Formal methods in MpSoC architecture optimization

R. Ernst

TU Braunschweig

Overview

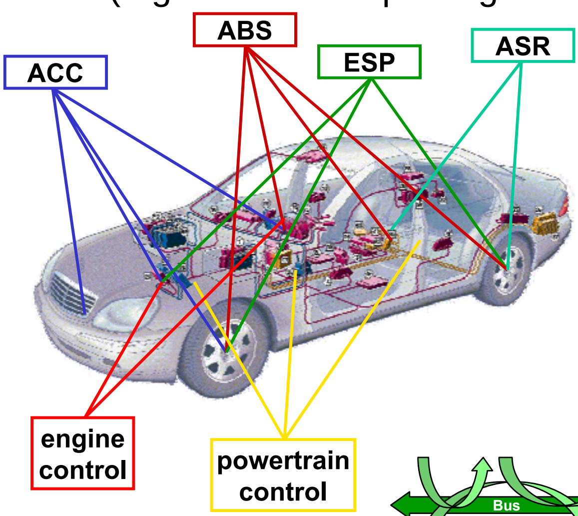
- introduction
- modeling requirements
- a popular simple model
- modeling and analyzing dynamic effects with streams
- compositional approach to global modeling
- applications and tools
- conclusion

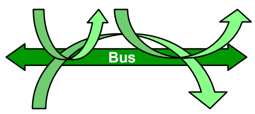
Introduction

- **MpSoC platforms are heterogeneous**
  - **components**
  - **networks**
  - **communication**
  - **scheduling (static, event, timing)**
  - **...**
- **complex dependencies and dynamic changes threaten design robustness**
- **verification is increasingly difficult and cannot easily capture all effects of concurrency**
- **problems well known from distributed real-time systems**

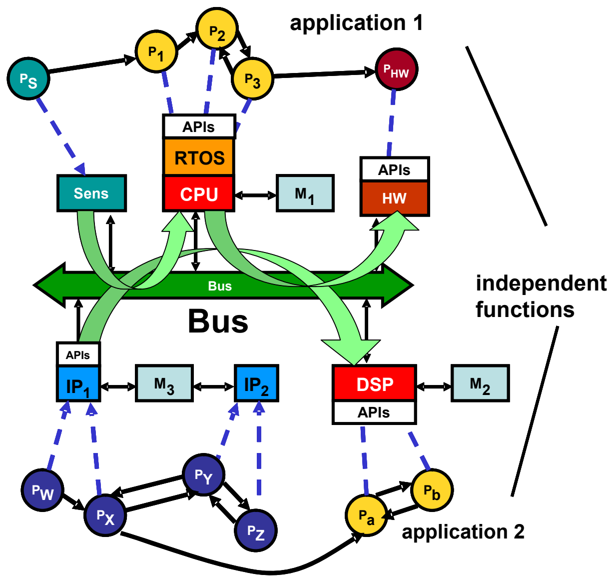
Example: Automotive

- non-functional dependencies of different subsystems – problem grows with system size
  (e.g. networks replacing buses)





