



# Real-Time & Embedded Systems

Uwe R. Zimmer - The Australian National University



# Real-Time & Embedded Systems 2019



## Organization & ToC

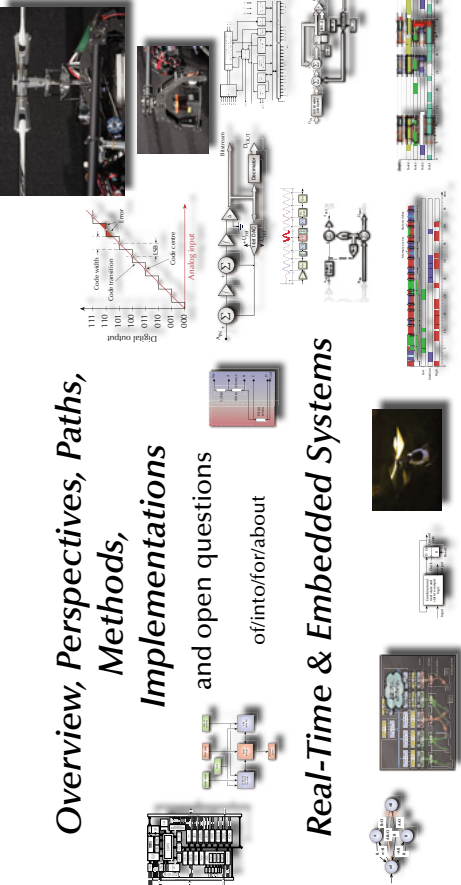
Uwe R. Zimmer - The Australian National University

# Organization & ToC

*what is offered here?*

Overview, Perspectives, Paths, Methods, Implementations and open questions of/into/for/about

## Real-Time & Embedded Systems



# Organization & ToC

*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 predictable and fault-tolerant, complex systems.
- ... would like to know more about the usage of >95% of all processors.

## Organization & ToC

### who are these people? – introduction

This course will be given by

Uwe R. Zimmer



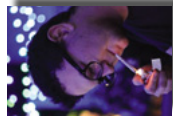
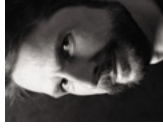
Tutoring and labs by



Calum Snowdon &  
Michael Bennett

Electronics design by

Mark Turner



## Organization & ToC

### Textbooks (sort of ...)

[Burns2009]

Alan Burns and Andy Wellings  
*Real-Time Systems and Programming Languages*  
Addison Wesley, fourth edition, 2009

[Burns2007]

Alan Burns & Andy Wellings  
*Concurrent and Real-Time Programming in Ada*  
Cambridge University Press, 2007

[McCormick11]

McCormick, J. W., Singhoff, F., & Hugues, J.  
*Building Parallel, Embedded, and Real-Time Applications with Ada*.  
Cambridge University Press, 2011.

... plus specific references for each topic (all on the course site).

## Organization & ToC

### how will this all be done?

#### Lectures:

- 2x 1.5 h lectures per week ... all the nice stuff  
Monday, 15:00 (Engineering Theatre) and Thursday 09:00 (Forestry Theatre)

#### Laboratories:

- 2 hours per week ... all the rough stuff  
time slots: on our web-site – all in CSIT laboratories  
-enrolment: <https://cs.anu.edu.au/streams/>

#### Resources:

- Introduced in the lectures and collected on the course page:  
<https://cs.anu.edu.au/courses/comp4330/> ... as well as schedules,  
slides, sources, link to forums, etc. pp. ... keep an eye on this page!

#### Assessment:

- Exam at the end of the course (70%) plus one assignment (30%)  
– both are tested in oral exams (unless enrolment numbers require otherwise).

