Networks on Chip

Axel Jantsch
Royal Institute of Technology, Stockholm

November 24, 2004

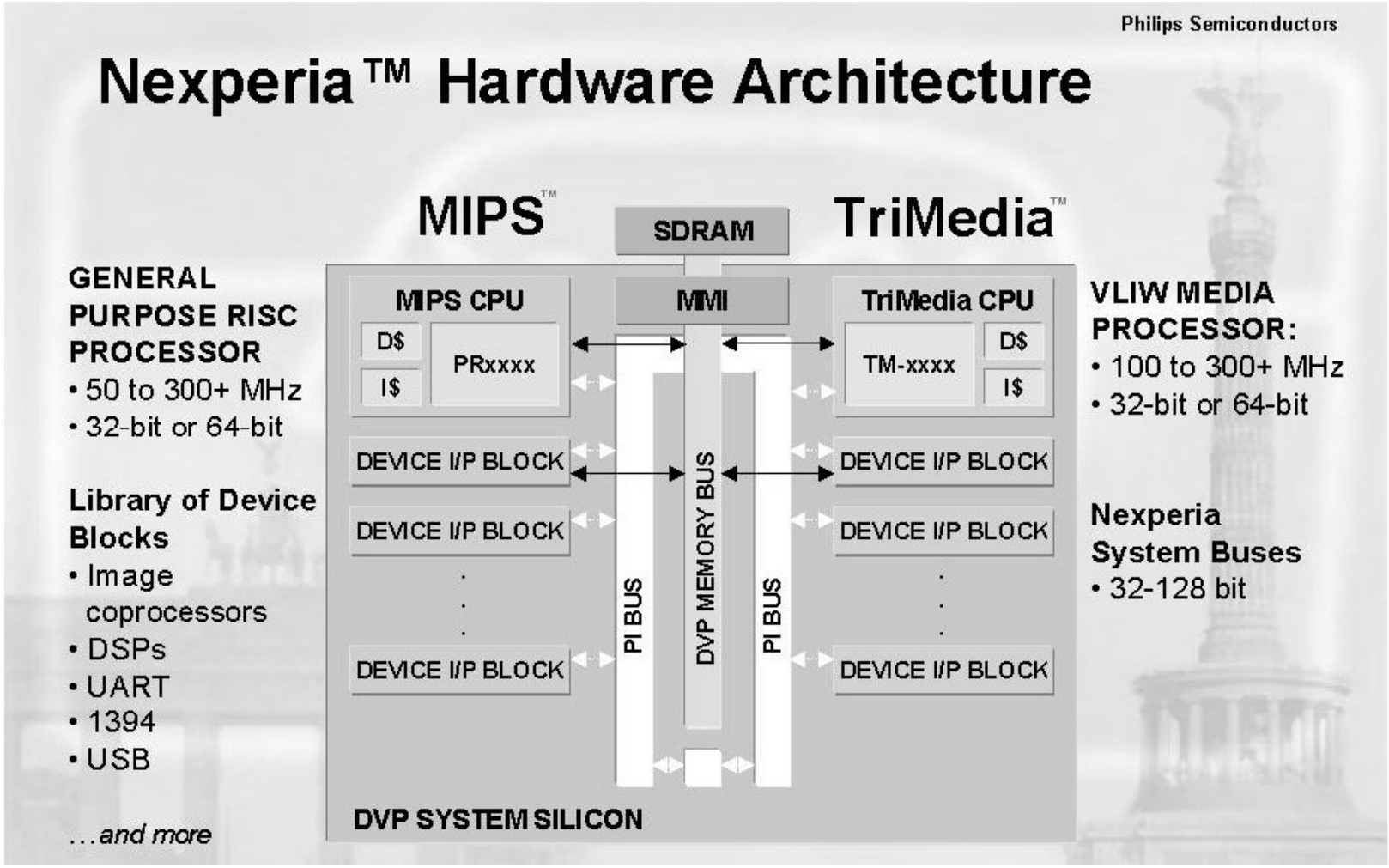
Overview

- NoC as Future SoC Platforms
  - ⋆ What is a good SoC platform?
  - ⋆ Is a NoC a good platform?
- Communication Performance in NoCs
- The Nostrum Network on Chip