feedback

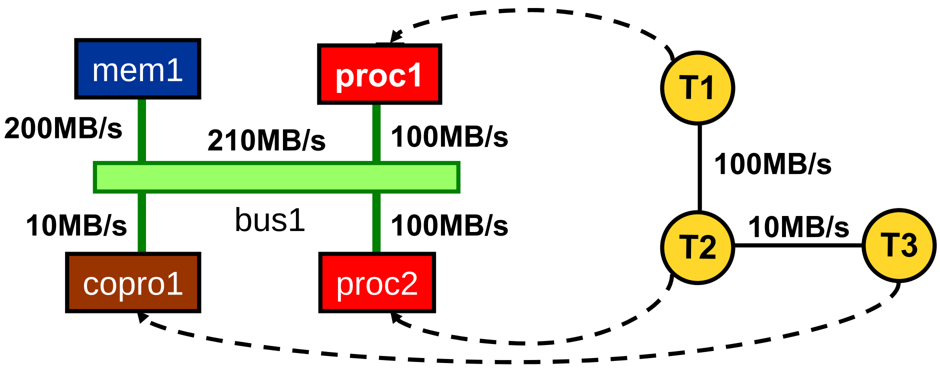


Modeling requirements

- optimization requires appropriate modeling
- simulation models
  - (detailed) HW behavior models
  - currently used in simulation based design space exploration
  - simulation time consuming – constrains optimization
  - executable code often not available at architecture design time
  - modeling flexibility requirements (“slack”) is difficult
- non-executable models for optimization
  - capture abstract resource load, timing relation and dependencies
  - various model semantics including models with interval and stochastic properties

A popular simple model

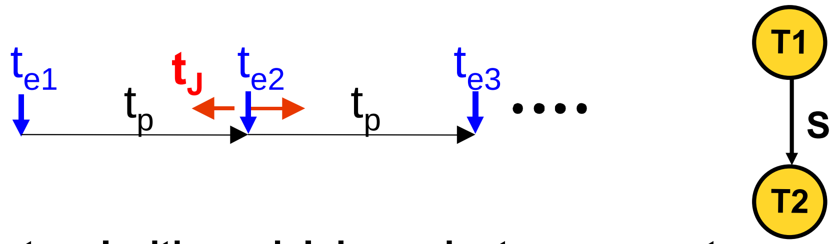
- reduction of dynamic effects to average or integral values



- allows application of weighted graph algorithms → *fast*
- frequently used in optimization tools
- no executable specification required
- does not reflect dynamic effects of transient loads, jitter, deadlines, buffer memory

Modeling dynamic effects with streams

- replace discrete signal values by event streams S



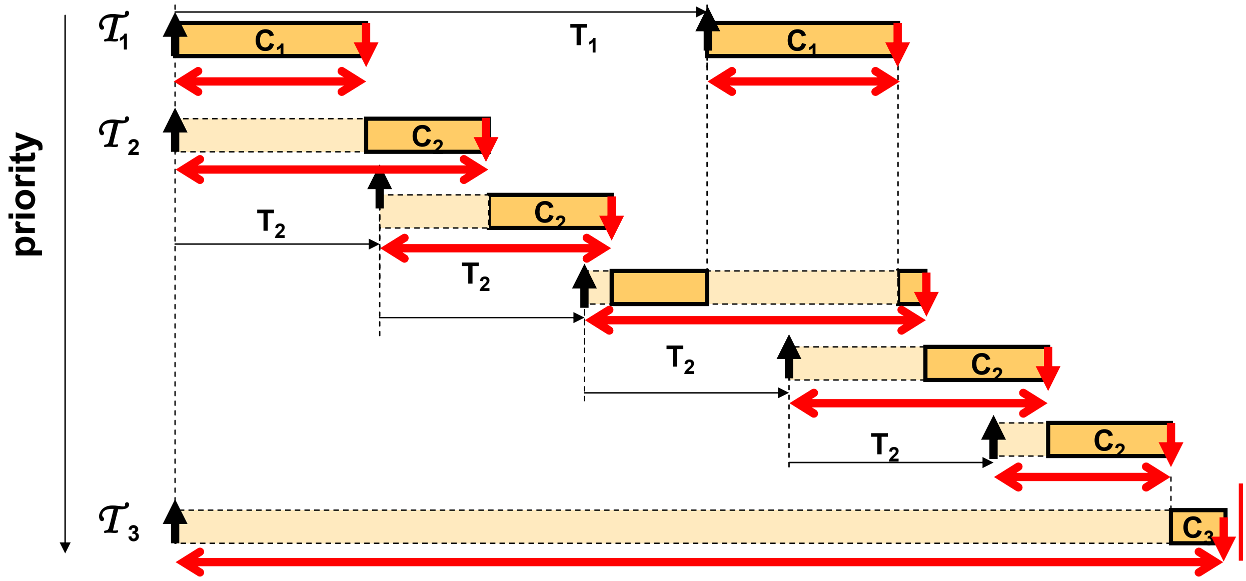
- S is tupel with model dependent components period, minimum distance, jitter, burst, ...
- standard model used in real-time system analysis
- applicable to processors and communication
- many algorithms available
- successfully used in automotive systems optimization
- commercial tools by Volcano, ETAS, ...

