Mirko Gagliardi

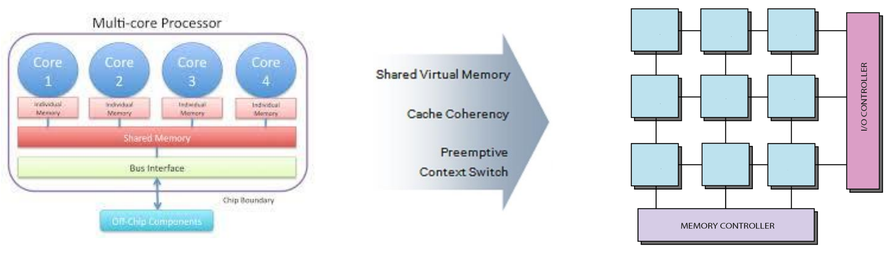
Tutor: Alessandro Cilardo

XXXI Cycle - II year presentation

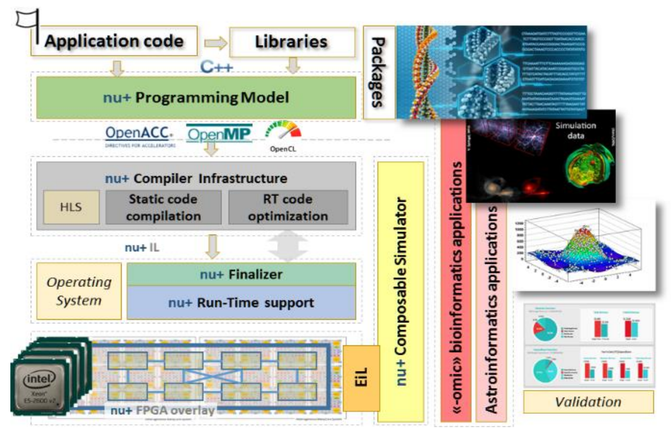
Exploring Advanced Computer Architectures and Techniques to Improve Performance and Power Efficiency in Manycore Era

Manycore Era

High-performance architectures mostly rely on general-purpose compute units such as CPUs connected through a shared bus. Those solutions deliver adequate performance while ensuring programmability and application portability. Unfortunately, they are affected by inherently limited power-efficiency and lack of scalability.



Current trends in HPC are increasingly moving towards manycore based solutions. Such systems are based on a considerable number of lightweight cores typically connected through a Network-on-Chip (NoC), providing a scalable approach to the interconnection of parallel on-chip systems.



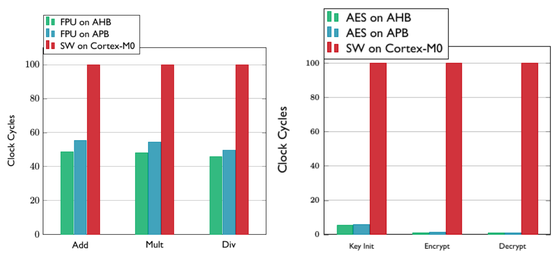
My research activity investigates manycore architecture exploration, the architectural requirements of emerging HPC applications and the most suitable power efficient and high-performance solutions.

Idea

Architectural customization might play a key role in modern high-performance architectures, as it enables unprecedented levels of power-efficiency compared to traditional systems.

Proof of concept

Two custom hardware IPs, namely an AES cipher and FPU, have been integrated in a SoC system based on Arm Cortex-M0 DesignStart project. Their performance are compared with a software version running on the Cortex-M0. The hardware overhead is less than 2% on the overall design.

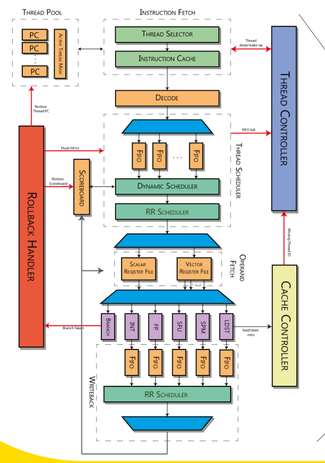


Problem

Modern open-source solutions are SoC based and not fully parametrizable (e.g. fixed ISA).

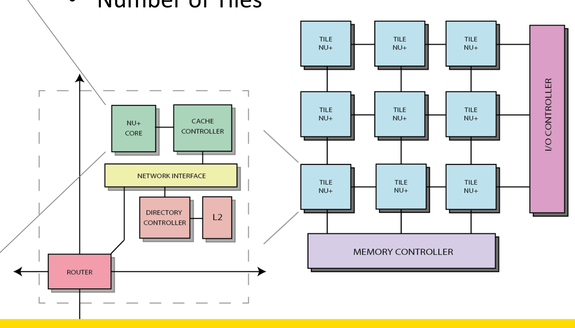
Open-source and customizable manycore system

The main objective of nu+ is to enable resource-efficient HPC based on special-purpose customized hardware. The nu+ manycore is a parameterizable regular mesh Network-on-Chip of configurable tile. Each nu+ tile has the same basic components, it provides a configurable GPU-like open-source softcore meant to be used as a configurable FPGA overlay.