## Organization & ToC

### Topics

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

## Organization & ToC

### Topics

1. **Introduction & Real-time languages**
- 1.1. Staking out the field
- 1.2. Features (and non-features) of a real-time system
- 1.3. Components of a real-time system
- 1.4. Real-time languages
  - Ada
  - Esterel
  - Pearl
  - VHDL
  - Timed CSP
  - 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**

## Organization & ToC

### Topics

1. **Introduction & Real-time languages**
2. **Physical coupling**
- 2.1. Physical values
- 2.2. Introduction to sensors
- 2.3. Frequently employed sensors
3. **Interfaces**
4. **Time & Embodiment**
5. **Asynchronism**
6. **Synchronisation**
7. **Scheduling**
8. **Resource control**
9. **Reliability & Fault-tolerance**

## Organization & ToC

### Topics

1. **Introduction & Real-time languages**
2. **Physical coupling**
3. **Interfaces**
- 3.1. Analogue signal chain in a digital system
- 3.2. Analog-Digital converters
- 3.3. Interface devices
- 3.4.  $\mu$ -controllers
4. **Time & Embodiment**
5. **Asynchronism**
6. **Synchronisation**
7. **Scheduling**
8. **Resource control**
9. **Reliability & Fault-tolerance**

## Organization & ToC

### Topics

1. **Introduction & Real-time languages**
2. **Physical coupling**
3. **Interfaces**
4. **Time & Embodiment**
- 4.1. What is time? / What is embodiment?
- 4.2. Time: notion, delays, time-out
- 4.3. Interfacing with time
- 4.4. Specifying timing requirements
- 4.5. Satisfying timing requirements
5. **Asynchronism**
6. **Synchronisation**
7. **Scheduling**
8. **Resource control**
9. **Reliability & Fault-tolerance**

## Organization & ToC

### Topics

1. *Introduction & Real-time languages*
2. *Physical coupling*
3. *Interfaces*
4. *Time & Embodiment*
5. *Asynchronism*
- 5.1. Interrupts, signals, exceptions
- 5.2. Atomic Actions
- 5.3. Asynchronous transfer of control
6. *Synchronisation*
7. *Scheduling*
8. *Resource control*
9. *Reliability & Fault-tolerance*

## Organization & ToC

### Topics

1. *Introduction & Real-time languages*
2. *Physical coupling*
3. *Interfaces*
4. *Time & Embodiment*
5. *Asynchronism*
6. *Synchronisation*
- 6.1. Variable-based synchronization
- 6.2. Message-based synchronization
7. *Scheduling*
8. *Resource control*
9. *Reliability & Fault-tolerance*

## Organization & ToC

### Topics

1. *Introduction & Real-time languages*
2. *Physical coupling*
3. *Interfaces*
4. *Time & Embodiment*
5. *Asynchronism*
6. *Synchronisation*
7. *Scheduling*
- 7.1. Basic real-time scheduling
- 7.2. Real-world extensions
- 7.3. Language support
8. *Resource control*
9. *Reliability & Fault-tolerance*

## Organization & ToC

### Topics

1. *Introduction & Real-time languages*
2. *Physical coupling*
3. *Interfaces*
4. *Time & Embodiment*
5. *Asynchronism*
6. *Synchronisation*
7. *Scheduling*
8. *Resource control*
- 8.1. Resource synchronization primitives
- 8.2. Resource reclaiming schemes
- 8.3. Real-time resource control
9. *Reliability & Fault-tolerance*

# Organization & ToC

## Topics

1. Introduction & Real-time languages
  - 9.1. Terminology
  - 9.2. Faults
  - 9.3. Redundancy
  - 9.4. Reduce & Formalise
2. Physical coupling
3. Interfaces
4. Time & Embodiment
5. Asynchronism
6. Synchronisation
7. Scheduling
8. Resource control
9. Reliability & Fault-tolerance