Required stream analysis input

- processes and communication models
  - execution time (interval)
  - communication volume (interval)
  - activation rules (time, event)
  - *dependencies (e.g. task graph, cycles, transactions, ...)*
- component models
  - available performance/bandwidth
  - scheduling strategies (processors and communication)
- objective functions and constraints (for interactive exploration)

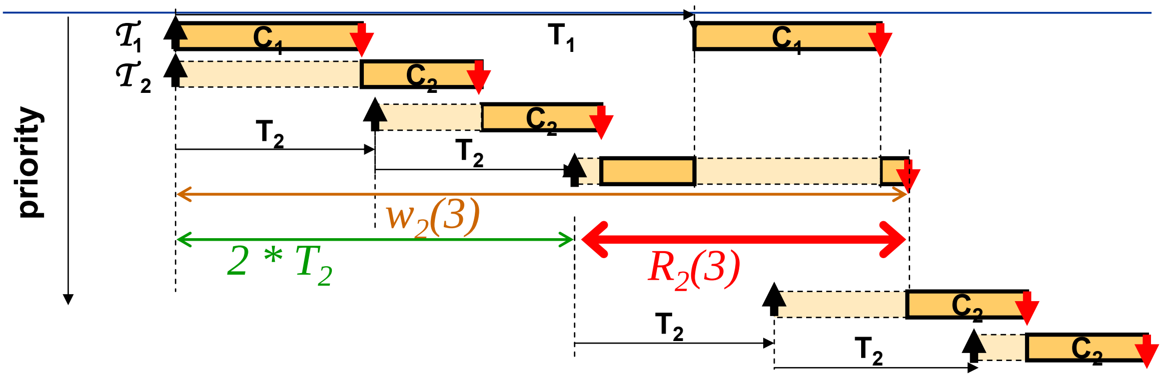
Analysis example: Formal Analysis by Lehoczky

Assume: system with periods T, static priorities and „core" execution times C



Find max total execution time, i.e. worst case response time R

Analysis uses "Busy Window" approach



$$w_i(q) = q\,C_i + \sum_{j \in \mathrm{hp}(i)} C_j \left\lceil \frac{w_i(q)}{T_j} \right\rceil$$

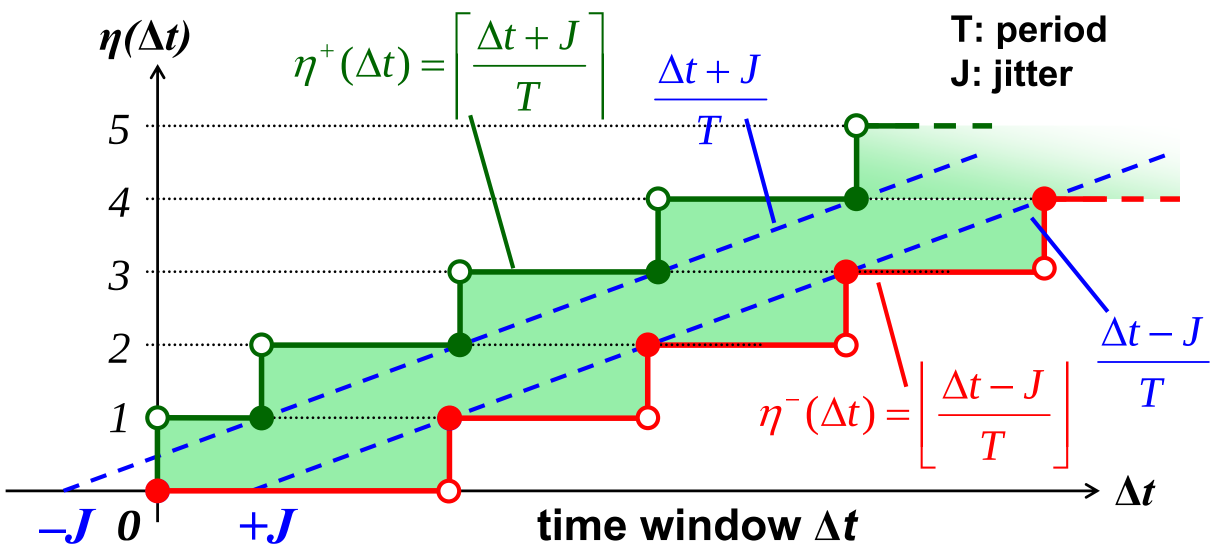
find fix point where equations hold!

