Formal Architecture Analysis in Communication Centric MpSoC Design

R. Ernst

TU Braunschweig

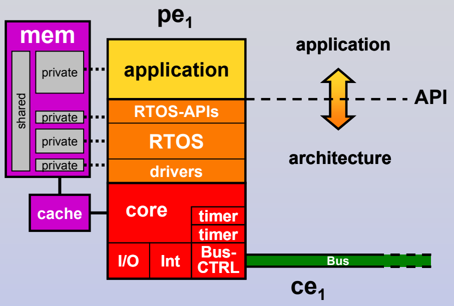
Overview

- communication centric design and systems integration
- formal heterogeneous architecture analysis – holistic and hierachical approaches
- event models and interfaces for iterative analysis
- example
- interactive optimization
- conclusion

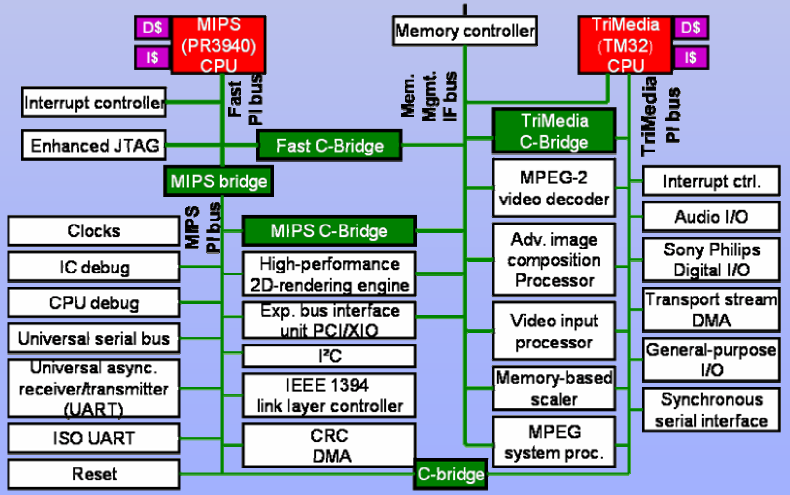
MPSOC architectures are heterogeneous

- heterogeneity resulting from
  - hardware and software component specialization
  - reuse

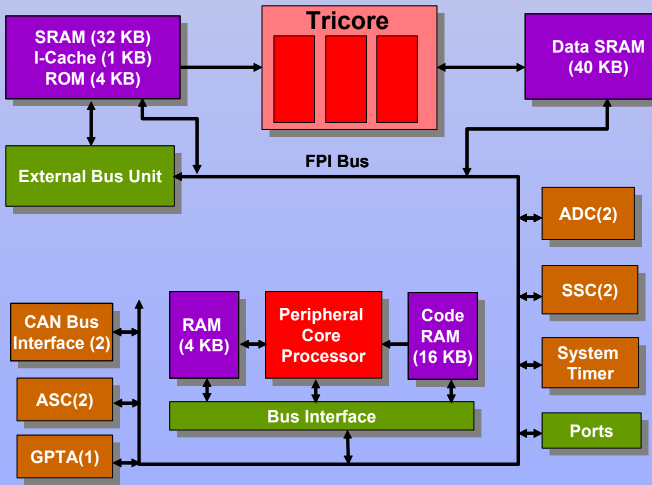
multilayered SW



Philips VIPER (consumer)

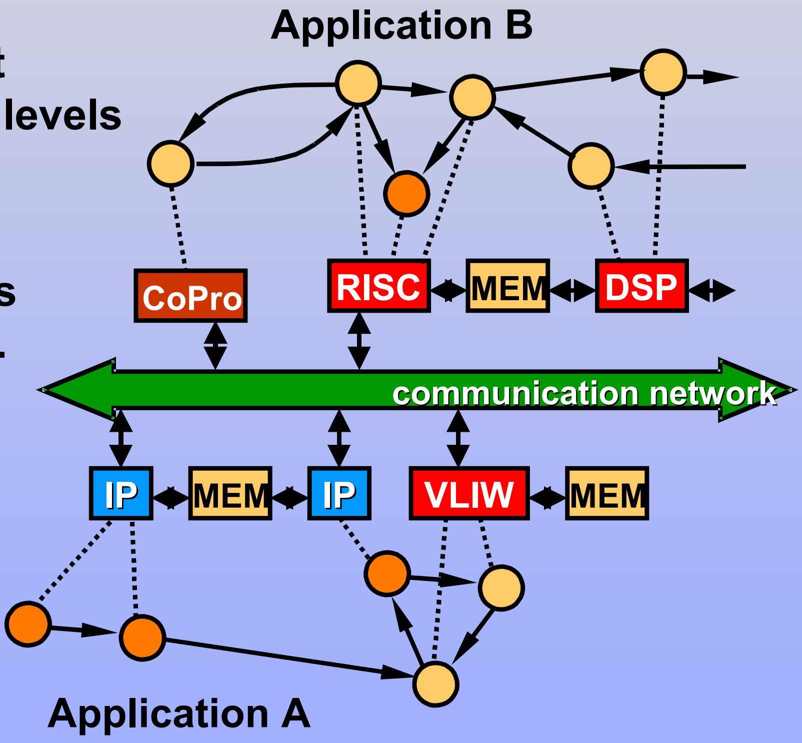


TriCore 1775 (automotive)



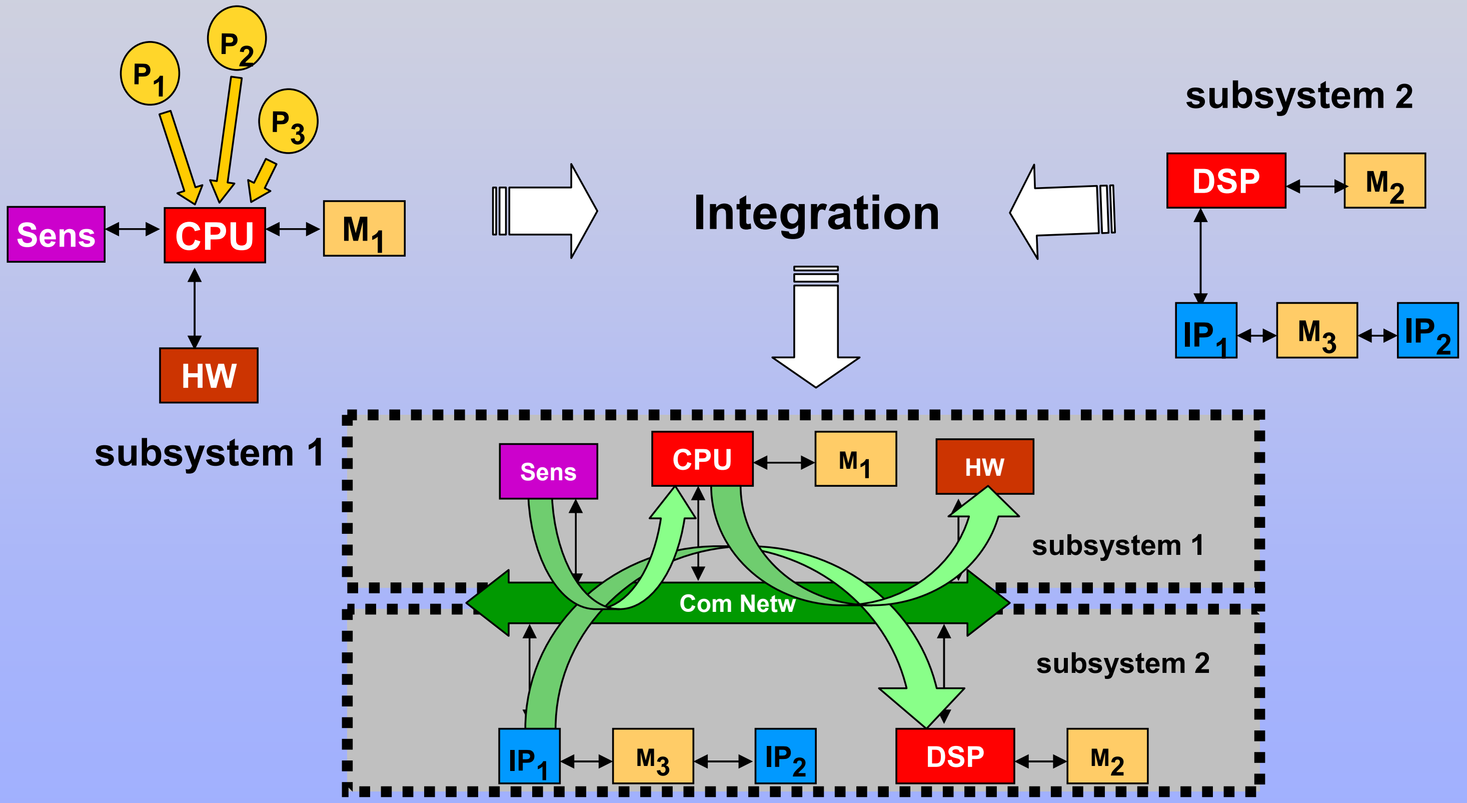
Communication Centric Design

- communication network as a backbone for systems integration
- upcoming design trend (ITRS 2001)
- state of the art:
  - off chip: busses w. different protocols and performance levels
  - on chip: Proprietary or standard buses with bridges AMBA, Sonics, Æthereal, ...
  - future: multi-stage networks on-chip as well as off-chip (PCI Express)

