Real-Time & Embedded Systems 2003

Uwe R. Zimmer – The Australian National University
Overview, Perspectives, Paths, Methods, and some Theory

into/for/about Real-Time & Embedded Systems
 anybody who …

  … would like to see immediate real-world involvement in his/her work

  … would like to learn how to create predictability and fault-tolerant complex systems

  … would like to know more about the usage of 95% of all µprocessors
The course will be given by

\textit{Uwe R. Zimmer}

and

\textit{Christfried Webers}
how will this all be done?

**Lectures:**
- 3 per week ... all the nice stuff and theory
  Tuesday, 13-14; Wednesday 16-17; Thursday 16-17 – all in PSYC - G06

**Laboratories:**
- 2 hours per week ... all the rough stuff and practice
  Thursday 11-13 or Friday 9-11 – all in CSIT N112
  laboratory-enrolment: https://cs.anu.edu.au/streams/

**Resources:**
- introduced in the lectures and collected on the course page:
  ... as well as schedules, slides, sources, etc. pp. ... keep an eye on this page!

**Assessment:**
- exam at the end of the course (70%) plus laboratories performance (30%)
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
   - Staking out the field
   - Features (and non-features) of a real-time system
   - Components of a real-time system
   - Real-time languages
     - Ada95
     - Esterel
     - Pearl
     - Real-time JAVA
     - POSIX

2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages

2. Physical coupling
   - Physical values
   - Introduction to sensors
   - Frequently employed sensors

3. Interfaces

4. Time & embodiment

5. Asynchronism

6. Synchronisation

7. Scheduling

8. Resource control

9. Reliability & fault-tolerance
Real-Time & Embedded Systems

Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
   - Analogue signal chain in a digital system
   - Analog-Digital converters
   - Interface devices
   - μ-controllers
4. Time & embodiment
5. Asynchronism
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
   • What is time? / What is embodiment?
   • Time: notion, delays, time-out
   • Interfacing with time
   • Specifying timing requirements
   • Satisfying timing requirements
5. Asynchronism
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
   - Interrupts, signals, exceptions
   - Atomic Actions
   - Asynchronous transfer of control
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Real-Time & Embedded Systems

Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronisation
   - Variable-based synchronization
   - Message based-synchronization
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronization
7. Scheduling
   - Basic real-time scheduling
   - Real-world extensions
   - Language support
8. Resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronization
7. Scheduling
8. Resource control
   • Resource synchronization primitives
   • Resource reclaiming schemes
   • Real-time resource control
9. Reliability & fault-tolerance
Topics in this course

1. Introduction & real-time languages
2. Physical coupling
3. Interfaces
4. Time & embodiment
5. Asynchronism
6. Synchronization
7. Scheduling
8. Resource control
9. Reliability & fault-tolerance
   - Terminology
   - Faults
   - Redundancy
   - Reduce & Formalise
Table of Contents

1. Introduction & Real-Time Languages
   1.1. Features (and non-features) of a real-time system
   1.2. Components of a real-time system
   1.3. Real-time languages criteria
   1.4. Examples of actual real-time languages:
       • Ada95, Esterel, Pearl, Real-time JAVA, POSIX

2. Physical coupling
   2.1. Physical phenomena
   2.2. Measuring temperature
       • Thermoelements, thermocouples, thermistors, noise temperature measurement (and others)
   2.3. Measuring range and relative speed
       • Triangulation, time of flight, intensity, Doppler methods, interferometry

3. Converters & Interfaces
   3.1. Analogue signal chain in a digital system
       • Sampling data aliasing Nyquist's criterion oversampling
       • Quantization (LSB, rms noise voltage, SNR, ENOB) – Missing codes, DNL, INL
   3.2. A/D Converters: flash, pipelined-flash, SAR, Σ-∆, n-th order Σ-∆
   3.3. Examples:
       • Fast and simple A/D converter example

4. Time & Space
   4.1. What is time? / What is embodiment?
       • Approaches by different faculties to understand the basis for this course
   4.2. Interfacing with time
       • Formulating local time-dependent constraints – Access time, delay processes, detect timeouts (in different languages)
   4.3. Specifying timing requirements
       • Formulating global timing-constraints – Understanding time-scope parameters (and expressing them in different languages)
   4.4. Satisfying timing requirements
       • Real-time logic and complex systems environment

5. Asynchronism
   5.1. Interrupts / Signals
       • Device / system / language / operating-system level interrupt control
       • Characteristics of interrupts and signals
   5.2. Exceptions
       • Exception classes / granularity / parametrization / propagation – Resumption and termination, specific language issues

6. Synchronization
   6.1. Shared memory based synchronization
       • Flags, condition variables, semaphores, conditional critical regions, monitors, protected objects.
       • Guard evaluation times, nested monitor calls, deadlocks, simultaneous reading, queue management.
   6.2. Message based synchronization
       • Synchronization models, addressing modes, message structures
       • Selective accepts, selective calls

7. Scheduling
   7.1. Basic real-time scheduling
       • Fixed Priority Scheduling (FPS) with Rate Monotonic (RMAO) Deadline Monotonic Priority Ordering (DMPO)
       • Earliest Deadline First (EDF)
   7.2. Real-world extensions
       • Aperiodic, sporadic, soft real-time tasks – Deadlines shorter than period – Cooperative and deferred pre-emption scheduling

8. Resource control
   8.1. Resource synchronization primitives
       • Evaluation criteria for resource synchronization methods
       • Atomicity, liveliness, and double interaction
   8.2. Resource reclaiming schemes
       • Basic reclaiming, early start, and restricted vector algorithms
       • Resource reclaiming with task migration
   8.3. Real-time resource control
       • Policy and run-time issues to be considered

9. Reliability
   9.1. Terminology
       • Faults, Errors, Failures – Reliability
   9.2. Faults
       • Fault avoidance, removal, prevention Fault tolerance
   9.3. Redundancy
       • Static (TMR, NMR) and dynamic redundancy
       • N-version programming, and dynamic redundancy in software design
   9.4. Reduce & Formalise
       • Ada95 Ravenscar profile
       • Real-time Logic

© 2003 Uwe R. Zimmer, The Australian National University