Configurability:

- Number of cores
- Number of Threads
- Number of HW lane
- Number of Tiles
- Memory parameters
- Chace set-size and way
- ISA



Manycore solutions exploration

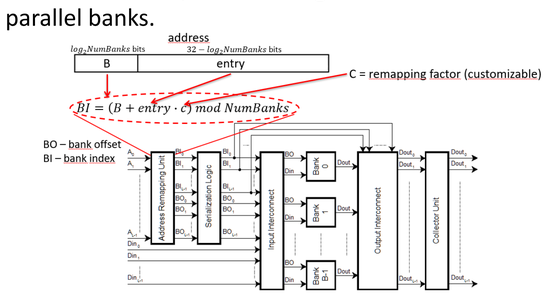
Hardware remapping

We explored a hardware remapping mechanism solution which enhances memory accesses in multi-banked memory systems. The proposed architecture enables a dynamic bank remapping hardware mechanism:

- allowing data to be redistributed across banks according to the specific access pattern,
- minimizing the number of conflicts and thereby improving the ultimate performance of the application.

Results

Results show that hardware bank remapping mechanism drastically reduces bank conflicts, even with a limited number of banks, while adding little resource overhead compared to a solution relying on a large number of parallel banks.



Lanes	Banks	Remapping factor				
		No Remap	1	2	4	8
4	16	262146	131072	262146	262146	262146
	32	262146	0	0	131072	262146
	64	262146	0	0	0	0
	128	262146	0	0	0	0
	256	131072	0	0	0	0
	512	0	0	0	0	0
	1024	0	0	0	0	0
8	16	183505	131073	183505	183505	183505
	32	183505	0	65536	131073	183505
	64	183505	0	0	0	65536
	128	183505	0	0	0	0
	256	131073	0	0	0	0
	512	65536	0	0	0	0
	1024	0	0	0	0	0

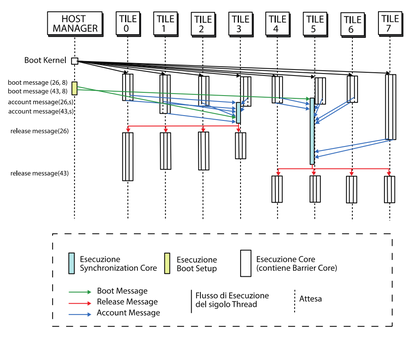
Lanes	Banks	Remapping factor				
		No Remap	1	2	4	8
16	16	109230	91756	109230	109230	109230
	32	109230	32768	65538	91756	109230
	64	109230	0	0	32768	65538
	128	109230	0	0	0	0
	256	91756	0	0	0	0
	512	65538	0	0	0	0
	1024	32768	0	0	0	0
32	16	61696	58256	61696	61696	61696
	32	59768	32769	45878	54615	59768
	64	59768	0	16384	32769	45878
	128	59768	0	0	0	16384
	256	54615	0	0	0	0
	512	45878	0	0	0	0
	1024	32769	0	0	0	0

Distributed synchronizer

The adoption of a distributed and NoC-based synchronization mechanism has been explored. At the heart of the proposed approach is a distributed synchronization master inspired by the directory-based coherence protocol.

Results

The centralized solution lacks scalability, even for small NoC configurations, and the final count is highly dependent on the position of the synchronization master. Our approach requires the same clock count for the smallest NoC, but results in improved scalability as the NoC size is increased.



	Barrier core	Boot setup	Centralized synchronization core (*per processing core*)			Distributed synchronization core (*per processing core*)		
			2×2	4×2	4×4	2×2	4×2	4×4
LUT	69	16	257	324	572	185	207	268
Flip-Flop	56	17	369	520	786	321	420	567

NoC size	Centralized	Distributed
2×2	26	26
4×2	164	135
4×4	263	216

Average clock cycles per thread

H2020 European project MANGO: exploring Manycore Architectures for Next-GeneratiOn HPC systems

chosen to participate in the Intel "*Hardware Accelerator Research Program*" (Nov 2016)

Currently involved in an internship in the architecture research group at Arm

Future Works:

- Explore scalable memory coherence solutions, such as hierarchical coherence.
- Explore distributed main memory solutions.
- Exploiting heterogeneity, in the current state the manycore is composed by equal tiles.
- Exploring non regular Network-on-Chip configurations.
- Automate the code partitioning among all tiles, allowing software users to be unaware of the underlying architecture.