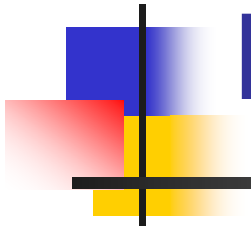


NoCs: Vision, reality, trends

Luca Benini
lbenini@deis.unibo.it
DEIS Università di Bologna

SoC: Enabler for Digital Convergence

4G/5G, DMB, WiBro, etc.

Today

> 100X

Performance
Low Power
Complexity
Storage

Future

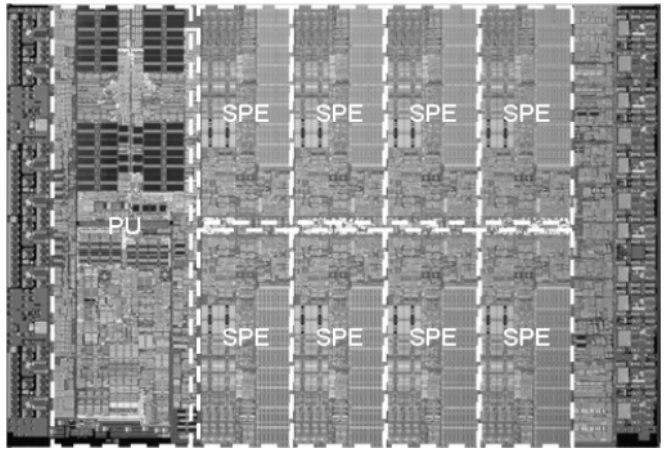
SoC

What about GP computing?

…chip level multiprocessing (CMP) architectures represent the future of microprocessors because they deliver massive performance scaling while effectively managing power and heat

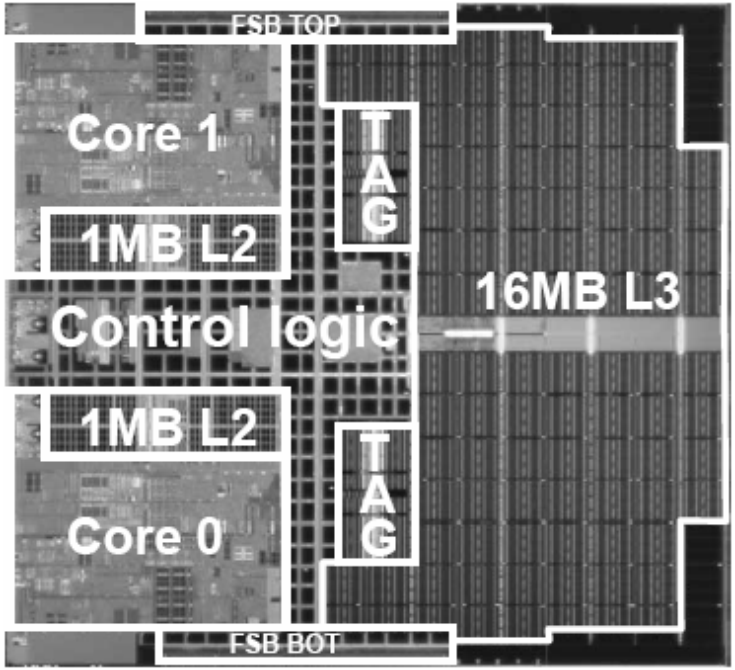
Source: White paper "Platform 2015: Intel Processor and Platform evolution for the next decade

IBM's Cell



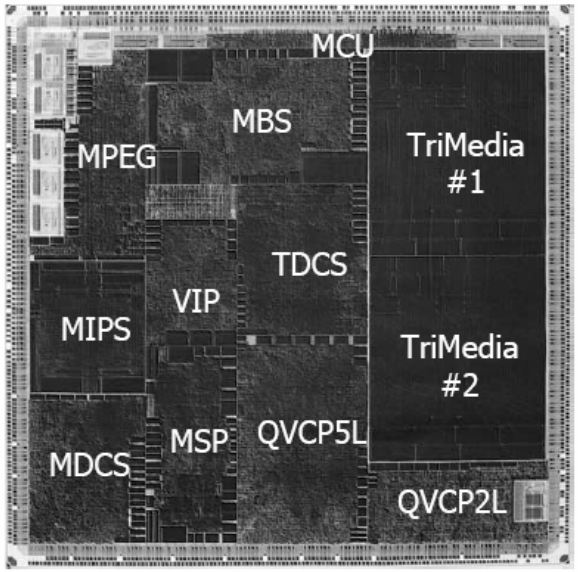
9PUs, 3.2GHZ@1V, 234MTx, 221mm2, 90nmSOI, 8Metals

INTEL's 2-Core Xeon

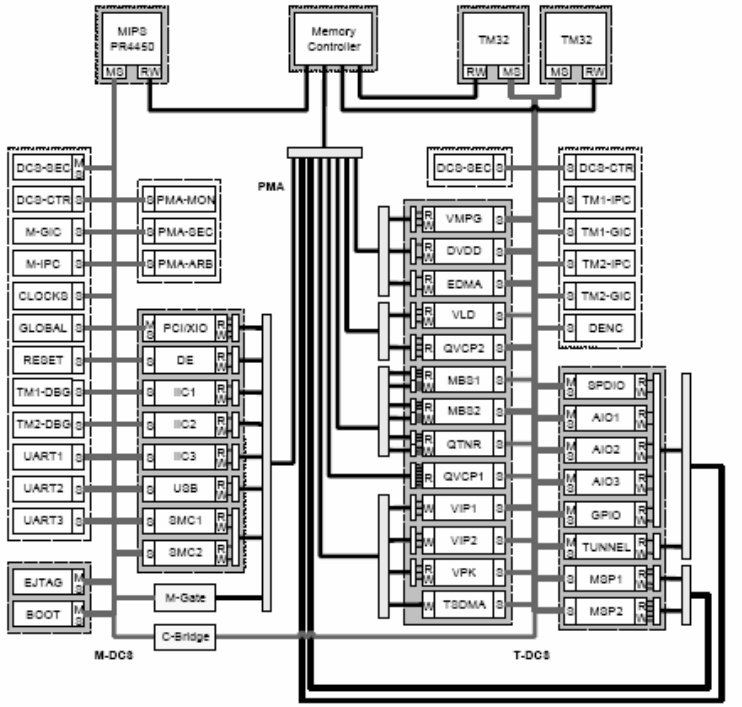


2CPUS, 3Ghz@1.25V, 1.33BTx, 435mm2, 65nm, 8Metals

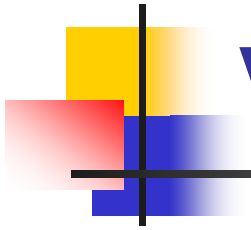
The communication bottleneck



130 nm 50M trans.
1 MIPS 2 Trimedia
60 IP blocks 250 RAMs
Up to 100 Gops 4 Watt



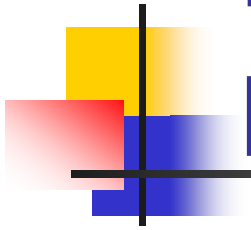
- In 45 nm, up to 10x on a single chip
- Chips become distributed systems!
  - Nothing is global (timing reference, power supply)
  - Communication-dominated!



Vision: what do we need?

