Mirko Gagliardi

Tutor: Alessandro Cilardo

XXXI Cycle - III year presentation

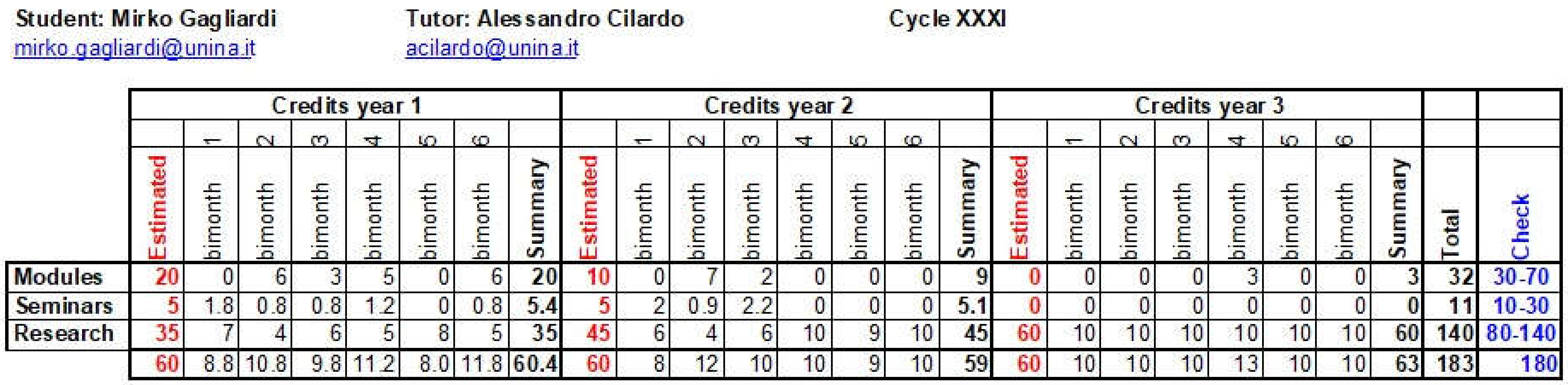
A Reconfigurable and Extensible Exploration Platform for Future Heterogenous Systems

Università degli Studi di Napoli Federico II

Background

- Master of Science:
  - In “Ingegneria Informatica” at University of Naples “Federico II”
- DIETI Group:
  - Seclab
- Type of Fellowship:
  - Doctoral Fellowship funded by CeRICT
- Collaboration:
  - With CeRICT in the context of the european project MANGO

Training Activities



Student: Mirko Gagliardi mirko.gagliardi@unina.it

Tutor: Alessandro Cilardo acilardo@unina.it

Cycle XXXI

	Credits year 1								Credits year 2								Credits year 3									
	Estimated	bimonth 1	bimonth 2	bimonth 3	bimonth 4	bimonth 5	bimonth 6	Summary	Estimated	bimonth 1	bimonth 2	bimonth 3	bimonth 4	bimonth 5	bimonth 6	Summary	Estimated	bimonth 1	bimonth 2	bimonth 3	bimonth 4	bimonth 5	bimonth 6	Summary	Total	Check
Modules	20	0	6	3	5	0	6	20	10	0	7	2	0	0	0	9	0	0	0	0	3	0	0	3	32	30-70
Seminars	5	1.8	0.8	0.8	1.2	0	0.8	5.4	5	2	0.9	2.2	0	0	0	5.1	0	0	0	0	0	0	0	0	11	10-30
Research	35	7	4	6	5	8	5	35	45	6	4	6	10	9	10	45	60	10	10	10	10	10	10	60	140	80-140
	60	8.8	10.8	9.8	11.2	8.0	11.8	60.4	60	8	12	10	10	9	10	59	60	10	10	10	13	10	10	63	183	180

Publications

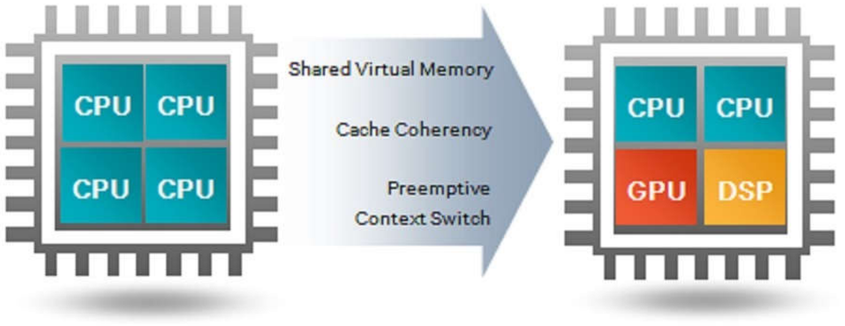
- Cilardo, A., Gagliardi, M. and Donnarumma, C., 2016, November. A Configurable Shared Scratchpad Memory for GPU-like Processors. In International Conference on P2P, Parallel, Grid, Cloud and Internet Computing.
- Cilardo, A., Gagliardi, M. and Passaretti, D., 2017, November. NoC-Based Thread Synchronization in a Custom Manycore System. In International Conference on P2P, Parallel, Grid, Cloud and Internet Computing.
- Gagliardi, M., Fusella, E. and Cilardo, A., 2018, July. Improving Deep Learning with a customizable GPU-like FPGA-based accelerator. In 2018 14th Conference on Ph. D. Research in Microelectronics and Electronics (PRIME).
- Zoni, D., Cremona, L., Cilardo, A., Gagliardi, M. and Fornaciari, W., 2018. PowerTap: All-digital power meter modeling for run-time power monitoring. Microprocessors and Microsystems.
- Cilardo, A., Gagliardi, M., Scotti, V. Lightweight hardware support for selective coherence in heterogeneous manycore accelerators. In 2019 Conference on Design, Automation and Test (DATE).

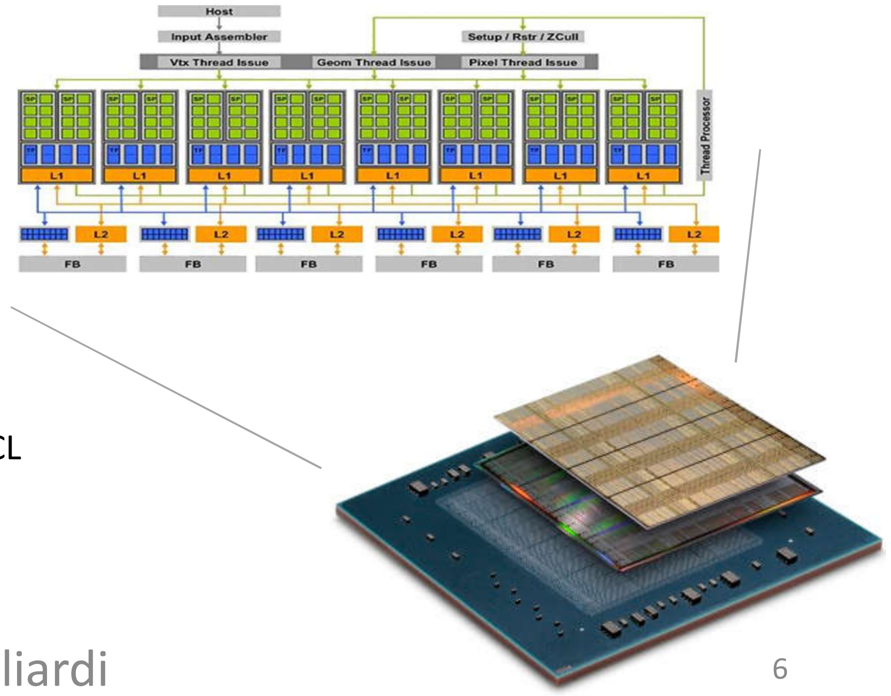
Experience abroad

- Internship in the architecture research group at **Arm Ltd**, Cambridge UK, from 05/06/2017 to 17/11/2017
  - **Topic**: Develop and maintain models and/or prototypes in support of Arm-based research enablement and in close collaboration with internal stakeholders. Main topic of interest: SoC design and FPGA Prototyping, Simulation and Modelling and IoT and Cloud Computing.
  - **Contribution**: ''Cortex-M-based SoC Design and Prototyping using Arm DesignStart'' whitepaper (https://developer.arm.com/research/research-enablement)

