

# Applying Open Standards to FPGA IP Interfaces

Shepard Siegel



**11<sup>th</sup> Annual Workshop on  
High Performance Embedded Computing  
MIT Lincoln Laboratory  
September 18-20, 2007**

- **Components must Travel**

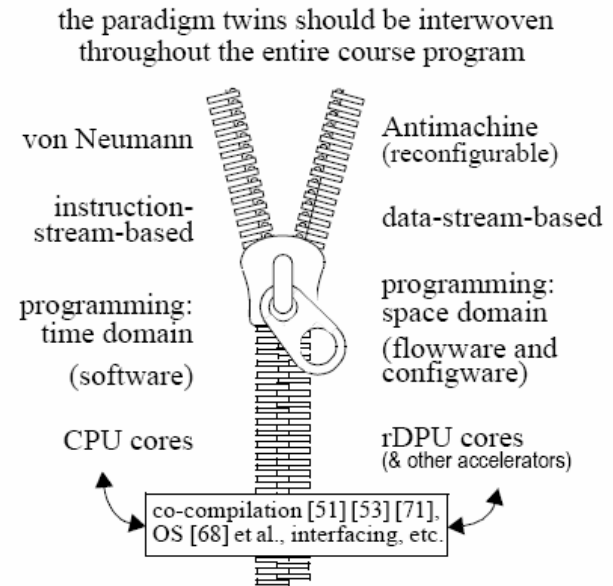
- Component-based design
- Interface before Implementation
- Nearly continuous evolution

- **Separation of Concerns**

- Computation ↔ Communication
- Specifier ↔ Implementer
- Application ↔ Architecture ↔ Implementation

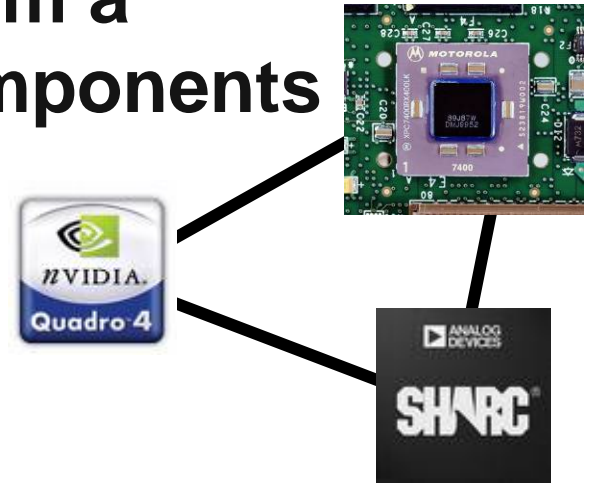
- **“The von Neumann Syndrome”** [Reiner Hartenstein 2007]

- Changing algorithms
- Changing resources
- No longer just centralized, serialized, program text!



- **Ability to compose solutions from a collection of heterogeneous components**

- Correct by Composition
- Exploit Processor Diversity



- **Constraints upon how components interact can improve interoperability**

- What the interface can “reveal”
- How the interface behaves - Interface semantics

- **Well-Defined Interoperability**
- **Bias Toward the Application**
- **Clearly Defined Choices**
- **Composition**
  - Completeness
  - Composition
  - Linear-Effort

- **Concerns**

- Throughput
- Latency
- Cost
- Space, weight and power
- Time-to-solution
- Reuse, scalability, tech refresh



**Commonly  
addressed**

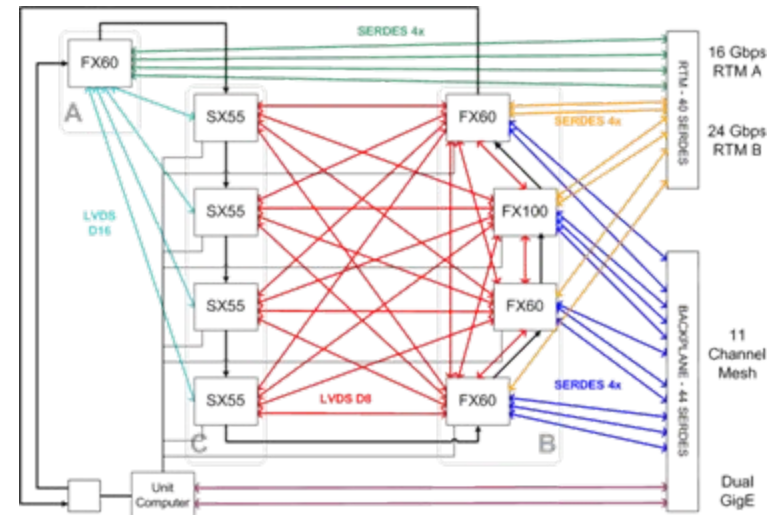
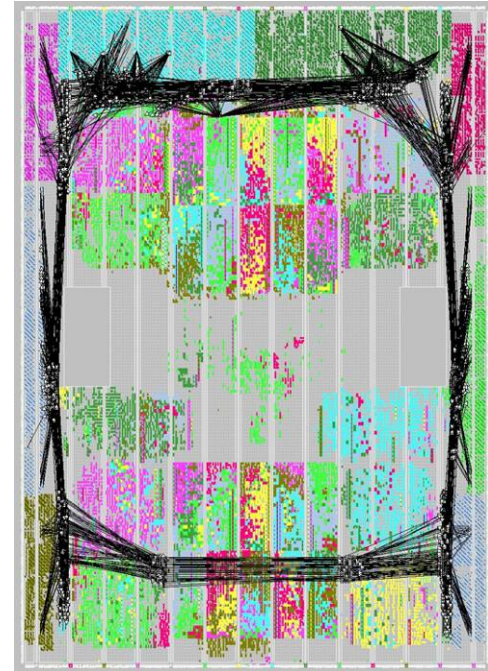