- Scalable
  - Don’t want to change the way I design communication architecture even if BW requirements scale up exponentially
- Predictable
  - I want to know what to expect (latency, bandwidth), and I want to be able to negotiate it
- Robust
  - Keeps going and going… Even if something is broken inside
- Efficient
  - Silicon is expensive, power is precious
- Easy
  - To create, update, analyze, verify

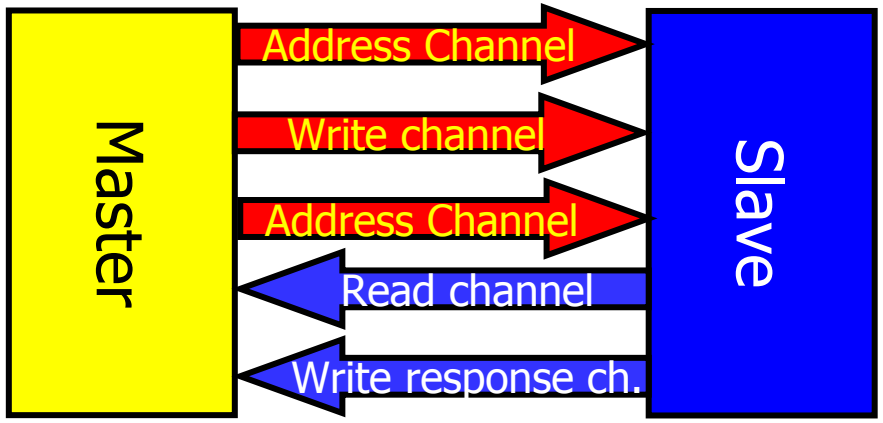
Reality: Never underestimate the power of evolution



- On-chip buses have done the job
- They are evolving fast
  - Protocols
  - Architectures
  - Design technology

Protocol evolutions

- Example: AMBA AXI vs. AMBA AHB

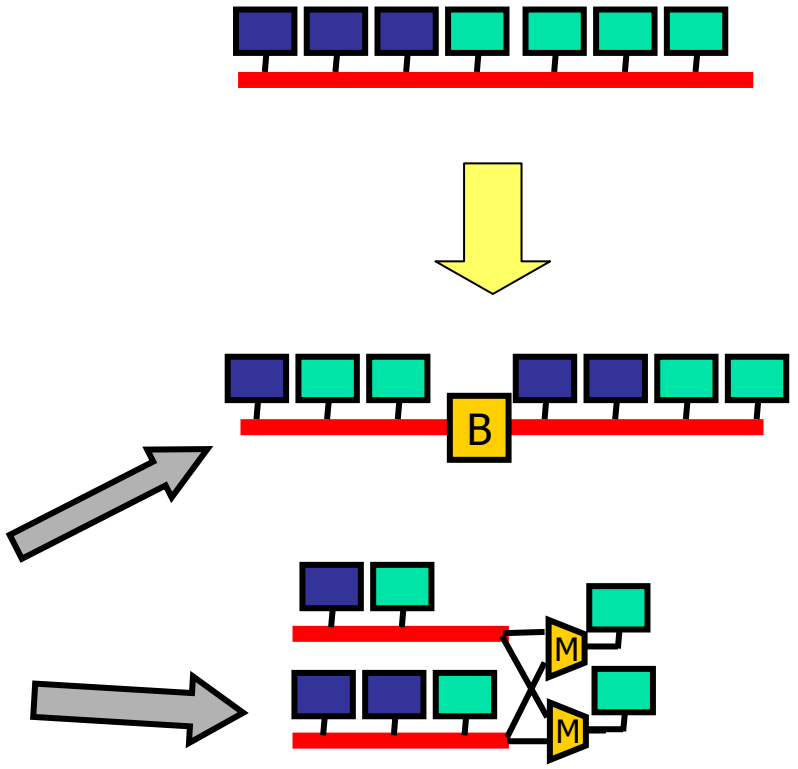


Fully *P2P session layer* protocol: can connect M to S with nothing in-between

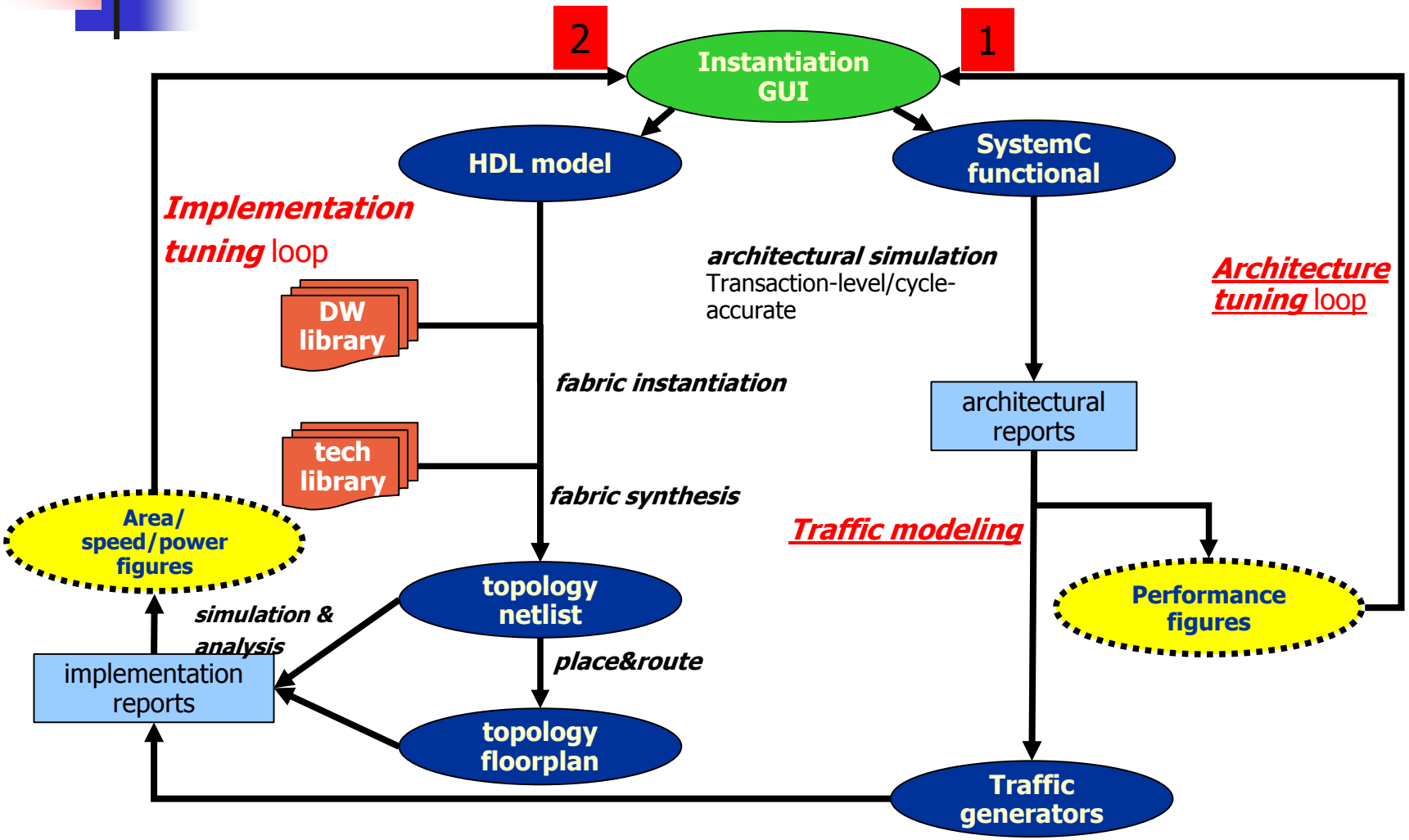
- Features
  - Multiple outstanding transactions, out of order completion
  - Multiple channels (5!)
  - Supports bus-based power management
  - Support for protection, caches, locking
  - **Allows pipelined implementation**
- **On a bus: 80% (vs. 50%) utilization with min. slowdown**
- **But: high area&latency cost, underperforms at low utilization**

