INDUSTRIAL REQUIREMENTS & SOLUTIONS FOR

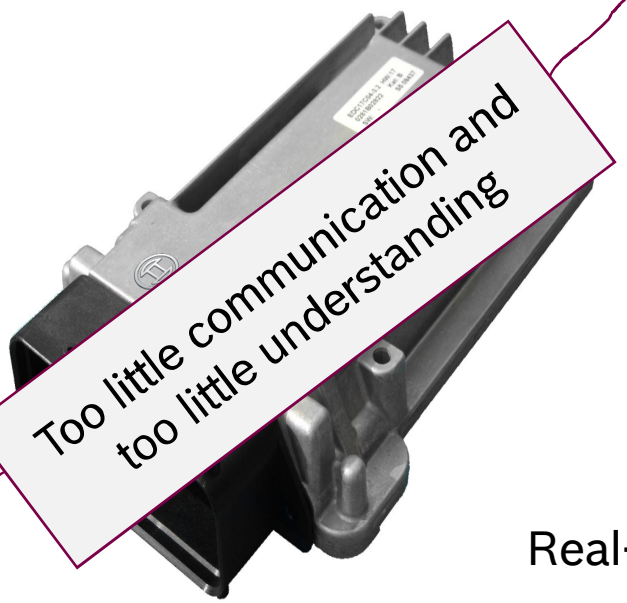
A SUITABLE INTERFACE BETWEEN FUNCTION DEVELOPMENT AND REAL-TIME SYSTEMS INTEGRATION

DIRK ZIEGENBEIN, ARNE HAMANN, ECKART MAYER-JOHN
CORPORATE RESEARCH, ROBERT BOSCH GMBH

BOSCH

A Suitable Interface for Real-Time Control
Two Disciplines – Two Worlds

Control Engineer

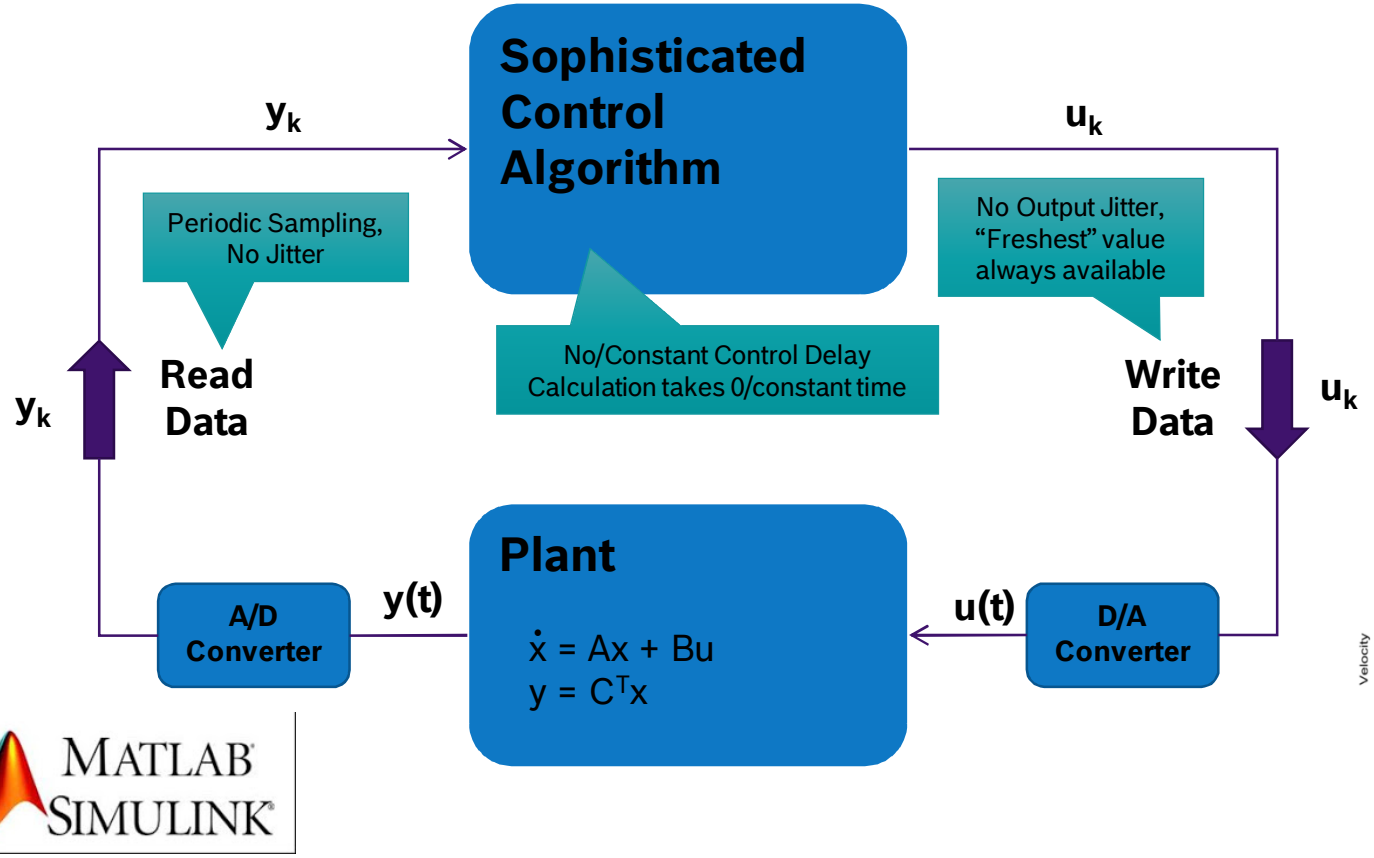


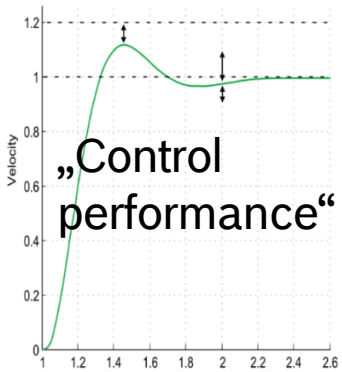
Real-Time Systems Engineer

A Suitable Interface for Real-Time Control

System as seen by the control engineer

Control Engineer





A Suitable Interface for Real-Time Control

System as seen by the real-time systems engineer

Real-Time Systems Engineer

Deadline = Period
WCET, WCRT

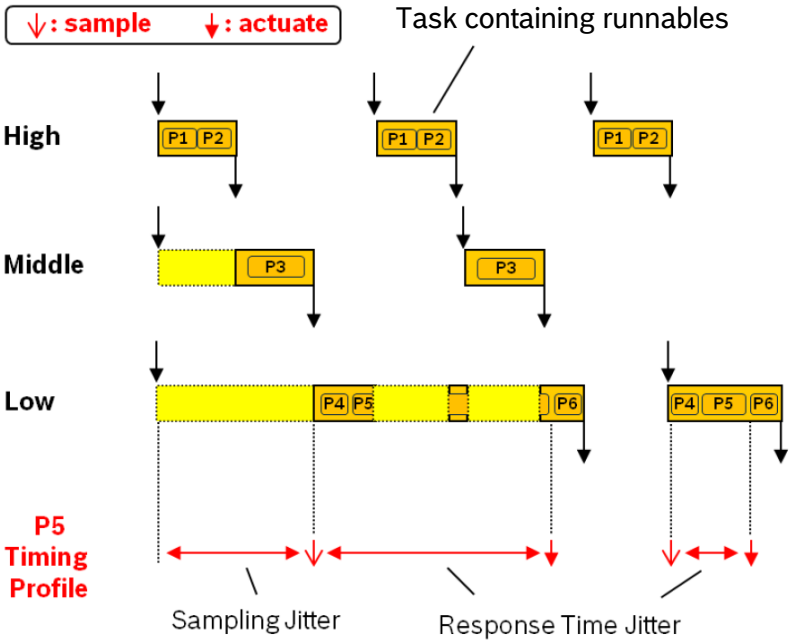
ECU

- Tasks
- Scheduling
- Cores, Memories

$$R_i = C_i + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{R_i}{T_j} \right\rceil \leq D_i = T_i$$

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

$$\ln 2 \approx 69{,}3\%$$



„Real-time performance“

BOSCH

A Suitable Interface for Real-Time Control

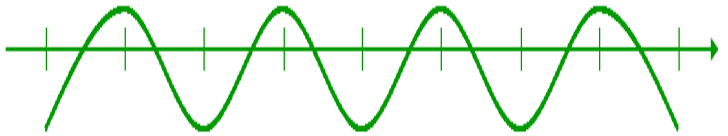
Two Worlds – One Goal

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

BOSCH

A Suitable Interface for Real-Time Control

Requirements for a suitable interface

Mathematical control model

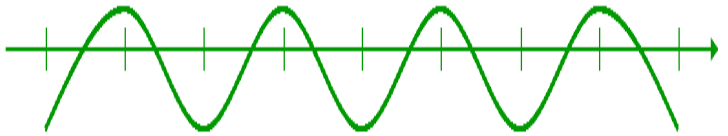
Correct function

Suitable Interface

Correct implementation

Implementation on HW/SW platform

Environment time (“reactivity”)



Concurrency

Requirements:

- **understandable, match problem domain**
- **reusable = composable + portable**
- **efficiently implementable**

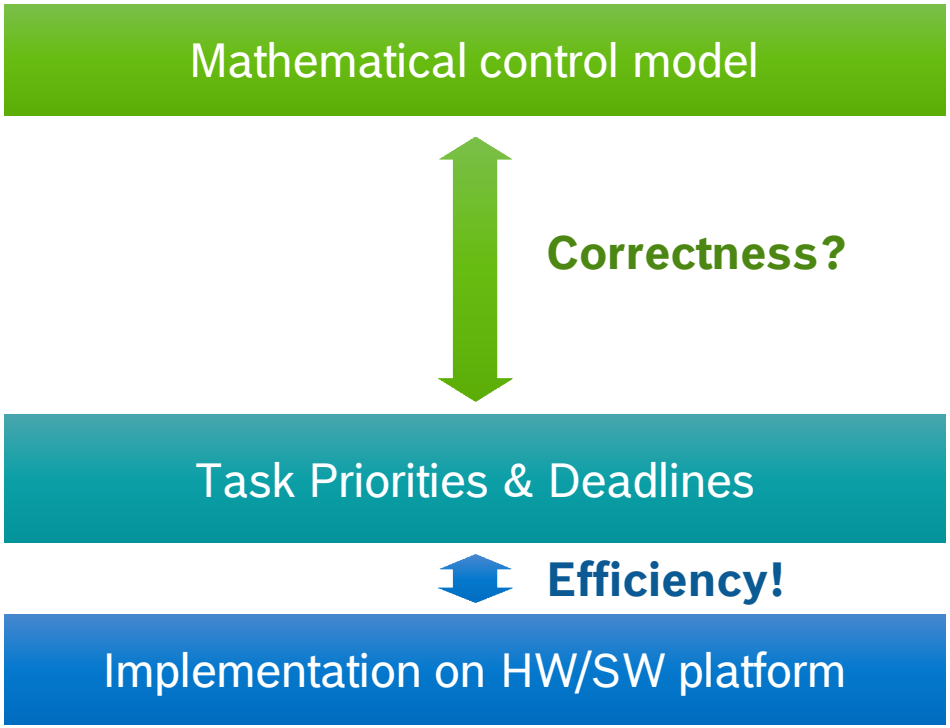
Distribution

Platform time (“schedulability”)

