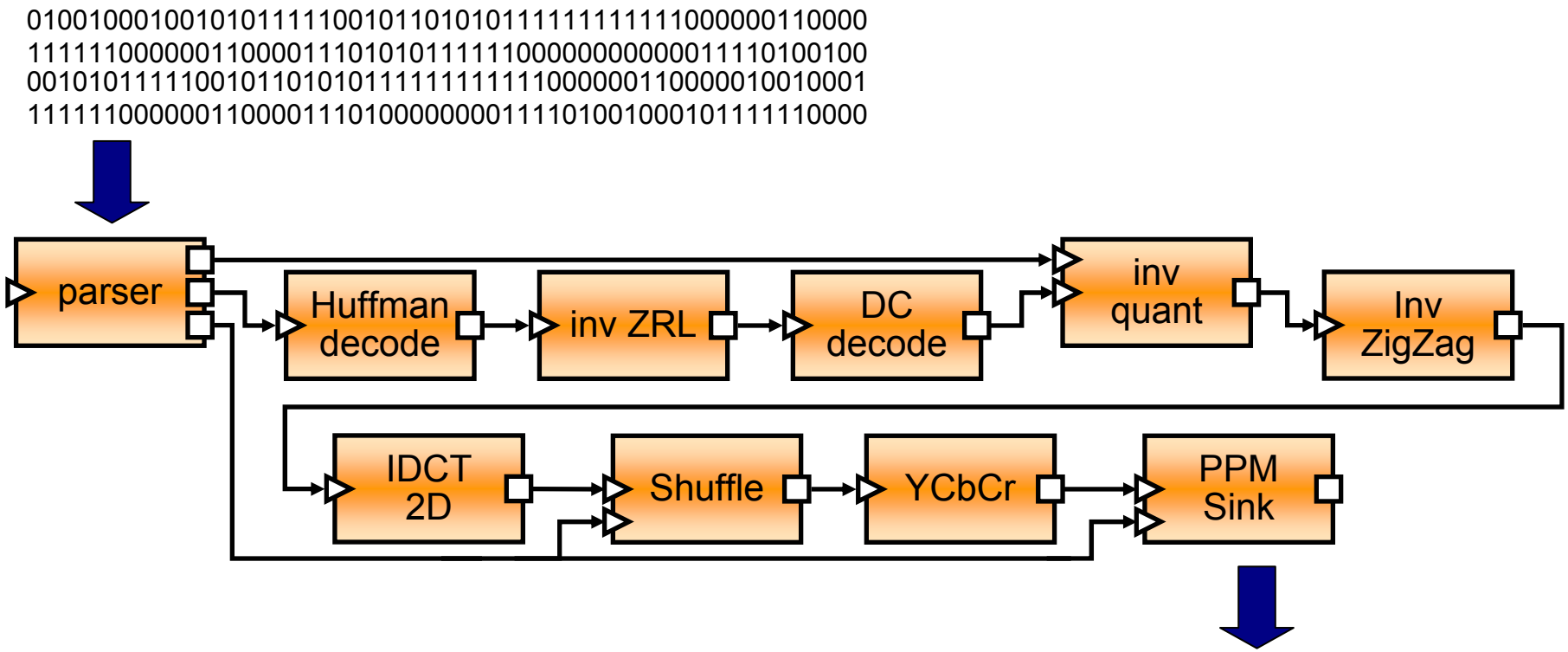
Automatic Design Space Exploration and Prototyping from Behavioral Models

Joachim Falk, Jens Gladigau, Michael Glaß, Martin Lukasiewycz,
Joachim Keinert, Felix Reimann, Martin Streubühr,
Christian Haubelt, Jürgen Teich

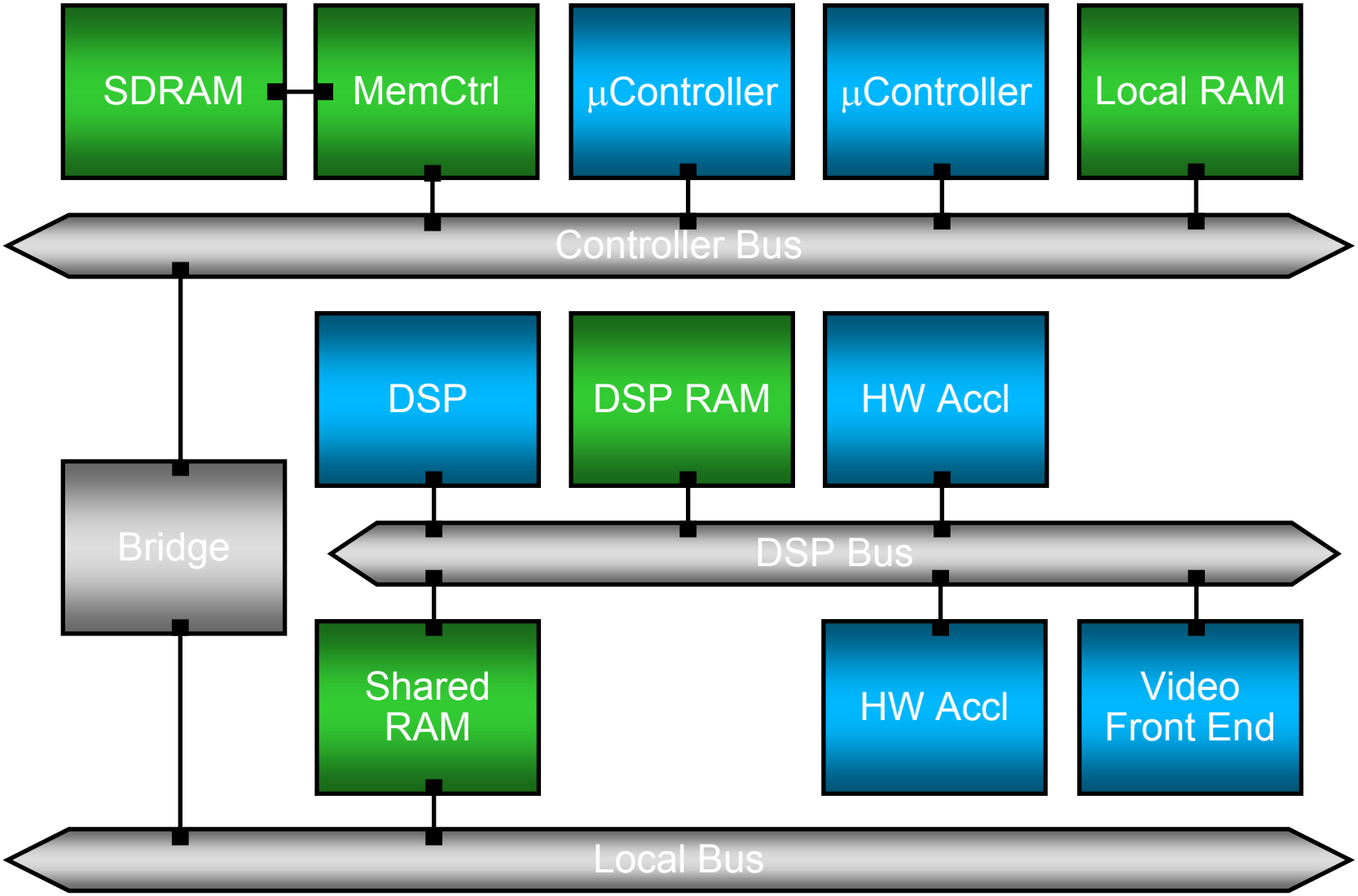
Hardware/Software Co-Design
University of Erlangen-Nuremberg

JPEG Decoder Application

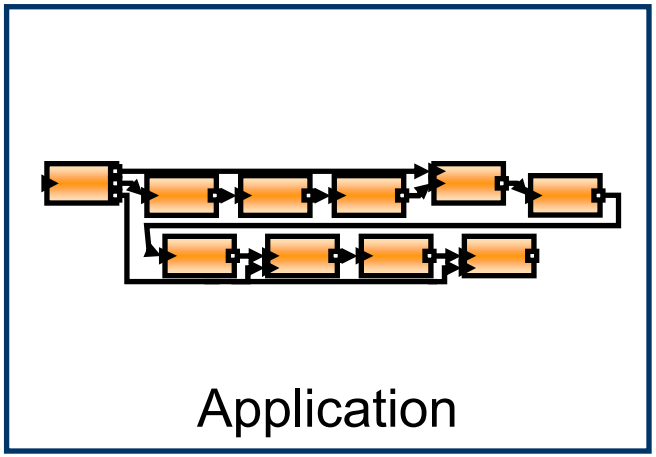


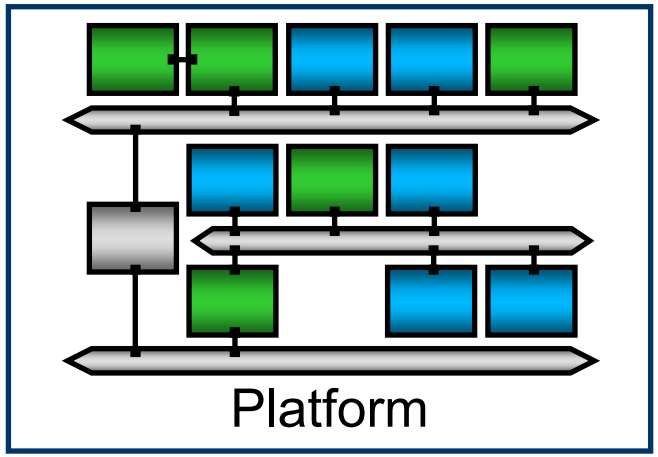
- Some Questions:
  - Run everything in software? How many processor cores?
  - Design a hardware accelerator for this application?
  - Compute only parts in hardware?

