TSAR : a scalable shared memory many-cores architecture with global cache coherence

Alain Greiner

(alain.greiner@lip6.fr)

Outlin
e

- © The TSAR project
- © The low power challenge
- © The scalability challenge
- © The SoCLib virtual prototyping platform
- © First experimental results
- © Conclusion

The TSAR Project

TSAR is a Medea+ project that started in june 2008.

One central goal of this project is to define and implement a coherent, scalable shared memory, many-cores architecture : up to 4096 cores...

The project leader is BULL.

Industrial partners

ACE (Netherland)

Bull S.A. (France)

Compaan (Netherland)

NXP Semiconductors (Netherland)

Philips Medical Systems (Netherland)

Thales Communications (France)

Academic partners

Université Paris 6 (LIP6)

Technical University Delft (CEL)

University Leiden (LIACS)

FZI (Karlsruhe)

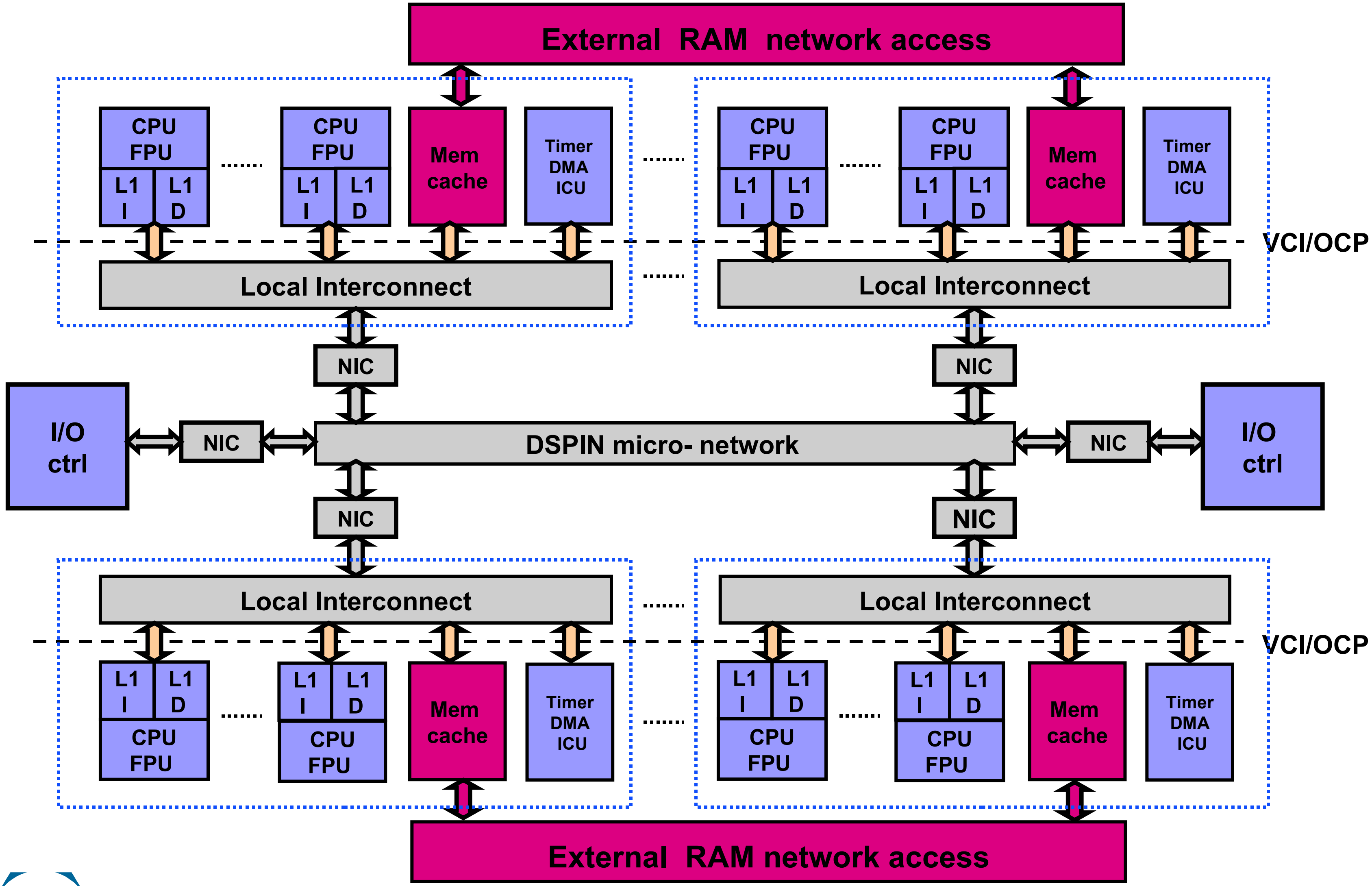
Main Technical Choices

- Processor : The architecture is independent on the selected processor core. It will an existing core such as : SPARC V8, MIPS32, PPC 405, ARM, etc.