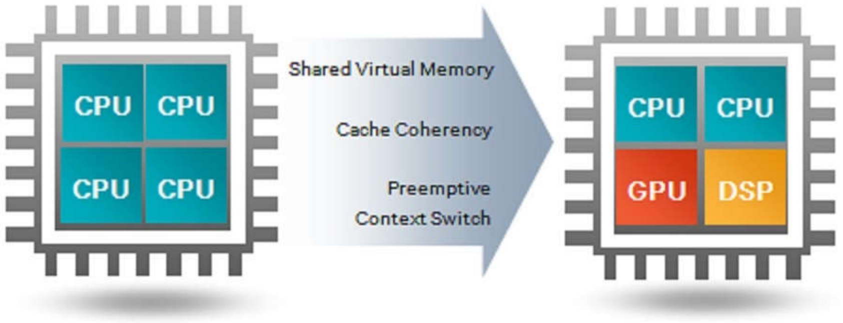
Heterogeneous Computing (1/2)

- **Problems:**
  - Power wall, memory wall, ILP wall
  - Disruption of the rising of clock frequency
- **Solutions:**
  - Multicore processors
  - Many Core system
  - Heterogeneous platforms, systems made of different computational units, with specialized accelerators that complement general purpose CPUs, such as DSPs, GPUs, co-processors, enabling significant benefits in terms of both power and performance
- **New paradigms:**
  - NoC-based interconnection infrastructure
  - GPGPU model and GPU-like cores
- **GPU-like paradigm:**
  - Match recent trends
  - also including FPGA and SoC manufacturers
  - Enable higher power-efficiency
  - Provide an effective answer to programmability issues
  - support for high-level languages and models, like OpenCL

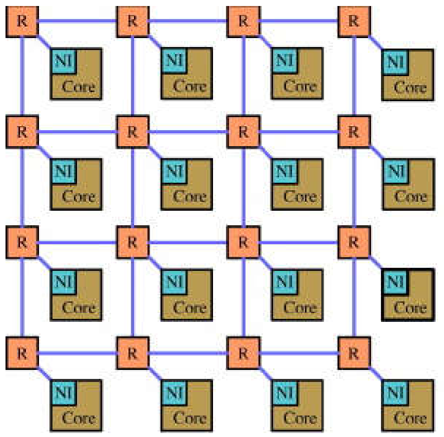




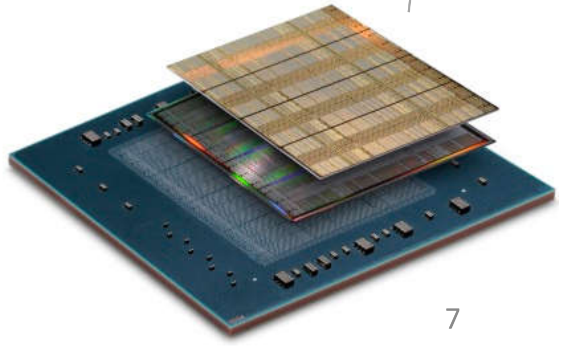
Heterogeneous Computing (2/2)



- **Programmability Issue:**
  - Programmers can only choose proprietary parallel programming languages (e.g. CUDA)
  - Existing programming toolkits have either been limited to a single product family
- **Shared Memory in heterogenous platforms:**
  - At the current state, parallel software developers are forced to manage main memory within their own application programs, and all data movement between host memory and device memory must be managed by the programmer
  - An ideal heterogenous computing model should allow programmers to create applications capable of using accelerators, with data movement between the host and accelerator implicit
  - Hardware coherence can offer a transparent view of the system to the final user



- **Open Challenges:**
  - Enabling a deeper customization of architectures to applications will eventually lead to computation efficiency
  - NoC-based architectures and GPGPU paradigms pose new challenges
  - Traditional solutions do not suit heterogeneity
  - Required further exploration



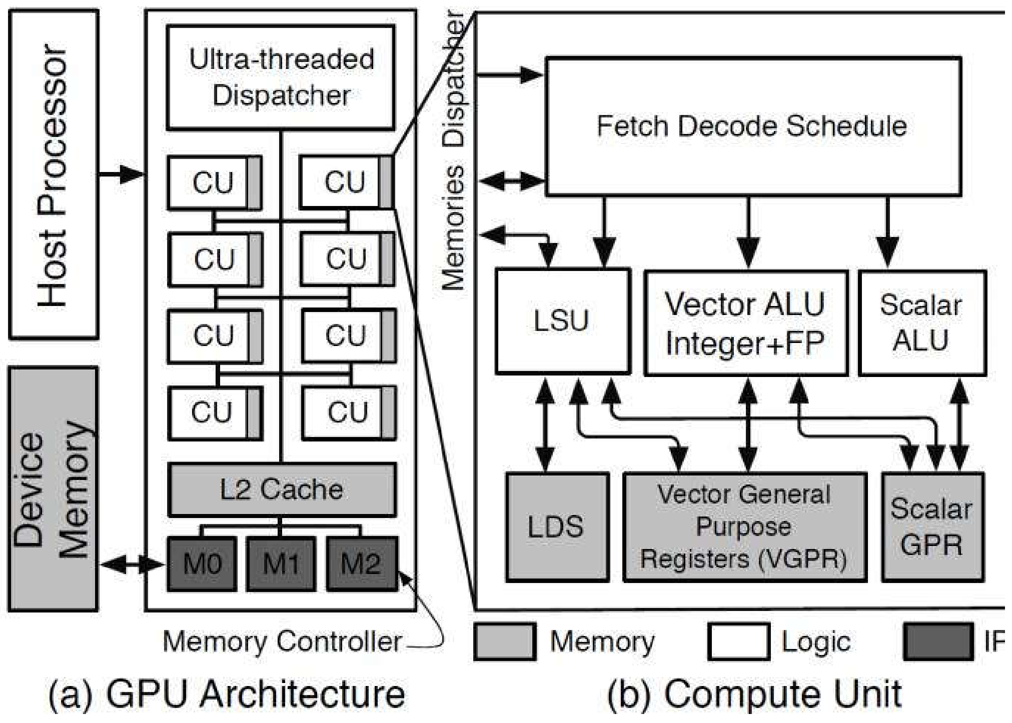
Related Works (1/2)

- Balasubramanian, Raghuraman, et al. "**Enabling GPGPU low-level hardware explorations with MIAOW: an open-source RTL implementation of a GPGPU.**" *ACM TACO*.

- Explores GPGPU and many-core paradigms starting from the canonical GPU organization.

- Investigates parallelism at the core level:
  - SIMD, VLIW, TLP

- Poses new types of research exploration simulation-based.

