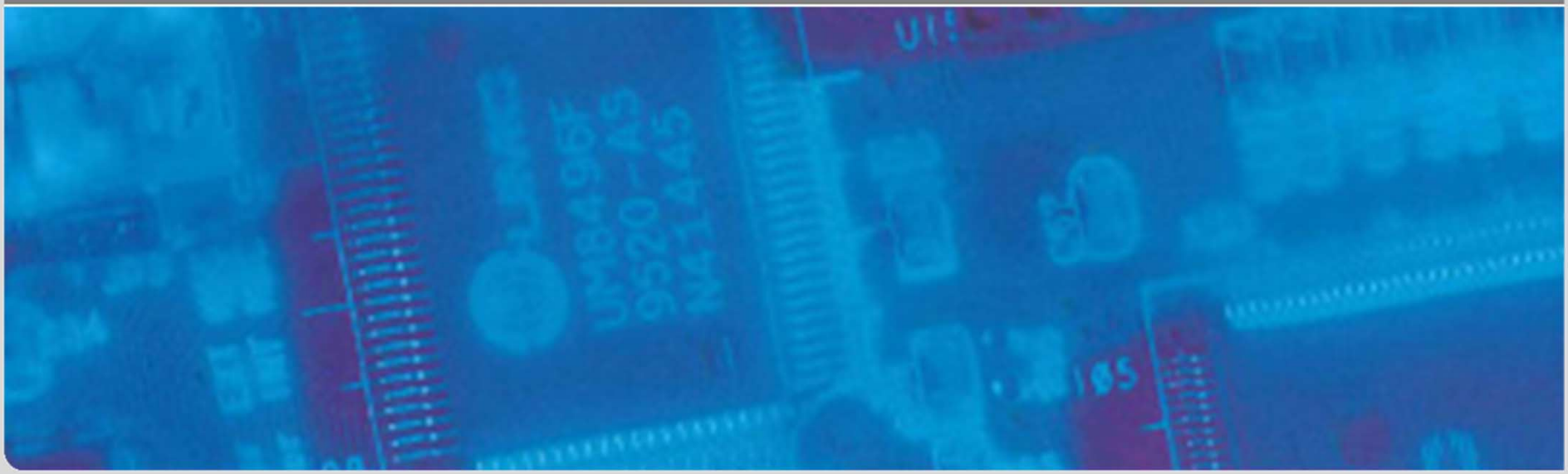
Karlsruhe Institute of Technology

Improving Parallel MPSoC Simulation Performance by Exploiting Dynamic Routing Delay Prediction

Christoph Roth, Harald Bucher, Simon Reder, Oliver Sander, Jürgen Becker

Institut für Technik der Informationsverarbeitung (ITIV)



KIT – University of the State of Baden-Wuerttemberg and
National Research Center of the Helmholtz Association

Outline

- Motivation
- Baseline Modeling and Simulation Methodology
- Dynamic Routing Delay Prediction
  - Overview
  - Router Shortest Path Graph
  - Local Quantum Calculation
  - Execution Semantics of the Lightweight Scheduler
- Case Study
- Conclusion and Outlook

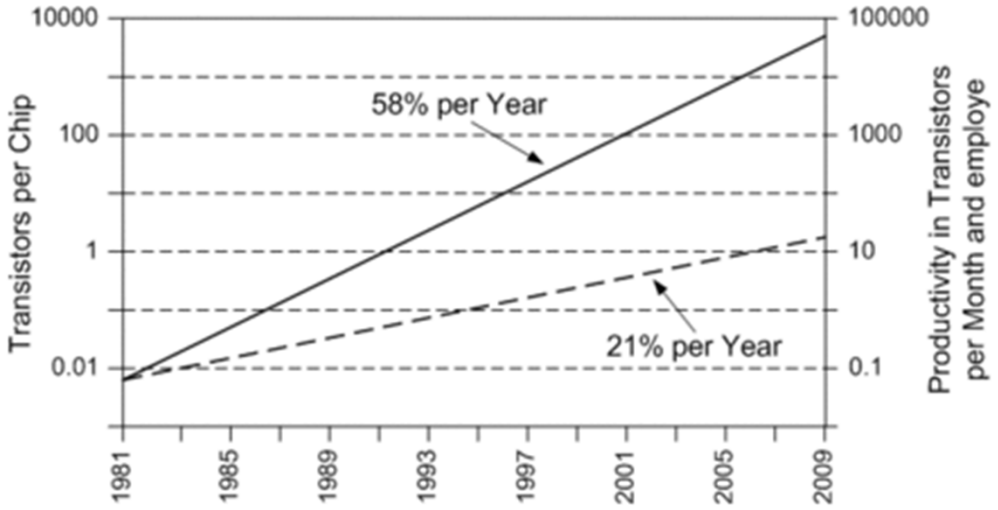
Motivation I

- Evaluation of Network-on-Chip (NoC) based Multi-Processor System-on-Chip (MPSoC) platforms regarding throughput, latency or jitter demands for …
  - rendering of effects like internal network congestion and buffer contention
  - detailed cycle-timed modeling of mechanisms like routing algorithm or flow control in order to measure their impact
  - consideration of the whole NoC and not only a single router
  - utilization of realistic application-generated traffic