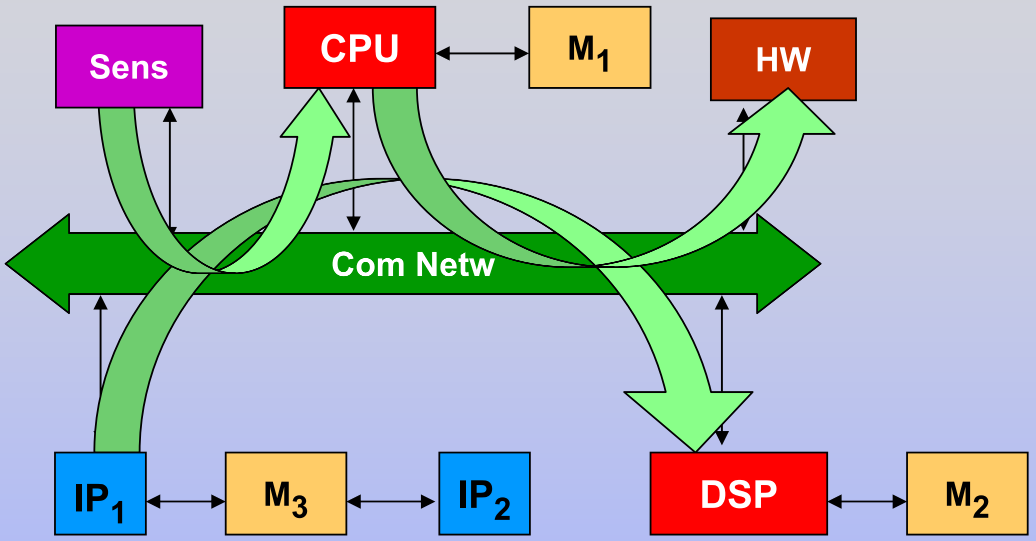


Design as integration problem

- Communication centric design is to a large extend an integration problem



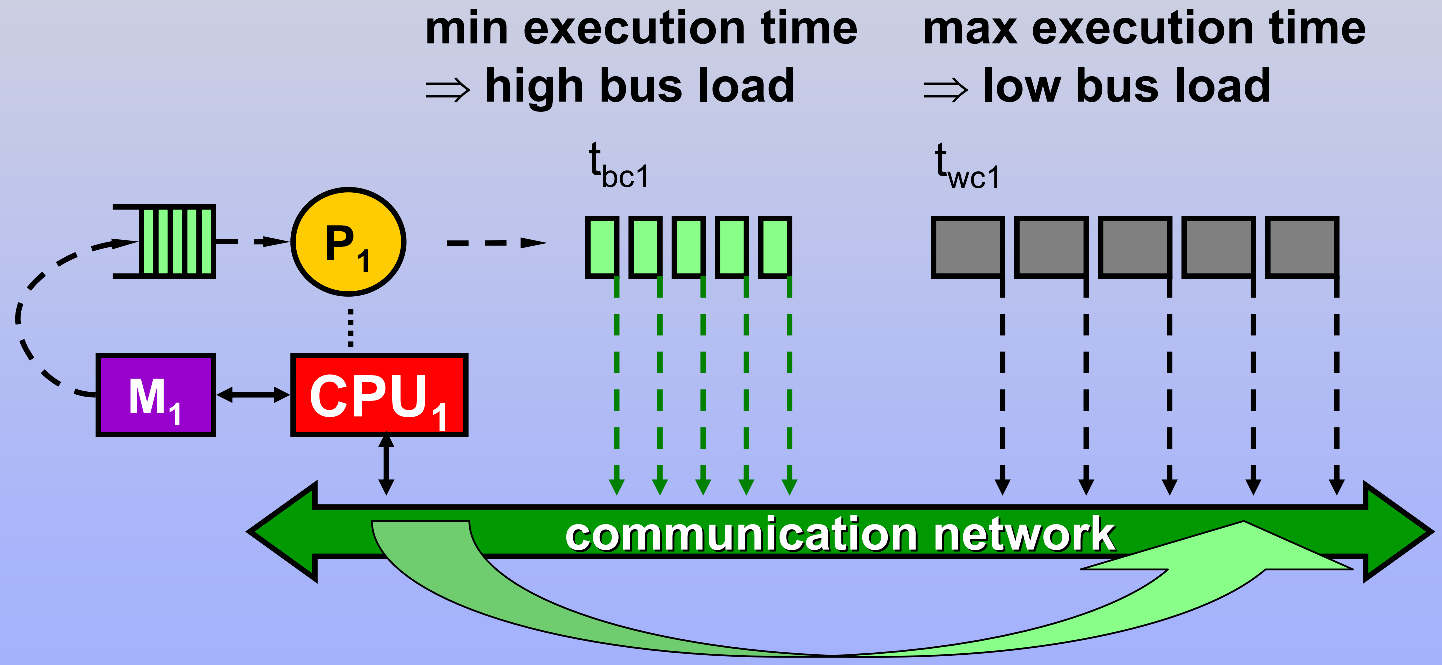
Complex run-time interdependencies



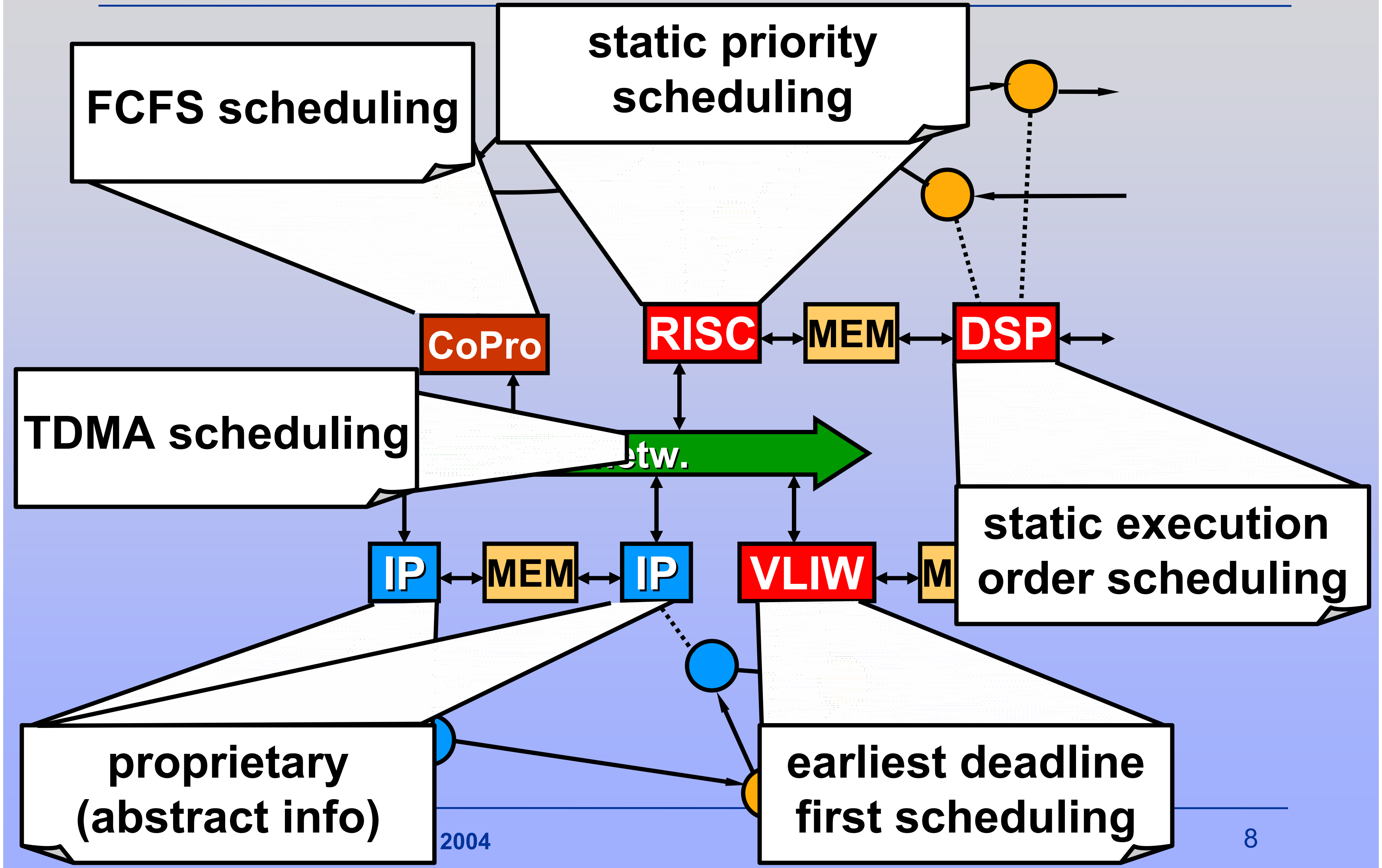
- run-time dependencies of independent components via communication
- influence on timing and power

Interdependency example

- complex non-functional interdependencies
- complex system corner cases

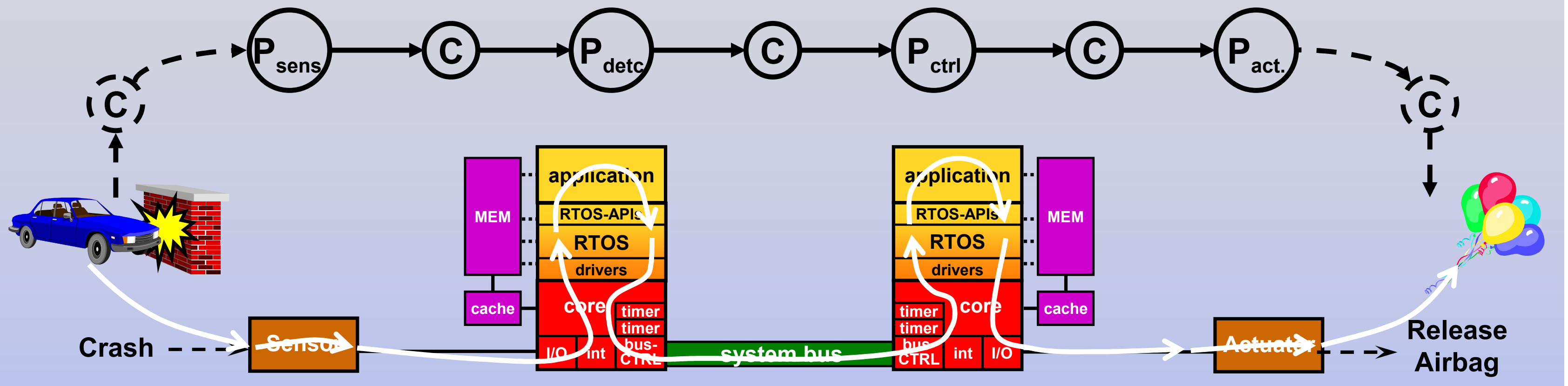


Heterogeneous resource sharing - example



FCFS scheduling
static priority scheduling
CoPro
RISC
MEM
DSP
TDMA scheduling
etw.
static execution order scheduling
IP
MEM
IP
VLIW
M
proprietary (abstract info)
earliest deadline first scheduling

Complex performance objectives and constraints



Reaction time of airbag after crash ?

$$\underbrace{t_{crash} + t_{sens}}_{\text{physical delay}} + t_{csens} + t_{detc} + t_{fbus} + t_{ctrl} + t_{cact} + \underbrace{t_{act} + t_{airbag}}_{\text{physical delay}}$$

Architecture analysis - state of the art

- current approach: target architecture co-simulation, performance simulation
- simulation challenges
  - identification of *system* performance corner cases
    - different from *component* performance corner cases
    - complex phase and data dependent “transient” run-time effects w. scheduling anomalies
    - target architecture behavior unknown to the application function developer (cp. functional HW test)

    ⇒ test case definition and selection ?
  - simulation of incomplete application specifications ?
    - early design space exploration before code implementation is available

Alternative: Formal approaches

- conservative design
  - uses TDMA and synchronization strategy
  - assigns fixed communication bandwidth share to individual subsystems
  - results in decoupling
  - can be extended to fault tolerance strategy (TTA)
  - requires system synchronization
  - *paid by latency and timing overhead*
- formal architecture analysis
  - many results known from real-time analysis
  - already commercial tools for *single* ECU analysis available
  - needs research to extend to larger heterogeneous scalable networks
  - excellent research area with obvious practical impact

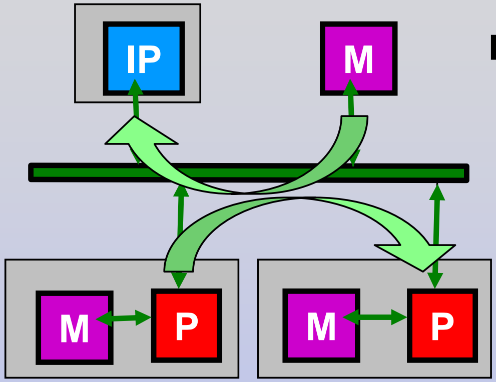
Formal performance analysis

- formal techniques known for individual components and subsystems (RMS, static scheduling etc.)
- heterogeneity is problem
- here focus on system timing
  - power minimization must regard performance expectations
  - most power minimization techniques (supply voltage scaling, threshold voltage scaling, power down modes) are based on run time data

System performance analysis approaches

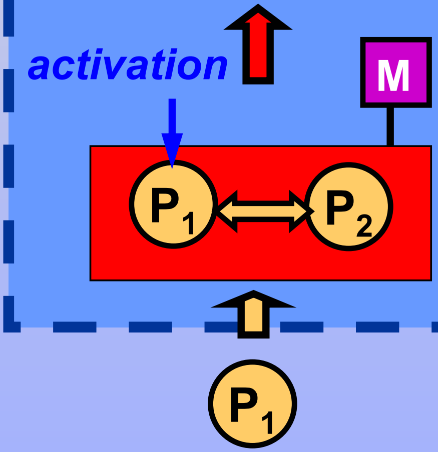
- global approach
  - analysis scope extension to several subsystems
- flow based hierachical approach
  - global flow analysis combined with local scheduling analysis

Performance model structure



model type	influenced by
system model	• communication pattern • shared memory access • environment model

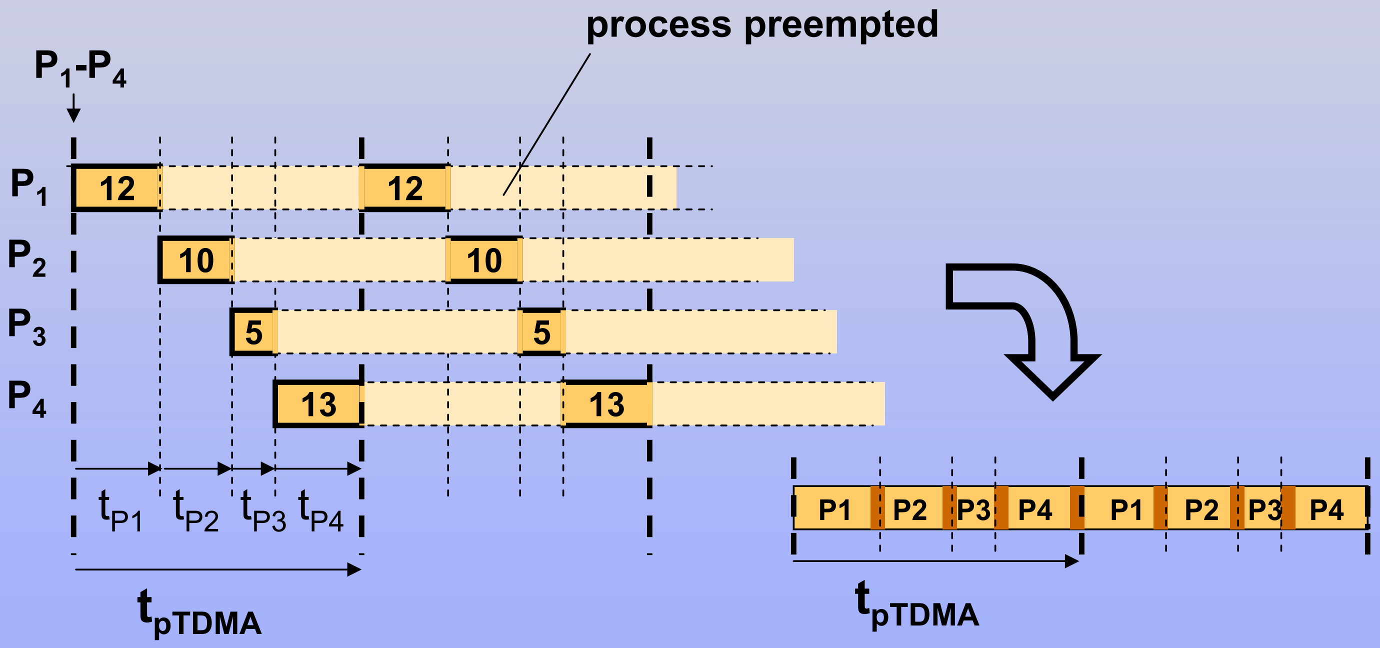
single component real-time analysis



model type	influenced by
component & communication execution model	• resource sharing strategy • process activation • component state (caches)
process execution model	• execution path – data dependent • path execution – arch. dependent • communication – data & arch. dep.

