Towards EXtreme scale technologies and accelerators for euROhpc hw/Sw supercomputing applications for exascale: The TEXTAROSSA approach

Giovanni Agosta, Marco Aldinucci, Carlos Alvarez, Roberto Ammendola, Yasir Arfat, Olivier Beaumont, Massimo Bernaschi, Andrea Biagioni, Tommaso Boccali, Bérenger Bramas, et al.

Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale: the TEXTAROSSA Approach*

Giovanni Agosta[a], Marco Aldinucci[f], Carlos Alvarez[h], Roberto Ammendola[n], Yasir Arfat[f], Massimo Bernaschi[c], Olivier Beaumont[g], Andrea Biagioni[j], Tommaso Boccali[l], Berenger Bramas[g], Carlo Brandolese[a], Barbara Cantalupo[f], Mauro Carrozzo[c], Daniele Cattaneo[a], Alessandro Celestini[c], Massimo Celino[b], Paolo Cretaro[j], Iacopo Colonnelli[f], Pasqua D'Ambra[d], Marco Danelutto[e], Roberto Esposito[f], Lionel Eyraud-Dubois[g], Antonio Filgueras[h], William Fornaciari[a], Ottorino Frezza[j], Andrea Galimberti[a], Francesco Giacomini[k], Brice Goglin[g], Abdou Guermouche[g], Francesco Iannone[b], Michal Kulczewski[i], Francesca Lo Cicero[j], Alessandro Lonardo[j], Alberto R. Martinelli[f], Xavier Martorell[h], Giuseppe Massari[a], Simone Montangero[m], Gianluca Mittone[f], Raymond Namyst[g], Ariel Oleksiak[i], Paolo Palazzari[b], Pier Stanislao Paolucci[j], Federico Reghenzani[a], Sergio Saponara[e], Francesco Simula[j], Federico Terraneo[a], Samuel Thibault[g], Massimo Torquati[e], Matteo Turisini[j], Piero Vicini[j], Miquel Vidal[h], Davide Zoni[a], Giuseppe Zummo[b]

[a] *DEIB – Politecnico di Milano, Italy, [name].[surname]@polimi.it*
[b] *ENEA, Italy, [name].[surname]@enea.it*
[c] *Istituto per le Applicazioni del Calcolo (IAC), CNR, Rome, Italy, [name].[surname]@cnr.it*
[d] *Istituto per le Applicazioni del Calcolo (IAC), CNR, Naples, Italy, pasqua.dambra@cnr.it*
[e] *University of Pisa, Italy, [name].[surname]@unipi.it*
[f] *University of Torino, Italy, [name].[surname]@unito.it*
[g] *Inria, France, [name].[surname]@inria.fr*
[h] *BSC, Spain, [name].[surname]@bsc.es*
[i] *PSNC, Poland, ariel@man.poznan.pl, kulka@man.poznan.pl*
[j] *INFN, Sezione di Roma, Italy, [name].[surname]@roma1.infn.it*
[k] *INFN, CNAF, Italy, [name].[surname]@cnaf.infn.it*
[l] *INFN, Sezione di Pisa, Italy, [name].[surname]@pi.infn.it*
[m] *University of Padova and INFN Sezione di Padova, Italy, [name].[surname]@pd.infn.it*
[n] *INFN, Sezione di Roma Tor Vergata, Italy, [name].[surname]@roma2.infn.it*

Abstract

In the near future, Exascale systems will need to bridge three technology gaps to achieve high performance while remaining under tight power constraints: energy efficiency and thermal control; extreme computation efficiency via HW acceleration and new arithmetic; methods and tools for seamless integration of

*This manuscript is an extended version of [1]