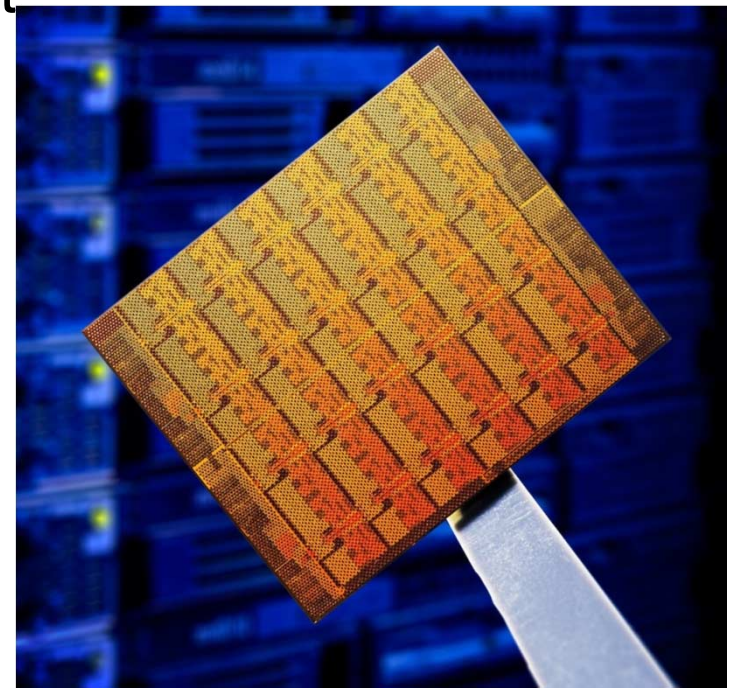
Full system simulation is a commonly used approach for performing such evaluations

Motivation II

- The gap between MPSoC complexity and productivity in design tools evermore increases
- Simulation-based evaluation and verification consumes much of the MPSoC design effort



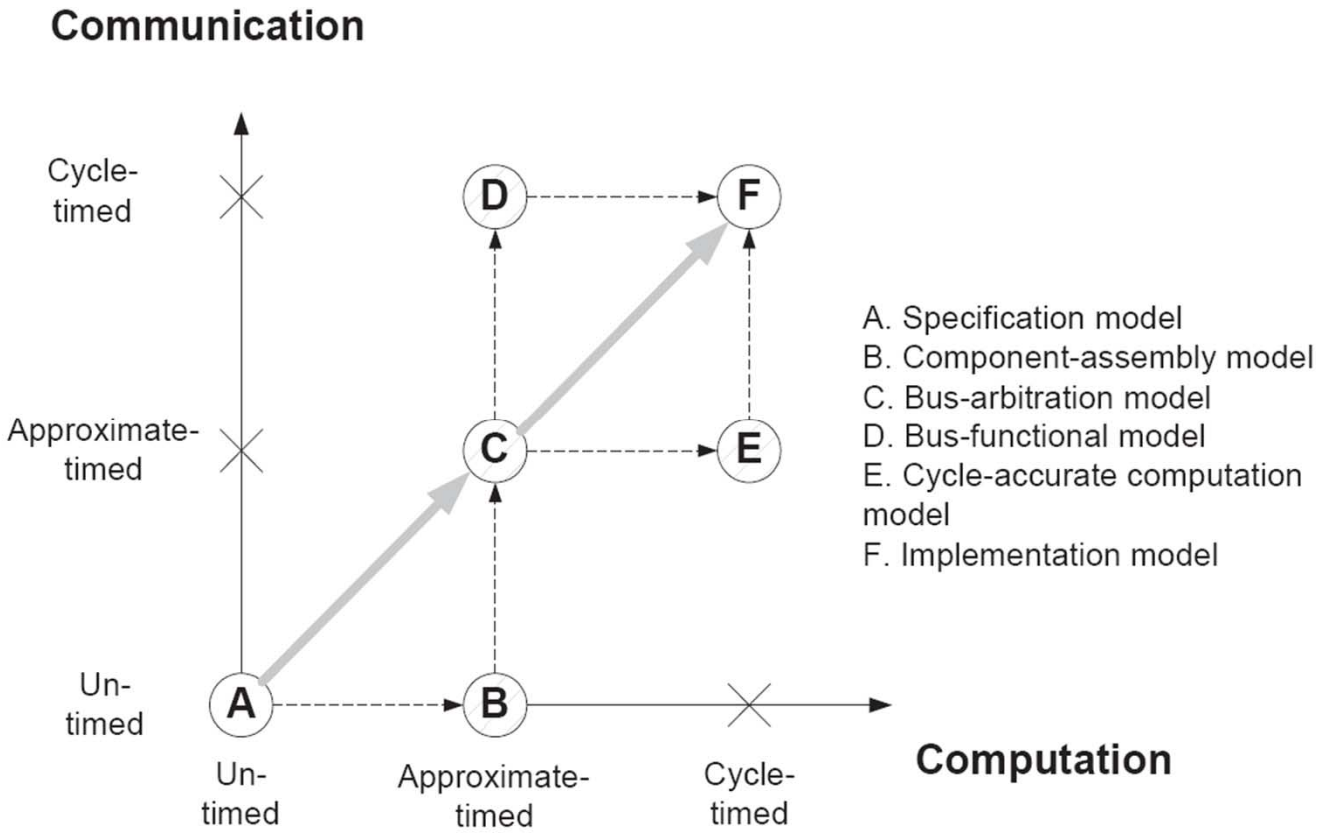
Source: http://www.imd.uni-rostock.de/noc/index.php?id=36