Inspired by Tom Henzinger

BOSCH

A Suitable Interface for Real-Time Control
Current Interface: Task Priorities & Deadlines



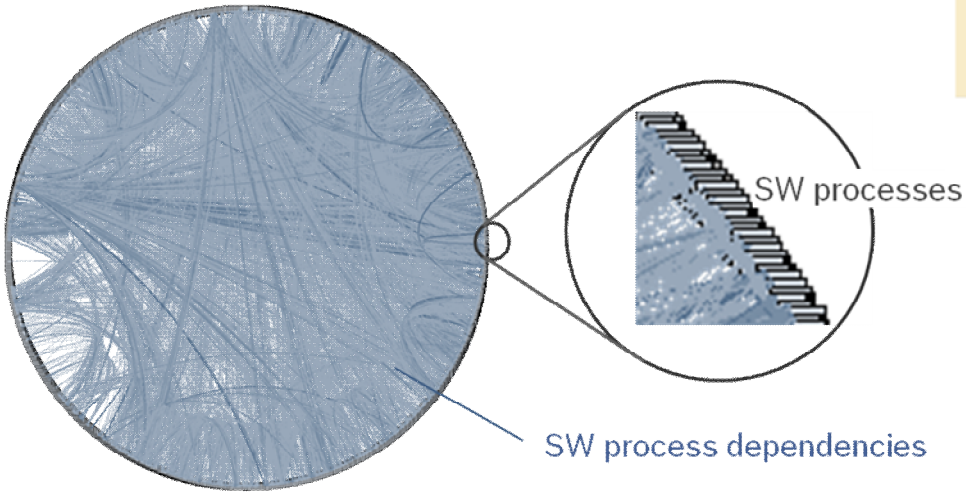
- **Understandable?**
- **Reusable?**
- **Efficient? YES**

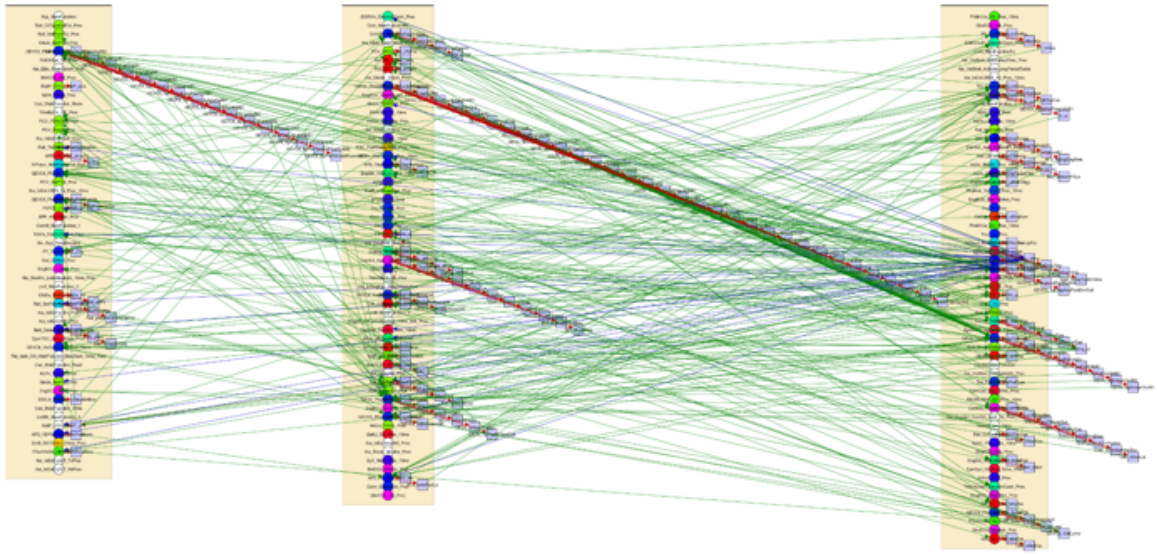
Inspired by Tom Henzinger

A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?

- **It is not a single task that matters…**

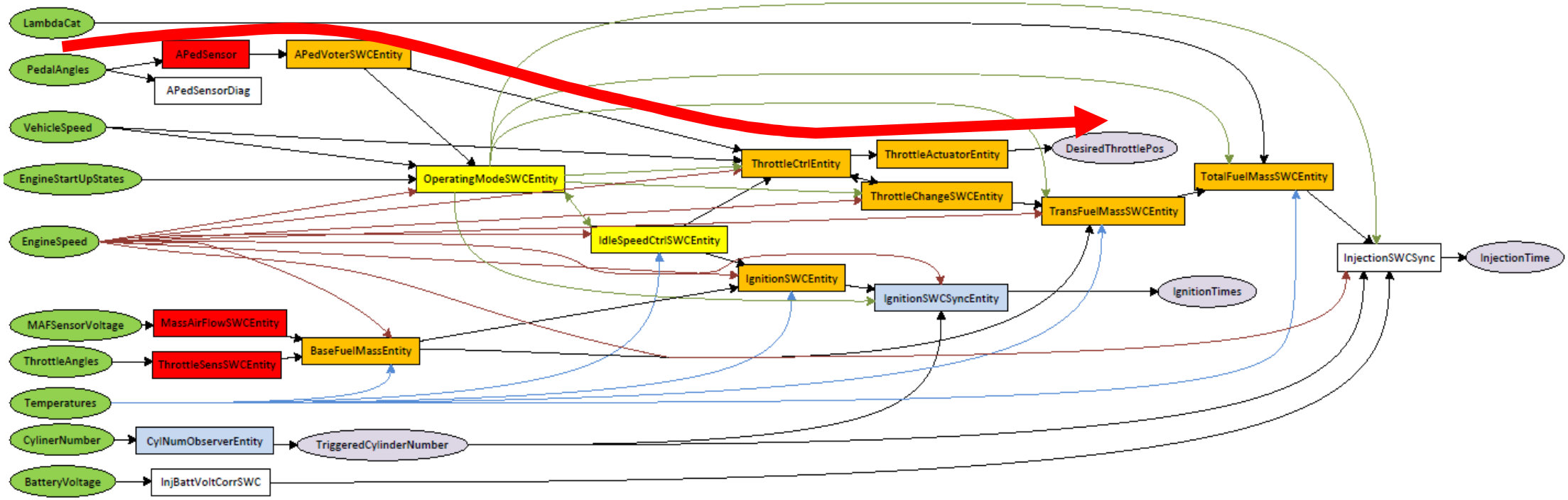
  …but communicating tasks.





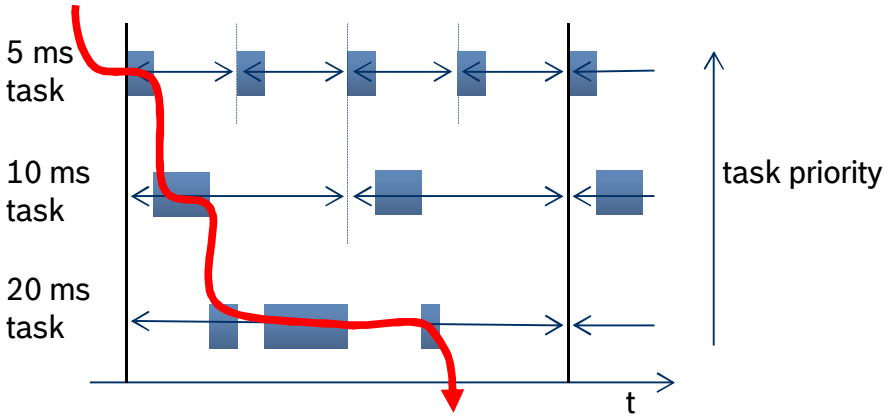
A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?

- **The cause-effect chains spanning several tasks matter.**

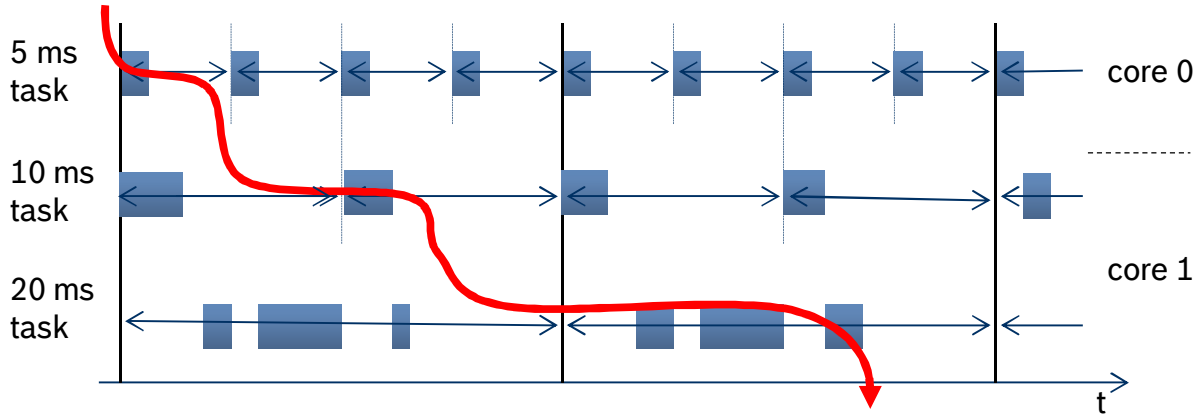


Simplified software structure of an combustion engine control system (out of „Benchmarking, System Design and Case-studies for Multi-core based Embedded Automotive Systems“ by Piotr Dziurzanski, Amit Kumar Singh, Leandro S. Indrusiak, Björn Saballus)

A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?