$$R_i(q) = w_i(q) - (q-1)\,T_i$$

A generalized approach: Network calculus

- uses *arrival* curves
  - $\eta^+(\Delta t)$ maximum number of activating events occuring in time window Δt
  - $\eta^-(\Delta t)$ minimum number of activating events occuring in time window Δt
  - d^- minimum event distance - limits burst density
- processing represented by corresponding *service* curves
- used in networking applications
- requires new analysis algorithms → real time calculus (Thiele et al.)

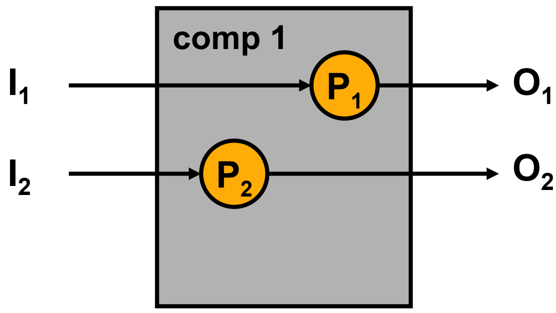
Example: Periodic signal with jitter J

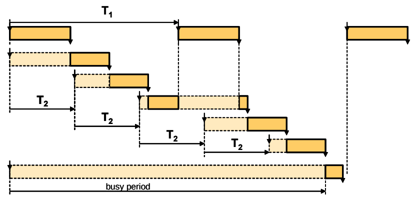


- Event curves *η(Δt)* describe upper and lower bounds of events in time Δt

Component I/O function

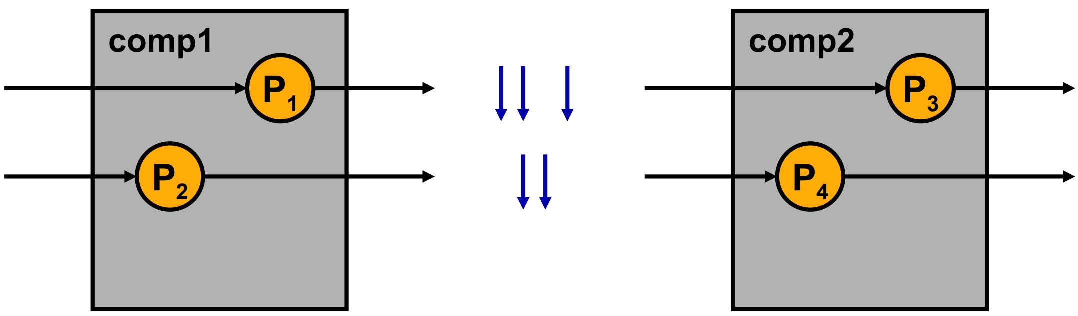
- analysis provides stream I/O function
- input stream interpreted as activation or „register“ (time triggered scheduling)