- Interconnect : The TSAR architecture is clusterized, with a two-level interconnect. The global (inter-cluster) interconnect will use the DSPIN Network on Chip that supports the VCI/OCP standard.

- Memory : The Tsar architecture will support a (NUMA) shared address space. The memory is logically shared, but physically distributed, with cache coherence enforced by hardware, using a directory-based protocol.
  The physical address is 40 bits.

Clusterized Architecture



Outline

© The TSAR project

© The low power challenge

© The scalability challenge

© The SoCLib virtual prototyping platform

© First experimental results

© Conclusion

Low-power architecture (1)

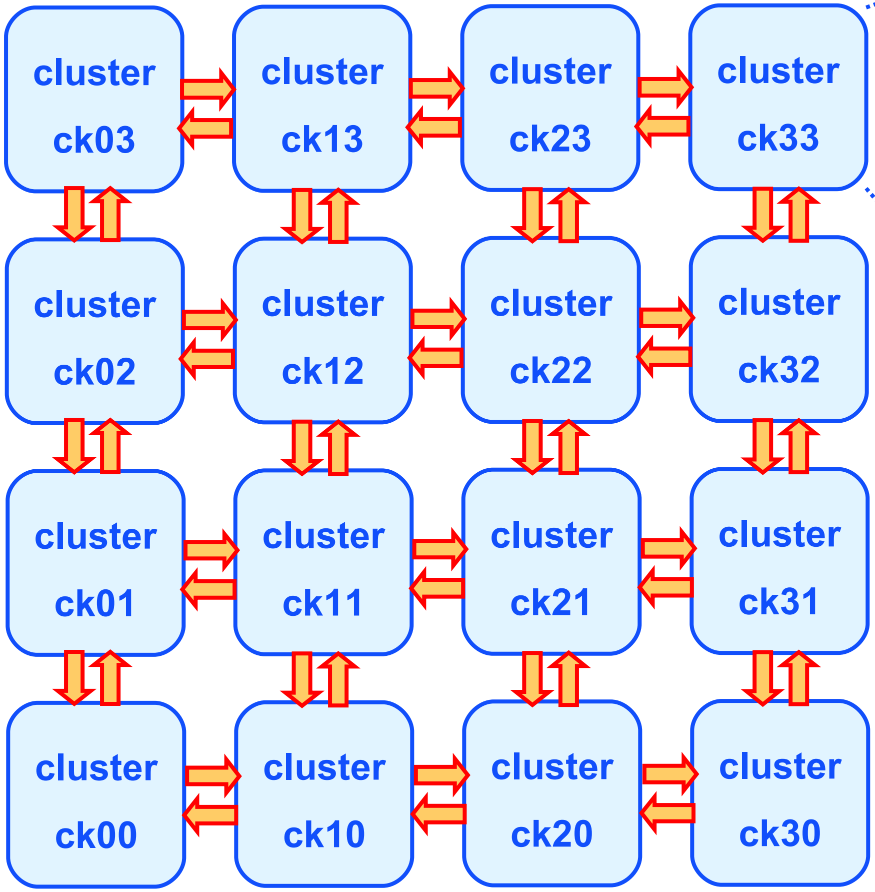
	Olf fashion Microprogrammed	Single instruction issue pipeline	Out of order Superscalar
instruction /cycle	0.2	1	2
power consumption	1	2	20
Oper/Joule	0.2	0.5	0.1

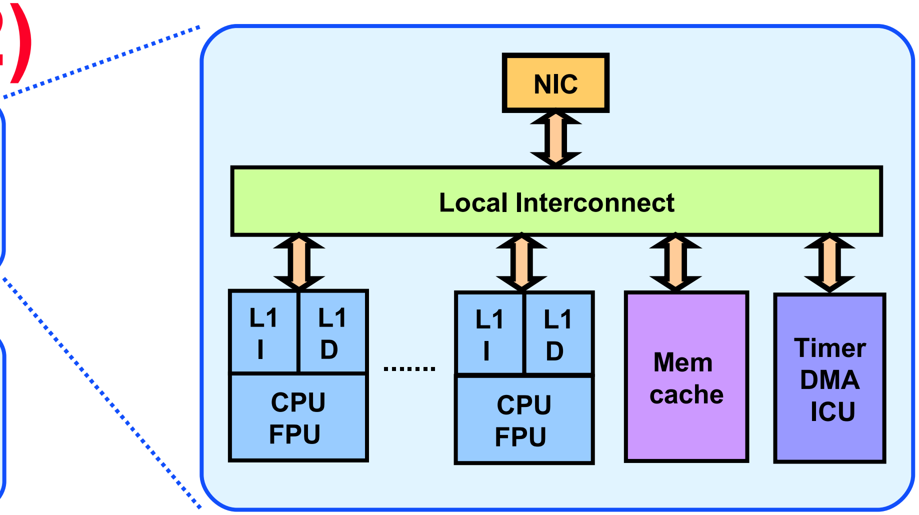
The TSAR processor will be an existing, simple 32 bits RISC processor ,

with single instruction issue (Mips32 in the first prototype) :