MPSoC Platform



Mapping





Automatic Mapping

Agenda

- Overview
- Design Space Exploration
- Embedded Software Generation
- Prototyping
- Case study
- Conclusions

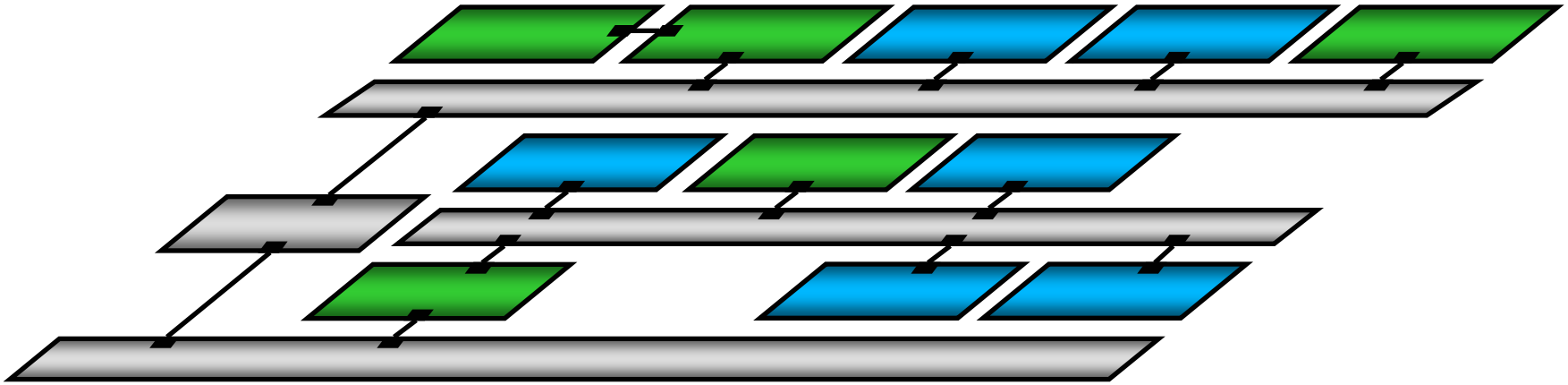
System Synthesis

- System synthesis comprises:
  - Resource allocation
  - Actor binding
  - Channel mapping
  - Transaction routing

- Idea:
  - Formulate system synthesis problem as 0-1 ILP
  - Use Pseudo-Boolean (PB) solver to find a feasible solution
  - Use Multi-Objective Evolutionary Algorithm (MOEA) to optimize Decision Strategy of the PB solver

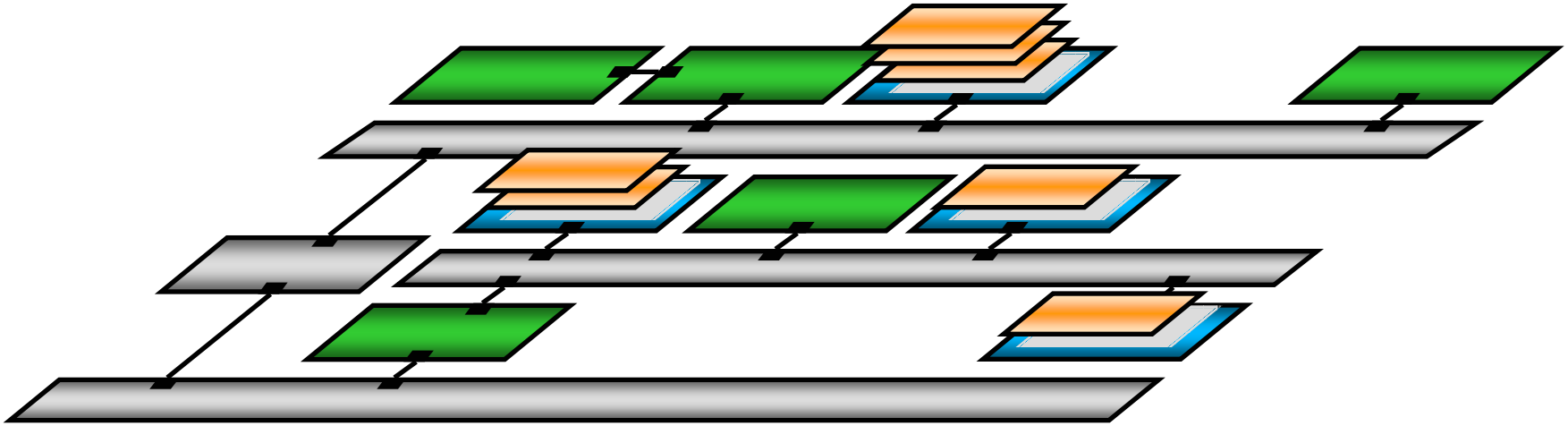
System Synthesis (Resource Allocation)

- R denotes the set of resources
- Resource activation $\alpha: R \rightarrow \{0,1\}$
- $\alpha(r) = 1$ denotes resource activation
- $\alpha(r) = 0$ denotes resource deactivation



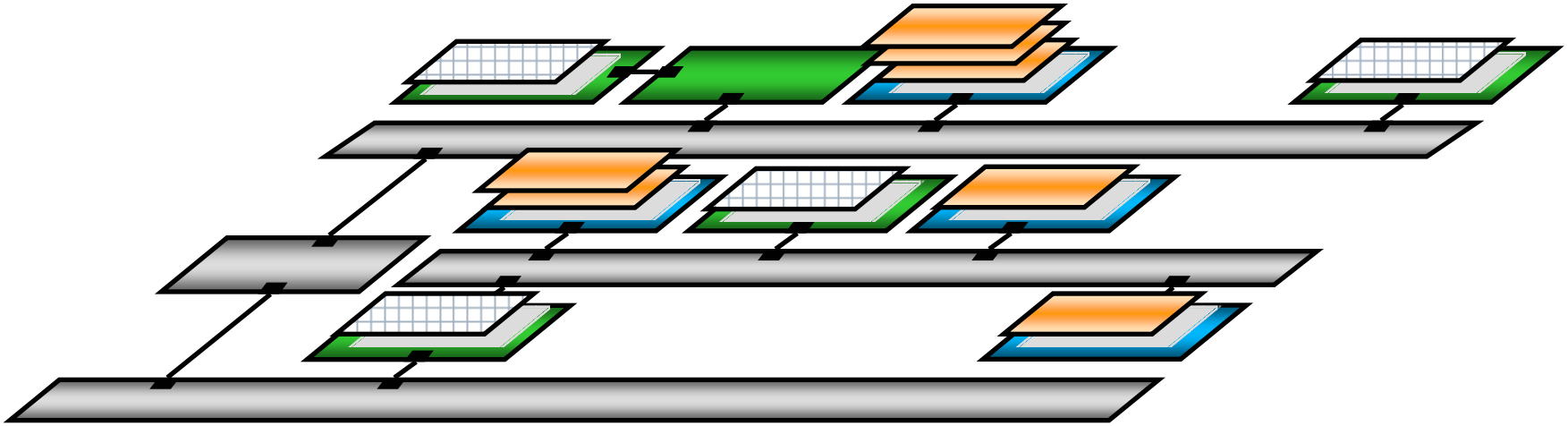
System Synthesis (Actor Binding)

- A denotes the set of actors
- Actor binding activation $\alpha: A \times R \rightarrow \{0,1\}$
- $\alpha(a,r) = 1$ binds actor a onto resource r
- $\forall a \in A: \sum \alpha(a,r) = 1$ (Each actor is bound exactly once)



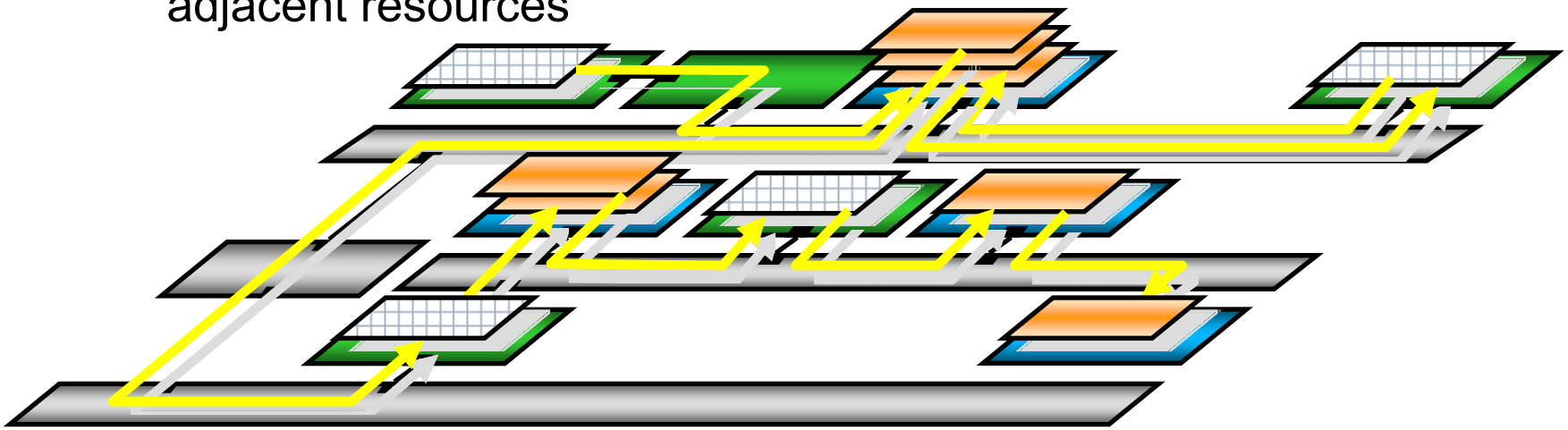
System Synthesis (Channel Mapping)

- C denotes the set of channels
- Channel mapping activation $\alpha: C \times R \rightarrow \{0,1\}$
- $\alpha(c,r) = 1$ maps channel c onto resource r
- $\forall c \in C: \sum \alpha(c,r) = 1$ (Each channel is mapped exactly once)

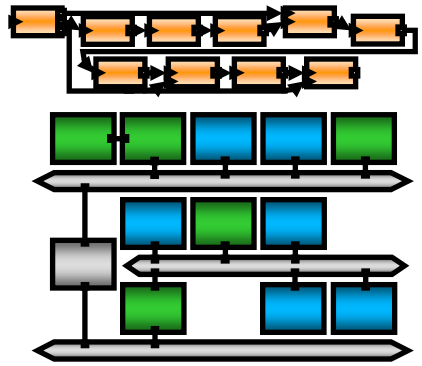


System Synthesis (Transaction Routing)

