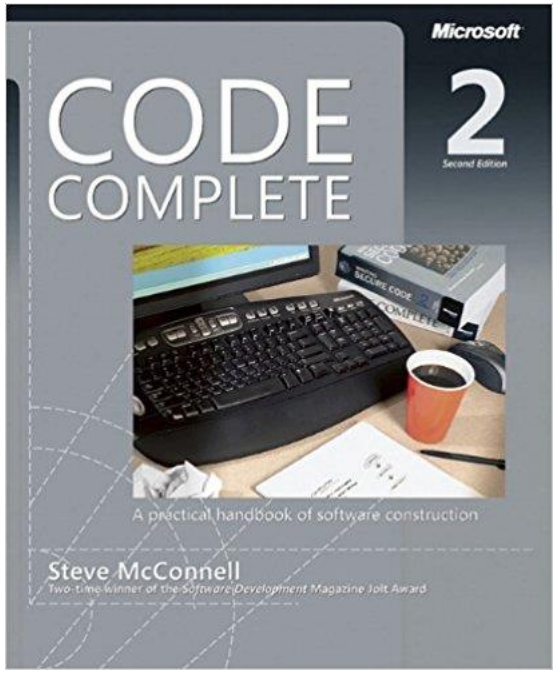
Learning goals: Fault Tolerance Basics

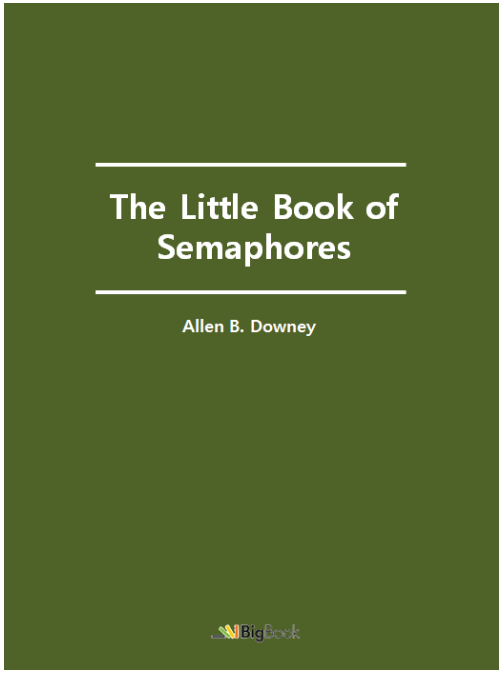
- Understand and use terms (like): Reliability. Failure vs fault vs error. Failure modes. Acceptance test. Fault prevention vs. tolerance. Redundancy, Static vs. Dynamic. Forward/ Backward error recovery.
- Understand, use and evaluate techniques (like): N-version programming. Recovery blocks. Error detection. Failure mode merging. Acceptance tests.

Recommended reading

- Burns & Wellings (old textbook)
- Code Complete
- Little Book of Semaphores
- PDFs available on Blackboard







Quality Assurance

Separate slides