Hardware-Software Codesign

10. Performance Analysis of Distributed Embedded Systems

Lothar Thiele
System Design

- Specification
  - System Synthesis
    - Estimation
      - SW-Compilation
      - Instruction Set
      - HW-Synthesis
        - Intellectual Prop. Block
        - Machine Code
        - Net lists
      - Intellectual Prop. Code
Contents

- Overview
- Real-Time Calculus
- Modular Performance Analysis
- Examples
Formal Analysis vs. Simulation

- e.g. delay

Worst-Case
- Real System
- Simulation
- Formal analysis

- Best-Case

- upper bound
- lower bound
Analysis and Design

Embedded System = Computation + Communication + Resource Interaction

Analysis:
Infer system properties from subsystem properties.

Design:
Build a system from subsystems while meeting requirements.
Modular Performance Analysis

- Task graphs
- Application
- Mapping Scheduling
- System Model
- Performance Model
- Analysis
- Analysis Results

- Load Model (Environment)
- Service Model (Resources)
- Architecture
- Processing Model (HW/SW)

- Data sheets
- Measurements
- Formal specification
- Input traces
- architecture diagrams
- Component simulation
- Formal specification

System Model

Processing Model (HW/SW)
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream

Task

Processor

Load Model

Processing Model

Service Model
Modular System Composition

Diagram showing the composition of CPU, BUS, and DSP systems with connections labeled $\beta_{CPU}$, $\beta_{BUS}$, and $\beta_{DSP}$. Connections are made through GPC and TDMA modules.
Overview

System view

Modular Performance Analysis (MPA)

Real-Time Calculus (RTC)

Min-Plus Calculus, Max-Plus Calculus

Mathematical view
Contents

- Overview

- *Real-Time Calculus*

- Modular Performance Analysis

- Examples
Foundation

Real-Time Calculus can be regarded as a **worst-case/best-case variant of classical queuing theory**. It is a formal method for the analysis of distributed real-time embedded systems.

**Related Work:**


Comparison of Algebraic Structures

- **Algebraic structure**
  - set of elements \( S \)
  - one or more operators defined on elements of this set

- **Algebraic structures with two operators** \( \oplus, \odot \)
  - plus-times: \((S, \oplus, \odot) = (\mathbb{R}, +, \times)\)
  - min-plus: \((S, \oplus, \odot) = (\mathbb{R} \cup \{+\infty\}, \inf, +)\)

- **Infimum**:
  - The infimum of a subset of some set is the greatest element, not necessarily in the subset, that is less than or equal to all other elements of the subset.
  - \(\inf\{[3, 4]\} = 3, \quad \inf\{(3, 4]\} = 3\)
  - \(\min\{[3, 4]\} = 3, \quad \min\{(3, 4]\} \) not defined
Comparison of Algebraic Structures

- **Joint properties**: □
  - Closure of □: \( a \square b \in S \)
  - Associativity of □: \( a \square (b \square c) = (a \square b) \square c \)
  - Commutativity of □: \( a \square b = b \square a \)
  - Existence of identity element for □: \( \exists \nu : a \square \nu = a \)
  - Existence of negative element for □: \( \exists a^{-1} : a \square a^{-1} = \nu \)
  - Identity element of ⊕ absorbing for □: \( a \square \varepsilon = \varepsilon \)
  - Distributivity of □ w.r.t. ⊕: \( a \square (b \oplus c) = (a \square b) \oplus (a \square c) \)

- **Example**:
  - plus-times: \( a \times (b + c) = a \times b + a \times c \)
  - min-plus: \( a + \inf\{b, c\} = \inf\{a + b, a + c\} \)
Comparison of Algebraic Structures

- **Joint properties**: $\boxplus$
  
  Closure of $\boxplus$: $a \boxplus b \in S$
  
  Associativity of $\boxplus$: $a \boxplus (b \boxplus c) = (a \boxplus b) \boxplus c$
  
  Commutativity of $\boxplus$: $a \boxplus b = b \boxplus a$
  
  Existence of identity element for $\boxplus$: $\exists \varepsilon : a \boxplus \varepsilon = a$