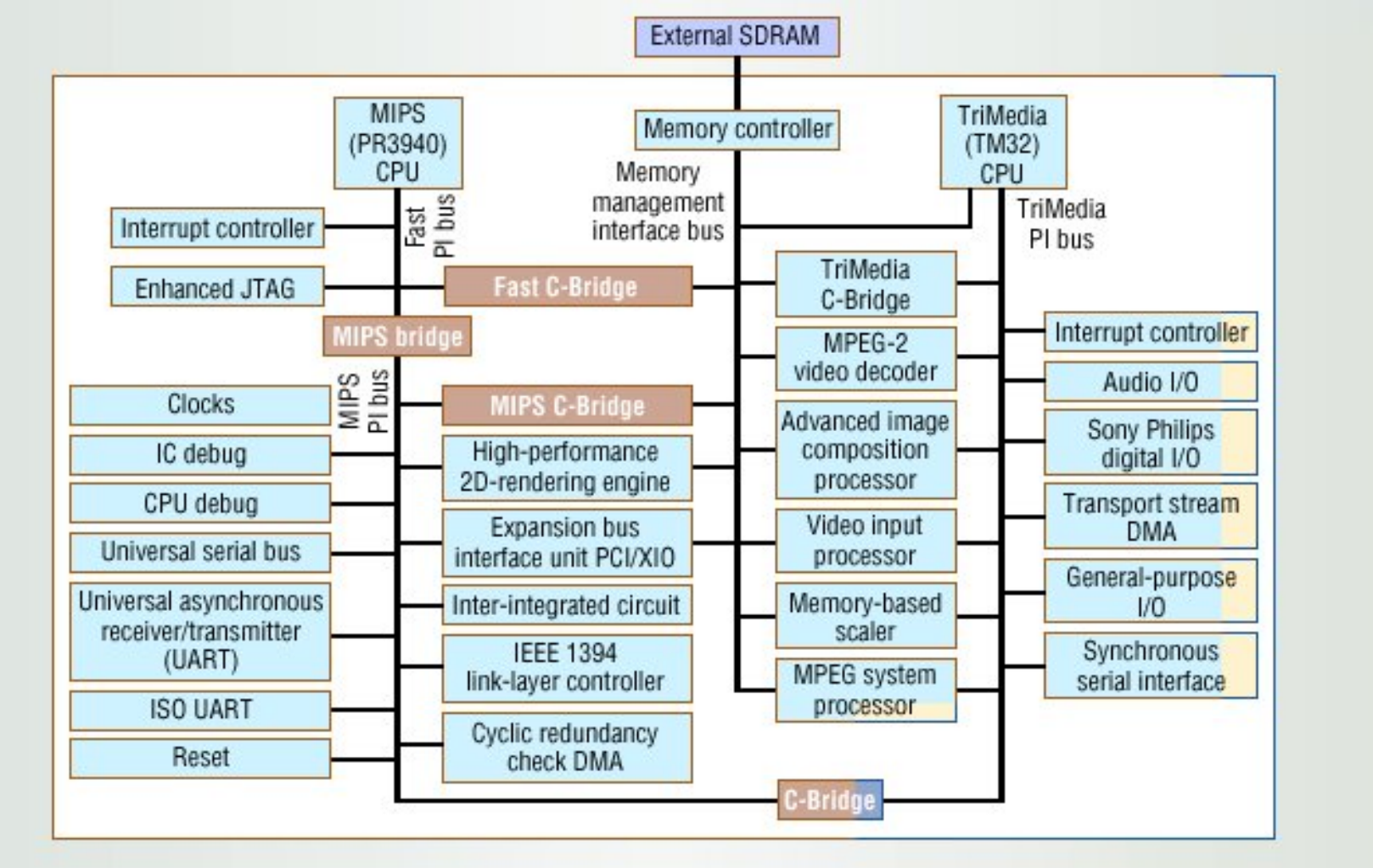
What is a SoC Platform

1. Communication infrastructure
2. Resource management services
3. Design methodology and tools
4. Library of HW and SW IP blocks