Example: Time division multiple access (TDMA)

- periodic assignment of fixed time slots
- applicable to processing and communication



TDMA example

TDMA

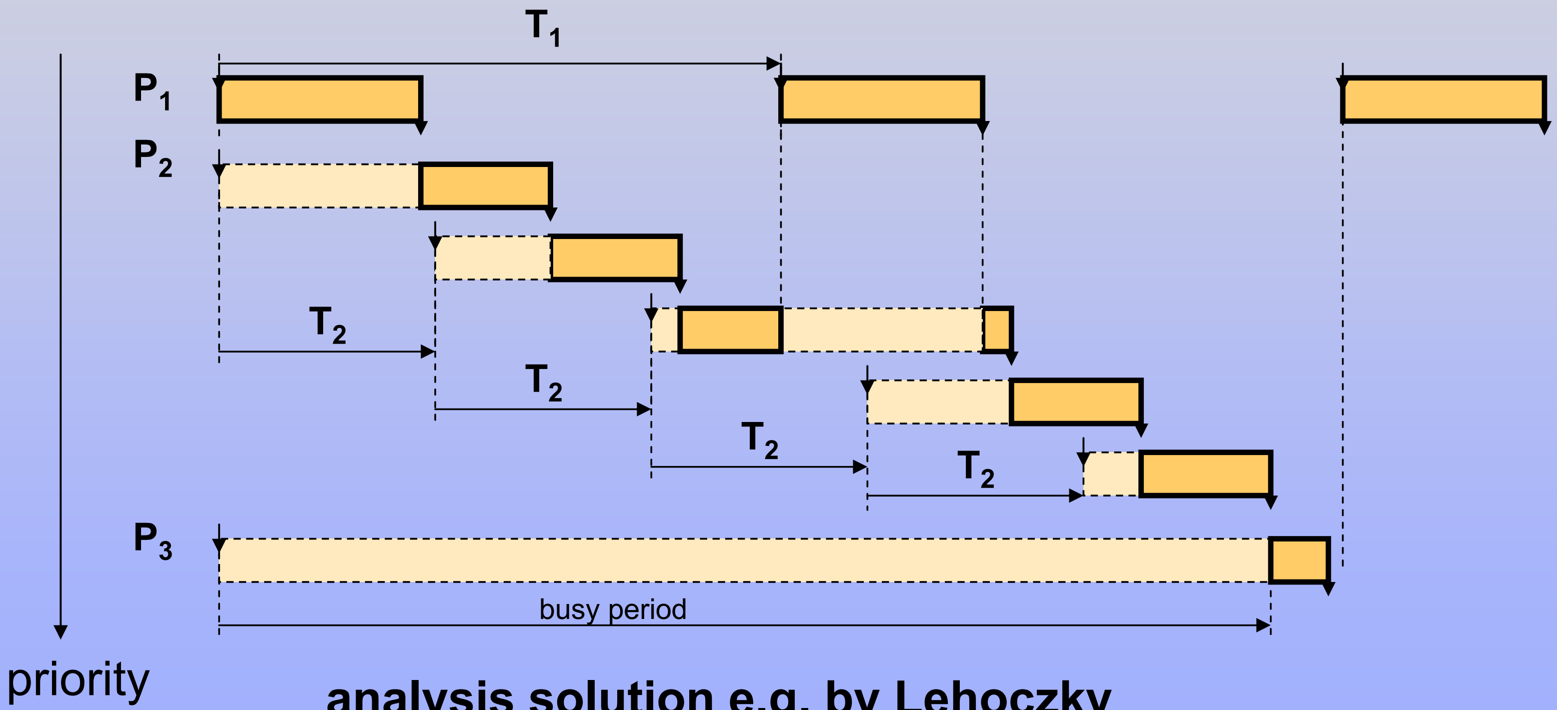
- predictable and independent performance down scaling allows to merge individual solutions

$$t_{peTDMA}(P_i, pe_i) = \left\lfloor \frac{t_{pe}(P_i, pe_i) - t_{csw}}{t_{Pi}} \right\rfloor \cdot t_{pTDMA} + t_{pe}(P_i, pe_i) \bmod t_{pi}$$

- can be applied to complete systems (Giotto)
- problems
  - low resource utilization
  - extended deadlines

Ex: Static priority with arbitrary deadlines

- complex execution sequence - may create output bursts
- found in communication scheduling and multiprocessing

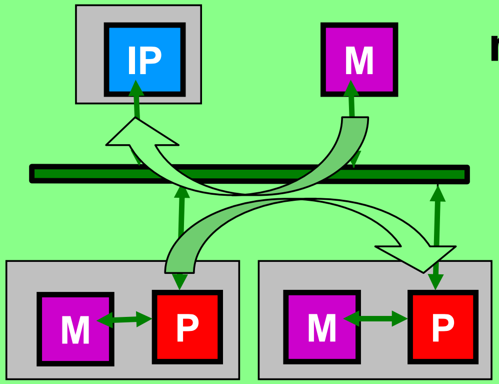


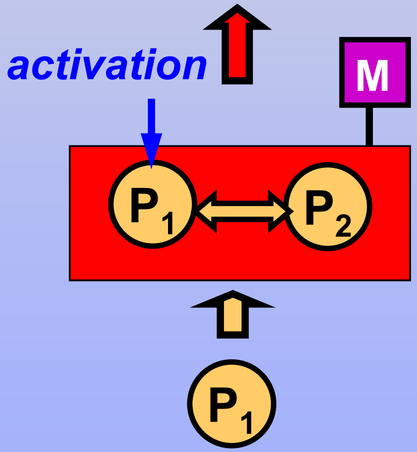
analysis solution e.g. by Lehoczky

Single component RTA summary

- numerous formal approaches for local analysis known
  - time or event driven
  - with context switch, blocking, overlapping process activations, ...
  - examples: TDMA, static priority

Performance model structure





model type	influenced by
system model	• communication pattern • shared memory access • environment model
component & communication execution model	• resource sharing strategy • process activation • component state (caches)
process execution model	• execution path – data dependent • path execution – arch. dependent • communication – data & arch. dep.

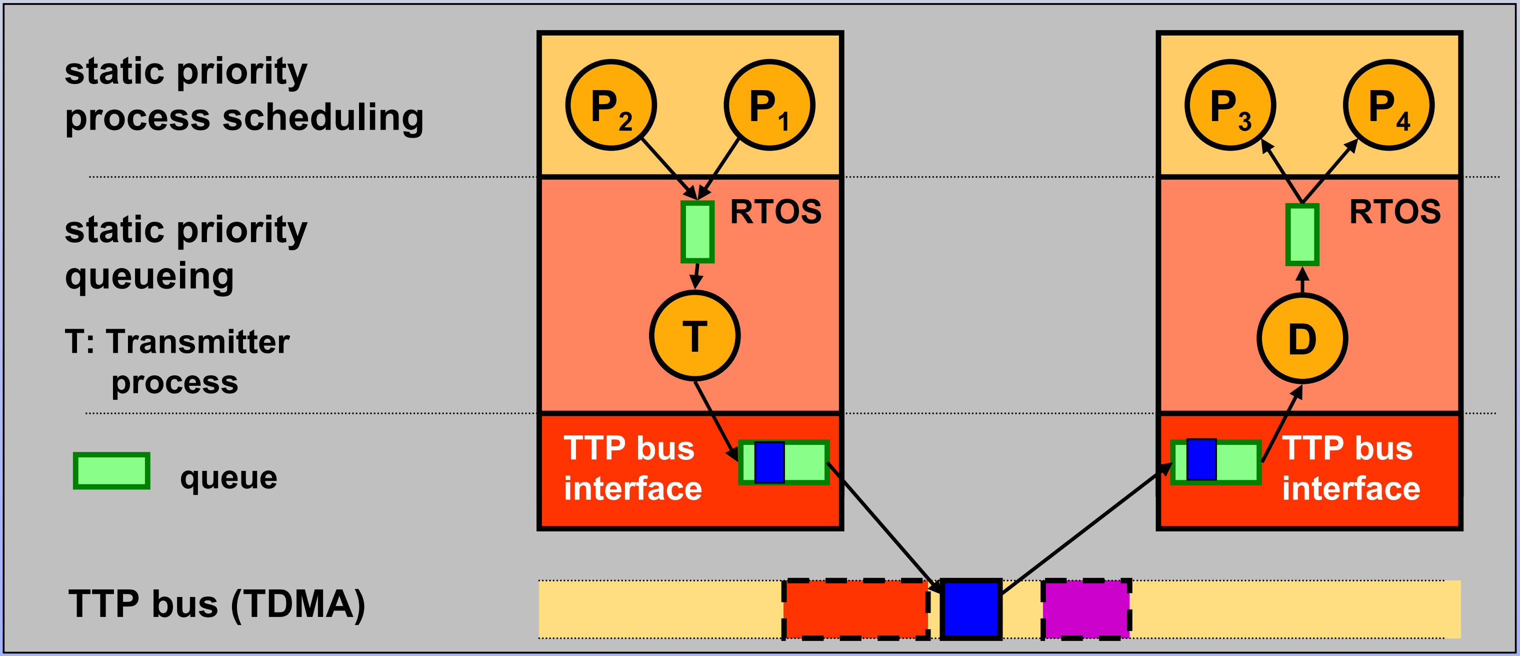
global real-time system analysis

System performance analysis approaches

- global approach
  - analysis scope extension to several subsystems
- flow based hierachical approach
  - global flow analysis combined with local scheduling analysis

Analysis scope extension

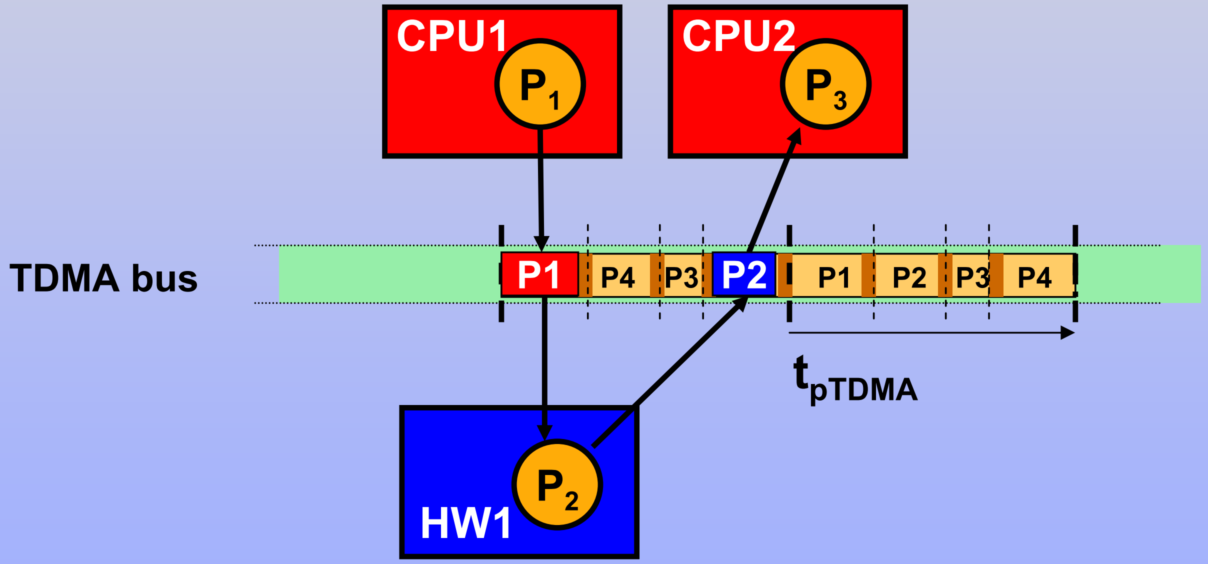
- coherent analysis („holistic“ approach)
- example: Tindell 94, Palencia/Harbour 98, Pop/Eles (DATE 2000, DAC 2002, …): TDMA + static priority – automotive applications



- problem: scalability

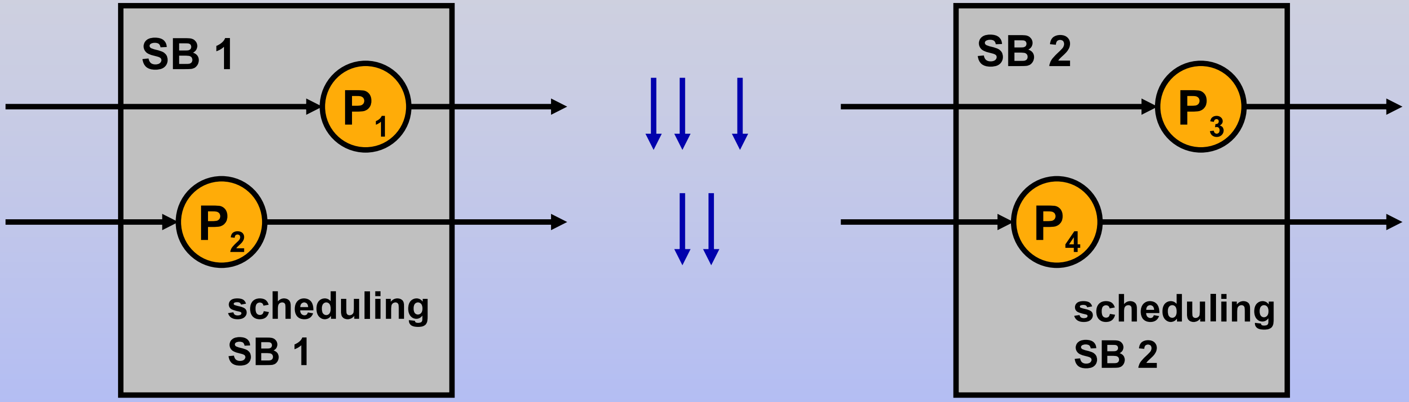
Analysis scope extension - 2

- benefit: scope extension can take global system knowledge into account
- example: using dependency information to detect that P2 can send in the same TDMA round as P1, if $t_{P2} < t_{P3} + t_{P4}$



