



NOC OVERVIEW

Introduction

- The trend toward multi-core processing chips is now a well established one
 - Mass market production of dual-core and quad-core processor chips
 - Trend toward massive multi-core chips based on much simpler cores (e.g. Sun UltraSparc T1, Nvidia 8800 GT/GTX GPUs)
 - Heterogeneous multi-core chips (processors plus accelerators) proposed (mostly for the embedded market)
- Beyond a certain number of cores (say, 8 to 16), an on-chip network with point-to-point links becomes necessary to interconnect cores, cache banks and memory controllers among them (and possibly with on-chip routers for external communication as well) without the constraints imposed by buses.

The Road to Multicore System (I)

- 2001: IBM release its first commercial Dual-Core CPU, the **IBM POWER4**
- 2002: **POWER4+** is released
- 2004: IBM release the Dual-Core **IBM POWER5**
- 2005: **POWERPC 970MP** (used in the Apple PowerMac G5), **Cell Microprocessor** (8-Cores), **Xbox 360 CPU** (3-Cores)
- 2003: HP release the **HP PA-RISC 8800** Dual-Core CPU



The Road to Multicore System (II)

- 2004: Sun Microsystems releases the **Sun UltraSPARC IV Dual-Core CPU**
- 2005: **Sun UltraSPARC T1** (codename **Niagara**) 8-Core CPU
- 2007: Sun Rock, **Niagara 2** 8-Core CPUs (in the pipeline)
- 2005: **AMD Opteron** (Egypt, Italy, Denmark), **Athlon 64 x2** (Manchester, Toledo)
- 2006: AMD Athlon 64 x2 (Windsor), Athlon **Turion 64 x2** (Taylor)
- 2007/8: AMD Deerhound, Greyhound, Zamora **Quad-Core CPU** (in the pipeline)



The Road to Multicore System (III)

- 2005: **Intel Pentium D** (Smithfield, Presler)
- 2006: **Intel Core Duo** (Yonah), **Core 2 Duo** (Conroe, Merom, Woodcrest)
- 2006: intel fabs **80-core teraflop** processor
- 2007: intel Kentsfield, Clovertown **Quad-Core CPUs** (in the pipeline)
- 2009: intel Yorkfield, **8-Core CPU** (45nm) (in the pipeline)



Multi-Core Embedded Devices

- Embedded devices usually run just one or a few applications
- General-purpose processors are not suitable for running those applications because they are either too slow or too power hungry
- Ultra low-power processors combined with hardware accelerators are the preferred choice for most designers
- Recent designs for the embedded market use multi-core processors to reduce power consumption
- Future embedded systems may use a large number of heterogeneous cores to deliver the best tradeoff between performance and power consumption

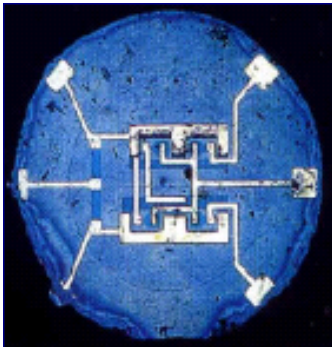


Multi-Core Chips are the Way of the Future !!!

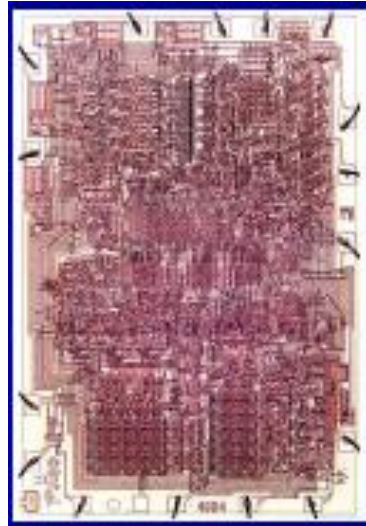
But.....