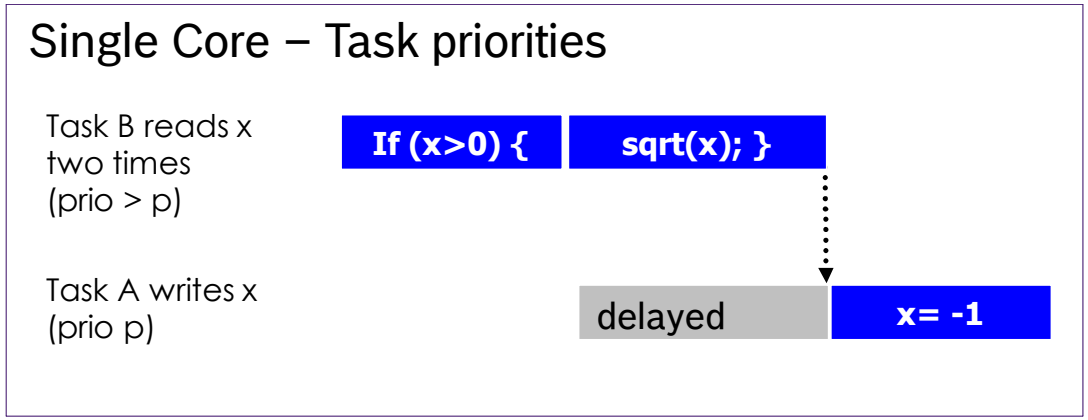
Single-Core

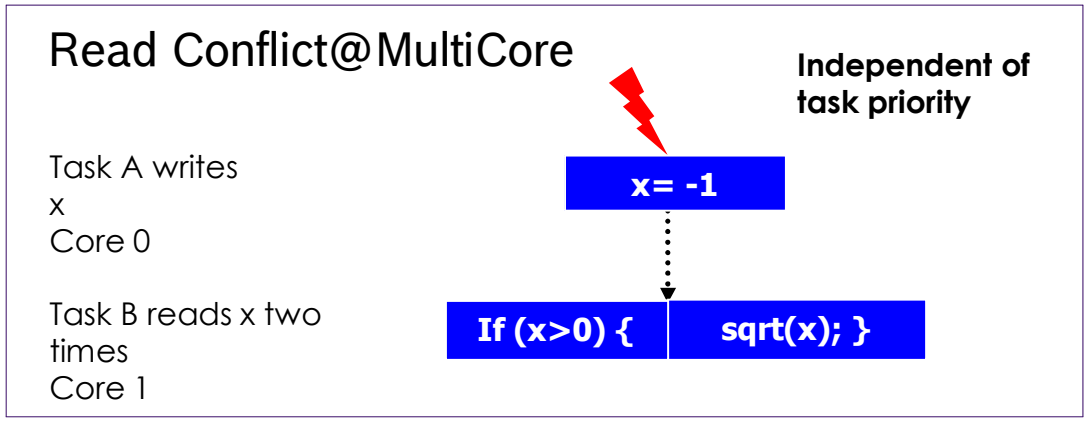


Possible Multi-Core Scenario

- 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

A Suitable Interface for Real-Time Control
Additional Problem: Data inconsistency

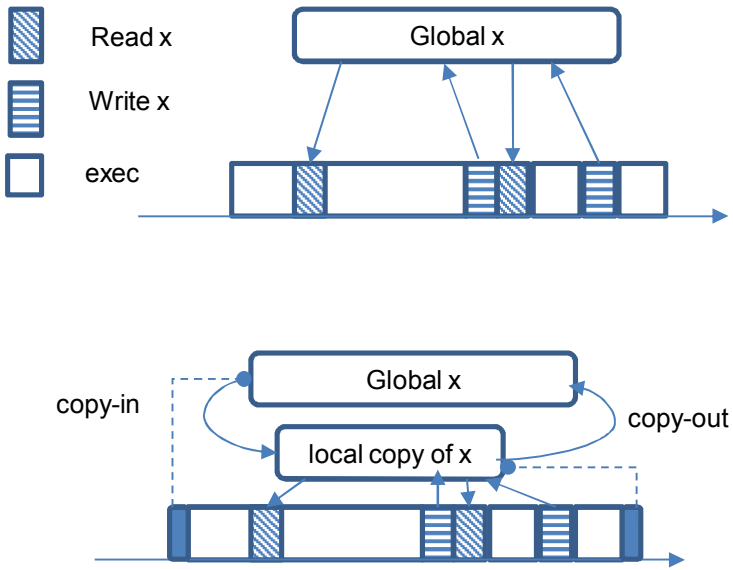




- Single core: Legacy code contains implicit assumptions about priorities and thus execution sequences
- Multi-core: These assumptions often break the functionalities and require lots of debugging of race conditions

→ Need for data consistency

A Suitable Interface for Real-Time Control
Implicit communication to achieve data consistency



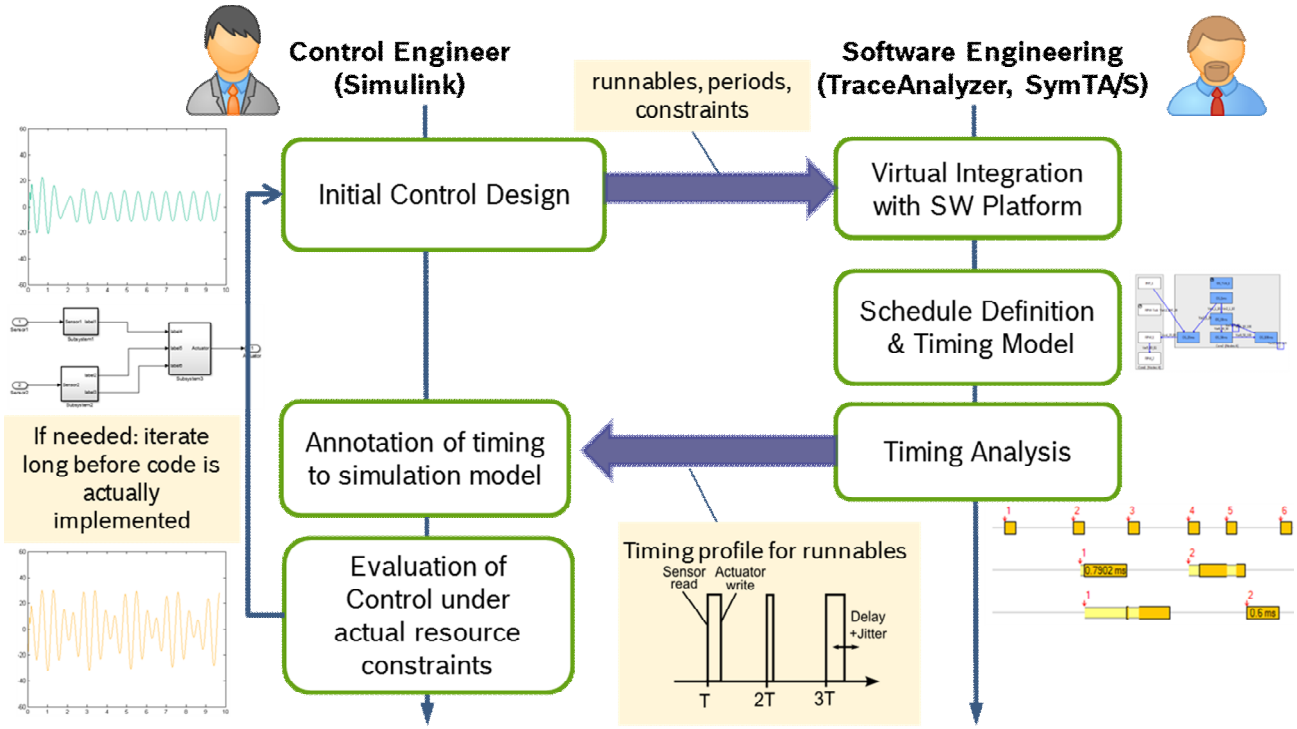
- Explicit communication
  - No regulations in place, each function directly reads and writes labels
  - Possible races are handled using locks by the developers
- Implicit communication
  - Local copies are created for each read label at the beginning of the task
  - All computations work on the local 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

A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Understandable?

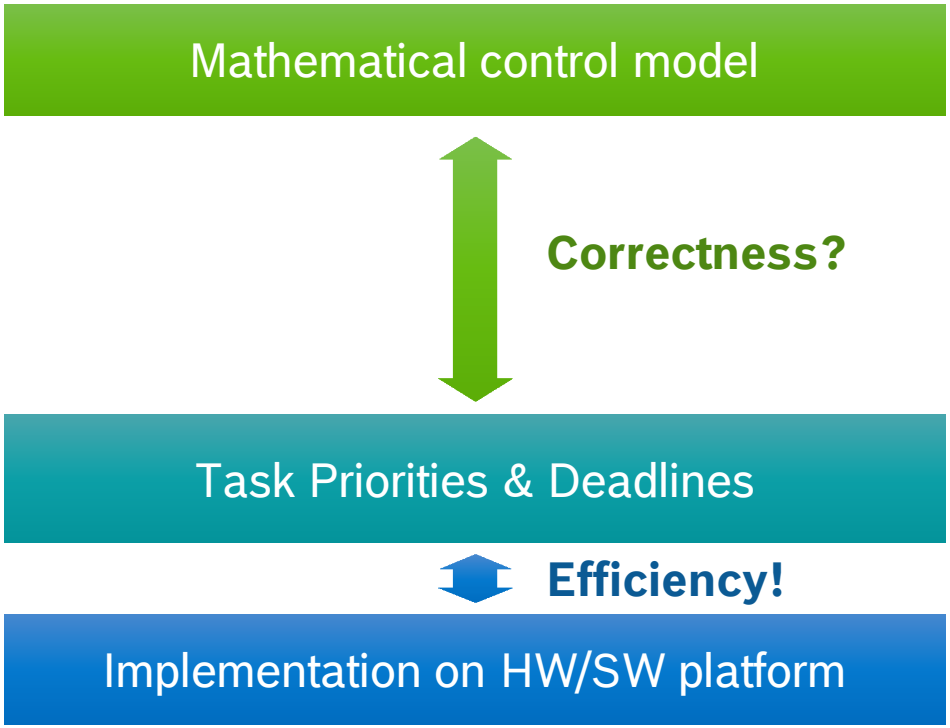
- **Jitter & non-determinism in latency of cause effect chains does not map well to control theory**

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

- More details:
  Lampke et al.,"Model-Based Co-Engineering of Control Algorithms & Real-Time Systems," SAE World Congress 2015



A Suitable Interface for Real-Time Control
Current Abstraction: Task Priorities & Deadlines