Hierarchical approach

- independently scheduled subsystems are coupled by data flow

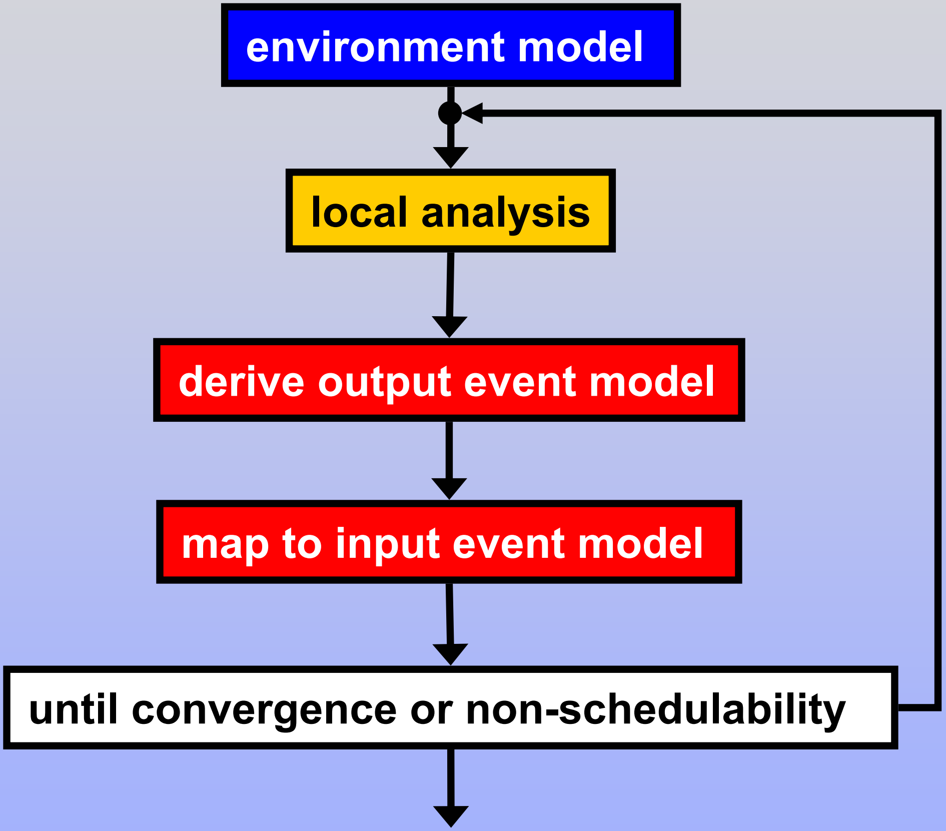


⇒ subsystems coupled by stream of data

⇒ interpreted as activating events

⇒ coupling corresponds to event propagation

Event propagation and analysis principle


environment model
local analysis
derive output event model
map to input event model
until convergence or non-schedulability

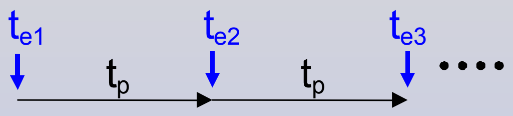
- very flexible and composable !

Challenge: Local model interfacing

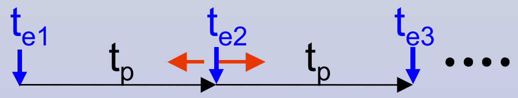
- approaches
  - generalized event model
    - arrival and service curves derived from Network Calculus Chakraborty/Thiele/Gries/Künzli …
    - develop new analysis approaches for these models
  - event stream model adaptation
    - use abstract interface stream properties to couple local analysis

Popular event stream models

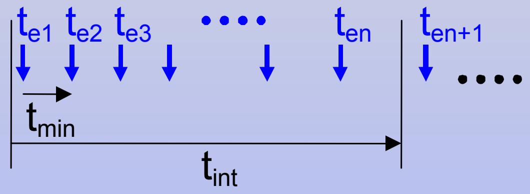
- observation: real-time analysis literature assumes few quasi standard event models at input
  - periodic event stream



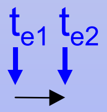
  - periodic event streams with jitter



  - periodic event streams with burst



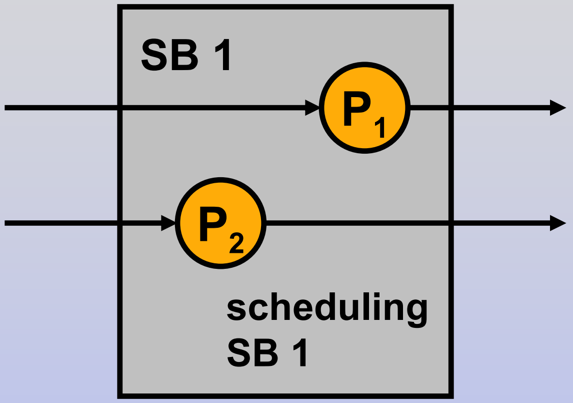
  - sporadic events with minimum event separation

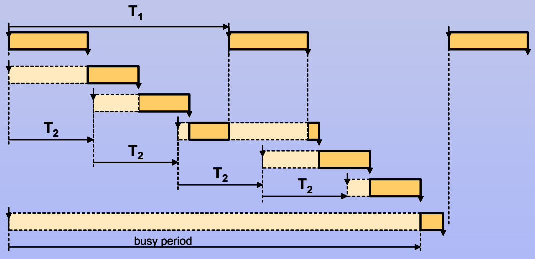


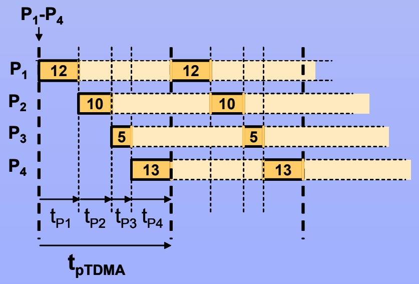
- comparable event models appear at output
- volume of generated events is fixed or interval (data dependent)

Input – output model relation

- any scheduling increases jitter
- jitter grows along signal path
- increasing jitter leads to
  - burst and transient overloads
  - higher memory requirements
  - power peaks

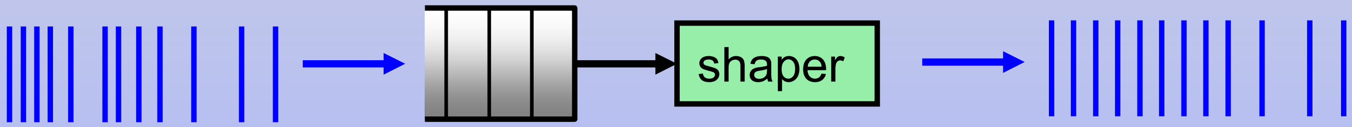




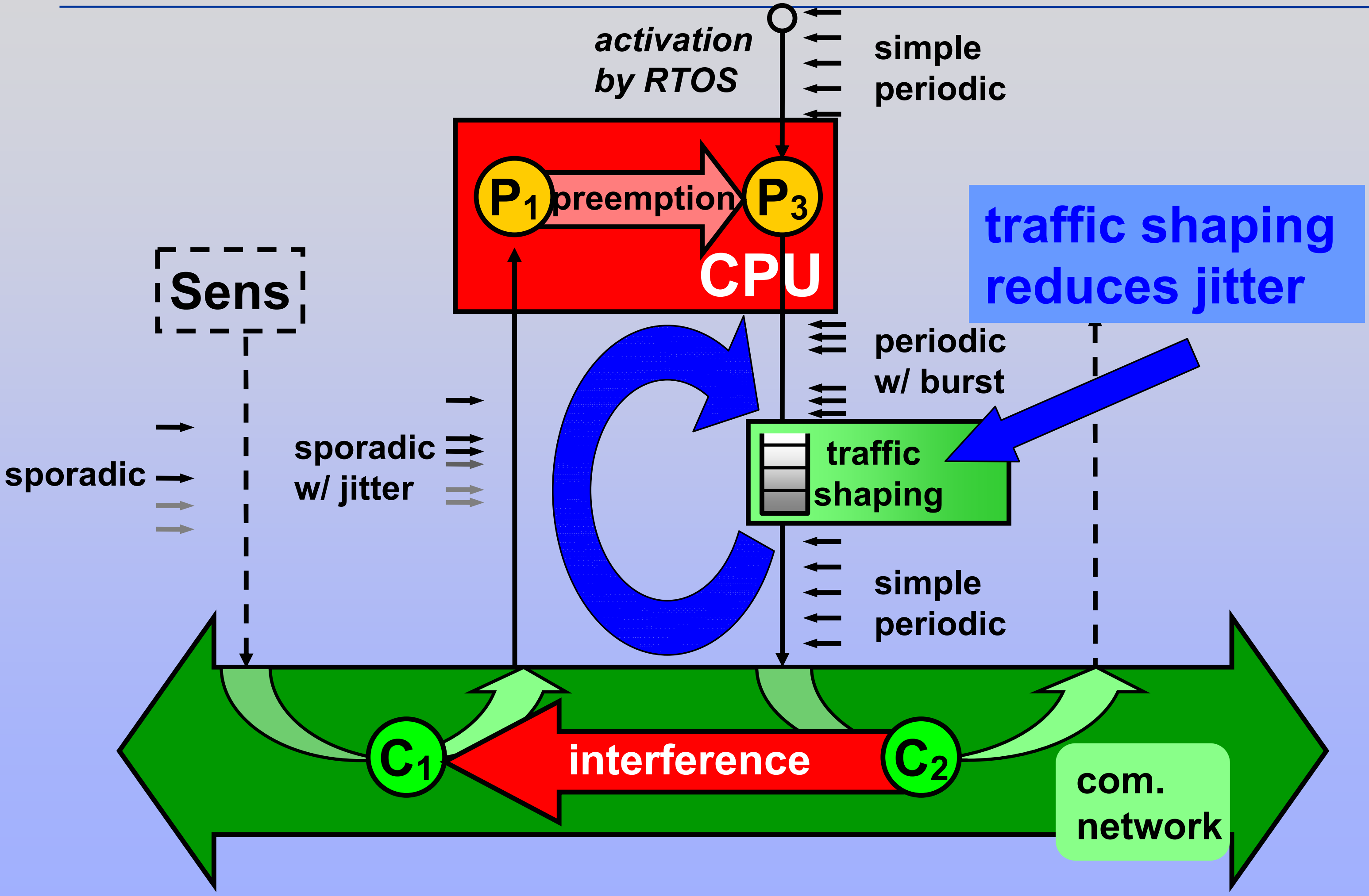


Reducing transient load in design

- synchronization
- minimum event separation using „traffic shaping“
- requires buffering and possibly increases latency

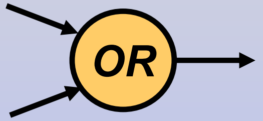


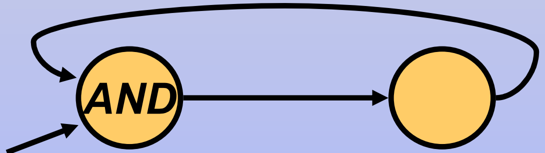
Ex: Traffic shaping to remove dependency cycle

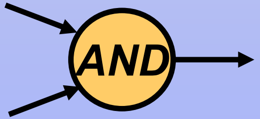


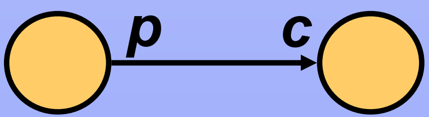
RTA event stream models are not sufficient

- event model transitions needed to couple different subsystems and scheduling domains
- more complex activation models needed
  - defined by application models
  - OR activation
    - typical in event driven systems
  - AND activation and loops
    - typical for data flow models
    - unusual for real-time system *models* (acyclic task graphs)
  - multi rate activation
    - typical for data flow models
    - unusual for real-time systems