Platform Example: Nexperia

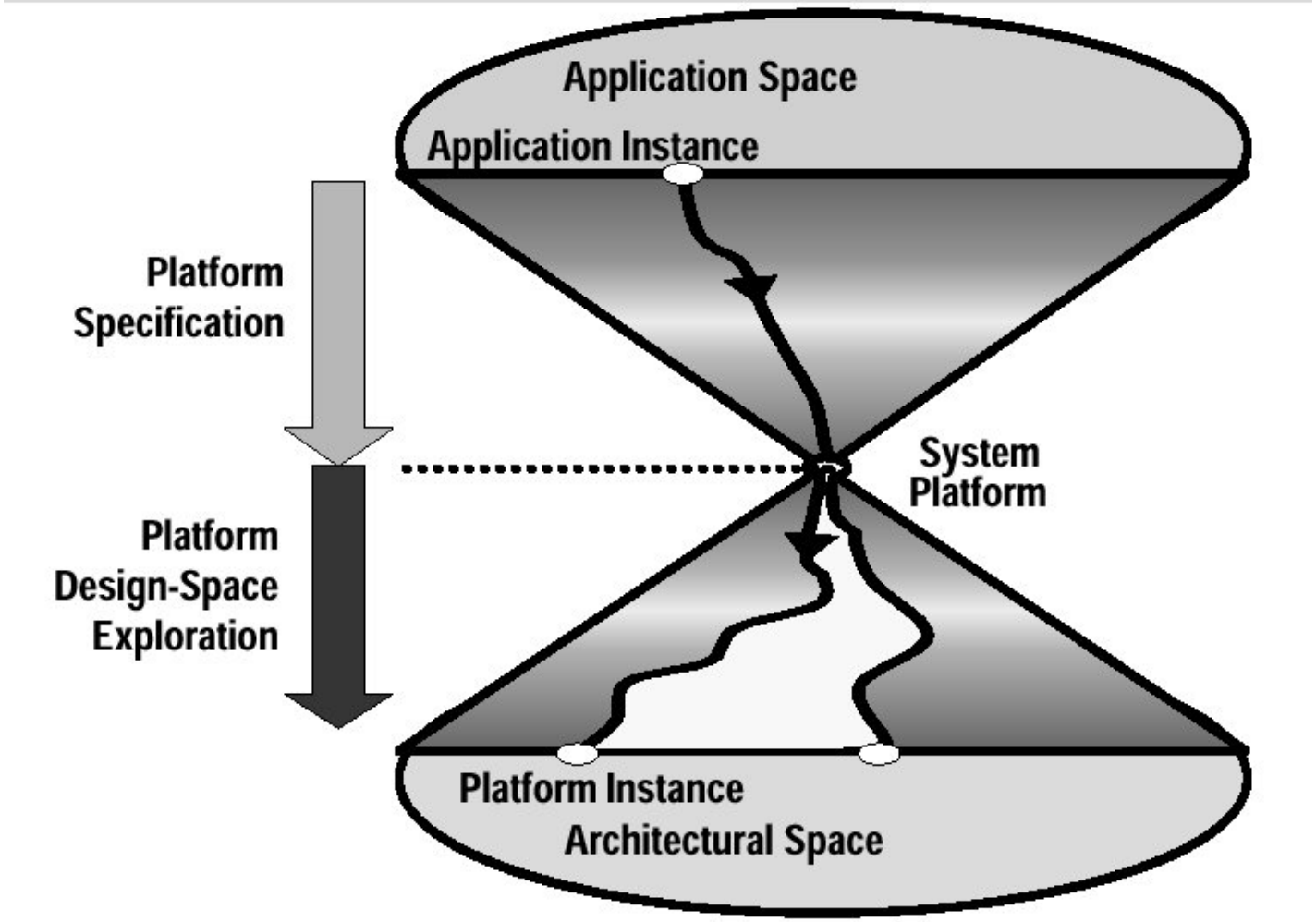


Platform Example: Viper Processor based on Nexperia



From IEEE Computer, vol 36, no. 4, April 2003.

