From multi to many-core: network on chip challenges

On-board software and multicores: opportunities and issues

CCT SIL-IRE, Mardi 16 Juin 2015

Marc Boyer

ONERA – The French Aerospace Lab

June 16, 2015
exact information are not always public
the meaning of words sometimes (...) differs
personal experience on limited set of chips

A good overview: “ISSCC TRENDS” http://isscc.org/trends

This and other related topics have been discussed at length at ISSCC 2015, the foremost global forum for new developments in the integrated-circuit industry.
Next session: Jan. 31 – Feb. 4 2016, San Francisco, CA
http://isscc.org
ONERA experience I

ONERA has been involved in several many-core projects:

- **MARC (2009-2012)**: Intel Many-Core Application Research Community academic program
- **MACSIMA (2013-2017)**: ONERA internal research project
  - topic: radar, planification and control-command application on the same many-core
- **DREAMS (2013-2017)**: FP7 European project
  - topic: mixed-criticality on multi-core
- **CAPACITÉS (2014-2018)**: LEOC (Logiciel Embarqué et Objets Connectés) french-founded project,
ONERA experience II

- topic: multicore for critical embedded systems
- partners: Airbus, Airbus Helicopter, ARMINES, Dassault Aviation, Inria, IRISA, IRIT, IS2T, Kalray, MBDA, ONERA, OpenWide, Probayes, Real-Time At Work, Safran, Supersonic Imagine, TIMA, Verimag

15 publications (2011-2015):
[16, 19, 23, 17, 20, 15, 21, 3, 1, 8, 6, 7, 11, 13, 14]
Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
Outline

Many-cores are arriving

Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
Many-cores are arriving

Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC ?

Conclusion
Some trends [25]

- **CHIP COMPLEXITY**
  - Transistor Count (Millions)
  - 5.6 billion!!!

- **CORE COUNT**
  - Cores per Die

- **TOTAL ON DIE CACHE**
  - Cache Size (MB)

- **CLOCK FREQUENCY**
  - Clock Frequency (MHz)

The relentless march of process technology brings more integration and energy-efficient performance to mainframes, enterprise and cloud servers. ISSCC 2015 features IBM's high-frequency 8-core, 16-thread System z mainframe processor in 22nm SOI with 64MB of eDRAM L3 cache and 4MB/core eDRAM L2 cache. The SPARC M7 processor from Oracle implements 32 S4 cores, a 1.6TB/s bandwidth 64MB L3 Cache and a 0.5TB/s data bandwidth on-chip network (OCN) to deliver more than 3.0x throughput compared to its predecessor. 280 SerDes lanes support up to 18Gb/s line rate and 1TB/s total bandwidth. Intel's next generation Xeon processor supports 18 dual-threaded 64b Haswell cores, 45MB L3 cache, 4 DDR4-2133MHz memory channels, 40 8GT/s PCIe lanes, and 60 9.6GT/s QPI lanes. It has 5.56B transistors in Intel's 22nm tri-gate HKMG CMOS and achieves a 33% performance boost over previous generations.

The maximum core clock frequency seems to have saturated in the range of 5-6GHz, primarily limited by thermal considerations. The nominal operating frequency of the power-limited processors this year is around 3.5GHz. Core counts per die are typically above 10, with increases appearing to slow in recent years. Cache size growth continues, with modern chips incorporating tens of MB on-die.

The trend towards digital phase-locked loops (PLL) and delay locked loops (DLL) to better exploit nanometer feature size scaling, and reduce power and area continues. Through use of highly innovative architectural and circuit design techniques, the features of these digital PLLs and DLLs have improved significantly over the recent past. Another new trend evident this year is towards fully digital PLLs being synthesizable and operated with non-LC oscillators. The diagram below shows the jitter performance vs. energy cost for PLLs and multiplying DLLs (MDLL).
Cache memory

More cores, and more cache
- cache consumes few energy
- cache is efficient

But...
Cache memory

More cores, and more cache
- cache consumes few energy
- cache is efficient

But...
- how to ensure cache coherency with 32 cores?
Cache memory

More cores, and more cache
- cache consumes few energy
- cache is efficient

But...
- how to ensure cache coherency with 32 cores?
- why?
Cache memory

More cores, and more cache
- cache consumes few energy
- cache is efficient

But...
- how to ensure cache coherency with 32 cores?
- why?
- local cache or local memory?
More cores, and more cache
- cache consumes few energy
- cache is efficient

But...
- how to ensure cache coherency with 32 cores?
- why?
- local cache or local memory?
- implicit or explicit communications?
  - message passing vs shared memory
Cache memory

More cores, and more cache

- cache consumes few energy
- cache is efficient

But...