**Historically  
2<sup>nd</sup> - order**

- **A Multi-chip, board, system focus**

- More layers of physical hierarchy
- A pure SoC approach can be less-aware

- **Scale is Important**
  - Amplifies platform leverage
- **System-on-Chip (SoC)**
  - Collection of IP cores
  - Often for a single-application
  - Often (helpful) constraints apply
- **Systems of Chips, Boards, Systems**
  - Collection of components on a platform
  - Often application-agnostic



- **ASIC and SoC involve the mapping of applications to architecture at design-time**
- **Reconfigurable Computing (hetero core, FPGA) can add value by delaying the platform mapping**
- **Scenario-Oriented Computing [Pieper 2007] finds a balance. Merit is measured by components and their interactions**

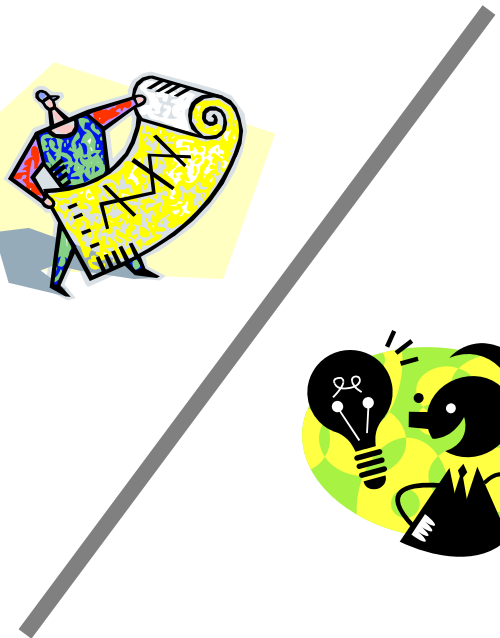
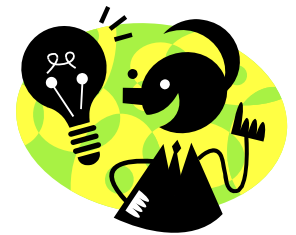
- **“Interface before Implementation”**  
is a proven software methodology
- **Now is time of great change for expressing functionality, especially for hardware (e.g. FPGA) design**
  - RTL has run out of runway
  - Reasoning about clock ticks at the system level is crazy!
  - Excellent candidates for a “Hardware Imagination Language” are emerging
    - Bluespec
  - Wish to avoid taking any action that would prevent the use of a particular ESL
- **Open, Standard Interfaces allow us this partition**

