UNIVERSITÀ DEGLI STUDI DI ROMA "SAPIENZA"
DEPARTMENT OF INFORMATION ENGINEERING, ELECTRONICS AND TELECOMMUNICATIONS (DIET)

Doctorate
Information and Communications Technologies
Ciclo XXX

CHARACTERIZATION AND OPTIMIZATION OF NETWORK TRAFFIC IN CORTICAL SIMULATION

Andrea Biagioni

Tutor:
Pier Stanislao Paolucci

Co-Tutor:
Maria Gabriella Di Benedetto

Rome, 2017

PhD Advisory Board

Elio Di Claudio
Sapienza University of Rome
Rome, Italy

Vincenzo Eramo
Sapienza University of Rome
Rome, Italy

Aurelio Uncini
Sapienza University of Rome
Rome, Italy

Contents

List of Tables

List of Figures

1 Introduction

1.1 The Exascale value

Some of the key challenges, faced not just by individual companies but by civilisation as a whole, will be enabled by huge computing power. Exascale computing is the label we use for the next 50- to 100-fold increase in speed over the fastest supercomputers in broad use today, that is at least a billion billion operations per second[1].

The benefits of Exascale computing could potentially impact every person [1]. For example, computer models can be adopted to reduce pollution caused by burning fossil fuels, increasing by 25-50% the efficiency of combustion systems in engines and gas turbines, and lowering the emissions. The use of alternative energy sources — cost-effective solar energy, more efficient wind turbines, improved management of the electric power grid — is another challenge that benefits from computationally intensive optimization methods and fast computers.

Computers play a crucial role in the research for new materials, created with complex calculations that simulate their behaviour in nature, and using massive databases of known compounds to identify good combinations. Deep learning techniques and classical simulations are in use today in this field; Exascale computing can enable faster and more complex design.

The advantages of Exascale computing will flow from classical simulations but also from large-scale data analysis, deep machine learning, and often the integration of the three methods. Examples of the latter are healthcare (precision medicine) and biology. The understanding of the molecular basis of key protein interactions and the automation of the analysis of information from millions of patient records to determine optimal treatment strategies will accelerate medicine research. Decision on the right treatments by modeling drug responses

[1] The prefix *Exa-* denotes a power 10^{18}.

requires the search of one trillion drug combinations.

Everyone is concerned about climate change and climate modeling. The computational challenge for doing oceanic clouds, ice and topography are all tremendously important. Today we need at least two orders of magnitude improvement for weather prediction models to foresee weather events such as hurricanes by using much higher spatial resolution, incorporating more physics, and assimilating more observational data.

Improved gathering and analysis of numerous types of data from databases, sensors and simulation results, and conducting thousands of potential scenarios can mitigate health hazards, reduce crime, and improve the quality of life in cities by optimizing infrastructure — *e.g.* transportation, energy, housing.

In parallel to the aboved discussed implications for the society, fundamental scientific questions in fields such as high-energy physics can be addressed, and can benefit of increased computing — *i.e.* Lattice Quantum Chromo Dynamics simulation (LQCD), exploration of dark-energy and dark-matter, coupled N-body/hydrodynamics/radiation transport codes for structure formation in the early Universe and controlled fusion reactions.

1.2 The brain simulation challenge

There are several additional important applications that Exascale computing will advance, but they will require even more computing power to be accomplished, such as the reverse engineering of the human brain to understand complex neural systems.

The Human Brain Project (HBP) [2] Flagship was launched by the European Commission's Future and Emerging Technologies (FET) scheme in October 2013, and is scheduled to run for ten years. The HBP has the following main objectives: (i) create and operate a European scientific Research Infrastructure for brain research, cognitive neuroscience, and other brain-inspired sciences; (ii) gather, organise and disseminate data describing the brain and its diseases; (iii) simulate the brain; (iv) build multi-scale scaffold theory and models for the brain; (v) develop brain-inspired computing, data analytics and robotics.

Brain simulation can reduce the need for animal experiments, study diseases in unprecedented *in-silico* experiments, and improve the validation of data and experiments with computational validation. Simulations allow to reconstruct and simulate detailed multi-level models of the brain, displaying emergent structures and behaviours. Models can be simulated and reconstructed at different levels of description, from abstract to highly detailed molecular and cellular models. Because of the incredible complexity of the human brain, analyses and simulations of the brain require massive computational power, and hence incredibly powerful computers, as well as immense storage capacity. Key areas of research

include novel visualization methods, innovative approaches for dynamic resource management on supercomputers and new methods for brain simulation, focusing in particular on linking extreme scale data-processing challenges to the exploitation of scalable compute resources and on using accelerator technologies to address computational challenges.

1.3 General challenges for Exascale systems

The emerging Exascale computing architecture will not be simply 1000x today's petascale architecture. All the proposed Exascale computer systems designs share some of the following challenges [3]:

- System power is the primary constraint for the Exascale system: simply scaling up from today's requirements for a petaflop computer, the exaflop computer in 2020 would require 200 MW, which is untenable. The target is 20-40 MW in 2020 for 1 exaflop.
- Memory bandwidth and capacity are not keeping pace with the increase in flops: technology trends against a constant or increasing memory per core. Although the memory per flop may be acceptable to applications, memory per processor will fall dramatically, thus rendering some of the current scaling approaches useless.
- Clock frequencies are expected to decrease to conserve power; as a result, the number of processing units on a single chip will have to increase – this means the Exascale architecture will likely be high-concurrency – billion-way concurrency is expected.
- Cost of data movement (see Figure 1.1), both in energy consumed and in performance, is not expected to improve as much as that of floating point operations, thus algorithms need to minimize data movement, not flops.
- The I/O system at all levels – chip to memory, memory to I/O node, I/O node to disk – will be much harder to manage, as I/O bandwidth is unlikely to keep pace with machine speed.
- Reliability and resiliency will be critical at the scale of billion-way concurrency: "silent errors", caused by the failure of components and manufacturing variability, will more drastically affect the results of computations on Exascale computers than today's petascale computers.

- Programming model will be necessary: heroic compilers will not be able to hide the level of concurrency from applications — a hierarchical approach to parallelism is needed.

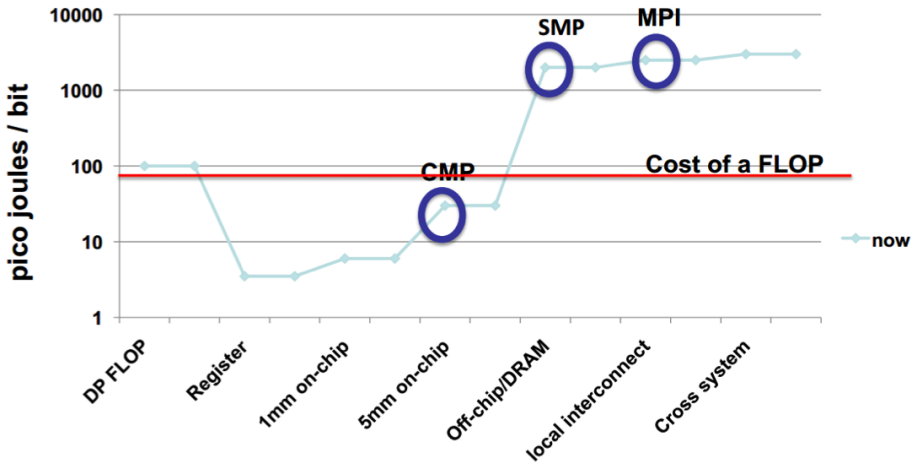

Figure 1.1: Energy cost of data movement [4]

1.3.1 Focusing on interconnect and power-efficiency: a co-design approach

Considering the above presented list of challenges that the Exascale systems have to face in the next future, a deeper attention will be given in this thesis to the interconnect and the power consumption.

The data movement challenge involves the whole hierarchical organization of components in HPC systems — *i.e.* registers, cache, memory, disks. Running scientific applications needs to provide the most effective methods of data transport among the levels of hierarchy. On current petaflop systems, memory access at all the levels is the limiting factor in almost all applications. This drives the requirement for an interconnect achieving adequate rates of data transfer, or throughput, and reducing time delays, or latency, between the levels [5].

Power consumption is identified as the largest hardware research challenge. The annual power cost to operate the system would be above 2.5 B$ per year for an Exascale system using current technology. The research for alternative power-efficient computing device is mandatory for the procurement of the future HPC systems.

Finally, a preliminary approach will be offered to the critical process of co-design. Co-desing is defined as the simultaneos design of both hardware and software, to implement a

desired function. This process both integrates all components of the Exascale initiative and illuminates the trade-offs that must be made within this complex undertaking.

1.4 HPC systems in the world

Building supercomputers capable to reach a peak-performance of the order of the exaflop, *i.e.* 10^{18} floating-point (FP) operations per second, is a clear priority worldwide (see Figure 1.2 and Figure 1.3), with programs in Europe [6], USA [7], China [8] and Japan [9].

The TOP500 [10] table shows the 500 most powerful commercially available computer systems in the world ranked by their performance on the LINPACK [11] Benchmark — a measure of a system's floating point computing power obtained solving a dense n by n system of linear equations $Ax = b$. At the moment of writing, China holds the top two spots for fastest computers in the world, Switzerland (the only European peer in the top ten) holds the third, Japan occupies the seventh and eigth positions with the U.S. in the fourth, fifth and sixth spots. An overview of the main systems in the world according to this ranking allows to sketch the state of the art — focused on computing and interconnect — and to understand the role played by the major actors in the HPC scenario.

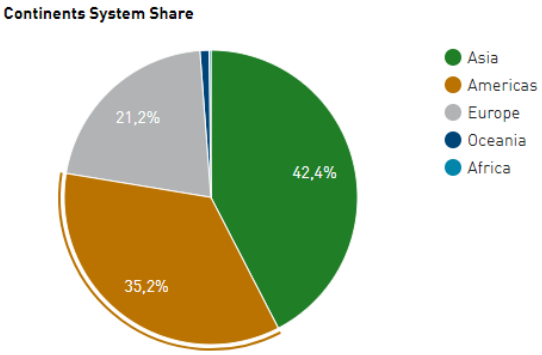

Figure 1.2: Continents system share.

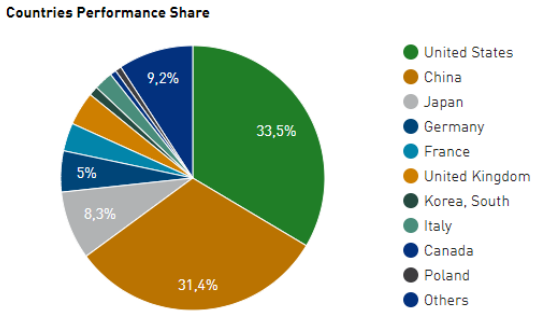

Figure 1.3: Countries system share

Sunway TaihuLight [12] at the National Supercomputing Center in Wuxi and Tianhe-2 [13] at the National Super Computer Center in Guangzhou lead the chinese scene. The former adopts a custom solution for both the computing system and the interconnect, a switched fabric technology similar to the one by Infiniband [14]; the latter has developed a custom network (ThExpress-2 [15]) with k-nominal or k-ary tree topology, but exploiting standard Intel processors.

Piz Daint [16] at the Swiss National Supercomputing Centre (CSCS) in Europe is a massively parallel multiprocessor supercomputer based on Cray XC50 [17]. It consists of In-

tel Xeon processors coupled with NVIDIA Tesla P100 accelerators, connected together by Cray's proprietary Aries interconnect shaping a dragonfly topology [18].

Cray is the provider of two out of three major HPC systems from United States of America, Titan [19] at the Oak Ridge National Laboratory and Cori [20] at the Berkeley National Energy Research Scientific Computer Center (NERSC). Titan employs AMD Opteron CPUs in conjunction with NVIDIA Tesla K20x GPUs and the network is based on Cray's Gemini interconnect shaping a 3D-Torus topology. Cori is based on Cray XC40 consisting of Intel Xeon PHI processors interconnected by Aries. The third american HPC system is Sequoia [21] at the Lawrence Livermore National Laboratory (LLNL) in California. Sequoia is a petascale BlueGene/Q supercomputer constructed by IBM. Both the computing chip [22] (Power BQC 16C) and network system [23] shaping a 5D-Torus topology are proprietary.

The Japan scene is led by two systems. The Oakforest-PACS system is located in the Information Technology Center at the University of Tokyo's Kashiwa Campus, but everything is carried out jointly by the University of Tokyo and the University of Tsukuba. The system is made up of computational nodes using Intel Xeon Phi high performance processors with Knights Landing architecture that uses many-core processor technology. The nodes are connected by Intel Omni-Path Architecture in a Full Bisectional bandwidth Fat Tree Topology. Finally, the K computer [24] at the RIKEN Advanced Institute for Computational Science (AICS) in Kobe, Japan. The processor is the Sparc64 by Fujitsu and the node are interconnected by custom interconnect — Tofu [25] — in a 6D-Torus topology.

The description of the top eight HPC system of the top500 list reflects the general HPC status for what regards processors — Intel totally dominates the scene as depicted in Figure 1.4 — and accelerators — NVIDIA is the leader and Intel is the major competitor, Figure 1.5.

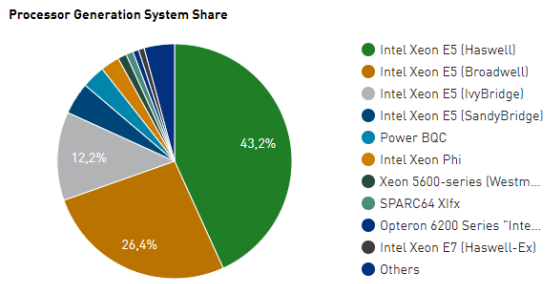

Figure 1.4: Processors system share.

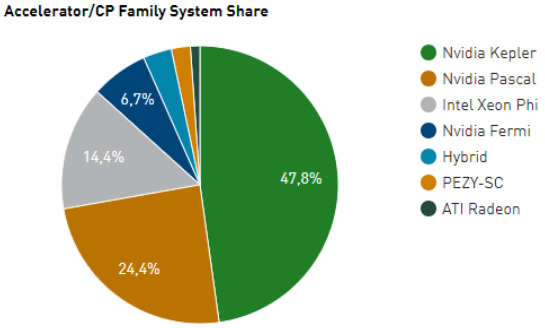

Figure 1.5: Accelerators system share

On the contrary, the interconnect market is leaded by off-the-shelf components — *i.e.* Ethernet and Infiniband, Figure 1.6 — but the design and implementation of proprietary and

custom networks seems to be a valuable solution to achieve the highest performance, as showed in Figure 1.7.

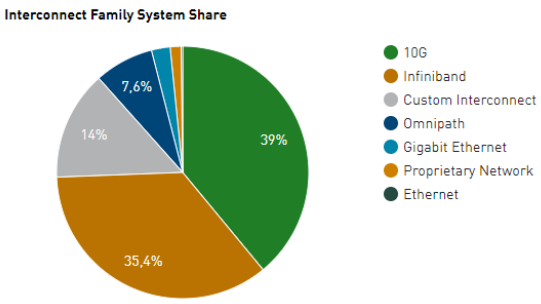

Figure 1.6: Interconnects system share.

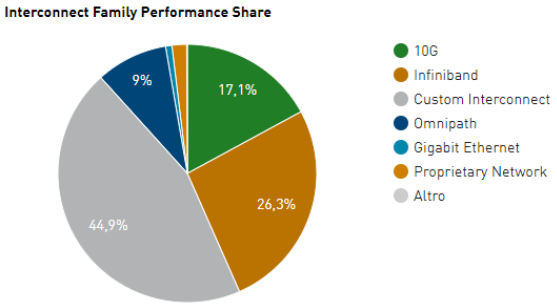

Figure 1.7: Interconnects performance share

1.5 What next in Exascale HPC?

As stated before, one of the major issue in the race towards the Exascale is minimizing the requirement of system power. Nowadays, a valuable solution is a hybrid CPU/GPU computing system. The green500 list, ranking computers from the TOP500 list of supercomputers in terms of energy efficiency, is leaded by hybrid systems composed by Intel Xeon processor and NVIDIA Tesla P100 GPU as accelerator.

Two competitors are trying to gain positions in the market exploiting their energy-efficiency features: FPGA and ARM processor. A comparison of energy-efficiency in terms of floating point operation per watt is listed in Figure 1.8. Altera and Xilinx FPGAs and the ARM Mali GPU show a performance/watt ratio doubling the results achieved by the most powerful NVIDIA GPUs available in the market — the Pascal P100, not reported in the list, that provides ~ 35 Gflops/W.

FPGAs offer the combination of software-like flexibility and hardware-like performance, providing hard real-time computations, parallelism, and high use of I/O pin count, and including protocol independent high-speed serial links. All these features allow the FPGAs to be connected to almost every application specific circuit. FPGAs provide IPs to be compliant with many industrial standards — *e.g.* HBM, PCIe, DDR3, Ethernet — and DSP cells for computing power, and they are gaining more and more attention in the research scenario as demonstrated by the numbers of publication (2905) tagged with the keywords “FPGA” in the 2015 [27]. FPGAs are used in a great variety of application fields, among others: communication, networks, neuro-computing, processor design, and data acquisition. Finally, these devices integrate processors in a single die — System On Chip (SoC) — providing higher

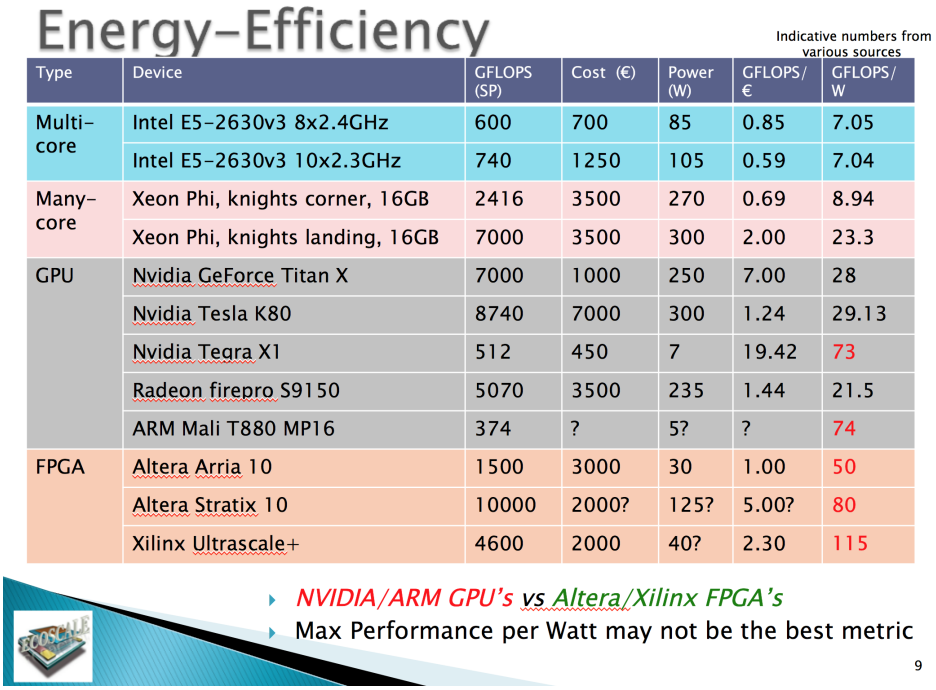

Figure 1.8: An energy-efficiency comparison among CPUs, GPUs and FPGAs [26].

integration between the processor and the programmable logic of the FPGA.

The latest SoC-FPGA generations are equipped with $4 \div 8$ ARM cores. ARM is a provider of Hardware/Software Intellectual Properties, developing the architecture and licensing it to other companies, who design their own products, for instance Android and Apple phones and tablets, RaspberryPI, Arduino, set-top box and multimedia. ARM is the industry leader in power-efficient processor design. ARM processors consume about 2 to 3 times less electric energy for a given amount of computation relative to Intel-based processors, and are widely used in embedded consumer electronics, including smartphones and tablets. As a result, many research and industry programs perceive ARM-based microservers as a potential successor of x86 and POWER-based servers in hyperscale datacenters and supercomputers [28, 29, 30, 31]. Indeed, the premonition is that using low-power processors could pave the way towards Exascale due to its tight power budget.

1.6 Structure of the thesis

In Chapter 2, the ExaNeSt project is described; the projects aims at providing a hierarchical interconnect for the Exascale HPC infrastracture.

A set of a hierarchical performance models and simulators, as well as real applications,

have been developing, following the co-design approach for an effective system design. The focus of this thesis is on cortical simulation, and a spiking neural network simulator developed in the framework of the Human Brain Project is described in Chapter 3.

A tuned version of the simulator can be used as mini-application benchmark to evaluate and compare performances and energy-to-solution capabilities on low-power and traditional computing architecture, as described in Chapter 4.

Finally, Chapter 5 describes the design and the implementation of a particular solution obtained with a Network IP for HPC infrastracture, and the performance achieved.

1.7 Personal contribution

Most of the content of the thesis reflects the activities I have carried out during these years as member of the APE Research Group [32] and as participant in the European projects ExaNeSt and EuroExa. In particular:

- I have been in charge as Lead Editor of the "*D3.2 Deliverable: Suitable Interconnects Technologies & Multi-tiered Topologies*", for the task T3.2 "Design", from April 2016 to August 2016, aiming at defining the hierarchical multi-tiered network of ExaNeSt.

- I presented an oral contribution [33] at the International Conference on Computing on High-Energy and Nuclear Physics (CHEP 2016), mainly focused on the topics described in Section 2.

- I am in charge as Task Leader of T6.2 "*Hardware Software Integration and API's*" in ExaNeSt (from April 2017 to May 2018). This activity aims at producing the definition of the ExaNet platform and the design and implementation of the ExaNet Network IP, with personal greater focusing on the APElink development, described in Section 5.

- I actively participate to the definition of the benchmarking approach described in Section 4, and to the discussion of the results, although not direclty contributing to the coding of the Distributed and Plastic Spiking Neural Network simulation engine described in Section 3.

- I presented an oral contribution [34] at the International Conference on Parallel Computing and HPC (ParCo 2017), focusing on the results reported in Section 3 and Section 4.

Despite the fact that some topics related to the Human Brain Project have been addressed in this thesis (in particular, concerning the development of the cortical simulator), no ethical issues are involved. Indeed, my activity has been focused exclusively on computing, with no direct access to clinical data, and thus no concerns about security issues are advanced.

2

The ExaNeSt Project

Contents

2.1 Introduction

With the relentless advances in microelectronics technologies and computer architecture, the High Performance Computing (HPC) market has undergone a fundamental paradigm shift. The adoption of low-cost, Linux-based clusters extended HPC's reach from its roots

in modeling and simulating of complex physical systems to a broader range of applications, from cloud computing and deep learning to automotive and energy.

Today, low-energy-consumption microprocessors (the core element of a microserver) dominate the embedded, smartphone and tablets markets, outnumbering x86 devices both in volume and in growth rate. If these trends continue, we can expect to see microservers benefiting from the same economies of scale that in the past favored personal computers over mainframes and, more recently, commodity clusters over custom supercomputers.

The ExaNeSt project [35], started on December 2015 and funded in EU H2020 research framework (call H2020-FETHPC-2014, n. 671553), is a European initiative aiming at developing the system-level interconnect, a fully-distributed NVM (Non-Volatile Memory) storage and the cooling infrastructure for an ARM-based Exascale-class supercomputer. This technological approach for a scalable and low-energy solution to computing is shared with other projects, with the common goal to deliver a European HPC platform: (i) ExaNoDe [36], that focuses on delivering low-power compute elements for HPC, and (ii) ECOSCALE [37], that focuses on integrating FPGAs and providing them as accelerators in HPC systems.

Besides the power-efficiency of compute nodes, several additional challenges have to be overcome in the road towards Exascale. Modern HPC technology promises "true-fidelity" scientific simulation, enabled by the integration of huge sets of data coming from a variety of sources. As a result, the problem of *Big Data* in HPC systems is rapidly growing, fueling a shift towards *data-centric* HPC architectures, that are expected to work on massive amounts of data, thus requiring low-latency access to fast storage. Current storage devices and interconnection networks together provide latencies of the order of hundreds of microseconds, which limit the scalability of data-hungry application models. ExaNeSt aims to address these challenges by storing data in fast storage devices, which will reside close to the processing elements.

2.2 ExaNeSt objectives

ExaNeSt will develop an in-node storage architecture, leveraging low-power NVM devices. The distributed storage system will be accessed by a unified low-latency interconnect, enabling scalability of either storage and I/O bandwidth together with the compute capacity. The unified RDMA-enhanced network will be designed and validated using a testbed based on FPGAs and passive copper and/or active optical channels, allowing the exploration of interconnection topologies, congestion-minimizing routing functions and support to system resiliency. ExaNeSt also addresses packaging and liquid cooling, which are of strategic importance for the design of realistic systems, and aims at an optimal integration which will be

dense, scalable and power efficient.

In an early stage of the project, an ExaNeSt system prototype, characterized by 500+ ARM cores, will be available acting as platform demonstrator and hardware emulator. A set of relevant ambitious applications, including HPC codes for astrophysics, spiking neural networks simulation, engineering, climate science, materials science and big data will support the co-design of the ExaNeSt system. These applications are fundamental to define the requirements for the ExaNeSt architecture and to provide specifications during the design phase. They will be ported accordingly to ultimately evaluate the final solution. The main selection criterion is that the applications should be diverse, substantial, mature and relevant to the Exascale. Thanks to the large variety of software identified (with different algorithms and communication patterns) the interconnect and storage will be tuned over a complex set of data.

2.2.1 Unified interconnects & In-node storage

Fast non-volatile memory (NVM), *i.e.* flash-based, is a key enabling technology for data-centric HPC. Aiming at avoiding excessive latency and energy consumption, ExaNeSt will place these storage devices close to the compute nodes, and make them accessible through fast custom-made interconnects; for comparison, in traditional supercomputers, the storage devices are located in a central location, *i.e.* behind a SAN [1]/NAS [2] network. Placing fast storage devices close to compute elements can significantly improve the latency and the energy efficiency, as data will frequently be available in the local NVMs. Additionally, with this architecture, the capacity and the I/O bandwidth of the storage subsystem scale together with the compute capacity, thus securing that the system maintains its balance scaling it out to millions of nodes.