Compositional approach to global modeling

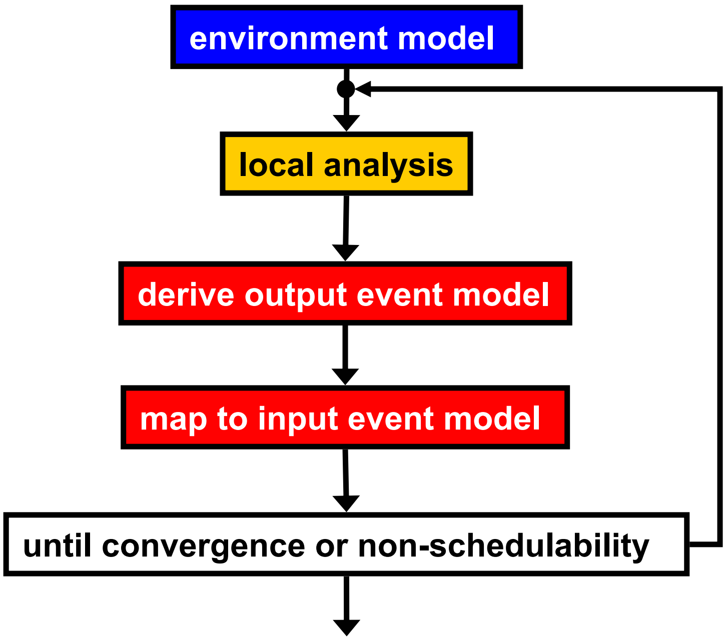
- independently scheduled subsystems are coupled by data flow
- enables analysis of differently scheduled components



⇒ subsystems coupled by **streams**

⇒ coupling corresponds to **event propagation**

Event propagation and analysis principle



- very flexible and composable !

