WHAT THE HECK IS A DEADLINE…

OR:

INDUSTRIAL REQUIREMENTS & SOLUTIONS FOR A SUITABLE INTERFACE BETWEEN FUNCTION DEVELOPMENT AND REAL-TIME SYSTEMS INTEGRATION

DIRK ZIEGENBEIN, ARNE HAMANN, ECKART MAYER-JOHN CORPORATE RESEARCH, ROBERT BOSCH GMBH
A Suitable Interface for Real-Time Control

Two Disciplines – Two Worlds

Control Engineer

Real-Time Systems Engineer

Too little communication and too little understanding
A Suitable Interface for Real-Time Control
System as seen by the control engineer

Sophisticated Control Algorithm

Plant
\[ \dot{x} = Ax + Bu \]
\[ y = C^T x \]

Periodic Sampling, No Jitter

No/Constant Control Delay
Calculation takes 0/constant time

No Output Jitter, “Freshest” value always available

A/D Converter

y(t)

D/A Converter

u(t)

Write Data

Read Data

"Control performance"
A Suitable Interface for Real-Time Control System as seen by the real-time systems engineer

ECU
- Tasks
- Scheduling
- Cores, Memories

**Deadline = Period WCET, WCRT**

\[ R_i = C_i + \sum_{j \in hp(i)} C_j \left[ \frac{R_i}{T_j} \right] \leq D_i = T_i \]

\[ \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n \cdot \left( \sqrt{2} - 1 \right) \ln 2 \approx 69.3\% \]

Real-Time Systems Engineer

"Real-time performance"
A Suitable Interface for Real-Time Control
Two Worlds – One Goal

Mathematical control model

How to get there efficiently & correct?

Implementation on HW/SW platform

Environment time (“reactivity”)

Concurrency

Distribution

Platform time (“schedulability”)

Inspired by Tom Henzinger

Ziegenbein, Hamann, Mayer-John | 10/15/2017

© Robert Bosch GmbH 2016. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
A Suitable Interface for Real-Time Control

Requirements for a suitable interface

- Understandable, match problem domain
- Reusable = composable + portable
- Efficiently implementable

Mathematical control model

Correct function

Suitable Interface

Correct implementation

Implementation on HW/SW platform

Environment time ("reactivity")

Concurrency

Requirements:

Platform time ("schedulability")

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control

Current Interface: Task Priorities & Deadlines

- Understandable?
- Reusable?
- Efficient? YES

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?

▶ It is not a single task that matters...

...but communicating tasks.
A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?

The cause-effect chains spanning several tasks matter.

Simplified software structure of an combustion engine control system (out of „Benchmarking, System Design and Case-studies for Multi-core based Embedded Automotive Systems“ by Piotr Dziurzanski, Amit Kumar Singh, Leandro S. Indrusiak, Björn Saballus)
**A Suitable Interface for Real-Time Control**

**Task Priorities & Deadlines – Understandable?**

- Priorities on a single-core lead to an "implicit" developer’s fiction
  - Fast to slow communication “immediate”

- Task-core-mapping influences functional behavior
  - Implicit assumption “immediate” is broken
  - Major pain point for multi-core migration
Additional Problem: Data inconsistency

- Single core: Legacy code contains implicit assumptions about priorities and thus execution sequences
- Multi-core: These assumptions often break the functionalities and require lots of debugging of race conditions

→ Need for data consistency
Implicit communication to achieve data consistency

- **Explicit communication**
  - No regulations in place, each function directly reads and writes labels
  - Possible races are handled using locks by the developers

- **Implicit communication**
  - Local copies are created for each read label at the beginning of the task
  - All computations work on the local copies
  - The local copies are written back to the shared memory at the end of the task
  - Result: data consistency on task level: all functions operate on the same data set
Task Priorities & Deadlines – Understandable?

- Jitter & non-determinism in latency of cause effect chains does not map well to control theory

- Iterative simulation-based work flow to consider real-world timing in control engineering

A Suitable Interface for Real-Time Control

Current Abstraction: Task Priorities & Deadlines

Mathematical control model

Correctness?

Task Priorities & Deadlines

Efficiency!

Implementation on HW/SW platform

- Understandable? NO
- Reusable?
- Efficient? YES

Inspired by Tom Henzinger

Ziegenbein, Hamann, Mayer-John | 10/15/2017
A Suitable Interface for Real-Time Control

Reusable = Composable + Portable

- **Composable**
  - Locality of changes
  - Key for development efficiency

- **Portability**
  - Program can be mapped to different platforms
  - Key for validity of SiL/HiL tests
  - Following from this: interface needs to talk only about environment time
A Suitable Interface for Real-Time Control

Task Priorities & Deadlines – Reusable?

- Not composable
  - Example: Increase in execution time of T1 increases latency of cause effect chain

- Not portable
  - Latency depends on Task-core mapping

---

Ziegenbein, Hamann, Mayer-John | 10/15/2017
© Robert Bosch GmbH 2016. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
A Suitable Interface for Real-Time Control

Current Interface: Task Priorities & Deadlines

- Understandable? NO
- Reusable? NO
- Efficient? YES

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control
Solution: Logical Execution Time

Mathematical control model

Correct function

Logical Execution Time (LET)

Correct implementation

- Understandable?
- Reusable?
- Efficient?