Architectural Extensions

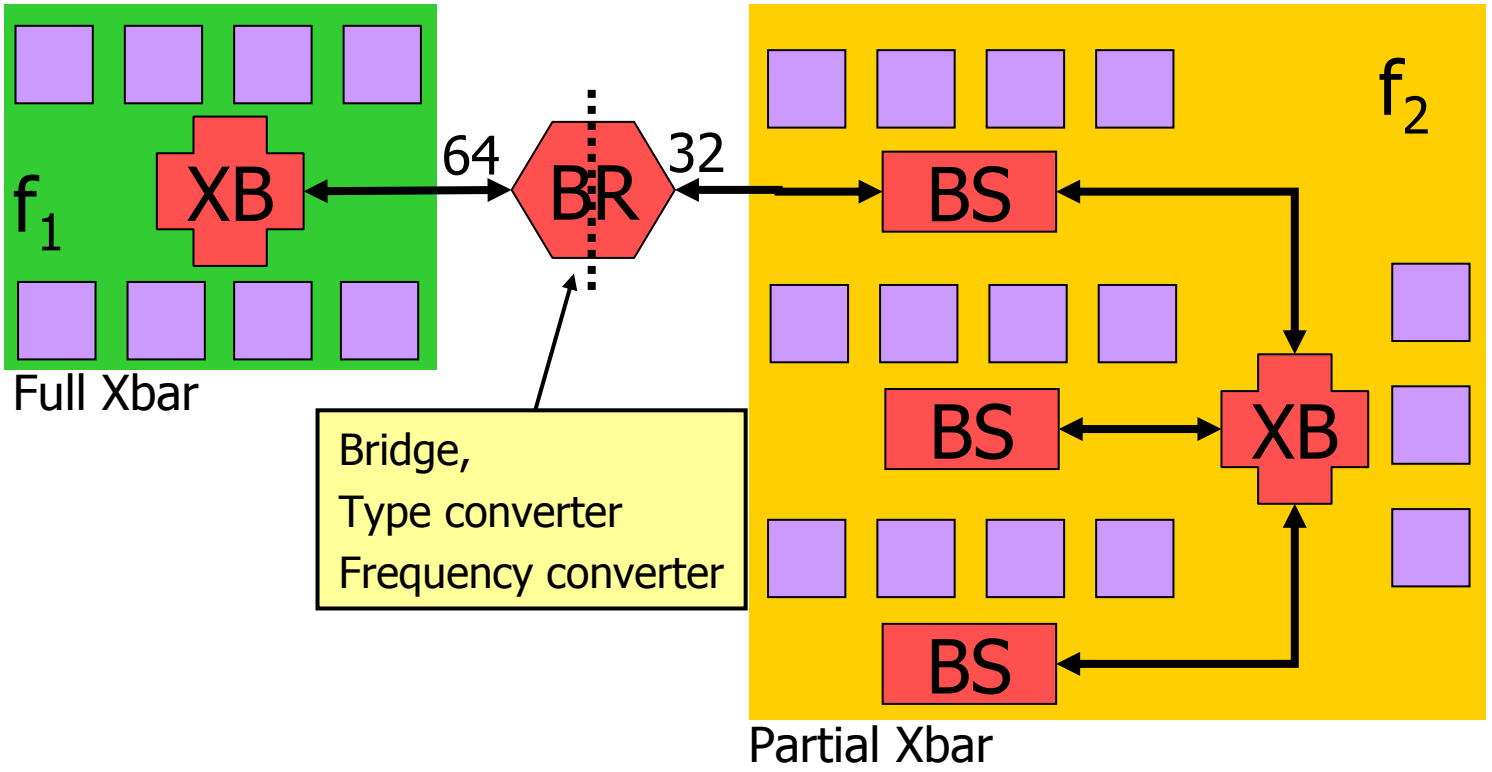
- Single shared bus is clearly non-scalable
  - Even at 100% usable bandwidth
- "Patch" bus architecture
  - Clustering & Bridging
  - Multi-layer/Multibus



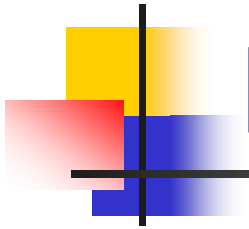
Design technology



State of the art interconnect



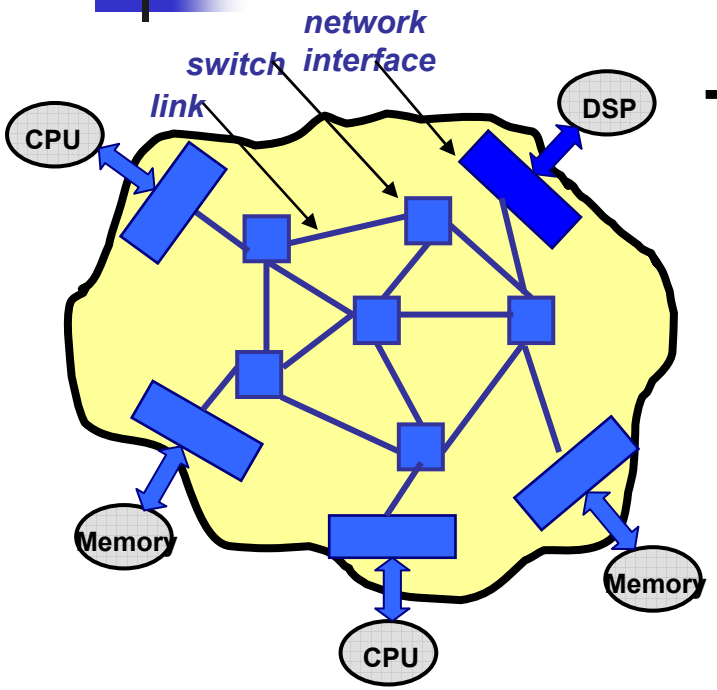
In AXI, OCP cores are unaware of topology



Reality: are we done?

- ✓ Flexible & extensible
  - Multiple frequency domains, bus widths (reduced cost)
  - More bandwidth with clusters and Xbars
- ✘ Easy
  - Difficult to tune, analyze and verify
- ✘ Scalable (in architecture)
  - Bad in multi-hop: high cost bridges, with deadlocks
- ✘ Scalable (at the physical level)
  - Spaghetti wires everywhere: wiring limits performance too!
- ✘ Robust
  - Several single points of failure

Are NoCs the answer?



The rules of the game:

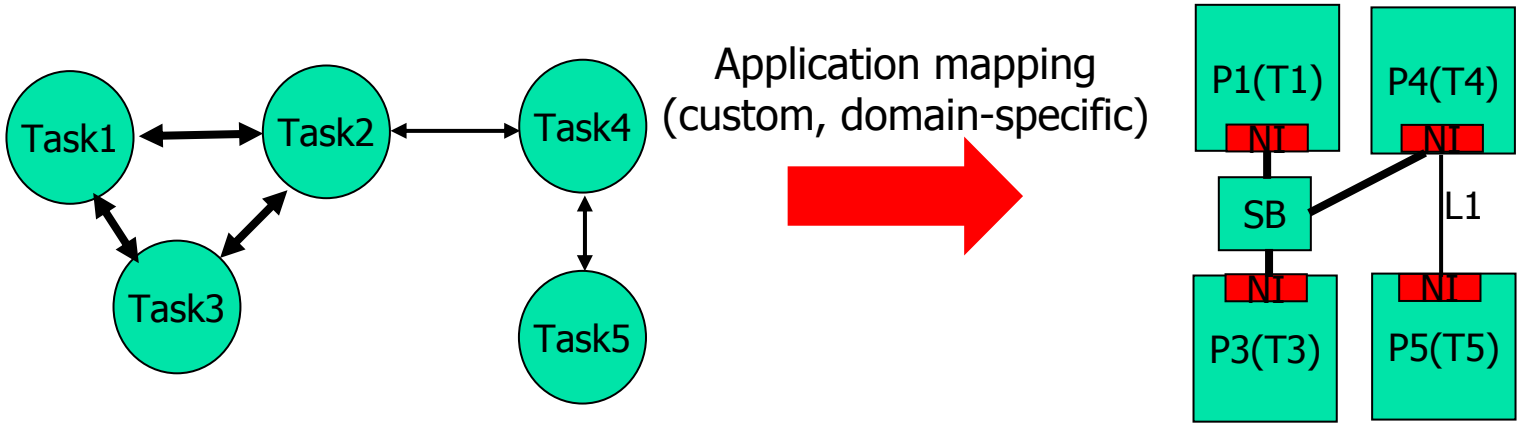
- Match and exceed efficiency and flexibility of evolutionary solutions
- Address their limitations
  - Scalability (architectural & physical)
  - Ease of design
  - Robustness

A piece of good news: with new P2P protocols (AXI, OCP), migration to NoC is transparent to cores!