However, such a novel storage organization does not come without new challenges. To keep the system within the power and cost constraints, a single *unified interconnect* will be designed to handle both storage and application traffic. Storage flows are typically bursty, responsible for backlogs and queuing delays inside the network, and thus they need to be dealt with carefully in order for them not to saturate the interconnect. A well-designed interconnect should segregate flows, through priority queues, or provide congestion control in order to protect the latency-sensitive computation messages. Additionally, the network should minimize the hops, while providing high bisection bandwidth.

Backplane interconnects can deliver high-bandwidth connectivity among the devices that

[1] Storage Area Network
[2] Network Attached Storage

reside in the same chassis or rack. In ExaNeSt, this concept will be extended, exploiting the opportunities offered by system packaging, to provide high-bandwidth connections among "neighbours" across different levels of the hierarchy (computing nodes on the same daughter-board, daughter-boards on the same blade, blades on the same chassis, etc.). Two examples of such an interconnect are shown in Figure 2.1. One alternative is to have a direct topology based on the inherent networking capacity of the daughter-boards. The second alternative is to build an indirect topology based on off-the-shelf networking solutions; hybrid networks, with both direct and non-direct connections can be interesting for exascale systems.

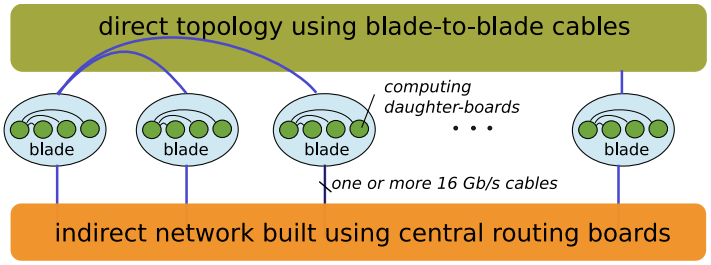

Figure 2.1: Networks that can be tested on the Exanest's prototype: indirect topologies use central switching nodes; direct topologies have direct channels between blades (outer circles); inner circles denote computing daughter-boards within blades. A hybrid network would have both direct and non-direct channels.

In order to reduce the latency of (fast storage or computation) flows, user-initiated Remote Direct Memory Access (RDMA) virtualized mail-boxes are designed, giving applications the ability to use hardware resources directly in user space. The target is to minimize the number of context switches and of data copies inside end-hosts, thus enabling fast inter-process communication.

2.2.2 Rack-level shared memory

Another important feature, in order to improve the performance of many big-data and HPC applications, is the provisioning of fast, extensible DRAM memory. In ExaNeSt, the memory attached to each compute node is of modest size — tens of GBytes per compute node. In order to make large DRAM available to each compute node, remote memory sharing based on *UNIMEM* is enabled, a technology first developed within EuroServer [30]. UNIMEM offers the ability to access areas of memory located in remote nodes. To eliminate the complexity and the costs of system-level coherence protocols [38], the UNIMEM architecture defines that each physical memory page can be cached at only one location. In principle, the node that caches a page can be the page owner (the node with direct access to the memory

device) or any other remote node; however, in practice, it is preferred that remote nodes do not cache pages.

In ExaNeSt, UNIMEM works on a large installation with real applications enabling a *virtual* global address space, rather than a physical one. A global virtual memory page is not necessarily bound upon a specific node or a particular physical memory page. This improves security, allows page migration, and can also simplify multi-programming, just as virtual memory did in the past for single node systems.

ExaNeSt ties computing devices, DRAM memories and SSD disks close together in nodes, in order to reduce the energy consumption of data transfers; it packs many of these nodes within the rack, and connects them using hybrid and hierarchical high-capacity interconnects.

2.2.3 Packaging & Cooling

ExaNeSt adopts the packaging and cooling technology of Iceotope. Iceotope is leader in Totally Liquid Cooled (TLC) technology for computing infrastructures. Since the company's inception, it was recognized that liquid cooling is the future for datacenters, especially for the growth in extreme scale and density. The drivers for Iceotope's focus on TLC versus other methods of cooling include benefits in terms of efficiency, density, total cost of ownership (TCO), as well as the potential for increased performance of processors, an almost silent operation, unless of a dependence on costly, traditional datacenter infrastructures.

The cabinet's secondary coolant is a low cost, bespoke coolant, designed with high electrical resistance and excellent thermal properties, with over twice the heat capacity of mineral oil and half the viscosity. This coolant is circulated by fully-redundant, ultra-efficient pumps, consuming only a fraction of a percent of the energy they move. Each chassis, which is a metal enclosure connected into the rack cooling systems, can accommodate insertion of up to 9 blades, as shown in Figure 2.2.

Each blade is a sealed, self-contained entity, immersed in a sophisticated, non-conductive, engineered fluid: *the primary coolant*. This coolant and the interior of the blade are designed to encourage a state of ultra-convection, known as a convective cell. This cell harnesses natural convection to rapidly moving the heat from the electronics to a secondary (rack level) coolant. When a blade is inserted into its chassis, special valves access the cooling midplane so that the secondary coolant has access to its hotplates to draw the heat away from the inner sealed entity or chamber covering the electronics.

The current Iceotope technology is designed for 72 blades per rack, at 720 Watt per blade, or 52 kW per rack, which allows for a floor density of up to 14 kW/m^2. The roadmap

Figure 2.2: Iceotope's chassis with immersion liquid-cooling.

to exascale calls for a power density of 360 kW per rack. In ExaNeSt the cooling cell is modified to be "hybrid", taking advantage of both phase change and convective flow. This important innovation will require the development of some early stage technology. A new backplane is developed for power supply and signal I/O, and changes the power distribution to 400V DC in order to be able to cope with the currents involved in such a small area.

2.2.4 Applications

The design of the ExaNeSt infrastructure will be driven and tested against scientific and industrial applications widely used in the HPC and big-data arena. The project partners have therefore selected a set of representative and ambitious test applications. Astronomers contribute with cosmological n-body and hydro-dynamical code(s) suited to perform large-scale, high-resolution numerical simulations of cosmic structures formation and evolution [39, 40, 41, 42]. In the Engineering field, where extreme scaling would be of large benefit for scientist and engineers, two applications have been identified: computational fluid dynamics (CFD) [43] and radiation shielding [44]. One application in the area of material science simulation [45], one in the area of weather and climate simulation [46] and the MonetDB analytical DBMS [47] will be used. Finally, in the field of Brain Simulation [48], a natively distributed application representative of plastic spiking neural network simulators, DPSNN, has been selected. This application is deeply described in Chapter 3.

To benefit from the ExaNeSt infrastructure, applications must be re-designed to take advantage of the features that the project provides. A new generation of exascale-ready applications will be developed during the project, and these will be used to make the final tests of the prototype.

Applications are therefore playing three important roles:

- They identify a set of initial requirements to drive the development of exascale-class platforms.
- They will be used to test and refine the ExaNeSt prototypes. Applications will also be used as benchmarks from specific domains, to provide a real comparison against competing solutions.
- Finally, applications will be used as proof of concept to inform the design and development of systems software such as management, control, fault-tolerance, HPC libraries and communication libraries.

2.3 Interconnects

Current HPC systems employ one or more ultra-fast interconnects dedicated to inter-processor communication, and a separate, frequently commodity-based network for storage traffic. The most advanced inter-processor interconnects, although customized to provide ultra-low latencies, typically assume benign, synchronized processor traffic [49].

ExaNeSt, driven by strong power and cost incentives, focuses on a tight integration of fast storage NVMs at the node level using UNIMEM to improve on data locality. To fully exploit new NVMs with access latencies approaching a few tens of microseconds, we have to connect them in a low-latency, system-wide interconnect with sub-microsecond latency capabilities. In this project, we advocate the need for a unified, cross-layer optimized, low-power, hierarchical interconnect that provides equidistant communication among compute and storage devices merging inter-processor traffic with a major part of storage traffic. This consolidation of networks is expected to bring significant cost and power benefits, as the interconnect is responsible for 35% of the power budget in supercomputers, and consumes power even when it idles [50].

ExaNeSt will address the different levels of the interconnect, examining suitable low-power electrical and optical technologies and appropriate topologies. A network topology matching the structure of the applications running on top of it enables maximum efficiency.

In practice, interconnect topologies are severely constrained by system packaging. We address system packaging and topology selection in tandem, aiming at multi-tier interconnects [51], to address the disparate needs and requirements at separate building blocks inside the rack. Furthermore, we will address *inter-rack* interconnects, which span the entire system. This is considered separately because of the fundamentally disparate power and cost constraints that reign outside the enclosure of a rack. Both commodity and proprietary, electronic and optical interconnects will be examined and "judged" based on their readiness and power/performance trade offs.

The frequency and volume of checkpoint/resume traffic in exascale systems, as well as the presence of storage inside the interconnect, mandates sophisticated congestion control [52] and trouble-shooting diagnostics. Therefore, the allocation of shared resources such as interconnect links should be optimized for application requirements. ExaNeSt will provide quality-of-service (QoS) inside the interconnect, using hints and directions from higher layers. Small messages, such as synchronization and storage meta-data, will be prioritized appropriately. Support for QoS is required in order to isolate flows with different latency/throughput requirements and to prioritize latency-sensitive messages.

We plan to design a novel rate-based congestion control mechanism that will react to critical events, such as filled queues or links experiencing high fan-in, and will slow down or dynamically reroute the offensive flows at the sourcing host or RDMA engine. Small "synchronization" messages will be exchanged using remote load and store commands, as defined by the UNIMEM architecture, and in ExaNest these messages will be accelerated appropriately by the network interfaces and the interconnect. For larger messages, we will provide multi-channel RDMA engines, which can be shared among different threads, processes, VM's, or compute nodes. Our multi-channel RDMA will also provide performance isolation to its users for improved privacy and QoS.

2.3.1 Multi-tiered, scalable interconnects for unified data and storage traffic

The development of an interconnect technology suitable for exascale-class supercomputers is one of the main goals of the project; we envision it as a hierarchical infrastructure of separate network layers interacting through a suitable set of communication protocols. Topologies in the lowest tiers are hardwired due to choices made in the prototype design phase. However, design at the network level is configurable and will be the subject of study throughout the next year. An overview of the foreseen interconnects is in Table 2.1.

The *Unit* (described in Section 2.3.2) of the system is the Xilinx Zynq UltraScale+ FPGA,

	Hierarchy	Switching			Fanout T1-T2	Bandwidth	Latency
Tier 4	System	Optical	Ethernet		< 200 rack		
Tier 3	Rack	Optical	Ethernet	APEnet	$3 \div 10$ chassis		
Tier 2	Chassis		Ethernet	APEnet	$36 \div 96$ nodes		
Tier 1	Mezzanine Blade			APEnet	$4 \div 16$ nodes	T1 = 320 Gbps T2 = 800 Gbps	400 ns
Tier 0	Node	AXI Xbar		APEnet	4 FPGAs	LVDS; 14.4 Gbps HSS; 32 Gbps	50 ns 400 ns
FPGA	Unit	AXI Xbar			4 cores	~ 25 Gbps	300 ns
A53	Core	AXI Xbar					

Table 2.1: The ExaNeSt multi-tiered network. Track-1 and Track-2 are indicated as T1 and T2.

integrating four 64-bit ARMv8 Cortex-A53 hard-cores running at 1.5 GHz. This device provides many features, the following being the most interesting: (i) a very low latency AXI interface between ARM subsystem and programmable logic, (ii) cache-coherent accesses from the programmable logic and from the remote unit and (iii) a memory management unit (MMU) with two-stages translation and 40-bit physical addresses, allowing external devices to use virtual addresses and thus enabling user-level initiation of UNIMEM communication.

The *Node* — described in Section 2.3.3 — is the Quad-FPGA Daughter-Board (QFDB) containing four Zynq Ultrascale+ FPGAs, 64 GB of DRAM and 512 GB SSD storage connected through the ExaNeSt Tier 0 network. The inter-FPGA communication bandwidth and latency affect the overall performance of the system. As a consequence, at QFDB level, ExaNeSt provides two different networks, one for low-latency exchanges based on LVDS channels and AXI protocol, the other for high-throughput transmissions through High Speed Serial links (HSS) based on the APEnet communication protocol described in Chapter 5.

For inter-node communication, the QFDB provides a connector with ten bidirectional HSS links for a peak aggregated bandwidth of 20 GB/s. Four out of ten links connect neighbouring QFDBs hosted on the *Blade* (also known as Blade) ((Tier 1). The first Mezzanine prototype (Track-1) — described in Section 2.3.4 — enables the mechanical housing of 4 QFDBs hardwired in a 2D cube topology (a square) with two HSS links (2×16 Gb/s) per edge and per direction. The remaining six HSS links, routed through SFP+ connectors, are mainly used to interconnect mezzanines within the same Chassis (Tier 2). Furthermore, they can also be exploited to modify the Intra-Mezzanine topology.

ExaNeSt will develop two liquid-cooled prototypes — Track-1 and Track-2. Track-1 will be used to test the interconnects, storage and system software technologies developed in the project. Track-2 will allow denser racks benefiting from the new liquid cooling that will be developed by Iceotope.

Track-1 enables the safe mechanical housing of four QFDBs in a custom-made blade.

Nine such blades will fit within an 11U (approximate height, the blade are hosted vertically) chassis. Thus each chassis hosts 36 QFDBs, meaning 576 ARM cores and 2.3 TB of DDR4 memory — approximately 43 cores and 210 GB of memory per 1U of cabinet height. Finally each Track-1 rack will host 3 chassis.

Track-2 will enable a mezzanine made of 16 QFDBs, with 6 blades fitting into a shorter approximately 8U height half-depth chassis. Thus, a "full depth" system can host 12 blades (6 blades on each side) or 192 QFDBs in 8U of cabinet height, — *i.e.* approximately 24 QFDBs, 384 cores and 1.5 TB per 1U of cabinet height. This translates to a compute density of 384 cores plus 96 powerful FPGAs and 1.5 TB of DDR4 memory per 1U of cabinet height. Table 2.2 summarizes the Track-1 and Track-2 set-up.

	Track-1	**Track-2**
cores per blade	64	256
memory per blade [GB]	256	1024
FPGAs per blade	16	64
cores per chassis	576	1536
memory per chassis [GB]	2304	6144
FPGAs per blade	144	384
core per rack	1728	15360
memory per rack [GB]	6912	61440
FPGAs per blade	432	3840
core per equivalent 1u	∼43	384
memory per equivalent 1u [GB]	∼173	1536
FPGAs per equivalent 1u	∼11	96

Table 2.2: The ExaNeSt Track-1 and Track-2 overview

The Inter-Chassis (Tier 3) and Inter-Rack (Tier 4) interconnects round up the multi-tiered network. In ExaNeSt the adoption of custom (photonics) solutions despite of COTS top-of-the-rack router is under evaluation.

Optical interconnects

Photonic interconnects are envisaged to overcome the so-called communications bottleneck of its electronic counterparts. An extensive research has been carried out regarding both on-chip and rack-to-rack photonic interconnects in terms of power, bandwidth and latency. Regarding board-to-board interconnects, it is expected that the bit rate per channel and the number of wavelength division multiplexing (WDM) channels will continue to grow in coming years, with the total capacity per link potentially reaching 1 Tb/s, using 40 channels $\times$ 25 Gb/s per channel. ExaNeSt will explore the most suitable technology in terms of efficiency and performance constraints so as to design an all-optical proof-of-concept switch. The plan is to use 2×2 and 4×4 optical switches as the main building blocks. Based on

this design, we will fabricate a small-scale prototype able to fulfill demanding speed data transmission rates with low losses and low latency.

Resiliency

We target a unified monitoring scheme in the interconnect which, in collaboration with appropriate agents located at different layers of the system stack, will timely overlay critical events concerning the power consumption, the load and health of network links, and system endpoints.

Network-level tolerance and recovery from soft or hard errors is an integral part of system resiliency and a key technology in exascale systems [53, 54]. We plan to leverage RDMA communication to enable dynamic routing and also to recover corrupted and undelivered packets. As the RDMA'ed packets have guaranteed space at the receiving endpoint, we can tolerate out-of-order delivery without being exposed to the danger of destination-induced deadlocks.

An element of special importance is to ensure application level optimization (task placement and scheduling) in order to minimize I/O and other communication overheads by exploiting temporal and spatial locality. Finally, we plan to focus on HPC libraries (*e.g.* MPI collectives) and storage (*e.g.* metadata) traffic acceleration, using a software/hardware co-design approach. All-to-all and scatter-gather collective communications are commonly found in many HPC applications [55, 56] and become more demanding as the scale and the parallelism of the applications increase. We will study possible optimizations for HPC-relevant traffic patterns and also extend them to accelerate storage [57]. Equipping the interconnect with hardware multicast emerges as an interesting communications accelerator.

2.3.2 The ExaNeSt Unit

Field-Programmable Gate-Arrays (FPGA's) that integrate ARMv8 hard macros form an excellent and very convenient platform for the R&D because they offer, at the same time, (i) high-performance UNIMEM-compatible cache coherent interfaces; (ii) reconfigurable logic that enables experimentation with novel interconnect and storage protocols; and (iii) arithmetic hard-macros and SRAM blocks embedded in reconfigurable logic, that can provide novel and high-performance FP and other dedicated accelerators.

ExaNeSt will use the ZU9EG model of Xilinx Zynq UltraScale+ FPGA. A block diagram of this FPGA is shown in Figure 2.3. This is the first and only model of Zynq UltraScale+ available in 2016. In particular, for compactness, we will use the small package, with 900

soldering balls. The main features of this FPGA that make it appropriate for ExaNeSt and the three related projects are the following:

- Four 64-bit ARM Cortex-A53 cores running at 1.5 GHz.
- DRAM controller for high-throughput external DDR4 main memory.
- ACE port: cache-coherent accesses from the programmable logic and from remote nodes, as required by the UNIMEM architecture.
- AXI ports: very low latency external interfaces, directly in the native system protocol.
- MMU500: System (I/O) MMU with two stage translation and 40-bit physical addresses (maximum foreseen in current ARMv8 implementations, suffices for 1 TB window into the global address space); also allows external devices to use virtual addresses, thus enabling user-level initiation of UNIMEM communication.
- High throughput communication: multiple, wide on-chip interfaces; we use 16 (out of the 24 available on the FPGA) high-speed-serial (HSS) external links — *i.e.* 6 intra-QFDB, 4 intra-Mezzanine and 6 inter-Mezzanine. The large number of links and reconfigurable resources inside these FPGAs will be used to implement high-performance interconnection network routers.
- High capacity for floating-point (FP) arithmetic acceleration: besides the (restricted) capabilities of the Mali-400 GPGPU that this FPGA offers inside its processing system, the FPGA reconfigurable logic includes 2.5 thousand Digital Signal Processing (DSP) slices, where each slice is a 32-bit add-multiply unit and can operate at 300 to 400 MHz; this amounts to an aggregate peak capacity around one Tera fixed-point multiply-accumulate operations per second.

2.3.3 The ExaNeSt Node: QFDB

Each QFDB (Quad FPGA Daughterboard) features 4 Zynq Ultrascale+ FPGAs. One of those is named "Network" FPGA and is responsible for routing external traffic providing connectivity to the external world through ten 16 Gbps High Speed Serial Links. The bottom right FPGA, named the Storage FPGA, provides connectivity to the NVMe memory — *i.e.* the M.2 SSD device of capacity half to one TeraBytes — through PS-GTR transceivers implementing a 4xPCIe Gen 2.0 channel. With four 64-bit ARM cores per FPGA, we get 16 cores per QFDB. Whereas these ARM cores provide a modest computing power (24

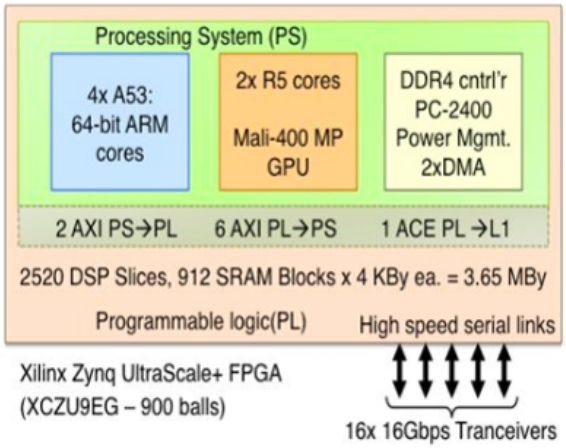

Figure 2.3: The ExaNeSt unit.

GFLOPS in total), the programmable-logic fabric of ZU9EG provides many-thousand DSP slices, with an aggregate peak capacity around 1 to 4 TeraFLOPS per QFDB, depending on the operation and its (single or double) precision. The two FPGAs of each pair on the QFDB are connected together through both 16 LVDS pairs overall (*i.e.* 8 in each direction) and two 16 Gbps High Speed Serial Links (or GTH channels) provide a low-latency communication channel and a high-bandwidth communication channel, respectively. The LVDS pairs offer a total bandwidth of up to 12.8 Gbps in each direction between each FPGA pair, while two GTH transceivers offer a total bandwidth of up to 32 Gbps. A 16 GByte DDR4 memory module is connected to the Processing System of each FPGA. The power consumption of the first version of the QFDB is expected to be $\sim$ 120 Watts. Finally, each FPGA can boot either from 2 QSPI non-volatile memories, or from a Micro SD card. In Figure 2.4 and 2.5, the QFDB and its block diagram are shown.

2.3.4 The ExaNeSt blade

The topology of the mezzanine is hardwired and almost fixed in Track-1. Each Mezzanine will feature 8 connectors to host Daughterboards, 32x SFP+ connectors for communication with remote QFDBs (residing on different mezzanines) and 6x RJ45 connectors for GbE connections for management. The project budget limits the total number of FPGAs to be acquired (the most costly component). The willingness to experiment anyway cooling and engineering solutions at a largest scale, and the desire to prove ExaNeSt solutions also on previous generation components led to a compromise where each Mezzanine hosts four QFDBs, two KALEAO Gen0 boards and two thermal-only mock-up boards (see Figure 2.6 and Figure 2.7).

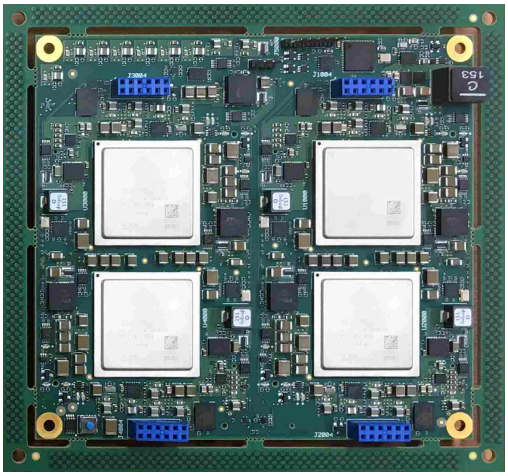

Figure 2.4: The ExaNeSt node: the QFDB

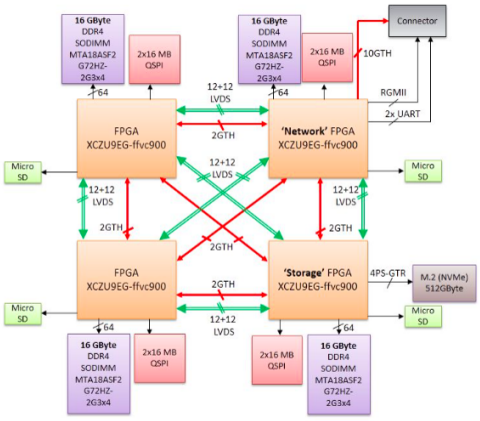

Figure 2.5: The node block diagram

The QFDBs provides a connector hosting the transceiver for 10 HSS links (or channels), 6 of which are connected to external link cages (SFP+) for an aggregate bandwidth of 96 Gbps and the remaining 4 are used to connect to neighbouring (same-mezzanine) QFDBs. The QFDBs are hardwired in a 2D-cube topology with two HSS links (2×16 Gb/s) per edge and per direction. External connection could be exploited at Tier 1 to implement an all-to-all topology by means of the diagonals to optimize latency following application requirements.

The 2 slots dedicated to KALEAO Gen0 boards are connected to each other using two channels (2 HSS links per direction), thus providing a bandwidth of 2×16 Gb/s per direction; 4 additional channels from each KALEAO Gen0 slot are used to connect to SFP+ cages, providing a bandwidth of 4×16 Gb/s to the external world. Four transceivers from each connector are left unconnected on each of these slots. Finally, the thermal-only mock-up slots have all their transceivers unconnected.

In Track-2, the plan is to double the number of slots (and thus the computing power) per blade, using a double-sided mezzanine board. Thus, in stage 2, there will be 16 slots available for daughterboards (DB). On each DB connector, there will be again a total of 10 transceivers, 4 of which will be routed to the backplane, and from there to the switching blades; this leaves us with 6 transceivers available for channels of the intra-mezzanine interconnect.