**Could the INTERCONNECT stand in their
way ???**

A history perspective



First integrated circuit
Fairchild Semiconductor
1959

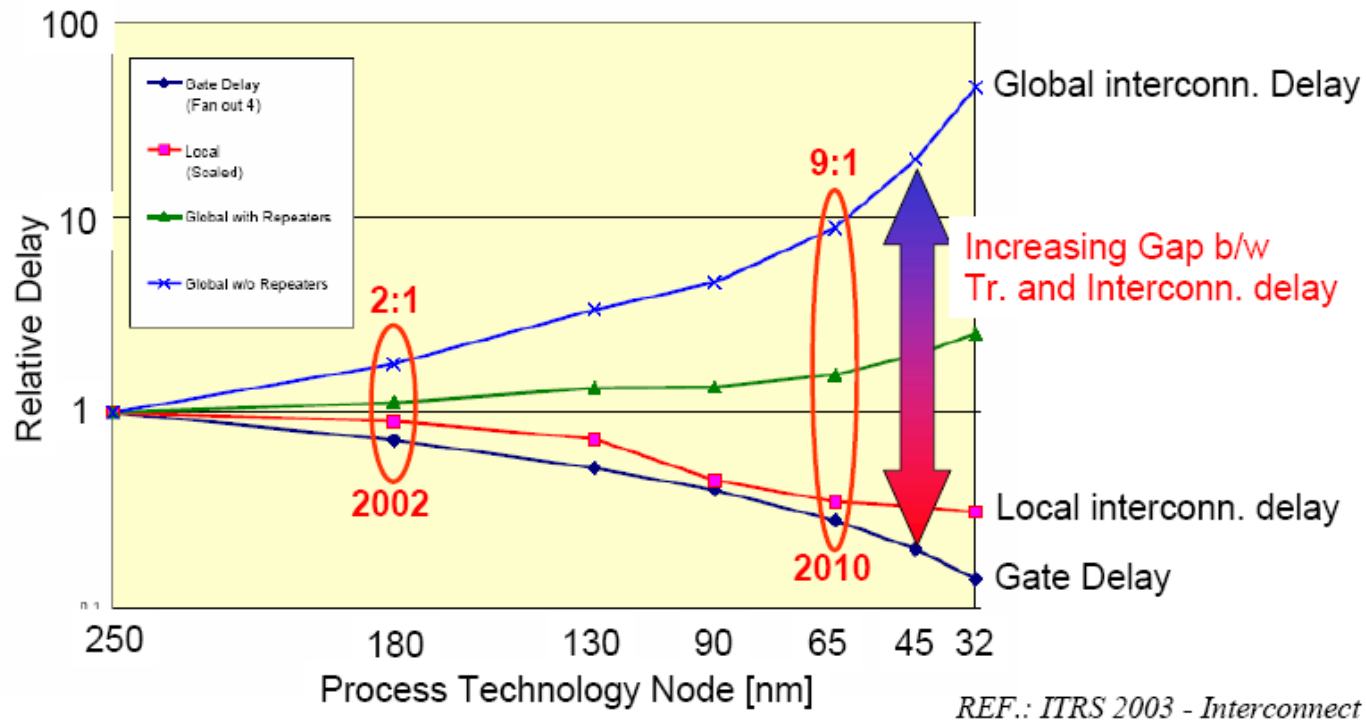


First microprocessor
Intel 4004
1971



Pentium4
Intel Corporation
2002

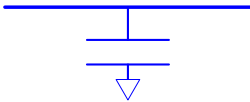
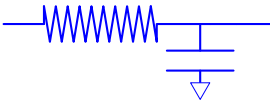
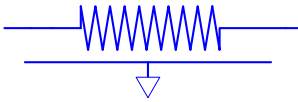
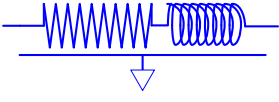
Trend in Interconnect Delay



■ Interconnect delay dominates gate delay

- Global interconnect delay continuously increasing
- Need multiple clock cycles to cross chip die
- Limits the performance of microprocessors

Interconnect model

Technology	$> 1\mu\text{m}$	$1.0 \sim 0.5\mu\text{m}$	$0.5 \sim 0.25\mu\text{m}$	$< 0.25\mu\text{m}$
Resistance	Ignord	Lumped	Distributed	Distributed
Capacitance	Area Capacitance	Area, 2-D fringing	2/3-D fringing, Lateral coupling	Lateral coupling dominant
Inductance	Ignored	Ignored	Ignored	Important
Circuit model	Lumped C	Lumped RC	Distributed RC	Distributed RC
Circuit Symbol				

- Significant inductance effect with technology scaling
- The **INTERCONNECT** is most sensitive on process variation
 - Clock network and global wires span the whole chip area
 - Interconnect delay affects the entire system performance

Interconnect Networks

■ Power distribution networks

- consume about 30% on-chip metal
- IR, Ldi/dt noise
- RLC resonances

■ Clock distribution networks

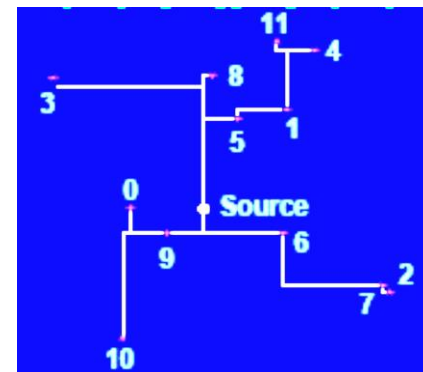
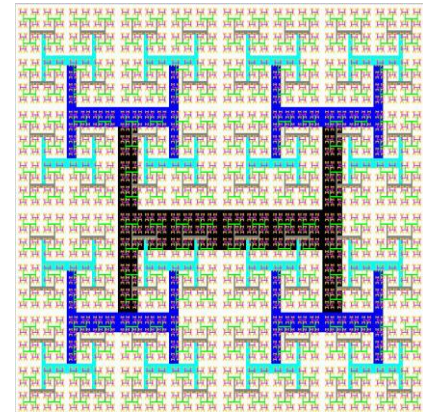
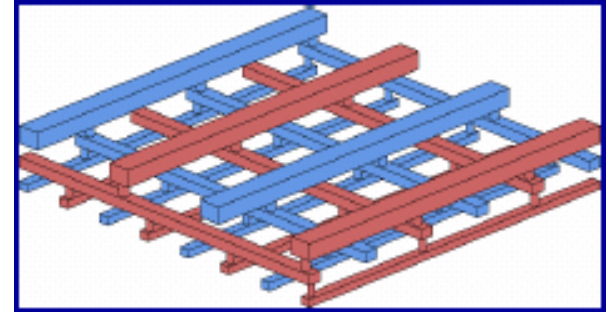
- Consume up to 70% of the total power
- Clock skew, jitter