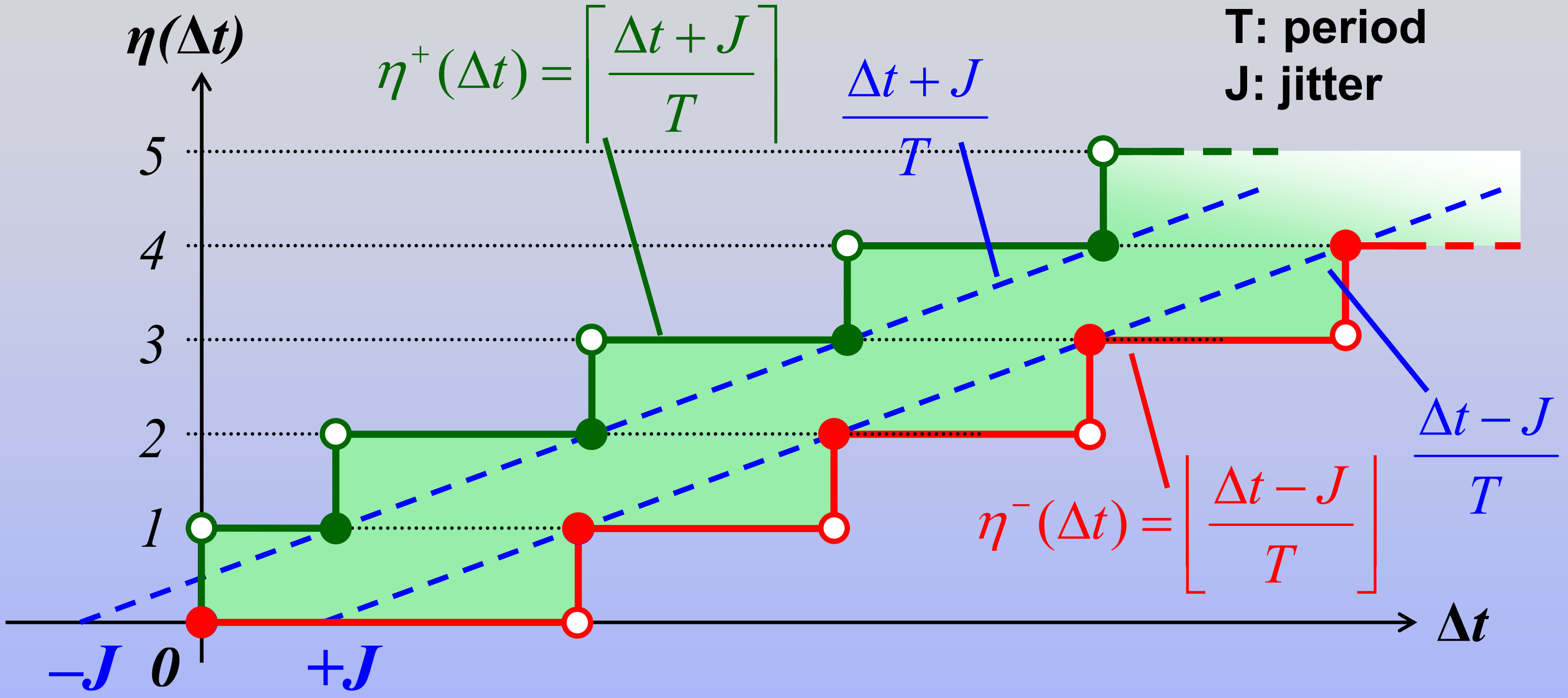




Event model interfacing

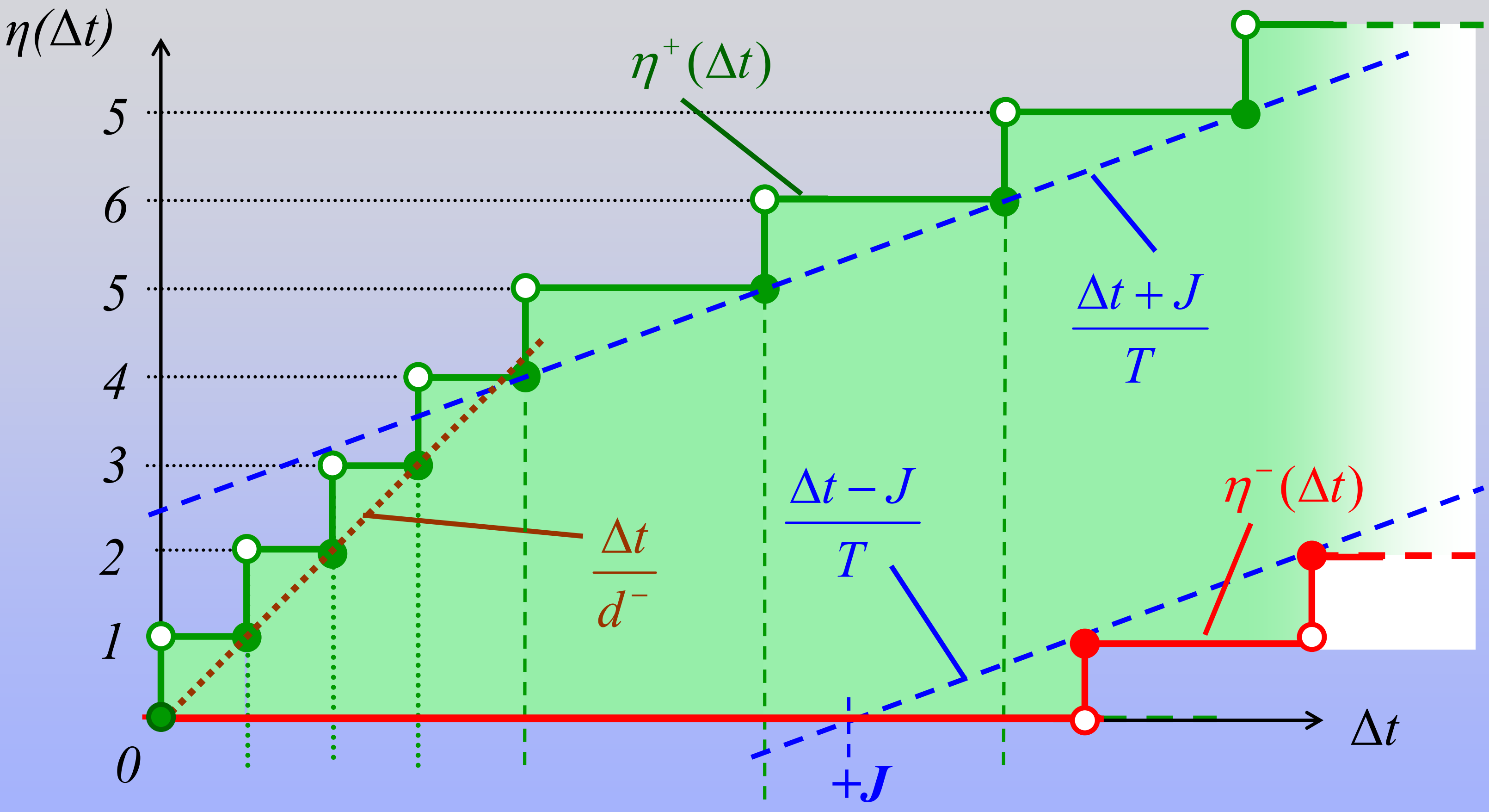
- idea: use network calculus + additional information as intermediate mathematical formalism
- arrival curves of network calculus
  - $\eta^+(\Delta t)$ maximum number of activating events occuring in time window Δt
  - $\eta^-(\Delta t)$ minimum number of activating events occuring in time window Δt
  - d^- minimum event distance - limits burst density

Example: Periodic signal with jitter J



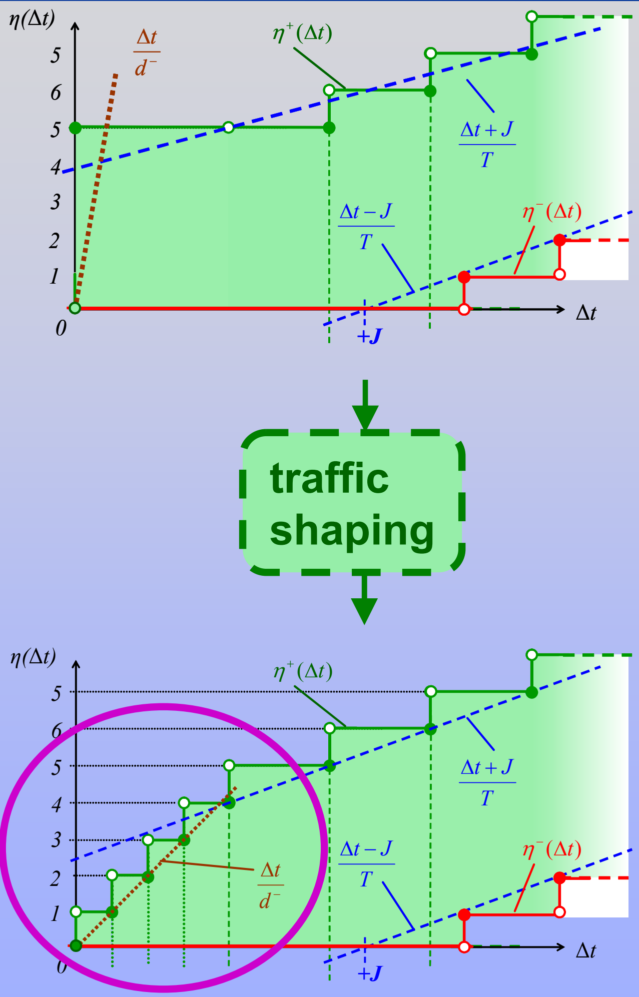
- Event curves *η(Δt)* describe upper and lower bounds of events in time Δ*t*