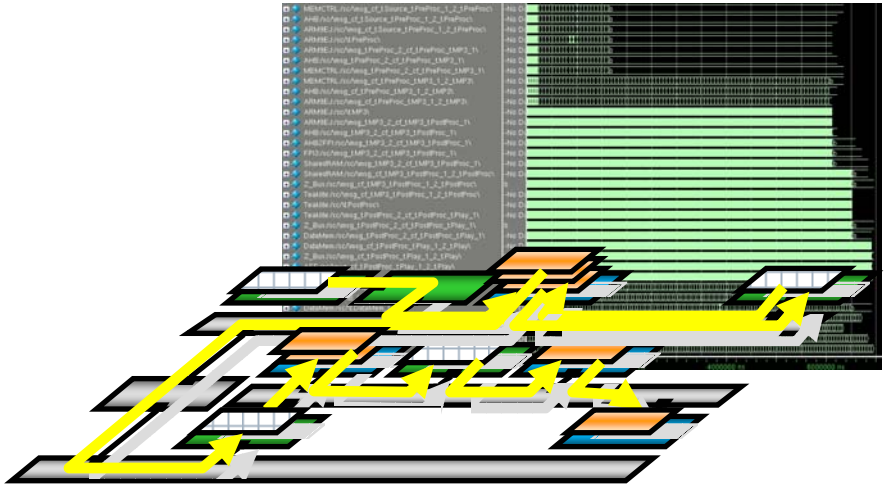
- Additional variables for transactions (memory accesses)
  $t_{a,c,r,n}$, $t_{c,a,r,n} \in \{0,1\}$
- n is the communication step
- Set of constraints
  - Transaction has to start or end at the resource a sending/receiving actor is bound to, respectively
  - Each transaction must not visit a resource more than once
  - Successive communication steps must be performed on adjacent resources

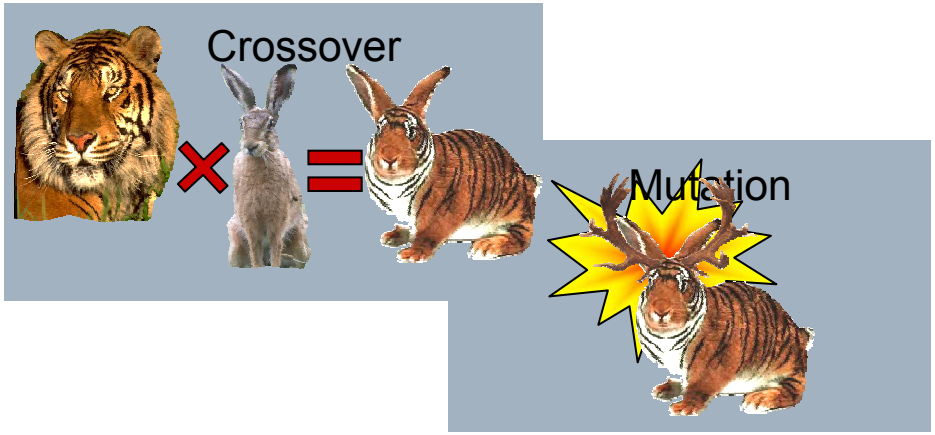


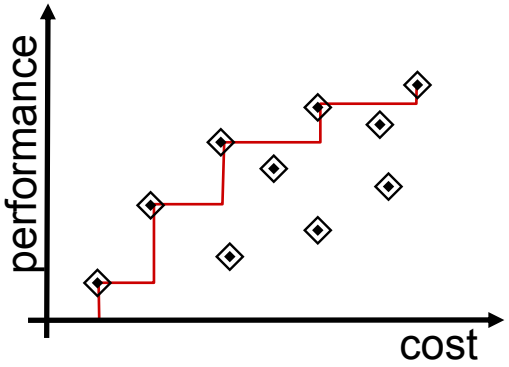
Design Space Exploration



PB Solver



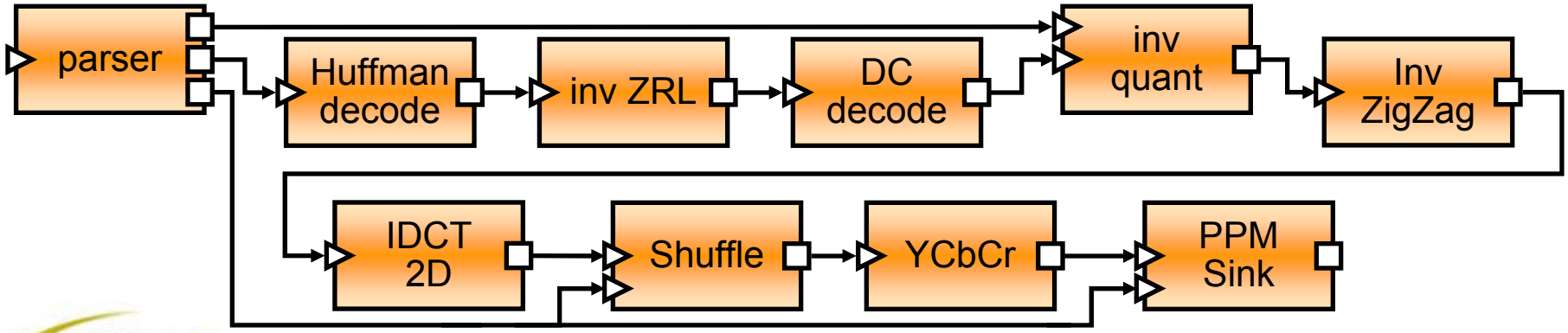




Agenda

- Overview
- Design Space Exploration
- Embedded Software Generation
- Prototyping
- Case study
- Conclusions

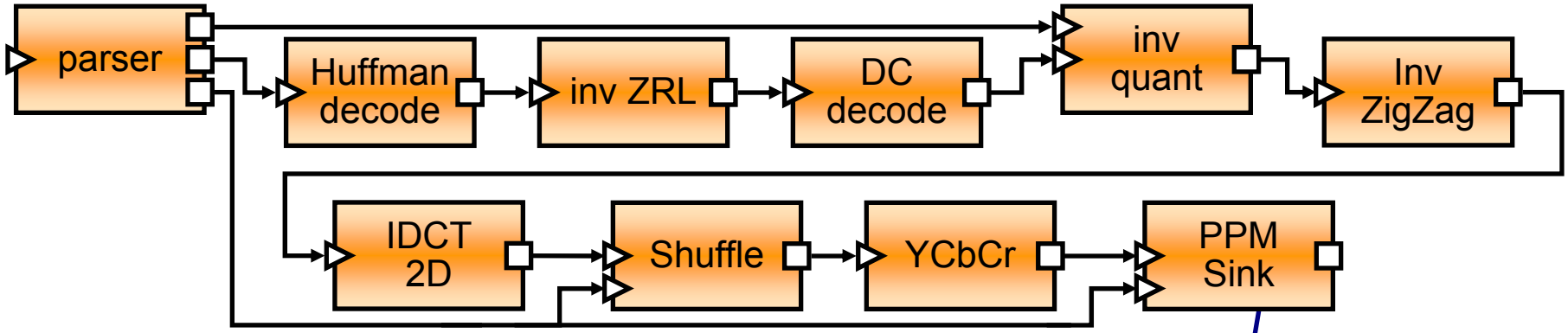
SystemC Advantages



SYSTEMC™

- Permits hardware/software co-design
- Actor-oriented design
  - Network graph
  - Channels
  - Actors
- Actors are only allowed to communicate via channels

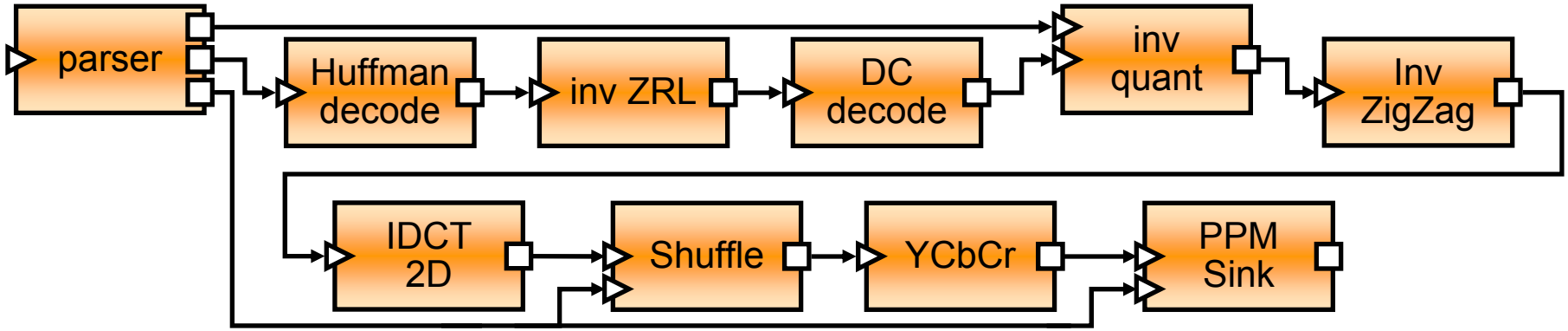
SystemC Disadvantages




```
class PPMSink: public sc_module {
  void process()   {
    while(1) {
      dimX = i2.read();
      dimY = i2.read();
      pixels = dimX*dimY;
      printHeader();
      for (int n=0; n<pixels; n++)
        printPixel(i1.read());
  }
};
```

- Complex control flow and unstructured channel accesses prevent proper analysis leading often to suboptimal application implementations