- **Differences** $\boxplus$:
  
  - **plus-times**: Existence of a negative element for $\boxplus$:
    
    $\exists (-a) : a \boxplus (-a) = \varepsilon$
  
  - **min-plus**: Idempotency of $\boxplus$: $a \boxplus a = a$
Comparison of System Theories

- **Plus-times system theory**
  - signals, impulse response, convolution, time-domain

  \[ h(t) = (f \ast g)(t) = \int_{0}^{t} f(t - s) \cdot g(s) \, ds \]

- **Min-plus system theory**
  - streams, variability curves, time-interval domain, convolution

  \[ R'(t) \geq (R \otimes g)(t) = \inf_{0 \leq \lambda \leq t} \{ R(t - \lambda) + g(\lambda) \} \]
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream $R(t)$ Processor $C(t)$ Task $R'(t)$

Load Model $\alpha(\Delta)$ Service Model $\beta(\Delta)$ Processing Model
From Streams to Cumulative Functions

- **Data streams**: $R(t) = \text{number of events in } [0, t)$
- **Resource stream**: $C(t) = \text{available resource in } [0, t)$

![Graph showing R(t) and C(t)]
From Event Streams to Arrival Curves

Event Stream

number of events in in $t=[0 .. 2.5]$ ms

Arrival Curves $\alpha = [\alpha^l, \alpha^u]$

maximum / minimum arriving events in any interval of length 2.5 ms
From Resources to Service Curves

Resource Availability

available service in $t=[0 .. 2.5]$ ms

Service Curves $\beta = [\beta^l, \beta^u]$

maximum/minimum available service in *any interval* of length 2.5 ms
Example 1: Periodic with Jitter

A common event pattern that is used in literature can be specified by the parameter triple \((p, j, d)\), where \(p\) denotes the period, \(j\) the jitter, and \(d\) the minimum inter-arrival distance of events in the modeled stream.
Example 1: Periodic with Jitter

periodic

periodic with jitter
Example 1: Periodic with Jitter

Arrival curves:

\[ \alpha^l(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor \]

\[ \alpha^u(\Delta) = \min \left\{ \left\lfloor \frac{\Delta + j}{p} \right\rfloor, \left\lfloor \frac{\Delta}{d} \right\rfloor \right\} \]
Example 2: TDMA Resource

- Consider a real-time system consisting of \( n \) applications that are executed on a resource with bandwidth \( B \) that controls resource access using a TDMA policy.

- Analogously, we could consider a distributed system with \( n \) communicating nodes, that communicate via a shared bus with bandwidth \( B \), with a bus arbitrator that implements a TDMA policy.

- **TDMA policy**: In every TDMA cycle of length \( \bar{c} \), one single resource slot of length \( s_i \) is assigned to application \( i \).
Example 2: TDMA Resource

- **Service curves** available to the applications / node $i$:

\[
\beta_i^l(\Delta) = B \max \left\{ \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor (\bar{c} - s_i) \right\}
\]

\[
\beta_i^u(\Delta) = B \min \left\{ \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor (\bar{c} - s_i) \right\}
\]
Greedy Processing Component (GPC)

Examples:
- computation (event – task instance, resource – computing resource [tasks/second])
- communication (event – data packet, resource – bandwidth [packets/second])
Greedy Processing Component

Behavioral Description

- Component is triggered by incoming events.
- A fully preemptable task is instantiated at every event arrival to process the incoming event.
- Active tasks are processed in a greedy fashion in FIFO order.
- Processing is restricted by the availability of resources.
Greedy Processing Component (GPC)

If the resource and event streams describe available and requested units of processing or communication, then

\[ C(t) = C'(t) + R'(t) \]
\[ B(t) = R(t) - R'(t) \]

\[ R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \} \]