Example: Periodic signal with burst

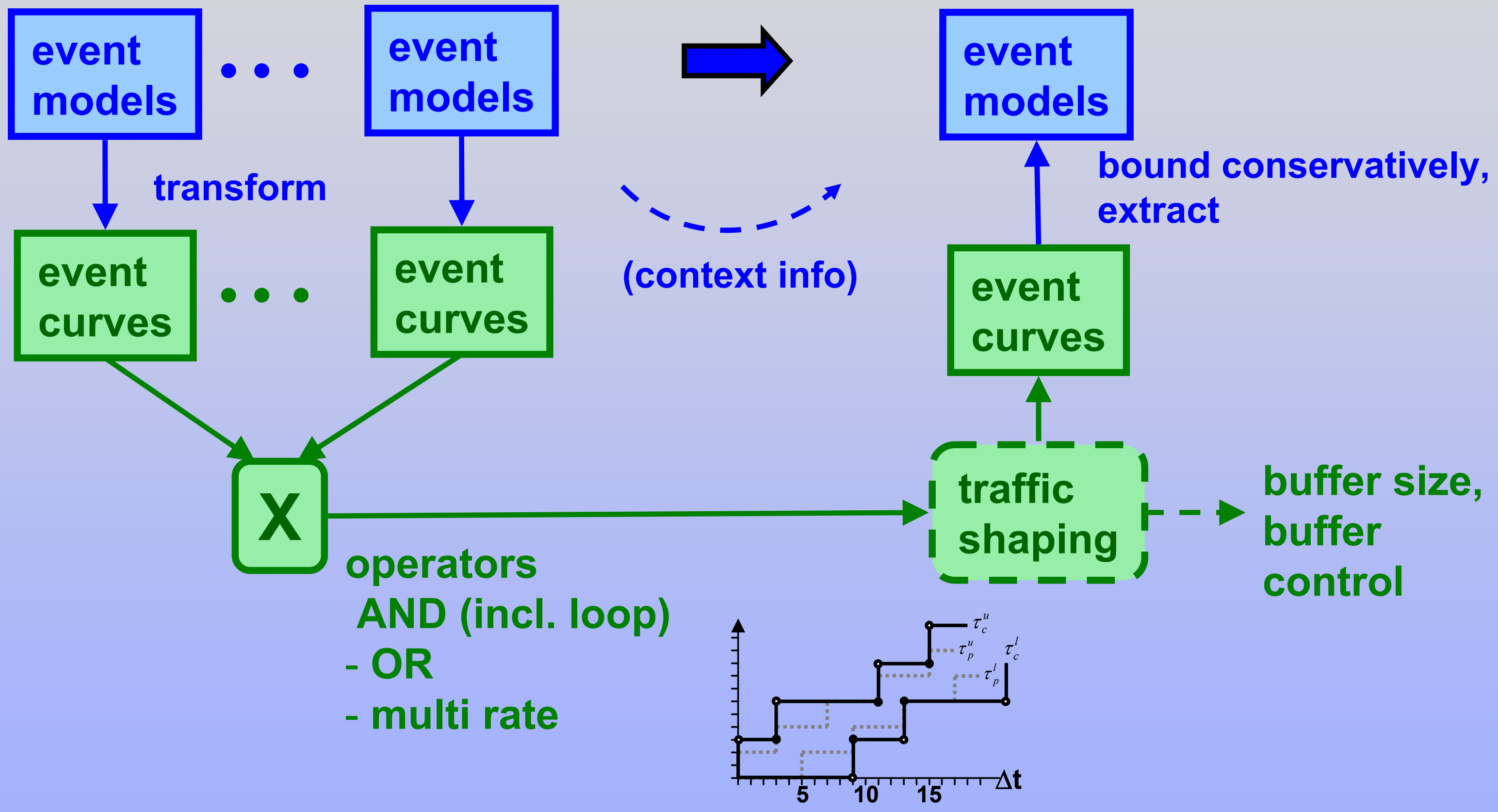


d^- : minimum event distance

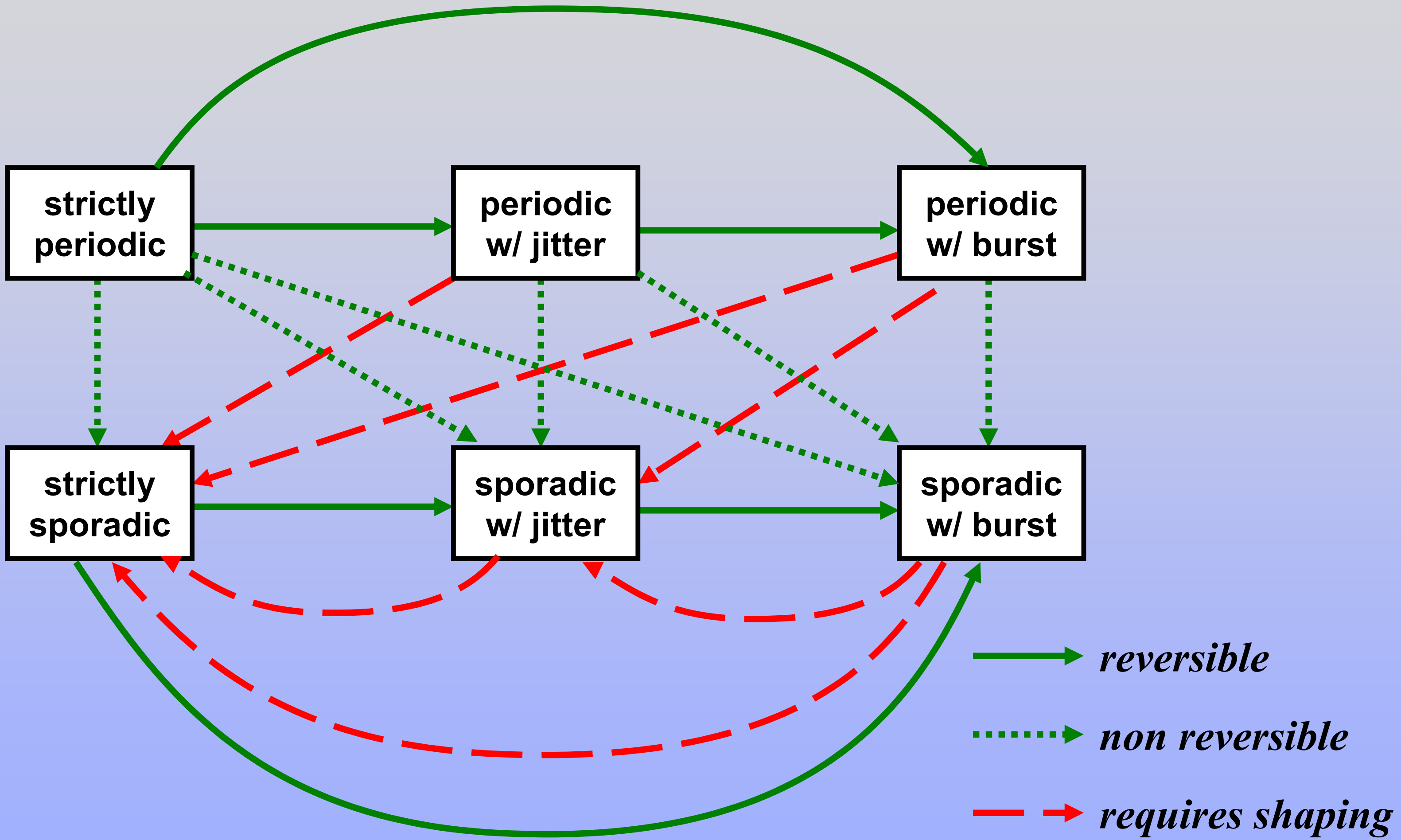
Traffic shaping - example



Event model transformation - principle



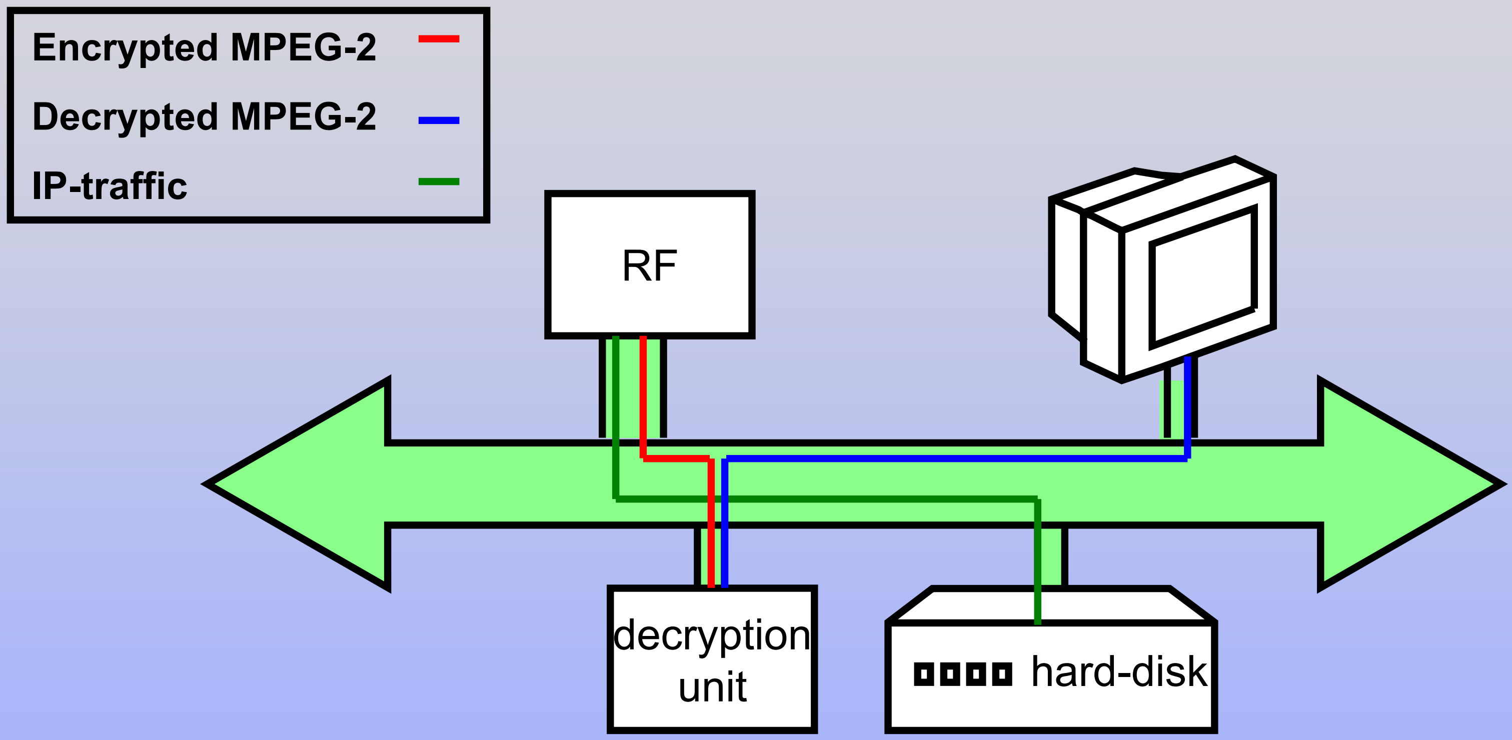
Event models and transformations



Using global dependencies

- „inter frame“ dependencies
  - solutions known from RTA can be applied to iterative global analysis
  - dependent activations are grouped in „transactions“
- „intra frame“ dependencies
  - data dependent data volume or execution time
  - solutions known from RTA can be applied to iterative global analysis
- can be combined leading overall to less conservative analysis results (effect similar to holistic approaches)