Source:http://www.intel.com/pressroom/archive/releases/2009/20091202comp_sm.htm

Possible Solution I: Raising Abstraction Level

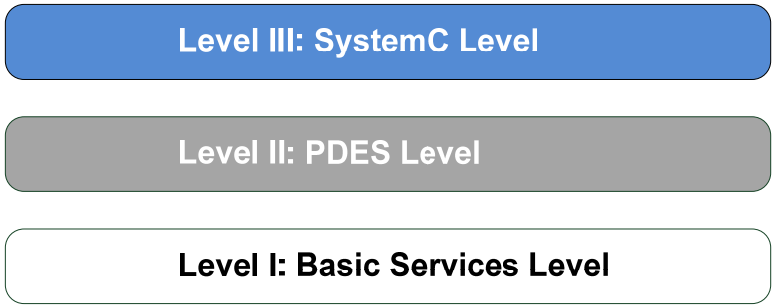
- Abstraction of communication and computation in **transaction level modeling** (TLM)



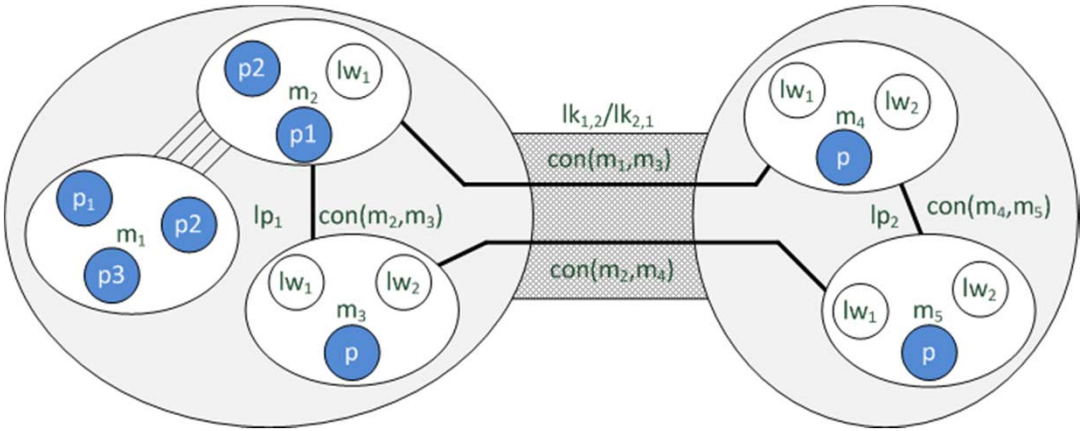
Source: Cai, L. & Gajski, D. Transaction level modeling: an overview *Hardware/Software Codesign and System Synthesis, 2003. First IEEE/ACM/IFIP International Conference on,* 2003, 19-24

Possible Solution II: Parallel Simulation

- Parallel SystemC framework: Runtime system for simulating multi-cores on multi-cores



- Support for different execution strategies like **logical process** (LP) based **parallel discrete event simulation** (PDES)

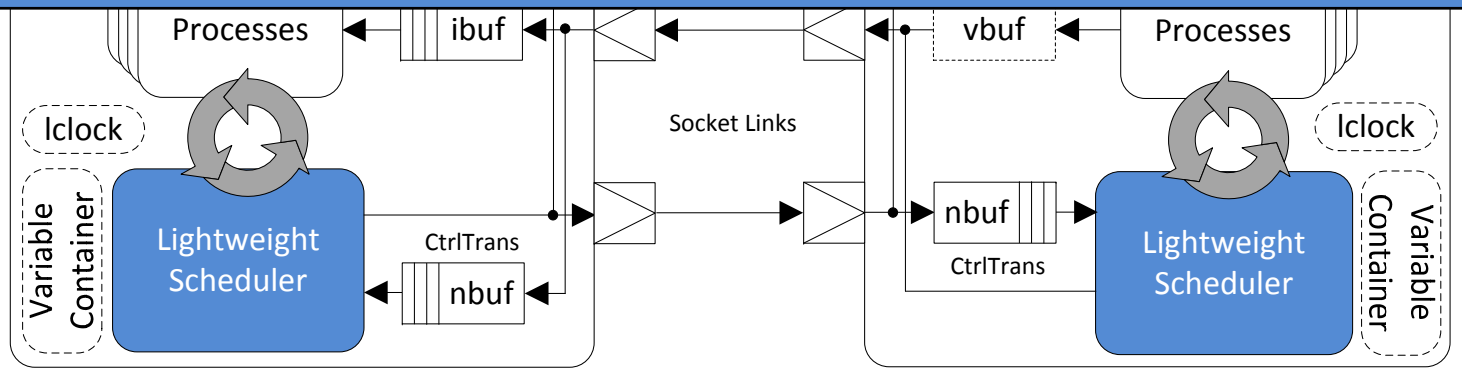


C. Roth et al., “Asynchronous parallel MPSoC simulation on the Single-Chip Cloud Computer,” in System on Chip (SoC), 2012 International Symposium on, 2012.