- **Why**

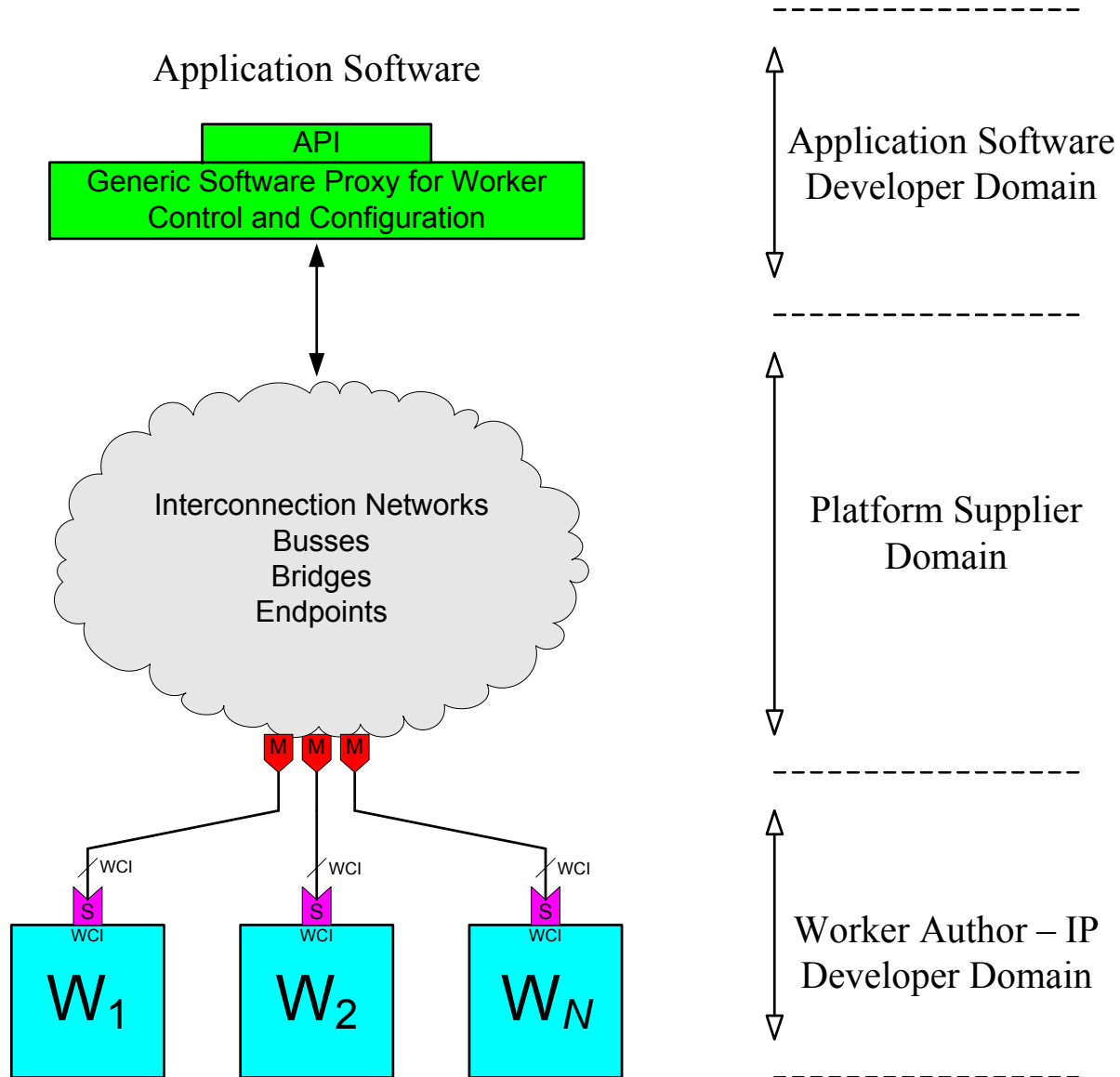
- Reason about and solve simpler problems
- Compose them to attack grand challenge
- Scalability

- **Separation of**

- Communication and Computation
- Application and Architecture
- Architecture and Implementation
- Specifier and Implementer



# Domains of Responsibility



- **Allow the addition of a component without changing the relevant behavior of an existing assembly [Jantsch 2004]**

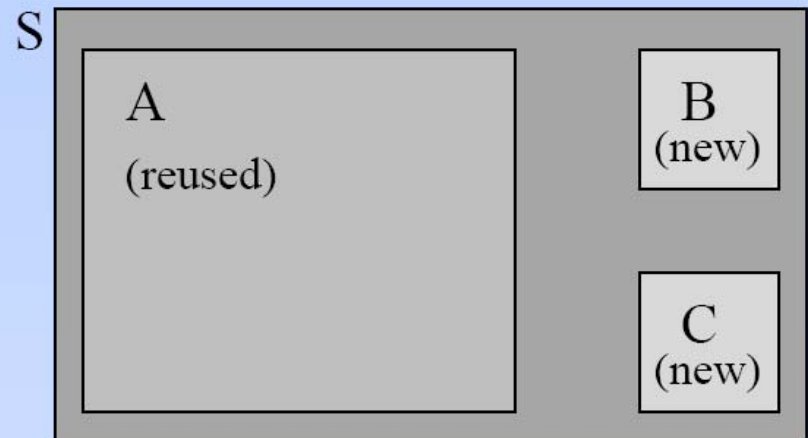
## Arbitrary Composability

Given a set of components  $C$  and combinators  $O$ .

Let  $A_1$  be a component assemblage.  $(C, O)$  is **arbitrary composable** if

$$A_1 + B \Rightarrow A_2$$

can be done for any  $B \in C, + \in O$  **without changing the relevant behaviour of  $A_1$ .**



- **Design process where effort depends on number of (not size of) assemblies [Jantsch 2004]**
  - Related to interaction-complexity

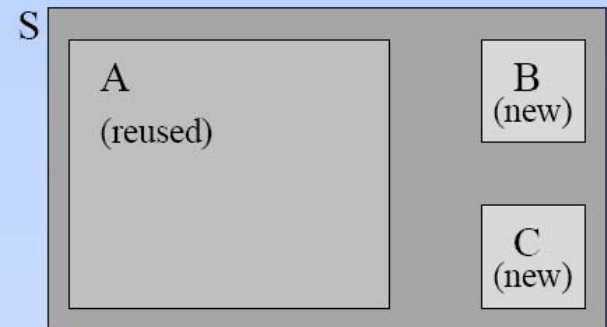
## Linear Effort Property

Given a set of components  $C$  and combinators  $O$ .