Conservation Laws
Greedy Processing

- For all times $u \leq t$ we have $R'(u) \leq R(u)$ (conservation law).
- We also have $R'(t) \leq R'(u)+C(t)–C(u)$ as the output can not be larger than the available resources.
- Combining both statements yields $R'(t) \leq R(u) + C(t) – C(u)$.
- Let us suppose that $u^*$ is the last time before $t$ with an empty buffer. We have $R(u^*) = R'(u^*)$ at $u^*$ and also $R'(t) = R'(u^*) + C(t) – C(u^*)$ as all available resources are used to produce output. Therefore, $R'(t) = R(u^*) + C(t) – C(u^*)$.
- As a result, we obtain

$$R'(t) = \inf_{0 \leq u \leq t} \{R(u) + C(t) – C(u)\}$$
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream → R(t) → Task → C(t) → R'(t)

Load Model

Processing Model

Service Model

\[ \alpha(\Delta) \]

\[ \beta(\Delta) \]
Abstraction

\[ R(t) \xrightarrow{\text{GPC}} C(t) \xrightarrow{\text{GPC}} C'(t) \xrightarrow{\text{GPC}} R'(t) \xrightarrow{\text{GPC}} \alpha(\Delta) \xrightarrow{\text{GPC}} \alpha'(\Delta) \]

- **time domain** cumulative functions
- **time-interval domain** variability curves
Some Definitions and Relations

- $f \otimes g$ is called **min-plus convolution**
  \[ (f \otimes g)(t) = \inf_{0 \leq u \leq t} \{ f(t - u) + g(u) \} \]

- $f \bowtie g$ is called **min-plus de-convolution**
  \[ (f \bowtie g)(t) = \sup_{u \geq 0} \{ f(t + u) - g(u) \} \]

For **max-plus convolution and de-convolution**:

- $(f \bar{\otimes} g)(t) = \sup_{0 \leq u \leq t} \{ f(t - u) + g(u) \}$
- $(f \bar{\bowtie} g)(t) = \inf_{u \geq 0} \{ f(t + u) - g(u) \}$

- Relation between convolution and deconvolution
  \[ f \leq g \otimes h \iff f \bowtie h \leq g \]
Arrival and Service Curve

The arrival and service curves provide bounds on event and resource functions as follows:

\[
\alpha^l(t-s) \leq R(t) - R(s) \leq \alpha^u(t-s) \quad \forall s \leq t
\]

\[
\beta^l(t-s) \leq C(t) - C(s) \leq \beta^u(t-s) \quad \forall s \leq t
\]

We can determine valid variability curves from cumulative functions as follows:

\[
\alpha^u = R \ominus R; \quad \alpha^l = R \overline{\ominus} R; \quad \beta^u = C \ominus C; \quad \beta^l = C \overline{\ominus} C
\]

One proof:

\[
\alpha^u = R \ominus R \Rightarrow \alpha^u(\Delta) = \sup_{u \geq 0} \{R(\Delta + u) - R(u)\} \Rightarrow
\]

\[
\alpha^u(\Delta) = \sup_{s \geq 0} \{R(\Delta + s) - R(s)\} \Rightarrow \alpha^u(t-s) \geq R(t) - R(s) \quad \forall t \geq s
\]
Abstraction

\[ C'(t) \]
\[ R(t) \rightarrow \text{GPC} \rightarrow R'(t) \rightarrow \alpha(\Delta) \rightarrow \text{GPC} \rightarrow \alpha'(\Delta) \]
\[ C''(t) \]

time domain cumulative functions

time-interval domain variability curves
The Most Simple Relations

- The output stream of a component satisfies:
  \[ R'(t) \geq (R \circ \beta^l)(t) \]

- The output upper arrival curve of a component satisfies:
  \[ \alpha^{u'} = (\alpha^u \circ \beta^l) \]

- The remaining lower service curve of a component satisfies:
  \[ \beta^{l'}(\Delta) = \sup_{0 \leq \lambda \leq \Delta} (\beta^l(\lambda) - \alpha^u(\lambda)) \]
Two Sample Proofs

$$R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \}$$

$$\geq \inf_{0 \leq u \leq t} \{ R(u) + \beta^l(t - u) \}$$

$$= (R \otimes \beta^l)(t)$$

$$C'(t) - C'(s) = \sup_{0 \leq a \leq t} \{ C(a) - R(a) \} - \sup_{0 \leq b \leq s} \{ C(b) - R(b) \} =$$