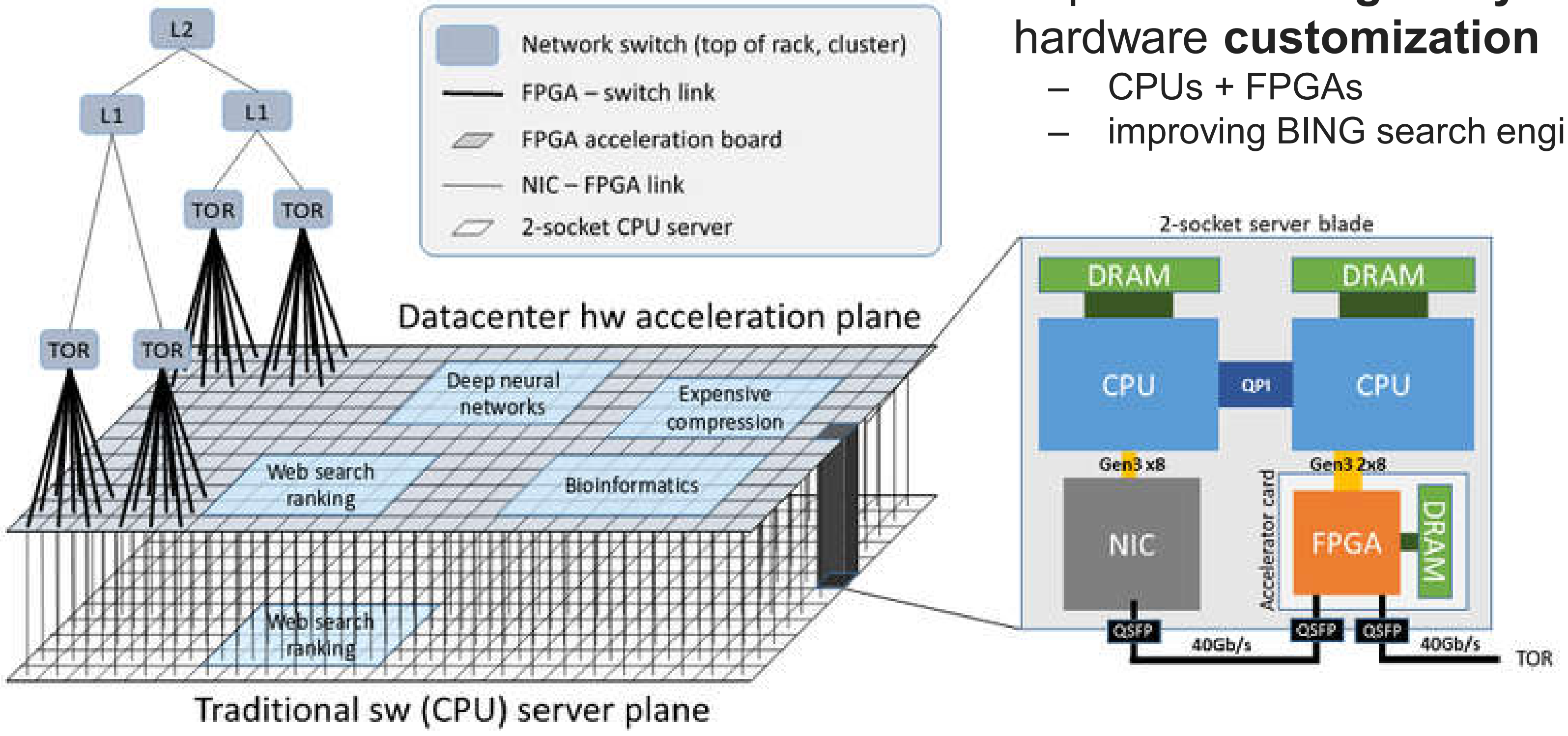


Related Works (2/3)

- Ovtcharov, Kalin, et al. "**Accelerating deep convolutional neural networks using specialized hardware.**" *Microsoft Research Whitepaper* 2.11 (2015).

- Microsoft Catapult Project

- Explores **heterogeneity** and hardware **customization**
  - CPUs + FPGAs
  - improving BING search engine



Mirko Gagliardi

Related Works (3/3)

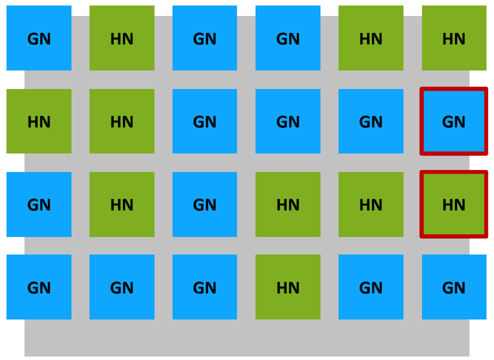
- Lee, Yunsup, et al. "**Exploring the tradeoffs between programmability and efficiency in data-parallel accelerators.**" *ACM SIGARCH Computer Architecture News*. Vol. 39. No. 3. ACM, 2011.
- Lee, Yunsup, et al. "**A 45nm 1.3 GHz 16.7 double-precision GFLOPS/W RISC-V processor with vector accelerators.**" *European Solid State Circuits Conference (ESSCIRC), ESSCIRC 2014-40th*. IEEE, 2014.
- Bush, Jeff, et al. "**Nyami: a synthesizable GPU architectural model for general-purpose and graphics-specific workloads.**" *Performance Analysis of Systems and Software (ISPASS), 2015 IEEE International Symposium on*. IEEE, 2015.
- Al-Dujaili, Abdullah, et al. "**Guppy: A GPU-like soft-core processor.**" *Field-Programmable Technology (FPT), 2012 International Conference on*. IEEE, 2012.
- Andryc, Kevin, Murtaza Merchant, and Russell Tessier. "**FlexGrip: A soft GPGPU for FPGAs.**" *Field-Programmable Technology (FPT), 2013 International Conference on*. IEEE, 2013.

MANGO project

- **MANGO:** exploring **M**anycore **A**rchitectures for **N**ext-**G**enerati**O**n HPC systems

- **MANGO project**
  - NoC-based manycore architecture
  - Heterogeneous system

- **Project Goals:**
  - Investigate the architectural implications of the emerging requirements of HPC applications.
  - Aiming at the definition of new-generation high-performance, power-efficient, deeply heterogeneous architecture.
  - Achieve extreme resource efficiency in future QoS-sensitive HPC through ambitious cross-boundary architecture exploration.



- **Approach**:
  - Explore new many-core architectures specifically targeted at HPC.
  - Involving interrelated mechanisms at various architectural levels:
    - heterogeneous computing cores
    - memory architecture
    - interconnect
    - runtime resource management
    - power monitoring and cooling

European Commission | Horizon 2020 European Union funding for Research & Innovation

Methodology

- Future many-cores require **exploration** over new infrastructures typical of this novel paradigm (such as NoCs and sparse directories).

- This work leverages on a **baseline heterogeneous platform** to evaluate novel solutions proposed.

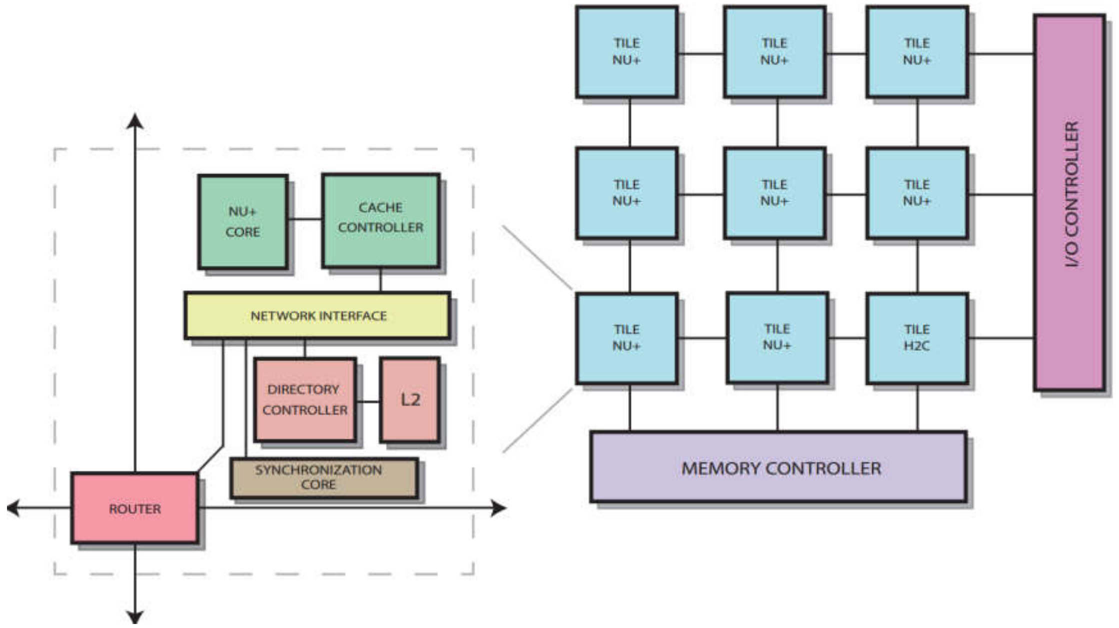
- This full-system enables us to better understand application-specific requirements through hardware customization, and also to evaluate the proposed solutions on a real system, running significant kernels extracted from typical workloads.