Let  $A_1, \dots, A_n$  be component assemblages.

A **design process** using  $C$  and  $O$  to build a system has the **linear effort property** if  $A_1, \dots, A_n$  can be integrated into a system  $S$  with an effort dependent on  $n$  but not on the size of the assemblages:  $I_{\text{effort}}(n)$ .

Total design effort for  $S$  is



$$\text{Deffort}(S) = \text{Deffort}(A_1) + \dots + \text{Deffort}(A_n) + I_{\text{effort}}(n)$$

- **An Open Interface Standard?**
  - Yes
- **An IDL?**
  - Yes
- **A Platform?**
  - Yes
- **A Bus?**
  - No
- **Weight of Implementation**
  - None explicitly, but
- **Composable in a latency-insensitive fashion?**
  - Yes: Transactions are rule-based interface methods



- **An OCP profile is shorthand for the configuration parameters describing an interface on a module**
- **Why?**
  - There are 91 configuration parameters in v2.2
    - Sensible defaults, but lots of “knobs”
    - Too much expression for most users
    - Little to no assurance of interoperability
  - Some combinations don't have tie-off rules
    - Wish to avoid this situation
- **Profiles introduced in v2 of the spec to**
  - Reduce the risk of incompatibility
  - Shorten the learning curve
  - Simplify the logic between dissimilar protocols

- **Define a small number of OCP profiles**
  - Like-to-Like Master/Slave is guaranteed to work
  - You can always add specific interface definitions
- **Superset Functionality**
  - Allow for the inclusion of most practical features
  - Synthesis, MAP, and PAR will trim unused logic
  - Prepares for tomorrow's scenarios today
- **Allow the addition of custom signaling**
  - Remain fully-compliant to OCP-IP specification
  - Highest-performance in like-to-like scenarios
  - Not a proprietary layer atop an open standard

- **Well-Defined Interoperability**
  - A full range of adaptation which includes:
    - Like-Like (glueless)
    - Tie-off / Ignore (glueless + tieoffs)
    - Well-Defined Adapters (canned adaptation)
    - Advice from one side to the other (generated IP)
- **Bias Towards the Application (Worker)**
  - Worker has it easy, specifies needs
  - Infrastructure adapts, copes, or complains
- **Clearly Defined Choices**
  - WIPs define *Profile Attributes*
  - Explicit map to OCP Config Parameters and Signals