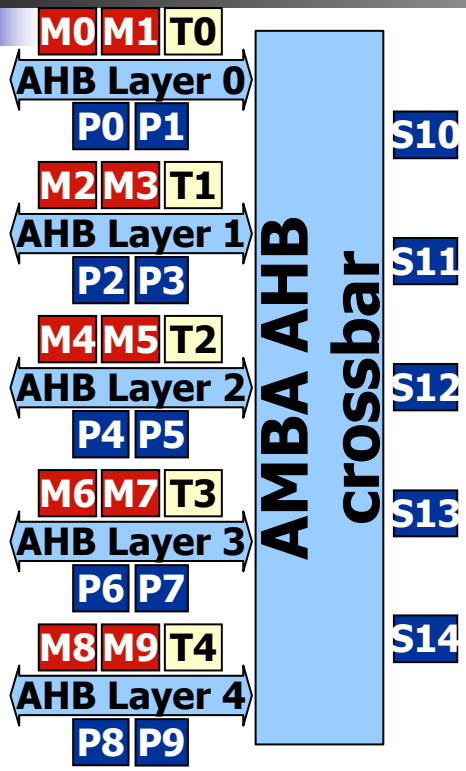
An example: **Xpipes**

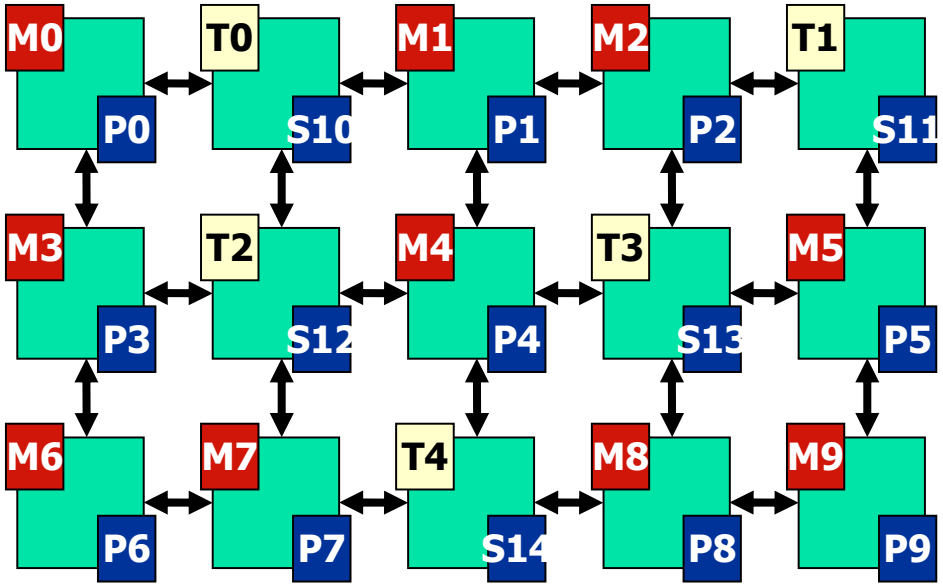
- Xpipes is a **heterogeneous** NoC infrastructure
- Four year lifetime, mature research project
  - UNIBO (architecture), Stanford/EPFL (design technology), UNICA (backend)

- Xpipes in numbers
  (4P switch, 28b flit, 5-FIFO, 130nm)
  - 18 FO4 delay (1GHz)
  - 15 KGates (NAND2)
  - 22.6 uW/MHz (22,6mW@1GHZ)
  - 2clk SW, 2clk NI
- Currently available in STM and UMC processes (130nm)
  - Up to 20x20 switches (36FO4)



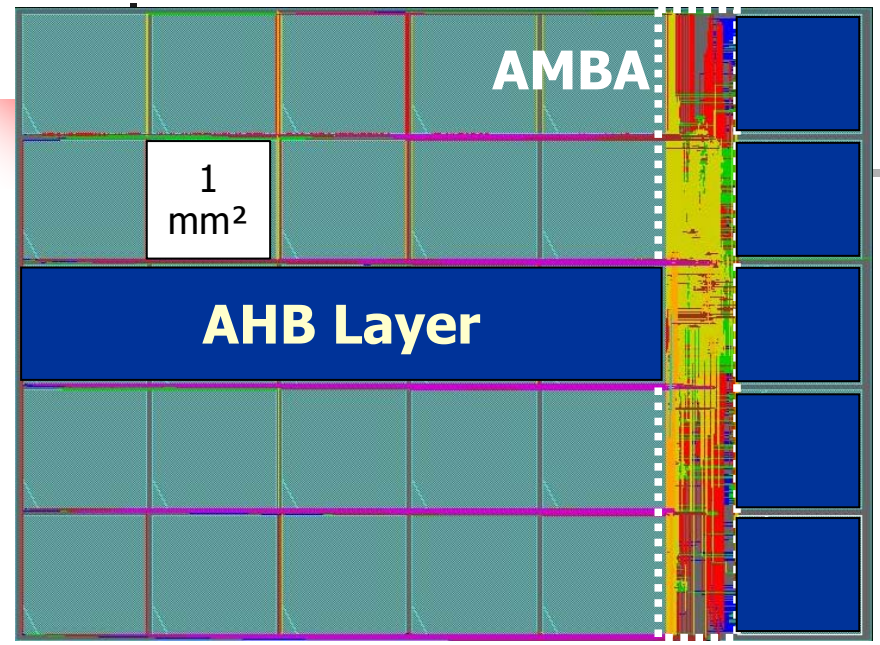
Assessing viability



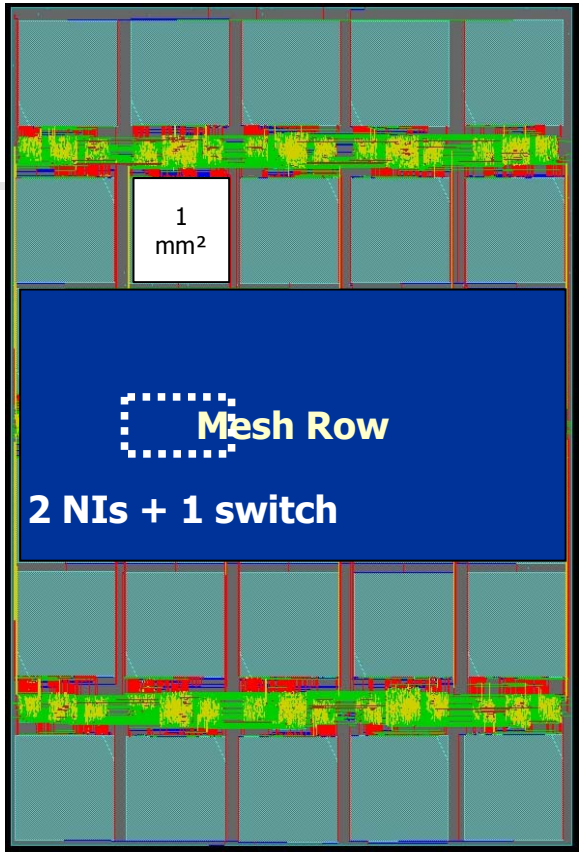


- Realistic Eterogeneous SoC (10 cores, 5 accelerators 10 private memories, 5 shared memories)
- Shared bus fully saturated!
- Both interconnect topologies were hand optimized by different teams

Comparison



Shared Slaves



- 130nm UMC library
- IPs: 1mm² obstructions (ARM cores, 32kB SRAM)
- Wire routing over the cores was forbidden
- Summary of results:
  - 2.7% vs. 17% post P&R timing degradation → Much improved physical scalability