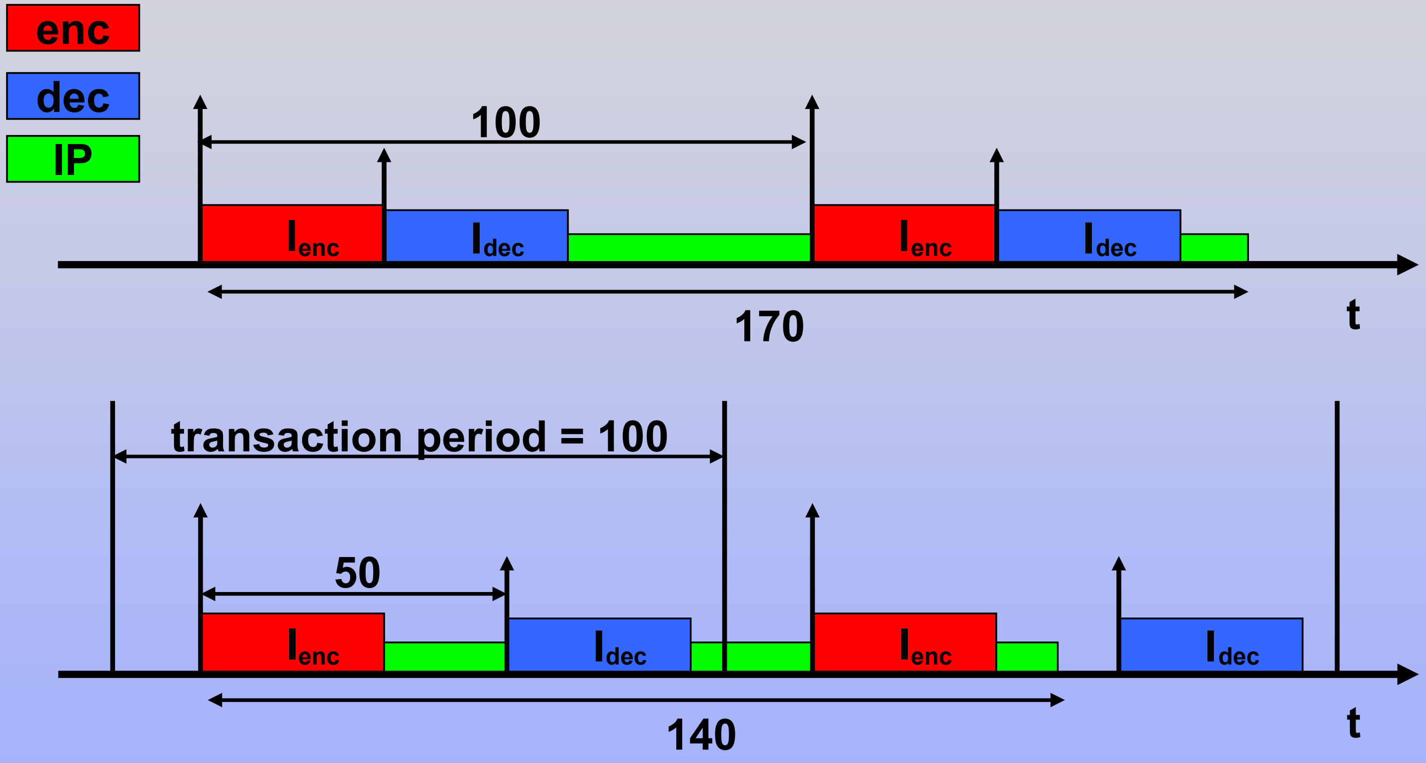
Transaction example



- set top box: decript video + download file via IP

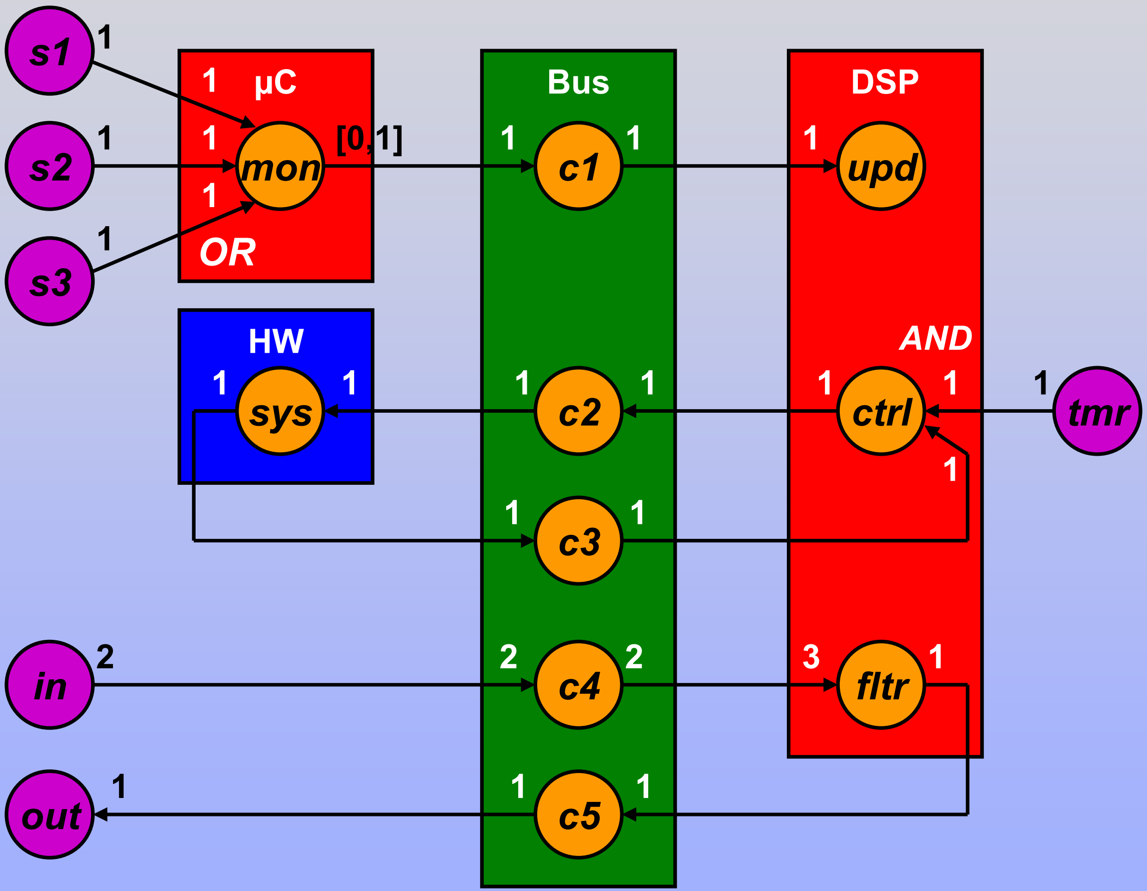
Response time impact

- no transaction considered

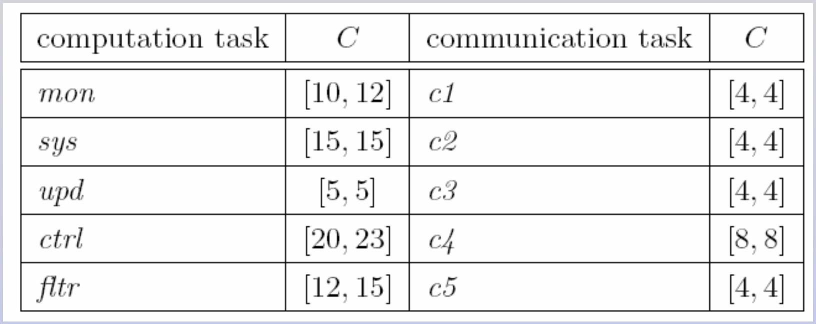


- transaction considered: enc and dec streams are coupled by decryption unit execution time

Putting it all together – a comprehensive example

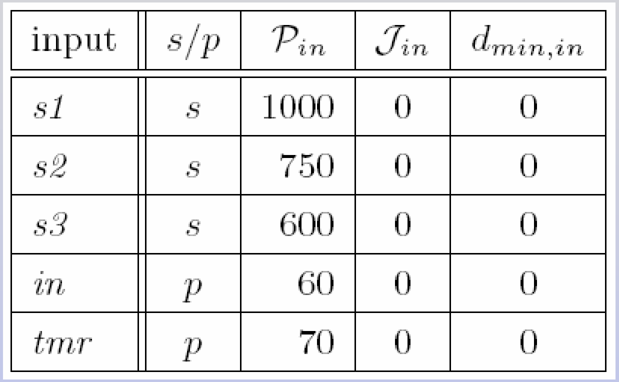


System parameters and constraints



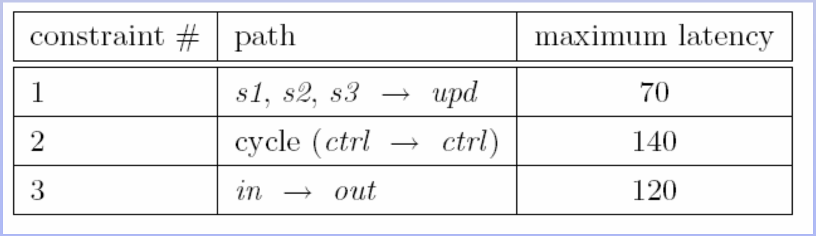
computation task	C	communication task	C
mon	$[10, 12]$	*c1*	$[4, 4]$
sys	$[15, 15]$	*c2*	$[4, 4]$
upd	$[5, 5]$	*c3*	$[4, 4]$
ctrl	$[20, 23]$	*c4*	$[8, 8]$
fltr	$[12, 15]$	*c5*	$[4, 4]$

core execution and communication times



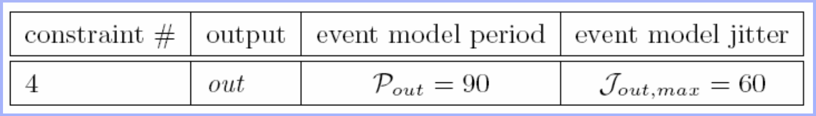
input	s/p	$\mathcal{P}_{in}$	$\mathcal{J}_{in}$	$d_{min,in}$
s1	s	1000	0	0
s2	s	750	0	0
s3	s	600	0	0
in	p	60	0	0
tmr	p	70	0	0

event models at external inputs



constraint #	path	maximum latency
1	*s1, s2, s3* $\rightarrow$ *upd*	70
2	cycle (*ctrl* $\rightarrow$ *ctrl*)	140
3	*in* $\rightarrow$ *out*	120

path latency constraints



constraint #	output	event model period	event model jitter
4	*out*	$\mathcal{P}_{out} = 90$	$\mathcal{J}_{out,max} = 60$

output jitter constraint

Results using the SYMTA/S tool

exp.	*Bus* priorities $\longrightarrow$	*DSP* priorities $\longrightarrow$	constr. 1	constr. 2	constr. 3	constr. 4
1	*c1, c2, c3, c4, c5*	*upd, ctrl, fltr*	45 √	77 √	n/a	119
2	*c1, c2, c3, c4, c5*	*fltr, upd, ctrl*	60 √	107 √	n/a	97
3	*c4, c5, c1, c2, c3*	*fltr, upd, ctrl*	74	130 √	95 √	41 √
4	*c1, c4, c5, c2, c3*	*fltr, upd, ctrl*	60 √	158	111 √	57 √

Static priority assignment
experiments with different *Bus* and *DSP* priorities

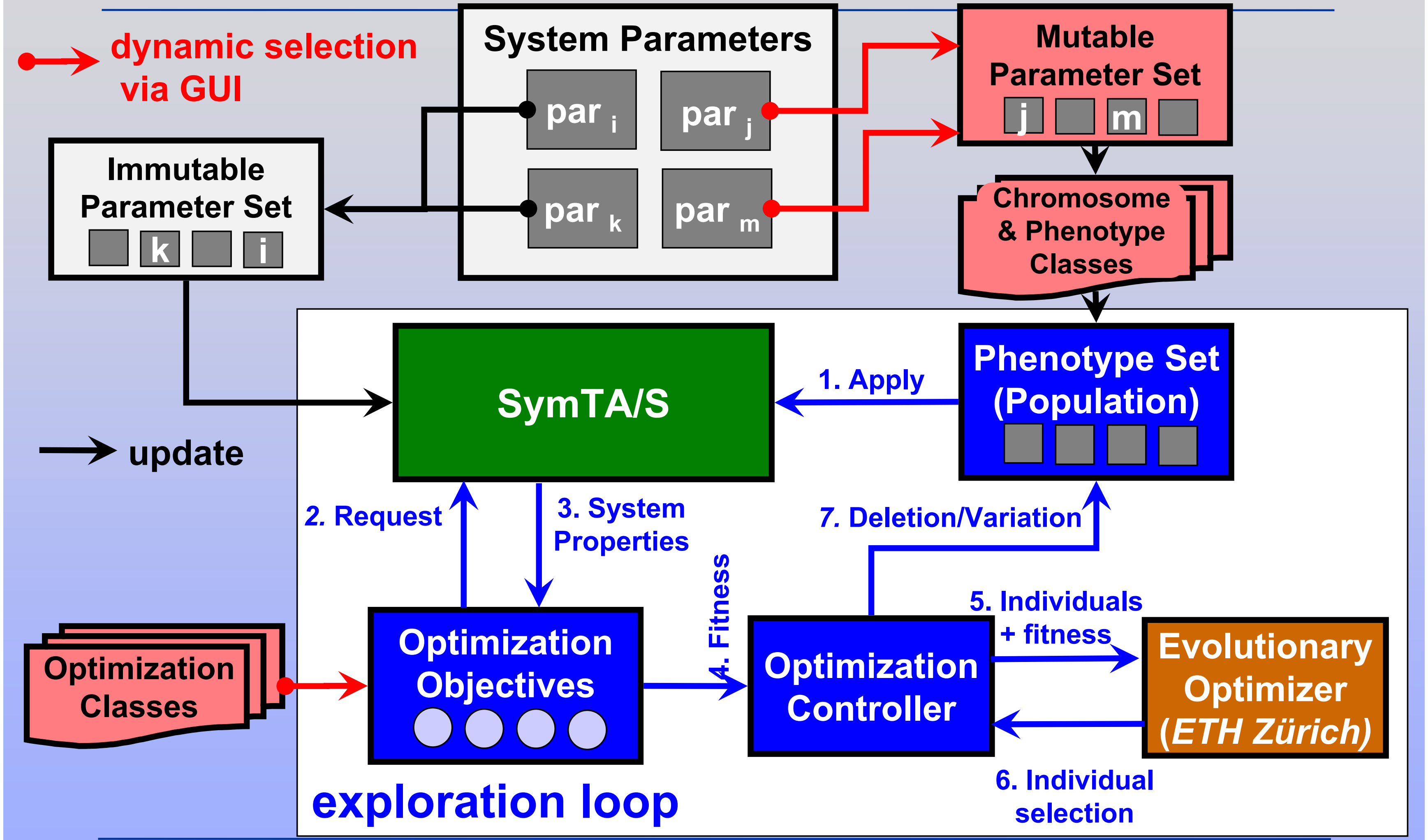
exp.	*Bus* tasks	*DSP* tasks	constr. 1	constr. 2	constr. 3	constr. 4
$4'$	*c1, c4, c5, c2, c3*	*fltr, upd, ctrl*	69 √	124 √	107 √	53 √

Scheduling anomaly
all constraints met with 80% speed of µC
same effect with traffic shaper at µC output

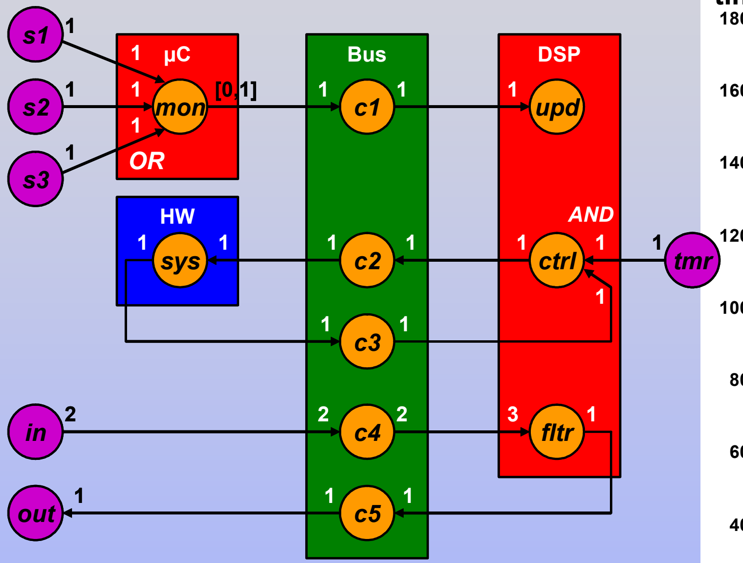
Current SYMTA developments

- interactive optimization
- sensitivity analysis

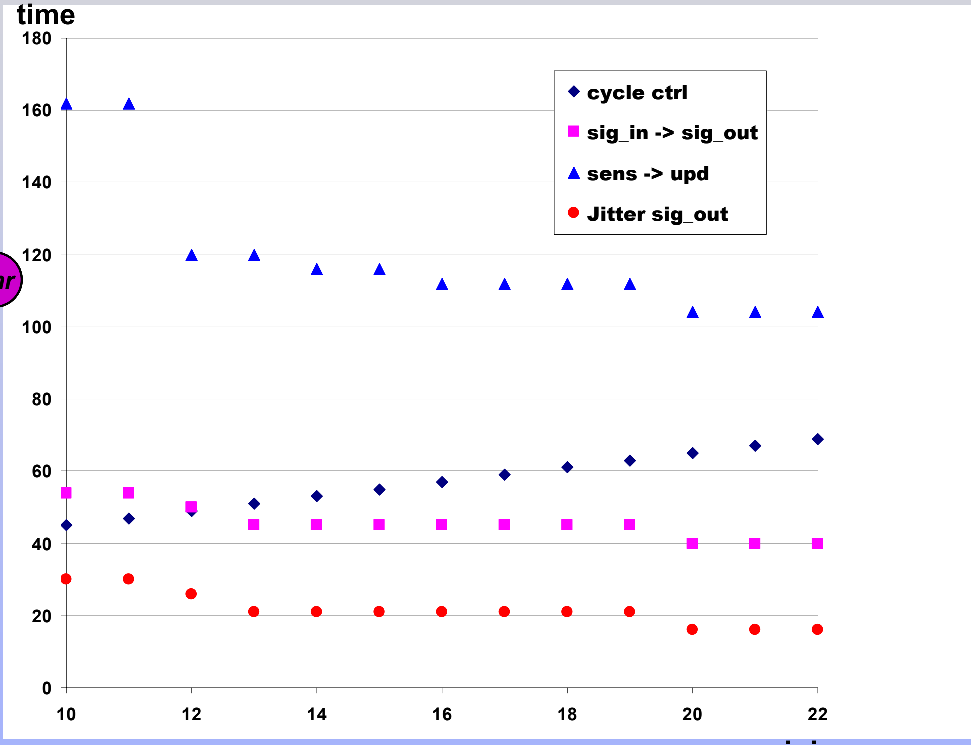
Interactive optimization w. SYMTA/S



Optimization using traffic shaping



Example



minimum event distance

System behavior with traffic shaper at mon output

Conclusion

- system integration is key MPSoC design problem
- architecture modeling and verification are key integration problems
- many hidden performance problems not reflected in system function
- lecture presented hierarchical analysis based on abstract event flow models
- explained recent work in stream operators, traffic shaper modeling and optimization
- work applied in several ongoing industry co-operations