- **Worker Control Interface (WCI)**

- Control and Configuration

- **Worker Streaming Interface (WSI)**

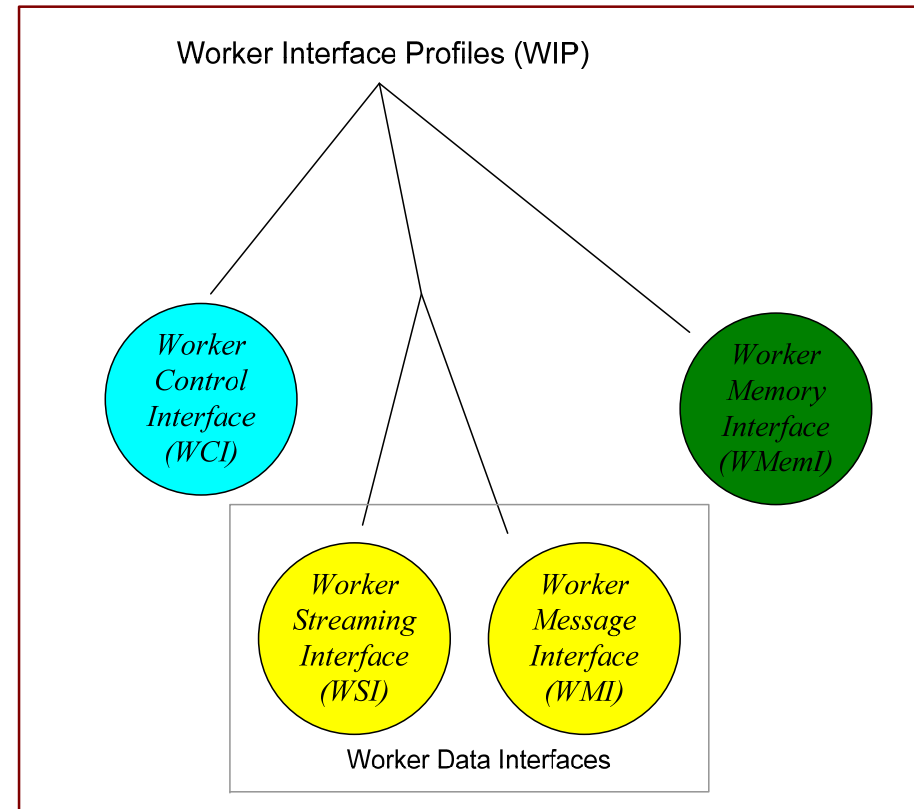
- Streaming Push-Only I/F

- **Worker Message Interface (WMI)**

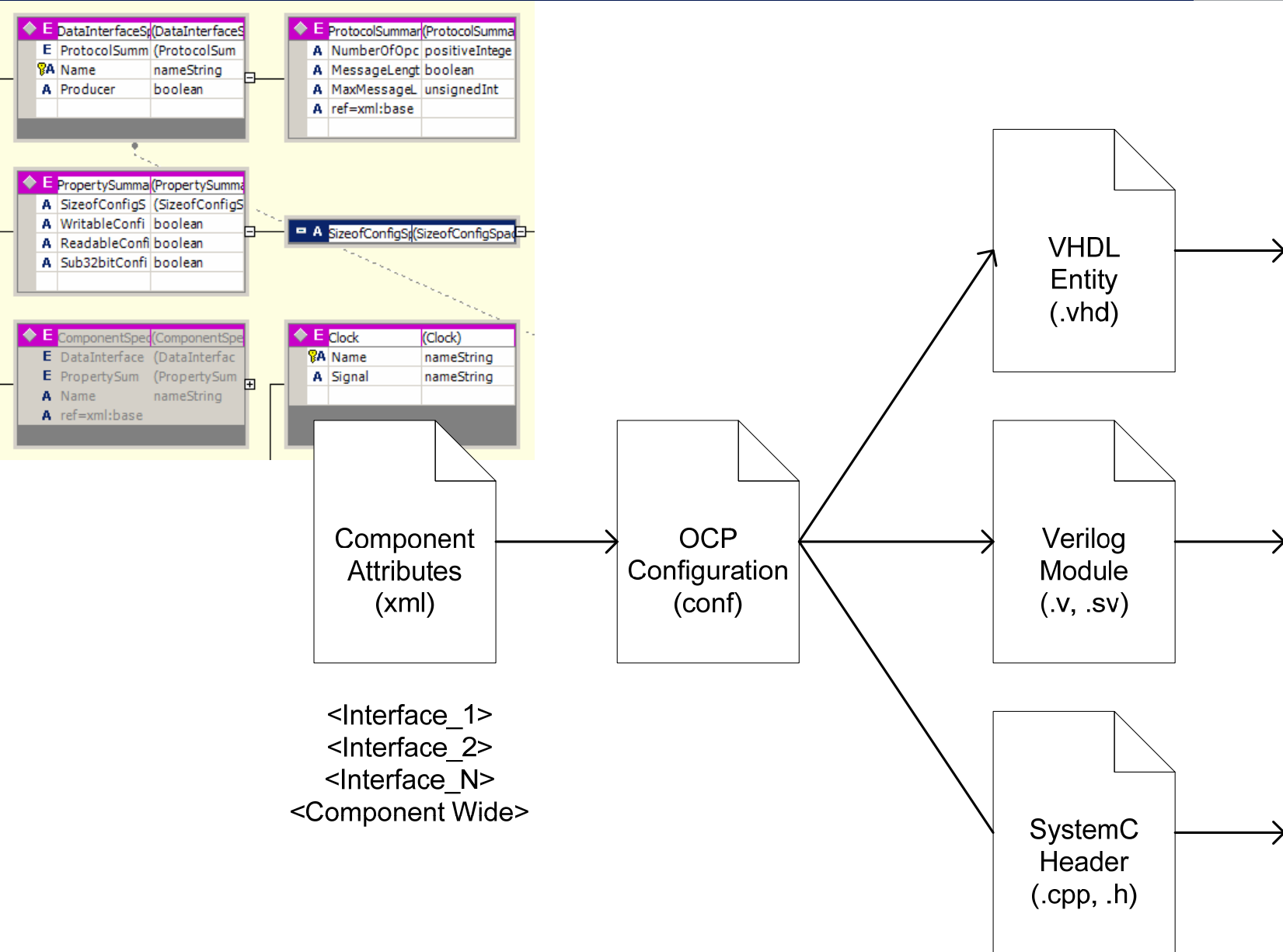
- Addressable or Streaming I/F

- **Worker Memory Interface (WMemI)**

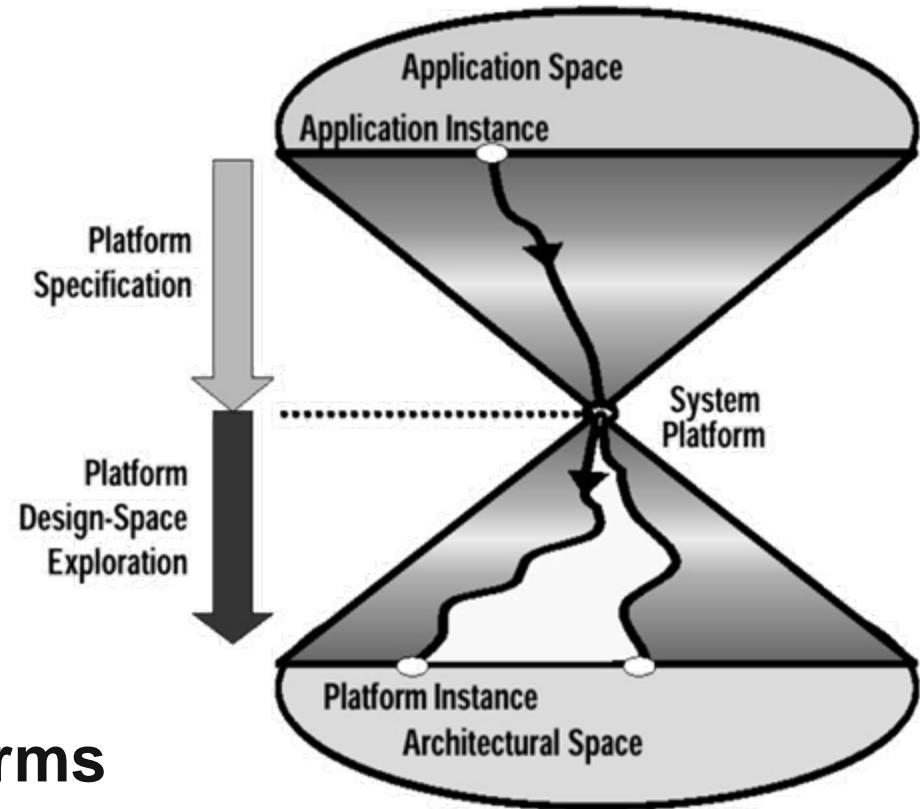
- Interface to attached Memory