- No superscalar architecture,
- No multi-threading
- No speculative execution
- No out of order execution

Low-power architecture (2)

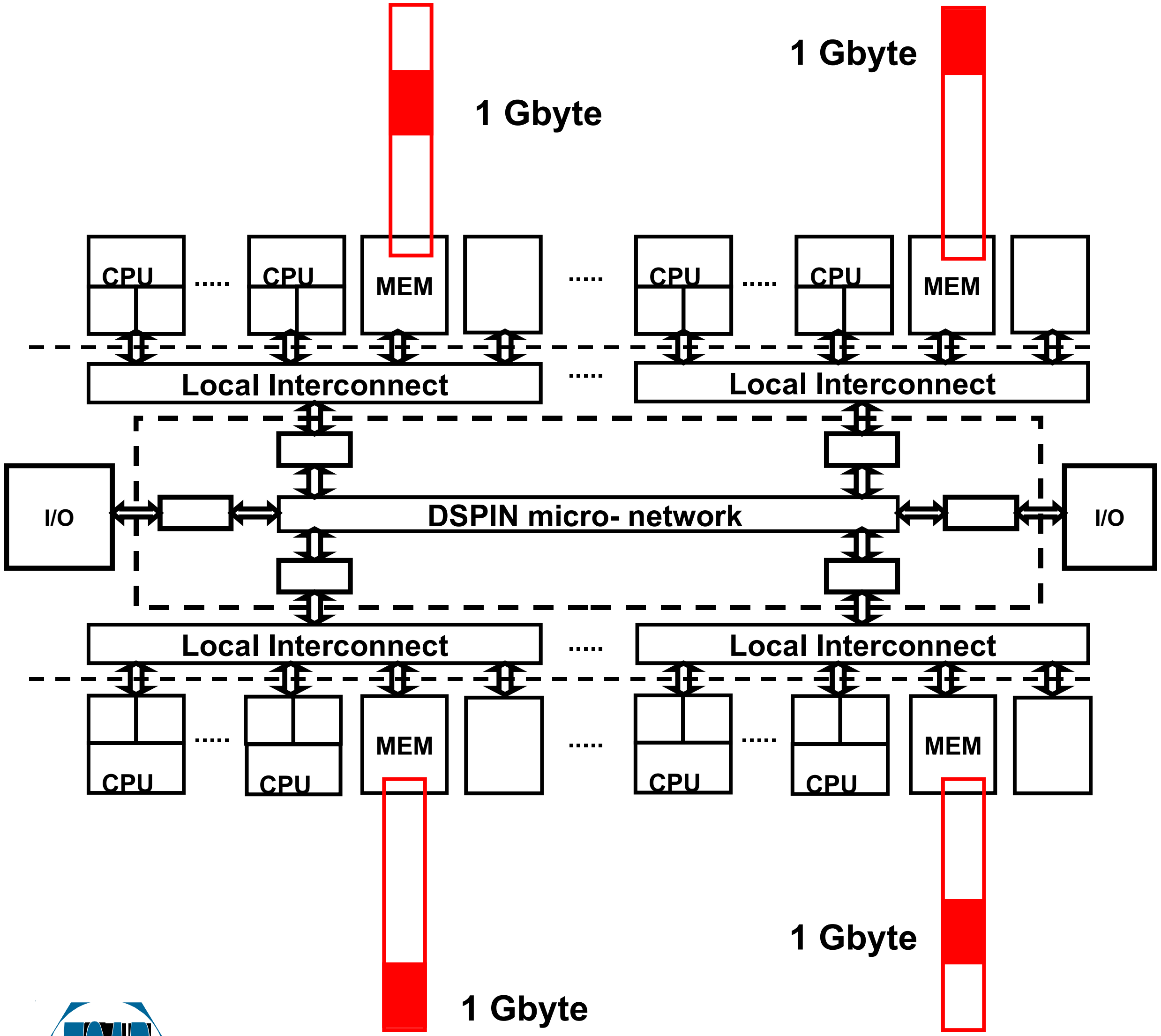




- TSAR will use the DSPIN Network on Chip technology developed by LIP6.
- each cluster is implemented in a different clock domain.
- inter-cluster communications use bi-synchronous (or fully asynchronous) FIFOs.

  => Clock frequency & Voltage can be independantly adjusted in each cluster.

Low-power architecture (3)



TSAR is a NUMA architecture

(Non Uniform Memory Access) :

local	remote	external
1	5	20

The physical memory space is statically distributed amongst the clusters.

The operating system can precisely control the mapping :

- tasks on the processors
- software objects on the memory caches.

Outline

© The TSAR project

© The low-power challenge

© The scalability challenge

© The SoCLib virtual prototyping platform

© First experimental results

© Conclusion

Cache coherence

© The coherence protocol must support up to 4096 processors.