Baseline TLM Method for NoC-based MPSoCs

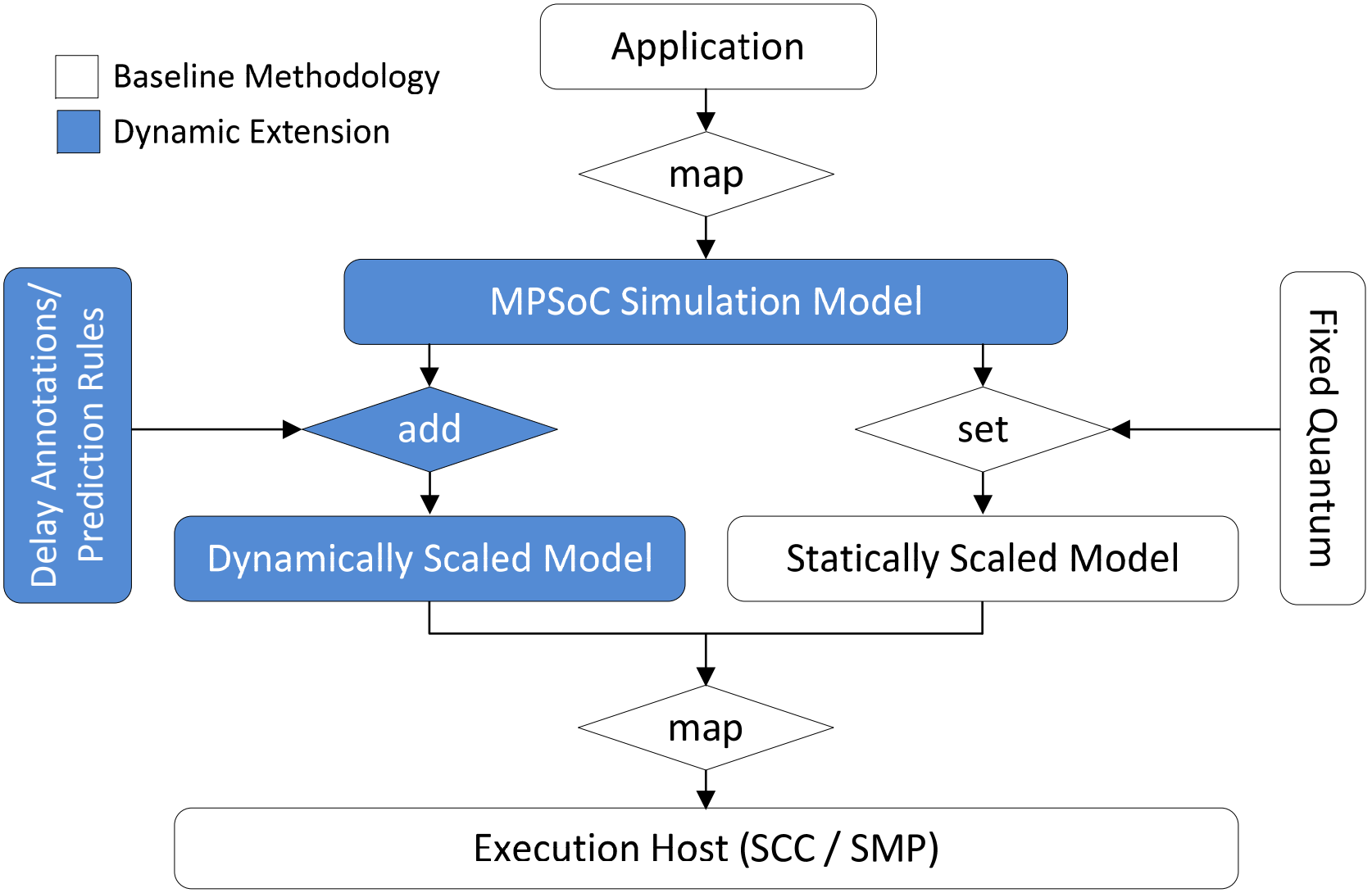
- Inherently parallelizable transaction level modeling method
- Intra-module behavior (e.g. router):
  - **Lightweight scheduler** and **processes** implement synchronous execution semantics
  - Maintenance of a **local time** in order to be able to run ahead of the global time
- Inter-module behavior (inspired by *):
  - Transaction-based communication
  - Adjacent modules regularly synchronize via control transactions after a predefined **global time quantum**
  - **Better performance but loss of timing accuracy if time quantum is chosen larger than one clock cycle**