Predictability is highly enhanced

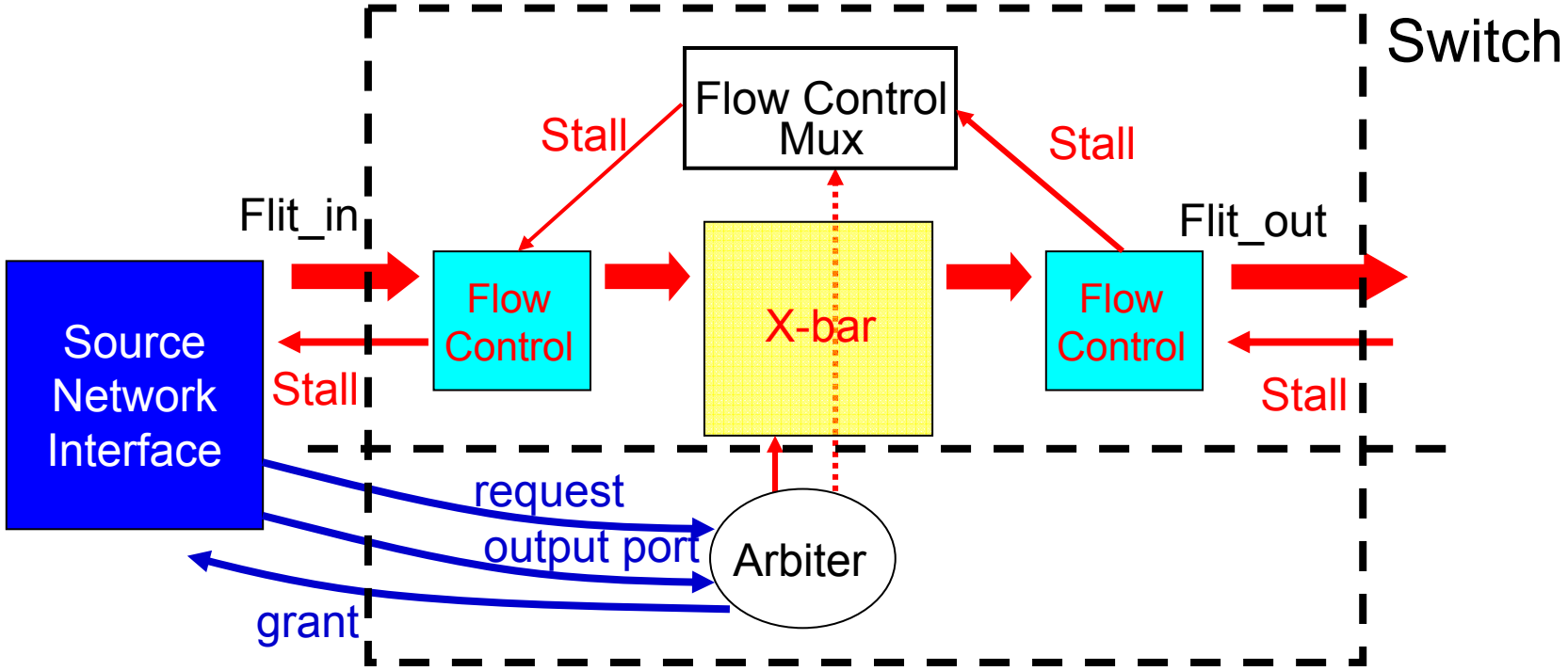
Efficiency: competitive with state-of-the art interconnect even in 130nm technology

  - 7x more area and 5x more power (mostly due to flip-flops in buffers)
  - Overall better energy efficiency for >4 Watts Proc&Mem power

Low latency design

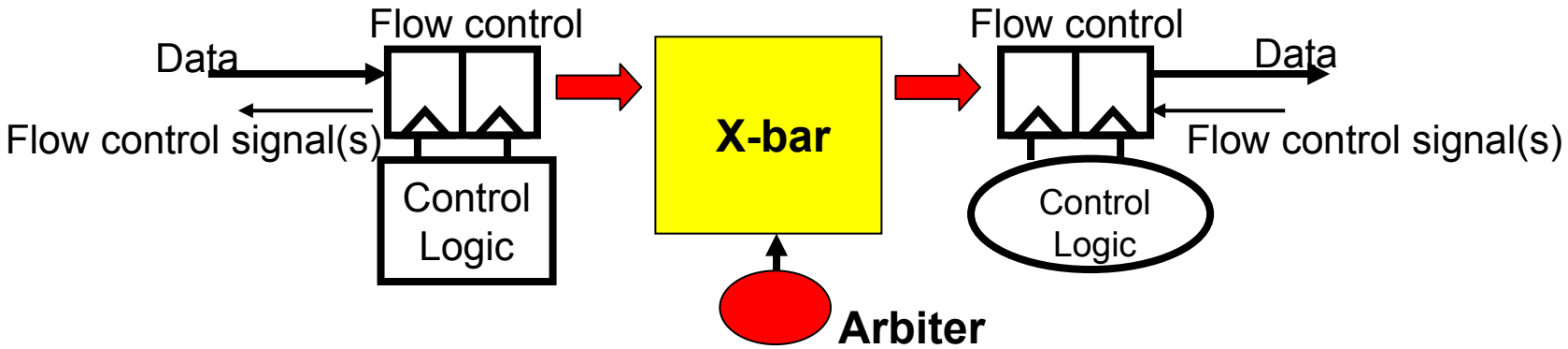
Separate data path and control path

1. Extract the control-path contribution from the critical path of switches
2. Overlap (in part) path setup with payload transfer

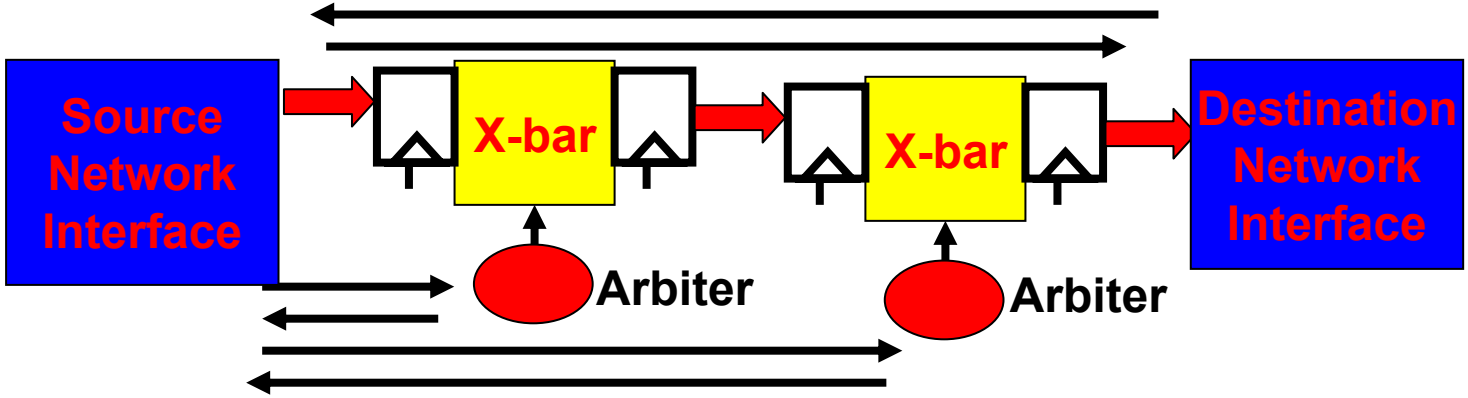


Flow control issues

- Link-level flow control slows down the datapath
  - The control logic delay adds up to the critical path



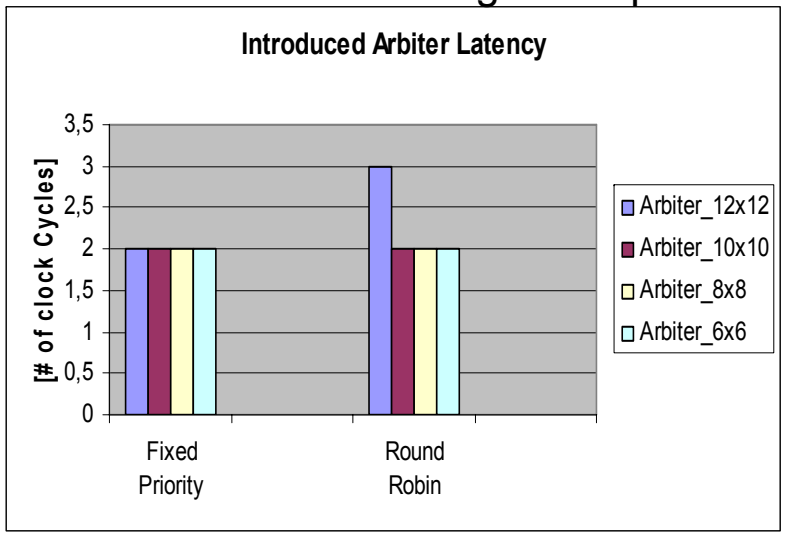
- Pure retiming stages speed up the network at the cost of E-to-E flow control
  - Credit-based needs high circuit set-up latency