© It must be guaranteed by the hardware for compatibility with software applications running on multi-core PCs.

© Both data & instruction caches are concerned by the cache coherence protocol.

© As snooping is not possible with a Network on Chip supporting a large number of simultaneous transactions, TSAR will rely on the general directory based approach : each memory cache must register all copies stored in the (4096 * 2) L1 caches...

The Write-Through / Write-Back issue

© In Write-Through policies, all write requests are immediately forwarded to the main memory : the L1 cache is always up to date.

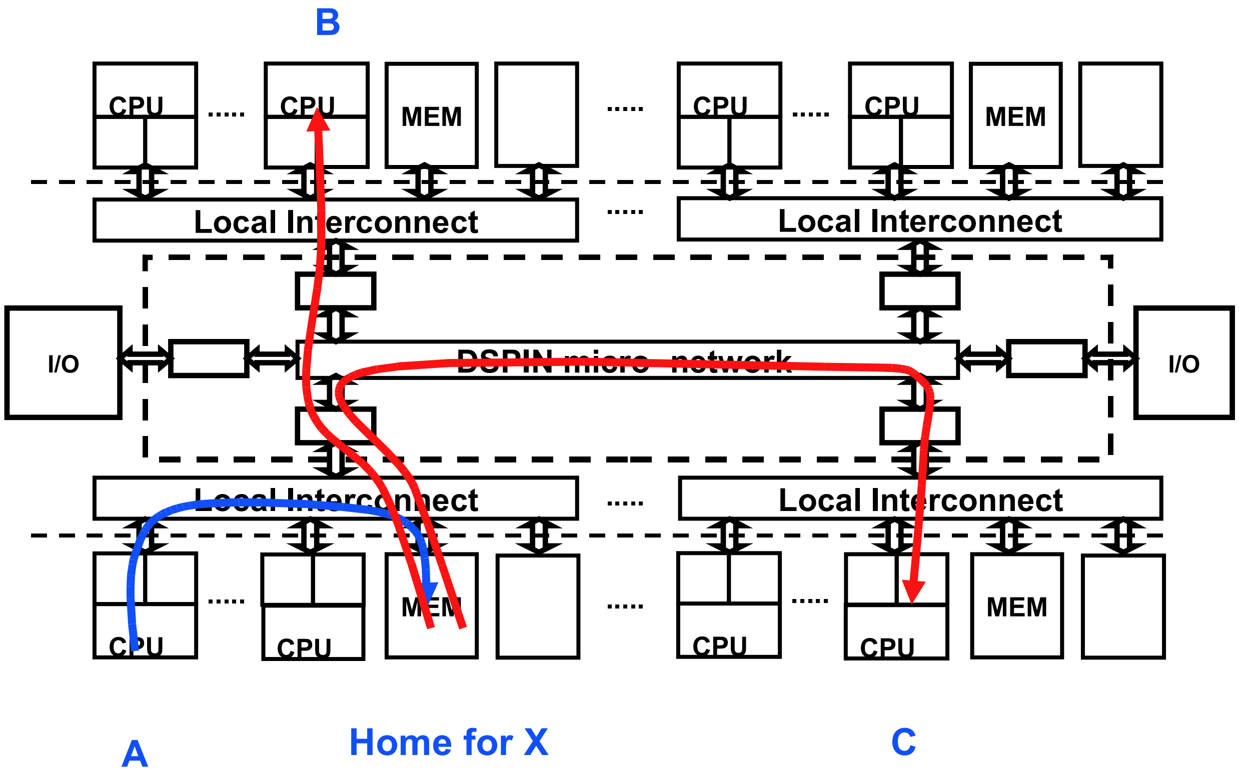
© In Write-Back policies, the L1 cache is updated, but the main memory is only modified when the modified cache line is replaced.

The Write-Back policy is generally used in discrete multi-processors architectures, as it saves transactions on the system bus, that is usually the architecture bottleneck, ... but it is - at least - one order of magnitude more complex than Write-Trough, because any L1 cache can acquire the exclusive ownership of any cache line.

The TSAR cache coherence protocol exploits the scalable bandwidth provided by the NoC technology, and choose to use a Write-Through policy.

... This is the main price to pay for protocol scalability...

Update / Invalidate messages



CPU A makes a write to an address X

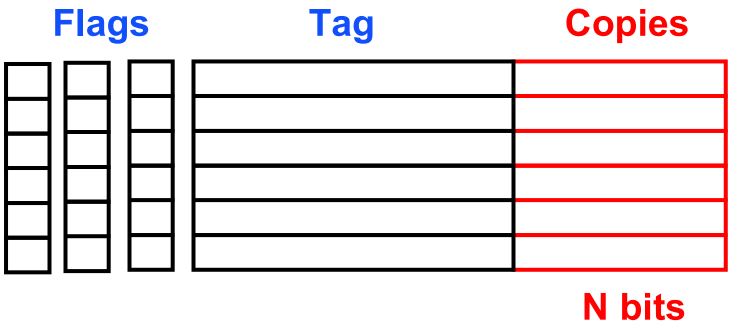
CPUs B & C have a copy

The memory cache that is the home directory for address X must send update or invalidate requests to CPU B & C

=> The key point is the implementation of the Distributed Global Directory...