■ Signals with multiple fan-out

■ Large global busses

■ Interconnect networks have become increasingly complicated with greater integration

- Accurate and efficient models are required



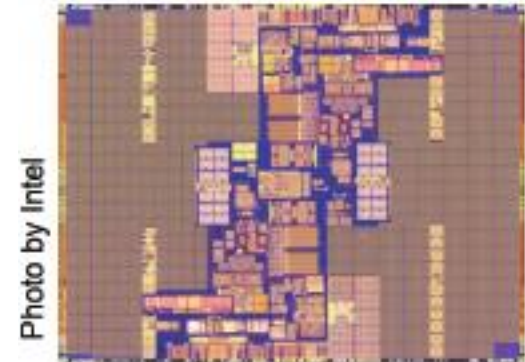
The Billion Transistor Era

- Feature sizes diminishing RAPIDLY into nanometer regime
- Transistor density skyrocketing
- Gate delay are scaling down
- What's about Global wiring delay?

- As wire cross sections decrease, resistance increases!!

- Interconnect are also an issue in terms of **AREA, POWER, and RELIABILITY**

- **Process-Independent & reliable Interconnect Architecture is needed in the billion transistor era.**



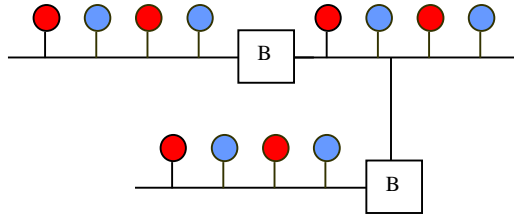
Intel Itanium 2
(Codename Montecito)
1.7 BILLION
transistors per die!



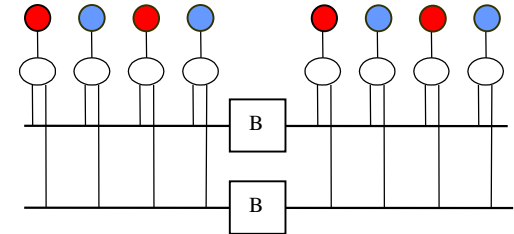
**The interconnect delay can
no longer be ignored !!!!**

Typical solution : Bus

Segmented
Bus



Multi-Level
Segmented
Bus



Original bus features:

- One transaction at a time
- Central Arbiter
- Limited bandwidth
- Synchronous
- Low cost

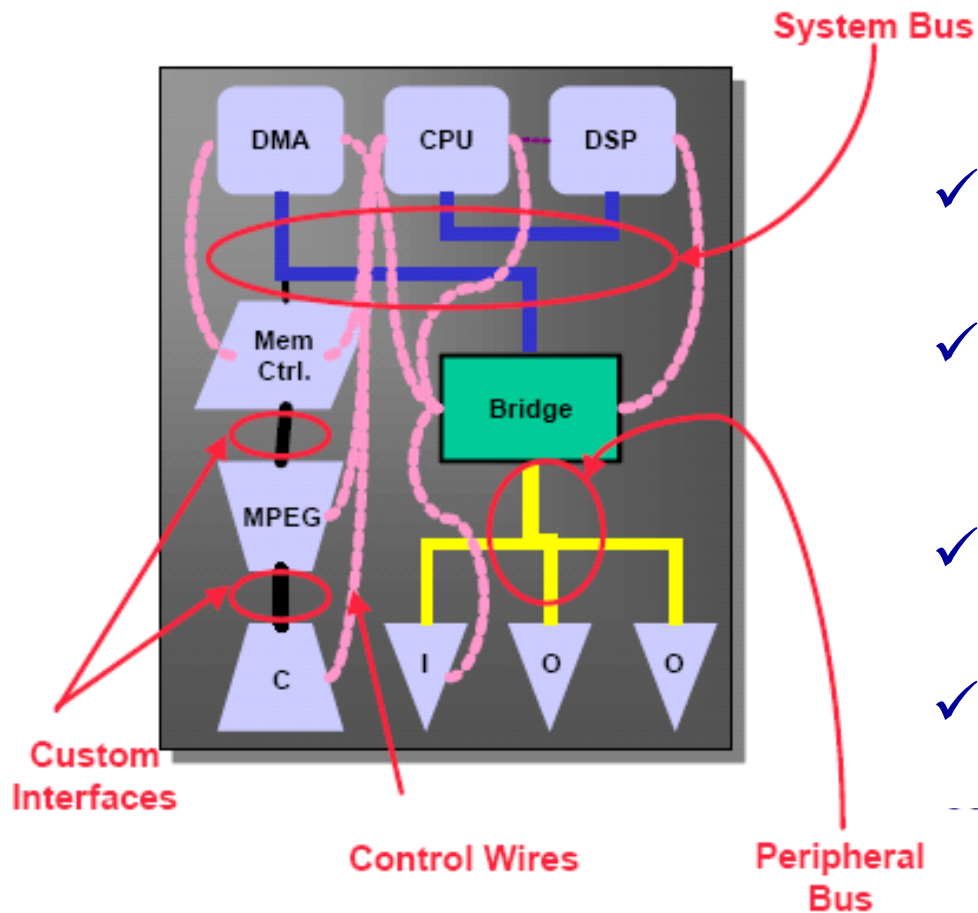
New features:

- Versatile bus architectures
- Pipelining capability
- Burst transfer
- Split transactions
- Transaction preemption and resume
- Transaction reordering...