Platform Based Design



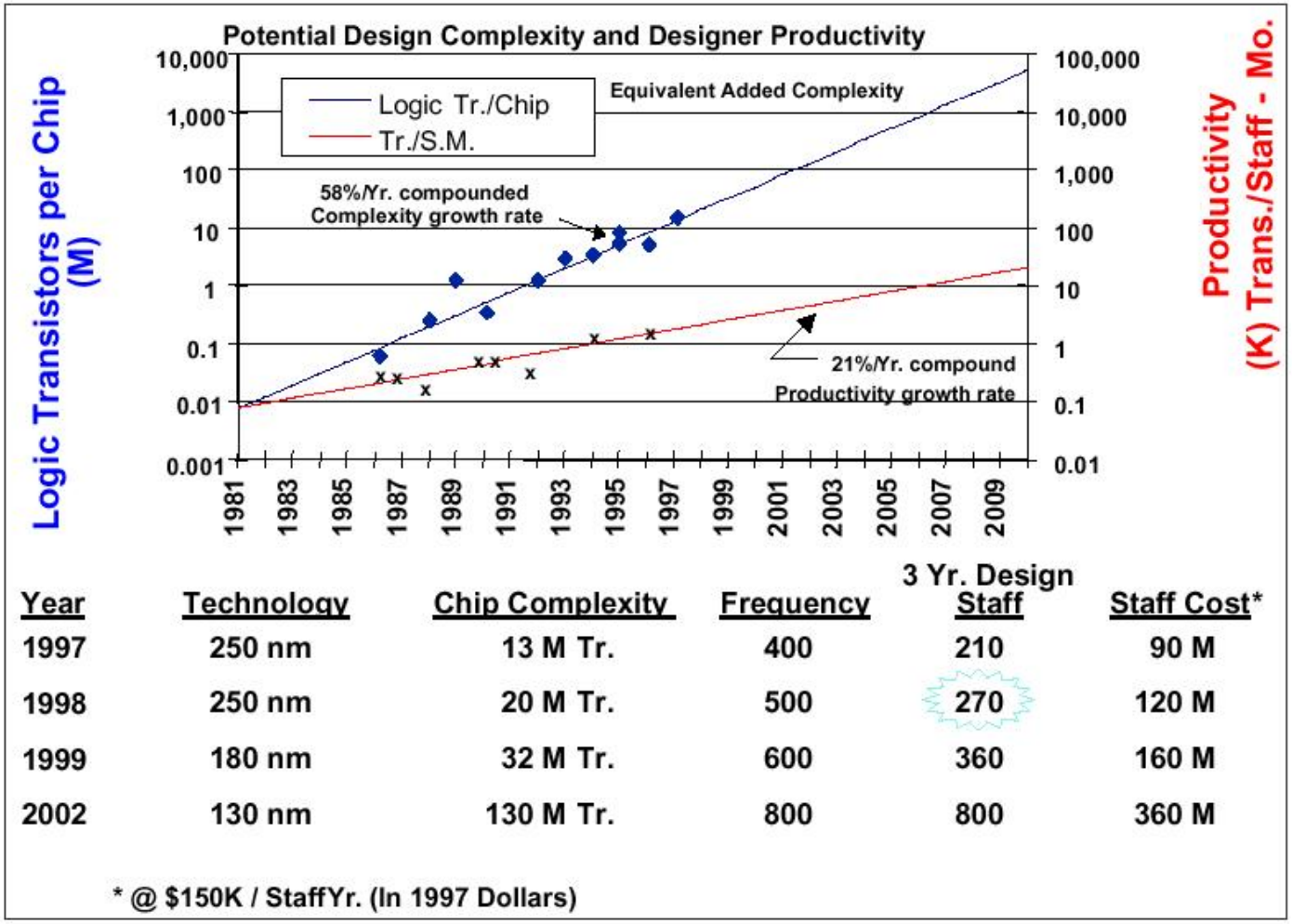
Platform Characteristics

- Tradeoff between efficiency and cost
- Application area specific
- Guarantees and Predictability
  - ⋆ Platform inherent guarantees
  - ⋆ Static guarantees
  - ⋆ Dynamic guarantees
  - ⋆ E.g. communication guarantees
    - ∗ Delivery
    - ∗ Minimum bandwidth and maximum delay
- Scalability