Goal: Dynamically adapt temporal decoupling locally to avoid loss of accuracy



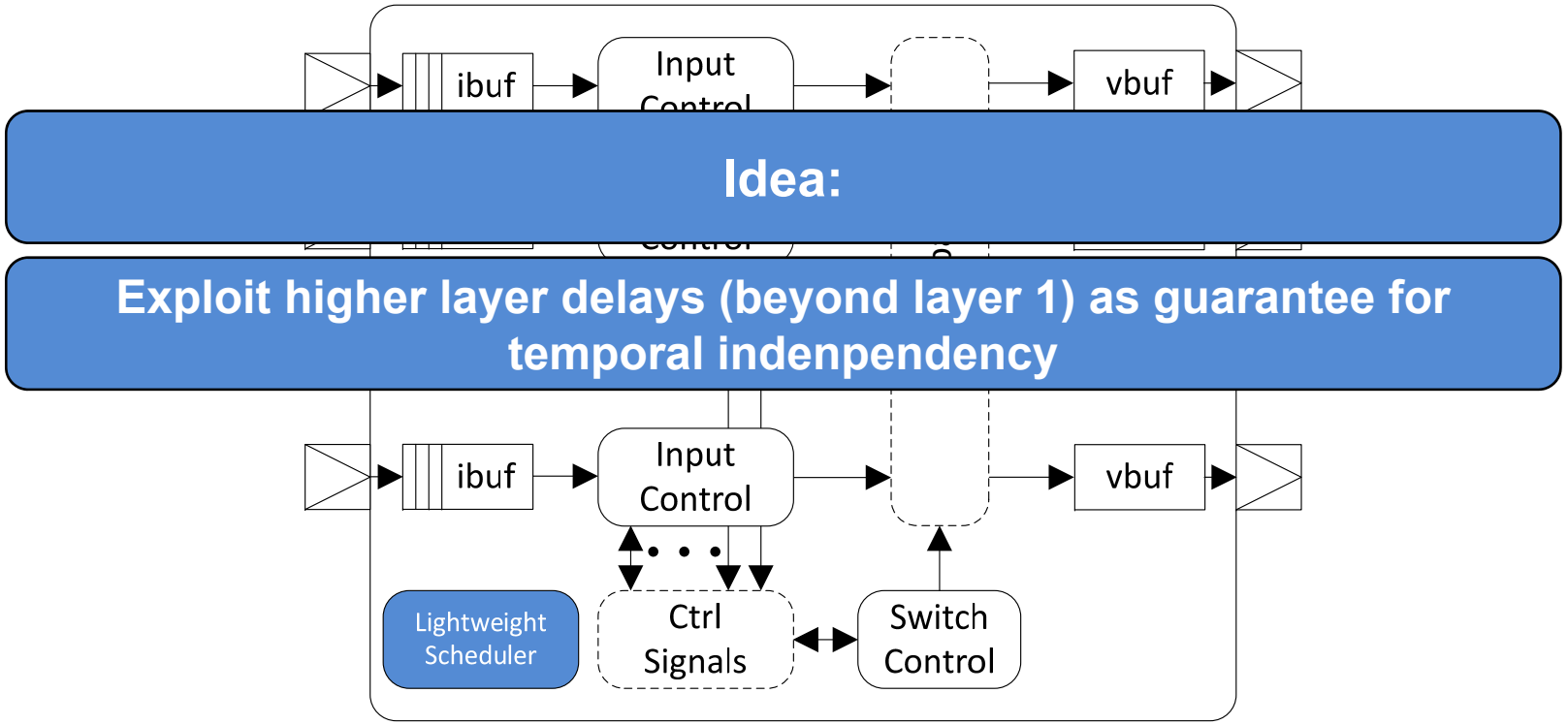
* A. Mello et al., "Parallel simulation of systemc tlm 2.0 compliant mpsoc on smp workstations," in Design, Automation Test in Europe Conference Exhibition (DATE), 2010, 2010.

Proposed Extension within this Work



Dynamic Routing Delay Predicition

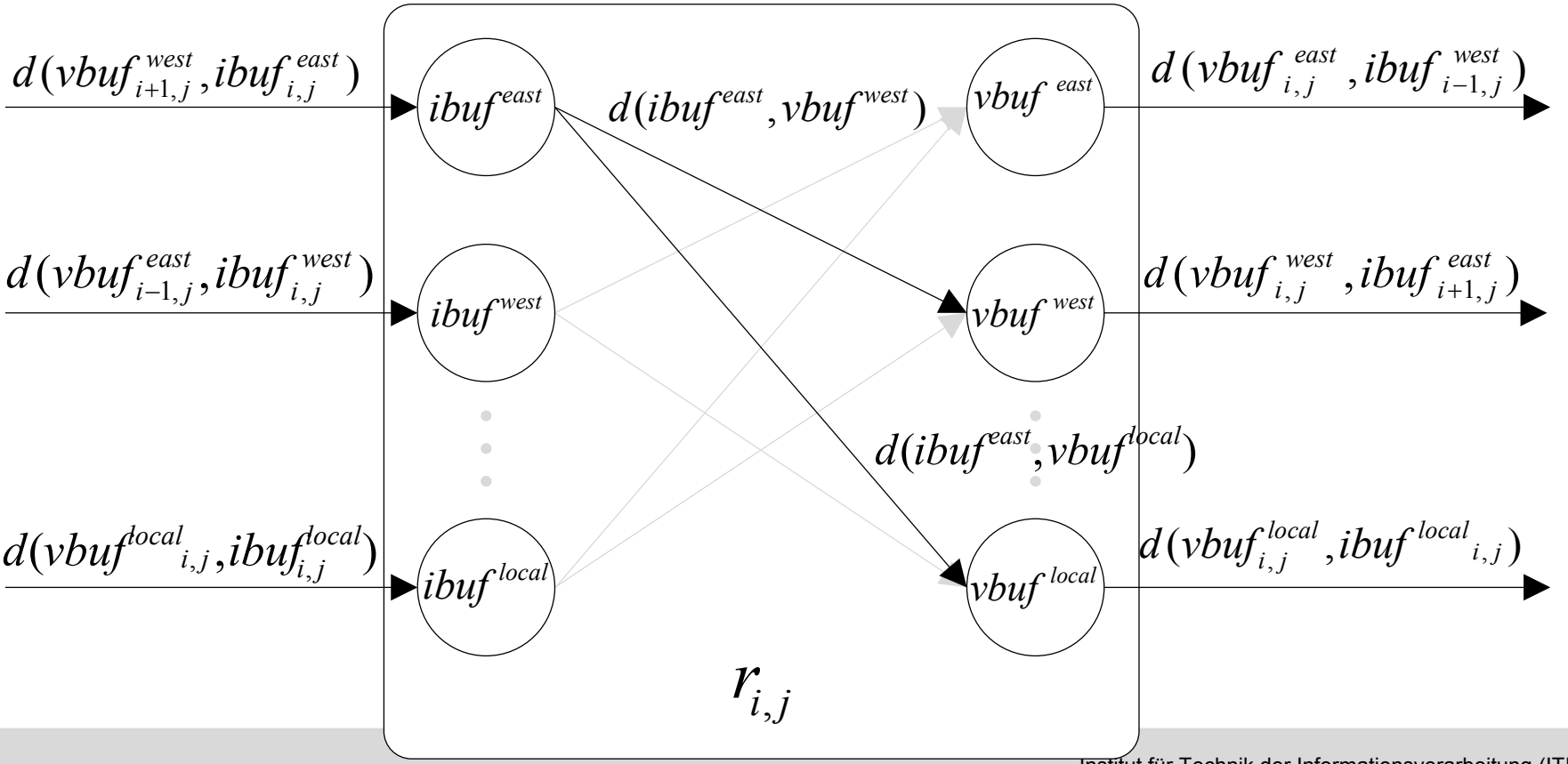
- Example: TL model of the Hermes* router
- ISO/OSI reference model analogy:
  - Buffers = Physical Layer (Layer 1)
  - Input Controllers = Data Link Layer (Layer 2)
  - Switch Controller = Network Layer (Layer 3)