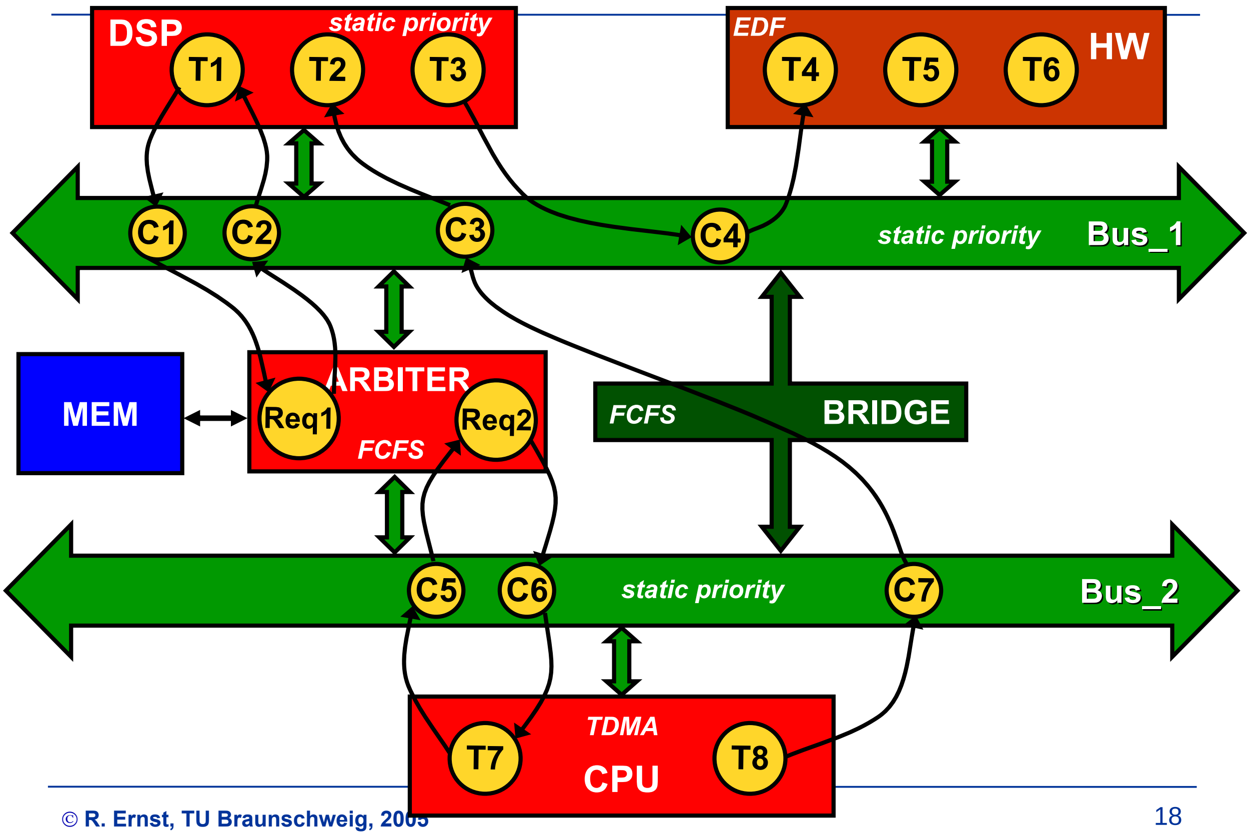
Enhancements

- parameters given as worst case or intervals
- task dependencies: task graphs, cycles
- stream properties may depend on system state
  - system scenarios
- memory access models
- stochastic stream properties
  - analysis using Markov Chains (Eles et al.)
  - very time consuming, new analysis algorithms required

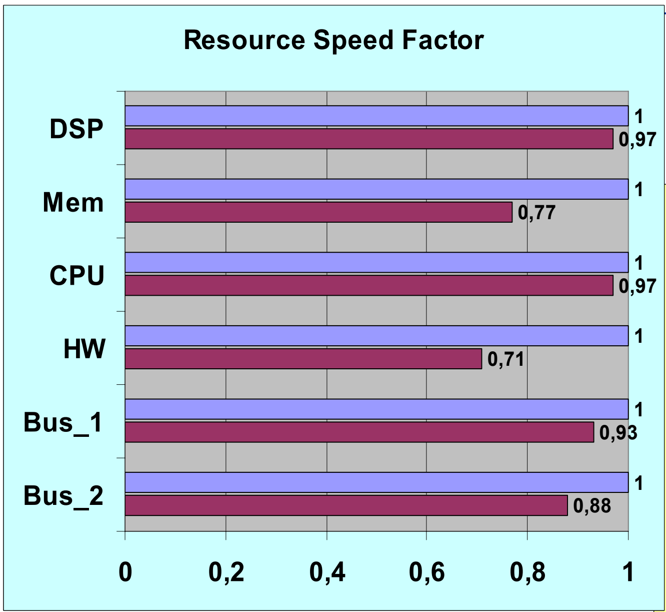
Formal analysis applications

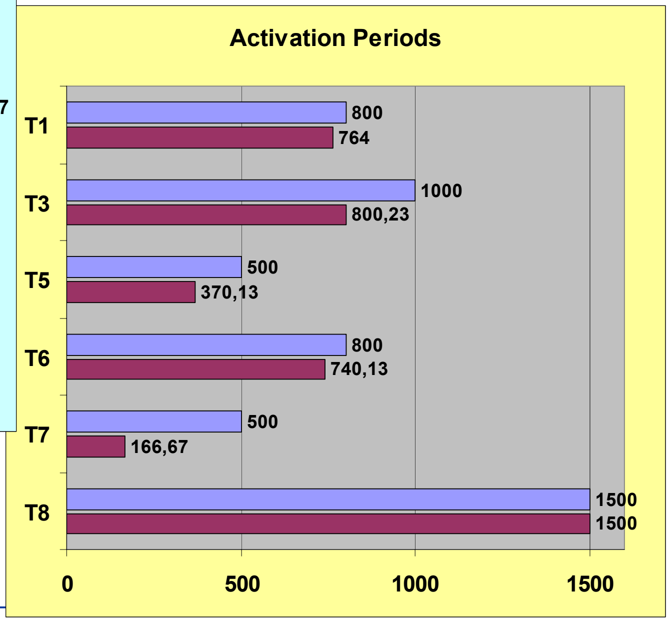
- performance, load, delay, jitter, (buffer) memory analysis (see also MpSoC 2004)
  - covers advanced techniques such as traffic shaping
- design space exploration (very fast!)
- sensitivity analysis (robustness)
- first commercial tools for compositional techniques available (SymTA/S of SymTAVision)
  - currently applied to message passing systems (VW, Bosch, BMW,)

