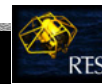


RES

Real-Time & Embedded Systems 2009

Uwe R. Zimmer – The Australian National University



Real-Time & Embedded Systems

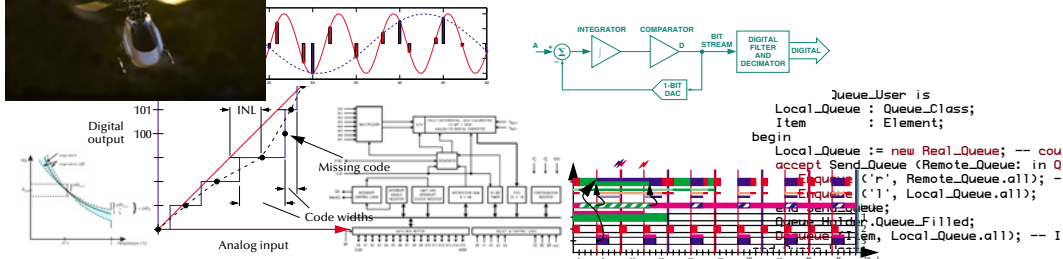


what is offered here?

Overview, Perspectives, Paths, Methods, and some Theory



into/for/about Real-Time & Embedded Systems



Real-Time & Embedded Systems



who could be interested in this?

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 μ processors



Real-Time & Embedded Systems



who are these people? – introduction



The course will be given by

Uwe R. Zimmer

and

Pat Bernardi





how will this all be done?

Lectures:

- 2 lectures à 1.5h per week ... all the nice stuff and theory
Monday & Tuesday, 9:00-10:30; in MCC T4, PSYC G8 resp.

Laboratories:

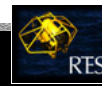
- 2 hours per week ... all the rough stuff and practice
Monday 11:00-13:00 or Tuesday 13:00-15:00 –in CSIT N114, N113 resp.
laboratory-enrolment: <https://cs.anu.edu.au/streams/>

Resources:

- introduced in the lectures and collected on the course page:
<http://cs.anu.edu.au/student/comp4330/>
... 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 | 6. Synchronisation |
| 2. Physical coupling | 7. Scheduling |
| 3. Interfaces | 8. Resource control |
| 4. Time & embodiment | 9. Reliability & fault-tolerance |
| 5. Asynchronism | |



Central textbook

Supporting literature

[Burns01]

Alan Burns and Andy Wellings
Real-Time Systems and Programming Languages
Addison Wesley, third edition, 2001

[Ari90]

M. Ben-Ari
Principles of Concurrent and Distributed Programming
Prentice Hall, 1990

[Cohen96]

Norman H. Cohen
Ada as a second language
McGraw-Hill series in computer science, 2nd edition

[Ada95RM] (on-line version available)

Ada Working Group
Ada 95 Reference Manual
– *Language and Standard Libraries*
ISO/IEC 8652:1995(E) with COR.1:2000,
June 2001

many more references and links are available on the course page

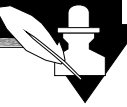


Table of Contents

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, thermoresistors, thermistors, noise temperature measurement) and others
- 2.3. Measuring range and relative speed
 - Triangulation, time of flight, intensity, Doppler methods, interferometry
- 2.4. Examples:
 - Time-of flight ultrasound, time-of-flight laser, Doppler current profiler

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, $\Sigma\text{-}\Delta$, n-th order $\Sigma\text{-}\Delta$
- 3.3. Examples:
 - Fast and simple A/D converter example

- Multi-channel A/D data logging interface example
- Simple 8-bit μ controller example
- Complex 32-bit μ controller example TPU: programming, atomic states, pen-gine scheduling, max. latency analysis NEXUS debugging port

3.4. General device handling / sampling control / language requirements

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 approach

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 / parametrisation / propagation – Resumption and termination, specific language issues

5.3. Atomic Actions

- Definition / requirements / failure cases / implementation / error recovery

5.4. Asynchronous transfer of control / Interrupts in context

- Interrupts and ATC in real-time Java and Ada95

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.
 - Synchronization and object orientation, blocking operations and re-queuing.
- 6.2. Message based synchronization
 - Synchronization models, addressing modes, message structures
 - Selective accepts, selective calls
 - Indeterminism in message based synchronization

7. Scheduling

- 7.1. Basic real-time scheduling
 - Fixed Priority Scheduling (FPS) with Rate Monotonic (RMPO) 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

- Fault tolerance in terms of exception handling considerations – Synchronized talks (priority inheritance, priority ceiling protocols)


7.3. Language support

- Ada95, POSIX  static, off-line analysis mostly – RT-Java  on-line, dynamic scheduling

8. Resource control

- 8.1. Resource synchronization primitives
 - Evaluation criteria for resource synchronization methods
 - Atomicity, liveness, and double interaction
- 8.2. Resource reclaiming schemes
 - Basic reclaiming, early start, and restriction 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