- We started with an **exploration phase** using significative application classes, such as deep learning-based algorithms, running on our baseline platform.
  - Identified bottlenecks and possible improvements, focusing on coherence- and NoC-related aspects.

- Each proposed solution has been **integrated** on both software and hardware levels in our baseline heterogeneous platform for validating and testing them in a real system.
  - Such approach also captures realistic results, implementation issues, and pitfalls not possible with typical simulation-based evaluation methods.

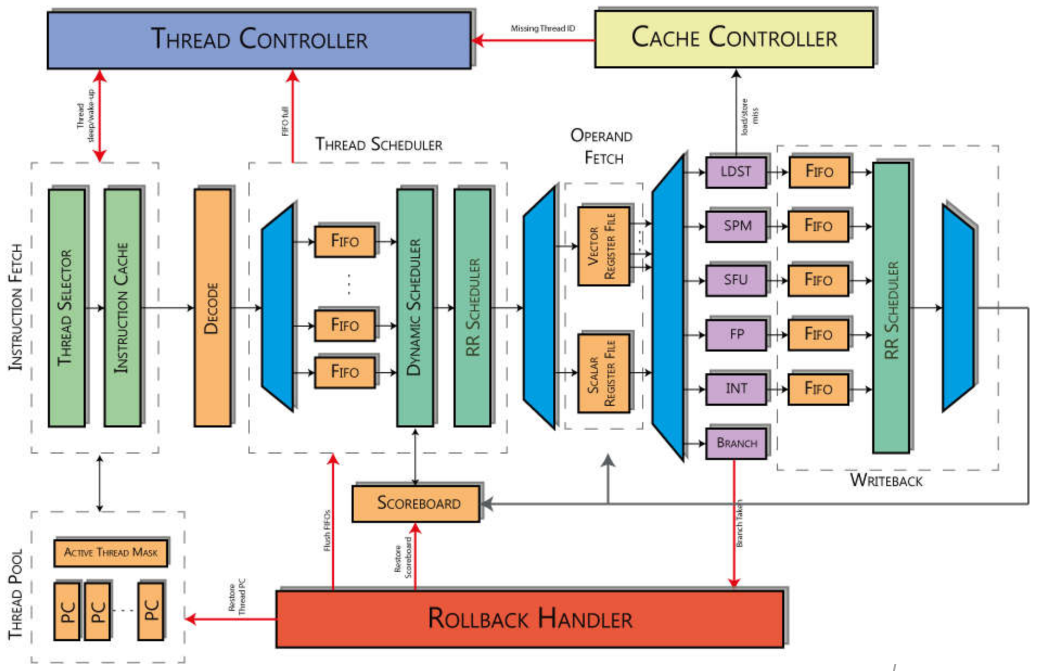
Nu+ Baseline Exploration Architecture

- Evaluation Platfrom:
  - Exploration phase and results are carried out from a baseline Heterogeneous system in line with modern trends
- High Parallelism
  - Multiple tiles interconnected through a mesh network
  - GPU-like core with a LLVM-based toolchain
- NoC-based
  - Hardware message passing support
  - Extendable Virtual Channel
  - Native support to directory-based coherence protocols.
- Coherence subsystem:
  - Hardware directory-based coherence support
  - Provides a coherence-independent interface to the accelerator



Nu+ Core Overview

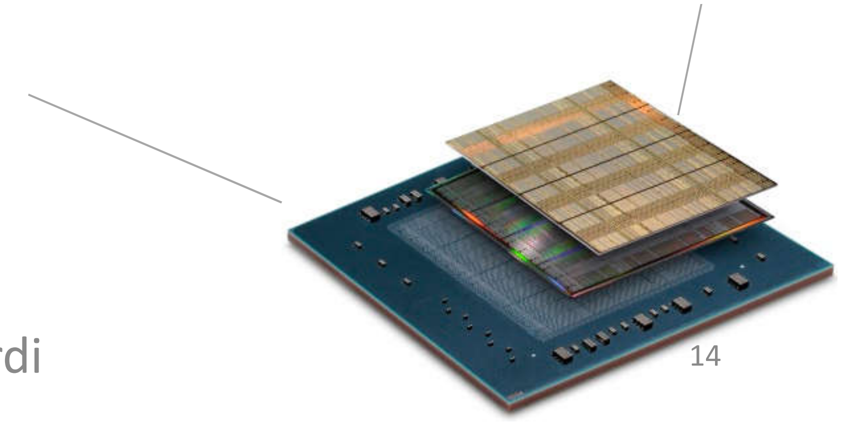
- RISC Core
  - In-order pipeline $\longrightarrow$ semplified control, thread switching to mask latencies
  - Parametrization $\longrightarrow$ thread number, hardware lane number, ...
  - Custom ISA



- Multithread support
  - Each thread with its own PC
  - N hardware lane shared among all threads
    - Resourse sharing

- High Parallelism
  - Scalar and Vectorial
  - Each thread has its own registers
  - 64 bit support (Integer and Floating Point)
    - N 32 bit registers or N/2 64 bit registers

- Variable latency operator support
  - SPM has a variable latency
  - Advance Writeback: automatic structural hazard detection

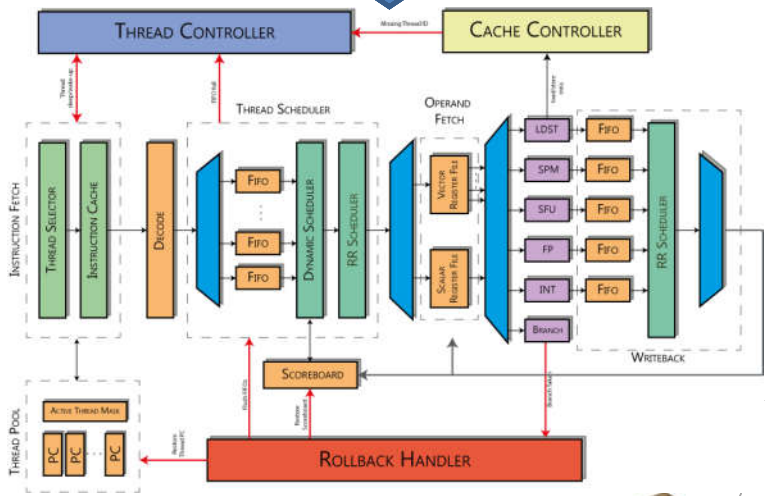


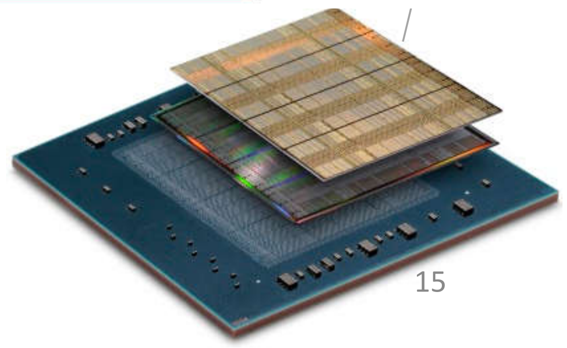
Exploration Phase (1/2)

- Explored how customization impacts performance.
- Using a deep learning kernel as a case study
- Evaluated different hardware configurations of our baseline platform.
- Identified major features, such as
  - Selective coherence
  - Parallel synchronization mechanism
- [1] investigates the adoption of different architectural features, i.e. **SIMD** paradigm, **multithreading**, and **non-coherent** on-chip memories for Deep Learning

$$y[n] = x[n] \cdot k[n] = \sum_k x[n] \cdot k[n-k]$$

```
for(row = 0; row < M; row++)
 for(col = 0; col < M; col++)
   for(krow = 0; krow < K; krow++)
     for(kcol = 0; kcol < K; kcol++)
       y[row][col] += k[krow][kcol] *
              x[row + krow][col + kcol];
```

