950

Real-Time & Embedded Systems 2019

10

Summary

Uwe R. Zimmer - The Australian National University

951

Summary

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:**
  - Ada, Esterel, Pearl, VHDL, Timed CSP, 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 adigital 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-Δ, n-th order Σ-Δ**
- **3.3. Examples:**
  - Fast and simple A/D converter example
  - Multi-channel A/D data logging interface example
  - Simple 8-bit μcontroller example
  - Complex 32-bit μcontroller example: TPU: μprogramming, atomic states, μengine 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 Ada

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**
  - Ada, POSIX (static, off-line analysis mostly) — RT-Java (on-line, dynamic scheduling)

8. Resource control

- **8.1. Resource synchronization primitives**
  - Evaluation criteria for resource synchronisation methods
  - Atomicity, liveliness, 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**
  - Ada Ravenscar profile
  - Real-time Logic

© 2019 Uwe R. Zimmer, The Australian National University

952

Summary

Summary

Introduction & Real-Time Languages

- **Features** (and non-features) of a real-time system
  - Features, definitions, scenarios, and characteristics.
- **Components** of a real-time system
  - Converters, interfaces, sensors, actuators, communication systems, controllers, …
- **Software layers** of a real-time system
  - Algorithms, operating systems, protocols, languages, concurrent and distributed systems.
- Real-time languages **criteria**
  - Mostly high integrity, predictable languages with means for explicit time scopes.
- Examples of actual real-time **languages**

© 2019 Uwe R. Zimmer, The Australian National University

953

Summary

Summary

Physical coupling

- **Physical phenomena**
- **Measuring temperature**
  - Thermoelements, thermocouples, Thermoresistors, Thermistors, Noise temperature measurement) and many others …
- **Measuring range and relative speed**
  - Triangulation,Time of flight, Intensity, Doppler methods, Interferometry
- Examples: **Common acoustical and optical sensors**

© 2019 Uwe R. Zimmer, The Australian National University

954

Summary

Summary

Converters & Interfaces

- **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
- **A/D converters:**
  - Integrating (Single- / Dual-slope), Flash, Pipelined, SAR, Tracking, Σ-Δ, Σ-Δ DDA, n-th order Σ-Δ.
- **Examples:**
  - Fast and simple A/D converter example: National Semiconductor ADC08200
  - Multi-channel A/D data logging interface example: National Semiconductor LM12L458
  - Simple 8-bit μ-controller example: Motorola MC68HC05, Propeller.
  - Complex 32-bit μ-controller examples: AVR32 and Motorola MPC565 (including TPUs).
- **General device handling / sampling control / language requirements**

© 2019 Uwe R. Zimmer, The Australian National University

955

Summary

Summary

Time & Space

- **What is time? / What is embodiment?**
  - Approaches by different faculties to understand the foundations of this course
- **Interfacing with time**
  - Formulating local, time-dependent constraints
  - Access time, delay processes, timers
  - Timeouts, asynchronous transfer of control
- **Specifying timing requirements**
  - Formulating global timing-constraints
  - Understanding time-scope parameters (and expressing them in different languages)
- **Satisfying timing requirements**
  - Real-time logic approach & Complex systems approach

© 2019 Uwe R. Zimmer, The Australian National University

956

Summary

Summary

Asynchronism

- **Interrupts / Signals**
  - Device / system / language / operating-system level interrupt control.
  - Characteristics of interrupts and signals.
- **Exceptions**
  - Exception classes / granularity / parametrisation / propagation.
  - Resumption and termination, specific language issues.
- **Atomic Actions**
  - Definition / requirements / failure cases / implementation / error recovery.
- **Asynchronous transfer of control / Interrupts in context**
  - Interrupts and ATC in real-time Java and Ada.

© 2019 Uwe R. Zimmer, The Australian National University

957

Summary

Summary

Synchronization

- **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.
- **Message based synchronization**
  - Synchronization models
  - Addressing modes
  - Message structures
  - Examples

© 2019 Uwe R. Zimmer, The Australian National University

958

Summary

Summary

Scheduling

- **Basic real-time scheduling**
  - Fixed Priority Scheduling (FPS) with Rate Monotonic (RMPO) and Deadline Monotonic Priority Ordering (DMPO).
  - Earliest Deadline First (EDF).
- **Real-world extensions**
  - Aperiodic, sporadic, soft real-time tasks.
  - Deadlines different from period.
  - Synchronized talks (priority inheritance, priority ceiling protocols).
  - Cooperative and deferred pre-emption scheduling.
  - Fault tolerance in terms of exception handling considerations.
- **Language support**
  - Ada, POSIX

© 2019 Uwe R. Zimmer, The Australian National University

959

Summary

Summary

Resource Control

- **Resource synchronization primitives**
  - Evaluation criteria for resource synchronization methods.
  - Atomicity, liveliness, and double interaction.
- **Resource reclaiming schemes**
  - Basic reclaiming
  - Early start algorithm
  - Restriction vector
  - Resource reclaiming with task migration
- **Real-time resource control**
  - Policy and run-time issues to be considered.

© 2019 Uwe R. Zimmer, The Australian National University

960

Summary

Summary

Reliability

- **Terminology**
  - Faults, Errors, Failures – Reliability.
- **Faults**
  - Fault avoidance, removal, prevention ☞ Fault tolerance.
- **Redundancy**
  - Static (TMR, NMR) and dynamic redundancy.
  - N-version programming, and dynamic redundancy in software design.
- **Reduce & Formalise**
  - Ravenscar profile.
  - Real-time logic.

© 2019 Uwe R. Zimmer, The Australian National University

961