* F. Moraes et al., “HERMES: an infrastructure for low area overhead´packet-switching networks on chip,” Integr. VLSI J., vol. 38, 2004.

Router Shortest Path Graph (RSPG)

- A $RSPG(V, E, D)$ is a directed graph with vertices $v \in V$, edges $e \in E$ and edge weights $d \in D$. Two vertices v^x and v^y are connected by a directed edge $e^{x,y} = (v^x, v^y)$ if there exists a causal dependency in terms of a minimum delay bound after which v^x may affect v^y. The weight of the edge $e^{x,y}$ is the minimum delay bound $d(v^x, v^y)$.



Calculation of Local Time Quanta and Next Synchronization Times between Modules

- Step I: Calculate **Dynamic Lookahead** from RSPG (delays are annotated to behavioral model)

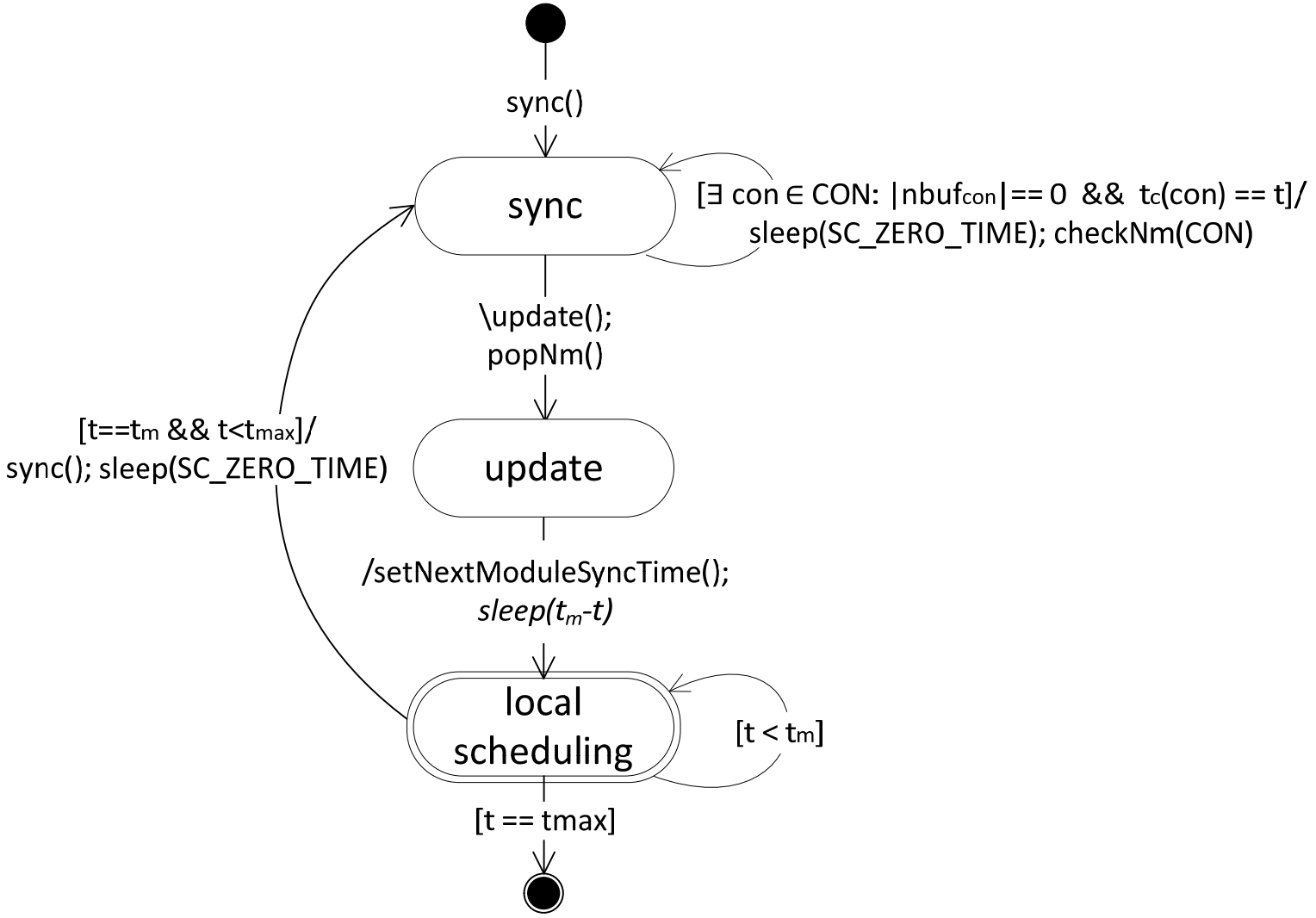
$$d^{y}_{min} = min_{x \in Dir}[d(ibuf^{x}, vbuf^{y})]$$