The optimal blade-internal topology for Track-2 is currently under analysis and it will be selected soon. As an example, a preliminary possible configuration is a Manhattan-street network (it can also be viewed as an irregular 3-D mesh): it consists of two parallel ladders, one for each face (*i.e.* side) of the mezzanine. The 4 nodes belonging to the same ladder floor (2 from each face) create a square, and are interconnected with all-to-all links. The diagonal

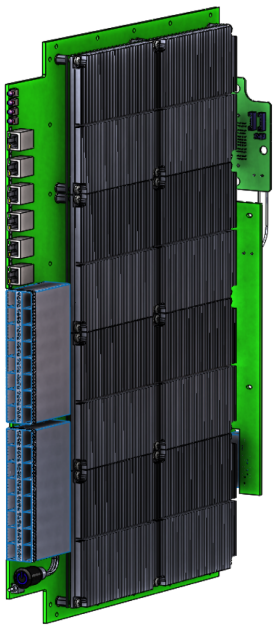

Figure 2.6: Rendering of the ExaNeSt mezzanine for Track-1

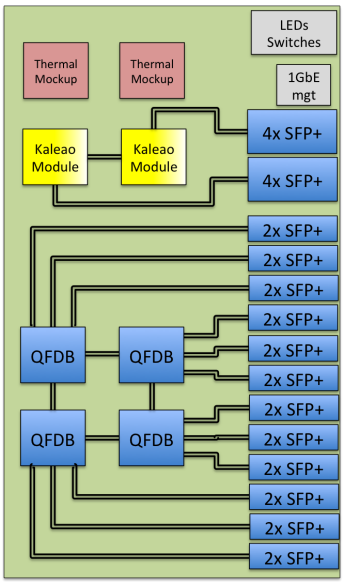

Figure 2.7: Block Diagram of the ExaNeSt mezzanine

links within each square do not necessarily cross, as they can traverse separate PCBs. Top and bottom floors have two channels for the (non-diagonal) connections of the square and one channel for their up and down connections. Middle floors have two channels for up and down connections and one channel for square connections — *i.e.* getting to the other side of the mezzanine. All diagonal connections consist of a single channel.

2.4 Topologies

ExaNeSt explores both *direct* blade-to-blade and *indirect* blade-switch-blade networks. The former type, with direct links (Inter-Mezzanine) between blades, is frequently called "switch-less" and has been employed in many HPC installations. These interconnects distribute the switching and routing functions to units that are integrated close to computing elements. The latter will be tested connecting the blades to commercially available components, based on ASICs or FPGAs.

2.4.1 Track-1

Each mezzanine provides 24 SFP+ connectors to communicate with other mezzanines within the same chassis. So many independent channels allow for a high level of flexibility to experiments with several direct network topologies. A first scenario is shown in Figure 2.8 where 2D torus topology is chosen to interconnect the QFDBs of the 9 blades of a chassis. The solid and dotted lines are the intra-Mezzanine and inter-Mezzanine I/O interfaces respectively. Since local (within the mezzanine) and remote (neighbouring mezzanine) QFDBs are in the same network hierarchy, 2 HSS links per direction for remote channels are used to balance the network capability. A 6×6 Torus topology is the resulting configuration, where the longest path consists of 6 hops implementing a Dimension-Order Routing (DOR) algorithm.

An additional design option would use the "diagonal" links to interconnect the QFDBs in a mezzanine resulting in a all-to-all topology. With this simple modification — which also requires the implementation of a more complex routing algorithm — two hops are saved on average, as sketched in Figure 2.9; the estimation for single hop latency is about 400 ns (see Section 5.3.4).

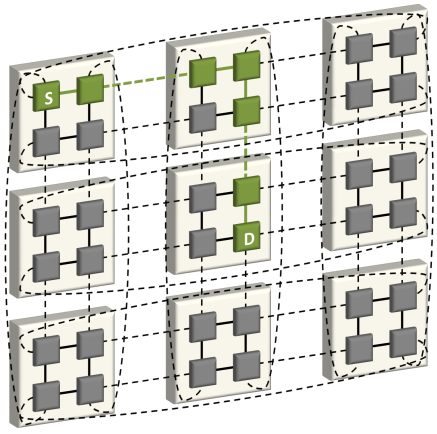

Figure 2.8: QFDBs within the chassis shape a 2D Torus topology (Tier 1/2).

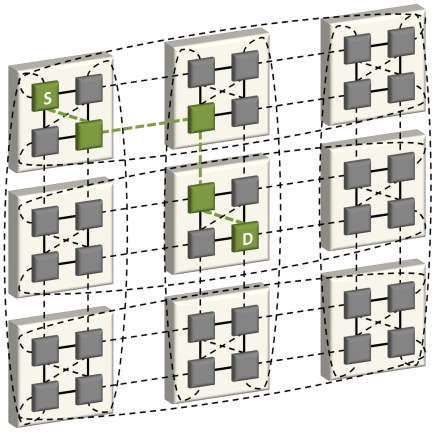

Figure 2.9: Performance boost due to the intra-Mezzanine (Tier 1) all-to-all topology.

A further latency reduction (3 hops for the longest path as depicted in Figure 2.10) is gained by connecting each QFDB of a Mezzanine with their counterparts on neighbouring Mezzanines, shaping four 3×3 2D torus networks (Figure 2.11). Moreover, counterparts QFDBs residing on Mezzanine in neighboring chassis (Tier 3) can be arranged in a 3D torus; in this way we exploit two additional external inter-Mezzanine channels eliminating the diagonal links on the QFDB. Each set of QFDBs is a 3D torus interconnect $3 \times 3 \times C$ where C is the number of chassis.

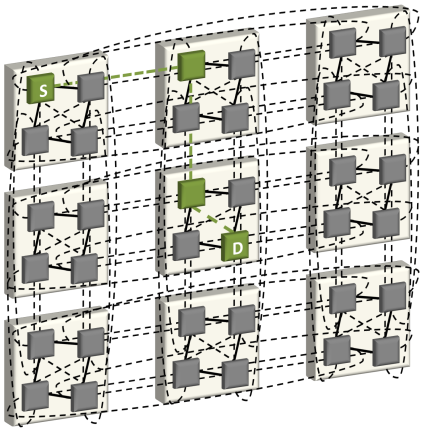

Figure 2.10: An alternative topology to the simple torus network.

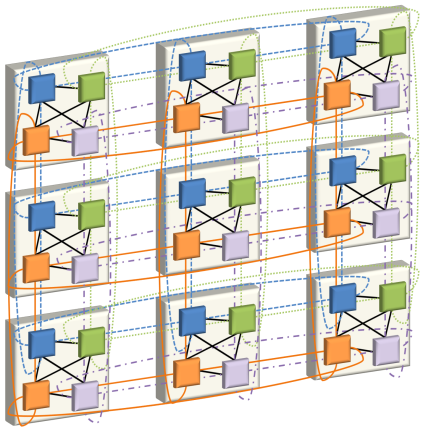

Figure 2.11: Four 2D torus networks interconnecting the mezzanines.

Another scenario foresees a Dragonfly [18] network implementation as in Figure 2.12. Each blade corresponds to a supernode (Figure 2.13) connected to the neighbouring nodes with just one inter-Mezzanine channel.

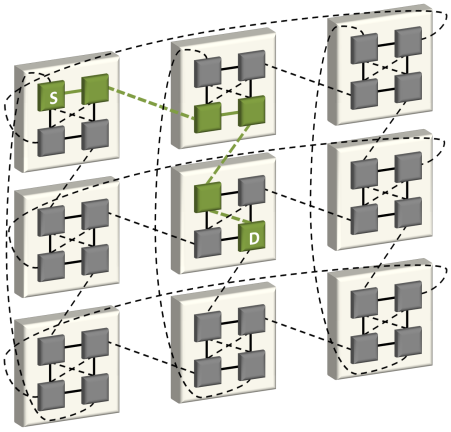

Figure 2.12: Dragonfly topology interconnecting Mezzanine Supernodes (Tier 2).

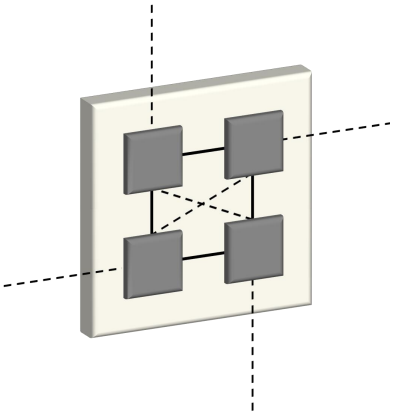

Figure 2.13: Each QFDB exploits only one SFP+ cable for inter-Mezzanine network.

Referring to Figure 2.14, the Xilinx FPGA Systems on Chip (SOCs) are the $4(p)$ terminals connected to each router. The $4(a)$ network FPGAs are the routers of the Group. The Group is the ExaNeSt Mezzanine/Blade corresponding to the supernode of the system. The Routers within the Group are connected through the local channels (Intra-Mezzanine channels) with a 2D-Cube (or all-to-all topology) in Track-1. Groups are connected with $1(h)$ global channel (Inter-Mezzanine channel), hence the radix of the Router within the Network FPGA is $k = p+a+h-1 = 8$. The mezzanine is instead connected with $a \times p = 16$ connec-

tions to terminals and $a \times h = 4$ connections to global channels, while all network FPGAs in a mezzanine act as a virtual router with radix $K = a(p + h) = 20$. Finally, mezzanines can be connected in all-to-all topology exploiting SFP+ cables. In this topology, every blade corresponds to a supernode. The supernodes are connected in an all-to-all fashion using SFP+ cables. Thus, this topology requires 8 SFP+ links per blade, in order to connect each blade to its eight peers. Within each blade, an all-to-all topology among the local QFDBs is assumed. To achieve this, we can dedicate 2 SFP+ per QFDB (per direction) in order to implement the diagonal links that are missing from the passive on-mezzanine interconnect. Each QFDB is connected to six SFP+ (bidirectional) links. Thus, with two of them dedicated for the missing diagonals, we are left with four bidirectional links. We can use two of them to connect to remote supernodes.

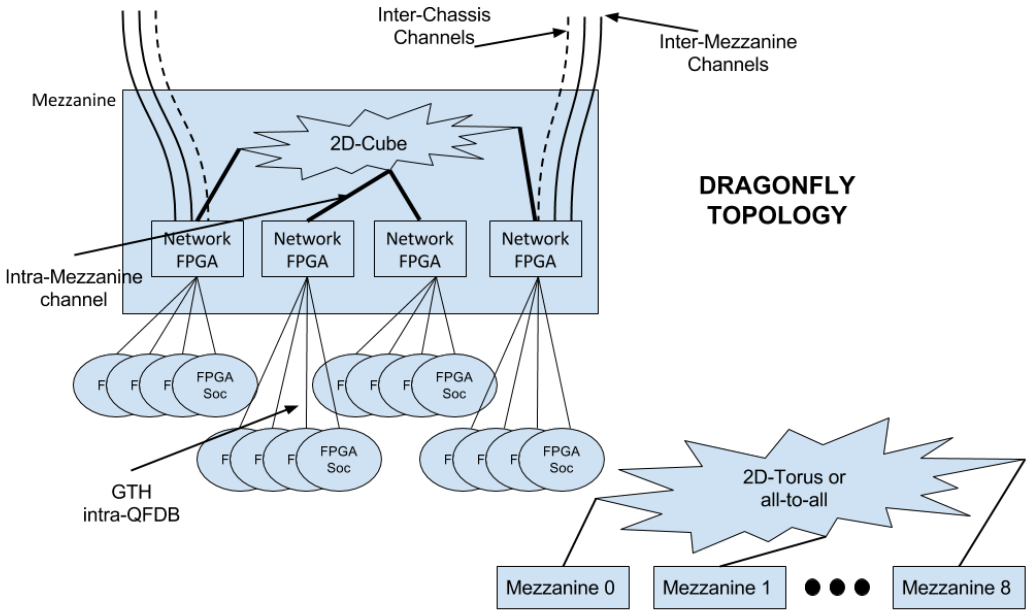

Figure 2.14: Dragonfly overview

2.4.2 Track-2

In Track-2, the ExaNeSt prototype will be radically reformed. The new cooling technology that will be developed within the project will allow a higher density of computing elements. ExaNeSt will demonstrate the new technology via a new chassis prototype, equipped with

mid-chassis switch blades, connected to computing blades through backplane HSS channels

In order to interconnect computing with switch blades, each computing mezzanine will route four bidirectional HSS links from every local computing element (QFDB) to an edge backplane connector. The backplane will in turn route these HSS links to the two switching blades (two bidirectional HSS links from each QFDB to each switching blade). Each switching blade will implement a central router for 6 (blades) $\times 16$ (QFDBs per blade) $\times 2$ (bidirectional HSS links per QFDB) $= 192$ bidirectional ports. Additional ports will be provided for uplinks, which will connect the chassis to its outside world.

Track-2 prototype will not provide the flexibility of Track-1, in terms of computing topologies that can be tested on top of it. Instead, the target for Track-2 is to develop the first version of a commercially viable system that will demonstrate the cooling technology that can support the extreme-density required for exascale-class supercomputers and data centers. In Track-2 chassis, the SFP+ cables are replaced by backplane HSS wiring, thus alleviating the cost and area overheads of cables, and at the same time increasing connectivity. The two mid-chassis switch blades will aggregate the traffic from the computing blades, and will also provide uplinks to other chassis or top-of-the-rack routers. This design with two discrete switching nodes is fault tolerant and can provide the resiliency properties expected of real products.

3

Distributed and Plastic Spiking Neural Network

Contents

3.1 Introduction

The main focus of several neural network simulation projects is the search for a) biological correctness; b) flexibility in biological modelling; c) scalability using commodity technology — *e.g.* NEST [58, 59], NEURON [60], GENESIS [61]. A second research line focuses more explicitly on computational challenges when running on commodity systems, with varying degrees of association to specific platform ecosystems [62, 63, 64, 65]. An alternative research pathway is the development of specialized hardware, with varying degrees of flexibility allowed — *i.e.* SpiNNaker [66], BrainScaleS [67].

The DPSNN simulation engine focuses along two lines: (i) having a quantitative benchmarking tool for the evaluation of requirements for future embedded and HPC systems — *e.g.* in the framework of the EuroExa, ExaNeSt [35] and EURETILE [48] projects — and (ii) the acceleration of the simulation of specific models in computational neuroscience — *e.g.* to study slow waves in large scale cortical fields [68, 69] in the framework of the CORTICONIC [70] and HBP [2] projects.

We present the impact of the range of lateral connectivity on the scaling of distributed point-like spiking neural network simulation when run on up to 1024 software processes (and hardware cores) for cortical models including tens of billions of synapses. It is worth noting that a simulation including a few tens of synapses is what is required to simulate the activity of one cm^2 of cortex at biological resolution (*e.g.* 54k neuron/mm^2 and about 5k synapses per neuron in the mouse visual cortical area [71]). The capability to scale a problem up to such a size allows simulating an entire cortical area.

Recent studies point out that lateral connectivity plays a central role in many different cerebral areas, from cat primary visual cortex [72], to rat neocortex [71, 73], just as examples. For instance, in rat neocortex, the impact of lateral connectivity on the pyramidal cells in layer 2/3 and layer 6A, results in $\sim 75\%$ of incoming remote synapses to neurons of these layers. Novel models should include exponential decay of connectivity to describe the longer-range distance dependent intra-areal non-local connection probability ($exp(\frac{-r}{\lambda})$). Decay constants in the range of several hundred microns have been proposed as matching the experimental results. This kind of intra-areal long-range lateral connectivity poses novel simulation requirements in comparison to previous studies that considered intra-areal synaptic connections dominated by local connectivity: local connections have been estimated as counting for at least 55% of total synapses, reaching also a ratio of 75% [74]. In previous studies, lateral connectivity has been often described with a shorter-range Gaussian model [75]($exp(\frac{-r^2}{2\sigma^2})$).

Here we present measures about the scaling of simulations of cortical area patches of

different sizes represented by two-dimensional grids of "cortical modules". Each cortical module includes 1240 single-compartment, point-like neurons (no dendritic tree is represented) each one receiving up to ~ 2050 recurrent synapses (instantaneous membrane potential charging) plus those bringing external stimuli. Assuming a 100 μm reticular spacing between neighbouring columns, the larger simulated problem size corresponds to the simulation of a cerebral cortex tile, represented by 11.4 M neurons and 29.6G recurrent synapses.

The increment in the range of remote connection is expected to have an impact on the performances of neural network simulators. Exponentially decaying lateral connectivity (longer-range) are compared to a Gaussian connectivity decay (shorter-range), mainly analyzing the scaling behaviour and the memory occupation of our Distributed and Plastic Spiking Neural Network simulation engine (DPSNN in the following) when used with the two connectivity distributions.

On the DPSNN simulator, the selection of the connectomic model is crucial, due to the fact that the synaptic messages exchanged between neurons correspond to communication tasks among MPI processes: the higher the number of lateral synaptic connections and the longer the distance is, the more intensive the communication task among processes become.

The article is structured as follows: in Section 3.2 we describe the main features of DPSNN and the parallel and distributed approach used for its implementation; the model used for the measurements of this paper is summarized in Section 3.3, with a specific description of the two different schemes adopted for the lateral intra-area connectivity used to test DPSNN scaling capabilities; subsequent sections report the Results of this work (Section 3.4) and the Discussion (Section 3.5).

3.2 Description of the Spiking Neural Network simulator

The Distributed and Plastic Spiking Neural Network is a mixed time- and event-driven spiking neural network simulation engine implementing synaptic spike-timing dependent plasticity. It has been designed from the ground up to be natively distributed and parallel, and should not pose obstacles against distribution and parallelization on several competing platforms. Coded as a network of C++ processes, it is designed to be easily interfaced to both MPI and other (custom) Software/Hardware Communication Interfaces.

In DPSNN, the neural network is described as a two-dimensional grid of cortical modules made up of single-compartment, point-like neurons spatially interconnected by a set of incoming synapses. Cortical modules are composed by several populations of exhitatory and inhibitory neurons. Cortical layers can be modelled by a subset of those populations. Each synapse is characterized by a specific synaptic weight and transmission delay, accounting

for the axonal arborization. The two-dimensional neural network is mapped on a set of C++ processes interconnected with a message passing interface. Each C++ process simulates the activity of a cluster of neurons. The spikes generated during the neural activity of each neuron are delivered to the target synapses belonging to the same or to other processes. The "axonal spikes", that carry the information about the identity of the spiking neuron and the original emission time of each spike, constitute the payload of the messages travelling across processes. Axonal spikes are sent only toward those C++ processes where a target synapse exists.

The knowledge of the original emission time of each spike and the transmission delay introduced by each synapse allows for the management of synaptic Spike Timing Dependent Plasticity [76], which produces effects of Long Term Potentiation/Depression (LTP/LTD) on the synapses.

3.2.1 Execution flow: a mixed time and event-driven approach

Two main phases characterize a DPSNN simulation: 1) the creation and initialization of the network of neurons and of the axonal arborization of synapses interconnecting the system; 2) the simulation of the network dynamic (neurons and synapses).

For the simulation phase, a combined event-driven and time-driven approach has been adopted, inspired by [77]: the dynamic of neurons and synapses is simulated when the event arises, while the message passing conveying the axonal spikes among processes, as well as the application of Long Term Plasticity (when activated) is performed at regular time steps.

The phase of simulation of the network dynamic can be further decomposed into the following steps: 2.1) neurons, that produced spikes during the previous time-driven simulation step, are identified and the corresponding contribution to STDP is calculated; 2.2) spikes are sent through axonal arborizations to the cluster of neurons where target synapses exist; 2.3) delivered axonal spikes are queued into a list for each process, for usage during the appropriate time-step according to the corresponding synaptic delay; 2.4) synapses inject currents into target neurons and the corresponding contribution to STDP is calculated; 2.5) neurons sort input currents coming from recurrent and external synapses; 2.6) neurons integrate their dynamic equation for each input current in the queue, using an event-driven solver.

At a slower timescale, every second in the current implementation, all STDP contribution are integrated in the Long Term Plasticity and applied to each single synapse.

3.2.2 Distributed generation of synaptic connections

The DPSNN simulation engine exploits its full parallelism also during the creation and initialization phase of the network. In a given process, each neuron $i = 1, .., N$ projects its set of recurrent synapses $j = 1, .., M$, characterized by individual delays $D^{i,j}$, plastic weights $W^{i,j}$ and target neurons $K^{i,j}$. Synaptic efficacies are randomly chosen from a Gaussian distribution with a given mean and variance, while synaptic delays can be generated according to exponential or uniform distribution. The moments of the distributions depend on the source and target populations that synapses interconnect. In addition to the recurrent synapses, the system simulates also a number of external synapses: they represent afferent (thalamo-)cortical currents coming from outside the simulated network.

3.2.3 Representation of spiking messages

Spike messages are defined using an Address Event Representation (AER), in which each spike is represented by two numbers: the identifier of the spiking neuron and the exact time of spiking. During simulation, spikes travel from the source to the target neuron. Spikes, whose target neurons belong to the same process, are packed in the axonal spike message.

The arborization of this message is carried out directly by the target process. Deferring as much as possible the arborization of the "axon" reduces the load on the communication network and unnecessary wait barrier.

To this purpose, preparatory actions are performed during the network initialization phase (performed once at the beginning of the simulation), to reduce the number of active communication channels during the iterative simulation phase.

3.2.4 Initial construction of the connectivity infrastructure

During the initialization phase, each process contributes to create the awareness about the subset of processes that should be listened to, during next simulation iterations, based on the information contained in the synaptic matrix interconnecting the cluster of neurons of the network. At the end of this construction phase, each "target" process should know about the subset of "source" processes that need to communicate with it, and should have created its database of locally incoming axons and synapses.

A simple implementation of the construction phase can be realized using two steps. During the first step, each source process informs other processes about the existence of incoming axons and about the number of incoming synapses. A single word, the synapse counter, is communicated among pairs of processes. Under MPI, this can be achieved by an

`MPI_Alltoall()`. This is performed once, and with a single word payload.

The second construction step transfers the identities of synapses to be created onto each target process. Under MPI, the payload — a list of synapses specific for each pair in the subset of processes to be connected — can be transferred using a call to the `MPI_Alltoallv()` library function. The number of messages depends on the lateral connectivity range and on the distribution of cortical modules among processes, while the cumulative load is always proportional to the total number of synapses in the system.

The knowledge about the existence of a connection between a pair of processes can be reused to reduce the cost of spiking transmission during the simulation iterations.

3.2.5 Delivery of spiking messages during the simulation phase

After initialization, the simulator enters the iterative simulation phase. At each iteration, spikes are exchanged between pairs of process connected by the synaptic matrix. The delivery of spiking messages can be split in two steps, with communications directed toward subsets of decreasing size.

During the first step, single word messages (spike counters) are sent to the subset of potentially connected target processes. On each pair of source-target process subset, the individual spike counter informs about the actual payload — *i.e.* axonal spikes — that will have to be delivered, or about the absence of spikes to be transmitted between the pair. The knowledge of the subset was created during the first step of the initialization phase, described in Section 3.2.4.

The second step uses the spiking counter info to establish a communication channel only between pairs of processes that actually need to transfer an axonal spikes payload during the current simulation time iteration. On MPI, both steps can be implemented using calls to the `MPI_Alltoallv()` library function. However, the two calls establish actual channels among sets of processes of decreasing size, as described just above.

For the simple two-dimensional grid of neural columns and for the mapping on processes used in this experiment, this implementation proved to be quite efficient, as reported by the measures presented in the Section 3.4, further refined in the Section 3.5. However, we expect that the delivery of spiking messages will be one of the key point still to be optimized when white area “connectome” is introduced, describing the communication channels among a multiplicity of remote cortical areas.

3.3 Neural Network Configuration

In this study the simulated model has been configured as in the following subsections.

3.3.1 Spiking Neuron Model and Synapses

The single-compartment, point-like neurons used in the measures reported in this paper are based on the Leaky Integrate and Fire (LIF) neuron model with spike-frequency adaptation (SFA) due to calcium- and sodium-dependent after-hyperpolarization (AHP) currents [78, 79]. The dynamic of the neuron is described by the following equations:

$$\frac{dV}{dt} = \frac{V - E}{\tau_m} - g_c \frac{c}{C_m} + \sum J_i \delta(t - t_i)$$

$$\frac{dc}{dt} = -\frac{c}{\tau_c}$$

where $V(t)$ represent the membrane potential and $c(t)$ the fatigue variable used to model the SFA as an activity-dependent inhibitory current. τ_m is the membrane characteristic time, C_m the membrane capacitance and E the resting potential. In the inhibitory neurons, the SFA term is equal to zero. When the membrane potential exceeds a threshold V_θ, a spike occurs. Thereafter, the membrane potential is reset to V_r for a refractory period τ_{arp}, whereas the variable c is increased by the constant amount α_c.

During the construction phase of the network, recurrent synapses are established between pre- and post-synaptic neurons, according to given probabilistic distance dependent connectivity law (see Section 3.3.2). Synaptic efficacies and delays are randomly chosen from a probabilistic distribution as already described in Section 3.2.2.

In addition to the recurrent synapses, the system simulates also a number of external synapses: they represent afferent (thalamo-)cortical currents coming from outside the simulated network, collectively modeled as a Poisson process with a given average spike frequency. The recurrent synapses plus the external synapses yield the number of total synapses afferent to the neuron, referred to as "total equivalent" synapses in the following.

For all the measurements in this work, synaptic plasicity for all the neurons has been disabled.

3.3.2 Cortical Columns and their connectivity

The neurons are organized in cortical modules (mimicking columns), each one composed of 80% excitatory and 20% inhibitory neurons. The modules are assembled in two-dimensional

grids, representing a cortical area slab, with a grid step $\alpha \sim 100$ μm (inter-columnar spacing).

The number of neurons in each cortical module was fixed to 1240, while the number of synapses projected by each neuron depends on the implemented connectivity.

The neural network connectivity is set by the user defining the probabilistic connection law between neural populations, spatially located in the two-dimensional grid of cortical modules. Connectivity can be varied according to the simulation needs, leading to configurations with different numbers of synapses per neuron. In order to study the scalability of the DPSNN simulator on large configurations, and to evaluate the impact of different connectivity loads, two neural systems have been considered in terms of connectivity rules.

The number of synapses projected to the same column (local connections) is kept fixed at 990, while the difference is in the remote connectivity: ~ 250 synapses for the shorter range case and ~ 1240 synapses for the longer one. In particular, the following lateral connectivity rules are adopted:

- Gaussian connectivity — shorter range and lower number of remote synapses: considering preeminent local connectivity with respect to lateral, the rule used to calculate remote connectivity has been set proportional to $A \cdot exp(\frac{-r^2}{2\sigma^2})$, with $A = 0.05$ and $\sigma = 100\mu$m being the lateral spread of the connection probability. The remote connectivity function is similar to that adopted by [75], although with different A and σ parameters. In this case only $\sim 21\%$ of the synapses are remotely projected and reach modules placed within a short distance, spanning a few steps in the two-dimensional grid of cortical modules. The majority of connections ($\sim 79\%$) is local to the module.