Preliminary results

Baseline Clock frequency 1,75 Ghz (10 FO4)

(End-to-end flow control. Flow control stages are pure timing stages)



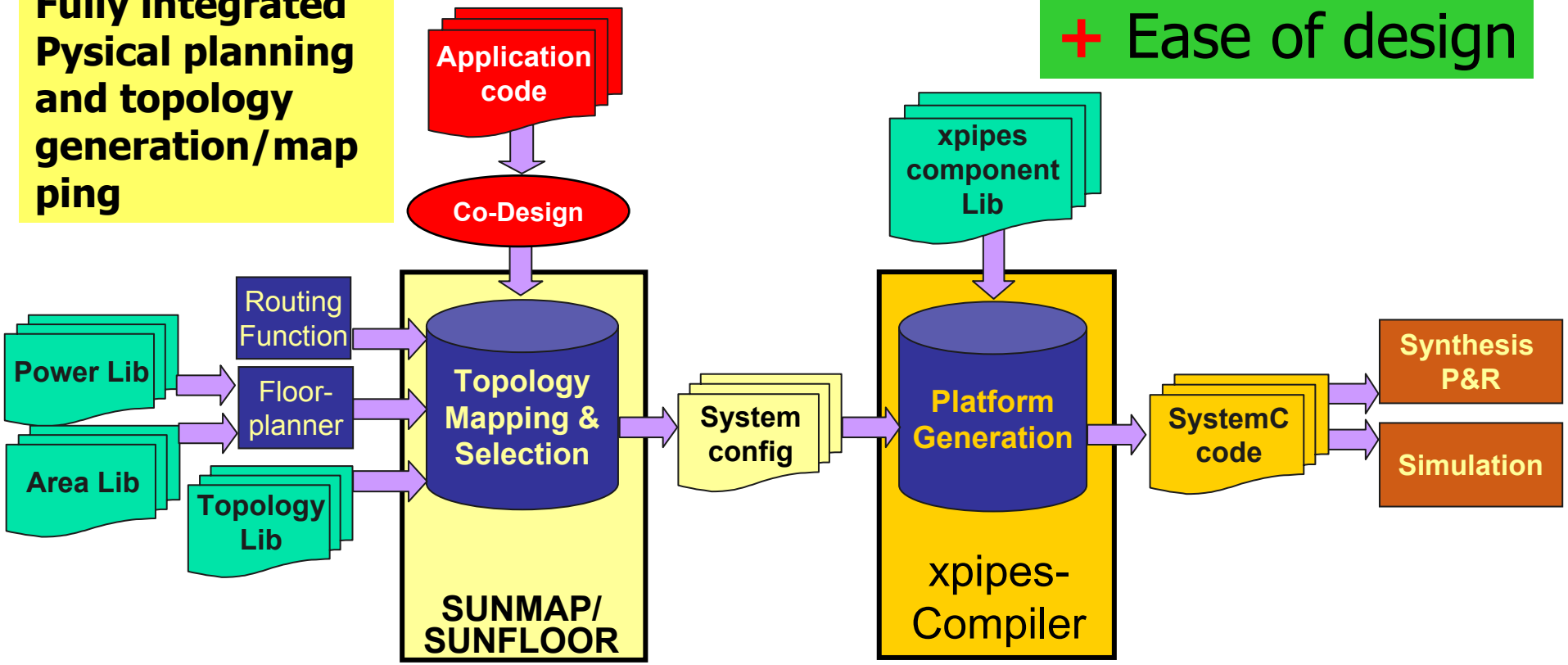
(UMC 0.13um technology)

- Arbitration delay results in a 2-cycles initial latency
- Interesting issues:
  - Circuit set-up and tear-down on a packet-by-packet basis?
  - Flow control largely impacts switch performance (i.e. critical data path)

Area and power reduction

Fully integrated Pysical planning and topology generation/map ping

+ Ease of design



Automated topo-map-floorplan:
1day (vs weeks) 8 vs. 15 switches, 25% less power, 10% speedup

Summary of trends

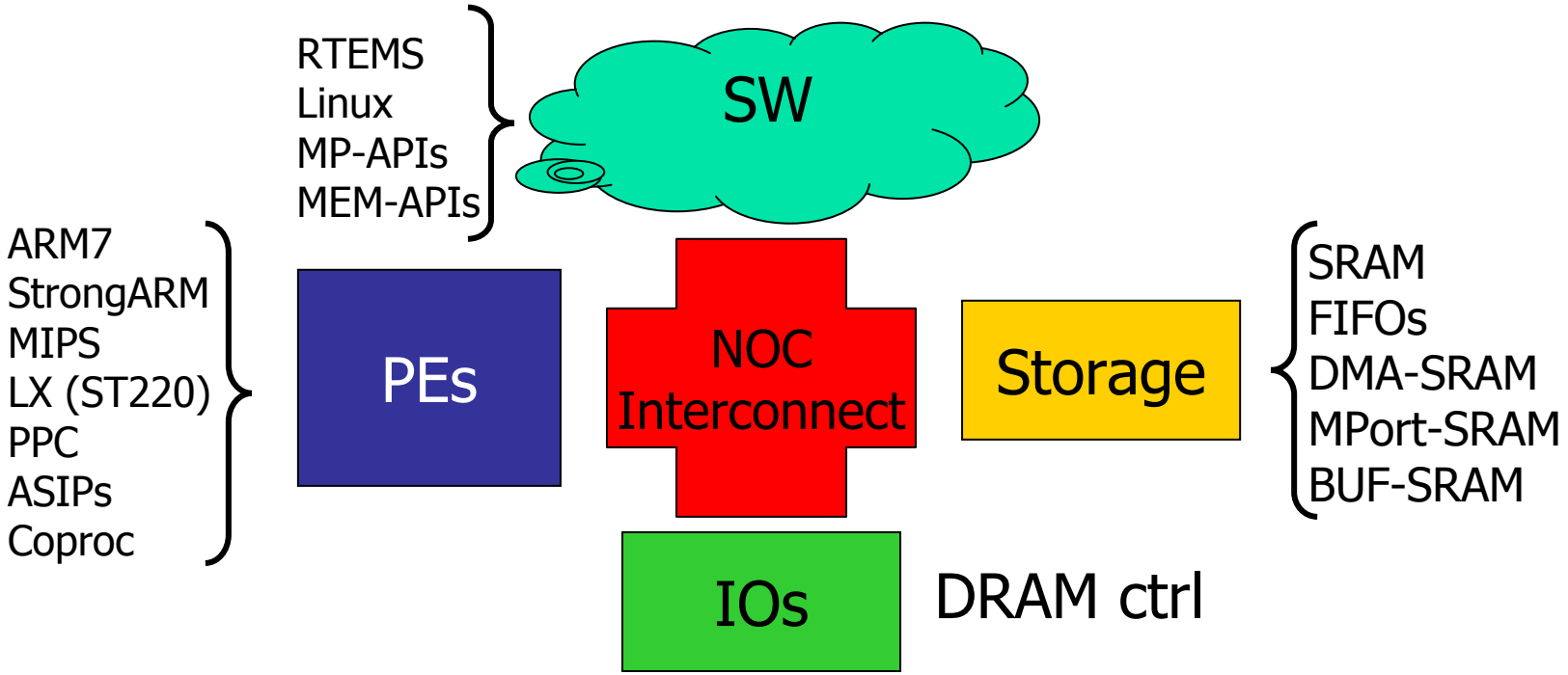
- Looks good
  - Competitive in efficiency even in 0.13um
    - Work harder for low latency (multi-hop and NIs!)
    - Work harder for low power (buffers & clock!)
    - Work much harder for area (buffers)
  - Interconnect predictability greatly improved
    - From spaghetti wiring to planned structured wiring
  - Stronger design technology is feasible
    - From iterative to "single pass" flows
- Promising areas for added advantages
  - Robustness (Fault tolerant NoCs)
  - QoS support
  - Dynamic adaptation (Mapping, routing, Power management…)

Better in 90,65 nm tech.