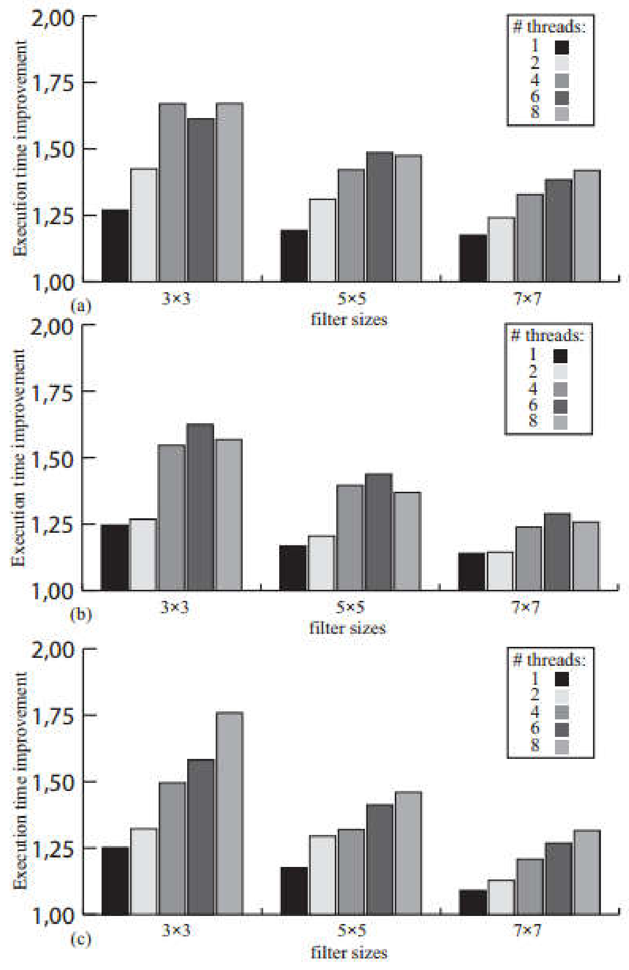




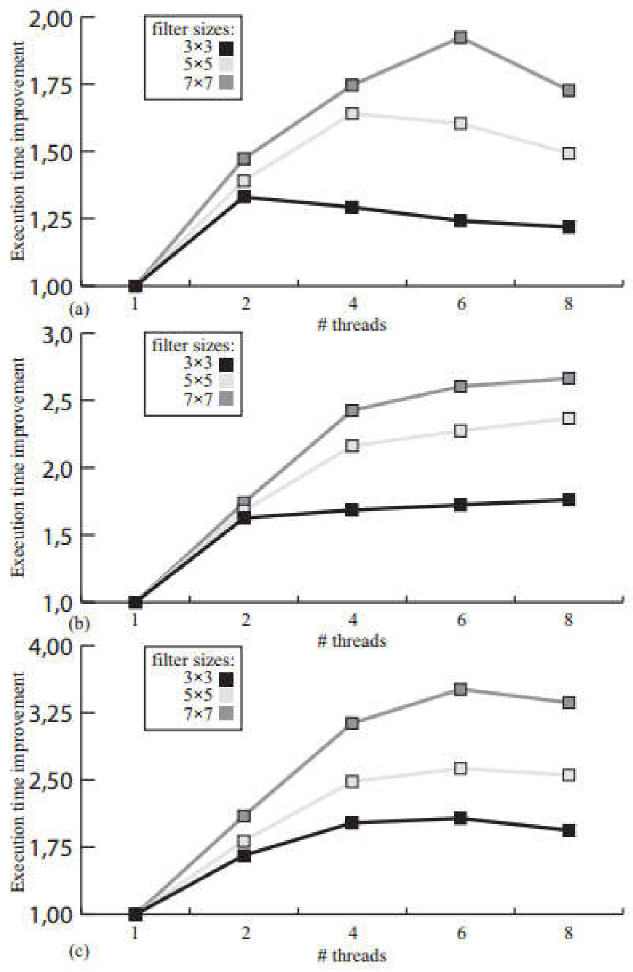
[1] Gagliardi, Mirko, Edoardo Fusella, and Alessandro Cilardo. "Improving Deep Learning with a customizable GPU-like FPGA-based accelerator." 2018 14th Conference on Ph. D. Research in Microelectronics and Electronics (PRIME). IEEE, 2018.

Exploration Phase (2/2)

- The speedup achieved using scratchpad memory

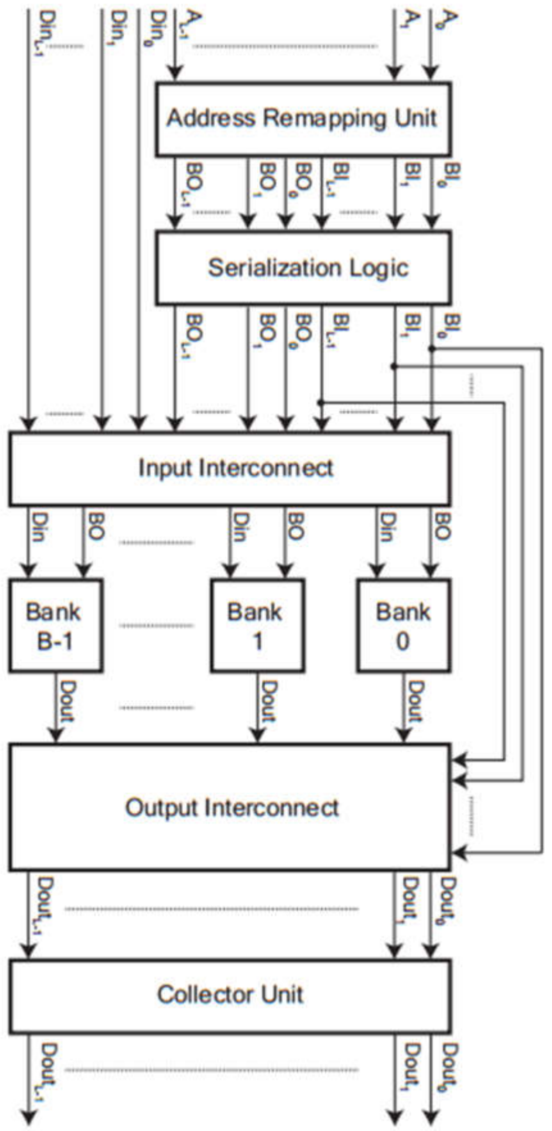


- Speedup over single-thread implementation when varying the number of threads and SIMD support



Customizable Shared Scratchpad Memory

- Selective coherence can increase performance.
- Proposed a novel scratchpad memory, based on the **NVIDIA model**, with a configurable bank remapping system to reduce bank conflicts
- The experimental results showed the performance implications with different configurations
- and demonstrated the benefits of both using:
  - A **customizable** hardware bank remapping function
  - A **non-coherent** memories for some kind of algorithms.

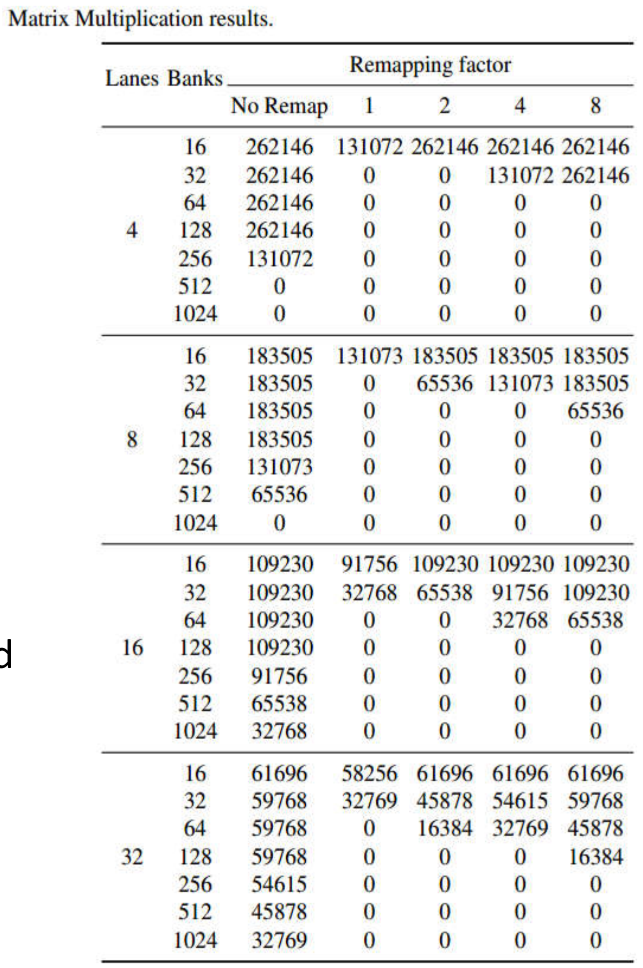


Results: application-driven customization

- **Hardware application-driven parameterization**
  - Each application has an ideal hardware configuration
  - E.g. same kernels do not need cache coherence
    - In this case a coherent cache is not needed
    - A non coherent scratchpad memory has less hardware overhead