Well-known industry solution:

- ARM → AMBA (Advanced Microcontroller Bus Architecture)
- Sonics → SiliconBackplane μ Network
- IBM → Core-Connect

Traditional SoC Nightmare



- ✓ *Variety of dedicated interfaces*
- ✓ *Poor separation between computation and communication*
- ✓ *Design Complexity*
- ✓ *Unpredictable performance*

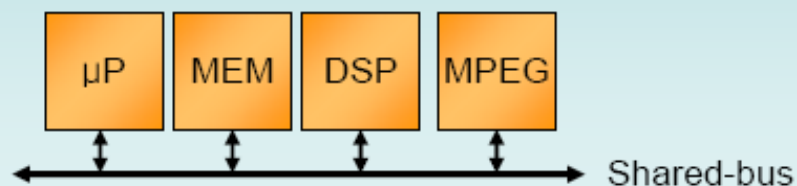
What do we need ??

- Replace Global wires with a **Resource-Constrained Network**
- Structured interconnect layout
- Electrical Properties **OPTIMIZED** and **WELL CONTROLLED**
- Scalable:
 - Don't want to change the way I design communication architecture even if BW requirement scale up exponentially
- Predictable:
 - Be able to expect (latency, bandwidth) and negotiate it
- Robust:
 - Keep going even if something is broken inside
- Efficient
 - Silicon is expensive, power is precious
- Easy:
 - To create, update, analyze, verify
- **NOCs are like IP Blocks for Wiring!**

Communication Centric

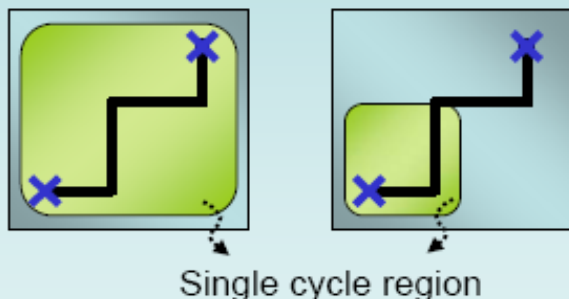
- **Yesterday: Computation-centric Design**

- Functions of IPs determines the application of the chip
- Communication was not a concern



- **Tomorrow: Communication-centric Design**

- Interconnection determines the performance of the system
- Wire-delay, Larger-bandwidth, Difficult to sync. clock