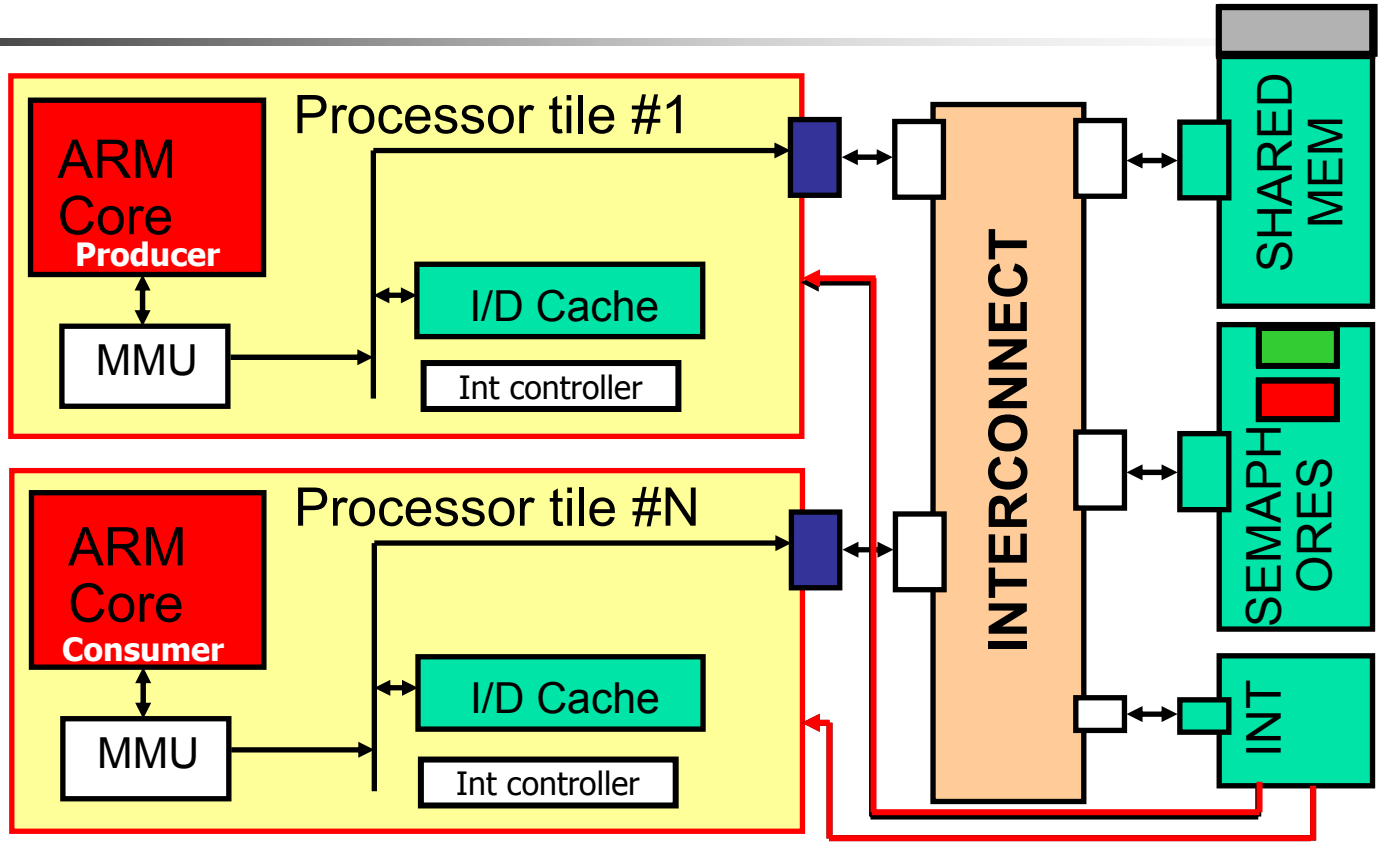
…But this is only a piece of the puzzle!

Trends: NoC-based Platforms

- A COMPLETE environment for MPSoC architectural design and exploration (HW & SW)