- how to ensure cache coherency with 32 cores?
- why?

local cache or local memory?

implicit or explicit communications?

- message passing vs shared memory

an old/new programming way
Many-cores are arriving
   Trends

Network on Chip
   Overview
   Routing
   Contention
   Other solutions
   Tiles

Three selected solutions
   Intel SCC
   Spidergon STNoC
   The Kalray MPPA

A real-time NoC?

Conclusion
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC ?

Conclusion
Network on chip

- how to connect chip elements?
- NoC for SoC vs NoC for multicore
  - homogeneous vs heterogeneous system
  - access to main memory and IO
- different approaches depending on manufacturer
- less information than on cores
A Survey of Research and Practices of Network-on-Chip

Figure: From bus to NoC

- **Bus**: shared resource
- **Pt-to-pt**: does not scale
- **NoC**:
  - set of shared resources
  - allow parallel communications
A common vocabulary

**Figure**: Architecture elements [2]

- **Core/tile**: could be also IO/RAM
  - write/read messages

- **Network adapter**
  - fragment/reassemble messages into packets
  - send/receive packets
  - flow control

- **Routing node**: commutation element
  - send/receive *flits* ($\approx 64$ bits)
  - also flow control
Core/tile: could be also IO/RAM
- write/read messages

Network adapter
- fragment/reassemble messages into packets
- send/receive packets
- flow control

Routing node: commutation element
- send/receive flits ($\approx 64$ bits)
- also flow control
Topologies

Fig. 9. Regular forms of topologies scale predictably with regard to area and power. Irregular forms of topologies scale in an asymmetric fashion as seen in Figure 10. Irregular forms of topologies are derived by mixing different forms in a hierarchical, hybrid, or asymmetric fashion. Examples are (a) 4-ary 2-cube mesh, (b) 4-ary 2-cube torus, and (c) binary fat tree. For a torus, k-ary n-dimensional tree-based topologies are useful for exploiting locality of traffic. Generally, mesh topology makes better use of links (utilization) while it has longer delays between routing nodes. Figure 9 shows examples of regular forms of topology. Most NoCs implement regular forms of network topology that can be laid out on a chip surface (a 2-dimensional plane) dictatedly for increasing size of regular forms of topology. Meshe-based topologies are useful for maximizing the utilization of links, while tree-based topologies are useful for exploiting locality of traffic. For more complex structures such as trees, finding the optimal shortest-path routing algorithm is a challenge on its own right. Such rings are then connected edge-to-edge to form a larger, scalable network. For example, the Octagon NoC for the k-ary n-cube is an example of a novel regular NoC topology. Its basic configuration is a ring of 8 nodes connected by 12 bidirectional links which provides two-hop communication between any pair of nodes in the ring and a simple, shortest-path routing algorithm. Such rings are then connected edge-to-edge to form a larger, scalable network. For more complex structures such as trees, finding the optimal shortest-path routing algorithm is a challenge on its own right. Such rings are then connected edge-to-edge to form a larger, scalable network. For more complex structures such as trees, finding the optimal shortest-path routing algorithm is a challenge on its own right. Such rings are then connected edge-to-edge to form a larger, scalable network. For more complex structures such as trees, finding the optimal shortest-path routing algorithm is a challenge on its own right. Such rings are then connected edge-to-edge to form a larger, scalable network. For more complex structures such as trees, finding the optimal shortest-path routing algorithm is a challenge on its own right. Such rings are then connected edge-to-edge to form a larger, scalable network.
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC ?

Conclusion
Routing:

XY: follows the row first, then moves along the column

Note: reverse communication uses another path

Source routing: source set the path in the header

Adaptative: route computed "on the fly"
minimize link/router load
research only?
Routing:
- **XY**: follows the row first, then moves along the column
  
  Note: reverse communication uses another path
Routing:
- **XY**: follows the row first, then moves along the column
  - Note: reverse communication uses another path
- **Source routing**: source set the path in the header
Routing:

- **XY:** follows the row first, then moves along the column
  - Note: reverse communication uses another path
- **Source routing:** source set the path in the header
- **Adaptative:**
  - route computed “on the fly”
  - minimize link/router load
  - research only?
Multicast: sending the same data to several cores

- multicast NoC: data send once, path sharing
- non multicast NoC: data sent several times, path competition
Multicast: sending the same data to several cores

- multicast NoC: data send once, path sharing
- non multicast NoC: data sent several times, path competition
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
They always are contentions

- arbitration is needed
  - link scheduling policy
  - storage is needed
    - memory allocation policy
      - such memory is expensive

⇒ large set of solutions
Forwarding: how to transmit data?