- Exponential decay connectivity — longer range and higher number of remote synapses: the connectivity rule for remote synapses calculation is proportional to $A \cdot exp(\frac{-r}{\lambda})$, with $A = 0.03$ and $\lambda = 290\mu$m (the exponential decay constant). This turns out into an increased number of remote connections (52%).

In summary, in case of Gaussian connectivity, the average number of projected synapses per neuron is about 1240, while in case of exponential connectivity this number rises up to ~ 2050.

In both systems, a cut-off has been set in the synapses generation, limiting the projection to the subset of modules with connection probability greater than 1/1000. This turns out into a centered stencil of connected modules of size 7×7 in the first case (Gaussian) and 21×21 in the second case (exponential decay).

For each connectivity scheme, measurements were taken on different problem sizes ob-

tained varying the dimension of the grid of modules and, once fixed the problem size, distributing it over a span of MPI processes to evaluate the scaling behaviour.

We selected three grid dimensions, which, *e.g.* for a columnar spacing of 100 μm, can be already considered representative of interesting biological cortical slab dimensions. Each problem size has been distributed over a different span of MPI processes. Table 3.1 summarizes the set of problem sizes used in our scaling measures. The number of processes over which each network size is distributed varied from a minimum, bounded by memory, and a maximum, bounded by communication (or HPC platform constraints).

Using the Gaussian shorter-range connectivity, an extensive campaign of measures has been conducted, spanning over the three configurations described above. On the contrary, just a preliminary set of measures were taken in the configuration with the longer range exponential connectivity: only a few trials on 24×24 and on 48×48 configuration networks have been performed, in order to compare the DPSNN simulator performances in the different configurations.

3.3.3 Normalized simulation Cost per Synaptic Event

Different network sizes and connectivity models have been used in this scaling analysis. This results in heterogeneous measures of the elapsed time due to different numbers of projected synapses and to the different firing rates of resulting models. For example, the observed firing rate is ~ 7.5 Hz for the shorter range connectivity scheme, and in the range between 32 and 38 Hz for the longer range one (all other parameters being kept constant). However, a direct comparison is possible converting the execution time into a simulation cost per synaptic event. This normalized cost is computed dividing the elapsed time per simulated second of activity by the number of synapses and by their mean firing rate. This way, a simple comparison among different simulated configurations is possible: measures from different simulations can be compared on the same plot. Our simulations include two kind of synapses: recurrent — *i.e.* projected by simulated neurons — and synapses bringing an external stimulus. Summing the number of events generated by recurrent and external synapses, in the following we can normalize the cost to the total number of equivalent synaptic events.

3.3.4 Hardware Platform

The server platform used to run the simulations herein described is GALILEO, a cluster of 516 IBM nodes provided by the CINECA [80] consortium. Each 16-core computational node contains two Intel Xeon Haswell 8-core E5-2630 v3 processors, with a clock of 2.40 GHz.

All the nodes are interconnected through an InfiniBand network. Due to the specific configuration of the server platform, no hyper-threading is allowed. Therefore, in all the following measures, the number of cores corresponds exactly to the number of MPI processes launched at each execution.

3.4 Results

3.4.1 Scaling for shorter range Gaussian connectivity

We collected a set of elapsed times simulating the different problem sizes detailed in Table 3.1, spanning the range from 1 to 1024 MPI processes (or, equivalently, hardware cores).

Grid	**Columns**	**Neurons**	**Number of Synapses**			
			Gaussian Connectivity		**Exponential Connectivity**	
			Recurrent	Total	Recurrent	Total
24×24	576	0.7 M	0.9 G	1.2 G	1.5 G	1.8 G
48×48	2304	2.9 M	3.5 G	5.0 G	5.9 G	7.4 G
96×96	9216	11.4 M	14.2 G	20.4 G	23.4 G	29.6 G

Table 3.1: Problem sizes for the comparison of simulator performances applied to exponential (longer-range) and Gaussian (shorter-range) connectivity

The values plotted in Figure 3.1 show how the execution time per synaptic event changes when the number of cores assigned to the problem is varied. In the ideal case (black dot line in the picture), doubling the used resources, execution time should halve. In our measures, the time needed to simulate the 24×24 grid (with 0.9 G recurrent synapses and 1.2 G total equivalent synapses) scales down from $2.75 \cdot 10^{-7}$ seconds per synaptic event, using a single core, to $4.09 \cdot 10^{-9}$ seconds per event using 96 cores. The actual speed-up is 67.3, loosing 30% compared to the ideal speed-up that in this case would be 96. The speed-up for the 48×48 grid (3.5 G recurrent, 5 G equivalent synapses) is 54.2, while the hardware resources increase by a factor 96. For the 96×96 grid (14.2 G recurrent/ 20.4 G equivalent synapses) the speed-up is 10.8 (16 would be the ideal).

Figure 3.2 represents the weak scaling: the problem size assigned to each core is kept constant and the total problem size is increased proportionally to the number of hardware cores. If normalized by the load per core, the lines corresponding to different loads/core should overlap.

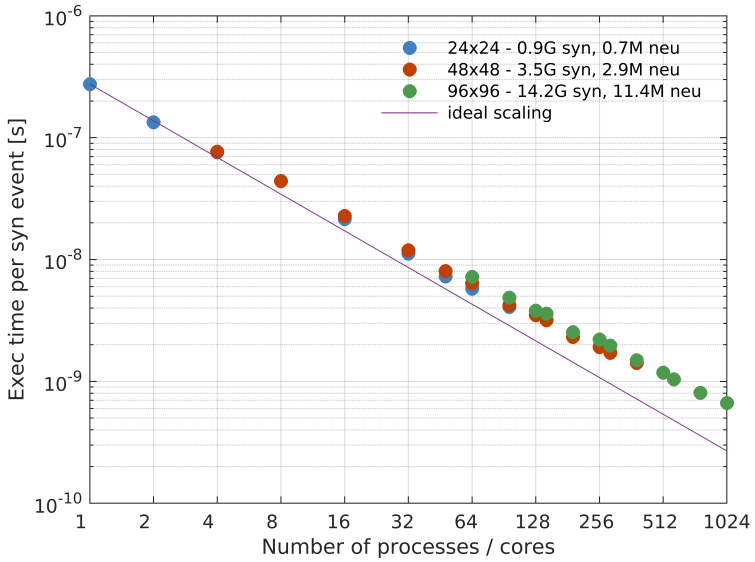

Figure 3.1: Strong scaling for Gaussian connectivity model: the measures are expressed in elapsed time per equivalent synaptic event.

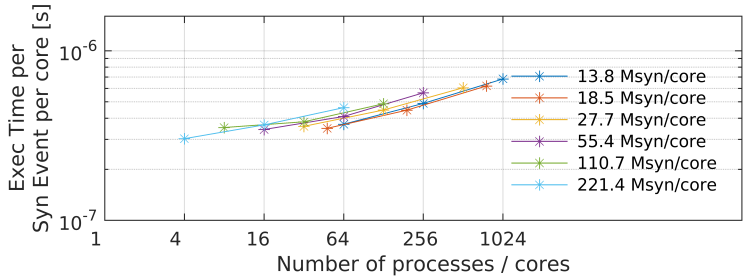

Figure 3.2: Weak scaling for Gaussian connectivity model.

3.4.2 Impact of longer range exponential decay connectivity

Figure 3.3 compares the impact of shorter and longer lateral connectivities on the strong scaling behaviour. Circles represent measurements for the Gaussian decay while squares involve the longer range exponential one.

The introduction of longer range connectivity increases the simulation cost per synaptic event (a $1.9 \div 2.3$ slow-down, see Figure 3.4). The actual elapsed simulation time increased up to 16.6 times for the exponential longer-range connectivity due to the combined effect produced by: (i) the number of synapses projected by each neuron is higher (by a factor 1.65), (ii) the firing rates expressed by the model is between 4.3 and 5.0 times higher and (iii) the higher cost of longer range communication and demultiplexing neural spiking messages. Point (iii) should be well estimated by the slow-down of the normalized simulation cost per synaptic event.

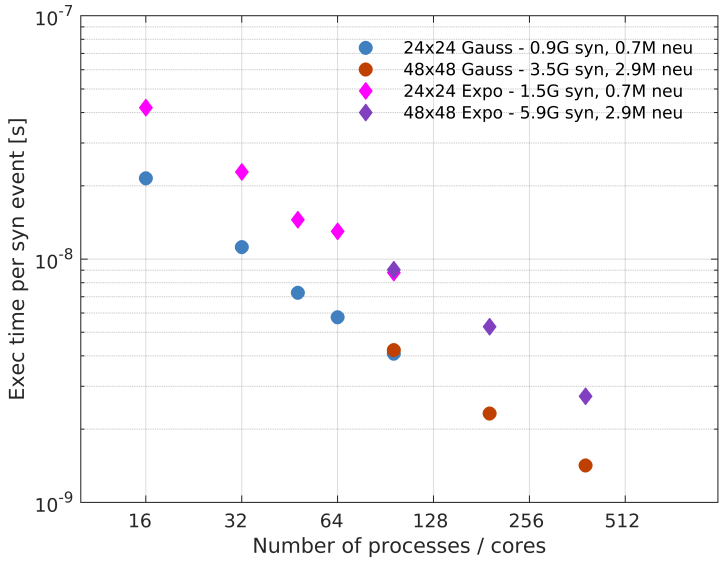

Figure 3.3: Impact of connectivity on DPSNN performances: the graph compares the execution time per synaptic event for the configurations with Gaussian connectivity (shorter range, lower number of synapses — circles) and the one with exponential connectivity (longer range, higher number — squares).

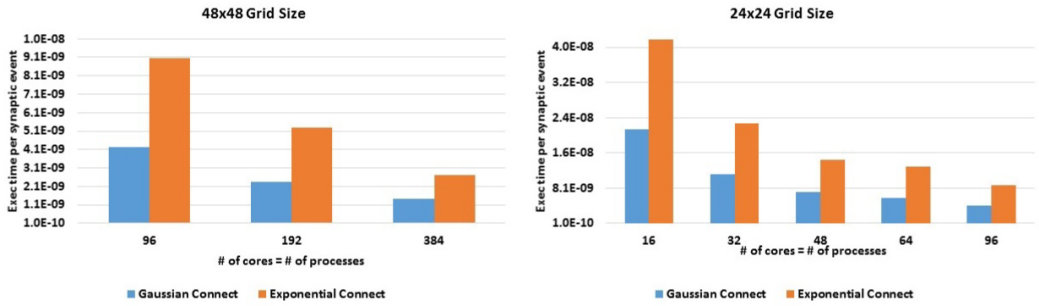

Figure 3.4: Time per simulated synaptic event increased between 1.9 and 2.3 times changing the decay of connection probability from the shorter range Gaussian scheme to the longer range exponential one.

The execution of longer range exponential connectivity on 96 cores, reached about 83% for the 48×48 (5.9 G recurrent synapses) and 79% of the ideal scaling for the 24×24 case (1.5 G recurrent synapses).

3.4.3 Memory cost per synapse

We measured the total amount of memory allocated by the simulator and divided it by the number of synapses to be represented. With no plasticity, each synapse should cost 12 Byte.

Peak memory usage is observed at the end of network initialization, when each synapse is represented at both the source and target process. Afterwards, memory is released on the source process. The forecast of minimum peak cost is therefore 24 Byte/synapse for static synapses. Figure 3.5) shows the maximum memory footprint for different networks sizes and projection ranges, distributed over different numbers of MPI processes. The values are in the range between 26 and 34 B per synapse. We observed that the growing cost for higher number of MPI processes is mainly due to the memory allocated by the MPI libraries.

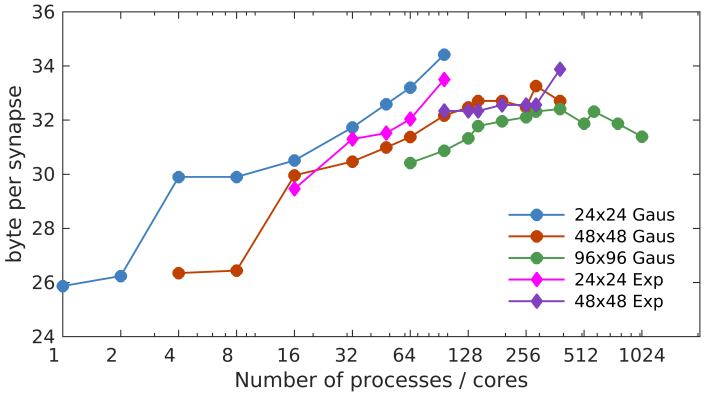

Figure 3.5: Memory occupation in byte per synapse for different configurations in the two connectivity systems

3.5 Discussion

Recent experimental results suggest the need of supporting long range lateral connectivity in neural simulation of cortical areas — *e.g.* modeled by simple exponential decay of the connection probability — with layer to layer specific decay constants, in the order of several hundreds microns. The distributed spiking neural net simulator DPSNN has been applied to two-dimensional grids of neural columns spaced at 100 μm connected using two schemes.

The longer-range connectivity model corresponds to an exponential connectivity decay ($\lambda = 290\mu$m) and to the projection of approximately $\sim$2050 synapses per neuron. The scaling measures are compared to those obtained with a shorter range Gaussian decay of the connectivity, with a decay constant of the order of the columnar spacing and a lower number of synapses per neuron ($\sim$1240). The impact of exponentially decaying connectivity is indeed observable, as expected, and increases the simulation cost per synaptic event between 1.9 and 2.3 times compared to the shorter range Gaussian connectivity law. Notwithstanding this increase, the strong scaling behaviour is satisfactory.

However, we note that a more realistic biological simulation of cortical areas could require a further extension of the connection stencil dimension, with the goal of projecting about ten thousand synapses per neuron. A further element to be considered in whole brain simulation will be the support of white matter connectome, which brings sparse connections at system scale. We demonstrated the DPSNN ability to efficiently simulate grids of neural columns, containing a total of 11.4 M LIF neurons with spike-frequency adaptation, and representing 20.4 G equivalent synapses (for both shorter and longer range connections) on a 1024 core execution platform, with a memory occupation always below 35 byte/syn.

4

DPSNN on ARM-based platforms

Contents

4.1 Introduction

The scaling of the performance of modern HPC systems and applications is strongly limited by the energy consumption. Electricity is the main contributor to the total cost of running applications, and energy-efficiency is becoming the principal requirement for computing devices. In this context, the performance assessment of processors with a high ratio of performance per watt is necessary to understand how to realize energy-efficient computing systems for scientific applications. Processors based on the ARM architecture lead the market of low-power and battery powered devices, such as tablets and smartphones. Several scientific communities are exploring non-traditional many-core processors architectures coming from the embedded market, from the Graphics Processing Unit (GPU) to the System-on-Chip (SoC), looking for a better tradeoff between time-to-solution and energy-to-solution. A number of research projects are active in trying to design an actual platform along this direction.

The Mont-Blanc project [81, 82], coordinated by the Barcelona Supercomputing Center, has deployed two generations of HPC clusters based on ARM processors, developing also the corresponding ecosystem of HPC tools targeted to this architecture. Another example is the EU-FP7 EUROSERVER [83] project, coordinated by CEA, which aims to design and prototype technology, architecture, and systems software for the next generation of datacenter "microservers", exploiting 64-bit ARM cores.

Fast simulations of spiking neural network models play a dual role: (i) they contributes to the solution of a scientific grand challenge — *i.e.* the comprehension of brain activity — and, (ii) by including them into embedded systems, they can enhance applications like autonomous navigation, surveillance and robotics. Therefore, these simulations assume a driving role in shaping the architecture of either specialized and general-purpose multi-core/many-core systems to come, standing at the crossroads between embedded and High Performance Computing. See, for example, [84], describing the TrueNorth low-power specialized hardware architecture dedicated to embedded applications, and [85], discussing the power consumption of the SpiNNaker hardware architecture, based on embedded multicores, dedicated to brain simulation. Worthy of mention are also [58, 65] as examples of approaches based on standard HPC platforms and general-purpose simulators.

The APE Research Group at INFN has developed a distributed neural network simulator [86] as a mini-application and benchmark in the framework of the EURETILE FP7 project [48]. Indeed, the Distributed and Plastic Spiking Neural Network with synaptic Spike-Timing Dependent Plasticity mini-application was developed with two main purposes in mind: as a quantitative benchmarking tool for the evaluation of requirements for future embedded and HPC systems, and as an efficient simulation tool addressing specific scientific problems in computational neuroscience. As regards the former goal, the ExaNeSt project [35] includes DPSNN in the set of benchmarks used to specify and validate the requirements of future interconnects and storage systems; as an example of the latter, the distributed simulation technology is employed in the study of slow waves in large scale cortical fields [68, 69] in the framework of HBP project.

This section describes the porting of DPSNN onto different ARM-based platforms and how running it on low-power CPUs, comparing the resulting computing and energy performances with traditional systems mainly based on x86 multicores. The characterization of DPSNN-generated data traffic is described, highlighting the limitations faced when the application is run on off-the-shelf networking components. The code organization and its compactness give to DPSNN a high degree of tunability, offering the opportunity to test different areas of the platform. The networking compartment is the most stressed when the simulated neural net — composed by a relatively low number of neurons, each one project-

ing thousands of synapses — is distributed over a large number of hardware cores. When the number of neurons per core grows, the impact of both computing and memory increases. For this reason, we employ DPSNN as a general benchmarking tool for HPC systems.

4.2 Porting DPSNN kernels on low-power test-bed

A couple of minor quirks are required to make either MPICH and OpenMPI work on an ARM boards cluster in order to directly port the DPSNN application to the ARM platform. The DPSNN is tested with the ARM Debian OS which made available MPICH 3.2 in packetized form while the available packetized OpenMPI is in version 2.0.2. As regards the software dependencies, the application leans on nothing else than an MPI-compliant compiler and library. Due to the limited resources of the testbed, the data-set must be sized accordingly to fit the available memory — *i.e.* a more constrained set of memory allocations in the initialization phase.

4.2.1 The trenz-based testbed: description and results

The trenz-based prototype is currently composed by four nodes. Each node consists of a TEBF0808 Trenz board which is equipped with a Trenz TE0808 UltraSOM+ module. The Trenz UltraSOM+ consists of a Xilinx Zynq UltraScale+ xczu9eg-ffvc900-1-e-es1 MPSoC and 2 Gbytes of DDR4 memory. The Zynq UltraScale+ MPSoC incorporates both a processing system and the programmable logic — not used to test the porting. The main characteristics of the processing system are the following:

- Quad-core ARM Cortex-A53 with frequency up to 1.2 GHz;
- Dual-core ARM Cortex-R5 with frequency up to 600 MHz (used mainly for RT network processing;
- 32 kbytes of instruction cache (per core);
- 32 kbytes of data cache (per core);
- 1 MByte of L2 cache.

All four nodes are connected together through a 1 Gbps Ethernet-based network. In this regard, some reshuffling of allocations relieving the pressure the application put on the memory subsystem of the Trenz boards was necessary in order to make it run.

# Nodes ÷ # Cores	Time (ARM)	Time (Intel)
1 ÷ 1	3656.5s	632.9s
1 ÷ 2	1964.6s	336.0s
1 ÷ 4	1151.8s	181.6s
2 ÷ 8	600.5s	83.2s
4 ÷ 16	317.1s	40.7s

Table 4.1: DPSNN runtimes.

Table 4.1 reports the simulation run-times of a reference configuration (5000 simulated milliseconds of an 8×8 grid of cortical columns, 1250 neurons per column) for different layouts of cores and cluster nodes, compared against those of a standard HPC platform (nodes are dual-socket servers populated with Intel CPUs — Xeon E5-2630 v2 clocked at 2.6 GHz) interconnected with an InfiniBand network interface. The speedup plot is depicted in Figure 4.1.

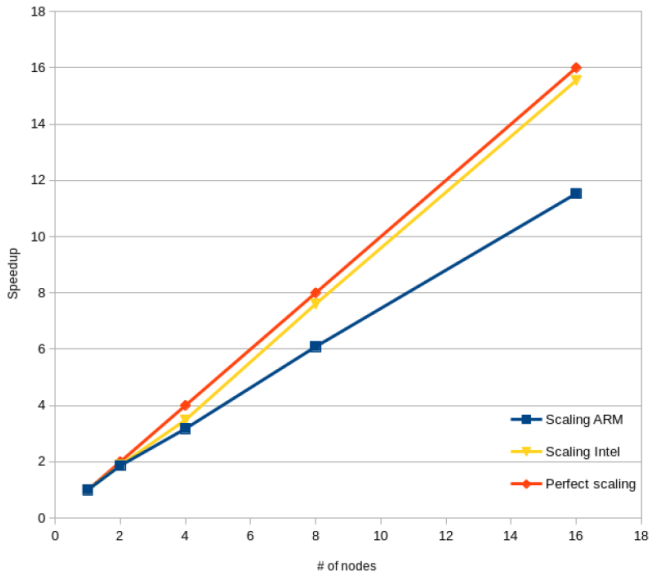

Figure 4.1: A comparison of DPSNN scaling on Intel- and Arm-based system.

4.3 Mini-application benchmarking tool

Evaluation of HPC hardware is a key element especially in the first stages of a project — *i.e.* definition of specification and design — and during the development and implementation. Features impacting performance should be identified in the analysis and design of new architectures. In the early stages of the development, full applications are too complex to run on the hardware prototype. In usual practice, hardware is tested with very simple kernels and

benchmarking tools which often reveal their inadequacy as soon as they are compared with real applications running on the final platform, showing a huge performance gap.

In the last years, a new category of compact, self-contained proxies for real applications called *mini-apps* has appeared. Although a full application is usually composed by a huge amount of code, the overall behaviour is driven by a relatively small subset. Mini-apps are composed by these core operations providing a tool to study different subjects: (i) analysis of the computing device — *i.e.* the node of the system; (ii) evaluation of scaling capabilities, configuring the mini-apps to run on different number of nodes, and (iii) study of the memory usage and the effective throughput towards the memory.

This effort is led by the Mantevo project [87], that provides application performance proxies since 2009. Furthermore, the main research computing centers provide sets of mini-applications, adopted when procuring the systems, as in the case of the NERSC-8/Trinity Benchmarks [88], used to assess the performance of the Cray XC30 architecture, or the Fiber Miniapp Suite [89], developed by RIKEN Advanced Institute for Computational Science (RIKEN AICS) and the Tokyo Institute of Technology.

4.3.1 miniDPSNN

The miniDPSNN benchmarking tool leverages the Hardware-Software Co-design approach that starts from the collection of application requirements for the initial development of the infrastracture and then pursues the testing of the adopted solution during the implementation. Thus, the application drives the research about the main components of a HPC system from its roots, by optimizing modeling and simulation of a complex physical system.

The analysis is based on the behaviour of a strong scaling test. Neurons are arranged into "columns", each one composed by about one thousand neurons; columns are then arranged into a bidimensional grid. Each excitatory neuron projects 80% of its synapses towards neurons residing in its own column, while the rest reache out to those in the neighbouring columns, according to the chosen remote connectivity, *i.e.* gaussian or exponential, as explained in Section 3.3.1. Instead, synapses of inhibitory neurons are projected only towards excitatory ones residing in their same column. When DPSNN runs, each process can either host a fraction of a column, a whole single column, or an integer number of columns.

Each core of the computing system hosts only one process optimizing the performance. Thus, the varying of the columns-per-process ratio — *i.e.* ratio of columns per core of the computing devices — throttles the application into different regimes, allowing to stress and test several elements of the platform. Be noted that in general, the hardware connection topology bears no resemblance whatsoever with the lateral connectivity of columns and neu-

rons, the exception being when running only one process per node, so that all outwards connectivity of a column impinges upon the network system of the node (see Figure 4.2).

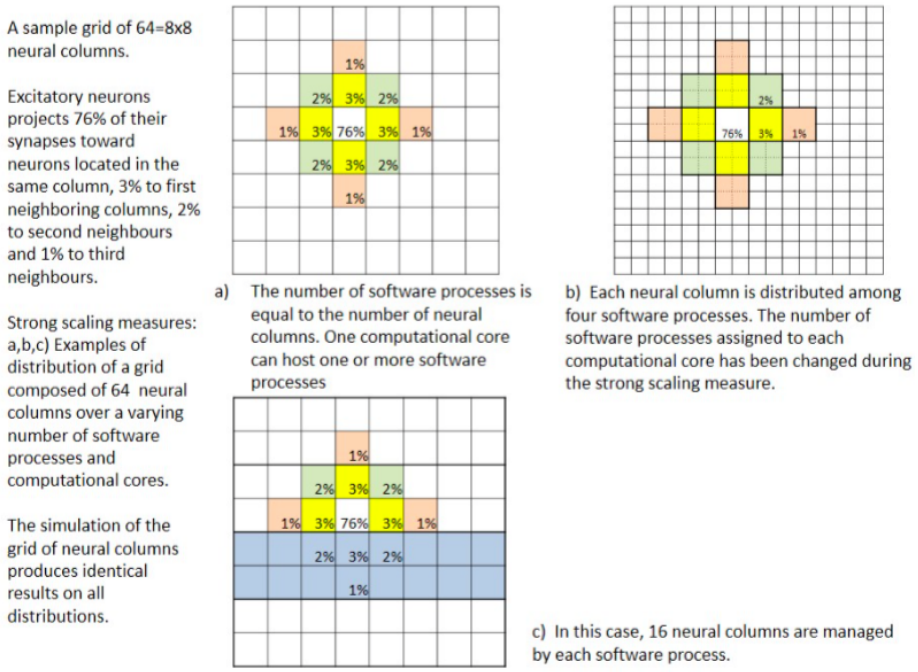

Figure 4.2: Columns/Process Distribution.

Here is a rundown of the application tasks that miniDPSNN performs and that allow to gauge the components of the architecture under test:

- **Computation**: processing of the time step in the dynamical evolution of the neuron.
- **Memory Management**: management of either axonal spikes organized in time delay queues and lists of synaptic spikes, both stored in memory.
- **Communication**: transmission along the interconnect system of the axonal spikes to the subset of processes where target neurons exist.
- **Synchronization**: at each time step, the processes deliver the spikes produced by the dynamics according to the internal connectivity supported by the synaptic configuration. This global exchange is currently implemented by means of synchronous MPI collectives; any offset in time when different processes reach these waypoints — whether it be by fluctuations in load or network congestion — causes idling cores and diminished parallelization.