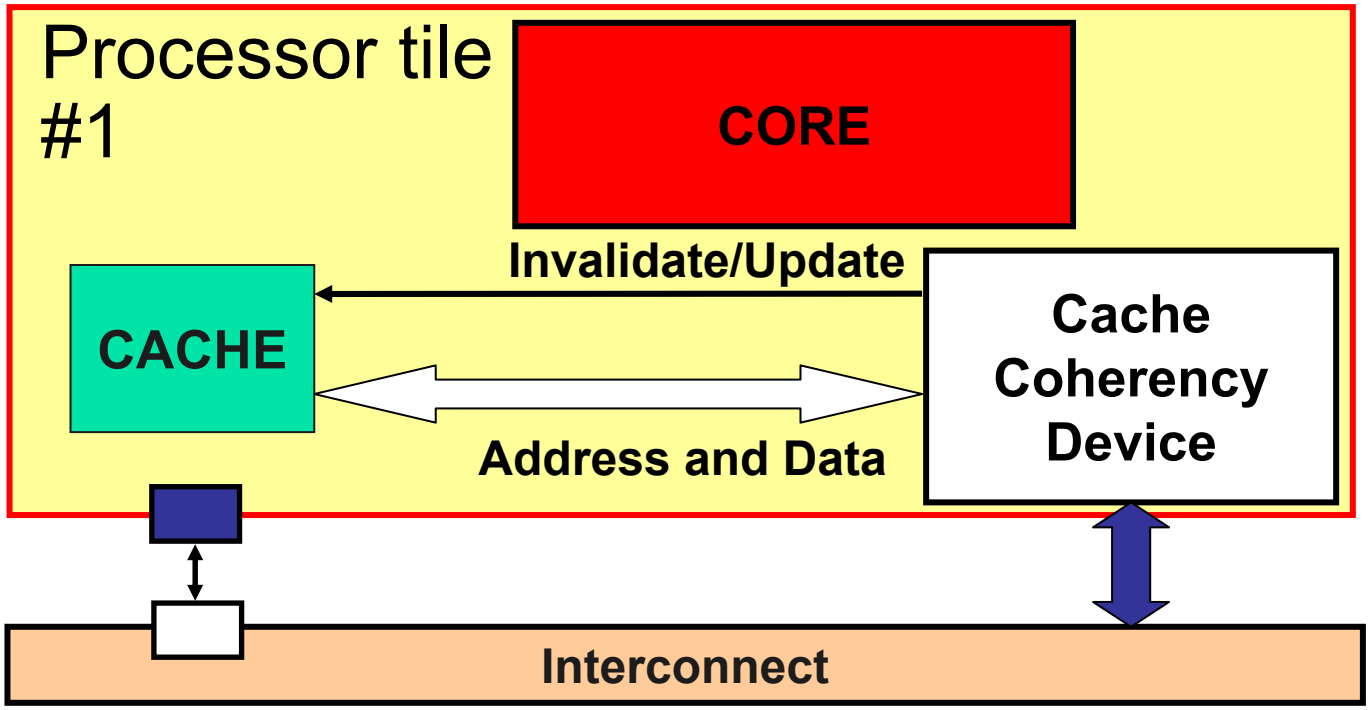


Basic processor architecture



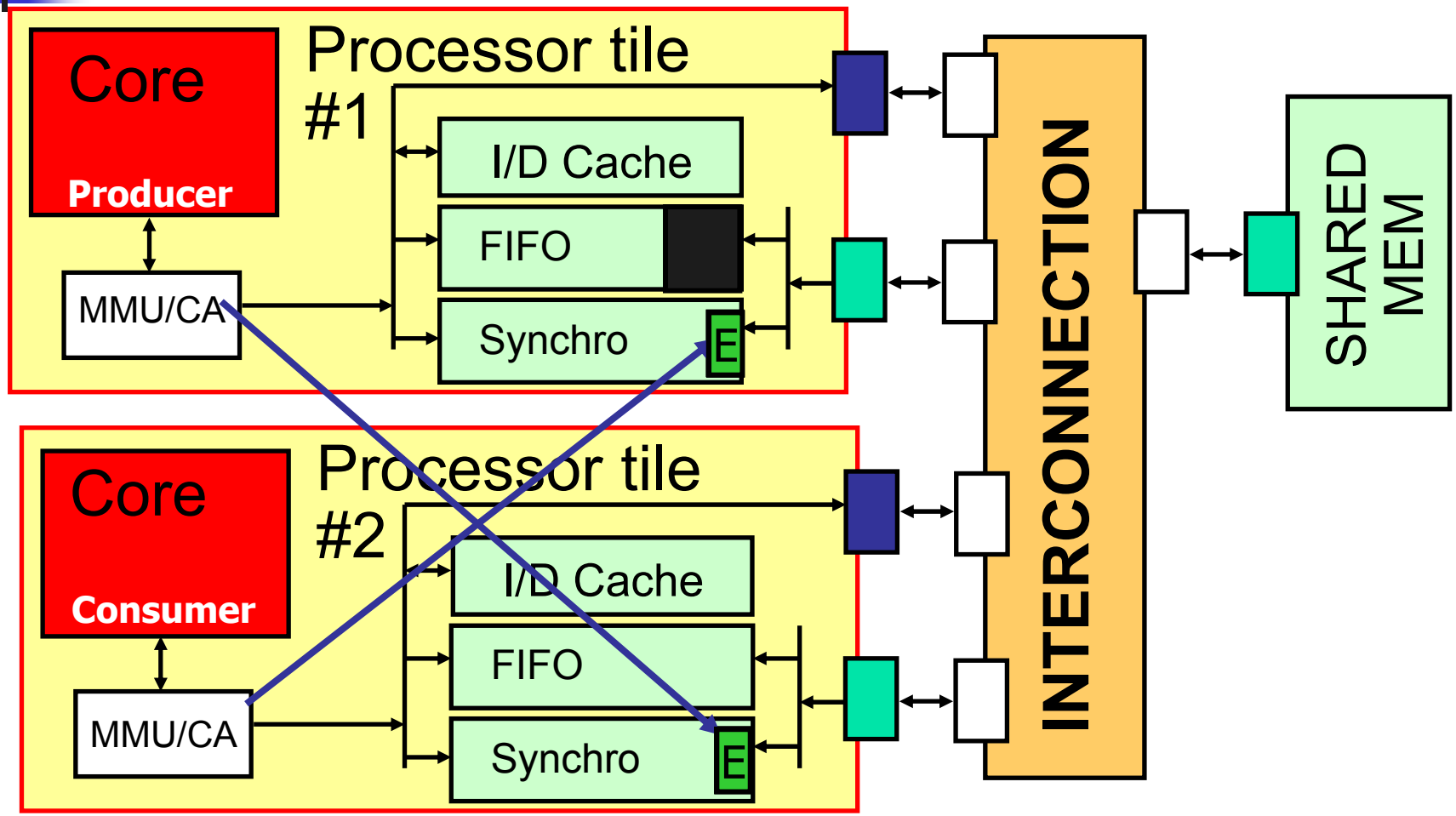
"Unfriendly" to NoC: lots of "blocking" reads

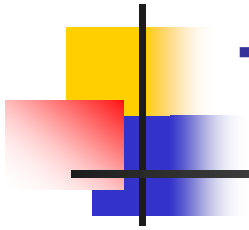
The trouble with UMA



"Unfriendly" to NoC: require distributed control of local memory state

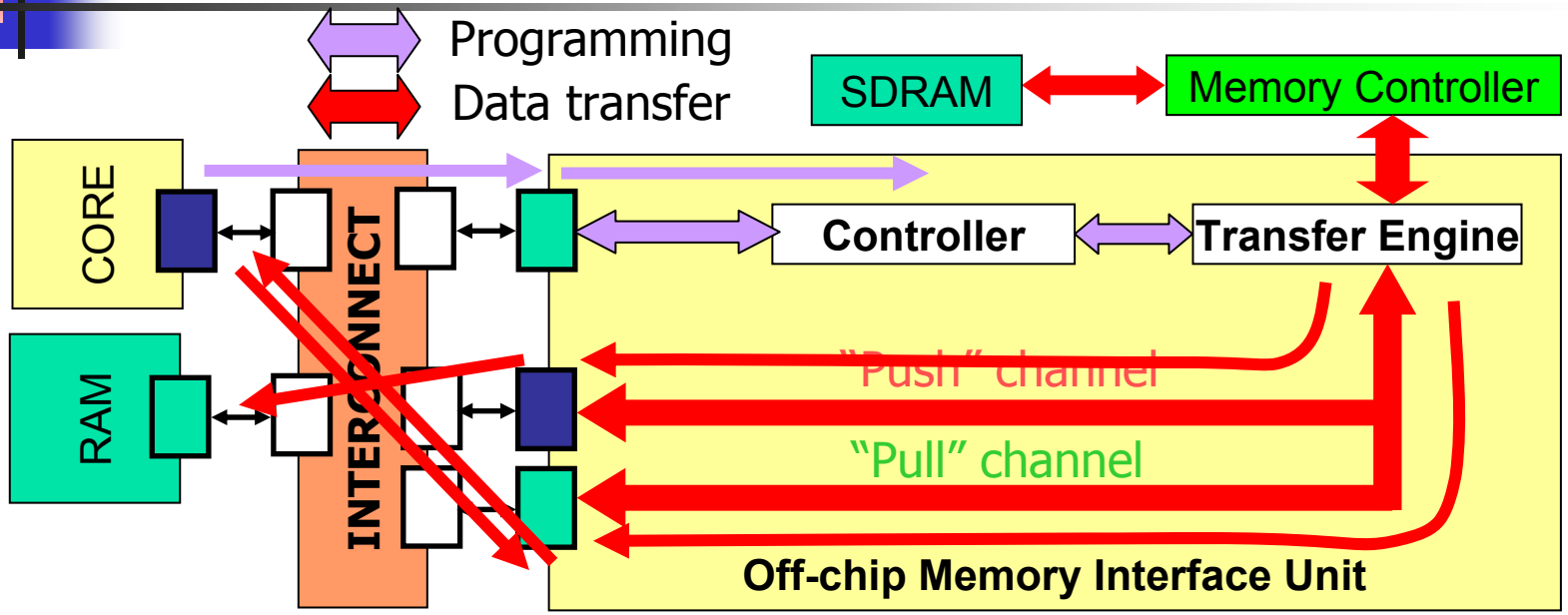
“NoC-friendly” stream processors



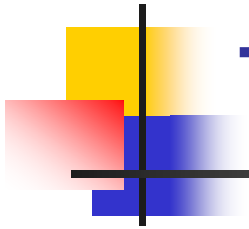


Toward stream processors

The memory bottleneck

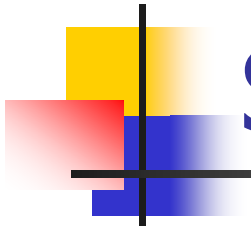


- *"Pull" memory channel*
  - *Control Block* keeps programmable table of objects to be moved
  - Table entries can be programmed by different cores
  - *Transfer Engine* shuffles data among bus and Memory Controller
  - Triggers bus or SDRAM transactions
  - *Memory Controller* handles SDRAM accesses
- Issues with wormhole: message-level deadlock



The SW view

- SW environments
  - OSes
  - Middleware
- Programming models
- Compilation, SW optimization



Summary

- Paradigm shift towards NoCs
- Designing NoCs
  - Huge design space
  - Demonstrate good potential
- Hardware/software interfaces, programming models and design methods have to be developed!