- [2] explores hardware ad hoc **customization** benefits. We describe how a configurable **non-coherent** memory affects performance and how different parallel kernels require different configurations

- **Main findings:**
  - A configurabile non coherent scratchpad memory presented
  - Parameterization highly affects performance
  - Hardware optimization (Bank Remapping)
    - A typically software optimization has been realized in hardware
    - Faster than software version
    - Minimum overhead and high impact

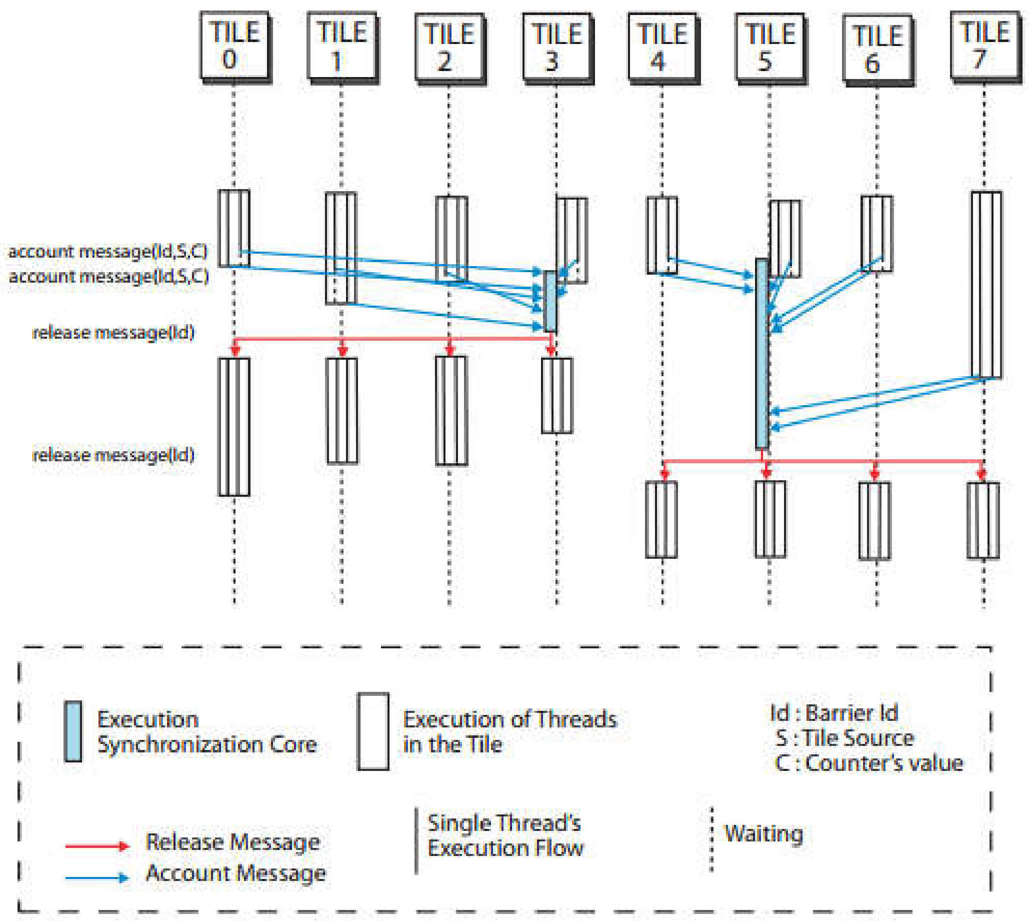

Matrix Multiplication results.

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
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

[2] Cilardo Alessandro, Gagliardi Mirko, Donnarumma Ciro, "***A Configurable Shared Scratchpad Memory for GPU-like Processors***." International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. Springer International Publishing, 2016.

Distributed Thread Synchronization

- The adoption of a distributed and **NoC-based** synchronization mechanism has been explored.
- At the heart of the proposed approach is a **customizable** and distributed synchronization master inspired by the directory-based coherence protocol.
- The proposed architecture can support multiple synchronizations for different application kernels running concurrently.
  - Essential in parallel architecture and **non-coherent** scenarios
  - Hardware messages, no main memory access



Results: distributed thread synchronization

- [3] explores the proposed **distributed** synchronization mechanism
- The first experiment runs a **single** barrier which involves all processing cores instantiated in the manycore, comparing the approach based on a centralized synchronization master with our distributed solution.

NoC Size	Centralized	Distributed
2x2	34	34
4x2	47	45
4x4	79	81
8x4	150	153

- The proposed architecture supports **multiple** barrier synchronizations being executed simultaneously on different subsets of cores.
  - The experiment synchronizes all the subsets, running the maximum number of supported barriers.

NoC Size	Centralized	Distributed
2x2	34	34
4x2	35	32
4x4	37.8	30
8x4	46.1	30

[3] Cilardo, Alessandro, Mirko Gagliardi, and Daniele Passaretti. "NoC-Based Thread Synchronization in a Custom Manycore System." International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. Springer, Cham, 2017.

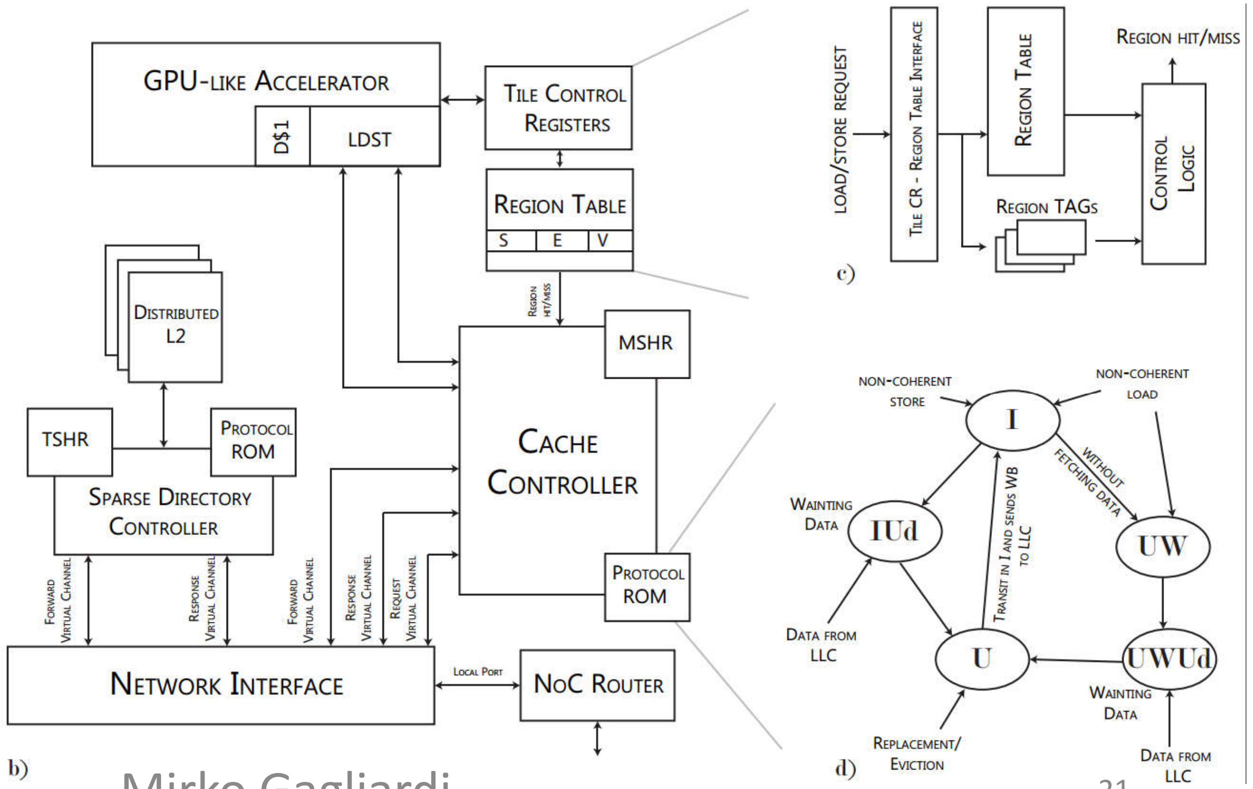
Selective coherence in many-cores (1/2)