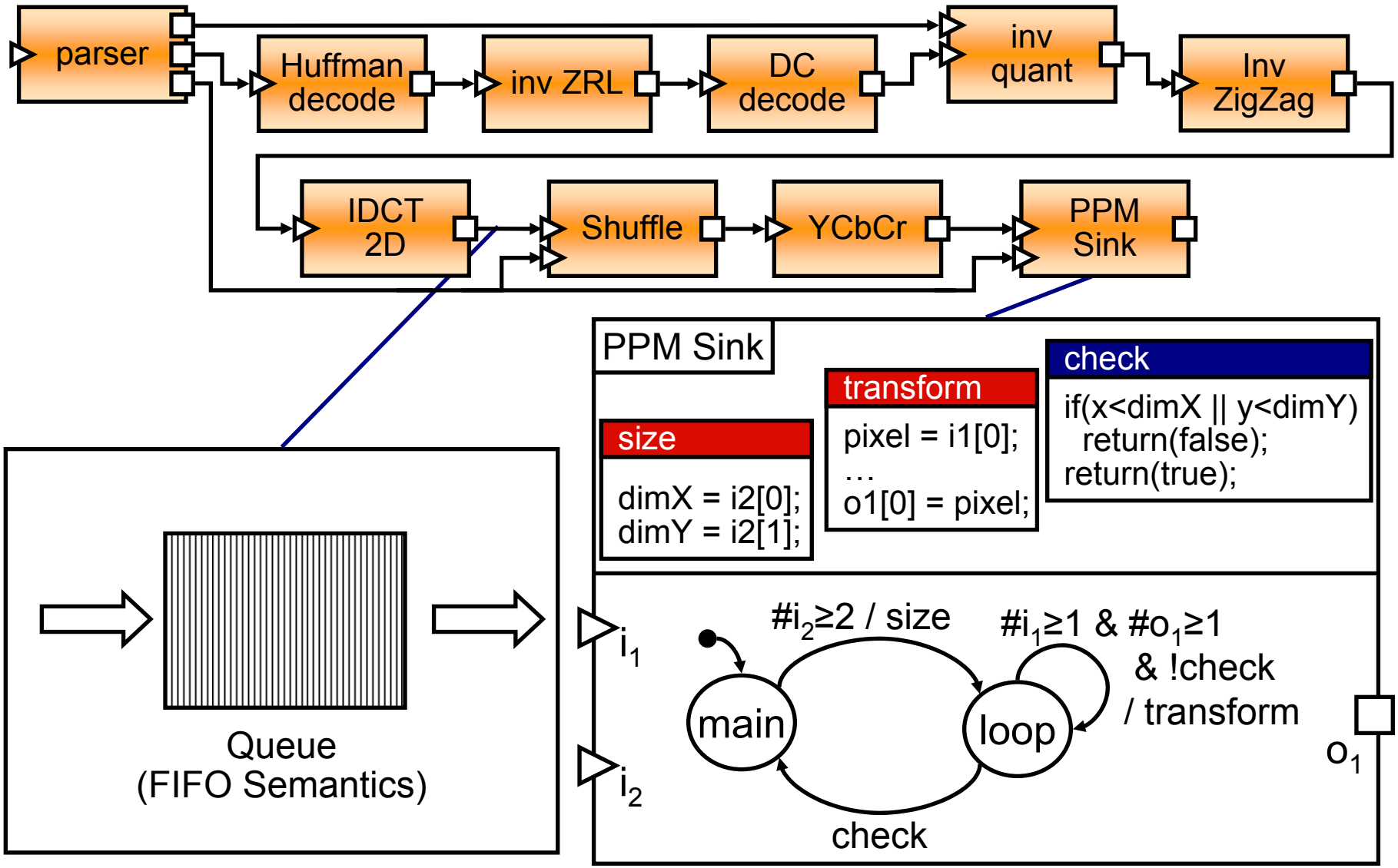
Single Processor Scheduling




```
schedule() {
  while(true) {
    forall actors a ∈ A {
      if is_executable(a) {
        execute(a)
      }
    }
  }
}
```

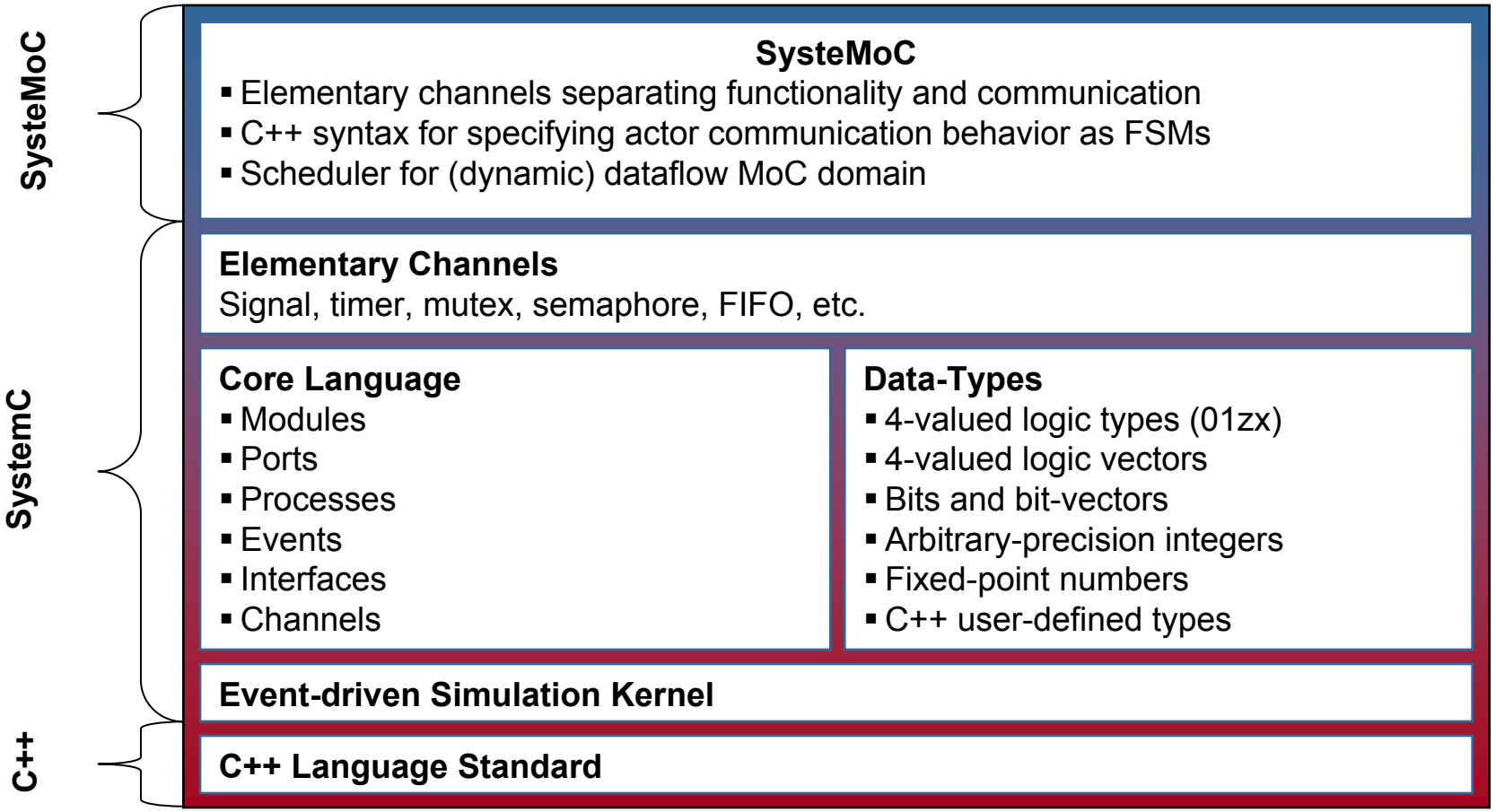
Potential scheduling overhead

SystemCoDesigner Modeling Approach

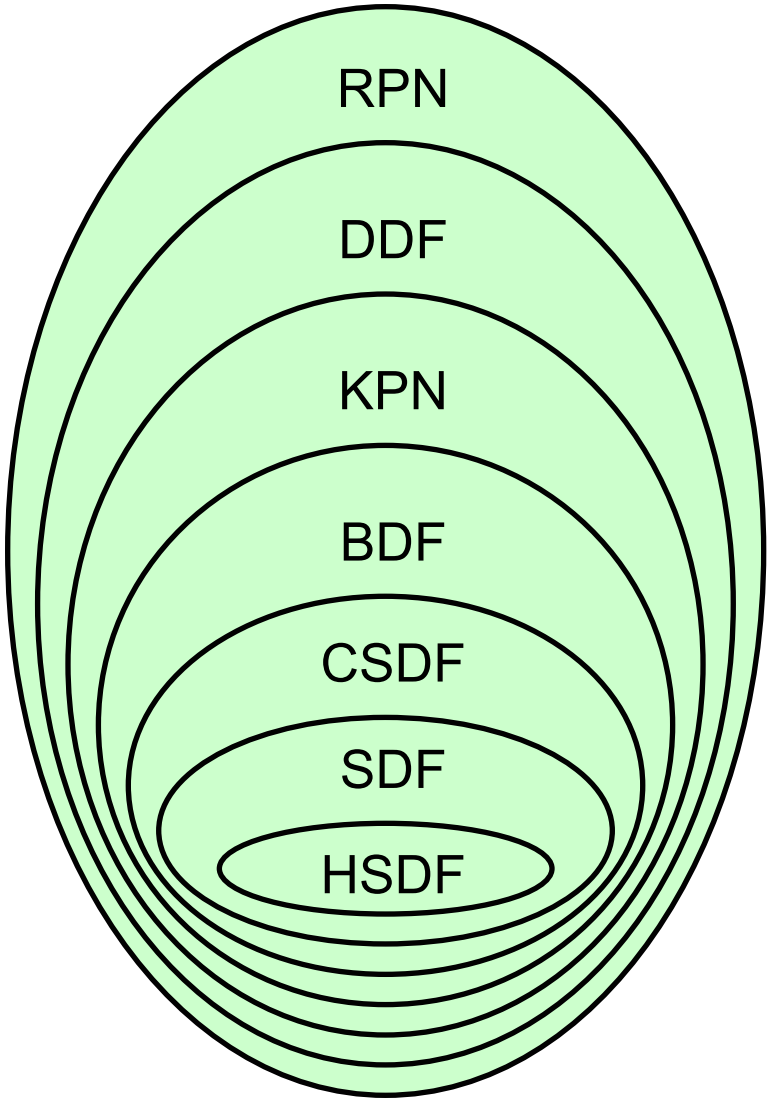


SysteMoC

- SysteMoC – SystemC library for actor-based design
  - Actor functionality (member functions and variables)
  - Communication behavior (activation pattern)