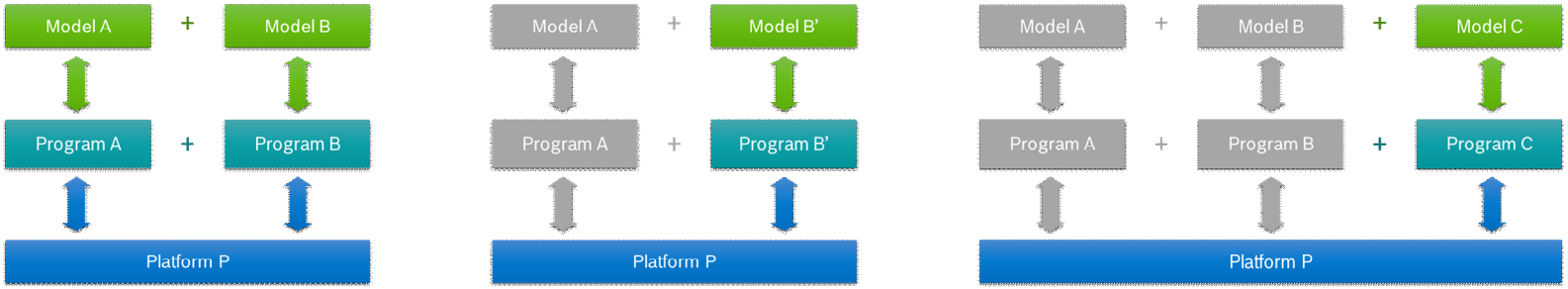


- **Understandable? NO**
- **Reusable?**
- **Efficient? YES**

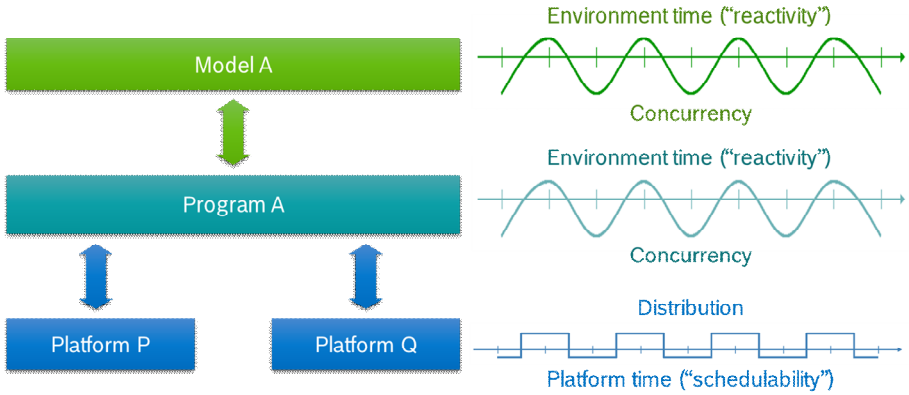
Inspired by Tom Henzinger

A Suitable Interface for Real-Time Control
Reusable = Composable + Portable

- Composable
  - Locality of changes
  - Key for development efficiency

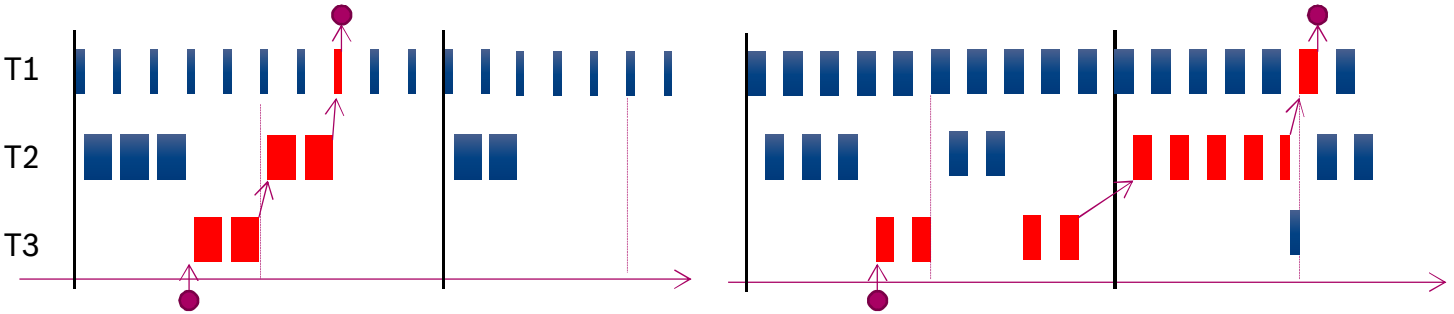


- Portability
  - Program can be mapped to different platforms
  - Key for validity of SiL/HiL tests
  - Following from this: interface needs to talk only about environment time

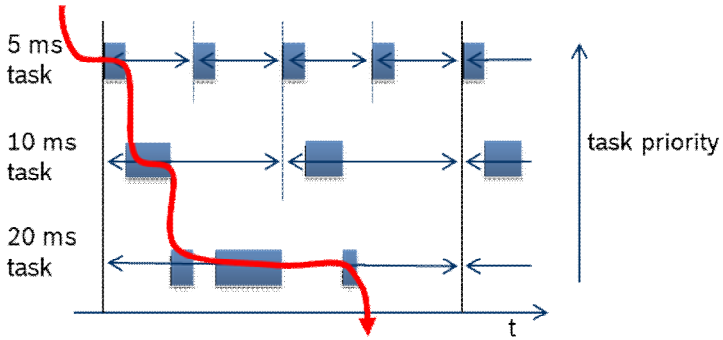


A Suitable Interface for Real-Time Control
Task Priorities & Deadlines – Reusable?

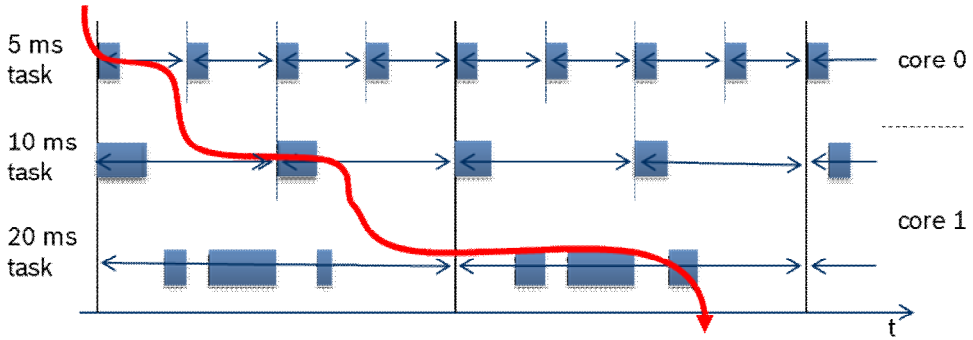
- Not composable
  - Example: Increase in execution time of T1 increases latency of cause effect chain



- Not portable
  - Latency depends on Task-core mapping

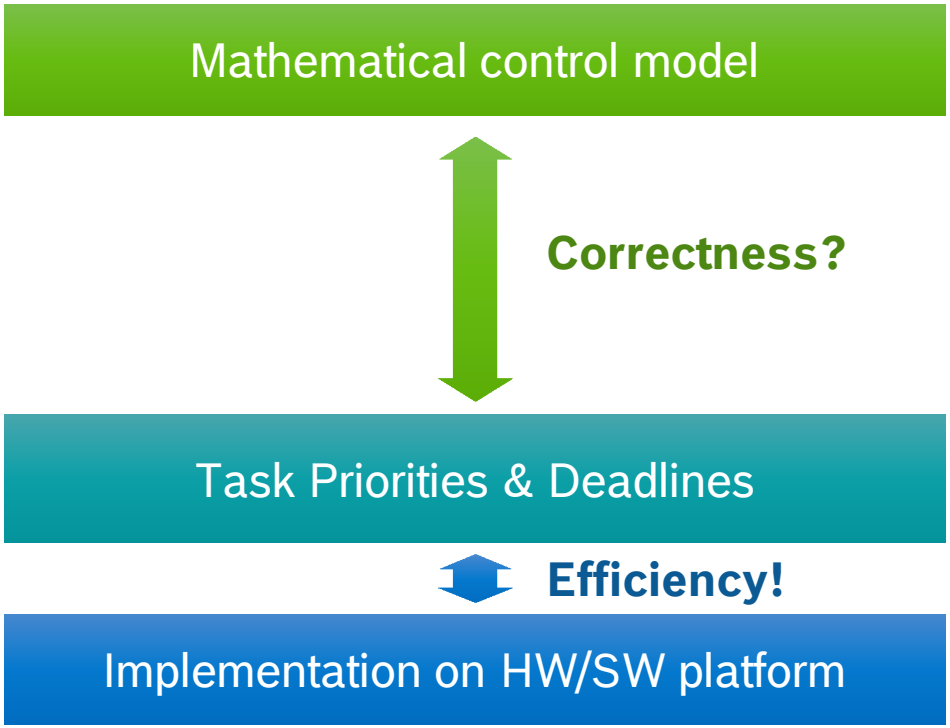


Single-Core



Possible Multi-Core Scenario

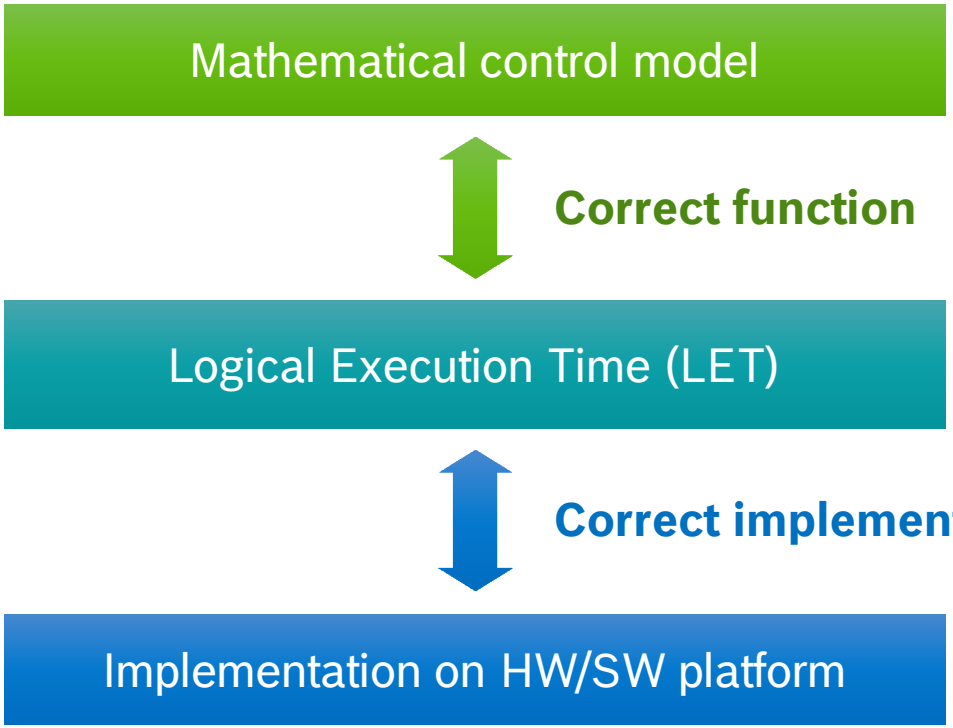
A Suitable Interface for Real-Time Control
Current Interface: Task Priorities & Deadlines



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

Inspired by Tom Henzinger

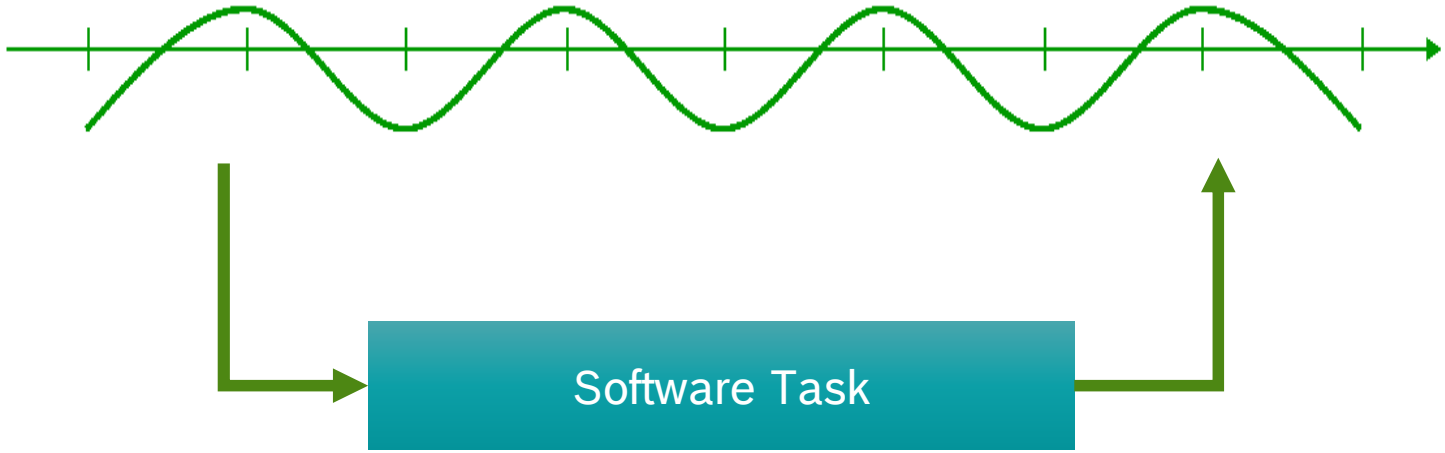
A Suitable Interface for Real-Time Control
Solution: Logical Execution Time



- Understandable?
- Reusable?
- Efficient?