- store & forward
- wormhole
- virtual circuit
- virtual cut-through
Store and forward

- common policy in network
- send and store full packets in routers
- require buffer size many times larger than packet size
Wormhole forwarding

- like Spacewire
- forward flits even while receiving
- send flits up to blocking
  ⇒ link flow control
- allow large messages / packets
- implies blocking, and even dead-locks
Virtual circuit enhancement

- reduces blocking
- require more logic
  - flit tagging
  - VC id allocation
  - memory sharing
  - arbitration policy
- allows per VC QoS
Virtual cut-through forwarding

- looks like wormhole restriction
Virtual cut-through forwarding

- looks like wormhole restriction
- or store&forward enhancement
Virtual cut-through forwarding

- looks like wormhole restriction
- or store&forward enhancement
- send packet only if enough space in next router
Virtual cut-through forwarding

- looks like wormhole restriction
- or store&forward enhancement
- send packet only if enough space in next router
  ⇒ require storage of full packet
## Forwarding: feedback

<table>
<thead>
<tr>
<th>Forwarding</th>
<th>Per node cost</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Latency</td>
<td>Memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>store &amp; forward</td>
<td>packet</td>
<td>packet</td>
<td>Common in networks</td>
<td></td>
</tr>
<tr>
<td>wormhole</td>
<td>header</td>
<td>header</td>
<td>Blocking</td>
<td></td>
</tr>
<tr>
<td>virtual cut-through</td>
<td>header</td>
<td>packet</td>
<td>A trade-off (?)</td>
<td></td>
</tr>
</tbody>
</table>

- wormhole has good mean performances
- ... but can lead to dead-locks
- virtual circuit has less blocking
- ... but require more memory and logic
- store& forward / virtual cut-through are well known
- ... but require more memory or small messages
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC ?

Conclusion
The time-triggered approach

Build a global TDMA schedule

+ avoid any contention
+ small network delays
- require periodic tasks/communications
- does it scale?
Connection oriented

Avoid contention in network, by establishing core-to-core connection, and resource reservation.

- increases latency (connection establishment)
- increases logic
- research only?
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
Tile-based solutions

- Initial architecture: [24], MIT, 2007
- Tile:
  - local multi-core
  - DRAM, I/O...
- NoC between tiles
- Hierarchical design
  ⇒ multi-core interferences + NoC interferences
Outline

Many-cores are arriving
Trends

Network on Chip
Overview
Routing
Contention
Other solutions
Tiles

Three selected solutions
Intel SCC
Spidergon STNoC
The Kalray MPPA

A real-time NoC ?

Conclusion
Outline

Many-cores are arriving
Trends

Network on Chip
Overview
Routing
Contention
Other solutions
Tiles

Three selected solutions
Intel SCC
Spidergon STNoC
The Kalray MPPA

A real-time NoC ?

Conclusion
Intel SCC architecture

- experimental processor [22]
- 24 tiles
- 2 cores per tile
- 2Tb/s bisection bandwidth
- explicit message passing (but virtual global addressing)
- Virtual Circuit forwarding
- 8 VC for the whole router
- crossbar output
Many-cores are arriving

Trends

Network on Chip

Overview
Routing
Contention
Other solutions
Tiles

Three selected solutions

Intel SCC

Spidergon STNoC

The Kalray MPPA

A real-time NoC ?

Conclusion
the ring and the optimization based on the spidergon cross-links only, the SFR method shows a few added cross-links, only one is "long". The FFS solution, which utilizes the degrees of freedom given by the initial mapping on spidergon, provides the best results in terms of area (3.77% and -8.03% respectively for power and network latency) with respect to the ring. Confirming the analysis of STMicroelectronics, the network on-chip architecture developed by STMicroelectronics. Starting from different initial mappings, these methods (FFR and FFS) have been compared in terms of network area, latency, and number of added links. For area, power consumption, and number of added links, the FFS solution has been selected as the best topology for each approach in the context of on-chip networks.