$$l(r_{i,j}, r_{m,n}) = d^{y}_{min}(i,j) + d(vbuf^{y}_{i,j}, ibuf^{z}_{m,n})$$

- Step II: Calculate **Local Time Quantum** and **Next Synchronization Time** of module connection

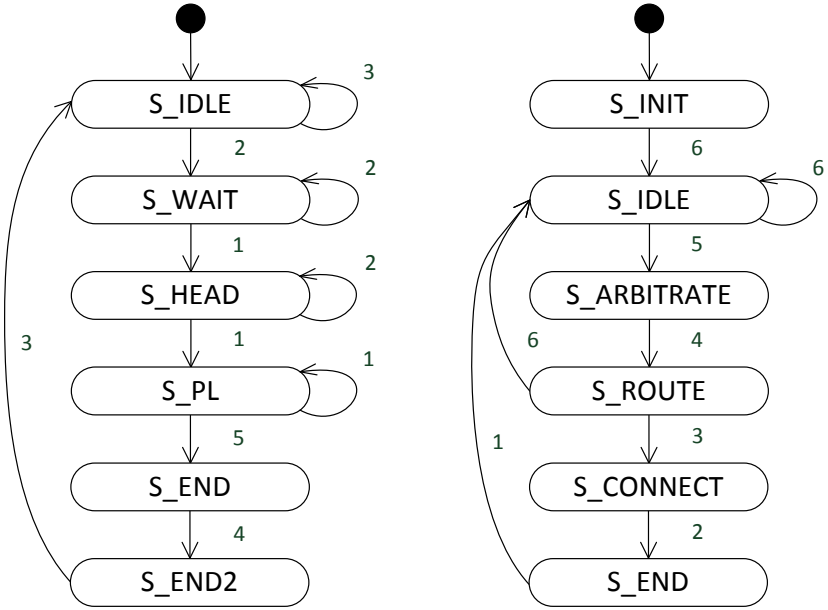
$$\begin{aligned} t_c(con) &= t + min[l(r_{i,j}, r_{m,n}), l(r_{m,n}, r_{i,j})] \\ &= t + t_l(con). \end{aligned}$$

Appropriate Lightweight Scheduler



Case Study: Hermes Multiprocessor System

- Layer 1: Buffer delay is still a constant of one clock cycle
- Layer 2 & 3: Annotation of state depending delays to Input Control (left) and Switch Control (right) state machines



- From these the dynamic lookahead can be calculated within the `sync()` action of the lightweight scheduler by calling `calc_lookahead()`

Implementation of Dynamic Lookahead Calculation

```
1:  function CALC_LOOKAHEAD(output y)
2:      lookahead l = ∞
3:      for all x ∈ InputController do
4:          delay d ← 0
5:          if s(x) = S_WAIT then
6:              d ← d_sc
7:          else if s(x) = S_HEAD ∨ s(x) = S_PL then
8:              if out(x) = y then
9:                  d ← d_ic(x)
10:             else
11:                 d ← pc(x) + d_sc^min
12:             end if
13:         else
14:             //all other states:
15:             d ← d_ic(x) + d_sc^min
16:         end if
17:         l ← min(l, d)
18:     end for
19:     l ← l + d_buf − t_cycle
20: end function
```