Tech.	180nm	45nm
Clock	1GHz	3GHz
1-cycle region	11mm	0.6mm

→ Needs **Systematic On-Chip Communication Structure**

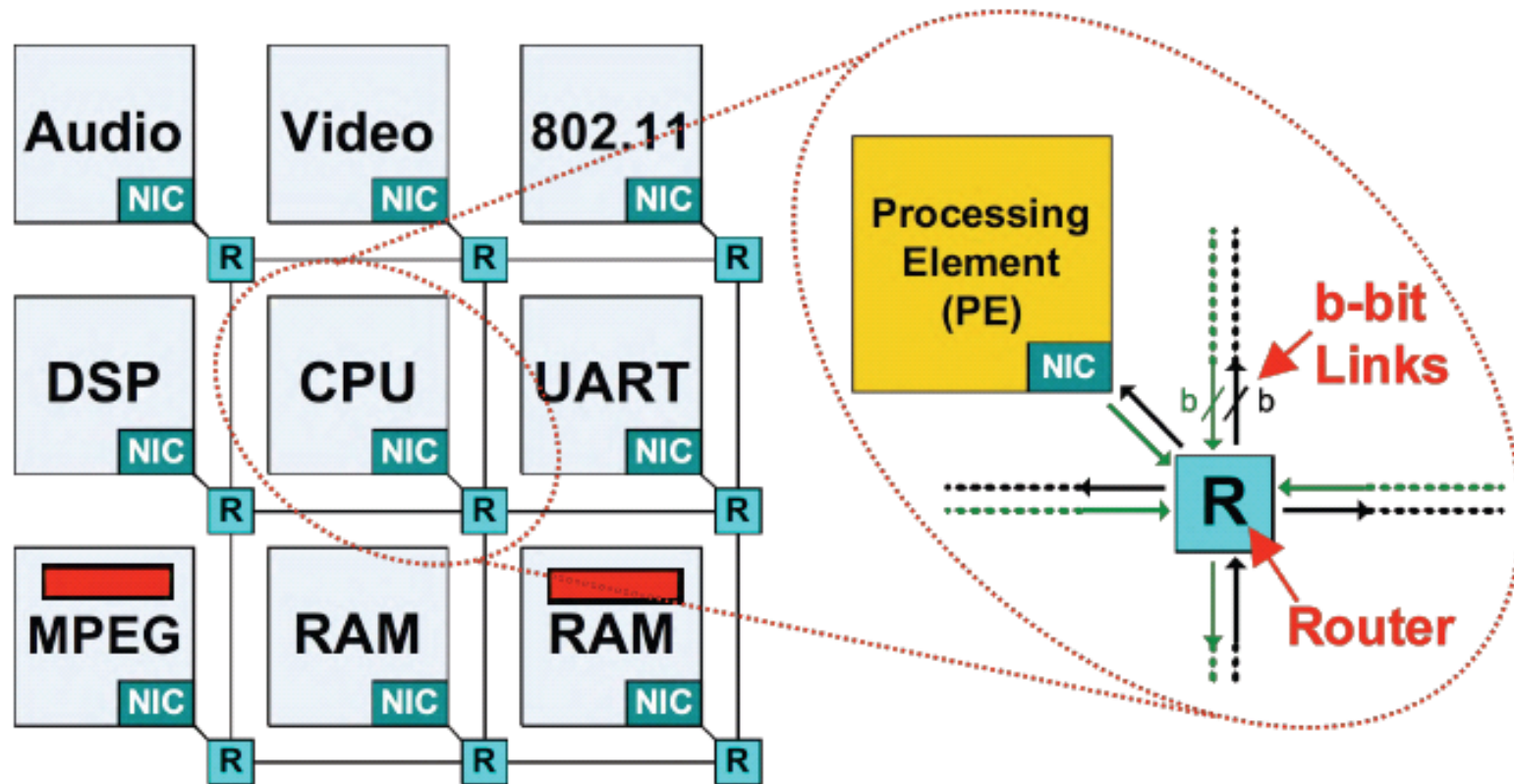
Solution – Network on Chip

- Networks are preferred over buses:
 - **Higher bandwidth**
 - **Concurrency, effective spatial reuse of resources**
 - **Higher levels of abstraction**
 - **Modularity - Design Productivity Improvement**
 - **Scalability**

Bus Pros & Cons		Network Pros & Cons	
Every unit attached adds parasitic capacitance, therefore electrical performance degrades with growth.	- +	Only point-to-point one-way wires are used, for all network sizes.	
Bus timing is difficult in a deep submicron process.	- +	Network wires can be pipelined because the network protocol is globally asynchronous.	
Bus testability is problematic and slow.	- +	Dedicated BIST is fast and complete.	
Bus arbiter delay grows with the number of masters. The arbiter is also instance-specific.	- +	Routing decisions are distributed and the same router is reinstanciated, for all network sizes.	
Bandwidth is limited and shared by all units attached.	- +	Aggregated bandwidth scales with the network size.	
Bus latency is zero once arbiter has granted control.	+ -	Internal network contention causes a small latency.	
The silicon cost of a bus is near zero.	+ -	The network has a significant silicon area.	
Any bus is almost directly compatible with most available IPs, including software running on CPUs.	+ -	Bus-oriented IPs need smart wrappers. Software needs clean synchronization in multiprocessor systems.	
The concepts are simple and well understood.	+ -	System designers need <i>reeducation</i> for new concepts.	

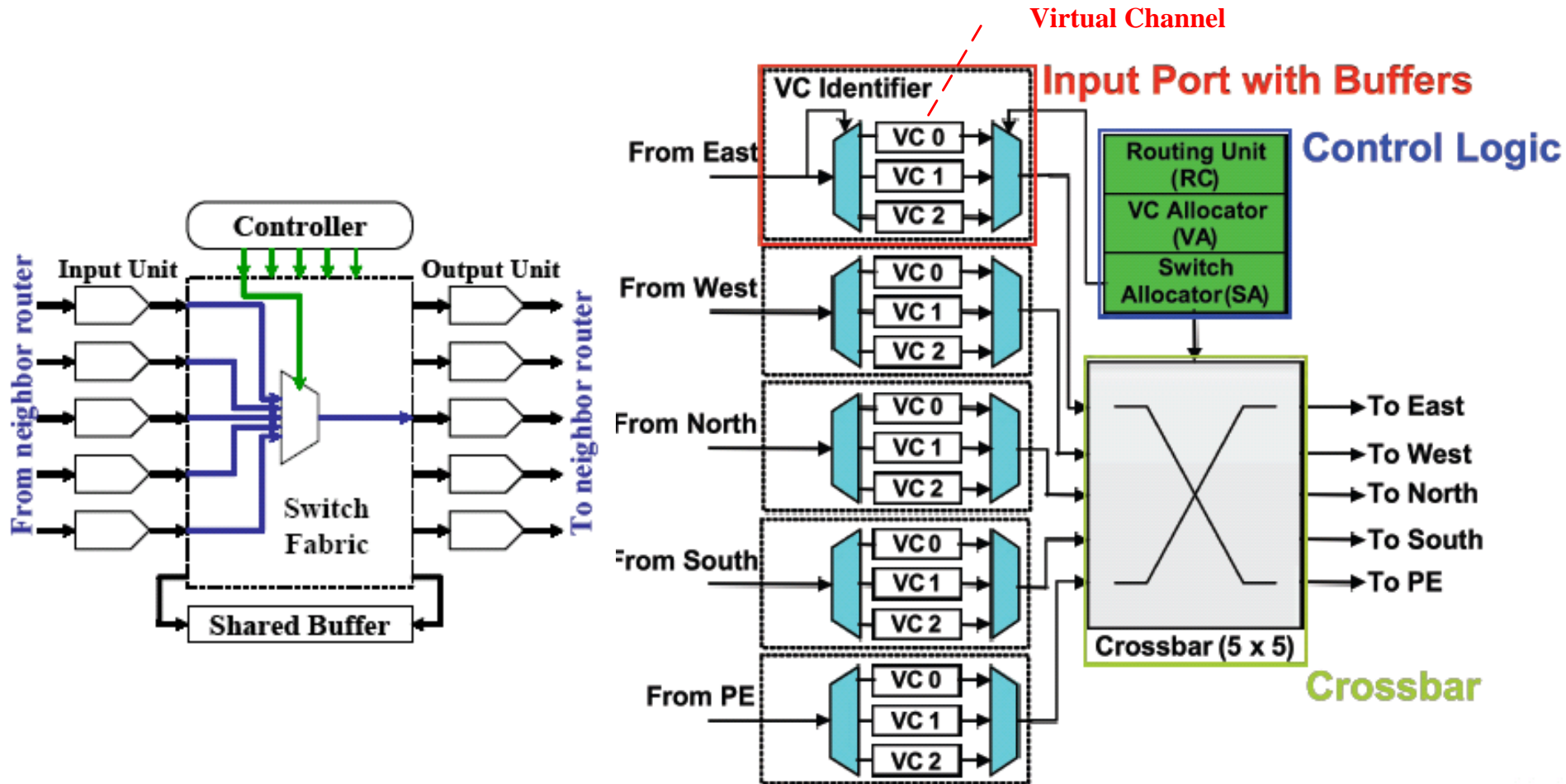
The overview of Network on Chip (NOC)

