founded by Philips

Performance Contracts for Modular MPSoC Integration

Pieter van der Wolf
NXP Semiconductors Research

MPSoC'08
June 23 - 27, 2008

founded by PHILIPS

SoC integration

Today's practice

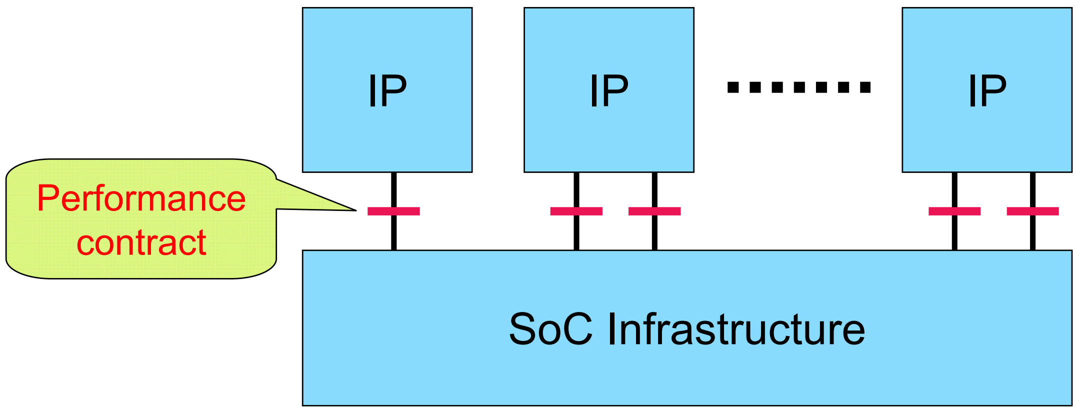
- Complexity of SoC integration increasing
- Communication requirements of IP modules often not explicit
- Performance of one IP module influenced by other IP modules

What if you could …

- Explicitly express communication requirements of IP modules
- Independently verify performance of IP modules and SoC infrastructure
- Reuse performance verification results when reusing IP modules

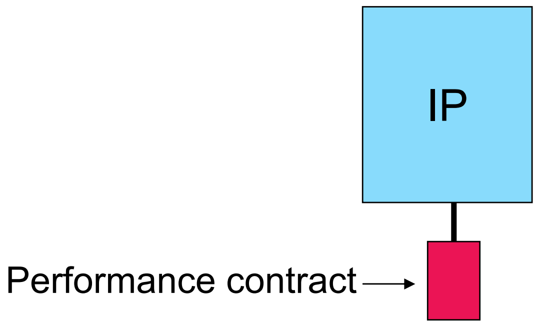
Introduction

- Problem: Performance of IP module depends on SoC infrastructure and other IP modules
- Goal: Enable modular design and verification of MPSoCs
- Performance contracts to decouple IP and SoC infrastructure



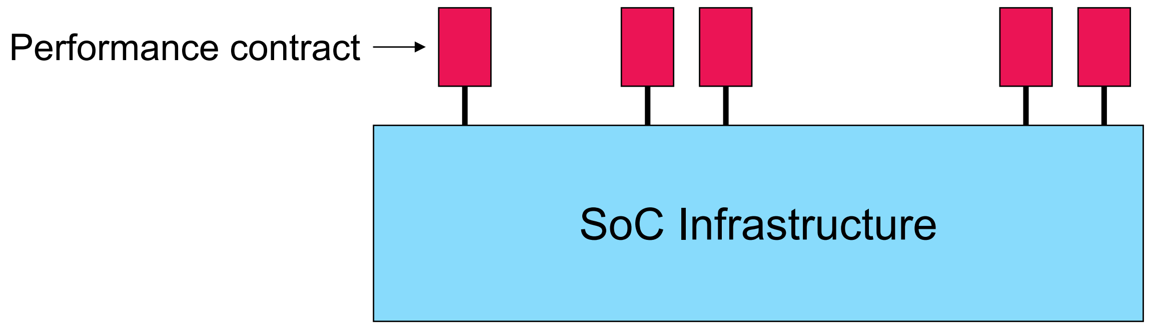
Verification of IP modules

- Verify IP module against performance contract
- Will task executed on IP meet deadlines with performance contract ?
- Performance contract models worst-case behavior of SoC infrastructure
- Per interface (AXI, OCP) there can be more than one contract
- One IP module can have more than one interface