Hierarchy of Streaming MoCs



BDF and larger: Turing complete

FunState ≈ RPN

RPN: Reactive Process Network

DDF: Dynamic Dataflow

KPN: Kahn Process Network

BDF: Boolean Dataflow

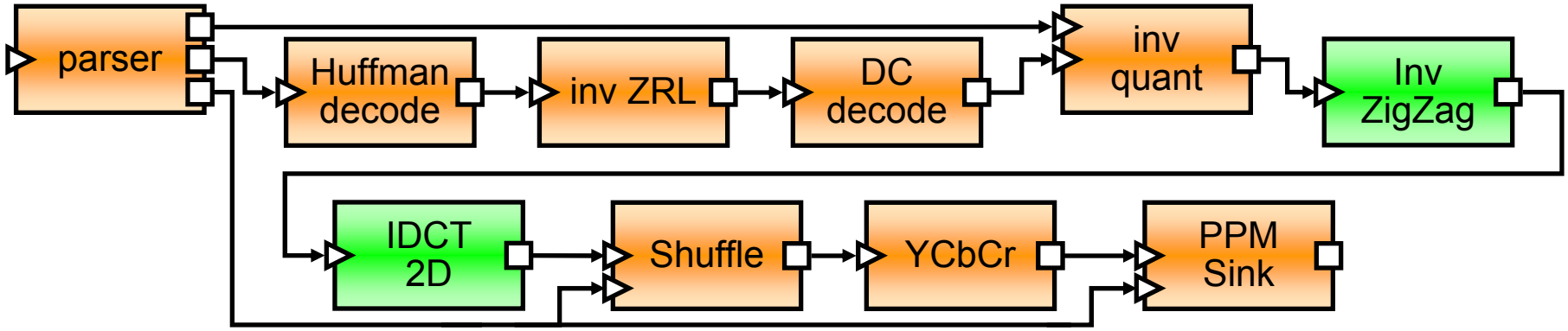
CSDF: Cyclo-Static Dataflow

SDF: Synchronous Dataflow

HSDF: Homogeneous SDF

[Basten@MoCC2008]

Improvement

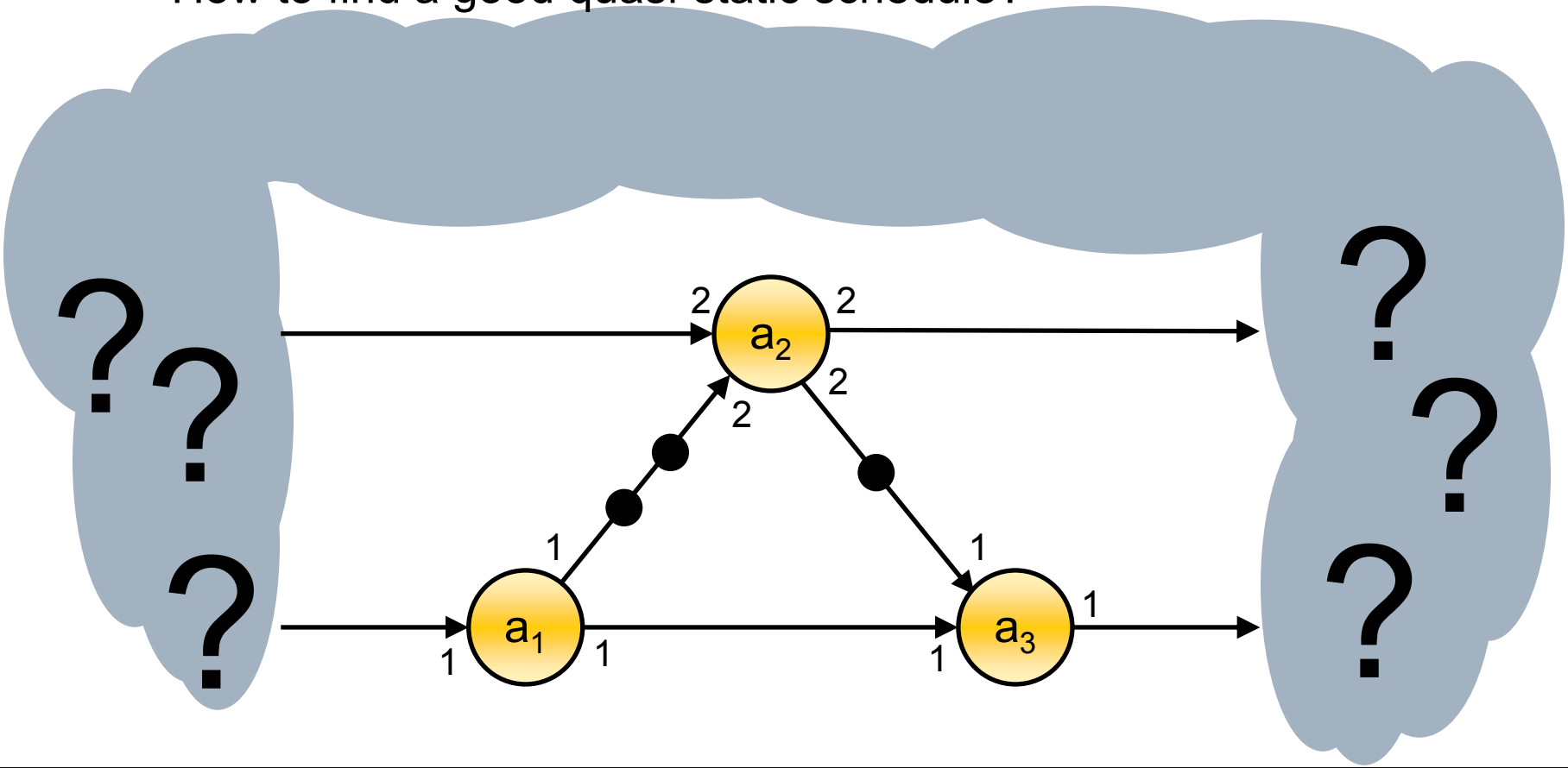



```
schedule() {
  while(true) {
    if is_executable(inv. ZigZag) {
      execute(inv. ZigZag)
      execute(IDCT2D)
    }
    forall actors a ∈ A_dyn {
      if is_executable(a) {
        execute(a)
      }
    }
  }
}
```

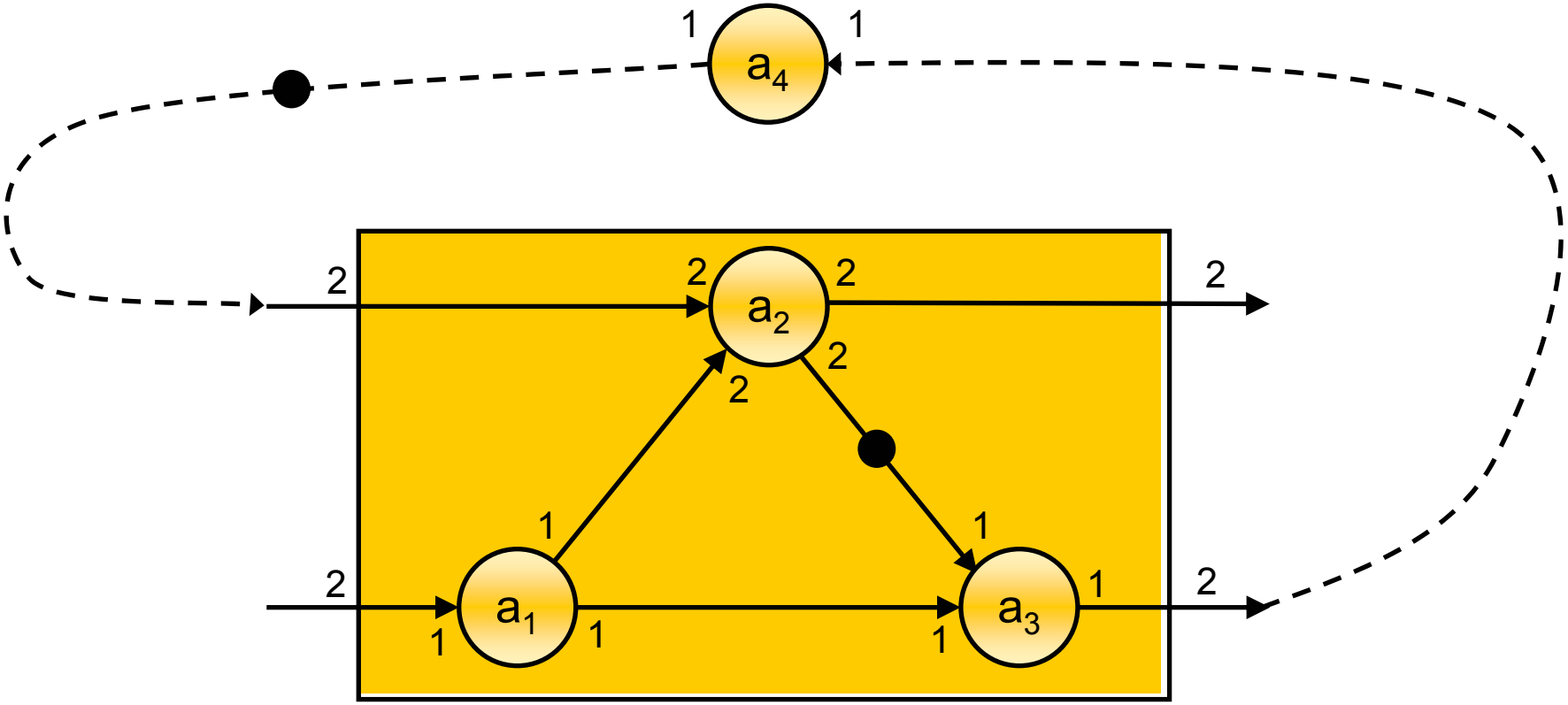
- JPEG Decoder example
  - speedup by a factor of two.
- Synthetic benchmarks
  - speedup up to a factor of 10.

The Situation

- We got a dataflow graph
  - Some actors, or even the topology, are unknown
  - But a subgraph of SDF actors is known
  - How to find a good quasi-static schedule?