Inspired by Tom Henzinger

A Suitable Interface for Real-Time Control
LET – Modeling Principle



Read input at time t

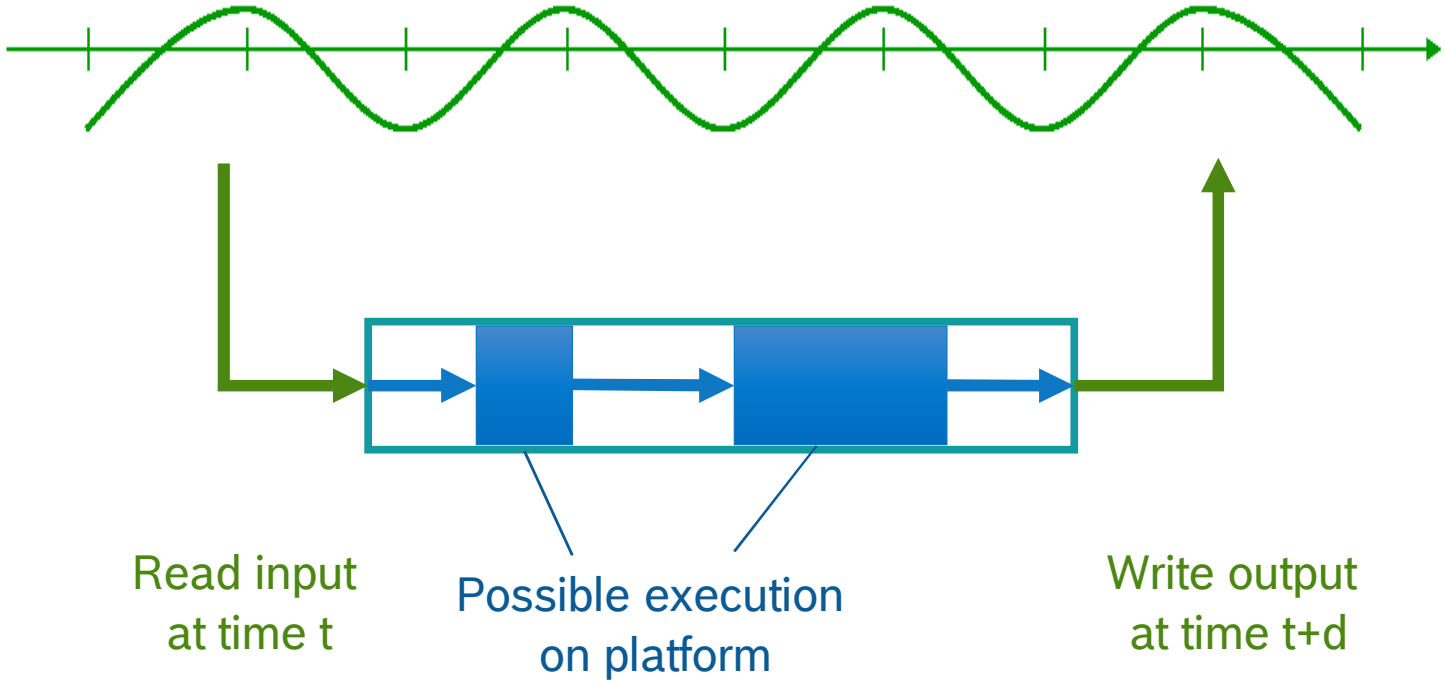
d is the task's logical execution time

Write output at time t+d

Control engineer's fiction: "platform is fast enough to compute the task in d"

Inspired by Tom Henzinger

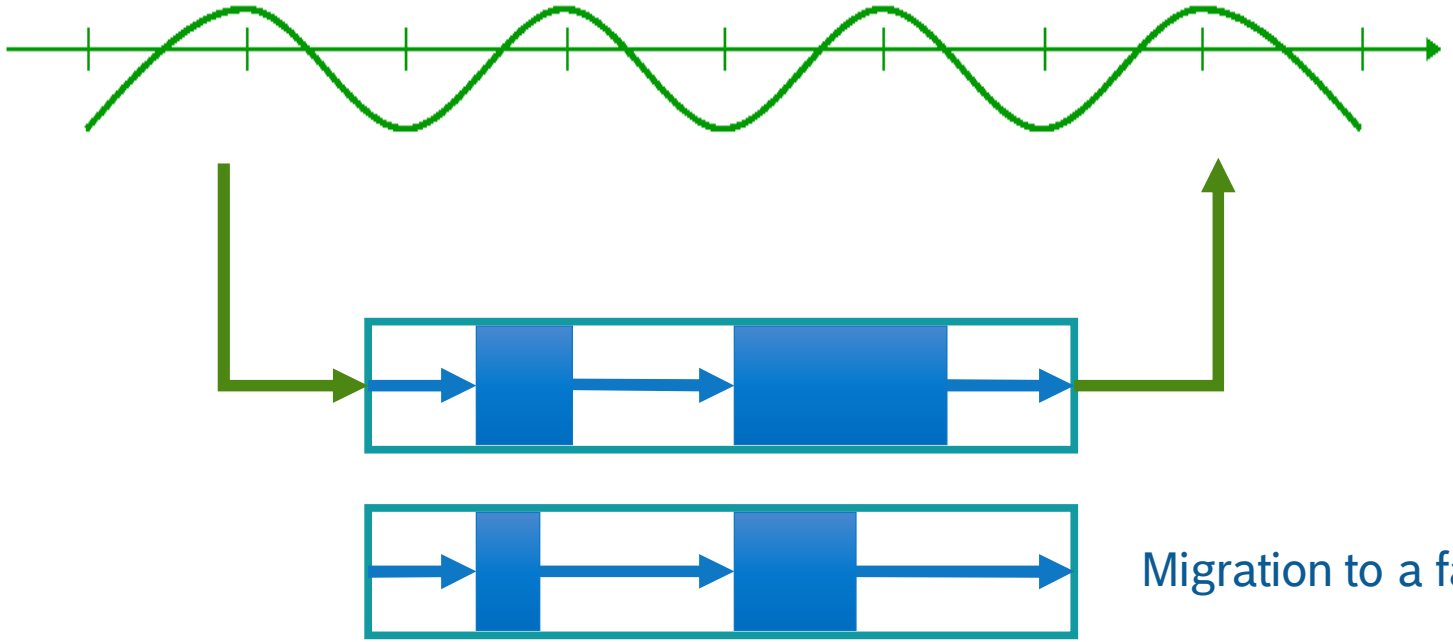
A Suitable Interface for Real-Time Control
LET – Implementation Principle



Control engineer's fiction: "platform is fast enough to compute the task in d"

Inspired by Tom Henzinger

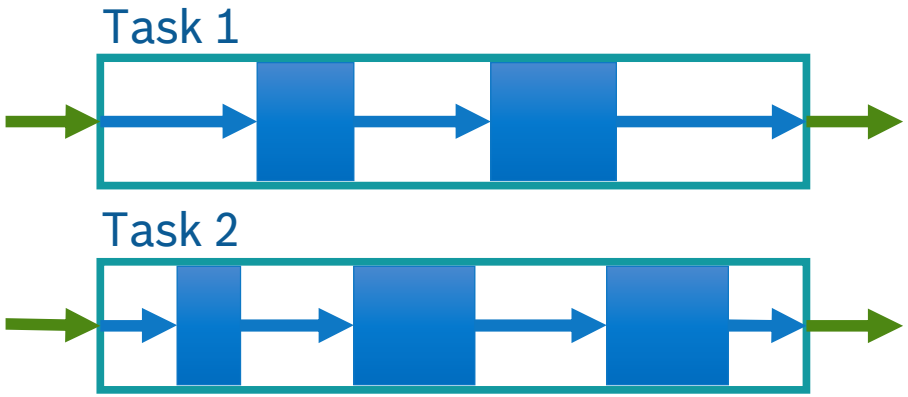
A Suitable Interface for Real-Time Control
LET – Portability



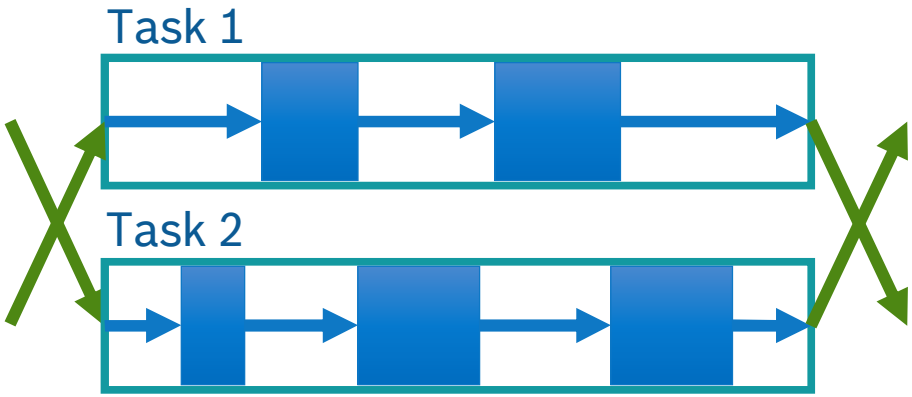
Migration to a faster platform

Inspired by Tom Henzinger

A Suitable Interface for Real-Time Control
LET – Composability & Determinism



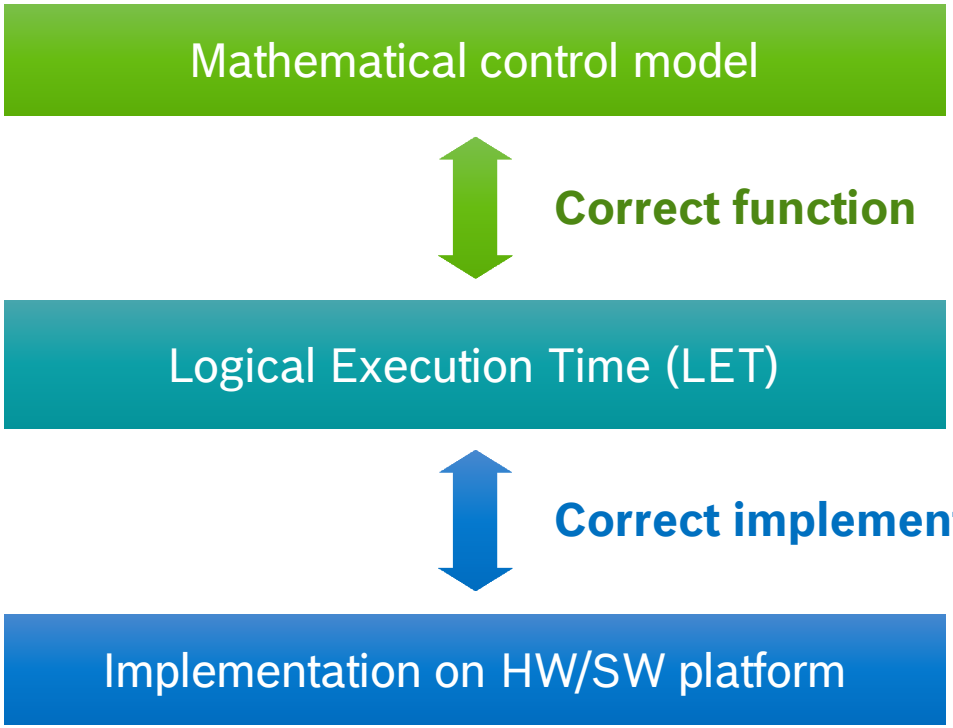
Composability
(no functional effect
of resource sharing)