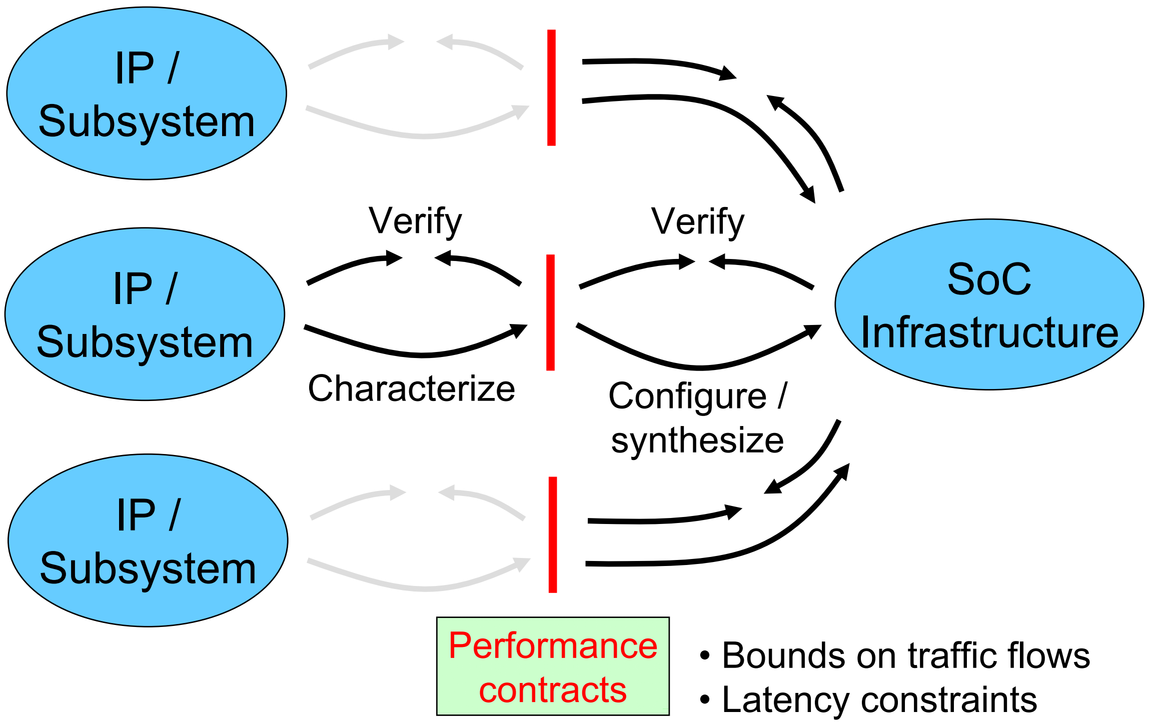


Verification of SoC infrastructure

- Verify SoC infrastructure against performance contracts of all IP modules
- For all use cases
- Performance contract is a model of the worst-case behavior of IP module in use case



SoC performance methodology



- Bounds on traffic flows
- Latency constraints

Performance contracts

Interaction between IP module and SoC infrastructure

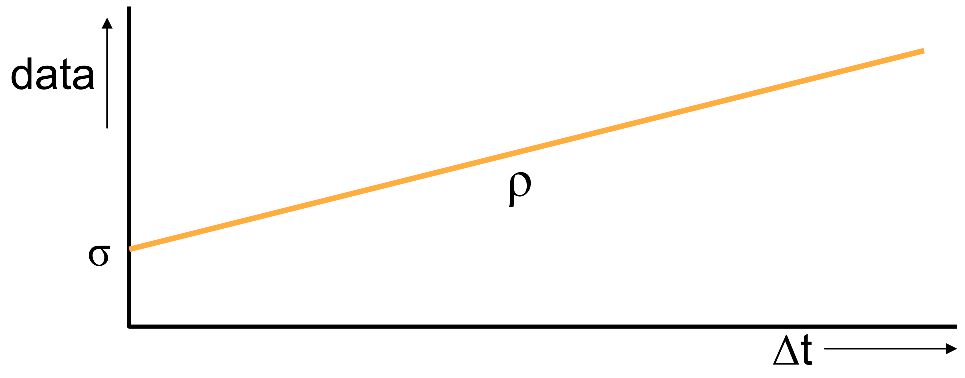
- IP module issues requests to initiate transactions
- SoC infrastructure provides responses with certain latencies

Performance contract is agreement between IP module and SoC infrastructure:

- If IP module keeps request workload below a specified bound
- SoC infrastructure provides responses so that latency constraint is met

Bound on traffic flow

- Express amount of transfered data for sliding window of arbitrary size
- ρ is long term average bandwidth (bytes/s)
- σ is burstiness constraint (bytes)
- For every window of size Δt: data $\leq \sigma + \rho * \Delta t$