# XML Transformations

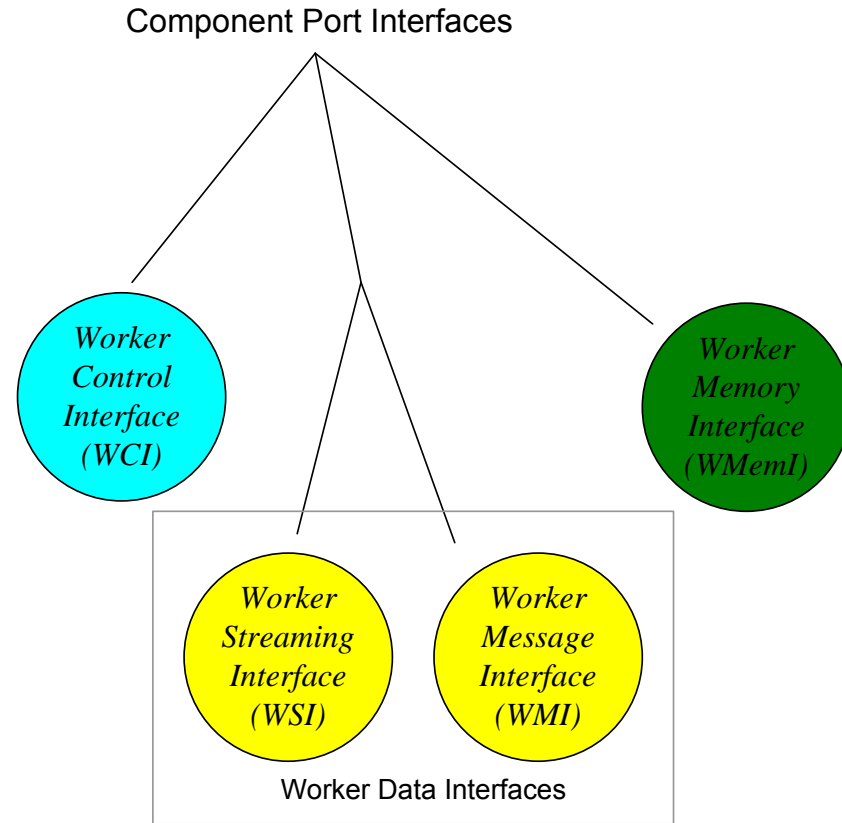


- **Vital concept to empower and exploit “separation of concerns” [ASV 2002]**
  - M+N effort to cover MxN space
    - Effort is additive
    - Effect is multiplicative
- **Consider**
  - Top-down
  - Bottom-up
  - Middle-out
- **There can be multiple platforms**
  - Amplifies the M+N → MxN leverage