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



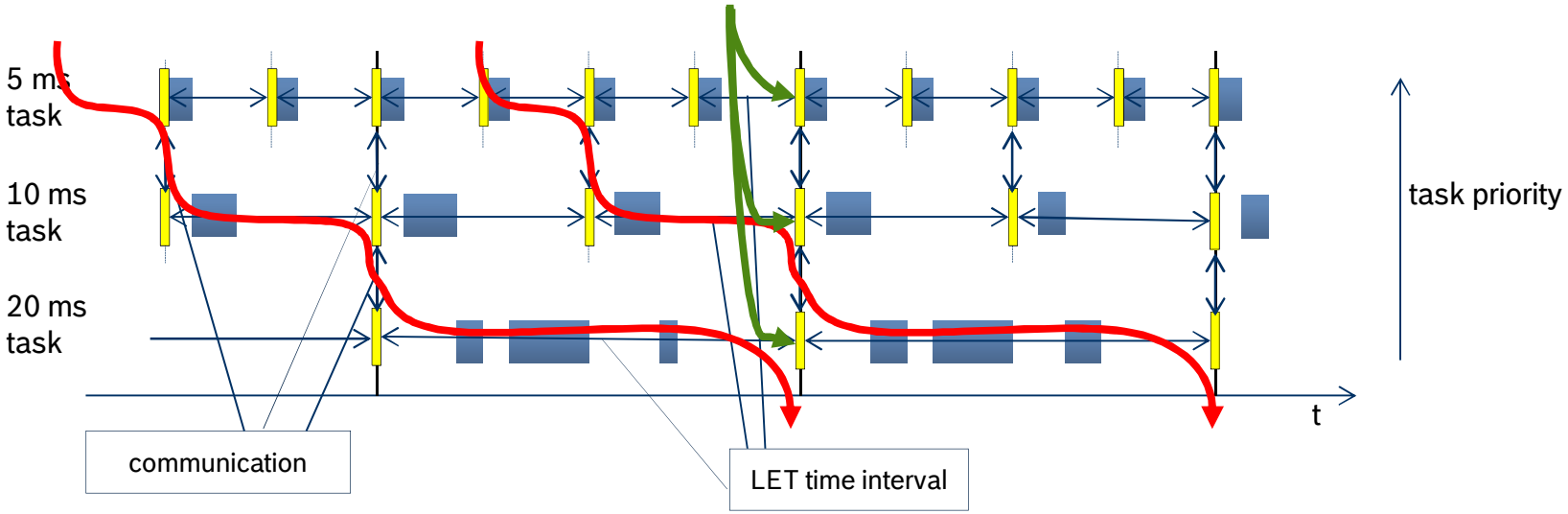
- **Understandable? YES**
- **Reusable? YES**
- **Efficient?**

Typical Doubts:

- **Resource penalty**
- **Latency penalty**

Inspired by Tom Henzinger

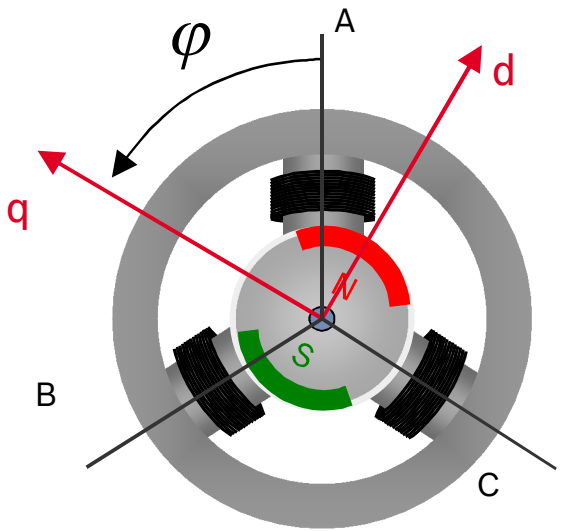
A Suitable Interface for Real-Time Control
“Timed Communication”: Tailored LET Solution @ Bosch



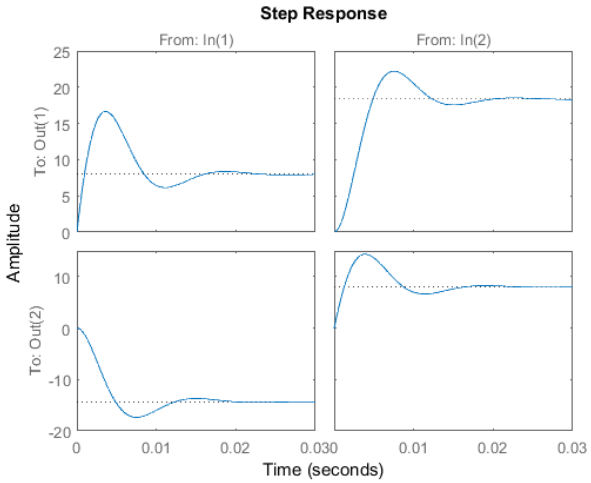
- Focused on Internal Determinism
  - Fixed (but increased) end-to-end latencies of cause and effect chains
  - Input data consistency of tasks with same LET interval start
- Logical execution time equals task period → enables efficient implementation
  - Average values from Engine Control Systems on 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

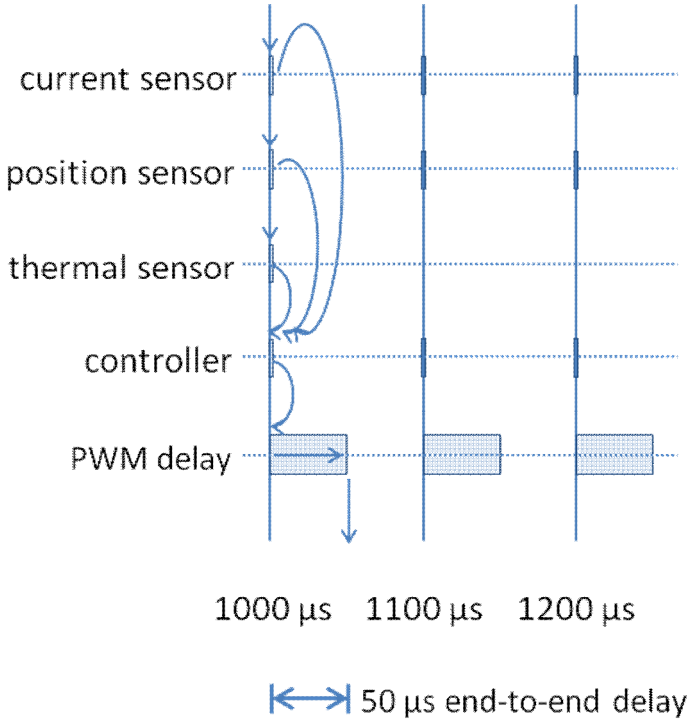


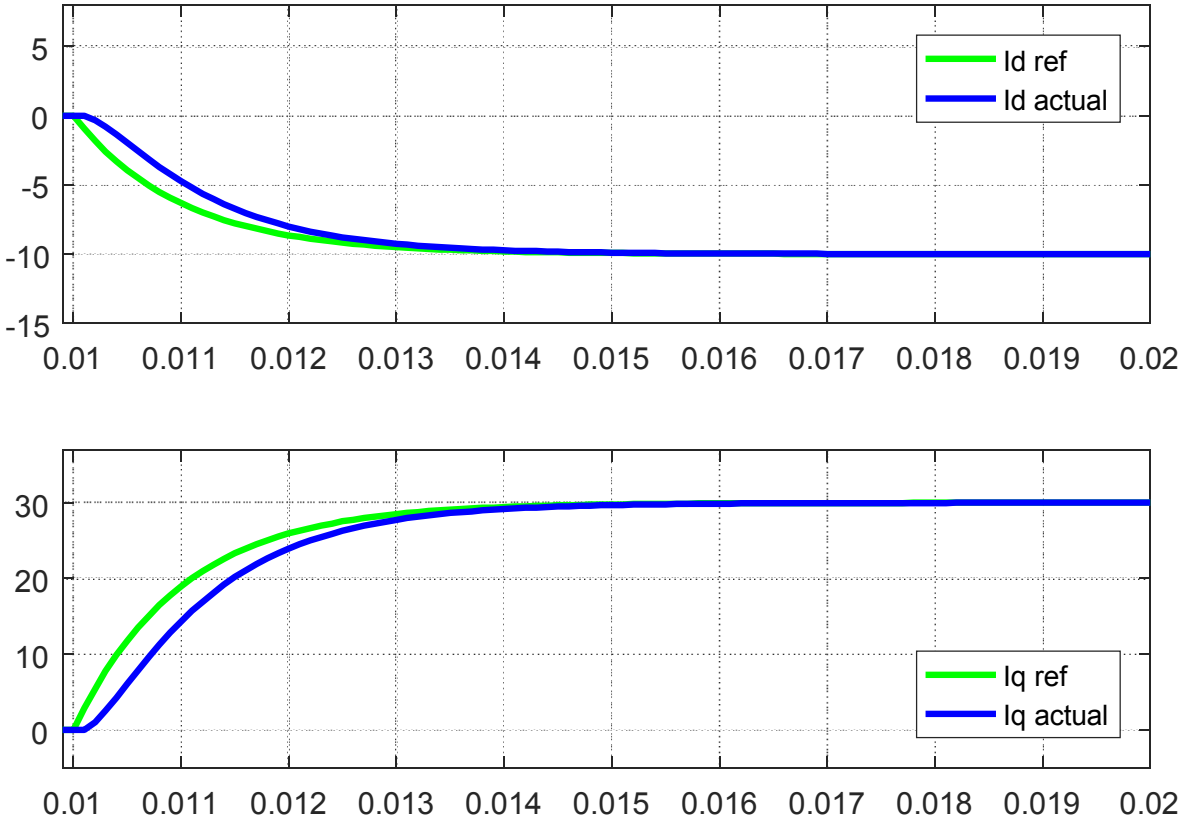
$$\begin{bmatrix} \dot{I}_d \\ \dot{I}_q \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{R}{L_d} & \frac{L_q \omega_{el}}{L_d} \\ -\frac{L_d \omega_{el}}{L_q} & -\frac{R}{L_q} \end{bmatrix}}_{=:A} \begin{bmatrix} I_d \\ I_q \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \end{bmatrix}}_{=:B} \begin{bmatrix} V_d \\ V_q \end{bmatrix}$$