Scalability of a Platform

- Performance - Cost
- Size
- Reliability
- Design methodology

Design Productivity Gap

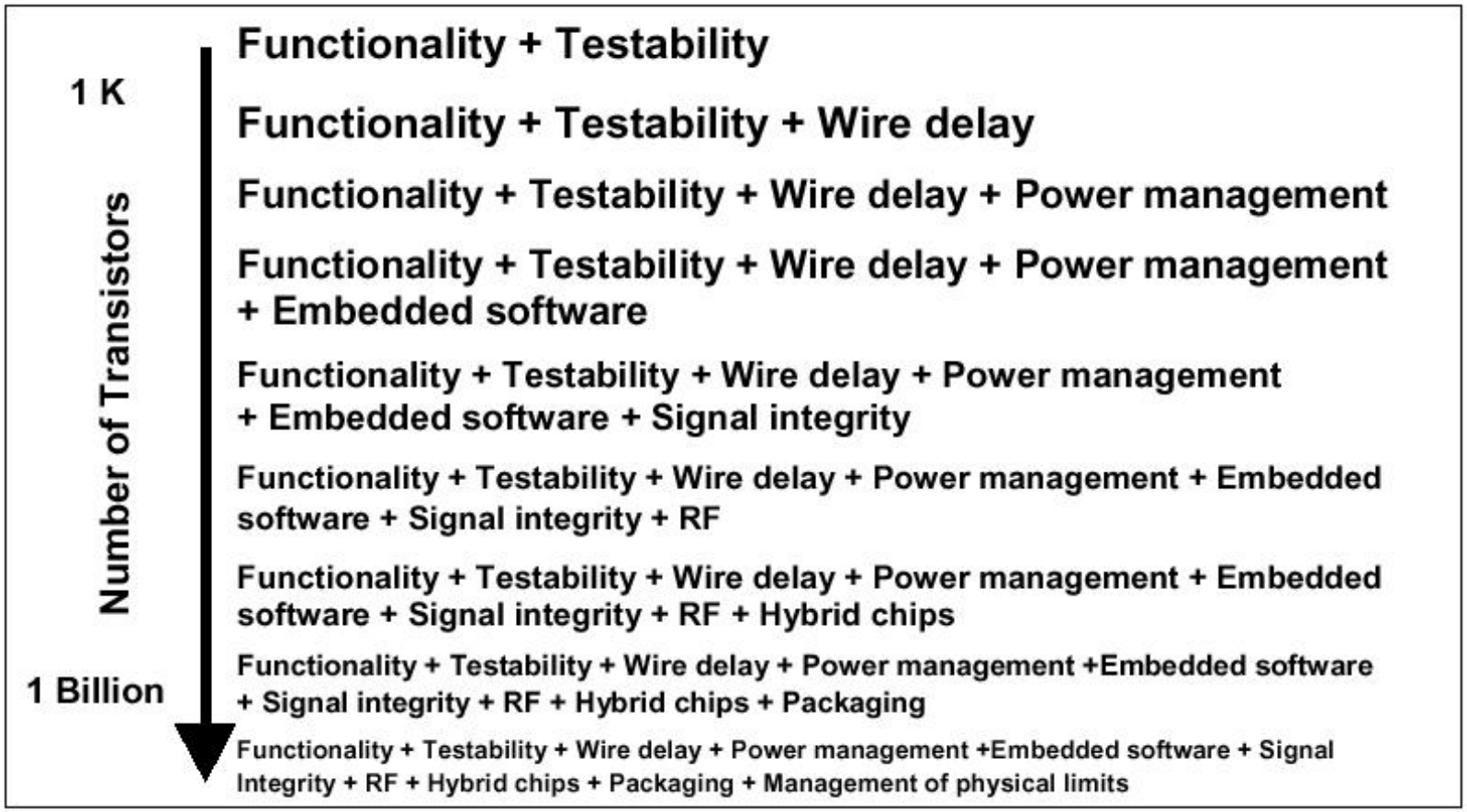


Year	Technology	Chip Complexity	Frequency	3 Yr. Design Staff	Staff Cost*
1997	250 nm	13 M Tr.	400	210	90 M
1998	250 nm	20 M Tr.	500	270	120 M
1999	180 nm	32 M Tr.	600	360	160 M
2002	130 nm	130 M Tr.	800	800	360 M

* @ $150K / StaffYr. (In 1997 Dollars)

International Technology Roadmap for Semiconductors 1999

Super-exponential Increasing Design Complexity



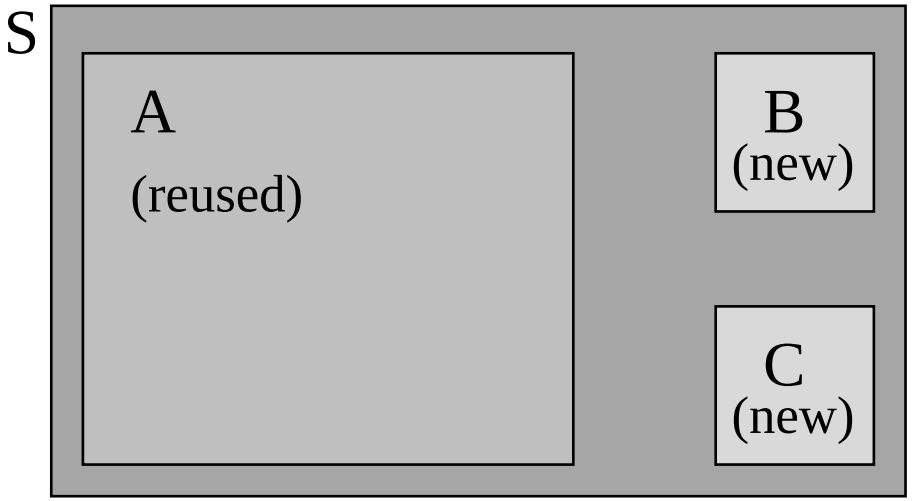
International Technology Roadmap for Semiconductors 1999

Arbitrary Composability

Given a set of components C and combinators O.
Let A_1 be a component assemblage.
(C, O) is arbitrary composable if

$$A_1 + B \Rightarrow A_2$$

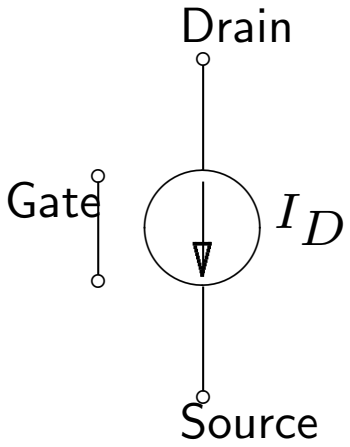
can be done for any $B \in C, + \in O$ without changing the relevant behaviour of A_1.