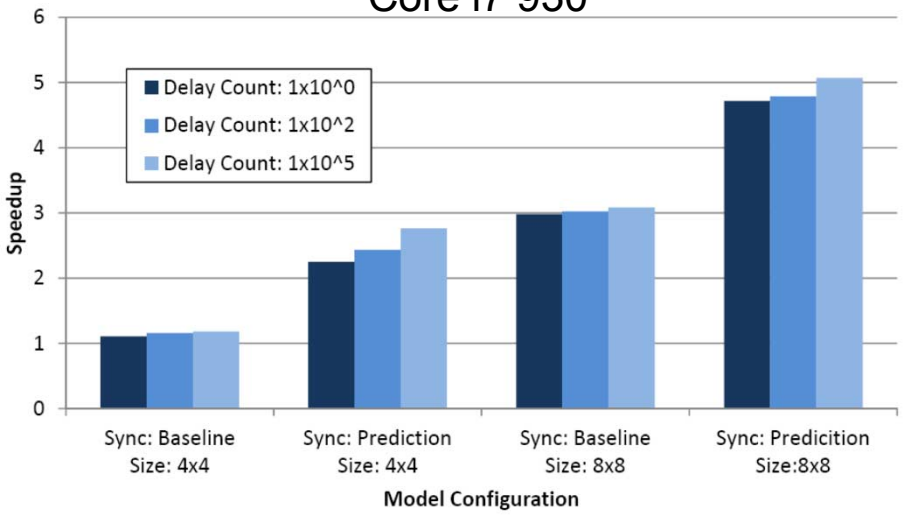
Evaluation: Host Independent Characteristics

Model Size	Delay Count	Prediction	Nm/Router Socket Link	Cycles/Nm
4x4	10^0	off	1×10^6	1.0
4x4	10^0	on	4.21×10^5	2.38
4x4	10^2	off	1×10^6	1.0
4x4	10^2	on	4.06×10^5	2.46
4x4	10^5	off	1×10^6	1.0
4x4	10^5	on	3.58×10^5	2.79
8x8	10^0	off	1×10^6	1.0
8x8	10^0	on	3.90×10^5	2.56
8x8	10^2	off	1×10^6	1.0
8x8	10^2	on	3.83×10^5	2.61
8x8	10^5	off	1×10^6	1.0
8x8	10^5	on	3.37×10^5	2.97

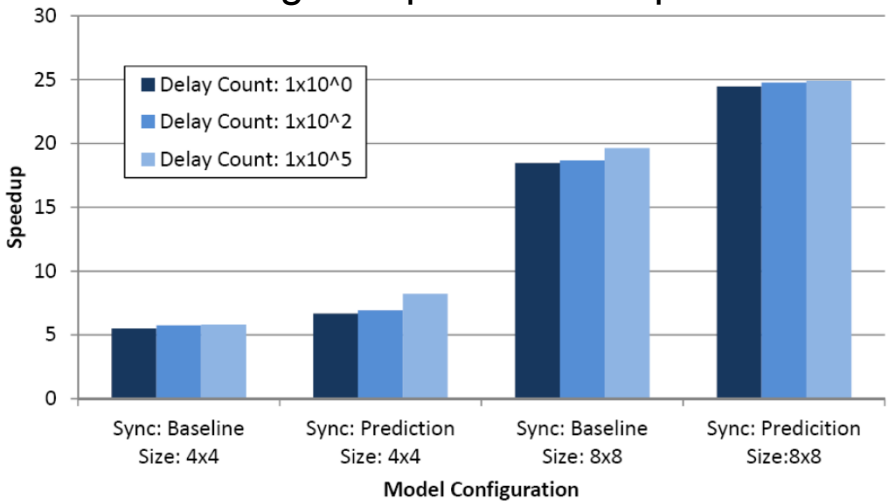
Evaluation: Host Dependent Characteristics

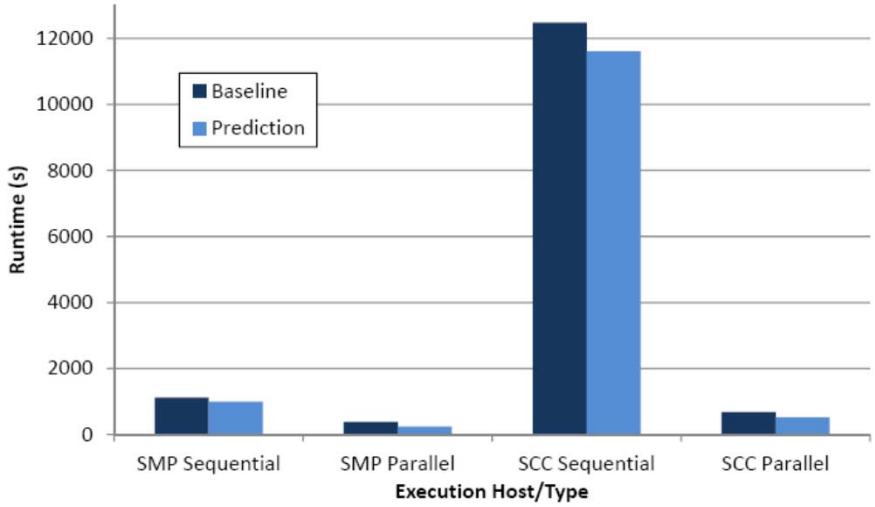
Karlsruhe Institute of Technology

Core i7 930



Single-chip Cloud Computer





Conclusion & Outlook

- Presentation of an extension for a modeling strategy for NoC-based MPSoCs using SystemC/TLM
- Maintenance of cycle accuracy despite temporal decoupling is possible by dynamically predicting and adapting the degree of temporal decoupling
- The number of control transactions is significantly reduced compared to the baseline strategy. This improves performance on different types of hosts.
- Future work comprises:
  - Application of the method to other types of modules like network interface of processor
  - Deeper investigation of scalability and application dependency
  - Automization of model partitioning and host mapping

Karlsruhe Institute of Technology

Thank you very much for your attention!