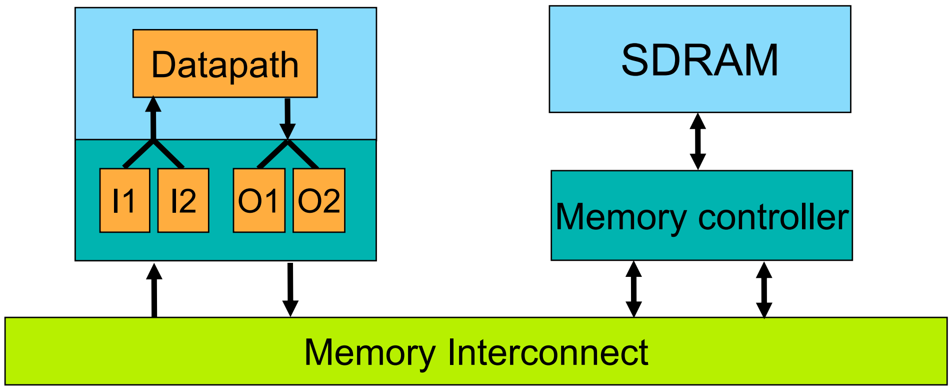
Latency constraint

- Latency constraint expressed as: $\mathrm{lat}_N \leq \delta + \lambda * N$
  - Latency for N consecutive transactions
- Different IP modules react differently to latency
  - Programmable processor
    - Cache miss results in stall time
    - The shorter latency the better
    - Average for a task matters
    - Deadlines may be in ms range
  - Function-specific HW IP module
    - Latency constraint for one or more transactions in certain period
    - Latency can be traded for buffer size

Programmable processor

- Temporal distribution of requests largely unpredictable
- Data size corresponds to cache line (32 B – 128 B)
- Processor stalls upon cache miss
- **Low latency** requirement
- Some (hardware or software) prefetch support to hide latency
- Deadlines at task-level often at millisecond timescale
  - Latencies can be averaged out over set of transactions
  - δ can be high
  - λ should be low

Function-specific HW IP for video processing



- Local DMA fills input buffers / stores output buffers
- No underflow of input buffer / overflow of output buffer allowed
- Predictable addressing scheme allows prefetch
  - Larger buffers give more relaxed latency constraints
- Latency tolerant but hard deadline for individual transactions
  - λ can be high
  - δ is 0

Other parameters in performance contract

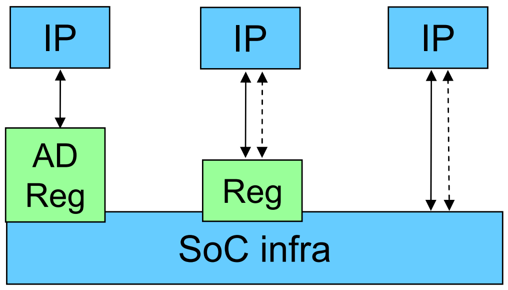
- Degree
  - Number of outstanding transactions allowed by SoC infrastructure
  - If more than degree outstanding: service not guaranteed
  - Forces IP to accept responses if it wants to maintain service
- Transaction sizes
  - Size of transactions issued under performance contract
  - For example specified as T_{max} and T_{min}
  - Distribution of transaction sizes

Contract parameters

Parameter	Description
σ	Workload burstiness
ρ	Workload long term average bandwidth
δ	Latency burstiness
λ	Long term average latency
degree	Maximum number of outstanding transactions
Transaction size	Transaction sizes (incl. distribution)

Impact on interface protocols

- Distinguish multiple flows going over single hardware interface
  - To allow explicit regulation of flows if IP does not comply
  - To enable use of flow information in SoC infrastructure for QoS
- Two mechanisms
  - Address discrimination (AD)
  - Explicit identification of flows



- Extend interface protocols with optional flow identification
  - So that different flows on interface can be distinguished
  - Extension of AXI / OCP with flowID

Next steps

- Standardization of performance contract semantics
  - For modeling IP communication requirements
  - For developing SoC performance methodology
- Deliver IP modules with performance contracts
  - Captured in data sheet of IP module
- Tools for automated characterization of IP modules
  - Automatically derive performance contract for IP module
- Performance analysis tools
  - Performance analysis of SoC infrastructure
  - Take performance contracts as input

Conclusion

- Performance contract decouples IP modules and SoC infrastructure
- Key concept for modular integration of MPSoCs
- Central concept in SoC performance methodology
  - Facilitates use of formal techniques for SoC performance analysis
- Need for further standardization
  - To enable reuse of IP modules across companies
  - To enable tool development by EDA vendors

Acknowledgements

- Tomas Henriksson
- Axel Jantsch (KTH Sweden)
- Alistair Bruce (ARM Ltd)
- And many others …