A NoC for SoC (\(\subseteq\)) [5, 18]
application specific topology
Time-Triggered scheduling
soon
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC ?

Conclusion
Kalray architecture

- A 256-cores chip [9]
- torus topology
- 16 tiles
- 16 “simple” cores per tile
Kalray Network Adapter

- 8 channels [10]
- explicit communications
- per channel traffic limiter

⇒ HW support for latency computation
2.1 General Problem Formulation

2.1.1 Variables and Constraints

- Application constraints: some flows require a minimal bandwidth guarantee: the application minimal bandwidth requirements;
- Link capacity constraints: the link capacity constraints are straightforward, the queue backlog constraints require more attention to details. Flows coming from different directions and sharing the same link may not occur as this will result in saturation of the link, so we avoid the aggregation issues implied by the FIFO-multiplexed in rate-latency network elements, the service curves offered to a single flow are not necessarily themselves rate-latency curves [12];
- Queue backlog constraints: any FIFO queue may not overflow;
- Over-booking of links so flows interfere on a node only if they share a link to the next node. This interface performs a round-robin (RR) arbitration a round-robin arbitration is work-conserving and the NoC link arbiters are work-conserving and the NoC.

2.2 MPPA D-NoC calculus

The MPPA D-NoC calculus is to compute the link, so we avoid the aggregation issues implied by the FIFO-multiplexed in rate-latency network elements, the service curves offered to a single flow are not necessarily themselves rate-latency curves [12];

over-booking of links so flows interfere on a node only if they share a link to the next node. This interface performs a round-robin (RR) arbitration is work-conserving and the NoC link arbiters are work-conserving and the NoC.

The packetization effects are taken care of by considering the latency is necessarily itself rate-latency curve [12];

the service curves offered to a single flow are not necessarily themselves rate-latency curves [12];

The link arbiters are work-conserving and the NoC link arbiters are work-conserving and the NoC.

The packetization effects are taken care of by considering the service curves offered to a single flow are not necessarily themselves rate-latency curves [12];

• virtual cut-through forwarding
• round-robin arbitration

Kalray Network router

Kalray Network router

Kalray Network router

Kalray Network router

Kalray Network router

Kalray Network router

Kalray Network router
Outline

Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
Real-time with NoC?

Challenge:

- bound Worst case Interference Time (WCIT)

⇒ bound NoC Worst Case Traversal Time (WCTT)
Real-time with NoC?

Challenge:
- bound Worst case Interference Time (WCIT)
  ⇒ bound NoC Worst Case Traversal Time (WCTT)

Real-Time NoC:
- there will be one: TT extension of STMicro Spidergon STNoC
- there are some HW mechanisms
  - deactivation of cache coherency
  - bandwidth limiters
Challenge:
- bound Worst case Interference Time (WCIT)
- bound NoC Worst Case Traversal Time (WCTT)

Real-Time NoC:
- there will be one: TT extension of STMicro Spidergon STNoC
- there are some HW mechanisms
  - deactivation of cache coherency
  - bandwidth limiters

Solutions:
- execution model
- analyse methods
Many-cores are arriving
  Trends

Network on Chip
  Overview
  Routing
  Contention
  Other solutions
  Tiles

Three selected solutions
  Intel SCC
  Spidergon STNoC
  The Kalray MPPA

A real-time NoC?

Conclusion
Conclusion

The bad news

The good news
Conclusion

The bad news

- no large real-time processor market

The good news

⇒ less implicit communication

The times they are changing from shared memory to message passing
**Conclusion**

<table>
<thead>
<tr>
<th>The bad news</th>
<th>The good news</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>no large real-time processor market</em></td>
<td><em>The times they are changing</em></td>
</tr>
<tr>
<td><em>neither NoC one</em></td>
<td><em>from shared memory to message passing</em></td>
</tr>
</tbody>
</table>

⇒ less implicit communication
Conclusion

The bad news

- no large real-time processor market
- neither NoC one
- the NoC is a common shared resource

The good news

⇒ less implicit communication

The times they are changing from shared memory to message passing
Conclusion

The bad news
- no large real-time processor market
- neither NoC one
- the NoC is a common shared resource

The good news
- cache coherence is no more affordable
  ⇒ less implicit communication

The times they are changing from shared memory to message passing
Conclusion

The bad news

- no large real-time processor market
- neither NoC one
- the NoC is a common shared resource

The good news

- cache coherence is no more affordable
  \[\Rightarrow\text{less implicit communication}\]

The times they are changing

- from shared memory to message passing
Outline

Why multi-cores
  Theoretical limits
  Architecture impact
Outline

Why multi-cores

Theoretical limits

Architecture impact
Some limits \([4, 12]\)

**Moore’s law**

The transistor density doubles every generation.