A MOS Transistor Model

$V_{DS} > V_{GS} - V_T$ (conducting state):

$$I_D = \frac{k'_n W W}{2L}(V_{GS} - V_T)^2(1 + \lambda V_{DS})$$



$V_{DS} < V_{GS} - V_T$ (sub-threshold state):

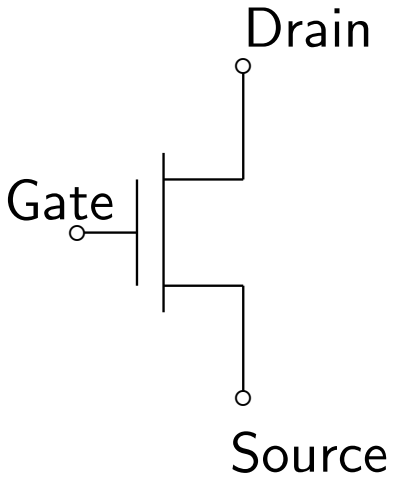
$$I_D = \frac{k'_n W W}{L}((V_{GS} - V_T)V_{DS} - \frac{V_{DS}^2}{2})$$

where

$$V_T = V_{T0} + \gamma(\sqrt{|-2\phi_F + V_{SB}|} - \sqrt{|-2\phi_F|})$$

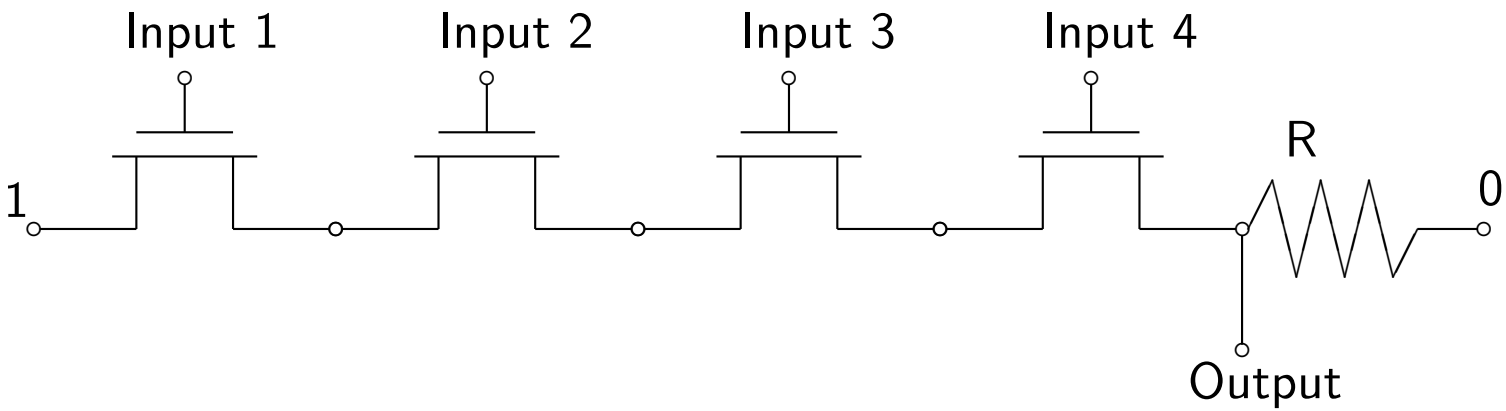
V_{DS} ... drain-source voltage
V_{GS} ... gate-source voltage
V_T ... threshold voltage
I_D ... drain-source current

A Transistor as Switch



Gate	Drain	Source
0	0	undefined
0	1	undefined
1	0	0
1	1	1

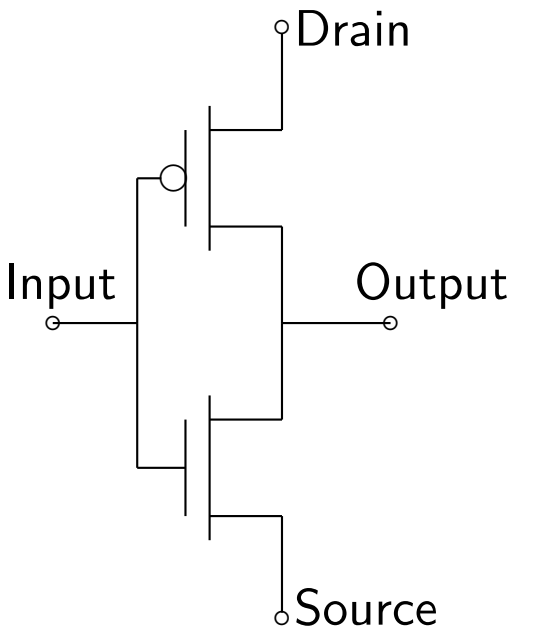
An AND Gate as Transistor Network



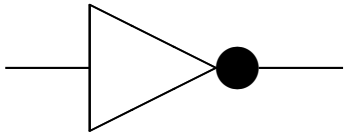
Two problems with arbitrary transistor networks:

- Output is not defined when input is 0.
- Voltage drop between drain and source is relevant but not visible.