- **Most IP “worker” interfaces fall into these categories**

- Control and configuration
- Streaming data
- Message data
- Memory interaction

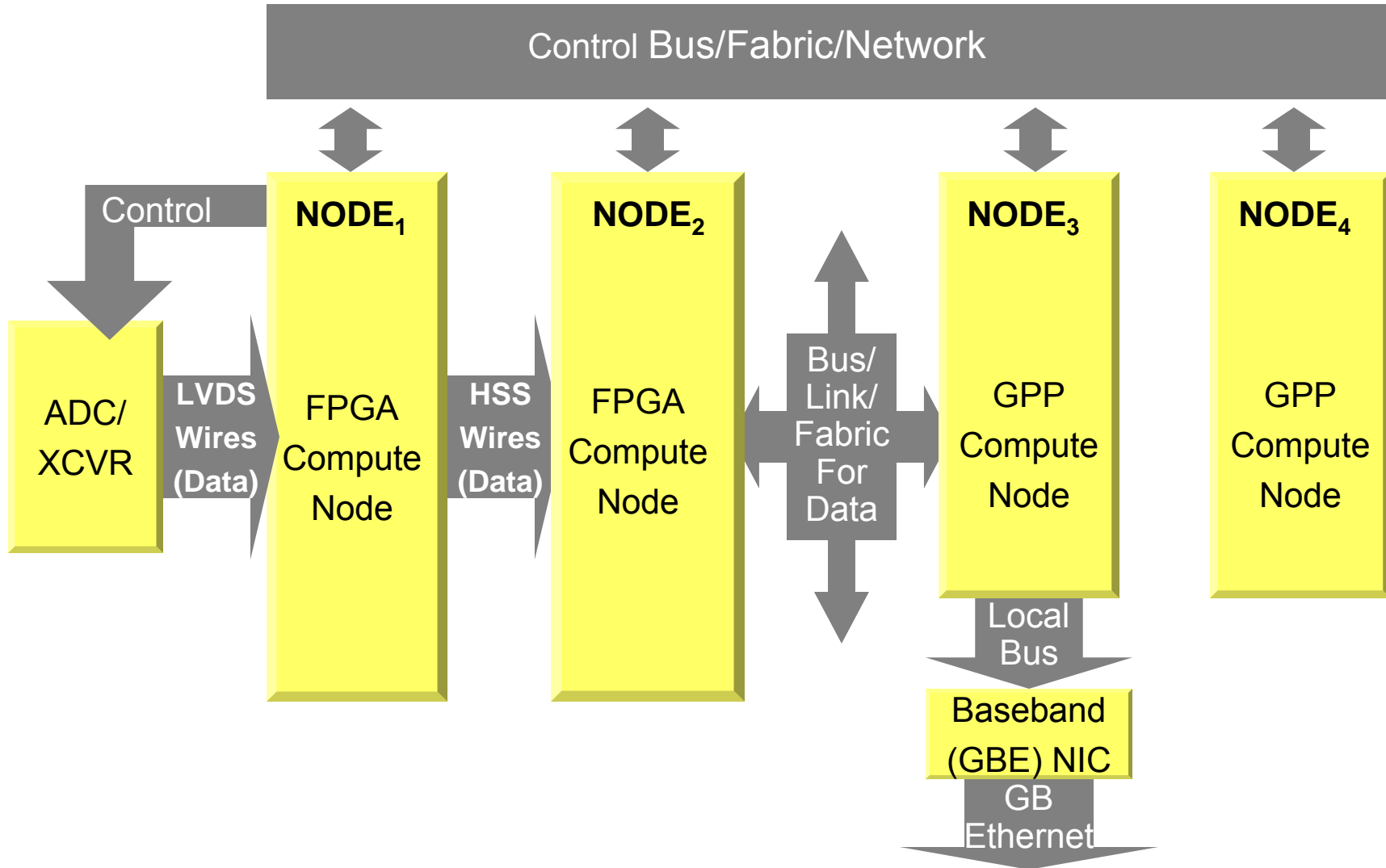


- **Worker Interface Profiles (WIPs)**

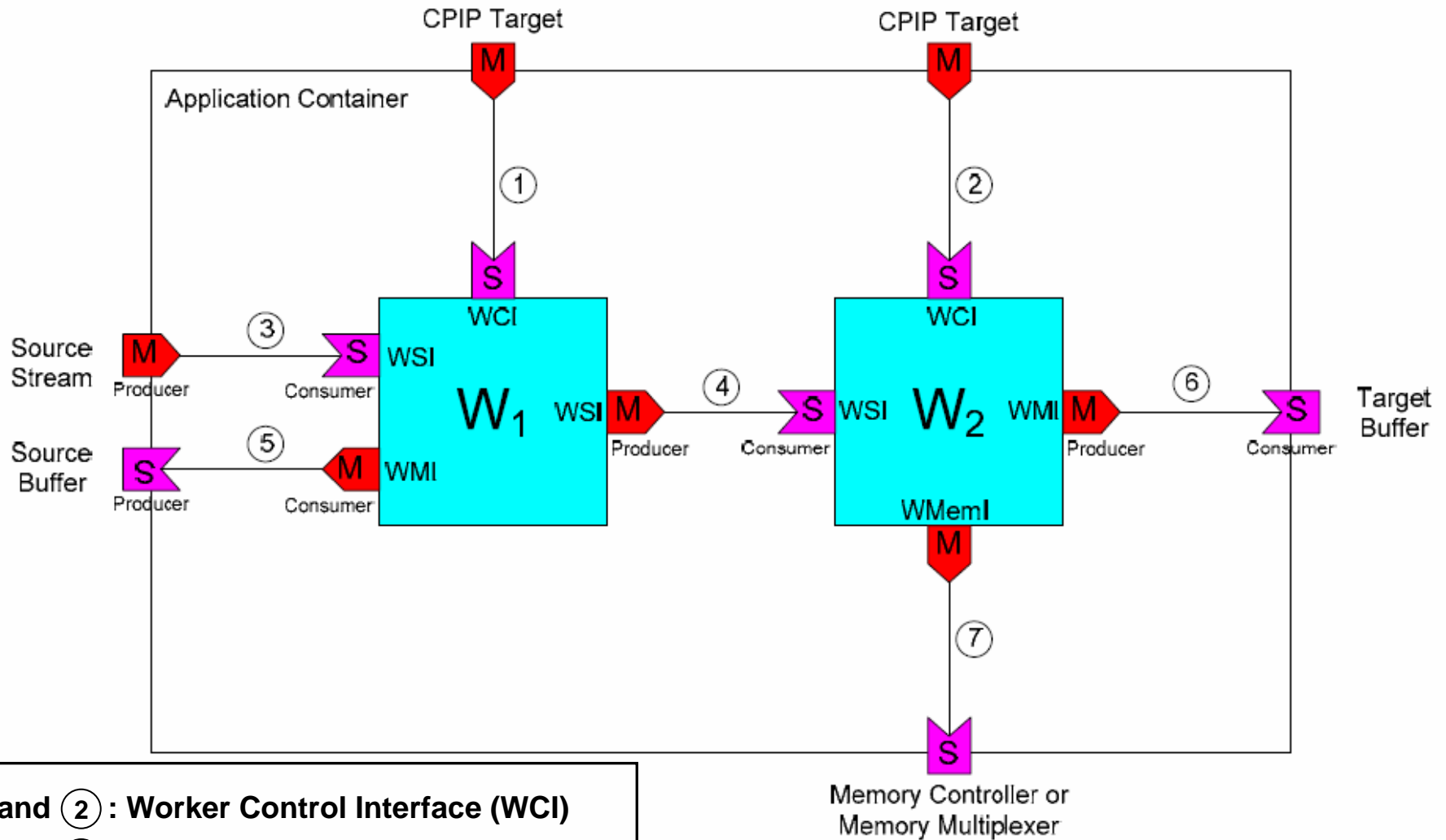
- Well-Defined Adaptability and Interoperability
- Independence from Specific Implementations
- HDL Language-Agnostic, HDL Language-Neutral
- Semantics hold under Hierarchy

- **Less coupling between IPs**
  - Workers “work”
  - Infrastructure “copes”
- **Facilitate interchanging components**
- **Provide for (virtual) application-specific platform**
  - Continuous application evolution
  - Continuous architecture evolution
  - Continuous verification refinement

# Heterogeneous System of Processors Example



# A "Worker" Component



- ① and ② : Worker Control Interface (WCI)
- ③ and ④ : Worker Stream Interface (WSI)
- ⑤ and ⑥ : Worker Message Interface (WMI)
- ⑦ : Worker Memory Interface (WMemI)

- [Bloomfield 2002] Bloomfield, John 2002 “*Partitioning Computational Tasks...*,” HPEC 2002  
<http://www.ll.mit.edu/HPEC/agendas/proc02/pdfs/7.3-bloomfield.pdf>