**Pollack’s rules**

Performance (of a single core) is roughly proportional to \(\text{square root}\) of the number of transistors.
### Amdahl’s law

Given a program, with fraction $f \in [0, 1]$ that can be executed in parallel. Then, the speed-up with $n$ cores is bounded by

$$
\frac{1}{(1 - f) + \frac{f}{n}}
$$

(1)
More limits \([4, 12]\)

**Amdahl’s law**

Given a program, with fraction \(f \in [0, 1]\) that can be executed in parallel. Then, the speed-up with \(n\) cores is bounded by

\[
\frac{1}{(1 - f) + \frac{f}{n}}
\]

**Gustafson’s law**

Given a program, with fraction \(f \in [0, 1]\) that can be executed in parallel, \(n\) processors allows to handle a problem \(\frac{n}{n+f(1-n)}\) larger.
Amdahl’s law

Given a program, with fraction \( f \in [0, 1] \) that can be executed in parallel. Then, the speed-up with \( n \) cores is bounded by

\[
\frac{1}{(1 - f) + \frac{f}{n}}
\]

Gustafson’s law

Given a program, with fraction \( f \in [0, 1] \) that can be executed in parallel, \( n \) processors allows to handle a problem \( \frac{n}{n + f(1-n)} \) larger.

Corollary

With more processors, programmers find more parallelism in problems...
### The power limit

The capacity ($C$) is technology dependant. Cache memory uses few energy.

\[ W = CV^2f \]

<table>
<thead>
<tr>
<th></th>
<th>Power</th>
<th>GFlop/Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono/Quadri-Core</td>
<td>150W</td>
<td>7GF/W</td>
</tr>
<tr>
<td>GPGPU</td>
<td>300W</td>
<td>10GF/W</td>
</tr>
<tr>
<td>Many-core</td>
<td>30W</td>
<td>70GF/W</td>
</tr>
</tbody>
</table>
Outline

Why multi-cores
  Theoretical limits
  Architecture impact
Symmetric multicores?

Forecast [12, 4]: next chips will have

- a few “large” cores for sequential part
- several “small” cores for parallel part
- on the same chip? (multi-core vs GPGPU)


Benoît Dupont de Dinechin, Yves Durand, Duco van Amstel, and Alexandre Ghiti. Guaranteed services of the noc of a manycore processor. In *Proc. of the 7th Int. Workshop on Network on Chip Architectures (NoCArc’14)*, Cambridge, United Kingdom, December 2014.


Angeliki Kritikakou, Olivier Baldellon, Claire Pagetti, Christine Rochange, and Matthieu Roy.
Run-time control to increase task parallelism in mixed-critical systems.

Angeliki Kritikakou, Claire Pagetti, Christine Rochange, Matthieu Roy, Madeleine Faugère, Sylvain Girbal, and Daniel Gracia Pérez.
Distributed run-time wcet controller for concurrent critical tasks in mixed-critical systems.

Alessandra Melani, Eric Noulard, and Luca Santinelli.
Learning from probabilities: Dependences within real-time systems.
Eric Noulard.  
Real-time scheduling on multi/many-processor: from specification to real-time execution, August 2015.  

The ROSACE case study: From simulink specification to multi/many-core execution.  

Gianluca Palermo, Giovanni Mariani, Cristina Silvano, Riccardo Locatelli, and Marcello Coppola.  
Mapping and topology customization approaches for application-specific stnoc designs.  

Wolfgang Puffitsch, Eric Noulard, and Claire Pagetti.  
Off-line mapping of multi-rate dependent task sets to many-core platforms.  
submitted to journal (currently reviewed).


Luca Santinelli, Frédéric Boniol, Eric Noulard, Claire Pagetti, and Wolfgang Puffitsch.  
Integrated development framework for safety-critical embedded systems.  

Michael Bedford Taylor.  
*Tiled Microprocessors.*  

Axel Thomsen, Boris Murmann, Andreia Cathelin, Aarno Pärssinen, Daniel Friedman, T. J. Watson, Stephen Kosonocky, Stefaon Rusu, Joo Sun Choi, Makoto Ikeda, and Eugenio Cantatore.  
Isscc 2015 trends.  