An Inverter as Transistor Network



(a)



(b)

Input	Output
1	0
0	1

(c)

Gate Based Abstraction Level

1. The primitive elements are defined by simple models, i.e. small truth tables in this case.
2. The primitive elements can be implemented in a wide range of technologies.
3. The model holds even for arbitrarily large networks of primitive elements.
4. Gates plus connectivity operators have the arbitrary composability property

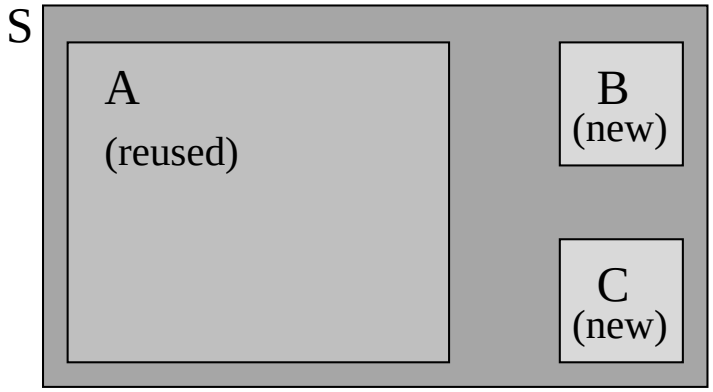
Platform and Composability

- A good platform has the arbitrary composability property.
- There are building blocks that can be added without changing the rest of the system.
- The building blocks can be:
  - ⋆ Computation resources
  - ⋆ Communication resources
  - ⋆ Storage resources
  - ⋆ I/O resources
  - ⋆ Resource manager modules (Scheduler, OS, ...)
  - ⋆ Features: Resources + System functionality
- The "relevant behaviour" includes functionality, performance, cost, reliability, power consumption.
- $\Longrightarrow$ We can make guarantees.

Linear Effort Property

Given a set of components C and combinators O.
Let $A_1, \ldots, A_n$ be component assemblages.

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



$$\texttt{Deffort}(S) = \texttt{Deffort}(A_1) + \cdots + \texttt{Deffort}(A_n) + \texttt{Ieffort}(n)$$

Methodology and Linear Effort

- A good platform comes with a methodology that has the linear effort property.
- The platform is then scalable with respect to capacity increase by reusing ever larger components.
- This implies an invariance with respect to hierarchy: Composition works as well for primitive components as for arbitrary assemblages.

Platform Summary

- A good Platform greatly restricts the design space.
- It trades in optimality for design efficiency and predictability.
- The arbitrary composability and the linear effort properties provide a scalable platform.
- The reuse of ever bigger assemblages and components is platform inherent.
- Predictability of functionality, performance, cost, power consumption and reliability is a prerequisite as well as a consequence for the arbitrary composability and the linear effort properties.

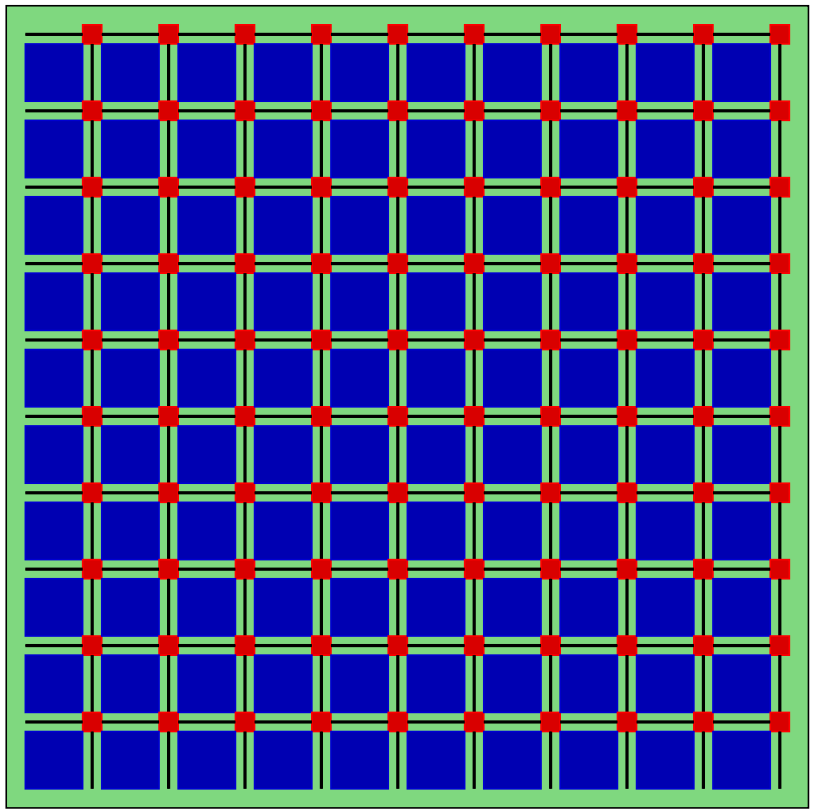
Is NoC a good Platform?

- Trends and challenges
- NoC Concepts
- How NoC addresses the stated problems
- Predictability
- Drawbacks
- NoC Design Process

Trends and Challenges

- Communication versus computation
- Deep submicron effects
- Power
- Global synchrony
- Design productivity
- Heterogeneity of functions

NoC Concepts



- Regular geometry
- Predictable physical and electrical properties
- No global wires
- No global synchrony
- Pre-developed communication infrastructure with known properties

How does NoC Address the Challenges?