Table 4.2 displays results obtained running on a standard HPC cluster based on Intel Xeon processors communicating over an InfiniBand interconnect, as a function of the configuration of the testbed — *i.e.* grid size, simulated seconds, allocated cores. The distribution of tasks is strongly dependent on the columns-per-core ratio. As already stated, the computation task becomes more demanding when increasing the number of columns per node — which means increasing the total number of neurons. Instead, reducing the columns-per-core ratio generates relatively more communication among processes, moving the focus of the test to the interconnect.

	12×12	24×24	48×48
Grid	0.18 M	0.71 M	2.86 M
Synapses	0.20 G	0.80 G	3.20 G
Columns	144	192	192
Columns/Core	**1**	**3**	**12**
Simulated Seconds	30	12	18
Wall-clock Seconds	1484	2148	15182
Computation	21.3%	34.2%	45.1%
Memory Management	17.1%	16.7%	16.9%
Communication	35.2%	10.7%	0.9%
Synchronization	22.9%	36.3%	36.2%

Table 4.2: miniDPSNN tasks overview.

4.3.2 miniDPSNN analysis of low-power and standard computing architectures in the real-time domain

In this domain, being "real-time" signifies a miniDPSNN workpoint such that the execution time — *i.e.* wall-clock time of the running application — is not greater than the simulated time. Accomplishment of this workpoint is obtained through an accurate configuration of parameters. Prelimary trials of DPSNN keeping pace with this real-time requirement are reported in this section. This working condition could be useful in the robotics application field.

The testbed is a standard strong scaling test of a 4×4 columns grid. Figure 4.3 shows the results of the test obtained simulating 10 s on the Intel-based platform.

Up to $8 \div 16$ cores, the architecture scales well, decreasing the execution time down to ~ 12 seconds. The execution time increases unexpectedly (~ 16 seconds) when distributing the problem over 32 cores, thus preventing the achievement of the target workpoint.

Singling out the times of the various tasks as reported in Figure 4.4 sheds some light on this behaviour. The communication quickly becomes more demanding when the problem is split over more than 16 processes, dominating the behaviour of the application. As

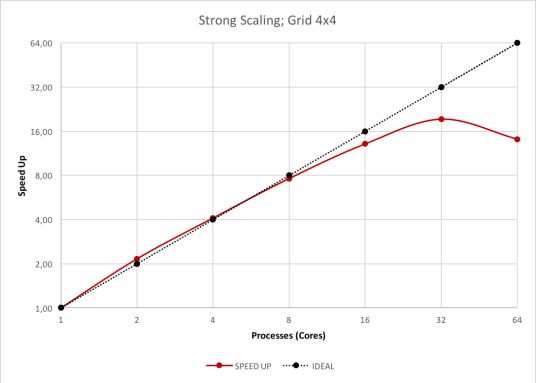

Figure 4.3: Strong scaling of a 4×4 column grid simulated on an Intel-based platform equipped with IB interconnect.

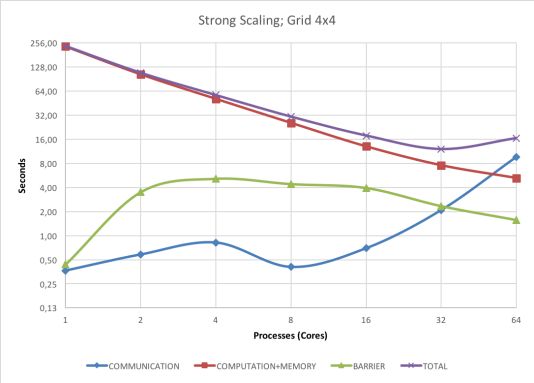

Figure 4.4: miniDPSNN analysis of the Intel-based platform.

mentioned before, the application stresses the interconnect when the column-per-core ratio decreases — a whole column or portions of column are managed by each core in the tested configuration. More than 80% of synapses remain within the column their projecting neuron belongs to. Communication between processes increases when the columns are split among them, clogging the network with an ever increasing number of small packets. The miniDPSNN highligths this "latency" limitation of the IB interconnect provided by the cluster. In general, COTS interconnects offer adequate throughput when moving large amounts of data, but tipically trudge when the communication is latency-dominated. This issue with communication — manifesting here with a number of computing cores which is, by today's standards, not large — is similar to that encountered by the parallel cortical simulator C2 [90] — targeting a scale in excess of that of the cat cortex — on the Dawn Blue Gene/P supercomputer at LLNL, with 147456 CPUs and 144 TB of main memory. The capability to replicate the behaviour of a supercomputer with a mini-app running on a limited number of 1U servers could be considered the proof of its effectiveness.

Similar results are obtained performing the same test on an ARM-based platform as showed in Figure 4.5 and Figure 4.6, although the analysis is limited by the available number of cores (16). All four nodes are connected together through a 1 Gbps Ethernet-based network.

The number of transmitted packets increases distributing the same problem over an increasing number of processes (core) as shown in Figure 4.7 while the payload generated by each process does not vary as shown in Figure 4.8 and the communication becomes more demanding.

Furthermore, the size of packets decreases (see Figure 4.9); the mean packet size is ~ 40 bytes when each core simulates the dynamics of a single column, as depicted in Fig-

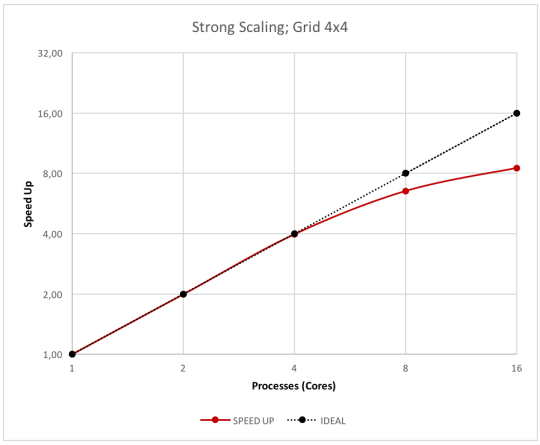

Figure 4.5: Strong scaling of a 4×4 column grid simulated on an ARM-based platform equipped with GbE interconnect.

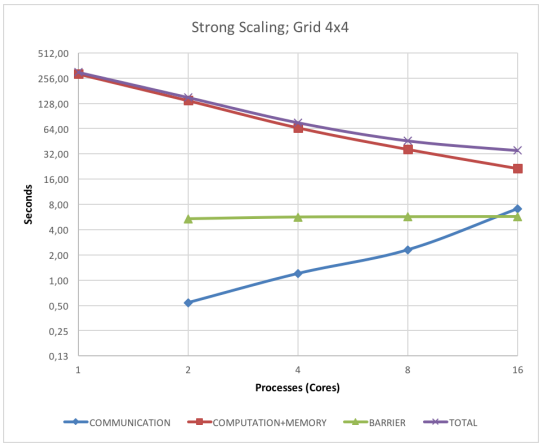

Figure 4.6: miniDPSNN analysis of the ARM-based platform.

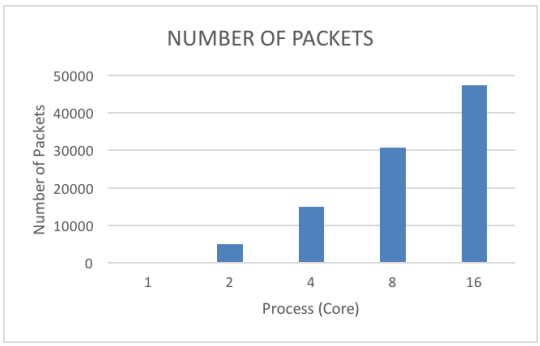

Figure 4.7: Packets generated during the simulation.

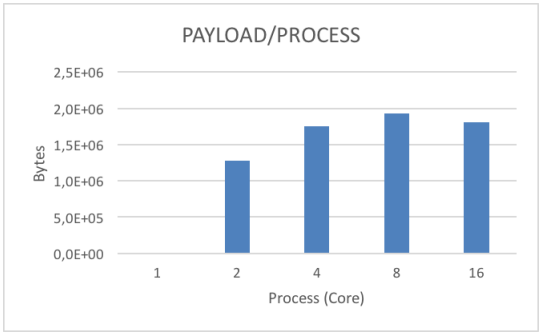

Figure 4.8: Payload generated by each process.

ure 4.10.

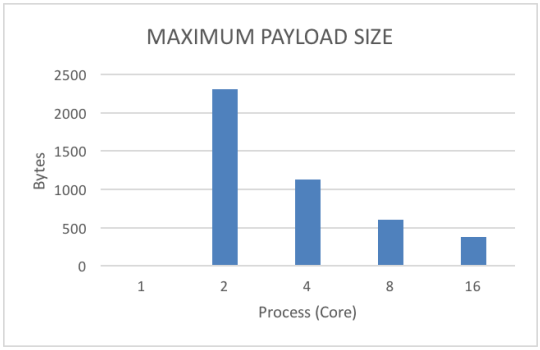

Figure 4.9: Maximum packet size produced the DPSNN simulation.

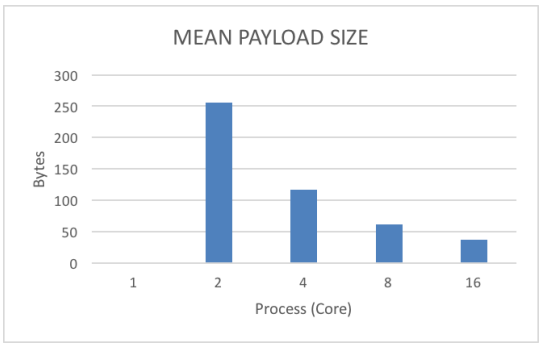

Figure 4.10: Mean packet size produced the DPSNN simulation.

The characterization of the traffic generated by the DPSNN over several off-the-shelf

interconnects allows to identify the main requirement for a network device of future exascale computing system simulating spiking neural network simulation: the network system should be optimized for the trasmission of small packets. In particular, performances are strongly influenced by (i) the design and implementation of a low-latency interconnect architecture, and (ii) the definition of a light and reliable communication protocol guaranteeing high throughput and optimizing the transfers of data packets with payload < 512 Bytes.

Finally a planned re-engineering of the DPSNN foresees a two-level hierarchy enforced via MPI communicators: one auxiliary process (called "broker") is added per node and communications are segregated to be only among processes belonging o the same node – *i.e.* exchanges that go only through intra-node, shared-memory channels – or among brokers – *i.e.* exchanges that only go through inter-node, remote interfaces. In this way, "local" exchanges among neighbouring neural columns (which, given the biologically plausible topology for the synaptic connectivity, make up the exchange bulk) can be contained to the fastest and possibly less congested intra-node channel while "distal" exchanges are gathered to the broker process of the node, then scattered to brokers of other nodes that take care of scattering them to the appropriate recipients.

4.3.3 Energy-to-Solution analysis

Instantaneous power, total energy consumption, execution time and energetic cost per synaptic event of a spiking neural network simulator distributed on MPI processes are compared when executed on different generations of low-power and traditional computing architecture to have a (limited) estimate of the trend.

The power and energy consumption reported were obtained simulating 3 s of activity of a network made of 18 M equivalent (internal + external) synapses: the network includes 10 K neurons (Leaky Integrate-and-Fire with Calcium-mediated spike-frequency adaptation), each one projecting an average of 1195 internal synapses and receiving an "external" stimulus, corresponding to 594 equivalent external synapses/neuron. A Poissonian spike train targets external synapses with an average rate of 3 Hz; synaptic plasticity is disabled. In response, the neurons fire trains of spikes at a mean rate of 5.1 Hz.

The power measurement equipment consists of a DC power supply, a high-precision Tektronix DMM4050 digital multimeter for DC current measurements connected to National Instruments data logging software and a high-precision AC power meter. The AC power of the high-end server node is measured by a Voltech PM300 Power Analyzer upstream of the main server power supply (measuring on the AC cable). For the SoCs, the DC current was instead sampled downstream of the power supply. Such difference should not affect

significantly the results, given the closeness to one of the $\cos\varphi$ factor of the server power supply.

First Generation Comparison

The traditional computing system — *i.e.* "server platform" — is based on a SuperMicro X8DTG-D 1U dual-socket server housing 8 computing cores residing on quad-core Intel Xeon CPUs (Westmere E5620@2.4 GHz in 32 nm CMOS technology). TThis "server platform" is juxtaposed to the "embedded platform": two NVIDIA Jetson TK1 boards, connected by an Ethernet 100 Mb mini-switch to emulate a dual-socket node, each board equipped with a NVIDIA Tegra K1 chip, *i.e.* a quad-core ARM Cortex-A15@2.3 GHz in 28 nm CMOS technology.

The "server platform" has 48 GB of DDR3 memory on-board, operating at 1333 MHz — 6 GB per core — while the "embedded platform" only has 2 GB running at 933 MHz — 0.5 GB per core. This makes for a considerable difference in terms of memory bandwidth — 14.9 GB/s for the ARM-based system against the 25.6 GB/s of the Intel-based one — which has an impact on DPSNN and its intensive memory usage, *e.g.* for delivering spikes to post-synaptic neuron queues.

Partitioning the neural grid onto 8 MPI processes, the simulation of 3 s of activity required 9.1 s on the "server platform" and 30 s on the "embedded platform", as shown in Figure 4.11.

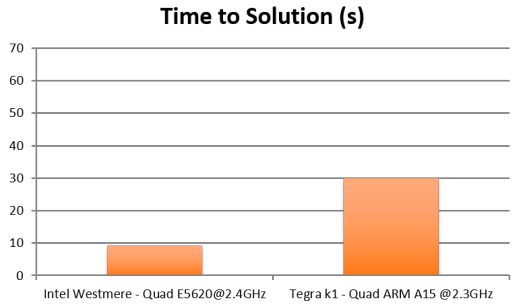

Figure 4.11: First generation time-to-solution result.

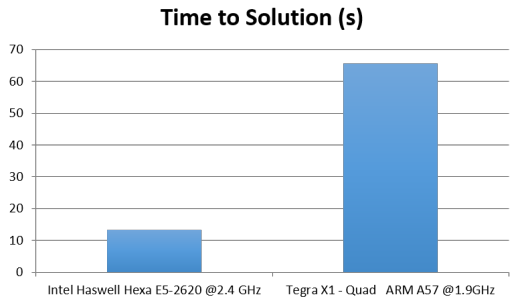

Figure 4.12: Second generation time-to-solution result. Note that the number of cores used in the first generation was the double of that used in this case.

Observed currents were $I_s = 1.15$ A ("server") and $I_e = 80$ mA ("embedded"), with 5 mA measure error. Therefore, the energies required to complete the same task on the two architectures were $E_s = 2.3$ KJ and $E_e = 528$ J (see Figure 4.15), while the observed instantaneous power consumptions were $P_s = 253$ W and $P_e = 17.6$ W (see Figure 4.13). Note

that we did not subtract any "base-line" power — *e.g.* power consumption after bootstrap, so the estimate is "pessimistic" in the sense that it includes the load of the complete system runnning.

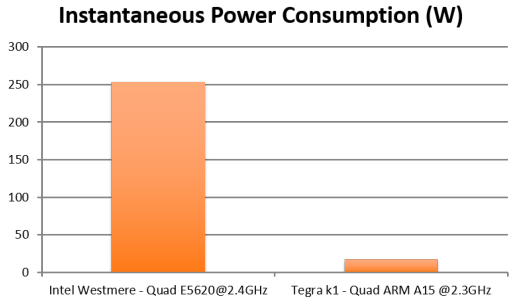

Figure 4.13: First generation power-to-solution result.

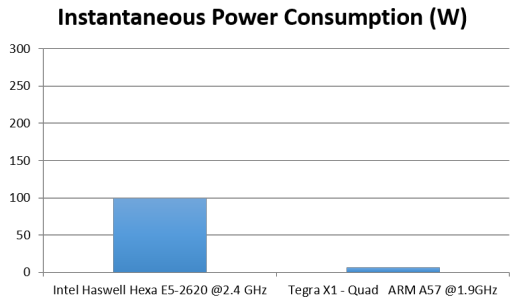

Figure 4.14: Second generation power-to-solution result. Note that the number of cores used in the first generation was the double of that used in this case.

The simulation produced a total of 235 M synaptic events: the total energetic cost of simulation can be estimated in 2.2 μJ/synaptic event on the "embedded platform" node and 9.8 μJ/synaptic event for the "server platform". The "server platform" dual-socket node is faster, spending 3.3 times less time than the "embedded platform" node. However, the "embedded platform" node consumes a total energy 4.4 times lower to complete the simulation task, with an instantaneous power consumption 14.4 times lower than the "server platform" node.

The energetic cost of the optimized Compass simulator of the TrueNorth ASIC-based platform, run on an Intel Core i7 CPU 950@3.07 GHz (45 nm CMOS process) with 4 cores and 8 threads, is 5.7 μJ/synaptic event, but excludes a significant base-line power consumption. Applying the same normalization, the results of the "embedded platform" are reduced of a factor $2 \div 4$ for the "server platform".

Second Generation Comparison

The performances are measured in executing the DPSNN code along with those of a coeval mainstream Intel processor architecture using a hardware/software configuration suitable to extrapolate a direct comparison of time-to-solution and energy-to-solution at the level of the single core. The measures are extended to the new generation NVIDIA Jetson TX1 SoC based on the ARMv8 architecture. The Jetson TX1 includes four ARM Cortex-A57 cores plus four ARM Cortex-A53 cores in big.LITTLE configuration.

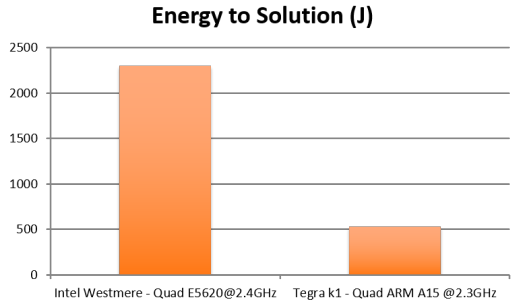

Figure 4.15: First generation energy-to-solution

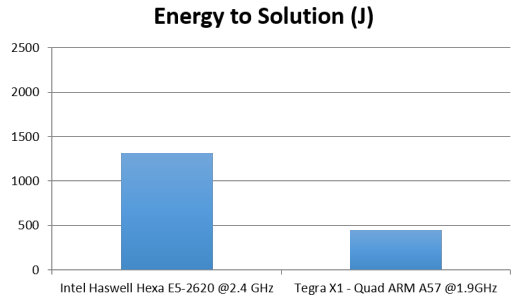

Figure 4.16: Second generation energy-to-solution result.

The "server platform" is a Supermicro SuperServer 7048GR-TR with two hexa-core Intel Haswell E5-2620 v3 @2.40 GHz. Four MPI processes are run on either platform, simulating 3 s of the dynamics of a network made of 10^4 Leaky Integrate-and-Fire with Calcium Adaptation (LIFCA) neurons connected via 18×10^6 synapses. Results are shown in Figure 4.12, Figure 4.14 and Figure 4.16 and can be summarized as follows: Although the x86 architecture is about $5\times$ faster than the ARM Cortex-A57 core in executing the simulation, the energy it consumes in doing so is $\sim 3\times$ higher [91].

5

APEnet race towards Exascale

Contents

5.1 Introduction

The Array Processor Experiment (APE) is a custom design for HPC, started by the Istituto Nazionale di Fisica Nucleare (INFN) and partnered by a number of physics institutions all over the world; since its start in 1984, it has developed four generations of custom machines (APE [92], ape100 [93], APEmille [94] and apeNEXT [95]). Leveraging the acquired know-how in networking and re-employing the gained insights, a spin-off project called APEnet developed an interconnect board based on FPGA that allows to assemble a PC cluster *à la* APE with off-the-shelf components.

The design of APEnet interconnect is easily portable and can be configured for different environments: (i) the APEnet [96] was the first point-to-point, low-latency, high-throughput network interface card for LQCD dedicated clusters; (ii) the Distributed Network Processor [97] (DNP) was one of the key elements of RDT (Risc+DSP+DNP) chip for the implementation of a tiled architecture in the framework of the EU FP6 SHAPES project [98]; (iii) the APEnet Network Interface Card, based on an Altera Stratix IV FPGA, was used in a hybrid, GPU-accelerated x86_64 cluster QUonG [99] with a 3D toroidal mesh topology, able to scale up to $10^4 \div 10^5$ nodes in the framework of the EU FP7 EURETILE project. APEnet+ was the first device to directly access the memory of the NVIDIA GPU providing GPUDirect RDMA capabilities and experiencing a boost in GPU to GPU latency test; (iv) the APEnet network IP — *i.e.* routing logic and link controller — is responsible for data transmision at Tier 0/1/2 in the framework of H2020 ExaNeSt project, as shown in Chapter 2.

Table 5.1 summarizes the APEnet families comparing the main features.

	APEnet	**DNP**	**APEnet+**	**APEnet+ V5**	**ExaNet**
Year	2003	2007	2012	2014	2017
FPGA	Altera Stratix III	ASIC	Altera Stratix IV	Altera Stratix V	Xilinx Ultrascale+
BUS	PCI-X	AMBA-AHB	PCIe gen2	PCIe gen3	AXI
Computing	Intel CPU	RISC+DSP	NVIDIA GPU	NVIDIA GPU	ARM+FPGA
Bandwidth	6.4 Gbps		34 Gbps	45 Gbps	32 Gbps
Latency	6.5μs		4μs	5μs	1.1μs

Table 5.1: The APEnet roadmap to Exascale

In Section 5.2, the main elements of the APEnet interconnect architecture are described. The last generation is presented in Section 5.3 and the testbed and performance achieved are shown in Section 5.3.

5.2 APEnet Interconnect architecture

The APEnet interconnect has at its core the DNP acting as an offloading network engine for the computing node, performing internode data transfers; the DNP has been developed as a parametric Intellectual Property library; there is a degree of freedom in choosing some fundamental architectural features, while others can be customized at design-time and new ones can be easily added. The APEnet architecture is based on a layer models, as shown in Figure 5.1, including physical, data-link, network, and transport layers of the OSI model.

The physical layer — **APEphy** — defines the data encoding scheme for the serialization of the messages over the cable and shapes the network topology. APEphy provides point-to-point bidirectional, full-duplex communication channels of each node with its

neighbours along the available directions (*i.e.* the connectors composing the IO interface). APEphy is strictly dependent on the embedded transceiver system provided by the available FPGA. It is normally based on a customization of tools provided by the FPGA vendor — *i.e.* DC-balance encoding scheme, deskewing, alignment mechanism, byte ordering, equalization, channel bonding. In APEnet+ and APEnet+ V5, four bidirectional lanes, bonded into a single channel with usual 8b10b encoding, DC-balancing at transmitter side and byte ordering mechanisms at receiver side, allow to achieve the target bandwidth (34 Gbps [100] and 45 Gbps [101] respectively).

The data-link layer — **APElink** — establishes the logical link between nodes and guarantees reliable communication, performing error detections and corrections. APElink [102] is the INFN proprietary high-throughput, low-latency data transmission protocol for direct network interconnect based on word-stuffing technique, meaning that the data transmission needs submission of a magic word every time a control frame is dispatched to distinguish it from data frames. The APElink manages the frame flow by encapsulating the packets into a light, low-level protocol. Further, it manages the flow of control messages for the upper layers describing the status of the node (*i.e.* health status and buffer occupancy), and transmitted through the APElink protocol.

The network layer — **APErouter** — defines the switching technique and routing algorithm. The Routing and Arbitration Logic manages dynamic links between blocks connected to the switch. The APErouter applies a dimension order routing [103] (DOR) policy: it consists in reducing to zero the offset between current and destination node coordinates along one dimension before considering the offset in the next dimension. The employed switching technique — *i.e.* when and how messages are transferred along the paths established by the routing algorithm, *de facto* managing the data flow — is Virtual Cut-Through [104] (VCT): the router starts forwarding the packet as soon as the algorithm has picked a direction and the buffer used to store the packet has enough space. The deadlock-avoidance of DOR routing is guaranteed by the implementation of two virtual channels [105] for each physical channel.

The transport layer — **APE Network Interface** — defines end-to-end protocols and the APEpacket. The APE Network Interface block has basically two main tasks: on the transmit data path, it gathers data coming in from the bus interfacing the programming subsystem, fragmenting the data stream into packets — APEpacket— which are forwarded to the relevant destination ports, depending on the requested operation; on the receive side, it implements PUT and GET semantics providing hardware support for the RDMA (Remote Direct Memory Access) protocol that allows to transfer data over the network without explicit support from the remote node's CPU.