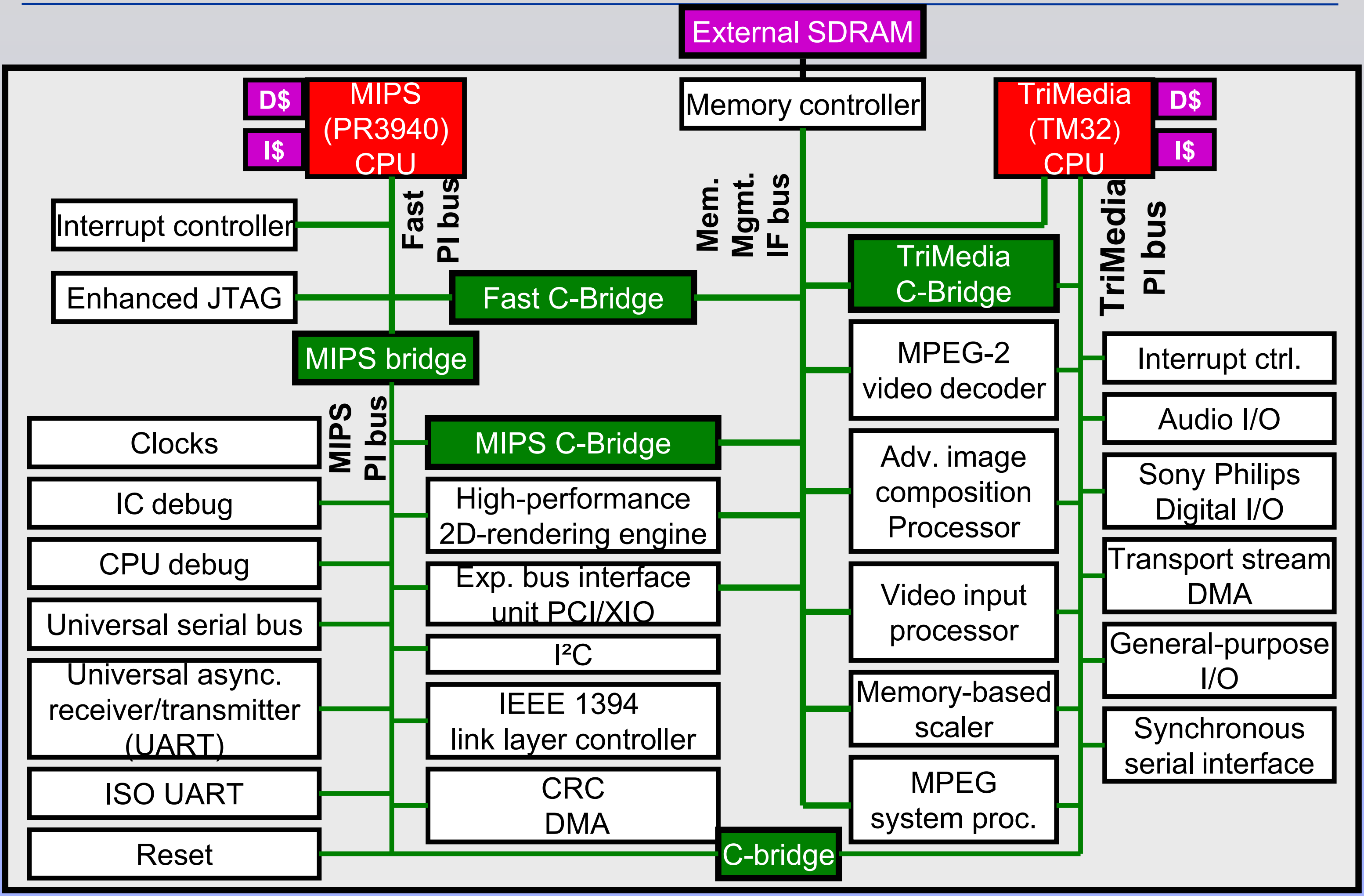
Acknowledgement

- The following persons made major contributions to this work
  - Arne Hamann
  - Rafik Henia
  - Marek Jersak
  - Razvan Racu
  - Kai Richter

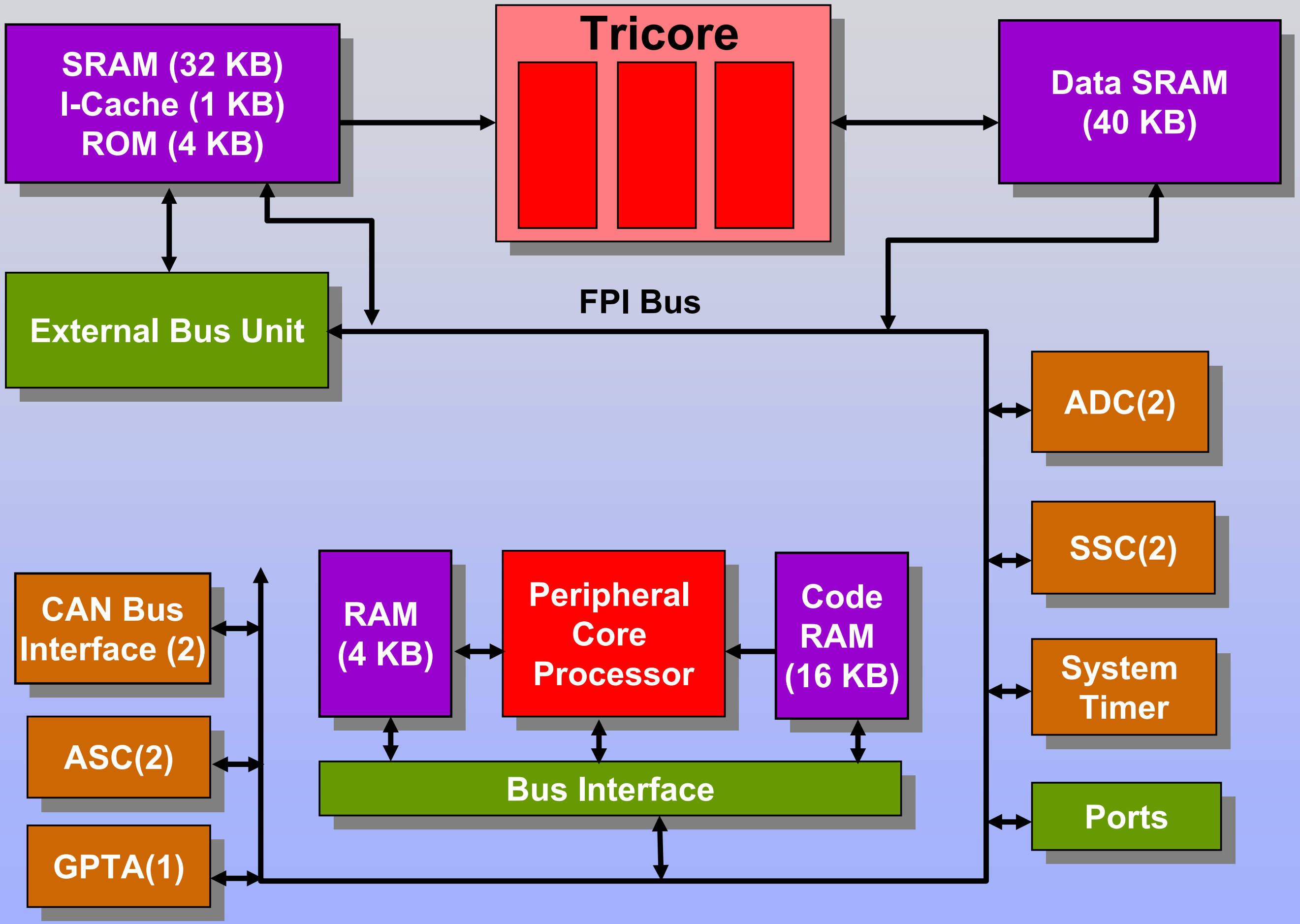
Literature

- see:
  - www.spi-project.org
  - www.symta.org

Nexperia example: Viper Setop Box

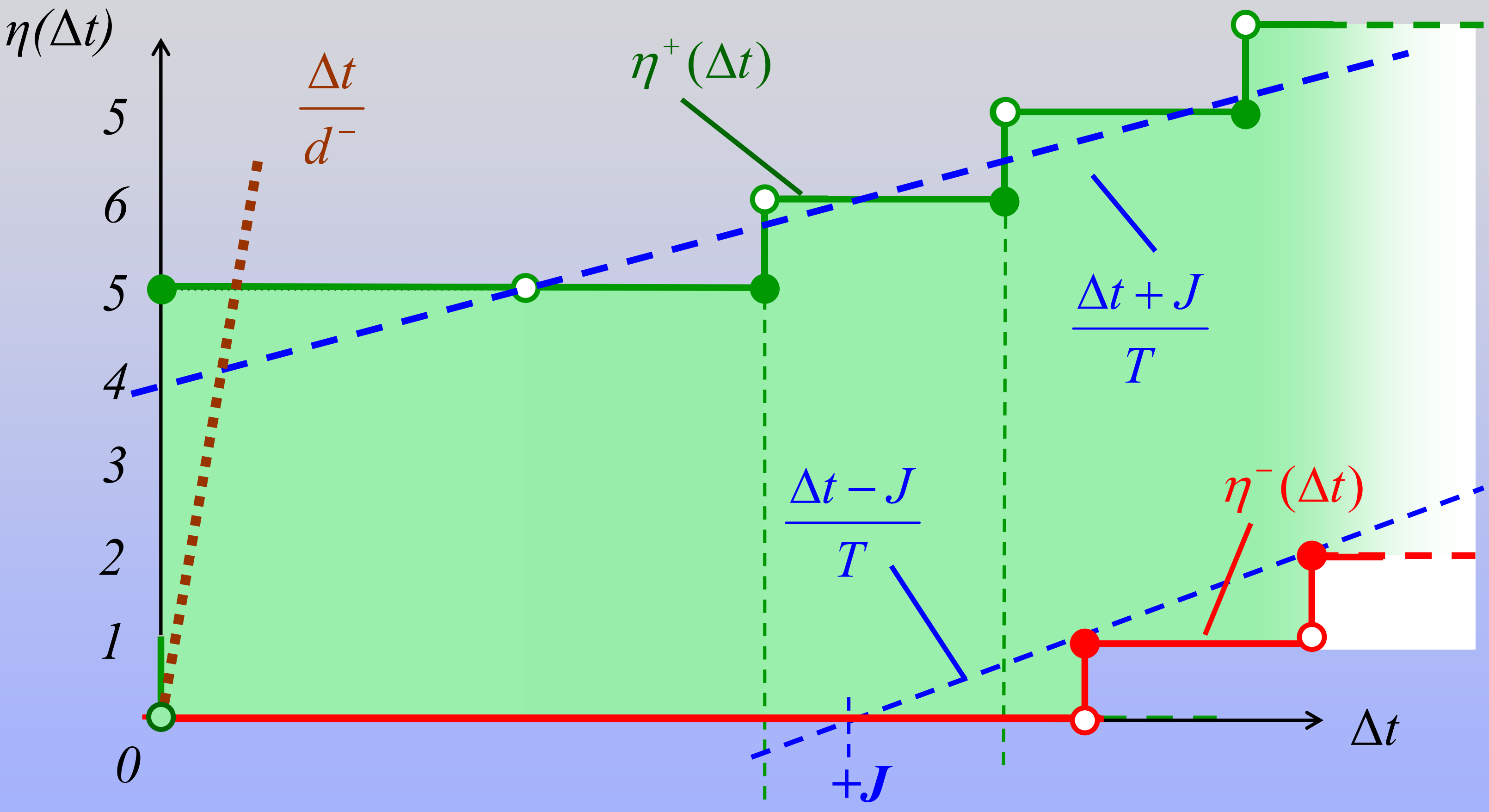


Automotive SoC complexity



Infineon TriCore 1775

Example: Periodic signal with burst


$\eta(\Delta t)$
$\frac{\Delta t}{d^-}$
$\eta^+(\Delta t)$
$\frac{\Delta t + J}{T}$
$\frac{\Delta t - J}{T}$
$\eta^-(\Delta t)$
Δt
$+J$