Directory implementation (1)

Memory Cache



memory cache directory

Solution 1 : vector of bits + multi-cast policy

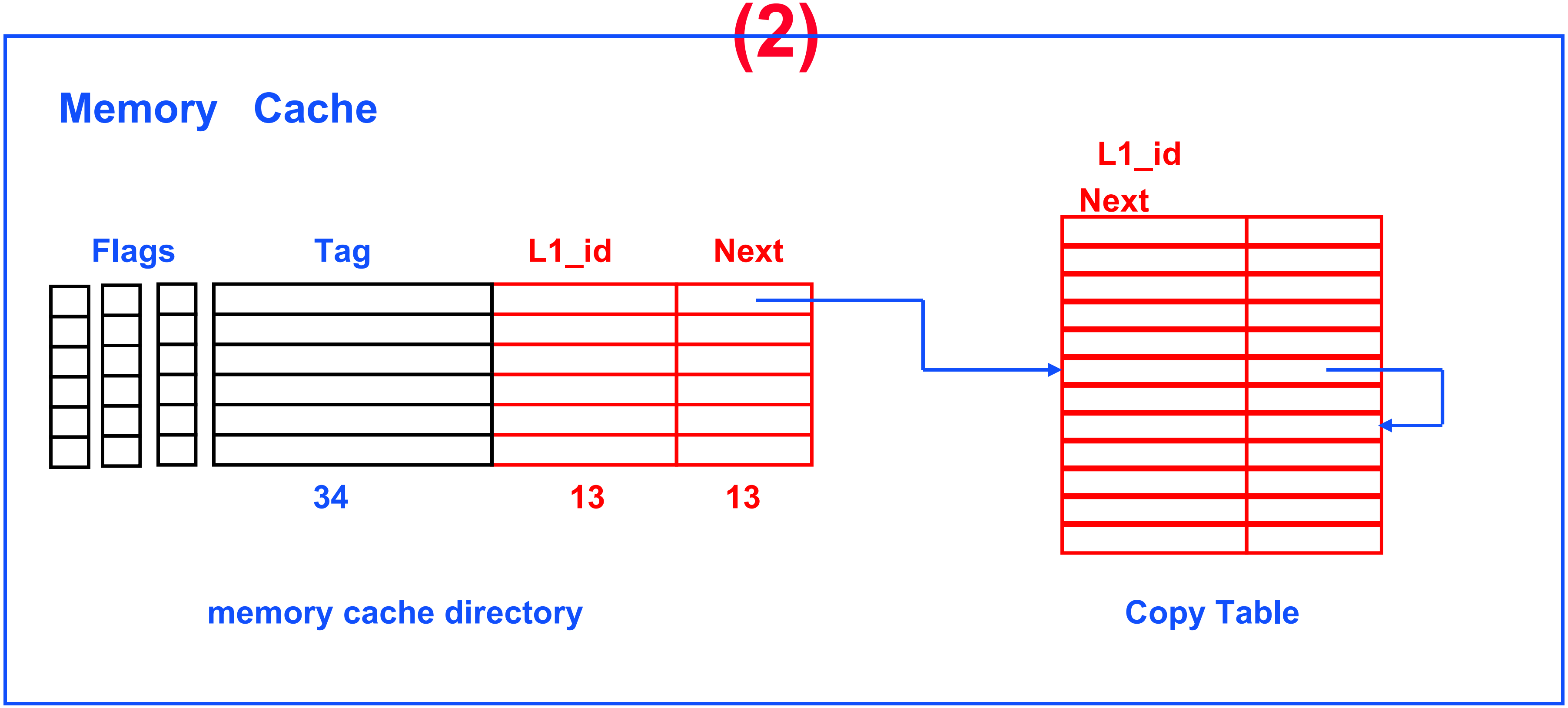
We use the characteristics of the memory cache :

The cache being inclusive, each cache line has a home directory in one single cache.

⇒ the cache directory is extended to include a vector of N bits (one bit per processor)

This approach is not scalable...

Directory implementation (2)

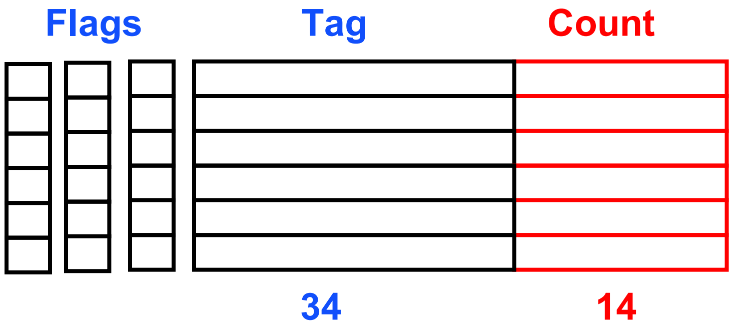


Solution 2 : chained lists of pointers in a local table + multi-cast policy

This solution is not scalable, as the Copy Table can overflow...