The full APE Network Interface offers a register-based space for configuration and status

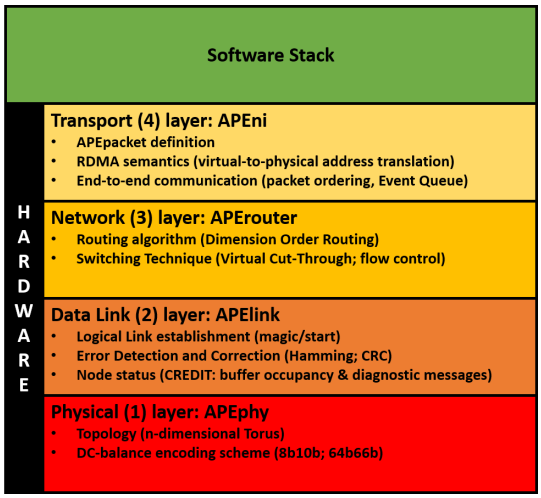

Figure 5.1: The layered architecture of APEnet

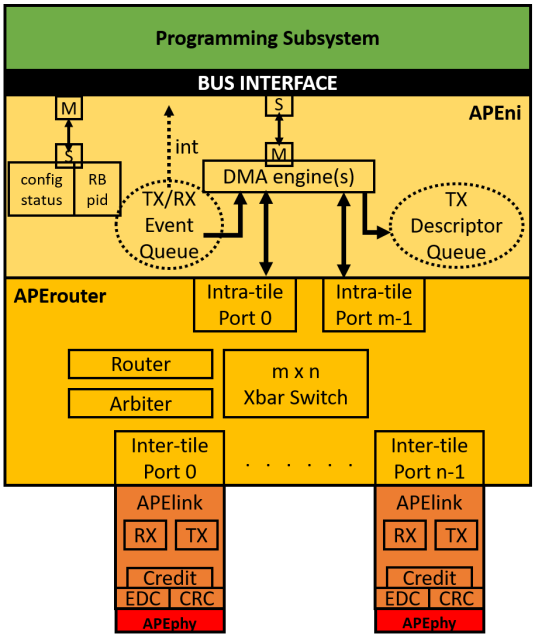

Figure 5.2: A block diagram of APEnet architecture

signalling towards the host. Further, it offers a variable size region for defining a number of ring buffers, each one linked to an OS process accessing the device. These regions are typically accessed in slave mode by the host, which is master (read/write of single 32-bit based registers). A certain number of DMA engines are used to move data to and from the device, plus other additional services: a TX descriptor queue (to issue buffer transfers from host to device) and an event queue (to notify different kind of completions to host). Single or Multiple DMA engines could manage the same intra-tile port.

The block diagram of the APEnet interconnect architecture is shown in Figure 5.2.

5.3 ExaNet

ExaNet is responsible for data communication at Tier 0/1/2 of the network interconnect of the ExaNeSt project. ExaNet is the product of a joint collaboration among the Foundation for Research and Technology [106] (FORTH) in Greece and the Istituto Nazionale di Fisica Nucleare (INFN) in Italy. The INFN APE research group [32] is responsible for the ExaNet Network IP providing switching and routing features and managing the communications over the High Speed Serial (HSS) links through different levels of the interconnect hierarchy: (i) the high-throughput intra-QFDB level (Tier 0) for data transmission among the four FPGAs of the ExaNeSt node; (ii) the intra-Mezzanine level (Tier 1) directly connecting

the network FPGAs of different nodes within the same mezzanine and (iii) inter-Mezzanine communication level (Tier 2) managing the connectivity of the Mezzanine based on SFP+ connectors and allowing for the implementation of a direct network among QFDBs within a chassis. The ExaNet Network IP mainly consists of two hardware components: (i) the APErouter, handling the routing and switching mechanism of the network IP as described in Section 5.3.3; (ii) the APElink I/O interface, managing the data transfers over the HSS links as reported in Section 5.3.4.

5.3.1 ExaNet development platform

The Trenz TEBF0808 system has been used since it features the same Xilinx Ultrascale+ MPSoC FPGA family chosen for the final prototype (XCZU9EG), being the early stages of the ExaNeSt system prototype so that the node is still under development. Preliminary tests were performed to validate the network, connecting up to four boards shaping a 2×2 mesh topology through the two SFP+ connectors available on each system (see Figure 5.3 and 5.4).

Figure 5.3: The ExaNet development platform shaping 2×2 topology.

Figure 5.4: The SFP+ connectors provided by the Trenz Board.

The testbed allows to validate the adoption of the APEnet architecture at both Tier 0 and Tier 1. The QFDB composed by four FPGAs matches perfectly with the testing platform. Furthermore, the development platform emulates the communication among the four network FPGAs of the QFDBs hosted within the track-1 mezzanine.

5.3.2 Packet structure

A modified version of the APEpacket is the data structure of the ExaNet communication system based on the latest generation of the APEnet protocol. Figure 5.5 outlines the ExaNet packet.

The packet is composed by a 128-bit *header*, a 128-bit *footer* and a *payload*. The maximum payload size is $256 \div 512$ bytes, being a good compromise between bandwidth per-

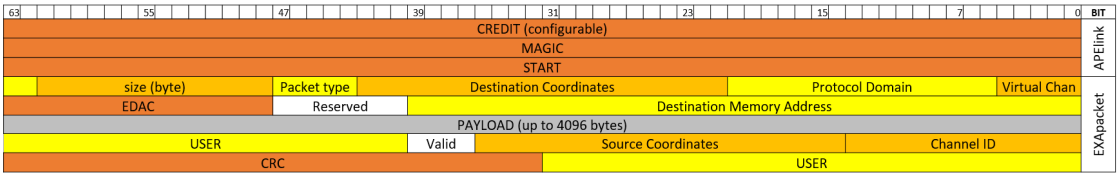

Figure 5.5: Format examples of packet and APElink protocol of the ExaNet interconnect

formances and routing efficiency (avoiding delay of high priority packet). Nevertheless, APElink protocol supports the size up to 4 KB, equal to the size of a page of a standard GNU/LINUX system.

The header contains the information to route packets to their proper destination:

- the Virtual Channels are used to avoid routing deadlocks and to prioritize packets;
- the Protection Domain ID carries the identifier of a system-level process group;
- the Destination Coordinates identifies each unit (FPGA) of the system;
- the Type field identifies different packets with specific encoding of header and footer fields or without payload;
- the Size field identifies the payload length;
- the Destination Memory Address specifies the (virtual) address at the target node;
- the Header Error Detection and Correction Code is used to avoid traffic generated by misrouted packets.

The footer encodes the following information:

- the Channel ID field identifies an outstanding packet that has been issued from a source node;
- the Source Coordinate field is used for the end-to-end packet acknowledgment and retransmission mechanism.
- The Valid field identifies the last valid byte within the last payload word;
- The User-Defined field is left for specific applications;
- The CRC field validates the content of the transmitted message.

5.3.3 APErouter

The APErouter block (see Figure 5.6) dynamically interconnects the intra-tile ports — *i.e.* the interface between the programming logic and the programming subsystem (see Figure 2.3) — and the inter-tile ports — *i.e.* the I/O interface with the other nodes — and comprises a fully connected switch, plus routing and arbitration blocks. Together with the Arbiter, the Routing Logic manages dynamic links between blocks connected to the switch; depending on the final destination, the router decides which switch port the packet must be delivered to among the possible ports, and the arbiter solves the contention between packets requiring the same resources.

APErouter hardware IP

The latest release of APErouter is the result of a porting and adaptation activity to make the IP synthesizable on Xilinx FPGAs. Furthermore, the IP is compliant with the new ExaNet Header format and manages different kinds of packet (`RDMA`, `ACK`, `RATE`). The current IP supports the byte alignment of the data structure in the memory – the previous releases were word aligned. Finally, the design has been arranged to be more flexible, in order to simplify the support of new network topologies and the introduction of important new features for HPC – adaptive routing algorithm and collective hardware acceleration.

Although the basic functionalities of the APErouter were verified in the past — on the QUonG prototype located in Rome — correct operation of the new release is currently tested on the mini-cluster composed by Trenz boards, arranged into a 2×2 mesh described in Section 5.3.1. Single and multiple hop tests have been performed and the results are shown in Section 5.4 and Section 5.4.2 The proper behaviour of the routing algorithm on a larger cluster is proved through the custom simulator described in Section 5.5.

The block diagram of the APErouter is depicted in Figure 5.6. The main components are briefly described:

- **Switch Port**: it contains transmitting (TX) and receiving (RX) FIFOs. The FIFOs are in show-ahead mode in the current release. Intra-tile ports are configurable and implemented in TX/RX mode or TX mode only. The header/footer FIFOs are 128×128 bit (2 KB), while the intra-tile and inter-tile payload FIFO are 4096×128 bit (64 KB) and 1024×128 bit (16 KB) respectively.

- **Switch Gate**: it connects data and control signals coming from the intra-tile and inter-tile ports with the crossbar. It manages the data flow preventing FIFOs overflow and guarantees proper transmission of the packet — header, payload, footer. The

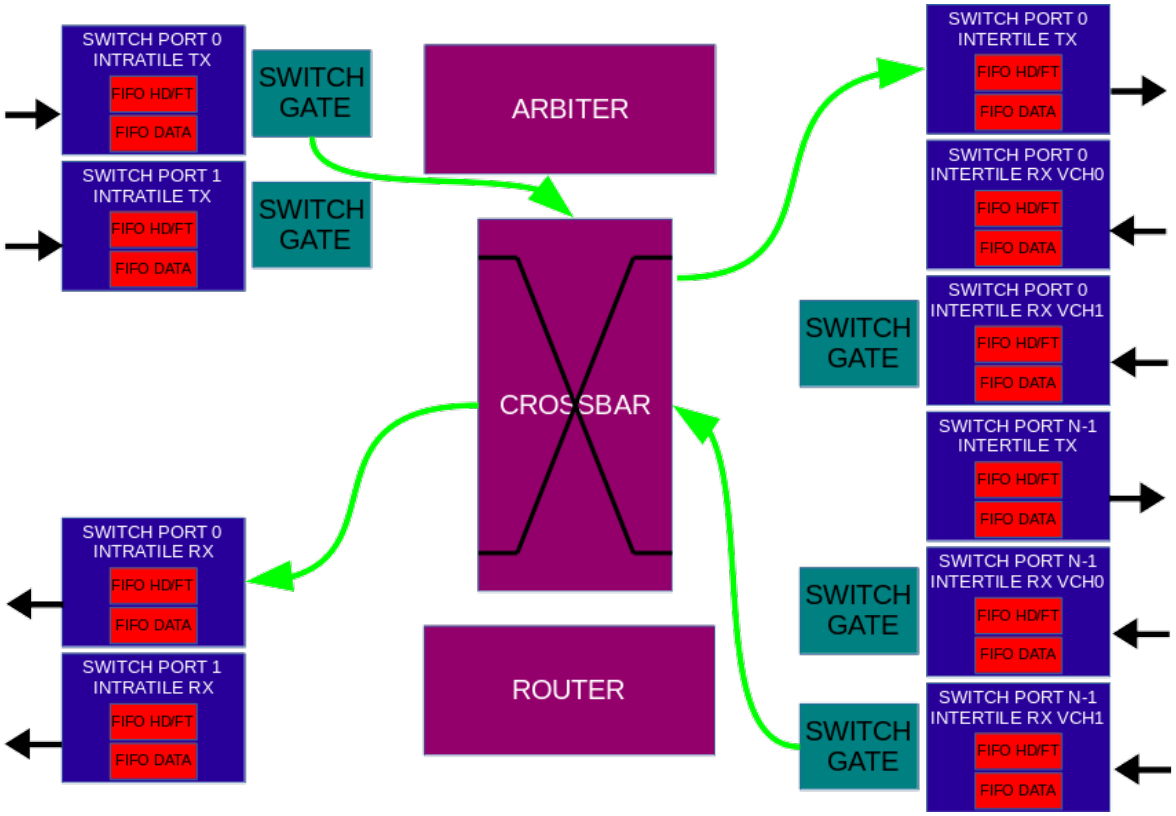

Figure 5.6: The block diagram of the APErouter on ExaNet prototype.

IP can be configured – as a synthesis parameter, not in run-time – to manage a packet length encoded either in words or in bytes. This latter functionality needs additional information about byte valid of the first payload word – header and footer are always assumed LSB aligned.

- **Router**: as already discussed in Section 5.2, APEnet applies a deterministic Dimension-Ordered Routing (DOR) policy: it consists in reducing to zero the offset between current and destination node coordinates along one dimension before considering the offset in the next dimension. The APEnet DOR router is able to handle more than one packet transaction. Specialized priority registers – writable at run-time — allow selecting the coordinates evaluation order (*i.e.* first Z is consumed, then Y and finally X) and disabling ports altogether. The employed switching technique, *i.e.* when and how messages are transferred along the paths established by the routing algorithm, is Virtual Cut-Through (VCT): the router starts forwarding the packet as soon as the algorithm has picked a direction and the buffer used to store the packet has enough space. The DOR algorithm is not *per se* deadlock-free, but deadlock-avoidance can be enforced by the implementation of two virtual channels for each physical channel.

The choice of the virtual channel is another important task for the router: it sends packets using the upper virtual channel if the offset between current and destination node is greater than zero, the lower virtual channel otherwise. The new routing function removes all the cyclic dependencies in the channel dependency graph (CDG), thus ensuring deadlock avoidance.

- **Arbiter**: it manages conflicts among the requests — packets coming from different ports could request the same destination port. The scheduling algorithm is configurable: Round Robin or Fixed Priority – the latter can be modified at run-time writing the proper configuration register.

The meaningful occupancy values of the main components of a 3×2 APErouter are shown in Table 5.2.

IP	LUT	LUT RAM	LUT FF	Registers	BRAM	GTH
Available Resources	274080	144000	274080	548160	912	16
APErouter 3×2	9599 (3.5%)	0	3162 (1.2%)	7649 (1.2%)	116.5 (12.7%)	0
Intra-tile Switch Gate	194	0	142	250	0	0
Inter-tile Switch Gate	300	0	152	256	0	0
Arbiter 3×2	2067	0	73	142	0	0
Router 3×2	424	0	77	1548	0	0
Intra-tile Switch Port	852	0	300	598	28.2	0
Inter-tile Switch Port	1285	0	312	702	16	0

Table 5.2: Overview of the APErouter hardware resources.

The current operating frequency of the APErouter is 156.25 MHz and the power consumption is 0.271 W according to the Xilinx estimation tool. The result is strongly dependent on the number of intra- and inter-tile ports provided, as shown in Figure 5.7.

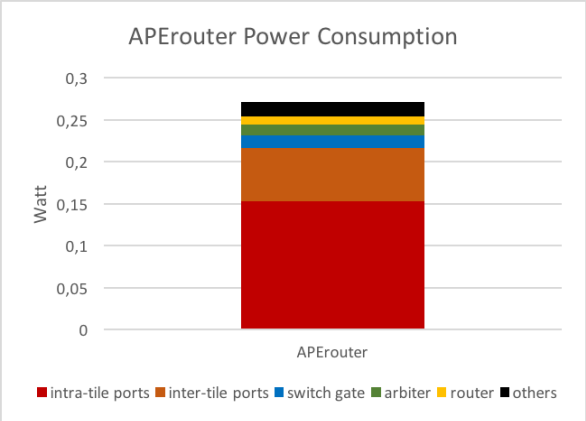

Figure 5.7: APErouter power consumption.

The results are influenced by the dimension of the FIFOs and the number of the implemented virtual channels. Although a fine tuning should reduce the power consumption, the achieved result is encouraging.

APErouter performance

The latency introduced by the APErouter— *i.e.* from the arrival of the header in intra-tile TX FIFO to the writing of the header in the inter-tile TX FIFO — is shown in Figure 5.8.

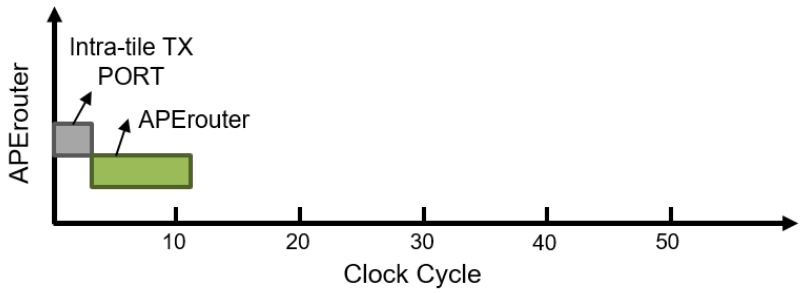

Figure 5.8: Intra-tile TX towards inter-tile TX latency.

The number of clock cycles is equal to 11 (about 70 ns at the current operating frequency of 156.25 MHz). The performance is the same for the following path: (i) *loopback*: intra-tile TX towards intra-tile RX; (ii) *send*: intra-tile TX towards inter-tile TX; (iii) *receive*: inter-tile RX towards intra-tile RX and (iv) *intermediate node*: inter-tile RX towards inter-tile TX.

5.3.4 APElink

The data packet is encapsulated in a lightweight protocol sketched in Figure 5.5. Two words – Magic/Start – are included into the data flow over the serial links to establish the logical link between nodes. Magic and Start width is always equal to the transceiver bus width, thus their transmission over the links takes 2 clock cycles only.

The Magic/Start sequence announces the transmission of the Header. The width of the header and the information stored in it are totally parametric in order to match the requirements of the framework.

Since misrouted packets are disruptive for the network, the highly critical header integrity is protected by an Error Correction Code (ECC). However, unrecoverable errors can still occur in the transmitted data flow. Necessarily, the entire packet is considered corrupted, when a faulty (unrecoverable) header is received. The data are not flagged and are collected following the logical sequence of the data stream, trusting the “Size” field stored in the header.

An “emergency” strategy is defined to manage the critical status and to avoid the stop/restart procedure of the data transmission and minimizing the loss of information. The received faulty packets are discarded until a Start is identified in the received stream — only the faulty packet is loss — guaranteeing the recovery of the logical alignment between the peers of the communication. APElink does not provide any acknowledgement or retransmission mechanism, to not affect performance of the transmission and to not force the implementation of additional memory buffers. These mechanisms are provided at a higher level of the networking framework (end-to-end communication managed by the transport layer).

Buffer availability is measured by credit; exchanging credits by two communicating nodes is mandatory to avoid buffer overflow. Outbound words consume it, causing transmission suspension as soon as a programmable credit threshold (TRED) is reached — *i.e.* credit is exhausted — and resuming as soon as info about newly available space bounces back to the transmitter — *i.e.* credit is eventually restored. This information is exploited by the router to manage the data flow implementing the VCT switching mechanism. Credit content is configurable according to the architectural choices: (i) **start/stop** commands for a latency-optimized interconnect — a 64-bit credit word is able to manage up to 8 virtual channels limiting the amount of transmitted extra bits; (ii) **usedword** value reporting the occupancy of the receiving buffer. The efficiency of the protocol is clearly affected (only two virtual channels are managed with a 64-bit credit), but more sophisticated routing algorithms — *i.e.* adaptive and fault-tolerant — can be implemented exploiting the precise information of the status of the receiving buffer of the neighboring nodes. The information contained in the credit is protected by redundancy (the status of the receiving FIFOs is repeated three times and the value is chosen by the majority). A whole credit is transmitted between two different packets considering the importance of the contained information. Besides, some information regarding the health of the node can be optionally embedded in the credits, allowing for a fault communication mechanism — LO|FA|MO [54] — that avoids single points of failure, and guaranteeing a fast broadcast of critical status to neighboring nodes. This embedding of diagnostic messages in the communication protocol limits the amount of additional overhead (no custom diagnostic packets are necessary) and prevents this flow from affecting overall performance.

APElink hardware IP

The APElink hardware IP manages the communication protocol over the serial links, adapting the inter-tile port interface (FIFO-based) of the APErouter with the outbound interface of the network adapter. The APElink IP consists of two main components: (i) the Transmission

Control Logic (TCL), a totally FPGA vendor-independent IP, that manages data and credit flow over the link (OSI Data Link, APElink) and (ii) the Transceiver, normally based on a customization of tools provided by the FPGA vendor, that implements the OSI Physical layer (APEphy).

The interface between TCL and Transceiver is based on a standard Ready/Valid mechanism, to be compliant with the AXI stream protocol (and not only), and to increase the compatibility of the APElink hardware component with different FPGA vendor IPs or custom transceiver controllers. The block diagram of current EXAnet APElink data transmission system is shown in Figure 5.9. An overview of the exploited FPGA hardware resources is reported in Table 5.3.

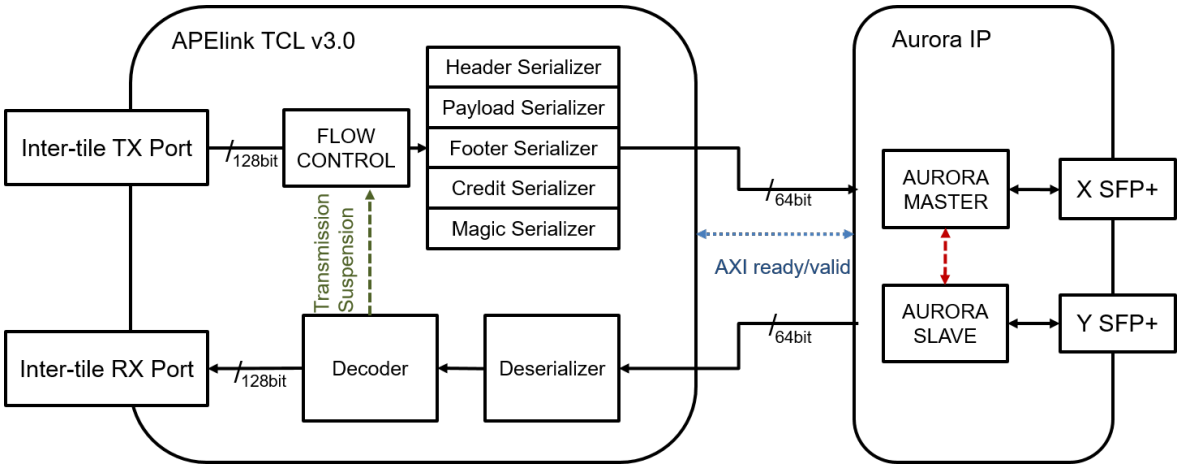

Figure 5.9: APElink block scheme.

IP	LUT	LUT RAM	LUT FF	Registers	BRAM	GTH
Available Resources	274080	144000	274080	548160	912	16
APElink TCL	**2627** (1.0%)	**0**	**849** (0.3%)	**2427** (0.4%)	**0**	**0**
RX CTRL	1364	0	351	0	0	0
TX CTRL	770	0	173	954	0	0
APEphy Aurora	**829** (0.3%)	**69** (0.1%)	**488** (0.2%)	**3107** (0.6%)	**2** (0.2%)	**2** (12.5%)
Master	405	35	244	1529	1	1
Slave	400	34	239	1508	1	1

Table 5.3: APElink hardware resources overview.

The transceiver IP is based on Xilinx Aurora 64B/66B core. Aurora 64B/66B is a lightweight, serial communications protocol for multi-gigabit links. It is used to transfer data between devices using one or many GTH transceivers. Connections can be full-duplex (data in both directions) or simplex (data in either one of the directions). The core supports the AMBA protocol AXI4- Stream user interface. Aurora 64B/66B cores automatically initialize a channel when they are connected to an Aurora 64B/66B channel partner. After

initialization, applications can pass data across the channel as frames or streams of data. Aurora 64B/66B frames can be of any size and can be interrupted any time by high priority requests. Gaps between valid data bytes are automatically filled with idles to maintain lock and prevent excessive electromagnetic interference. The Aurora 64B/66B protocol uses 64B/66B encoding. The 64B/66B encoding offers improved performance because of its very low (3%) transmission overhead, compared to 25% overhead for 8B/10B encoding.

The APElink operating frequency is 156.25 MHz in order to achieve 10 Gbps capability of the I/O system. The power consumption for each APElink TCL is limited to 0.009 W, as depicted in Figure 5.10, while the Aurora transceivers of the APEphy consume 0.337 W, without any major differences between the master and the slave channel (Figure 5.11).

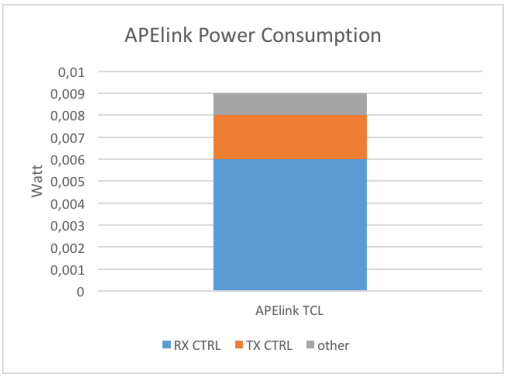

Figure 5.10: The APElink power consumption.

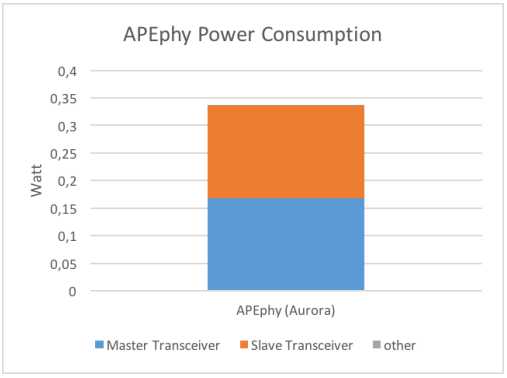

Figure 5.11: The APEphy power consumption.

APElink performance

The APElink hop latency is defined as the time to move the header from the inter-tile TX port of the APErouter of the sender node to the inter-tile RX port of the receiver node.

The hop latency L_L can be split into three main components (see Figure 5.12): (i) FIFO delay L_F, from the "write enable" signal asserted in the writing side to the "empty" signal not asserted in the read side; (ii) APElink Transmission Control Logic latency L_{TCL}, on both the sender (TX) and the receiver (RX); (iii) Aurora latency L_W, between the transceiver cores of the sender and receiver nodes:

$$L_L = L_F + L_{TCL} + L_W$$

The implemented FIFO and Xilinx Aurora IP latencies take 3 and 42 clock cycles respectively. The latency added by the Transmission Control Logic is 6 clock cycles only. In the current implementation, the APElink operates at 156.25 MHz with a capability of 10 Gbps,

thus the entire APElink hop latency is about 325 ns — *i.e.* **submicrosecond latency per hop**.

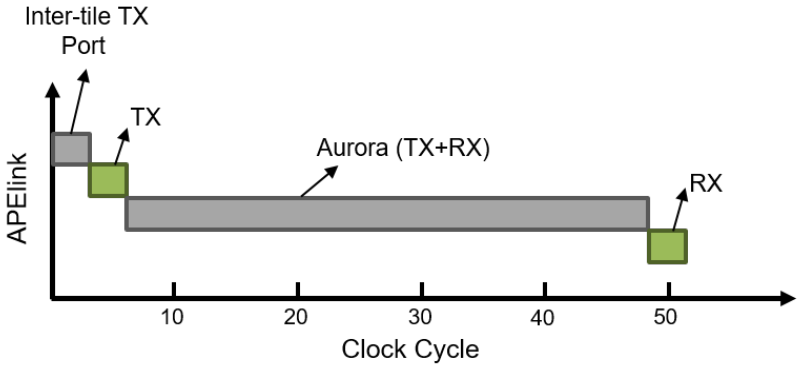

Figure 5.12: APElink hop latency.

