Design of Real-Time Software

Introduction

Reinder J. Bril
Technische Universiteit Eindhoven
Department of Mathematics and Computer Science
System Architecture and Networking Group
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
r.j.bril@tue.nl, http://www.win.tue.nl/~rbril

January 2014
Overview

• Real-Time Systems
  – Definition
  – Examples
  – Components

• Specification aspects
Real-time systems

• **Definition:** a system in which correct operation depends not only on logical results of computation of values *but also on the time these results are produced*
  - Stankovic in Byte, ‘92

• Hence, *performance* plays a role....

• .... though not just high, but *predictable*!
Predictable versus fast

- Example: inflation of an air bag

(by courtesy of Damir Isovic)

- Conclusions:
  1. real-time ≠ fast
  2. real time: fulfill individual timing requirements
Example: Anti-lock Braking System (ABS)

(by courtesy of Damir Isovic)

1. Brake pedal pushed

2. Pressure passed to the brake fluid

3. Wheel disc brakes squeezed

4. If the brake pedal is pushed too hard, the wheel will lock → a sensor detects this and notifies the controller

5. Controller releases the pressure on the discs by releasing some brake fluid in a container

6. The fluid is pumped back to repeat the pressure on the discs

7. Entire process is repeated about 15 times/sec
Example: ABS – cnt’d

• Electronic system:
  – *controller*: requires an ECU;
  – *sensor*:
    • detects that the wheel will lock;
    • either generates an interrupt or is “readable”
  – *actuator*: release and repeat the pressure on the discs

• Distributed:
  – controller, sensors, and actuators at different locations;
  – requires wires or a “network”;

• Embedded and invisible to the driver
Example: robots at conveyor belt
RT(C)S components

UI
Controlling software
Controlling system
Controlled system

Real-Time System
Real-Time Computer
Real-Time SW

Sensors
Actuators

IN
OUT

Reinder J. Bril, r.j.bril@tue.nl
TU/e Informatica, System Architecture and Networking
RT(C)S components (cnt’d)

- Terminals (sensors, actuators) are modelled inside a computer as a component
- Three interface elements for control (though not all parts need to be there at any time)
  - sensor related:
    1. events
    2. observe (internal state)
  - actuator related:
    3. command
- Control ‘steers’ the data streams
Top down refinement

- Requirement spec → Design
- Derived requirement spec → Design
- Derived requirement spec → Design

Real-time System

Controlled System

Real-time Computer System

Controlling System

Real-time SW

Controlling SW
Overview

- **Real-Time Systems**
- **Specification aspects**
  - Functional behaviour
  - Timeliness; constraints derived from
    - functional specification of the system
    - design choices of the system
  - Physical constraints and boundary conditions
    - distribution, concurrency
    - resources
    - size, power
  - Dependability
    - Availability, Reliability
    - Safety, Security
    - Predictability
Functional behaviour

• Top level description of functionality of (controlled) *system*. E.g.,
  • The items that go on the conveyor belt must be painted
  • The temperature must remain between 300K and 301K
  • $K$ items must be packed in each box
  • The traffic light system operates according to the rules
  • The video is displayed smoothly, with the highest possible quality

• *Basis for the relation between (sensor) input and (actuator) output*
Timeliness

- Derived constraints on controlling system (from functional behaviour) are determined by design choices in the system
  - “Not more that 5% of the paint is wasted”
    - opening the faucet must be precise enough w.r.t. speed of conveyor belt:
  - “Variations in temperature are within 0.05K”
    - response must be quick enough w.r.t. possible variations in environment and size and thermal isolation of system
  - “K items must be packed in each box”
    - response must be quick enough w.r.t. transport speed of items to pick one

- System design choices (decisions)
  - speed of conveyor belt (following productivity requirements)
  - size and thermal isolation
  - transport speed of items
Example: Water vessel

- **Requirements (functional behavior)**
  - Vessel should not overflow
  - Pump should not run dry

- **Properties**
  - **Pump**
    - states:
      - *off*: limited influx \( \Phi_{\text{in}} > 0 \) of water
      - *on*: limited outflux \( \Phi_{\text{out}} > 0 \) of water
    - operations: *start* and *stop*
  - **Sensors**
    - placed \( \Delta h \) from critical levels
    - readable, i.e. no trigger

- **Derived timing requirement**:
  - \( D_{\text{System}} \) time units needed for a level change \( \Delta h \)
    (for both pump *off* and pump *on*)
Water vessel: solution

- **System specification:** event or critical condition \( CC \)
  - water above (/below) water high (/low) sensor
  - upon occurrence, a *response* is required within \( D^{System} \) time

- **Design:** time-triggered (i.e.* polling*), periodic task \( \tau \):
  - period \( T^{\tau} \);
  - deadline \( D^{\tau} \);
  - actions:
    - read sensors (to detect \( CC \))
    - water above high sensor: \textit{start} pump
    - water below low sensor: \textit{stop} pump
Water vessel: solution

- Schedulability conditions:
  - general: $T^\tau + D^\tau \leq D_{\text{System}}$
  - for $D^\tau = T^\tau$: $2^*T^\tau \leq D_{\text{System}}$

- Note: independent of the scheduling algorithm!
Physical constraints

- Environment of the RTCS is given: the controlled system
  - though it may be part of the RTS design cycle
- Environment dictates part of the RTCS architecture
  - distribution, several distinct locations
  - concurrency
    - respond to simultaneous events
    - control concurrent physical processes
    - interconnect medium
  - embedding
    - limitations in resources: size, power, communication, memory
    - choice of processor or controller
    - software difficult to access
Dependability

- Availability: probability of the system being ready to use
- Reliability: expected time until not being available
- Safety: catastrophic states not reachable
- Security: protection of system assets
- Predictability: fixed and known system performance
  - ...under varying circumstances
- Maintainability: ease of repair, update etc.
Predictability

- Essential ingredient for real-time systems

- The predictability is endangered by
  - anomalies of the hardware and system software
    - use of DMA that locks the bus
    - cache behaviour
    - interrupts
    - memory management
  - constraints (dependencies between tasks)
    - concurrency control: locking of resource
    - precedence constraints
  - absence of transparency in high level languages
    - dynamic variables, garbage collection (memory management)
    - repetition, recursion