Processing Elements (PEs) interconnected via a **packet-based** network



- A NoC consists of resources and network. Network elements are switches, channels and resource-network interfaces

A conventional NOC router

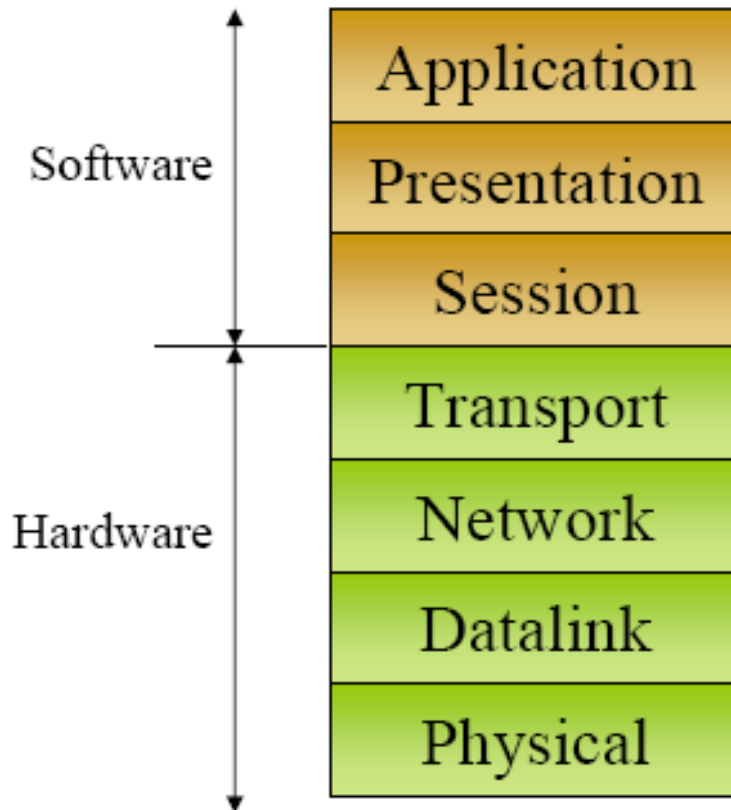


- A resource is a computation or storage unit. Switches route and buffer messages between resources.

Basic Requirement for NOC Design

- **Reuse of intellectual property blocks**
 - Best performance/energy ratio
 - Best mapping to application characteristics
- **Reuse of hardware (and architecture) and NOC platform**
 - Best complexity/cost and performance/cost ratio
 - Only way to even dream of achieving time to profit requirement
- **Reuse of design method and tool**
 - Only way to deal with heterogeneous application set
- **Partitioning of problems**
 - By encapsulation and hiding of complexity of the overall system
- **Different QoS must be supported**
 - Bandwidth
 - Latency

The OSI Reference Model Applied to NOCs



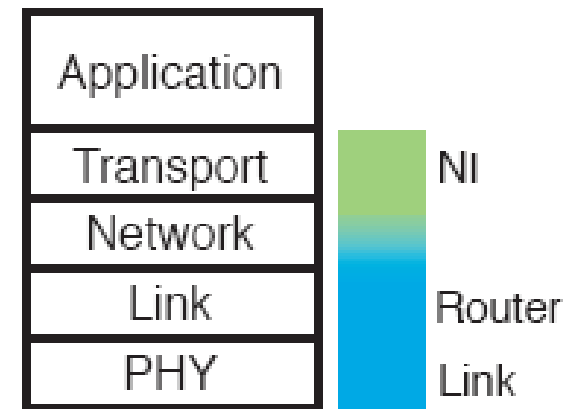
- Reference model for wired and wireless protocol design —Also useful guide for for conception and decomposition of NOCs
- Layered approach allows for orthogonalization of concerns and decomposition of constraints
- Not required to implement all layers of the stack
 - depends upon application needs and technology
- Layered structure must not necessarily be maintained in final implementation
 - e.g., multiple layers can be merged in implementation optimization