Directory implementation (3)

Memory Cache

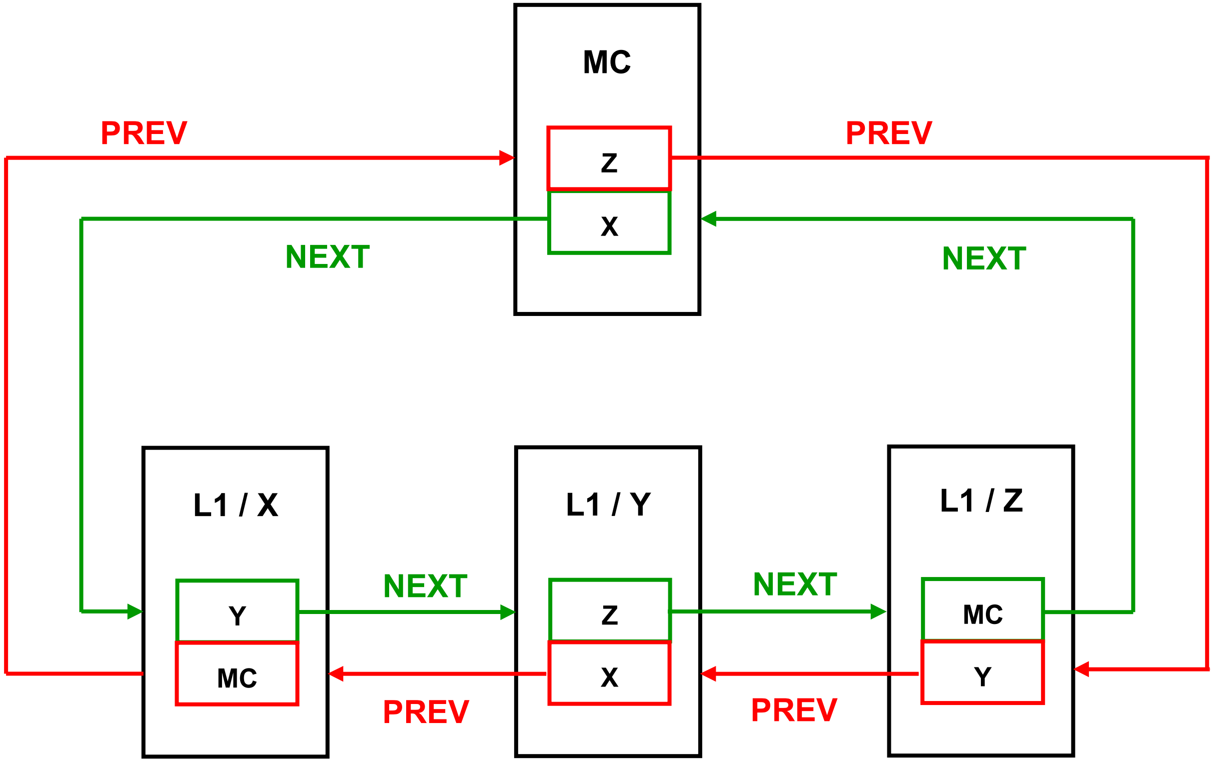


memory cache directory

Solution 3 : counter of copies + broadcast policy.

This solution is not scalable, as the coherence traffic will be too high...

Directory implementation (4)



Solution 4 : distributed linked list in memory cache & L1 caches + multi-cast policy

This solution is scalable, but the dead-lock prevention is a nightmare...

DHCCP cache coherence protocol

© TSAR implements the DHCCP protocol (Distributed Hybrid Cache Coherence Protocol), that is a mix of solution 2 & solution 3 :

- multicast / update when the number of copies is smaller than a predefined threshold
- broadcast / invalidate when the number of copies is larger than the threshold

© The DHCCP has been analysed, from the point of view of dead-lock prevention. Three types of traffic have been identified :

- Direct read/write transactions (L1 caches => Mem caches)
- Invalidate/Update/Cleanup transactions (Mem caches <=> L1 caches)
- External memory transactions (Mem caches => Ext. RAM)

=> Two fully separated virtual channels will be supported by the DSPIN network.

=> A broadcast service will be implemented in the DSPIN network.

=> access to the external RAM is implemented by a separated network.

Outlin
e

- © TSAR architecture principles
- © The low power challenge
- © The scalability challenge
- © The SoCLib virtual prototyping platform
- © First experimental results
- © Conclusion

The SoCLib Platform

© SoCLib is a cooperative project funded by the french authorities (“Agence Nationale de la Recherche”) : 6 industrial companies & 10 academic laboratories joined in 2006 to build an “open modeling & simulation platform for Multi-Processor System on Chip”.