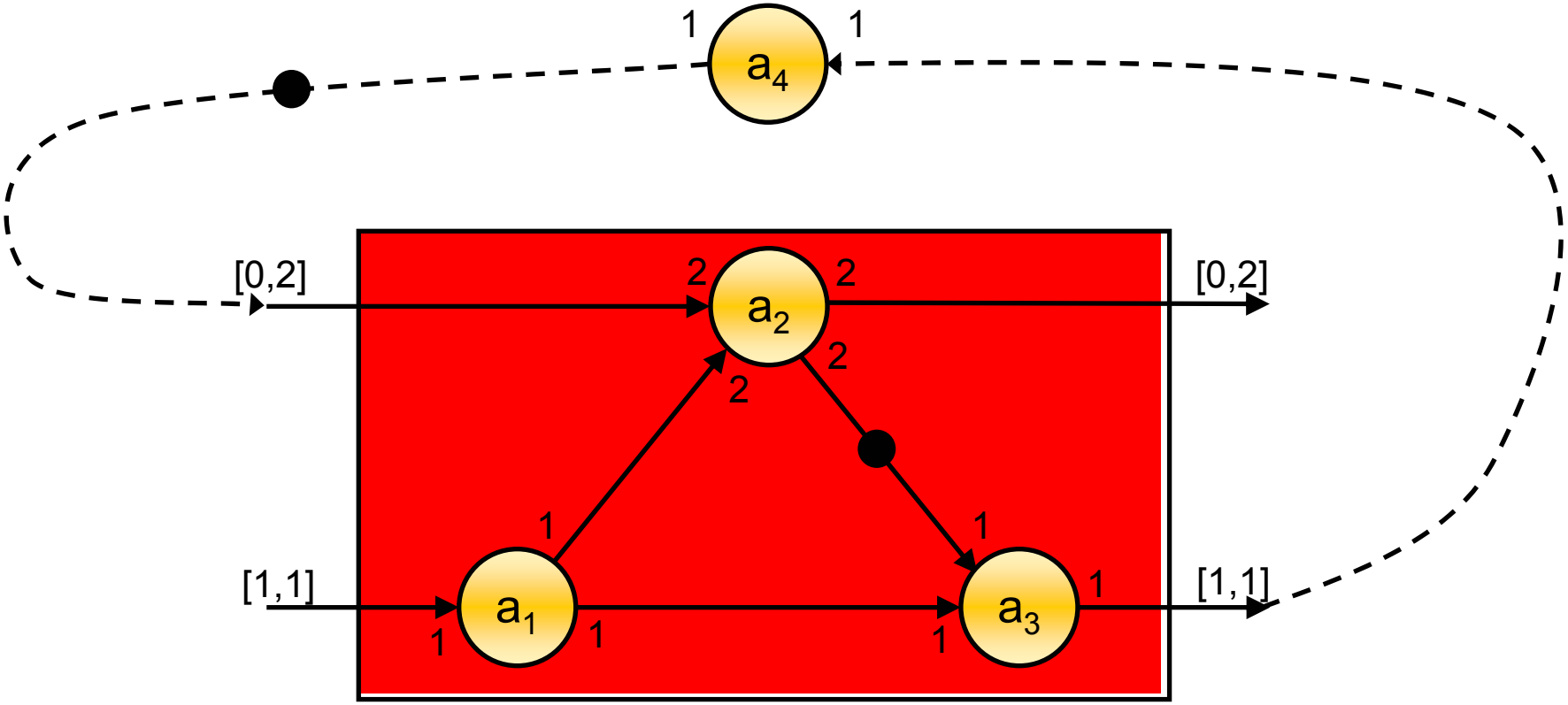


Clustering (SDF)



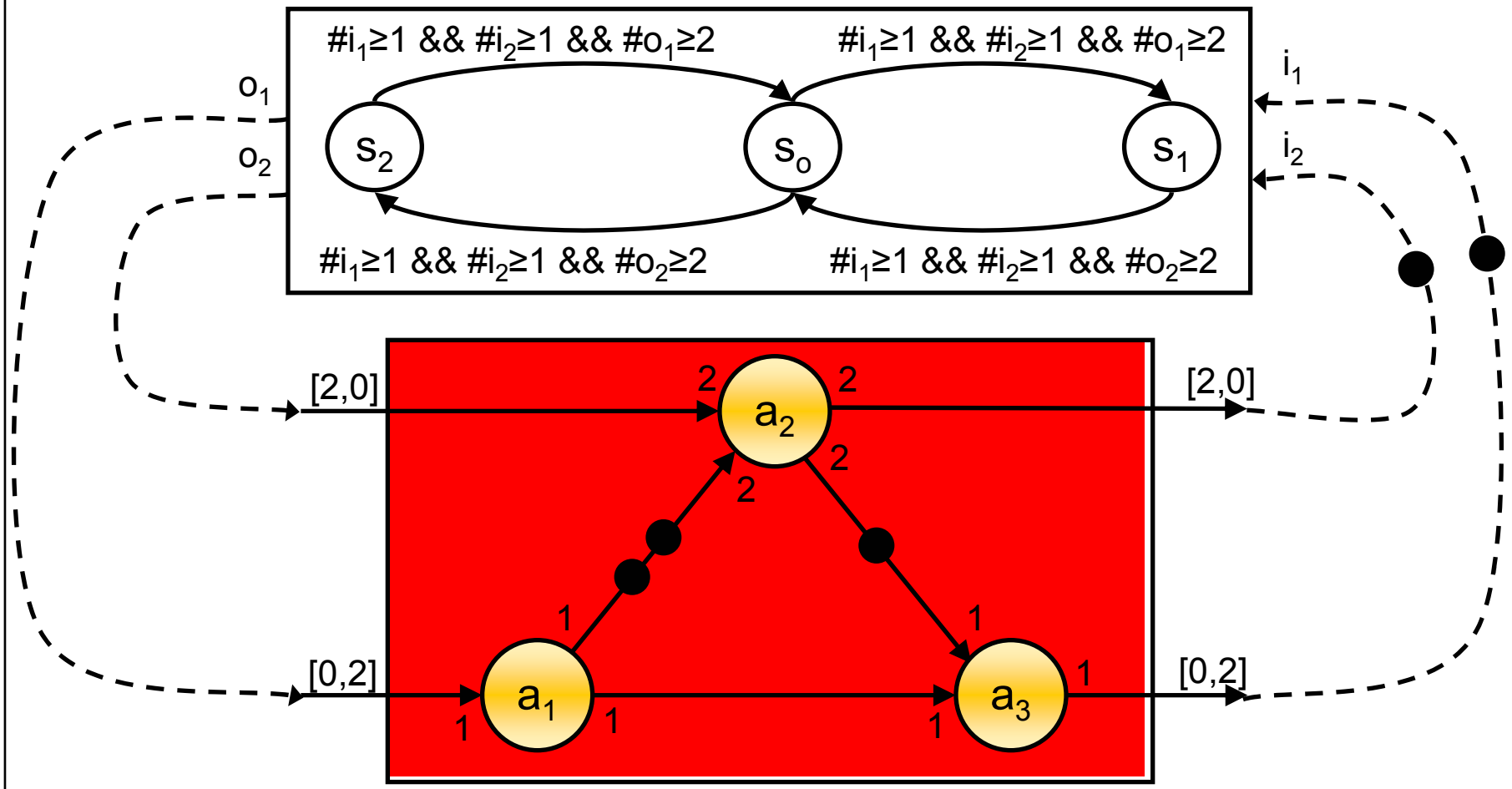
$\tau = (2*a_1, 1*a_2, 2*a_3)$

Clustering (CSDF)



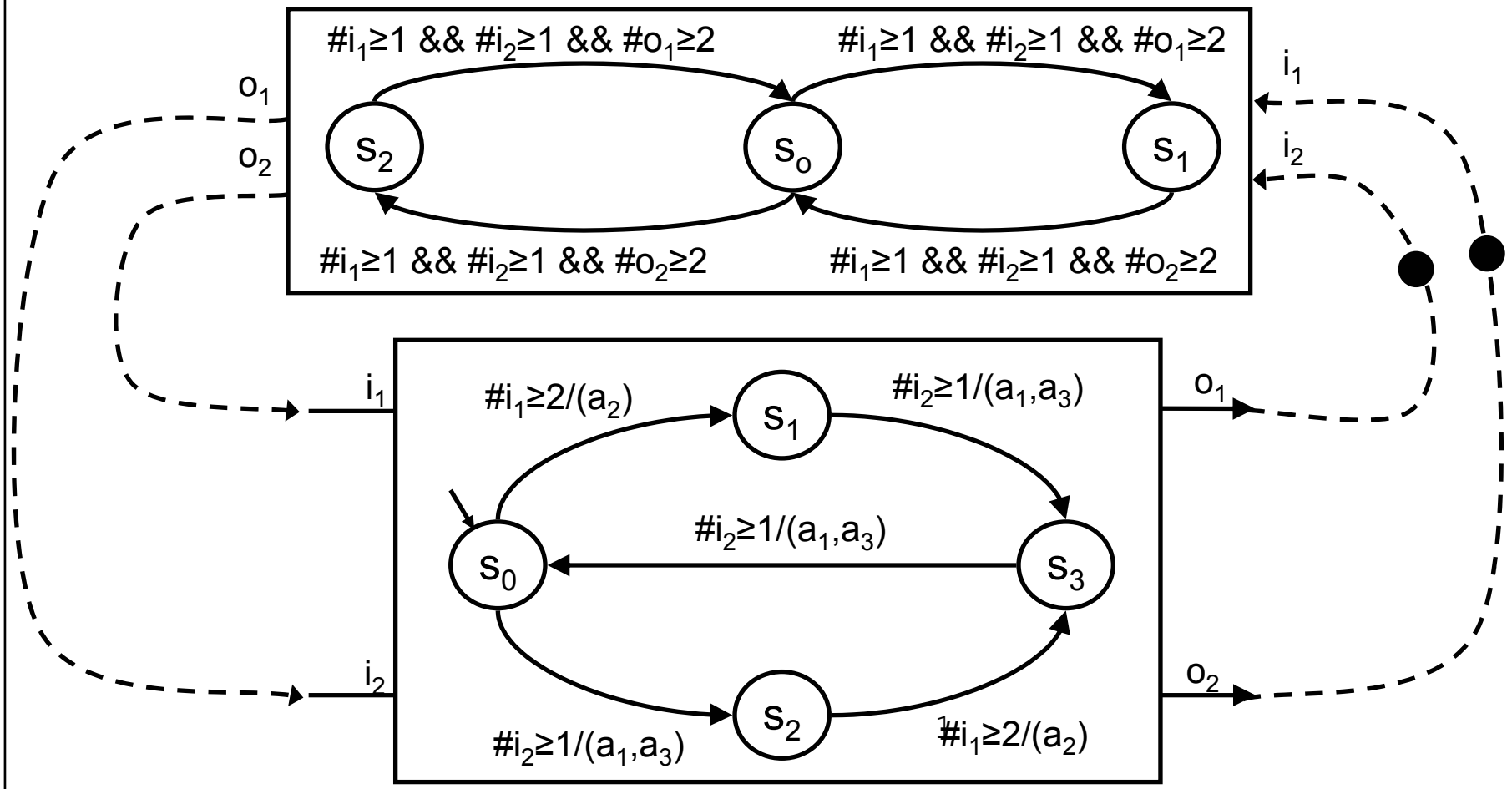
$\tau = ((1*a_1,1*a_3), (1*a_1,1*a_2, 1*a_3))$