# Organization & ToC

## Table of Contents

<p>1. Introduction &amp; Real-time Languages</p> <p>1.1. Features (and non-features) of a real-time system</p> <p>1.2. Components of a real-time system</p> <p>1.3. Real-time languages criteria</p> <p>1.4. Examples of actual real-time languages:</p> <ul style="list-style-type: none"> <li>• Ada, Esterel, Pearl, VHDL, Timed CSP, Real-time Java, POSIX</li> </ul> <p>2. Physical coupling</p> <p>2.1. Physical phenomena</p> <p>2.2. Measuring temperature</p> <ul style="list-style-type: none"> <li>• Thermocouples, thermocouples, temperature measurement) and others</li> </ul> <p>2.3. Measuring range and relative speed</p> <ul style="list-style-type: none"> <li>• Triangulation, time of flight, intensity</li> <li>• Doppler methods, interferometry</li> </ul> <p>2.4. Examples:</p> <ul style="list-style-type: none"> <li>• Light, light, ultrasonic, time-of-flight laser, Doppler current profiler</li> </ul> <p>3. Converters &amp; Interfaces</p> <p>3.1. Analogue signal chain in digital system</p> <ul style="list-style-type: none"> <li>• Sampling rate, aliasing, Nyquist's</li> <li>• Quantization (LSB, rms noise voltage, SNR, ENOB) –Missing codes, DNL, INL</li> </ul> <p>3.2. AD converters: flash pipelined, flash, SAR, <math>2^N</math>, <math>n</math>-th order <math>2^N</math></p> <p>3.3. Examples:</p> <ul style="list-style-type: none"> <li>• Single-chip AD converter example</li> <li>• Multi-channel AD data log,</li> <li>• AD interface example</li> </ul>	<p>cases / implementation / error recovery</p> <p>5.4. Asynchronous transfer of control</p> <ul style="list-style-type: none"> <li>• Interrupts and AIC, DIT, real-time Java and Ada</li> </ul> <p>6. Synchronization</p> <p>6.1. Shared memory based synchronization</p> <ul style="list-style-type: none"> <li>• Flags, critical sections, monitors, protected objects, Guard variables, mutex, read/write locks, semaphore</li> <li>• Synchronization and object orientation, blocking operations and re-queuing.</li> </ul> <p>6.2. Synchronization models, address</p> <ul style="list-style-type: none"> <li>• Ring modes, message structures</li> <li>• Selective accepts, selective calls</li> <li>• Mail transmission in message based communication</li> </ul> <p>7. Scheduling</p> <p>7.1. Basic real-time scheduling</p> <ul style="list-style-type: none"> <li>• Priority Scheduling, with Rate Monotonic Scheduling (RM), Monotonic Priority Ordering (DMPO)</li> </ul> <p>7.2. Real-world extensions</p> <ul style="list-style-type: none"> <li>• Deadlines shorter than period – Cooperative and deferred pre-emption scheduling – Fault tolerance in terms of resource management</li> <li>• Inheritance, priority ceiling protocols)</li> </ul> <p>7.3. Language support</p>	<p>8. Resource control</p> <p>8.1. Resource synchronization primitives</p> <ul style="list-style-type: none"> <li>• Evaluation criteria for resource synchronization methods</li> <li>• Atomicity, liveness, and double interaction</li> </ul> <p>8.2. Resource reclaiming schemes</p> <ul style="list-style-type: none"> <li>• Basic reclaiming, early start, and restriction vector algorithms</li> <li>• Resource reclaiming with reservation</li> </ul> <p>8.3. Real-time resource control</p> <ul style="list-style-type: none"> <li>• Policy and run-time issues to be considered</li> </ul> <p>9. Reliability</p> <p>9.1. Terminology</p> <ul style="list-style-type: none"> <li>• Faults, Errors, failures – Reliability</li> </ul> <p>9.2. Faults</p> <ul style="list-style-type: none"> <li>• Fault avoidance, removal, prevention, Fault tolerance</li> </ul> <p>9.3. Redundancy</p> <ul style="list-style-type: none"> <li>• Static (MRC, NMR) and dynamic (NMR, MRC) redundancy</li> <li>• System reprogramming and dynamic redundancy in software design</li> </ul> <p>9.4. Reduce &amp; Formalise</p> <ul style="list-style-type: none"> <li>• Ada Ravenscar profile</li> <li>• Real-time Logic</li> </ul>
---	--	---