Implementation on HW/SW platform

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control

LET – Modeling Principle

Control engineer’s fiction: “platform is fast enough to compute the task in d”

Inspired by Tom Henzinger

Ziegenbein, Hamann, Mayer-John | 10/15/2017

© Robert Bosch GmbH 2016. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Read input at time t

d is the task’s logical execution time

Write output at time t+d

Software Task
A Suitable Interface for Real-Time Control

LET – Implementation Principle

Control engineer’s fiction: “platform is fast enough to compute the task in d”
A Suitable Interface for Real-Time Control

**LET – Portability**

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control

LET – Composability & Determinism

Composability
(no functional effect of resource sharing)

Internal Determinism
• no data races
• fixed end-to-end timing for cause effect chains

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control

Solution: Logical Execution Time

- Understandable? YES
- Reusable? YES
- Efficient?

Typical Doubts:
- Resource penalty
- Latency penalty

Mathematical control model

Correct function

Logical Execution Time (LET)

Correct implementation

Implementation on HW/SW platform

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control
“Timed Communication”: Tailored LET Solution @ Bosch

- Focused on Internal Determinism
  - Fixed (but increased) end-to-end latencies of cause and effect chains
  - Input data consistency of tasks with same LET interval start
- Logical execution time equals task period \(\rightarrow\) enables efficient implementation
  - Average values from Engine Control Systems on Infineon Aurix: +0.5% RAM, -2% CPU utilization
A Suitable Interface for Real-Time Control

LET – Latency Penalty Acceptable?

Example: Control of a Permanent Magnet Synchronous Motor

\[
\begin{bmatrix}
I_d \\
\dot{I}_q
\end{bmatrix}
=\begin{bmatrix}
-\frac{R}{L_d} & \frac{L_q(\omega_c)}{L_d} \\
-\frac{L_q(\omega_c)}{L_q} & -\frac{R}{L_q}
\end{bmatrix}
\begin{bmatrix}
I_d \\
I_q
\end{bmatrix}
+\begin{bmatrix}
\frac{1}{L_d} & 0 \\
0 & \frac{1}{L_q}
\end{bmatrix}
\begin{bmatrix}
V_d \\
V_q
\end{bmatrix}
\]

\[
=: A
\]

\[
=: B
\]

25

Ziegenbein, Hamann, Mayer-John | 10/15/2017
© Robert Bosch GmbH 2016. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
A Suitable Interface for Real-Time Control

Simulation: 0.5-step Delay
A Suitable Interface for Real-Time Control

Simulation: 1-step Delay
A Suitable Interface for Real-Time Control

Simulation: 2-step Delay
A Suitable Interface for Real-Time Control

Simulation: 2-step Delay with 2-step Ahead Prediction

- Current sensor
- Position sensor
- Thermal sensor
- Controller
- PWM delay

900 μs 1000 μs 1100 μs

200 μs end-to-end delay

Id ref
Id actual with prediction
Id actual without prediction

Iq ref
Iq actual with prediction
Iq actual without prediction
A Suitable Interface for Real-Time Control

Solution: Logical Execution Time

- Understandable? YES
- Reusable? YES
- Efficient? YES

Typical Doubts:
- Resource penalty
- Latency penalty

Mathematical control model

Correct function

Logical Execution Time (LET)

Correct implementation

Implementation on HW/SW platform

Inspired by Tom Henzinger
A Suitable Interface for Real-Time Control

Life with LET

- Design control functions with standard blocks
- Integrate several control functions based on LET time structures

Logical Execution Time (LET)

- Map control functions to SW tasks and platforms
- Validate that tasks respect their LET intervals
  - E.g. based on AMALTHEA system models
- Some more details at:

https://www.eclipse.org/app4mc/
Unfortunately, for many automotive systems
- Worst case response times >> average case response times
  - Due to average-case optimized HW platforms (and legacy SW structure)
- Sporadic overload and missed deadlines are often tolerable
  - Functional impact negligible
  - Due to robust design (oversampling)

Consequences
- Pragmatic timing validation accepting “occasional” timing violations
- Established exception handling (e.g. use last instance value)
- Despite matching characteristics, little traction for more formal approaches like (m out of k) so far
A Suitable Interface for Real-Time Control
The White Knight: Electrical Powertrains

- Large efficiency gains possible by advanced control approaches
- Higher dynamics – shorter time constants
  - Periods of control functions significantly shorter (in 100µs range)
  - Current degree of oversampling not affordable in terms of computing power
  - High sensitivity of efficiency to deadline misses
- High interest in (m out of k) control theory and timing verification
Further Upcoming Challenge: Heterogeneity

- Extend the performance analysis methods and tooling to deal with
  - Heterogenous HW platforms
  - SW isolation mechanisms (e.g. hypervisors)
  - ...

- Develop constructive technologies enabling composability & portability for heterogeneous applications
  - Work-preserving freedom-from-interference
  - Abstractions for online performance-awareness

Funded by German Federal Ministry of Education and Research (BMBF)
THANK YOU

...AND THANKS TO LINDA WIJAYA JONG, DAKSHINA DASARI, JENS GLADIGAU, SIMON KRAMER, TOBIAS BEICHTER, ...