- Extending the **non-coherent** scratchpad paradigm to a directory-based coherence system

- Hardware coherence helps programmers, simplifying the programming model:
  - Overhead due non required coherence maintenance (non-shared memory blocks)

- Proposed a coherence subsystem with **non-coherent** block support

- Extention of the baseline coherence system:
  - Cache Controller
  - Non-coherent regions table
  - MSI protocol extended



Mirko Gagliardi

Selective coherence in many-cores (2/2)

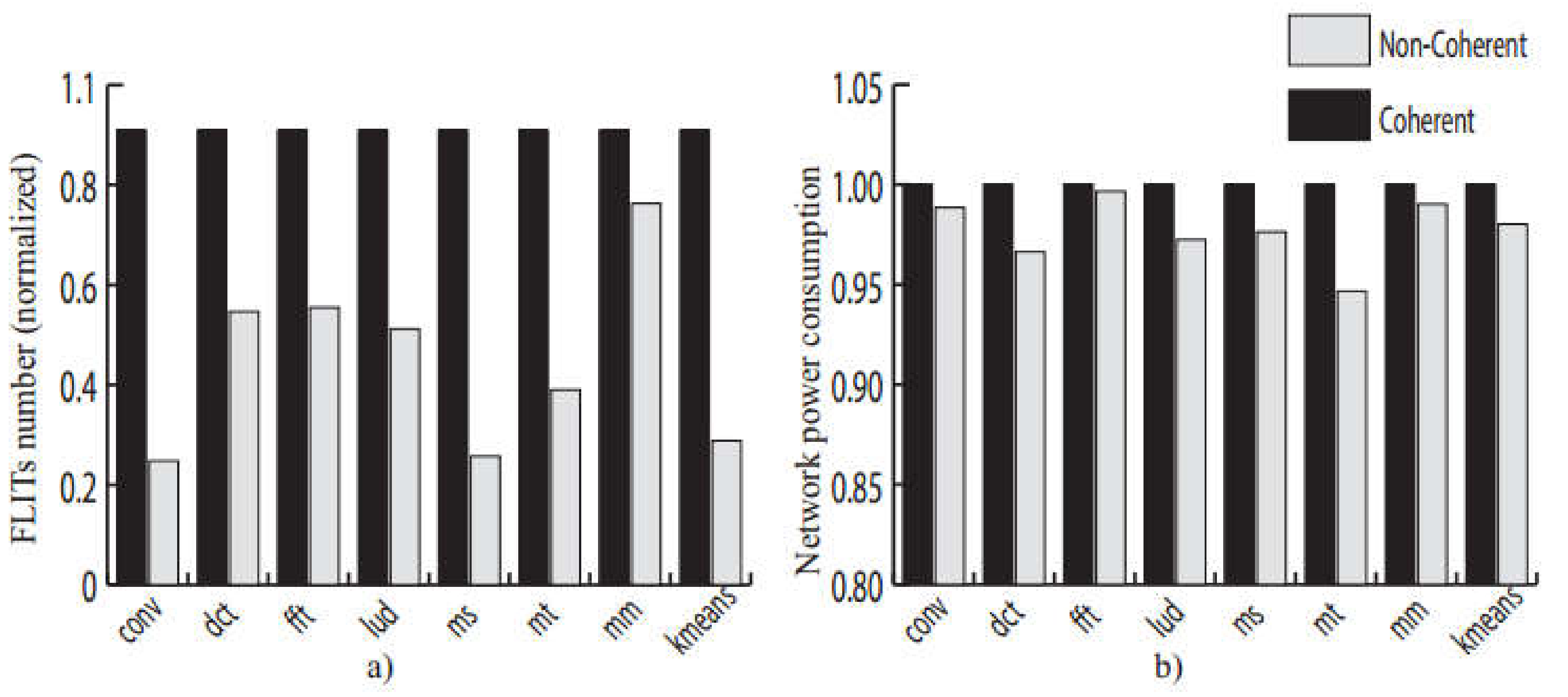
- [4] proposed an advanced coherence subsystem with a selective coherence maintenance system which allows coherence to be deactivated for blocks that do not requiring it.

- Experimental results showed that the use of a hybrid coherent and non-coherent architectural mechanism along with an extended coherence protocol can enhance performance.

[4] Alessandro Cilardo, Mirko Gagliardi, Vincenzo Scotti. "Lightweight hardware support for selective coherence in heterogeneous manycore accelerators" DATE 2019 conference, ongoing peer review.

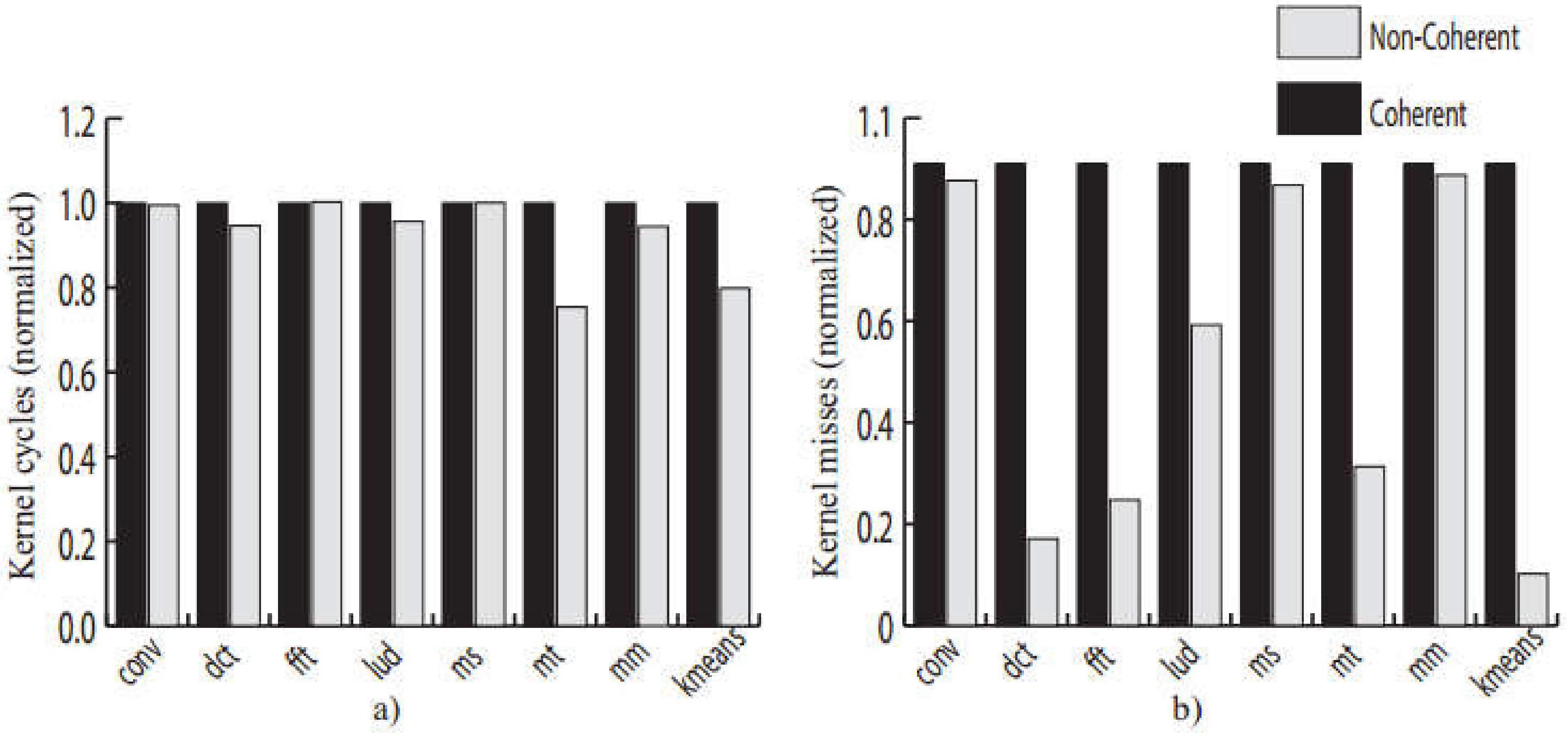
Results: selective coherence support (1/2)