© The core of the platform is a library of SystemC simulation models for reusable hardware components (IP cores), with a garanteed path to silicon.

© Several software tools (including embedded operating systems and communication middleware) are integrated in the platform.

© All SoCLib simulation models are available as open source software, under the LGPL license (www.soclib.fr)

SoCLib Partners

© ST Micro-electronics

© Thales

© Thomson

© Orange

© Magillem Data Service

© TurboConcept

- LIP6
- INRIA
- CEA-LIST
- ENST
- IRISA
- LESTER
- IETR
- CEA-LETI
- TIMA
- CITI

SoCLib technical choices

The main concern is interoperability :

- © All SoCLib components use the same communicationprotocol : VCI/OCP.
- © All SoCLib components will be packaged, using the IP-XACT (ex SPIRIT) description standard.
- © Two simulation models are provided for each IP core :
  - CABA : Cycle-Accurate / Bit-Accurate
  - TLM-DT : Transaction Level Model with Distributed Time

SoCLib Library (partial) content

General Purpose Processors	MIPS32 SPARC V8 MicroBlaze Nios ARM 7 ARM 9 PowerPC 450 PowerPC 750
Digital Signal Processors	ST200 C6X Texas
System Utilities	Interrupt controler Locks engine DMA controler MWMR controler
VCI/OCP Interconnects	Generic DSPIN Pibus Token Ring

Outlin
e

- © The TSAR project
- © The low power challenge
- © The scalability challenge
- © The SoCLib virtual prototyping platform
- © First experimental results
- © Conclusion