Example Application of Compositional Analysis

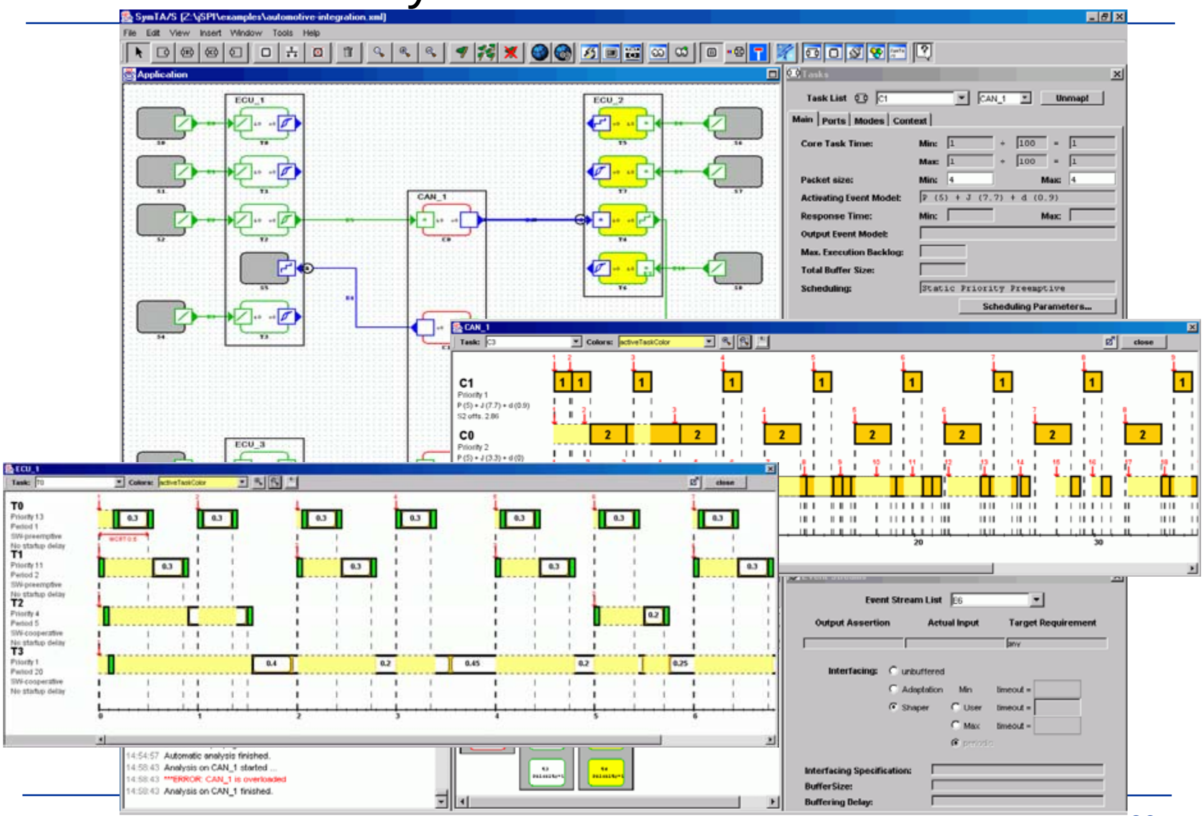


Sensitivity Analysis Results - Example





SymTA/S Screenshot



Conclusion

- event stream models are a powerful basis for fast optimization considering dynamic effects
- scalable via flexible composition rules
- supports sensitivity analysis to identify available "headroom" in a design and detect critical spots
- few data needed that are typically available at system specification
- first commercial tools available