$$= \inf_{0 \leq b \leq s} \{ \sup_{0 \leq a \leq t} \{ (C(a) - C(b)) - (R(a) - R(b)) \} \}$$

$$= \inf_{0 \leq b \leq s} \{ \sup_{0 \leq a - b \leq t - b} \{ (C(a) - C(b)) - (R(a) - R(b)) \} \}$$

$$\geq \inf_{0 \leq b \leq s} \{ \sup_{0 \leq \lambda \leq t - b} \{ \beta^l(\lambda) - \alpha^u(\lambda) \} \} \geq \sup_{0 \leq \lambda \leq t - s} \{ \beta^l(\lambda) - \alpha^u(\lambda) \}$$
Tighter Bounds

The greedy processing component transforms the variability curves as follows:

\[
\alpha^{u'} = [(\alpha^u \otimes \beta^u) \otimes \beta^l] \land \beta^u \\
\alpha^l' = [(\alpha^l \otimes \beta^u) \otimes \beta^l] \land \beta^l \\
\beta^{u'} = (\beta^u - \alpha^l) \ominus 0 \\
\beta^l' = (\beta^l - \alpha^u) \ominus 0
\]

Without proof … .
Delay and Backlog

\[ B = \sup_{t \geq 0} \{ R(t) - R'(t) \} \leq \sup_{\lambda \geq 0} \{ \alpha^u(\lambda) - \beta^l(\lambda) \} \]

\[ D = \sup_{t \geq 0} \{ \inf \{ \tau \geq 0 : R(t) \leq R'(t + \tau) \} \} \]

\[ = \sup_{\Delta \geq 0} \{ \inf \{ \tau \geq 0 : \alpha^u(\Delta) \leq \beta^l(\Delta + \tau) \} \} \]
Proof of Backlog Bound

\[ B(t) = R(t) - R'(t) = R(t) - \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \} \]

\[ = \sup_{0 \leq u \leq t} \{ (R(t) - R(u)) - (C(t) - C(u)) \} \]

\[ \leq \sup_{0 \leq u \leq t} \{ \alpha^u(t-u) - \beta^l(t-u) \} \]

\[ \leq \sup_{0 \leq \lambda} \{ \alpha^u(\lambda) - \beta^l(\lambda) \} \]
Contents

- Overview
- Real-Time Calculus
- Modular Performance Analysis
- Examples
System Composition

How to interconnect service?

Scheduling!

\[ \beta_{CPU} \]

\[ \beta_{BUS} \]

\[ \beta_{DSP} \]
Scheduling and Arbitration

- FP/RM
  - $\beta$
  - $\alpha_A \rightarrow \alpha'_A$
  - $\alpha_B \rightarrow \alpha'_B$

- GPC

- GPS
  - $\beta'$

- EDF
  - $\beta$
  - $\alpha_A \rightarrow \alpha'_A$
  - $\alpha_B \rightarrow \alpha'_B$

- TDMA
  - $\beta'$

- RR
  - $\beta$
  - $\alpha_A \rightarrow \alpha'_A$
  - $\alpha_B \rightarrow \alpha'_B$

- GPC

- share
  - $\alpha_A \rightarrow \alpha'_A$
  - $\alpha_B \rightarrow \alpha'_B$

- sum
  - $\beta'$

- TDMA

- GPC
  - $\beta'_{s1}$ $\beta'_{s2}$
Complete System Composition

\[ \beta_{CPU} \]
\[ \alpha \]
\[ \beta_{BUS} \]
\[ \beta_{DSP} \]

\[ \alpha' \]
Extending the Framework

- New HW behavior
- New SW behavior
- New scheduling scheme
- ...