The OSI Reference Model Applied to NOCs

- **Physical Layer:** The NOC physical layer protocols define such things as signal voltages, timing, bus width, and pulse shape. A particular concern at this layer is the synchronization of signals
- **Data link Layer:** is responsible for reliable *transfer of data* over the physical link, and may include error detection and correction functions
- **Network Layer:** provides a *topology-independent view* of the end-to-end communication to the upper level protocol layers
- **Transport Layer** protocols establish and *maintain end-to-end connections*. This abstraction hides the topology of the network, and the implementation of the links that make up the network

The OSI Reference Model Applied to NOCs

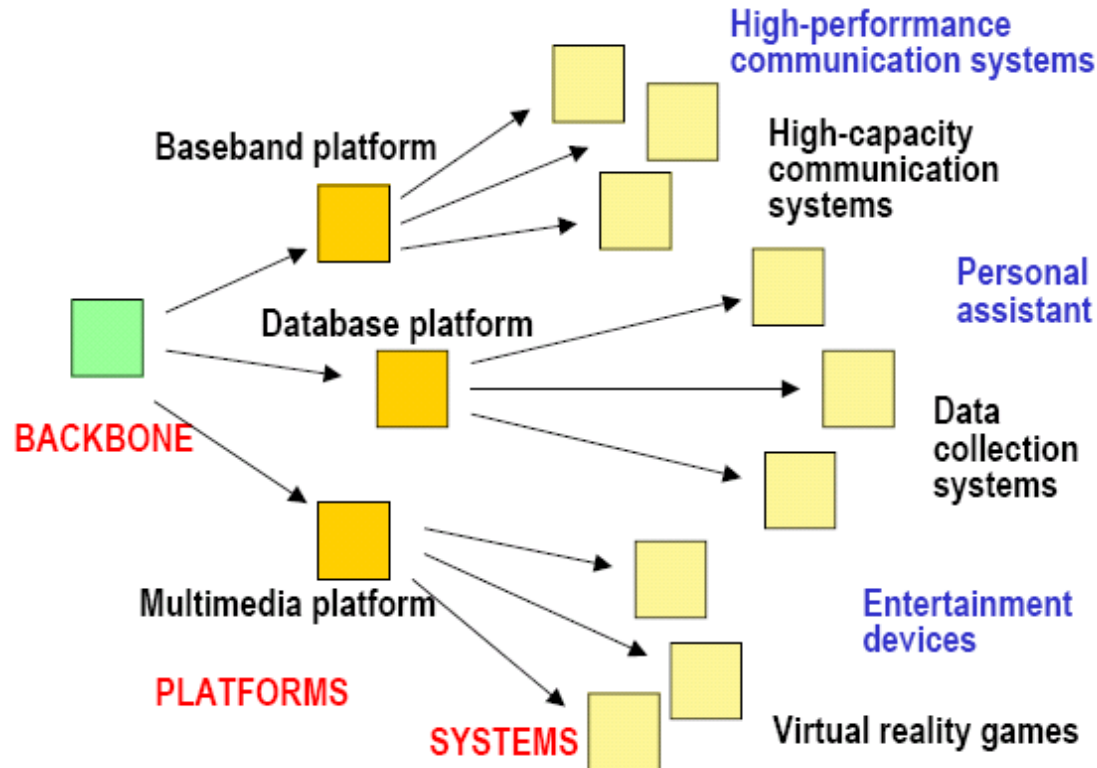
- **Session Layer** protocols *add state* to the end-to-end connections provided by the transport layer. A common session protocol is synchronous messaging, which requires that the sending and receiving components rendezvous as the message is passed
- **Presentation Layer** is concerned with the representation of data within messages. Protocols at this level *convert data* into compatible formats
- **Application Layer** would *define a function* that does exactly this by utilizing the functions defined at lower stack layers



Design Challenges for NOC

- Most of the design challenges for on-chip networks are very similar to the network designers:
 - Selection of a suitable topology and routing algorithm
 - Definition of efficient flow control and switching techniques
 - Designing a compact and fast router
 - Designing flexible and efficient network interfaces
 - Providing support for fault tolerance
 - Plus some additional support depending on the application area: collective communications, QoS, congestion management

Development of NOC based Systems



- The NoC systems use a specific architectural approach that is called “integrated distributed embedded system”



NOC Based System Design (I)

■ Backbone Design

- A development platform for all NoC based systems
- It encapsulates topological and communication issues such as channels, switches, and network interfaces
- Its design focus is the network communication resources, e.g. switches and interfaces, and NoC system services and performance of different region topologies
- Definition of region requires that potential applications are analyzed and modeled

NOC Based System Design (II)