A Suitable Interface for Real-Time Control

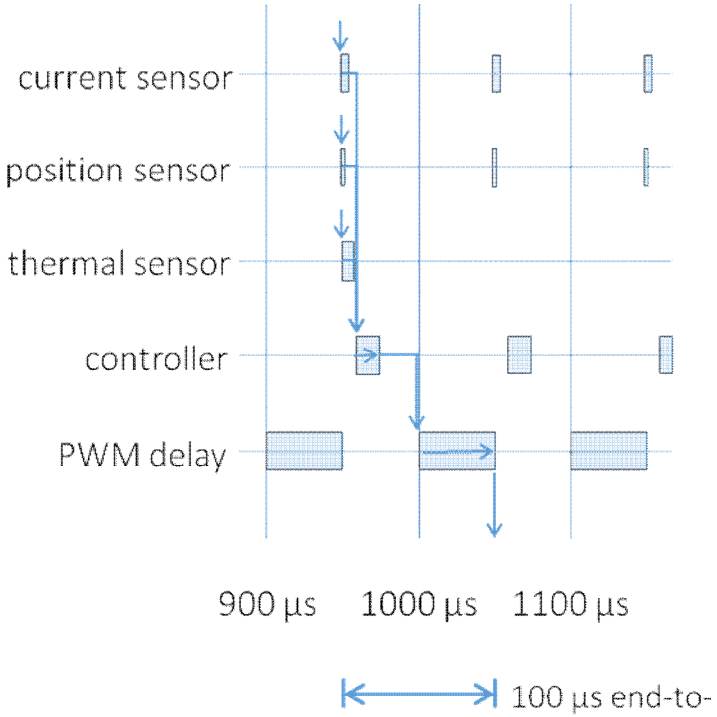
Simulation: 0.5-step Delay

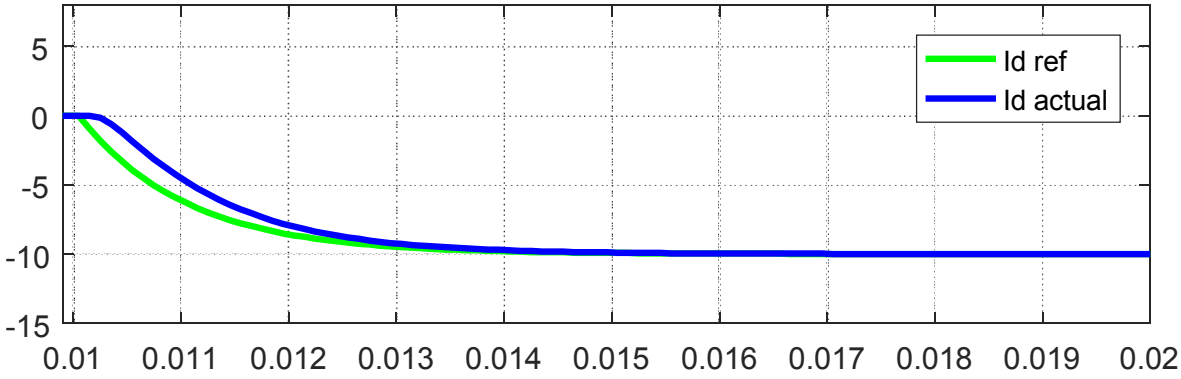


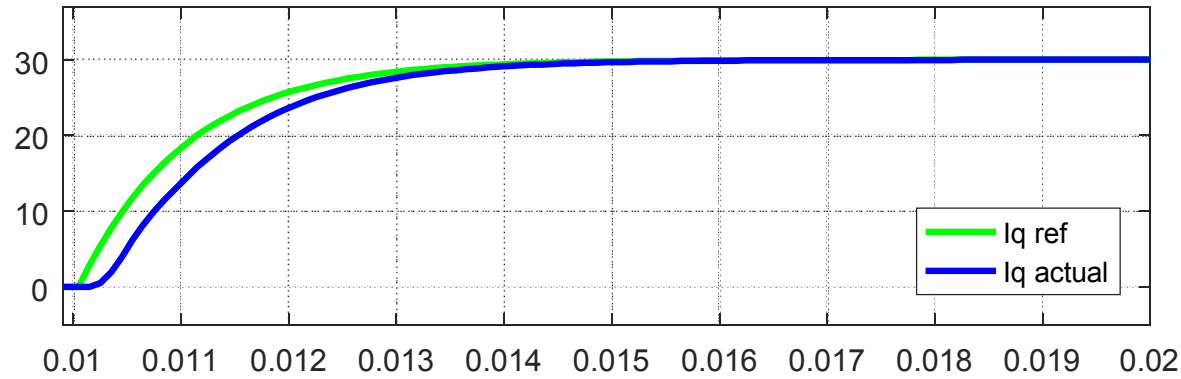


A Suitable Interface for Real-Time Control

Simulation: 1-step Delay

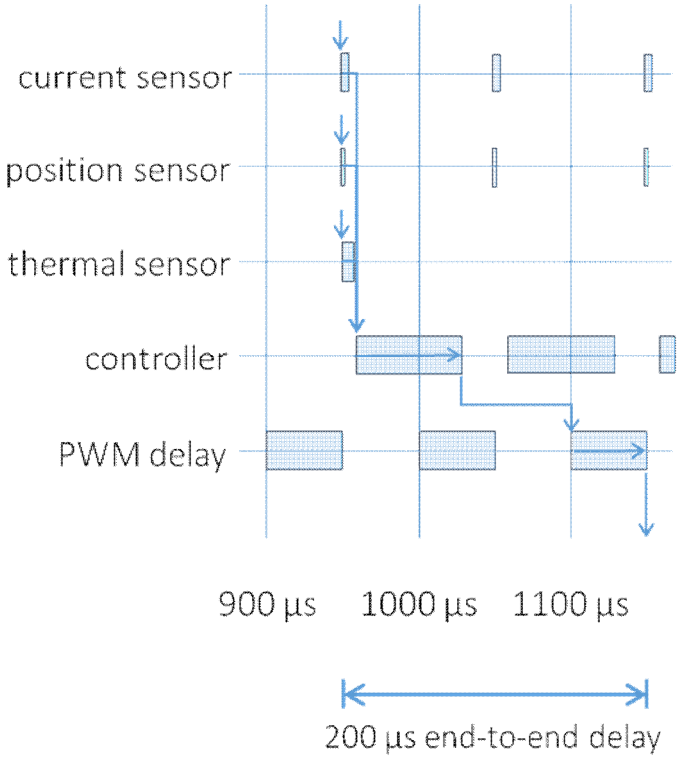


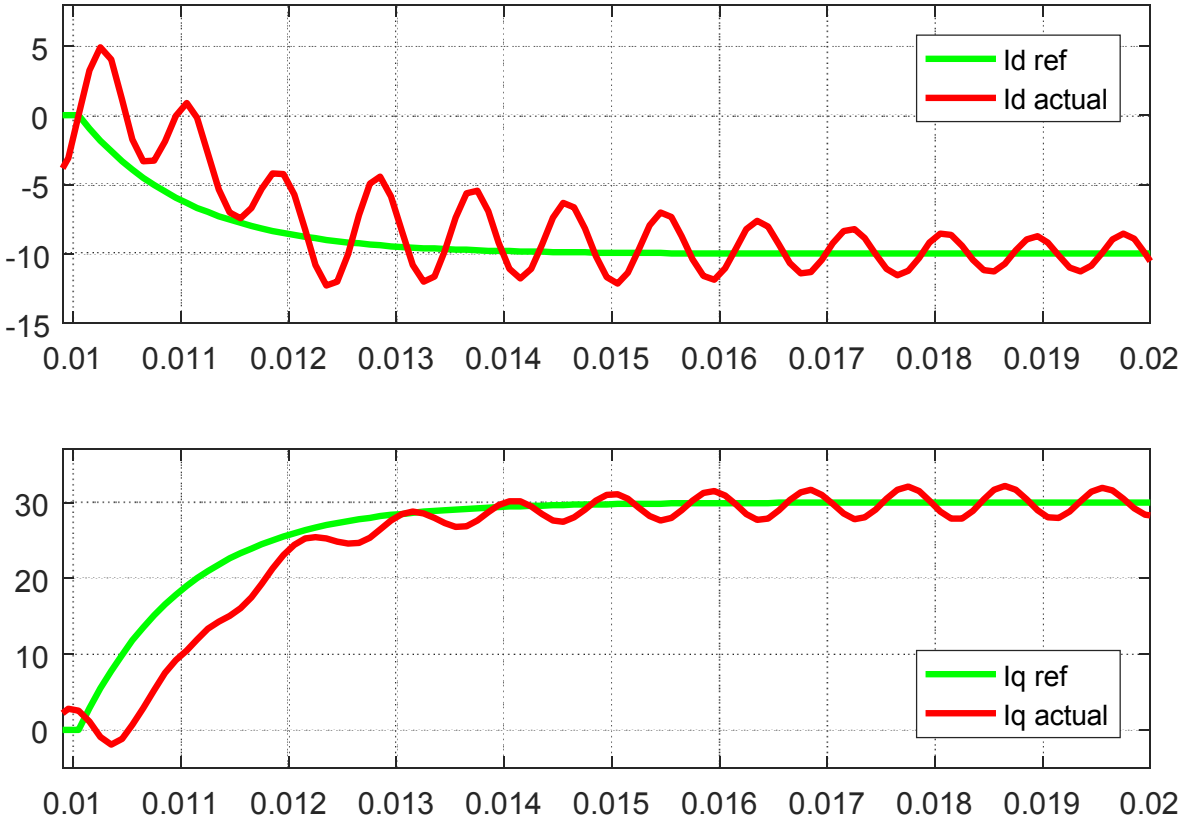




A Suitable Interface for Real-Time Control

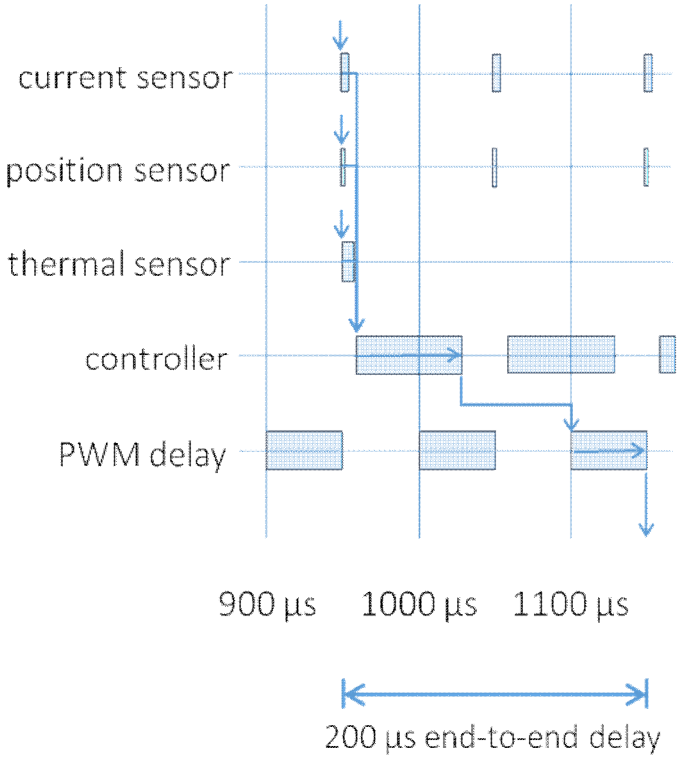
Simulation: 2-step Delay

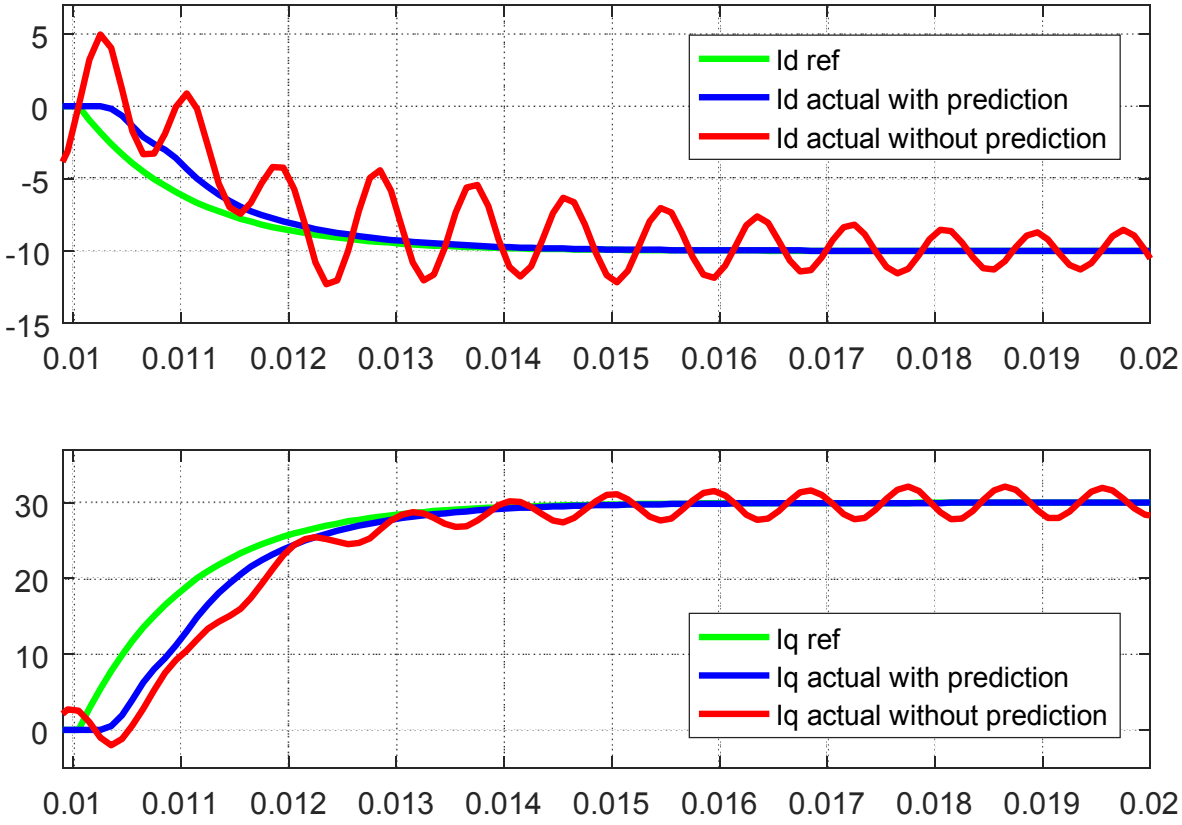




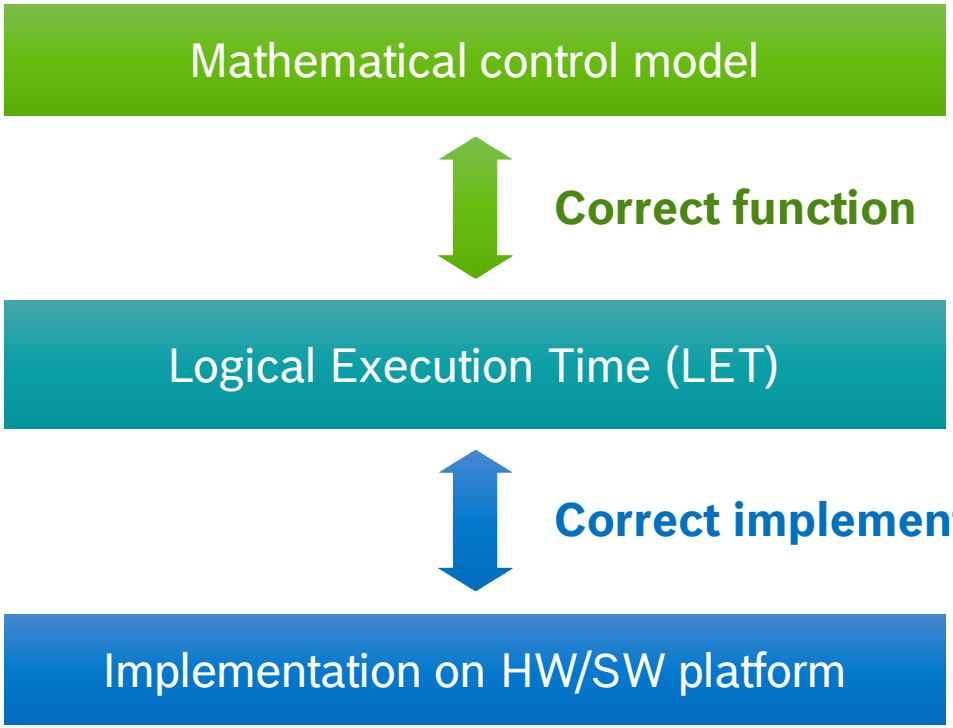
A Suitable Interface for Real-Time Control

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





A Suitable Interface for Real-Time Control
Solution: Logical Execution Time



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

Typical Doubts:

- **Resource penalty**
- **Latency penalty**

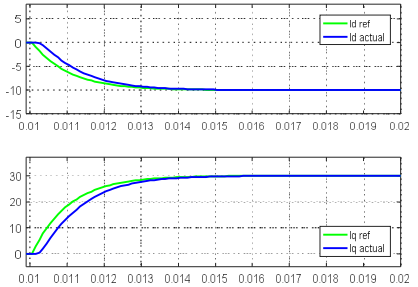
Inspired by Tom Henzinger

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

z^{-1}

z^{-2}



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:
  - Kramer et al., “Real World Automotive Benchmarks for Free”, WATERS 2015
  - Hamann et al., “Communication Centric Design in Complex Automotive Embedded Systems”, ECRTS 2017