- [DeMan 2002] De Man, Hugo 2002 “*Design of Microelectronic Systems,*” HJ94 Lecture 2  
<http://homes.esat.kuleuven.be/~iverbauw/Courses/HJ94/index.html>
- [Evans 2004] Evans, Eric 2004 “*Domain Driven Design,*” Addison Wesley, Chapter 10  
<http://domaindrivendesign.org/>
- [Feldkamp 1983] Feldkamp, L.A. 1983 “*Practical Cone-Beam Algorithm,*” Optical Society of America  
<http://www.osa.org>
- [Gabriel 1983] Gabriel, Steven and Bennett, Phillip “*System for Spatially Transforming Images,*”  
United States patents 4,472,732, 4,463,372 and 4,468,688. (Assigned to Ampex Corporation)
- [Hartenstein 2007] Reiner Hartenstein “*The von Neumann Syndrome*”  
<http://hartenstein.de>
- [Jantsch 2004] Jantsch, Axel 2004 “*Networks on Chip,*” Network on Chip Seminar  
<http://web.it.kth.se/~axel/presentations/2004/LiTH-I.pdf>
- [Pieper 2007] S. Pieper, J. Paul, M. Schulte 2007 “*A New Era of Performance Evaluation,*” IEEE  
Computer September 2007 <http://www.computer.org>
- [ASV 2002] Sangiovanni-Vincentelli, Alberto Feb 2002 “*Defining Platform-Based Design,*” EETimes  
<http://www.gigascale.org/pubs/141/platformv7eetimes.pdf>