Find new relations:

\[ \alpha'(\Delta) = f_{\alpha}(\alpha, \beta) \]
\[ \beta'(\Delta) = f_{\beta}(\alpha, \beta) \]

This is the hard part…!
Contents

- Overview

- Real-Time Calculus

- Modular Performance Analysis

- Examples
Case Study

6 Real-Time Input Streams
- with jitter
- with bursts
- deadline > period

3 ECU’s with own CC’s

13 Tasks & 7 Messages
- with different WCED

2 Scheduling Policies
- Earliest Deadline First (ECU’s)
- Fixed Priority (ECU’s & CC’s)

Hierarchical Scheduling
- Static & Dynamic Polling Servers

Bus with TDMA
- 4 time slots with different lengths
  (#1,#3 for CC1, #2 for CC3, #4 for CC3)

Total Utilization:
- ECU1 59 %
- ECU2 87 %
- ECU3 67 %
- BUS 56 %
## Specification Data

<table>
<thead>
<tr>
<th>Stream</th>
<th>(p,j,d) [ms]</th>
<th>D [s]</th>
<th>Task Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>(1000, 2000, 25)</td>
<td>8.0</td>
<td>T1.1 → C1.1 → T1.2 → C1.2 → T1.3</td>
</tr>
<tr>
<td>S2</td>
<td>(400, 1500, 50)</td>
<td>1.8</td>
<td>T2.1 → C2.1 → T2.2</td>
</tr>
<tr>
<td>S3</td>
<td>(600, 0, -)</td>
<td>6.0</td>
<td>T3.1 → C3.1 → T3.2 → C3.2 → T3.3</td>
</tr>
<tr>
<td>S4</td>
<td>(20, 5, -)</td>
<td>0.5</td>
<td>T4.1 → C4.1 → T4.2</td>
</tr>
<tr>
<td>S5</td>
<td>(30, 0, -)</td>
<td>0.7</td>
<td>T4.1 → C4.1 → T4.2</td>
</tr>
<tr>
<td>S6</td>
<td>(1500, 4000, 100)</td>
<td>3.0</td>
<td>T6.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1.1</td>
<td>200</td>
</tr>
<tr>
<td>T1.2</td>
<td>300</td>
</tr>
<tr>
<td>T1.3</td>
<td>30</td>
</tr>
<tr>
<td>T2.1</td>
<td>75</td>
</tr>
<tr>
<td>T2.2</td>
<td>25</td>
</tr>
<tr>
<td>T3.1</td>
<td>60</td>
</tr>
<tr>
<td>T3.2</td>
<td>60</td>
</tr>
<tr>
<td>T3.3</td>
<td>40</td>
</tr>
<tr>
<td>T4.1</td>
<td>12</td>
</tr>
<tr>
<td>T4.2</td>
<td>2</td>
</tr>
<tr>
<td>T5.1</td>
<td>8</td>
</tr>
<tr>
<td>T5.2</td>
<td>3</td>
</tr>
<tr>
<td>T6.1</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1.1</td>
<td>100</td>
</tr>
<tr>
<td>C1.2</td>
<td>80</td>
</tr>
<tr>
<td>C2.1</td>
<td>40</td>
</tr>
<tr>
<td>C3.1</td>
<td>25</td>
</tr>
<tr>
<td>C3.2</td>
<td>10</td>
</tr>
<tr>
<td>C4.1</td>
<td>3</td>
</tr>
<tr>
<td>C5.1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Periodic Server</th>
<th>p</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPS\textsubscript{ECU1}</td>
<td>500</td>
<td>200</td>
</tr>
<tr>
<td>SPS\textsubscript{ECU3}</td>
<td>500</td>
<td>250</td>
</tr>
<tr>
<td>DPS\textsubscript{ECU3}</td>
<td>600</td>
<td>120</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TDMA</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle</td>
<td>100</td>
</tr>
<tr>
<td>Slot\textsubscript{CC1a}</td>
<td>20</td>
</tr>
<tr>
<td>Slot\textsubscript{CC1b}</td>
<td>25</td>
</tr>
<tr>
<td>Slot\textsubscript{CC2}</td>
<td>25</td>
</tr>
<tr>
<td>Slot\textsubscript{CC3}</td>
<td>30</td>
</tr>
</tbody>
</table>
The Distributed Embedded System...
... and its MPA Model
Buffer & Delay Guarantees
Available & Remaining Service of ECU1
Input of Stream 3
Output of Stream 3
Automated Design Space Exploration

We use evolutionary algorithms for multi-objective optimization!
Network Processor Task Model