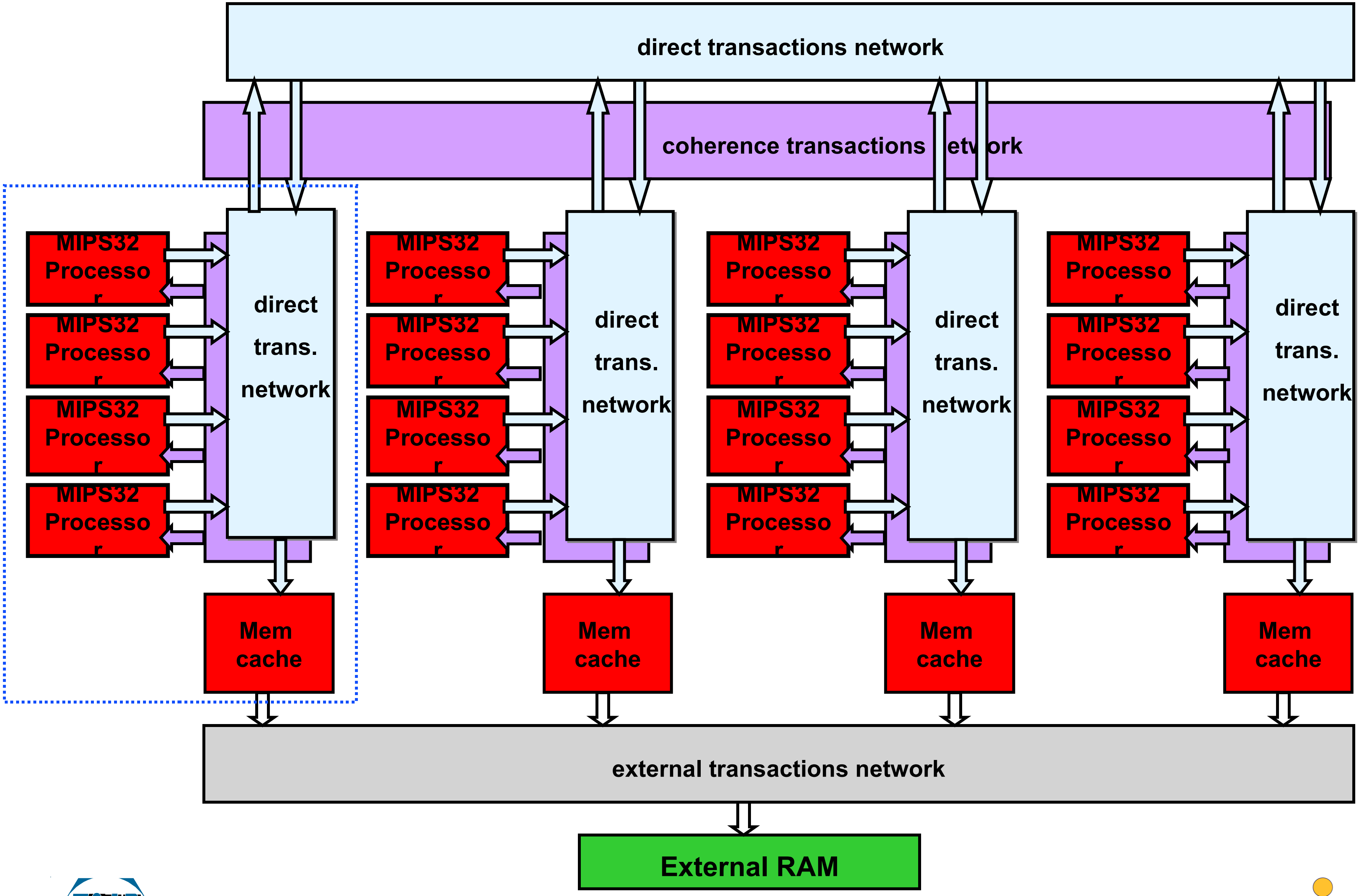
TSAR V0 virtual prototype

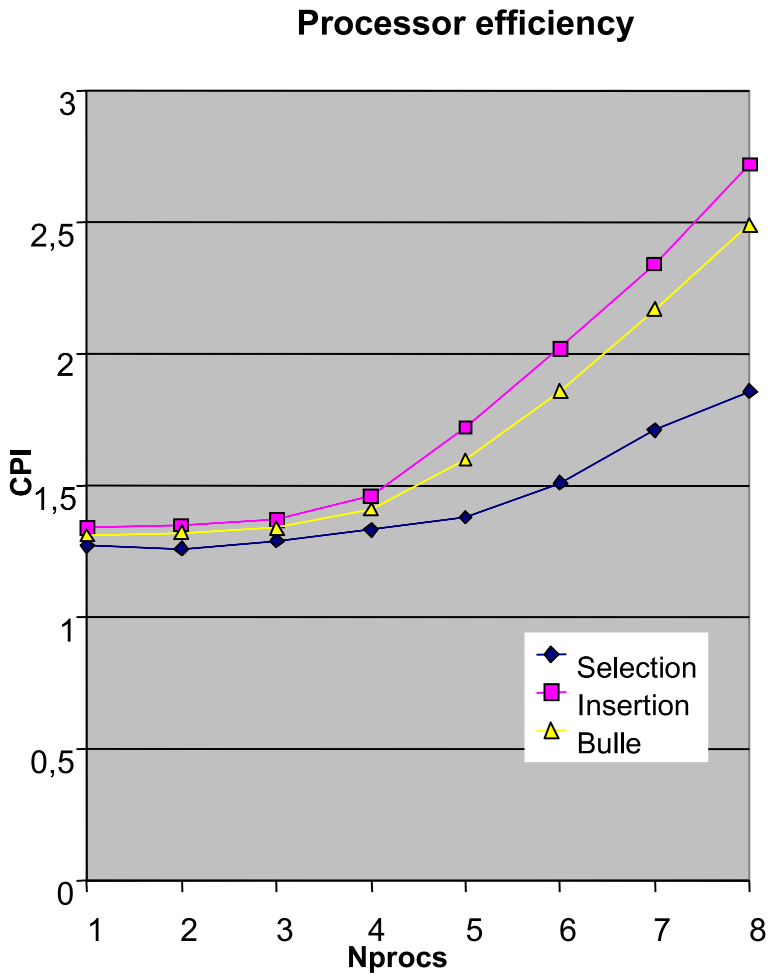
© The first - running - virtual prototype implements a simplified architecture :

- The DSPIN micro-network is replaced by a simplified interconnect (generic VCI micro-network, available in SoCLib).
- The HCCP coherence protocol is not fully implemented : broadcast/invalidate transactions are not supported, and the number of cores is limited to 16).

© The goal is to evaluate the hardware complexity of the “memory cache”, and the “optimal” number of processors per cluster.

© The multi-thread software application running on this virtual prototype is an insertion sort written in C (without embedded OS).


direct transactions network
coherence transactions etv ork
MIPS32 Processo r
MIPS32 Processo r
MIPS32 Processo r
MIPS32 Processo r
direct trans. network
Mem cache
external transactions network
External RAM



© In this measurement,

The CPI (CPI = average number of cycles per instruction) increases when the number of processor per cluster is larger than 4.

© The largest simulated configuration was a 16 processors architecture (4 clusters of 4 processors).

© The simulation speed (for cycle-accurate simulation) was 31 000 cycles/second on a 1.8 GHz Linux PC.

Conclusion

© The first year of the TSAR project resulted in the definition of an hybrid cache coherence protocol (HCCP), implementing different policies for data & instruction caches. We expect this architecture to be scalable up to 4096 processors (32 * 32 clusters), thanks to an advanced network on chip technology supporting both packet switched virtual channels and a broadcast service.

© The SoCLib virtual prototyping platform was very useful for quantitative analysis of various implementations of the TSAR architecture, especially to evaluate tradeofs between cost & performance.

© To promote the Tsar architecture, the SystemC virtual prototype will be available as open source software on the SoCLib WEB site : www.soclib.fr

Thank You

To get the slides : alain.greiner@lip6.fr