Clustering 2 (CSDF)

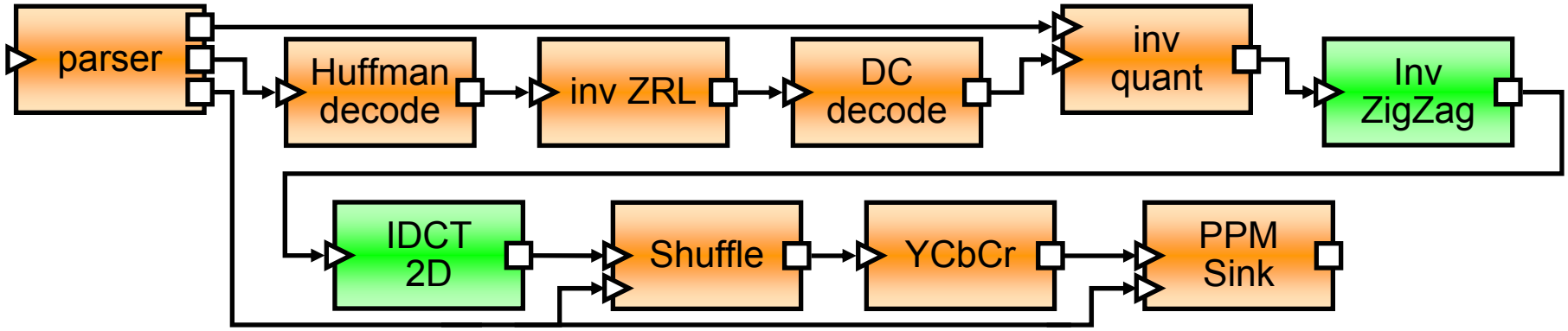


$\tau = ((1*a_2),(2*a_1,2*a_3))$

Clustering (dynamic dataflow)



Clustering Results

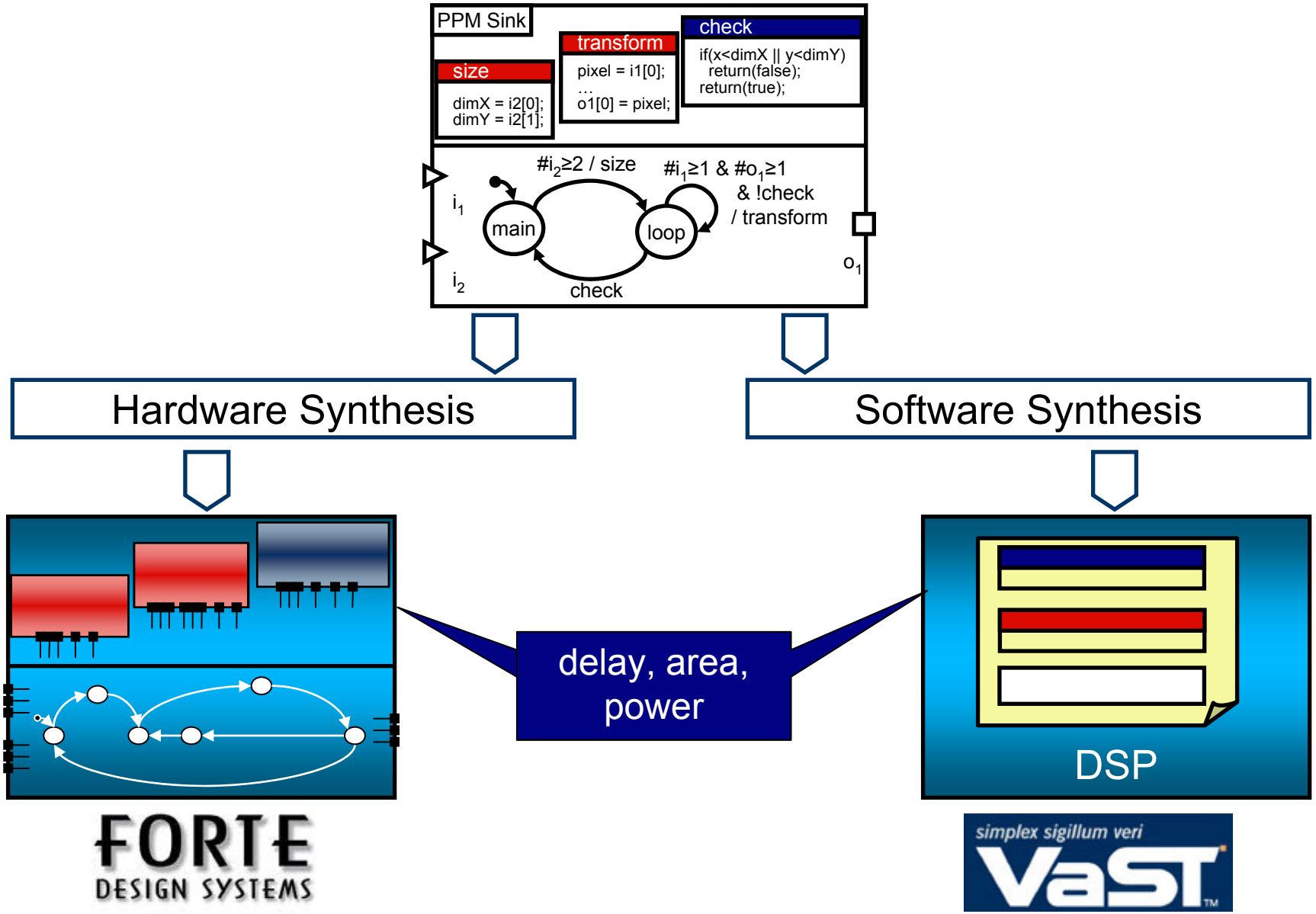


- Real world example [EMSOFT2008]
  - JPEG QCIF (176x144) decoding
  - Dynamic scheduling obtained 88 ms per frame
  - After clustering of the IDCT and the InvZigZag actor 42 ms per frame were obtained.
  - Whole system speedup by a factor of two.
- Synthetic benchmarks
  - Ten random graphs with 60 nodes and 5000 clusters
  - Maximal speedup found per graph is by a factor of 10.
  - Selection of actors for clustering has a huge impact on the obtained speedup.

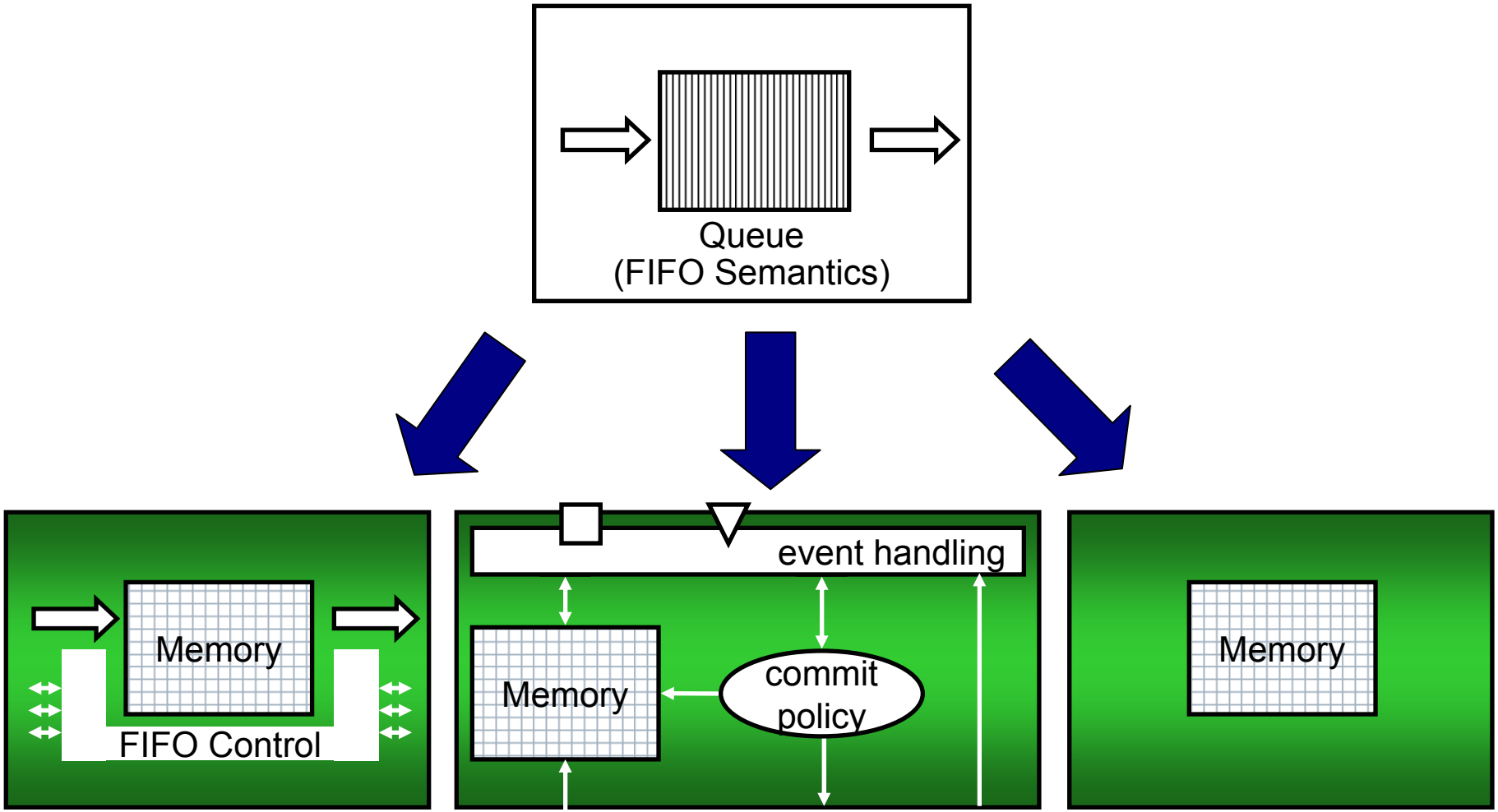
Agenda

- Overview
- Design Space Exploration
- Embedded Software Generation
- Prototyping
- Case study
- Conclusions