Flow NRT Encrypt

Flow NRT Decrypt

Encryption/Decryption

Voice Processing

Flow NRT IP Forward

Link Rx
Verify IP Header
Process IP Header
Classify
Route Look Up
IP header Modify
Calc Check Sum
ARP Look Up
Schedule
Link Tx

Flow RT Send

Voice Encoder
RTP Tx
UDP Tx
Build IP Header
Route Look Up
UDP Rx
RTP Rx
Dejitter
Voice Decoder

Swiss Federal
Institute of Technology

Computer Engineering
and Networks Laboratory
EXPO

Implementation Nr. 60541 (EXPO, Institute TIK, ETH Zurich)

Scenario: Scen2

Optimal Scaling Factor: 0.530

Total Memory: 8.295

- DSP
  Utilization: 79%

- CheckSum
  Utilization: 4%

- LookUp
  Utilization: 7%

Flow: RTSend
  Priority: 5
  Acc. Waiting Time in Queue: 0.000
  RTPtx
  VoiceEnc
  LinkTx
  Schedule
  Decrypt
  AHVerify
  Classify
  LinkRx

Flow: NRTDecrypt
  Priority: 4
  Acc. Waiting Time in Queue: 0.000
  ESPDecrypt
  ProcessIP
  IPModify
  LinkTx
  Schedule
  Decrypt
  Classify
  LinkRx

Flow: RTRcv
  Priority: 1
  Acc. Waiting Time in Queue: 0.000
  Dejitter
  VoiceDec
  ProcessIP
  RTPx
  Classify
  LinkRx

Flow: NRTForward
  Priority: 3
  Acc. Waiting Time in Queue: 23.068
  ProcessIP
  VerifyIP
  UDPx
Results

Performance for encryption/decryption

Performance for RT voice processing

<table>
<thead>
<tr>
<th>Task</th>
<th>NRT</th>
<th>RT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP</td>
<td>64%</td>
<td>39%</td>
</tr>
<tr>
<td>Cipher</td>
<td>71%</td>
<td>0%</td>
</tr>
<tr>
<td>LookUp</td>
<td>15%</td>
<td>6%</td>
</tr>
<tr>
<td>Classifier</td>
<td>27%</td>
<td>11%</td>
</tr>
<tr>
<td>DSP</td>
<td>35%</td>
<td>39%</td>
</tr>
<tr>
<td>LookUp</td>
<td>1%</td>
<td>6%</td>
</tr>
<tr>
<td>Classifier</td>
<td>1%</td>
<td>11%</td>
</tr>
</tbody>
</table>
Analysis vs. Simulation

[Graph showing comparison between simulation and analytical method for various loads and packet sizes.]
Design Space Exploration

- Determine mapping
- Determine performance network
- Solve system of equations
- Determine important parameters (end-to-end delay, throughput, buffer space output jitter, ...)
- Give feedback to optimization
RTCA Toolbox

Modular Performance Analysis with Real-Time Calculus

RTC Toolbox :: Overview

<table>
<thead>
<tr>
<th>Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTC Toolbox</td>
</tr>
<tr>
<td>PESIMDES</td>
</tr>
</tbody>
</table>

Real-Time Calculus Toolbox

Overview
The Real-Time Calculus (RTC) Toolbox is a free Matlab toolbox for system-level performance analysis of distributed real-time and embedded systems.

Latest News
[2006-10-02]: New tutorials and Java API released.
[2006-10-02]: BugFix released.
[2006-04-04]: First tutorial published.

Citation Information
If you use the RTC Toolbox for research purposes, we would be happy to hear about it and mention it in the manual. Please drop us a line at rtc@tik.ee.ethz.ch.

BibTeX entry for citation:

```bibtex
@misc{rtc,  
author = {Ernesto Wandelux and Lothar Thiele},  
title = {Real-Time Calculus (RTC) Toolbox},  
url = {http://www.mpa.ethz.ch/rtctoolbox},  
howpublished = {via \tt http://www.mpa.ethz.ch/rtctoolbox},  
year = {2006},  }
```

© 2006 Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland

www.mpa.ethz.ch/rtctoolbox