- **Platform design: create a computation platform for an intended application area.**
 - scaling of the network, definition of regions, design of the resource nodes, and definition of the system control.
 - First, understand the target systems' functionality thoroughly
 - Do not use exact applications as a starting point for architecture requirement definition due to the platform nature
 - Use of optimized virtual components and knowledge of application-area requirements
 - Characterizes application area domain and architecture, and estimates system quality

NOC Based System Design (III)

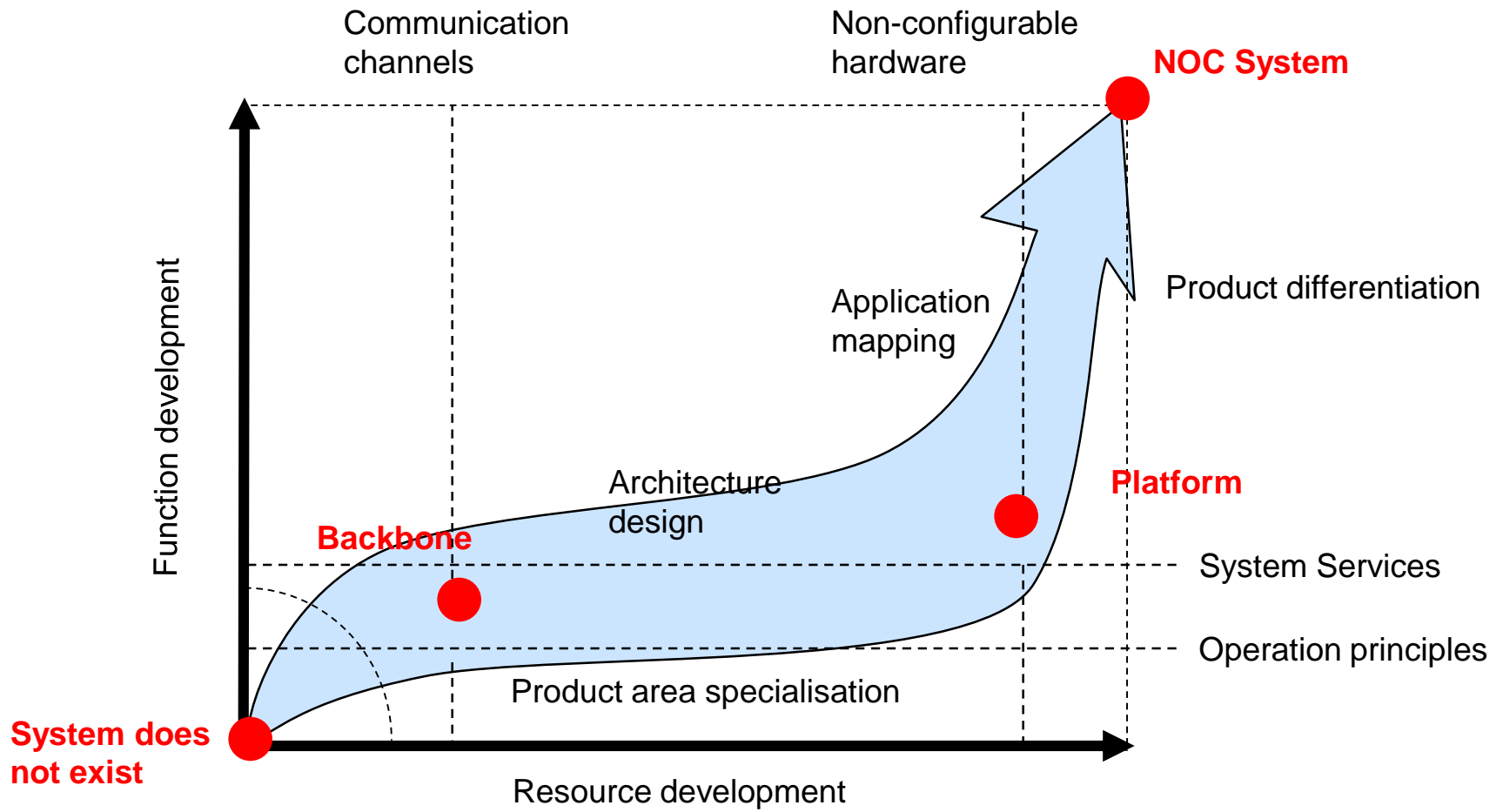
■ System design

- **Application mapping: application functionalities are mapped to the resources**
 - **Must support both dynamic and static mapping**
 - **Main problems: are the resource allocation, optimization of network usage and verification of performance and correctness.**
 - **Several modeling languages should be supported**
- **Main tasks: what to put into the NoCas resources, how to map functionality into those resources (resource selection), and how to validate the decisions.**
 - **Mappability of algorithms and architectures**
 - **Analysis of network behavior is a critical part**
- **The *development and verification environments* provides a virtual machine and development environment for software development, and tools for hardware design**

NOC Based System Design (IV)

Instance	Responsibilities during design
Backbone development	Region (specific resource) types Communication channels and switches Network interfaces of resources Communication protocols (specification)
Platform development	Region scaling Resource design (units, interconnections) Dedicated hardware blocks System level control (implementation of communication, diagnostics, monitoring)
Application development	Resource level control (OS) Functionality of resources (SW, configurable HW) Control of the network Functionality of the network

Using Design Space for NOC





Thanks for your attention