Actor Synthesis

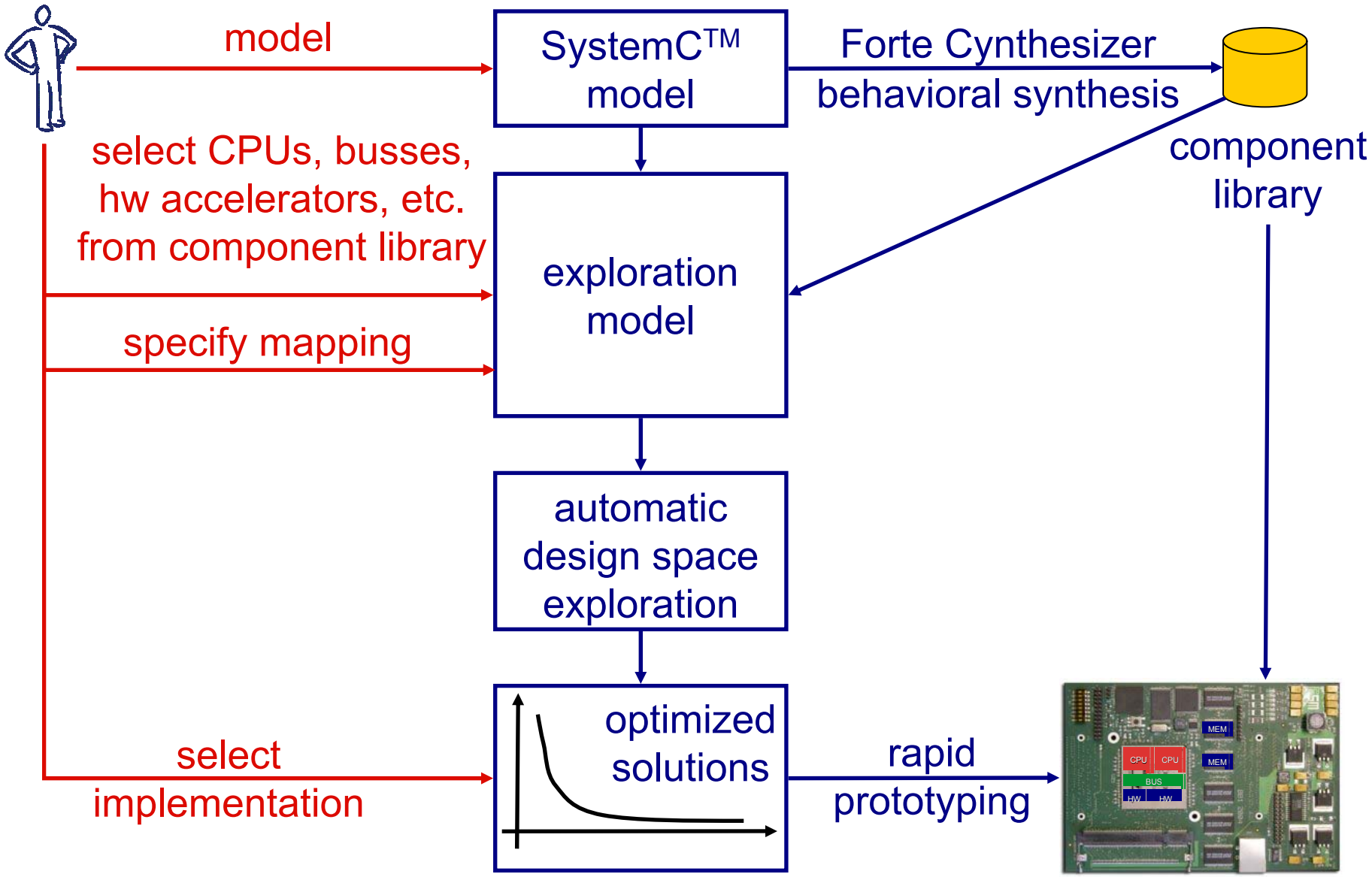


Communication Synthesis



HW/SW Performance Estimation

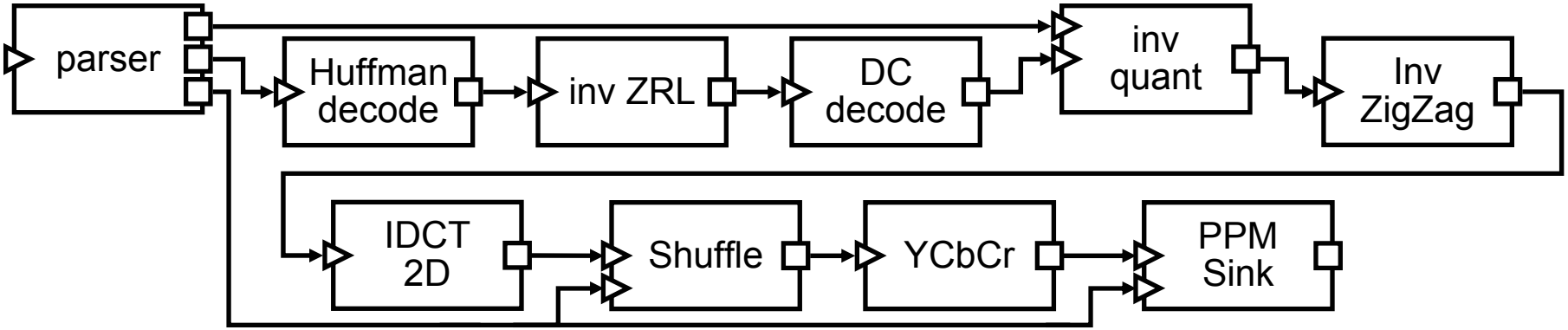
Design Flow


model
SystemC™ model
Forte Cynthesizer behavioral synthesis
component library
select CPUs, busses, hw accelerators, etc. from component library
exploration model
specify mapping
automatic design space exploration
optimized solutions
select implementation
rapid prototyping
CPU
CPU
BUS
HW
HW
MEM
MEM

Agenda

- Overview
- Design Space Exploration
- Embedded Software Generation
- Prototyping
- Case study
- Conclusions

Motion JPEG (1/2)



- Baseline profile without subsampling (interleaved/non-interleaved)
- 5650 SysteMoC LoC
  - 4 PDs specification and interface definition
  - 16 PDs module implementation
  - 14 PDs integration and debugging (~ 3 PDs integration)
  - 4 PDs SCD SysteMoC code adaptation
- Any HW/SW implementation on Xilinx Virtex II FPGA (50 MHz) ✓
  - HW only implementation QCIF@65frames/second)

Motion JPEG (2/2)

# SW Actors	Latency	Throughput	LUTs	FFs	BRAM
0	15.63 ms (12.61 ms)	65.0 fps (81.1 fps)	40 467 (44 878)	14 508 (15 078)	47 (72)
1	23.49 ms (25.06 ms)	43.0 fps (40.3 fps)	35 033 (41 585)	11 622 (12 393)	72 (96)
8	6 275 ms (4 465 ms)	0.16 fps (0.22 fps)	15 064 (17 381)	7 540 (8 148)	63 (63)
all	10 030 ms (8 076 ms)	0.10 fps (0.13 fps)	1 893 (2 213)	1 086 (1 395)	29 (29)

7,600 solutions evaluated
366 non-dominated solutions
evaluation time 30.44s/solution

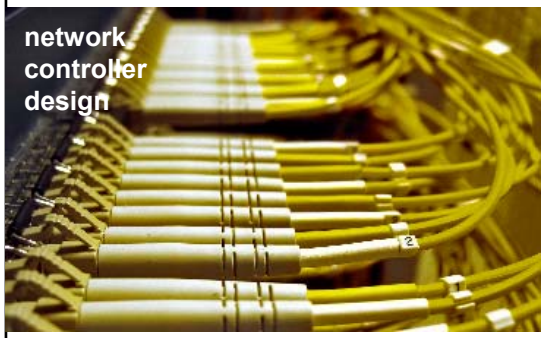
Agenda

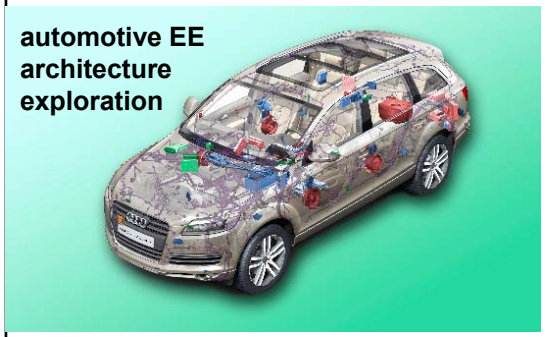
- Overview
- Design Space Exploration
- Embedded Software Generation
- Prototyping
- Case study
- Conclusions

Conclusions

- SystemCoDesigner automatically maps applications to MPSoC platforms
- SystemCoDesigner application model permits an automatic detection of restricted MoC
- Thus, making available design methods applicable
- SystemCoDesigner uses a combination of PB solver and MOEA to perform automatic design space exploration

Project Partners





- Alcatel-Lucent AG
- Audi AG
- Cadence Design Technologies
- Daimler AG
- Forte Design Systems
- Fraunhofer Institute for Integrated Circuits
- IBM Germany, GmbH
- Infineon Technologies AG
- VaST Systems