Challenges:

- Communication versus computation
- Deep submicron effects
- Power
- Global synchrony
- Design productivity
- Heterogeneity of functions

NoC Promises:

- Communication service stack is pre-developed
- Predictable electrical properties
- No global clock tree; Parallelization of computation
- GALS: Global asynchronous - local synchronous systems
- Reuse and predictability
- Different sub-systems are developed and implemented separately

Reuse

- Components and resources
- Communication infrastructure
- Application parts and features
- Design, simulation and prototype environment
- Verification effort

Predictability

- Communication performance
- Electrical properties
- Design and verification time

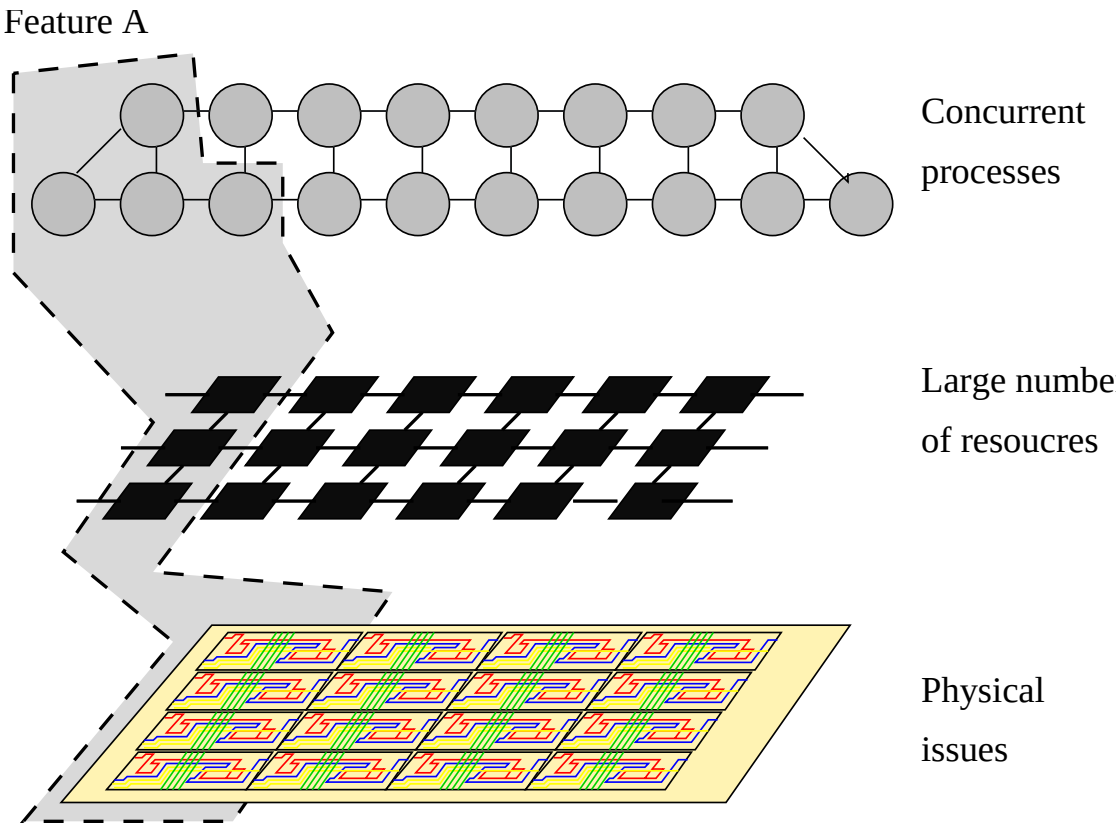
Disadvantages

Loss of optimality:

- Communication services are overdimensioned
  - ⋆ Too high bandwidth for the worst case
  - ⋆ Services included that are not required
- Resource slots have fixed size
  - ⋆ Smaller resources waste space
  - ⋆ Larger resources have to be split
  - ⋆ Can be rectified with the region concept.
- Standard and not application specific comunication services

A Network-on-Chip Based Design Process

- Configuring the platform
- Selecting resources
- Reuse of features
- Performance evaluation
- System integration



Part I - Summary

- To build large systems components and features must be heavily reused.
- Reused entities must be arbitrary composable at
  - ⋆ the physical level
  - ⋆ the architecture level
  - ⋆ the function level.
- Predictability and guarantees are prerequsites for and consequences of arbitrary composability.
- NoC based plaforms focus on composition at
  - ⋆ the physical level,
  - ⋆ the architecture level,
  - ⋆ the function and application level.

⇒ **They have the potential to deliver arbitrary composability.**