- Selective **non-coherece** benefits the overall number of messages flowing over the network-on-chip (Figure a)
- Hence, such a solution impacts on the overall dynamic power consumption of the networking instractructure (Figure b)



Results: selective coherence support (2/2)

- Number of cycles for each kernel in coherent and non-coherent configurations (Figure a)
- This solution greatly impacts the overall data miss count. Figure b shows the miss count of the whole many-core featuring 8 accelerators



Conclusions

- This dissertation explored future heterogeneous architecture features and requirements, exploiting customizability, and running kernel extracted from modern HPC-based workloads.

- The contributions of this dissertation are as follows:
  - First, this work explored how customization impacts performance, using a deep learning kernel as a case study for evaluating different hardware configurations of our baseline platform, as well as identifying major features, such as selective coherence, which might benefit many-core systems.
  - Second, we presented a non-coherent scratchpad memory with a configurable bank remapping system to reduce bank conflicts.
  - Next, we demonstrated how a distributed synchronization master better suits many-cores than standard centralized solutions.
  - Finally, we proposed an advanced coherence subsystem, based on the sparse directory approach, with a selective coherence maintenance system which allows coherence to be deactivated for blocks that do not requiring it.

- All the results of this dissertation carried out on a custom heterogeneous system, developed in the framework of the MANGO H2020 project.
  - A real system captures realistic results, implementation issues, and pitfalls not possible with typical simulation-based evaluation methods
  - The open-source exploration platform comes along with the presented results as part of the contributions of this dissertation

Future Works

- All the proposed solutions rely on software builtins and programmer's explicit commands.

- Future work could aim to ease the final user's duty:
  - making these mechanisms transparent to the final user, blending hardware monitors and compiler ad-hoc extensions.
  - It might give a standard programmer-friendly model, automatizing all the custom or target-specific features.

- In the case of the proposed coherence subsystem, such a mechanism could allow the compiler to automatically map specific memory blocks into the non-coherent memory space
  - As a result the programmer would be totally unaware of the non-coherent mechanisms.

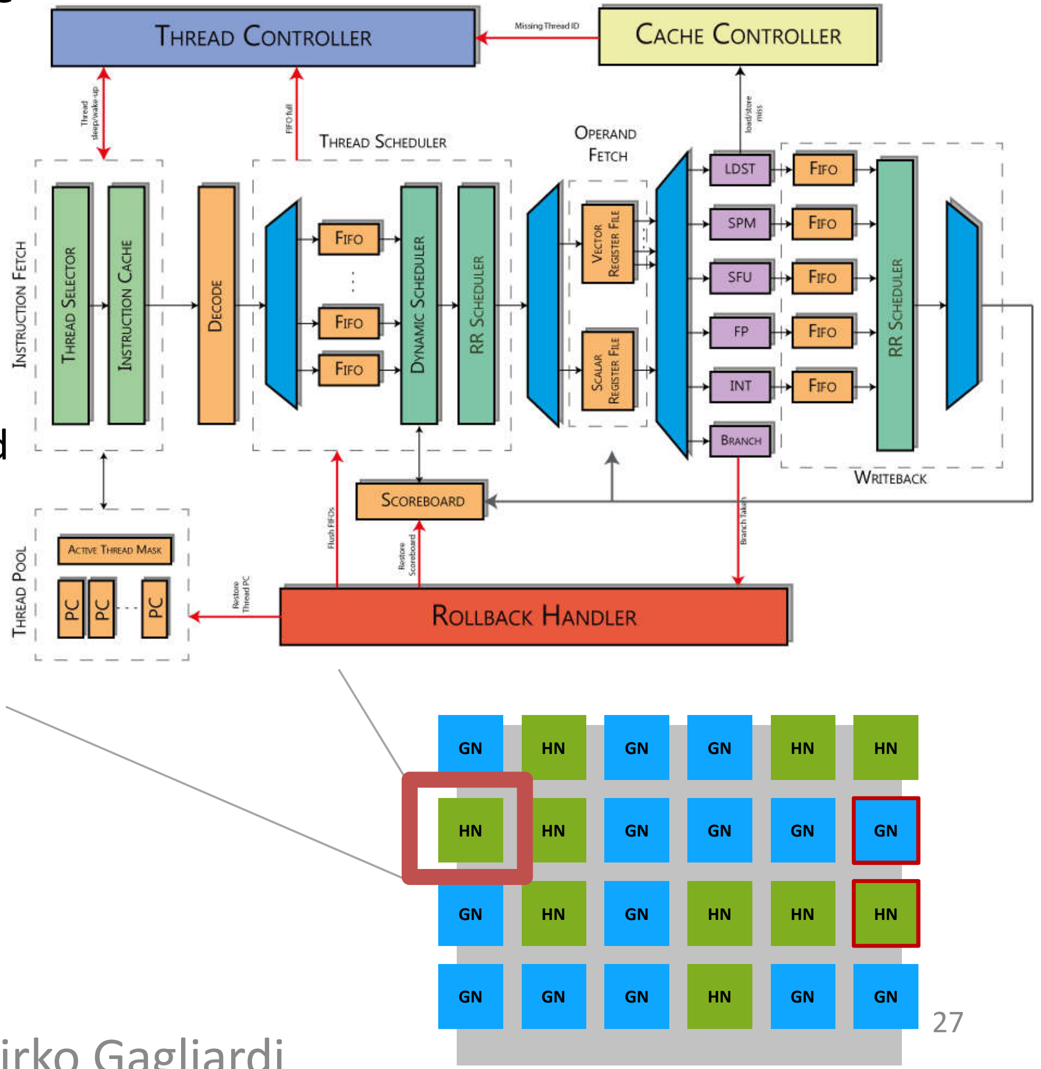
- Finally, further investigation in the context of heterogeneous systems is required, to exploit the full potential of modern mechanisms, such as selective coherence, and application-driven hardware customizability

Impact

- During the MANGO project, the baseline GPU-like core, enhanced with the selective coherence subsystem and the customizable scratchpad memory, has been widely tested and validated by successfully running complex application, provided by the project's partners, in the context of biomedical engineering.

- Now, the nu+ core has been integrated into the MANGO infrastructure

- Being part of a sophisticated heterogeneous system, running on a FPGA-based cluster



Thread Controller
Missing Thread ID
Cache Controller
Thread sleep/wake-up
FIFO full
load/store miss
Instruction Fetch
Thread Selector
Instruction Cache
Decode
Thread Scheduler
FIFO
FIFO
FIFO
Dynamic Scheduler
RR Scheduler
Operand Fetch
Vector Register File
Scalar Register File
LDST
SPM
SFU
FP
INT
Branch
FIFO
FIFO
FIFO
FIFO
FIFO
RR Scheduler
Writeback
Scoreboard
Thread Pool
Active Thread Mask
PC
PC
PC
Flush FIFOs
Restore Scoreboard
Restore Thread PC
Branch Taken
Rollback Handler
GN HN GN GN HN HN
HN HN GN GN GN GN
GN HN GN HN HN HN
GN GN GN HN GN GN