An estimation of the APElink efficiency is provided. As described above, the user data are contained in the payload only, so the protocol overhead (P) — additional clock cycles to manage the message communication — depends on the sizes of *start* (S), *magic* (M), *header* (H), *footer* (F) and *credit* (C):

$$P = \frac{S}{W} + \frac{M}{W} + \frac{H}{W} + \frac{F}{W} + n\left(\frac{C}{W}\right) + k\left(\frac{C+M}{W}\right)$$

where W is the capability of the transceiver (bytes transmitted per clock cycle). The n credits between two different packets do not need the submission of a magic word, instead a *magic* is mandatory to distinguish the k credits from the payload, *i.e.* within the packet, in order to manage the suspension of the packet transmission. Therefore, the efficiency of the APElink protocol is:

$$E_P = \frac{\frac{D}{W}}{P + \frac{D}{W}}$$

where D is the size of the payload — D/W is the number of clock cycles to transmit the payload of the packet.

In ExaNet the *header* and *footer* are 16 Byte long. The *Magic* and *Start* width is always equal to transceiver bus width, each wasting only a clock cycle. The Credit issuing the *start*/*stop* command is 8 Byte only and the heavier version with the receiving buffer status is 32 Byte long (considering 8 virtual channels). The estimation of the efficiency for different payload sizes and credit types is reported in Table 5.4.

	Efficiency	
Payload Size [Byte]	**start/stop**	**usedword**
16	0.22	0.17
32	0.36	0.29
64	0.53	0.44
128	0.70	0.62
256	0.82	0.76
512	0.90	0.86
1024	0.94	0.90
2048	0.96	0.92
4096	0.97	0.93

Table 5.4: APElink data transmission efficiency considering a transceiver bus of 8 bytes.

5.4 KARMA Test Framework

King ARM Architecture (KARMA) is a software-oriented test framework to validate the ExaNet Network IP. The main idea behind its design is the use of the multicore ARM Cortex-A53 Programming System (PS) to emulate in software the functionalities of the Network Interface (NI), exploiting the AXI low latency communication capabilities between the PS and the Programming Logic (PL) that implements the systems under test. This approach turned out to be very effective, allowing for the test and validation of the ExaNet Network IP since the earliest stages of its development. It also enabled the rapid prototyping of various architectural solutions for the interface between the NI and the Switch systems. Finally, using the framework we were able to characterize the performance of the two systems in terms of latency.

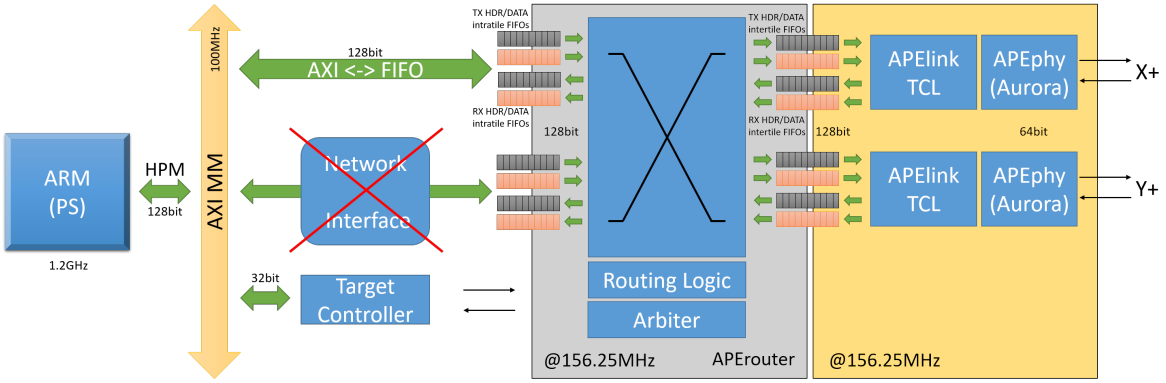

Figure 5.13: KARMA test framework for the ExaNet Network IP validation.

On the hardware side, the intra-tile ports are directly connected to the ARM HPM AXI port through an adapter IP, whose only purpose is the conversion between streaming and memory-mapped AXI protocols. Current KARMA does not implement any DMA-access to the intra-tile ports, so that ARM must issue a write for every single word into header/data

FIFOs, which is obviously suboptimal for bandwidth but appropriate for gauging the latency of small-sized packets.

Moreover, a set of configuration/status registers is accessible on the same AXI bus through the "Target Controller" IP, which allows the configuration of the router (*e.g.* setting coordinates and lattice size) and the probing of FIFOs and link status.

An overview of the KARMA test framework is depicted in Figure 5.13 and the resource usage of the ExaNet Network IP is reported in Table 5.5

IP	LUT	LUT RAM	LUT FF	Registers	BRAM	GTH
Available Resources	274080	144000	274080	548160	912	16
ExaNet Network IP	17287 (6.3%)	0	5577 (6.3%)	18954 (6.3%)	116.5 (12.7%)	0
APErouter 3×2	9599 (3.5%)	0	3162 (1.2%)	7649 (1.2%)	116.5 (12.7%)	0
APElink TCL (2x)	5253 (2.0%)	0	1698 (0.6%)	4854 (0.8%)	0	0
Target Controller	2468 (0.9%)	0	187 (0.1%)	6451 (1.1%)	0	0
APEphy Aurora	829 (0.3%)	69 (0.1%)	488 (0.2%)	3107 (0.6%)	2 (0.2%)	2 (12.5%)

Table 5.5: KARMA hardware resources overview.

The power consumption of ExaNet Network IP is reported in Figure 5.14. The ExaNet Network IP composed by a 3×2 APErouter, 2 APElink TCL and the target controller consumes 0.34 W. The total power consumption, considering the APEphy transceivers, is 0.677 W. The Zynq Ultrascale+ drains 2.822 W, thus the total power consumption of each board of the development platform is 3.5 W.

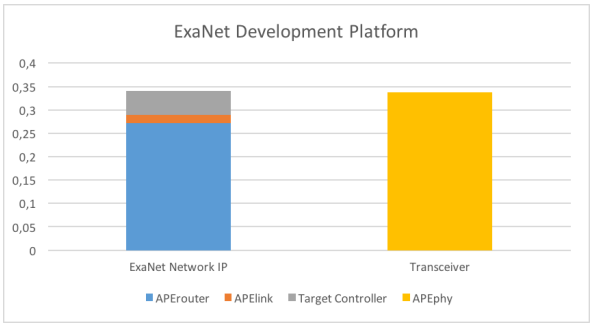

Figure 5.14: ExaNet Network IP power consumption.

5.4.1 Latency test

On the software side, a first test is implemented in user-space by simply writing commands and data to the hardware (using the `/dev/mem` to access the memory-mapped hardware). In this phase no interrupts, no system-wide locking and no easy virtual-to-physical address translation are implemented. A kernel-space device driver creates a "proc" file-system entry

to output debug and status information, together with the output of the internal configuration/status registers. The module also parses the device tree to find the IRQ number associated to the "NIC", and then assigns a callback function to handle the interrupt request generated by the arrival of new data.

Figure 5.15 shows the normal execution of a generic Send/Receive test execution using the kernel space module.

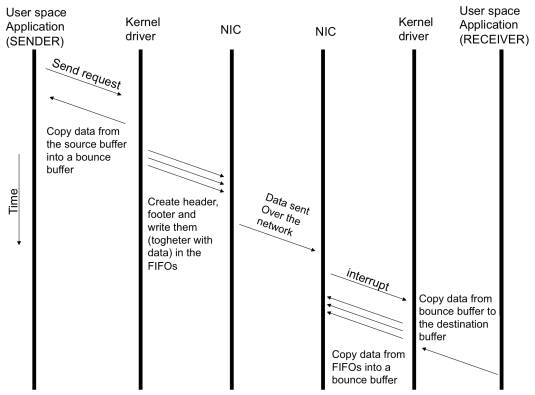

Figure 5.15: Send/Receive test execution using kernel module API.

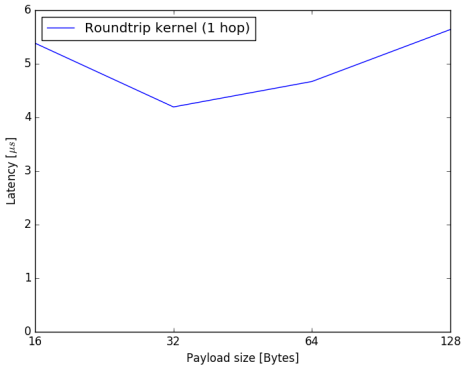

Figure 5.16: Latency for small packets in kernel space test.

In the sending phase, the kernel-space module copies data from the user buffer to a bounce buffer, then prepares header and footer and writes them onto the corresponding FIFOs. The receiving phase is just the opposite: arriving data are copied in a bounce buffer (waiting for the user-space process to request them) while header and footer are "consumed". Round-trip latencies between two boards have been measured at different sizes, up to 4 KB, as shown in Figure 5.16.

Because of the non-optimal bounce-buffering mechanism and the notoriously slow interrupt handling by GNU/Linux, in Figure 5.17 and Figure 5.18 we compare these results with a test where the kernel driver is bypassed by a user-space ping-pong application, again exploiting `/dev/mem` to directly access the memory-mapped hardware. The stated difference in time of 0.46μs for the two and one hops measurements provides an estimate of the single hop traversal time contribution to the total latency. The times spent by the ARM in reading ($\sim 0.4\mu$s, about 20 clock cycles per word) and in writing ($< 0.1\mu$s, 4 clock cycles per word) the intra-tile port are independent from the number of hops.

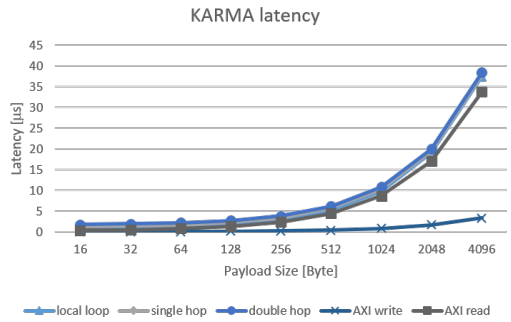

Figure 5.17: The Roundtrip latency for one and two hops.

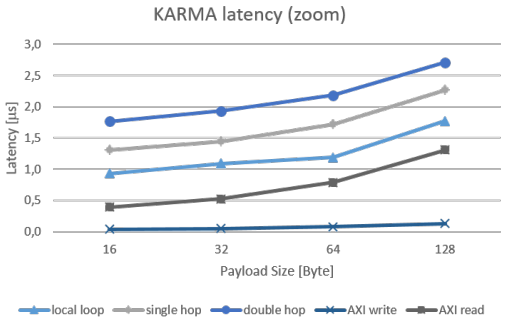

Figure 5.18: A small-packet, up to 128 Byte — zoom of the roundtrip latency.

5.4.2 Hardware Bandwidth Test

As stated before, the KARMA testbed is not the appropriate platform to evaluate the bandwidth of the hardware solution proposed.

Nevertheless, the ExaNet Network IP firmware is provided with a self-test mechanism to measure the bandwidth achieved by the APErouter and APElink. The self-test mechanism is composed by three simple IPs: (i) the *Traffic Generator* generates EXApackets and fills in the transmitting FIFOs; (ii) the *Consumer* flushes the receiving FIFOs avoiding the overflow; (iii) the *Performance Counter* stores the clock cycles needed to complete the data transfers. The Packets can be configured through configuration registers defining the type, the size, the destination coordinates and ports.

Figure 5.19 shows the APErouter achieved bandwidth while moving data between two different ports. The square markers denote the theoretical peak bandwidth considering the 128-bit bus operating at 156.25 MHz. The efficiency is 76% for a 512-byte packet — *i.e.* the maximum packet size — when the protocol overhead is 6.25%. The loss of performance is due to the not optimized pipeline of APErouter hardware IP. Some improvements are gained doubling the sending ports (*i.e.* $2\times$ Intra-Tile ports) and transmitting packets to the same target port; in this case, the efficiency at 512 byte is 89.5%.

The APElink result is shown in Figure 5.20. The theoretical bandwidth is limited to 10 Gbps due to the SFP+ connectors of the Trenz Boards. The efficiency is 90% for a 512-byte packet, in line with the estimation of Table 5.4.

5.5 Network simulator results

Benchmarking and characterization of an interconnection network depend on many parameters — *e.g.* traffic pattern, buffers and network sizes — therefore at very large scale simula-

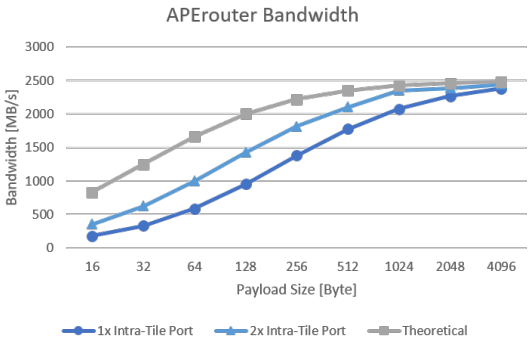

Figure 5.19: APErouter bandwidth.

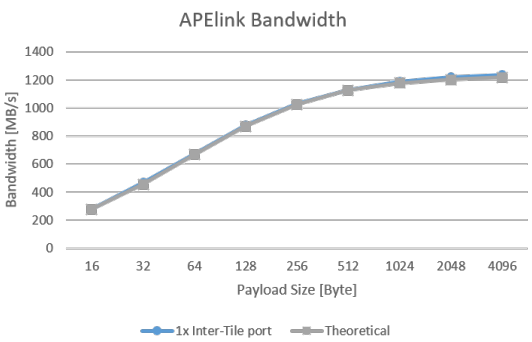

Figure 5.20: APElink bandwidth.

tions are mandatory to evaluate solutions and understand criticalities. The simulator [107] is implemented in a modular way to ease switching between different network designs (topologies, routing algorithms and traffic generators), and to allow for an effective way of measuring latency and accepted traffic.

For APEnet in ExaNeSt evaluation, we used the OMNeT++ [108] framework to implement the base functionality of the APEnet network in a proprietary simulation library.

As a first evaluation, we use synthetic benchmarks on different network configurations. All the nodes were producing traffic using a Bernoulli process and with a uniformly random destination. The tests were performed for 2D/3D torus and dragonfly topologies and for several routing algorithms, as listed in Table 5.6. Star-channel [109] is a minimal path fully adaptive routing algorithm for N-dimensional tori and meshes based on the e-cube. This algorithm adds an extra virtual channel in addition to the ones used by the e-cube to achieve full adaptivity. The Smart dimension-order is a partly adaptive non minimal routing algorithm for N-dimensional tori and meshes based on the e-cube. This algorithm takes advantage of the extra unused channel to provide adaptivity to the network. Finally the min-routing [18] is a minimal non adaptive routing algorithm for dragonflies and it can be considered as a base algorithm for adaptive or more complex ones.

In the test, the accepted traffic is normalized dividing it by the number of nodes in the network.

Network topology	**Routing Algorithm**
2D torus 10×10	e-cube, star-channel, smart dim-order
2D torus 32×32	e-cube, star-channel, smart dim-order
3D torus $10 \times 10 \times 10$	e-cube, star-channel, smart dim-order
Fully connected dragonfly “72 nodes”	min-routing
Fully connected dragonfly “1056 nodes”	min-routing

Table 5.6: Topology and routing algorithm analyzed.

Preliminary results for the network accepted traffic (Figure 5.21) show a linear region shared by all the different network configurations tested.

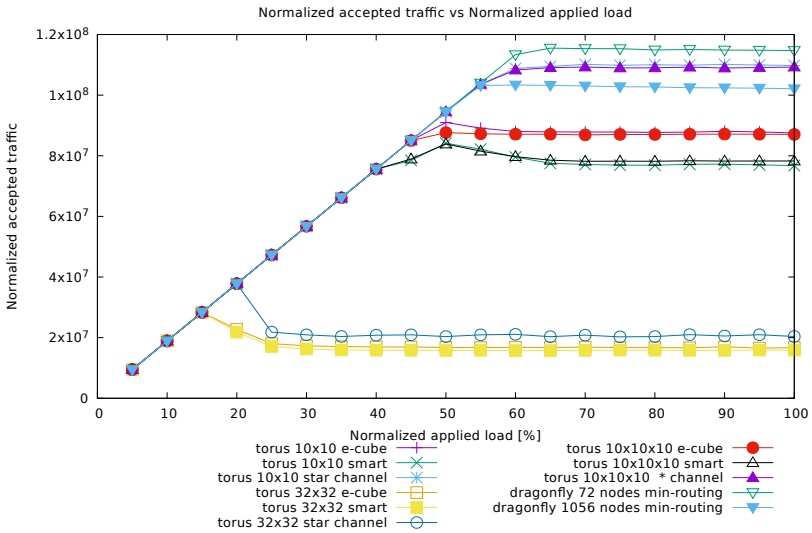

Figure 5.21: Normalized accepted throughput vs applied load.

When the network is in the linear region, it is below its critical congestion threshold and properly handles the incoming traffic; enhancing the applied load results in higher accepted traffic. If the applied load is above the saturation point, the accepted traffic starts to exit from the linear region of the plot and reaches a plateau. The plateau value could not correspond to the maximum value that the network is able to deliver due to congestion effects.

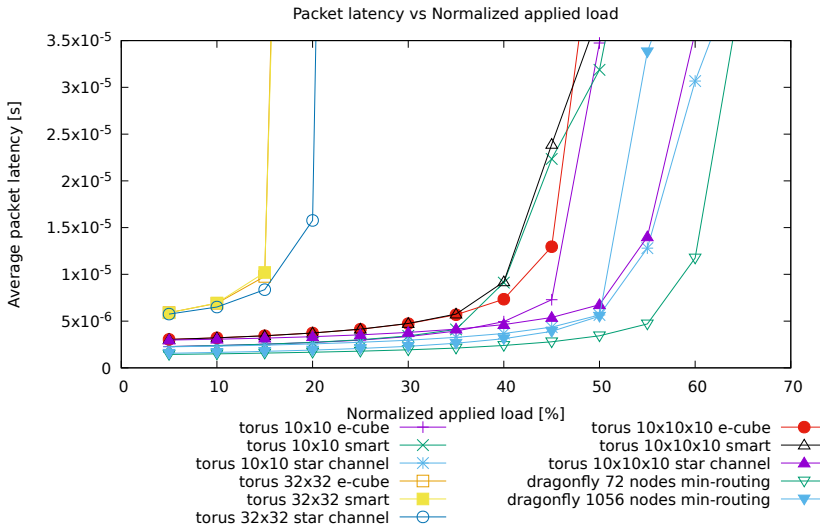

Figure 5.22: Latency vs applied load for the different configurations tested.

The latency graph in Figure 5.22 shows similar behavior with an upward slope when

applied load is increased. It is important to note that even if the network has not reached saturation and the accepted traffic has not plateaued yet, latency increases together with the applied load, leading to longer delivery times for the packets. This aspect must be taken into account when designing the system in order to meet the requirements.

The 2D tori handle $\sim$45% normalized applied load in a 10×10 network configuration, but only $\sim$20% in a 32×32 configuration. Tori are not optimized for uniform network traffic and the performance degrades quickly increasing the radius of the configuration. To reduce the radius, we can move from a 2D to a 3D torus. The fully adaptive star-channel routing algorithm provides a better use of the available network resources, resulting in higher sustained load and lower latency than those achievable by using the simpler e-cube (DOR) routing. The 72 nodes dragonfly performs better than 10×10 torus adopting the fully adaptive algorithm but on a smaller network, while the 1056 nodes setup shows better performance than the non-adaptive tori but with lower throughput than the fully adaptive ones.

CONCLUDING REMARKS

I have described two research topics currently explored by the INFN APE research group: (i) the study of modern technologies and networking strategies for the development of a European interconnect system for the Exascale High Performance Computing infrastracture in the framework of the EU H2020 ExaNeSt project, and (ii) the high resolution simulation of cortical activity expressed in different brain-states, *e.g.* the slow waves expressed during deep-sleep and anesthesia and the asynchronous neural regime characteristic of wakefulness. The joint between these research branches is the driving motive of this thesis. The achieved results are the first-stage product of a hopefully effective system design method based on co-design approach.

The Distributed and Plastic Spiking Neural Network (DPSNN) simulation engine is exploited as a source of requirements and architectural inspiration for future parallel/distributed computing systems. The miniDPSNN approach has been proven as an adequate tool of evaluation to outline software and hardware architectures dedicated to neural simulations. The characterization of the network traffic generated by a cortical simulator running on a standard computing system provides information for the definition of specification of alternative network adapter. The ever growing amount of small-sized packets generated by a strong scaling test shows the limitation of the adopted off-the-shelf interconnects — *i.e.* Infiniband and Gigabit Ethernet.

The result obtained with miniDPSNN drives to the specification of a network IP, characterized by a low-latency transfer optimized architecture and to the definition of a data transmission protocol providing high-throughput also for small dimensions of data payload. The APEnet architecture is at the basis of the ExaNeSt hierarchical multi-tiered network. The presented ExaNet Network IP provides a point-to-point latency of 1.3 μs in a standard ping-pong test. APErouter and APElink sport an efficiency of 76% and 90% respectively, transferring 512-bytes packets. Further, from the software point of view, a tuned version of the MPI-based communication within the DPSNN leveraging a hierarchical mechanism is planned for the next future.

ARM processors turned out to be an efficient solution in terms of power consumption. The energy-to-solution result obtained running the DPSNN application on ARM Cortex-A57 based platform is about three times lower than the x86 core architecture. The power consumption of the ExaNet board is < 4 W according to the preliminary estimations, with the ExaNet Network IP coupled with Aurora-based APEphy draining less than 1 W.

Finally, the APElink communication protocol provides two key elements for the achievement of a resilient interconnection network architecture: (i) the mechanism based on the

implementation of Error Correction Code to discard faulty packets avoiding the misrouting of garbage data traffic and (ii) the management of credit flow at datalink level, with minimal additional protocol overhead to create a general awareness of the health status of the computing system.

To sum up, the thesis explores three pillars of the race towards the Exascale (system power constraint, data movement optimization, hardware/software co-desing), introducing effective solutions. Nevertheless, this study is still a long way from the arrival. ExaNeSt is just getting in the second half period and the obtained results represent only the starting point. The node of the system, the QFDB, will be ready at the end of 2017. An evolution of the ExaNet Network IP equipped with a larger I/O interface will be developed to manage the inter-Mezzanine communication at Tier 2. The track-1 prototype will provide an adequate platform to test the n-dimensional torus and dragonfly topologies. A more sophisticated routing logic will be able to consume the coordinates in a more exotic way, or to recognize critical directions and then change appropriately the packet header to follow an alternative path to reach the destination (adaptive and fault-tolerant routing). The DPSNN provides an additional requirement coming directly by the construction of connectivity infrastructure and the delivery of spiking messages during the simulation phase, both based on widespread use of calls to the `MPI_Alltoallv` library function. An HPC computing system equipped with hardware offloading mechanism of collective communication functions should enhance overall application performance, especially for very large scale runs. This feature will be designed and tested exploiting ExaNet and the track-1 prototype, both representing the basis for the track-2 prototype. A patent [110] is pending on the topic, further details will be available in the next future. Finally, Track-2 platform will be finalized in the framework of the EU H2020 EuroEXA project, started in September 2017 with the goal to innovate across a new ground-breaking platform for computing in its support to deliver Exascale systems, collecting the achievements of ExaNeSt, ECOSCALE and ExaNoDe projects.

ACKNOWLEDGMENT

This work was carried out with support from the ExaNeSt project, funded by the European Union Horizon 2020 Research and Innovation Programme under Grant Agreement No. 671553, and from the Human Brain Project (HBP), Grant Agreement No. 720270 (HBP SGA1).

Bibliography

[1] P. Messina, "The exascale computing project," *Computing in Science Engineering*, vol. 19, no. 3, pp. 63–67, May 2017.

[2] "Human Brain Project," accessed: 22/Sep/2017. [Online]. Available: https://www.humanbrainproject.eu/en/

[3] "Advanced scientific computing research," accessed: 29/Sep/2017. [Online]. Available: https://science.energy.gov/ascr/

[4] J. Shalf, S. Dosanjh, and J. Morrison, *Exascale Computing Technology Challenges*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1–25. [Online]. Available: https://doi.org/10.1007/978-3-642-19328-6_1

[5] S. andPeteBeckmanandJackieChen andPhilColella andBillCollinsandDonaCrawford andJackDongarraandDougKothe andRustyLusk andPaulMessina andTonyMezzacappa andParvizMoin andMikeNorman andRobertRosner andVivekSarkarandAndrewSiegelandFredStreitzandAndyWhite andMargaretWright, "The Opportunities and Challenges of Exascale Computing," U.S. Department of Energy, Office of Science, Tech. Rep., 2010.

[6] "Europe towards exascale," accessed: 29/Sep/2017. [Online]. Available: https://ec.europa.eu/digital-single-market/en/news/europe-towards-exascale/

[7] "Exascale computing project," accessed: 29/Sep/2017. [Online]. Available: https://exascaleproject.org/

[8] "Chinas exascale supercomputer operational by 2020," accessed: 29/Sep/2017. [Online]. Available: http://english.gov.cn/news/top_news/2016/06/16/content_281475373200996.htm

[9] "Exascale supercomputer project (riken)," accessed: 29/Sep/2017. [Online]. Available: http://www.aics.riken.jp/en/topics/exascale-supercomputer-project-launched.html

[10] Accessed: 2017-02-02. [Online]. Available: http://www.top500.org

[11] J. J. Dongarra, P. Luszczek, and A. Petitet, "The linpack benchmark: Past, present, and future. concurrency and computation: Practice and experience," *Concurrency and Computation: Practice and Experience*, vol. 15, p. 2003, 2003.