APP4MC

https://www.eclipse.org/app4mc/

BOSCH

A Suitable Interface for Real-Time Control
So are we done yet?

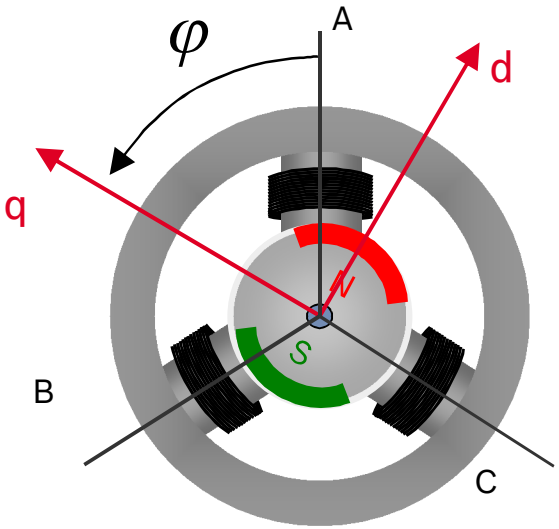
- Unfortunately, for many automotive systems
  - Worst case response times >> average case response times
    - Due to average-case optimized HW platforms (and legacy SW structure)
  - Sporadic overload and missed deadlines are often tolerable
    - Functional impact negligable
    - Due to robust design (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

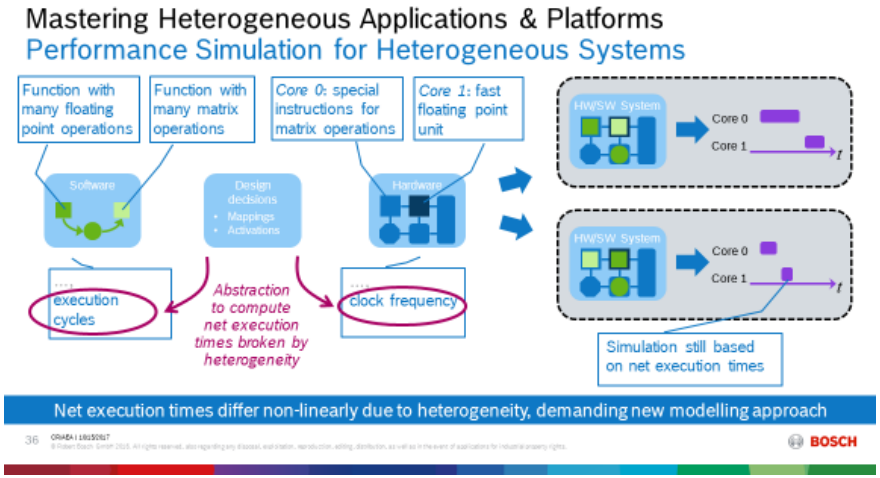
- Higher dynamics – shorter time constants
  - Periods of control functions significantly shorter (in 100µs range)
  - Current degree of oversampling not affordable in terms of computing power
  - High sensitivity of efficiency to deadline misses

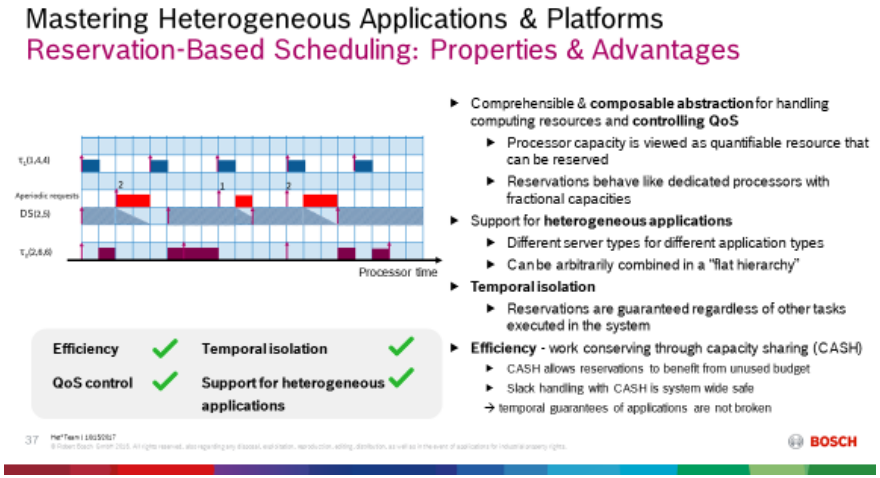


- **High interest in (m out of k) control theory and timing verification**

A Suitable Interface for Real-Time Control
Further Upcoming Challenge: Heterogeneity

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





- 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

THANK YOU

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

BOSCH