[12] H. Fu, J. Liao, J. Yang, L. Wang, Z. Song, X. Huang, C. Yang, W. Xue, F. Liu, F. Qiao, W. Zhao, X. Yin, C. Hou, C. Zhang, W. Ge, J. Zhang, Y. Wang, C. Zhou, and G. Yang, "The sunway taihulight supercomputer: system and applications," *Science China Information Sciences*, vol. 59, no. 7, p. 072001, Jun 2016. [Online]. Available: https://doi.org/10.1007/s11432-016-5588-7

[13] X. Liao, L. Xiao, C. Yang, and Y. Lu, "Milkyway-2 supercomputer: System and application," *Front. Comput. Sci.*, vol. 8, no. 3, pp. 345–356, Jun. 2014. [Online]. Available: http://dx.doi.org/10.1007/s11704-014-3501-3

[14] J. Liu, J. Wu, and D. Panda, "High performance rdma-based mpi implementation over infiniband," *International Journal of Parallel Programming*, vol. 32, no. 3, pp. 167–198, 2004. [Online]. Available: http://dx.doi.org/10.1023/B:IJPP.0000029272.69895.c1

[15] Z. Pang, M. Xie, J. Zhang, Y. Zheng, G. Wang, D. Dong, and G. Suo, "The th express high performance interconnect networks," *Frontiers of Computer Science*, vol. 8, no. 3, pp. 357–366, Jun 2014. [Online]. Available: https://doi.org/10.1007/s11704-014-3500-9

[16] "Piz daint," accessed: 29/Sep/2017. [Online]. Available: http://www.cscs.ch/computers/piz_daint_piz_dora/

[17] G. Faanes, A. Bataineh, D. Roweth, T. Court, E. Froese, B. Alverson, T. Johnson, J. Kopnick, M. Higgins, and J. Reinhard, "Cray cascade: A scalable hpc system based on a dragonfly network," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '12. Los Alamitos, CA, USA: IEEE Computer Society Press, 2012, pp. 103:1–103:9. [Online]. Available: http://dl.acm.org/citation.cfm?id=2388996.2389136

[18] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," in *2008 International Symposium on Computer Architecture*, June 2008, pp. 77–88.

[19] "Titan," accessed: 29/Sep/2017. [Online]. Available: https://www.olcf.ornl.gov/titan/

[20] "Cori," accessed: 29/Sep/2017. [Online]. Available: http://www.nersc.gov/users/computational-systems/cori/

[21] "Sequoia," accessed: 29/Sep/2017. [Online]. Available: https://hpc.llnl.gov/hardware/platforms/sequoia

[22] D. Chen, N. A. Eisley, P. Heidelberger, R. M. Senger, Y. Sugawara, S. Kumar, V. Salapura, D. L. Satterfield, B. Steinmacher-Burow, and J. J. Parker, "The ibm blue gene/q interconnection network and message unit," in *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2011, pp. 1–10.

[23] ——, "The ibm blue gene/q interconnection network and message unit," in *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2011, pp. 1–10.

[24] M. Yokokawa, F. Shoji, A. Uno, M. Kurokawa, and T. Watanabe, "The k computer: Japanese next-generation supercomputer development project," in *IEEE/ACM International Symposium on Low Power Electronics and Design*, Aug 2011, pp. 371–372.

[25] Y. Ajima, S. Sumimoto, and T. Shimizu, "Tofu: A 6d mesh/torus interconnect for exascale computers," *Computer*, vol. 42, no. 11, pp. 36–40, Nov 2009.

[26] Iacovos Mavroidis, "Ecoscale architecture and opencl tasks acceleration in hardware," 2017, https://agenda.infn.it/getFile.py/access?contribId=7&resId=1&materialId=slides&confId=13056.

[27] J. Romoth, M. Porrmann, and U. Rckert, "Survey of FPGA applications in the period 2000 2015 (Technical Report)," 2017.

[28] R. V. Aroca and L. M. G. Gonalves, "Towards green data centers: A comparison of x86 and {ARM} architectures power efficiency," *Journal of Parallel and Distributed Computing*, vol. 72, no. 12, pp. 1770 – 1780, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731512002122

[29] R. P. Luijten and A. Doering, "The dome embedded 64 bit microserver demonstrator," in *Proceedings of 2013 International Conference on IC Design Technology (ICICDT)*, May 2013, pp. 203–206.

[30] Y. Durand, P. M. Carpenter, S. Adami, A. Bilas, D. Dutoit, A. Farcy, G. Gaydadjiev, J. Goodacre, M. Katevenis, M. Marazakis, E. Matus, I. Mavroidis, and J. Thomson, "Euroserver: Energy efficient node for european micro-servers," in *2014 17th Euromicro Conference on Digital System Design*, Aug 2014, pp. 206–213.

[31] N. Rajovic *et al.*, "Supercomputing with commodity cpus: Are mobile socs ready for hpc?" in *2013 SC - International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2013, pp. 1–12.

[32] "Array processor experiment (ape)," accessed: 20/Oct/2017. [Online]. Available: http://apegate.roma1.infn.it/mediawiki/index.php/

[33] A. Biagioni *et al.*, "Low latency network and distributed storage for next generation hpc systems: the exanest project," October 2016, international Conference on Computing on High-Energy Physics (CHEP). [Online]. Available: https://indico.cern.ch/event/505613/contributions/2227431/attachments/1350266/2038268/oral-325.pdf

[34] ——, "The brain on low power scalable architectures: efficient simulation of cortical slow waves and asynchronous states." September 2017, international Conference on Parallel Computing and HPC (ParCo). [Online]. Available: http://www.hpc.cineca.it/content/parco-2017-program

[35] M. Katevenis *et al.*, "The ExaNeSt Project: Interconnects, Storage, and Packaging for Exascale Systems," in *2016 Euromicro Conference on Digital System Design (DSD)*, Aug 2016, pp. 60–67.

[36] "ExaNoDe," accessed: 2017-02-02. [Online]. Available: http://exanode.eu/

[37] I. Mavroidis *et al.*, "Ecoscale: Reconfigurable computing and runtime system for future exascale systems," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 696–701.

[38] J. Laudon and D. Lenoski, "The sgi origin: A ccnuma highly scalable server," *SIGARCH Comput. Archit. News*, vol. 25, no. 2, pp. 241–251, May 1997. [Online]. Available: http://doi.acm.org/10.1145/384286.264206

[39] R. Capuzzo-Dolcetta, M. Spera, and D. Punzo, "A fully parallel, high precision, n-body code running on hybrid computing platforms," *Journal of Computational Physics*, vol. 236, pp. 580 – 593, 2013. [Online]. Available: //www.sciencedirect.com/science/article/pii/S0021999112006900

[40] V. Springel, "The cosmological simulation code gadget-2," *Monthly Notices of the Royal Astronomical Society*, vol. 364, no. 4, p. 1105, 2005. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2966.2005.09655.x

[41] P. Monaco, T. Theuns, and G. Taffoni, "The pinocchio algorithm: pinpointing orbit-crossing collapsed hierarchical objects in a linear density field," *Monthly Notices of the Royal Astronomical Society*, vol. 331, no. 3, p. 587, 2002. [Online]. Available: http://dx.doi.org/10.1046/j.1365-8711.2002.05162.x

[42] T. Theuns *et al.*, "Swift: Task-based hydrodynamics and gravity for cosmological simulations," in *Proceedings of the 3rd International Conference on Exascale Applications and Software*, ser. EASC '15, 2015, pp. 98–102.

[43] M. Januszewski and M. Kostur, "Sailfish: A flexible multi-gpu implementation of the lattice boltzmann method," *Computer Physics Communications*, vol. 185, no. 9, pp. 2350 – 2368, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0010465514001520

[44] "OpenFOAM," accessed: 2017-02-02. [Online]. Available: http://openfoam.org/

[45] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," *Journal of Computational Physics*, vol. 117, no. 1, pp. 1 – 19, 1995. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002199918571039X

[46] "RegCM," accessed: 2017-02-02. [Online]. Available: http://www.ictp.it/research/esp/models/regcm4.aspx

[47] "MonetDB," accessed: 2017-10-24. [Online]. Available: https://www.monetdb.org/

[48] P. S. Paolucci *et al.*, "Dynamic many-process applications on many-tile embedded systems and HPC clusters: The EURETILE programming environment and execution platforms," *Journal of Systems Architecture*, vol. 69, pp. 29–53, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1383762115001423

[49] W. E. Denzel, J. Li, P. Walker, and Y. Jin, "A framework for end-to-end simulation of high-performance computing systems," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, ser. Simutools '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 21:1–21:10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1416222.1416248

[50] T. Hoefler, "Software and hardware techniques for power-efficient hpc networking," *Computing in Science Engineering*, vol. 12, no. 6, pp. 30–37, Nov 2010.

[51] A. K. Kodi, B. Neel, and W. C. Brantley, "Photonic interconnects for exascale and datacenter architectures," *IEEE Micro*, vol. 34, no. 5, pp. 18–30, Sept 2014.

[52] N. Chrysos, L. Chen, C. Kachris, and M. Katevenis, "Discharging the network from its flow control headaches: Packet drops and hol blocking," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 15–28, Feb 2016.

[53] M. Cuviello, S. Dey, X. Bai, and Y. Zhao, "Fault modeling and simulation for crosstalk in system-on-chip interconnects," in *Proceedings of the 1999 IEEE/ACM International Conference on Computer-aided Design*, ser. ICCAD '99. Piscataway, NJ, USA: IEEE Press, 1999, pp. 297–303. [Online]. Available: http://dl.acm.org/citation.cfm?id=339492.340030

[54] R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, P. S. Paolucci, D. Rossetti, F. Simula, L. Tosoratto, and P. Vicini, "A Hierarchical Watchdog Mechanism for Systemic Fault Awareness on Distributed Systems," *Future Generation Computer Systems*, vol. 53, pp. 90 – 99, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X14002751

[55] B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, T. Hoefler, J. Joyner, J. Lewis, J. Li, N. Ni, and R. Rajamony, "The percs high-performance interconnect," in *2010 18th IEEE Symposium on High Performance Interconnects*, Aug 2010, pp. 75–82.

[56] M. Xie, Y. Lu, K. Wang, L. Liu, H. Cao, and x. yang, "Tianhe-1a interconnect and message-passing services," *IEEE Micro*, vol. 32, no. 1, pp. 8–20, Jan 2012.

[57] J. Zhang, F. Ren, and C. Lin, "Modeling and understanding tcp incast in data center networks," in *2011 Proceedings IEEE INFOCOM*, April 2011, pp. 1377–1385.

[58] M.-O. Gewaltig and M. Diesmann, "Nest (neural simulation tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[59] S. Kunkel, A. Morrison, P. Weidel, J. M. Eppler, A. Sinha, W. Schenck, M. Schmidt, S. B. Vennemo, J. Jordan, A. Peyser, D. Plotnikov, S. Graber, T. Fardet, D. Terhorst, H. Mrk, G. Trensch, A. Seeholzer, R. Deepu, J. Hahne, I. Blundell, T. Ippen, J. Schuecker, H. Bos, S. Diaz, E. Hagen, S. Mahmoudian, C. Bachmann, M. E. Lepperd, O. Breitwieser, B. Golosio, H. Rothe, H. Setareh, M. Djurfeldt, T. Schumann, A. Shusharin, J. Garrido, E. B. Muller, A. Rao, J. H. Vieites, and H. E. Plesser, "Nest 2.12.0," Mar. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.259534

[60] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris, M. Zirpe, T. Natschläger, D. Pecevski, B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Vieville, E. Muller, A. P. Davison, S. El Boustani, and A. Destexhe, "Simulation of networks of spiking neurons: A review of tools and strategies," *Journal of Computational Neuroscience*, vol. 23, no. 3, pp. 349–398, Dec 2007. [Online]. Available: https://doi.org/10.1007/s10827-007-0038-6

[61] M. A. Wilson, U. S. Bhalla, J. D. Uhley, and J. M. Bower, "Genesis: A system for simulating neural networks," in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. Morgan-Kaufmann, 1989, pp. 485–492. [Online]. Available: http://papers.nips.cc/paper/182-genesis-a-system-for-simulating-neural-networks.pdf

[62] M. Mattia and P. D. Giudice, "Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses," *Neural Computation*, vol. 12, no. 10, pp. 2305–2329, Oct 2000.

[63] J. M. Nageswaran, N. Dutt, J. L. Krichmar, A. Nicolau, and A. V. Veidenbaum, "A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors," *Neural Networks*, vol. 22, no. 5, pp. 791 – 800, 2009, advances in Neural Networks Research: IJCNN2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608009001373

[64] E. M. Izhikevich and G. M. Edelman, "Large-scale model of mammalian thalamocortical systems," *Proceedings of the National Academy of Sciences*, vol. 105, no. 9, pp. 3593–3598, 2008. [Online]. Available: http://www.pnas.org/content/105/9/3593.abstract

[65] D. S. Modha, R. Ananthanarayanan, S. K. Esser, A. Ndirango, A. J. Sherbondy, and R. Singh, "Cognitive computing," *Commun. ACM*, vol. 54, no. 8, pp. 62–71, Aug. 2011. [Online]. Available: http://doi.acm.org/10.1145/1978542.1978559

[66] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the spinnaker system architecture," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2454–2467, Dec 2013.

[67] S. Schmitt, J. Klaehn, G. Bellec, A. Grübl, M. Guettler, A. Hartel, S. Hartmann, D. H. de Oliveira, K. Husmann, V. Karasenko, M. Kleider, C. Koke, C. Mauch,

E. Müller, P. Müller, J. Partzsch, M. A. Petrovici, S. Schiefer, S. Scholze, B. Vogginger, R. A. Legenstein, W. Maass, C. Mayr, J. Schemmel, and K. Meier, "Neuromorphic hardware in the loop: Training a deep spiking network on the brainscales wafer-scale system," *CoRR*, vol. abs/1703.01909, 2017. [Online]. Available: http://arxiv.org/abs/1703.01909

[68] M. Ruiz-Mejias, L. Ciria-Suarez, M. Mattia, and M. V. Sanchez-Vives, "Slow and fast rhythms generated in the cerebral cortex of the anesthetized mouse," *Journal of Neurophysiology*, vol. 106, no. 6, pp. 2910–2921, 2011.

[69] A. Stroh, H. Adelsberger, A. Groh, C. Rhlmann, S. Fischer, A. Schierloh, K. Deisseroth, and A. Konnerth, "Making waves: Initiation and propagation of corticothalamic ca2+ waves invivo," *Neuron*, vol. 77, no. 6, pp. 1136 – 1150, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0896627313000974

[70] "Eu fp7 corticonic grant n. 600806," 2013-2016.

[71] P. Schnepel, A. Kumar, M. Zohar, A. Aertsen, and C. Boucsein, "Physiology and impact of horizontal connections in rat neocortex," *Cerebral Cortex*, vol. 25, no. 10, pp. 3818–3835, 2015. [Online]. Available: +http://dx.doi.org/10.1093/cercor/bhu265

[72] A. Stepanyants, L. M. Martinez, A. S. Ferecsk, and Z. F. Kisvrday, "The fractions of short- and long-range connections in the visual cortex," *Proceedings of the National Academy of Sciences*, vol. 106, no. 9, pp. 3555–3560, 2009. [Online]. Available: http://www.pnas.org/content/106/9/3555.abstract

[73] C. Boucsein, M. Nawrot, P. Schnepel, and A. Aertsen, "Beyond the cortical column: Abundance and physiology of horizontal connections imply a strong role for inputs from the surround," *Frontiers in Neuroscience*, vol. 5, p. 32, 2011. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2011.00032

[74] A. Schz, D. Chaimow, D. Liewald, and M. Dortenman, "Quantitative aspects of corticocortical connections: A tracer study in the mouse," *Cerebral Cortex*, vol. 16, no. 10, pp. 1474–1486, 2006. [Online]. Available: +http://dx.doi.org/10.1093/cercor/bhj085

[75] T. C. Potjans and M. Diesmann, "The cell-type specific cortical microcircuit: Relating structure and activity in a full-scale spiking network model," *Cerebral*

Cortex, vol. 24, no. 3, pp. 785–806, 2014. [Online]. Available: +http://dx.doi.org/10.1093/cercor/bhs358

[76] S. Song, K. D. Miller, and L. F. Abbott, "Competitive hebbian learning through spike-timing-dependent synaptic plasticity," pp. 919–926, 2000.

[77] A. Morrison, C. Mehring, T. Geisel, A. Aertsen, and M. Diesmann, "Advancing the boundaries of high-connectivity network simulation with distributed computing," *Neural Computation*, vol. 17, no. 8, pp. 1776–1801, Aug 2005.

[78] L. Lapicque, "Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation," *J. Physiol. Pathol. Gen.*, vol. 9, pp. 620–635, 1907.

[79] G. Gigante, M. Mattia, and P. D. Giudice, "Diverse population-bursting modes of adapting spiking neurons," *Phys. Rev. Lett.*, vol. 98, p. 148101, Apr 2007. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.98.148101

[80] "Cineca," accessed: 22/Sep/2017. [Online]. Available: https://www.cineca.it/

[81] N. Rajovic, A. Rico, F. Mantovani, D. Ruiz, J. O. Vilarrubi, C. Gomez, L. Backes, D. Nieto, H. Servat, X. Martorell, J. Labarta, E. Ayguade, C. Adeniyi-Jones, S. Derradji, H. Gloaguen, P. Lanucara, N. Sanna, J. F. Mehaut, K. Pouget, B. Videau, E. Boyer, M. Allalen, A. Auweter, D. Brayford, D. Tafani, V. Weinberg, D. Brmmel, R. Halver, J. H. Meinke, R. Beivide, M. Benito, E. Vallejo, M. Valero, and A. Ramirez, "The mont-blanc prototype: An alternative approach for hpc systems," in *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov 2016, pp. 444–455.

[82] "The montblanc project," accessed: 27/Sep/2017. [Online]. Available: www.montblanc-project.eu

[83] M. Marazakis, J. Goodacre, D. Fuin, P. Carpenter, J. Thomson, E. Matus, A. Bruno, P. Stenstrom, J. Martin, Y. Durand, and I. Dor, "Euroserver: Share-anything scale-out micro-server design," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 678–683.

[84] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[85] E. Stromatias, F. Galluppi, C. Patterson, and S. Furber, "Power analysis of large-scale, real-time neural networks on spinnaker," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Aug 2013, pp. 1–8.

[86] P. S. Paolucci, R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, E. Pastorelli, F. Simula, L. Tosoratto, and P. Vicini, "Distributed simulation of polychronous and plastic spiking neural networks: strong and weak scaling of a representative mini-application benchmark executed on a small-scale commodity cluster," *arXiv:1310.8478*, Oct. 2013, http://arxiv.org/abs/1310.8478.

[87] M. A. Heroux, D. W. Doerfler, P. S. Crozier, J. M. Willenbring, H. C. Edwards, A. Williams, M. Rajan, E. R. Keiter, H. K. Thornquist, and R. W. Numrich, "Improving Performance via Mini-applications," Sandia National Laboratories, Tech. Rep. SAND2009-5574, 2009.

[88] M. J. Cordery, B. Austin, H. J. Wassermann, C. S. Daley, N. J. Wright, S. D. Hammond, and D. Doerfler, *Analysis of Cray XC30 Performance Using Trinity-NERSC-8 Benchmarks and Comparison with Cray XE6 and IBM BG/Q*. Cham: Springer International Publishing, 2014, pp. 52–72. [Online]. Available: https://doi.org/10.1007/978-3-319-10214-6-3

[89] "Fiber miniapp suite," accessed: 10/Oct/2017. [Online]. Available: http://fiber-miniapp.github.io/

[90] R. Ananthanarayanan, S. K. Esser, H. D. Simon, and D. S. Modha, "The cat is out of the bag: cortical simulations with 10^9 neurons, 10^{13} synapses," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, Nov 2009, pp. 1–12.

[91] D. Cesini *et al.*, "Power-efficient computing: Experiences from the cosa project," *Scientific Programming*, vol. 2017, p. 14, 2017, article ID 7206595.

[92] M. Albanese, P. Bacilieri, S. Cabasino, N. Cabibbo, F. Costantini, G. Fiorentini, F. Flore, A. Fonti, A. Fucci, M. Lombardo, S. Galeotti, P. Giacomelli, P. Marchesini, E. Marinari, F. Marzano, A. Miotto, P. Paolucci, G. Parisi, D. Pascoli, D. Passuello, S. Petrarca, F. Rapuano, E. Remiddi, R. Rusack, G. Salina, and R. Tripiccione, "The ape computer: An array processor optimized for lattice gauge theory simulations," *Computer Physics Communications*, vol. 45, no. 1, pp. 345 – 353, 1987. [Online]. Available: http://www.sciencedirect.com/science/article/pii/001046558790172X

[93] C. Battista, S. Cabasino, F. Marzano, P. S. Paolucci, J. Pech, F. Rapuano, R. Sarno, G. M. Todesco, M. Torelli, W. Tross, P. Vicini, N. Cabibbo, E. Marinari, G. Parisi, G. Salina, F. D. Prete, A. Lai, M. P. Lombardo, R. Tripiccione, and A. Fucci, "The ape-100 computer: (i) the architecture," *International Journal of High Speed Computing*, vol. 05, no. 04, pp. 637–656, 1993. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0129053393000268

[94] F. Aglietti, A. Bartoloni, C. Battista, S. Cabasino, M. Cosimi, A. Michelotti, A. Monello, E. Panizzi, P. Paolucci, W. Rinaldi, D. Rossetti, H. Simma, M. Torelli, P. Vicini, N. Cabibbo, W. Errico, S. Giovannetti, F. Laico, G. Magazz, and R. Tripiccione, "The teraflop supercomputer apemille: architecture, software and project status report," *Computer Physics Communications*, vol. 110, no. 1, pp. 216 – 219, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S001046559700180X

[95] F. Belletti, S. F. Schifano, R. Tripiccione, F. Bodin, P. Boucaud, J. Micheli, O. Pene, N. Cabibbo, S. de Luca, A. Lonardo, D. Rossetti, P. Vicini, M. Lukyanov, L. Morin, N. Paschedag, H. Simma, V. Morenas, D. Pleiter, and F. Rapuano, "Computing for lqcd: apenext," *Computing in Science Engineering*, vol. 8, no. 1, pp. 18–29, Jan 2006.

[96] R. Ammendola, M. Guagnelli, G. Mazza, F. Palombi, R. Petronzio, D. Rossetti, A. Salamon, and P. Vicini, "APENet: LQCD clusters a la APE," *Nuclear Physics B-Proceedings Supplements*, vol. 140, pp. 826–828, 2005.

[97] A. Biagioni, F. Lo Cicero, A. Lonardo, P. S. Paolucci, M. Perra, D. Rossetti, C. Sidore, F. Simula, L. Tosoratto, and P. Vicini, "The Distributed Network Processor: a novel off-chip and on-chip interconnection network architecture," *arXiv:1203.1536*, Mar. 2012, http://arxiv.org/abs/1203.1536.

[98] R. Leupers, L. Thiele, A. A. Jerraya, P. Vicini, and P. S. Paolucci, "Shapes:: a tiled scalable software hardware architecture platform for embedded systems," in *Proceedings of the 4th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS '06)*, Oct 2006, pp. 167–172.

[99] R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, P. S. Paolucci, D. Rossetti, F. Simula, L. Tosoratto, and P. Vicini, "QUonG: A GPU-based HPC system dedicated to LQCD computing," in *Application Accelerators in High-Performance Computing (SAAHPC), 2011 Symposium on*, July 2011, pp. 113–122.

[100] R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, P. S. Paolucci, D. Rossetti, A. Salamon, F. Simula, L. Tosoratto, and P. Vicini, "A 34 Gbps data transmission system with FPGAs embedded transceivers and QSFP+ modules," in *Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2012 IEEE*, 2012, pp. 872–876.

[101] R. Ammendola, A. Biagioni, O. Frezza, A. Lonardo, F. Lo Cicero, M. Martinelli, P. Paolucci, E. Pastorelli, D. Rossetti, F. Simula, L. Tosoratto, and P. Vicini, "Architectural Improvements and Technological Enhancements for the APEnet+ Interconnect System," *Journal of Instrumentation*, vol. 10, no. 02, p. C02005, 2015. [Online]. Available: http://stacks.iop.org/1748-0221/10/i=02/a=C02005

[102] R. Ammendola, A. Biagioni, O. Frezza, A. Lonardo, F. Lo Cicero, P. Paolucci, D. Rossetti, F. Simula, L. Tosoratto, and P. Vicini, "APEnet+ 34 Gbps Data Transmission System and Custom Transmission Logic," *Journal of Instrumentation*, vol. 8, no. 12, p. C12022, 2013. [Online]. Available: http://stacks.iop.org/1748-0221/8/i=12/a=C12022

[103] W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *Computers, IEEE Transactions on*, vol. C-36, no. 5, pp. 547–553, 1987.

[104] P. Kermani and L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique," *Computer Networks*, vol. 3, pp. 267–286, 1979.

[105] W. J. Dally, "Virtual-channel flow control," *SIGARCH Comput. Archit. News*, vol. 18, no. 2SI, pp. 60–68, May 1990. [Online]. Available: http://doi.acm.org/10.1145/325096.325115

[106] "Foundation for research and technology, (forth) - hellas," accessed: 20/Oct/2017. [Online]. Available: https://www.forth.gr/

[107] F. Pisani, "Interconnection networks simulations for computing systems dedicated to scientific applications at the exascale," 2015/2016.

[108] "OMNeT++ project." [Online]. Available: https://omnetpp.org/

[109] L. Gravano, G. D. Pifarre, P. E. Berman, and J. L. C. Sanz, "Adaptive deadlock- and livelock-free routing with all minimal paths in torus networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 12, pp. 1233–1251, Dec 1994.

[110] R. Ammendola, P. Vicini, P. S. Paolucci, A. Lonardo, O. Frezza, F. L. Cicero, M. Martinelli, A. Biagioni, and F. Simula, "Sistema per accelerare la trasmissione dati nelle interconnessioni di rete," Patent Pending 102 016 000 071 637, 7 8, 2016, patent Pending n. 102016000071637.