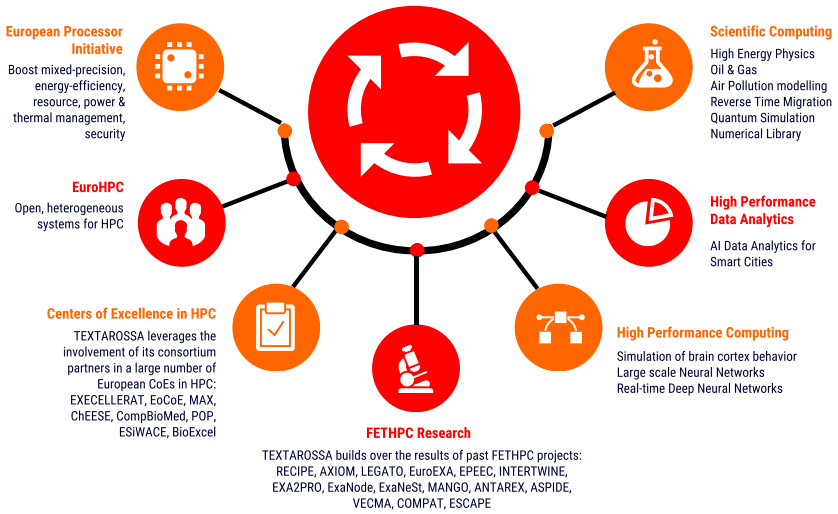

Figure 1: Impact strategy of TEXTAROSSA and positioning within the landscape of the European initiatives on HPC.

reconfigurable accelerators in heterogeneous HPC multi-node platforms. TEXTAROSSA addresses these gaps through a co-design approach to heterogeneous HPC solutions, supported by the integration and extension of HW and SW IPs, programming models and tools derived from European research.

1. Introduction

High Performance Computing is a strategic asset for countries and large companies alike. Such infrastructures are of key importance to support a variety of applications in domains such as oil & gas, finance, and weather forecasting. Recently, emerging domains have been gaining traction, such as bioinformatics, medicine, security and surveillance. These newer applications tend to fall in the classes of High Performance Data Analytics (HPDA) and High Performance Computing for Artificial Intelligence (HPC-AI). The trend in the design of such infrastructures is more and more exploiting heterogeneous hardware architectures to cope with the request of peak performance and to meet the need of achieving a "Green HPC". This paths have prompted Europe to align its research priorities in HPC along a Strategic Research Agenda (SRA[1]) resulting from wide consultations within the European Technology Platform for HPC (ETP4HPC), the PRACE initiative[2], and the PlanetHPC[3] initiative.

[1] https://www.etp4hpc.eu/sra.html (last accessed March 2022)
[2] https://prace-ri.eu (last accessed March 2022)
[3] https://cordis.europa.eu/project/id/248749 (last accessed March 2022)

The need to achieve high efficiency while remaining within reasonable power and energy bounds, is extensively discussed in the SRA, focusing also on the main technology challenges posed by these objectives. Such challenging goal can only be addressed with an holistic approach that takes into account multiple factors across the HPC hardware/software stack, including the use of application-specific, extremely efficient hardware accelerators, efficient software management of resources, data and applications, and efficient cooling systems. Together, these components can provide the desired computational power while keeping under control the power consumption of the supercomputer.

The TEXTAROSSA project originates from such a vision and aims at providing key technological advances on all three aspects and to validate them on new development platforms representative of future HPC systems, using a wide range of applications from different domains, selected considering traditional HPC application fields as well as coming from emerging domains. Figure 1 highlights the strategic goals of TEXTAROSSA, set against the EuroHPC initiative and the broader framework of European research in High Performance Computing. More in detail, TEXTAROSSA will implement the above-mentioned approach by pursing the following goals:

- **Technical goals**

  1. *Energy efficiency and thermal control* via innovative two-phase cooling technology at node and rack level, fully integrated in an optimized multi-level runtime resource management driven by power, energy, and thermal models fed by on-board sensor data.
  2. *Sustained application performance* through efficient exploitation of highly concurrent accelerators (GPUs and FPGAs) by focusing on data/stream locality, efficient algorithms and programming models, tuned libraries and innovative IPs.
  3. *Seamless integration of reconfigurable accelerators* by extending field-proven tools for the design and implementation such as Vitis and OmpSs@FPGA to support new IPs and methodologies such as mixed-precision computing and power monitoring and control.
  4. *Development of new IPs* for mixed-precision AI computing, data compression, security, power monitoring and control, and scheduling.
  5. *Integrated Development Platforms* by developing two architecturally different, heterogeneous Integrated Development Vehicles (IDVs), one as a dedicated testbed for two-phase cooling technology, and one supporting the wider range of project technical goals.

- **Strategic goals**

  1. *Alignment with the European Processor Initiative (EPI)* by testing, extending and boosting key technologies applicable to future EPI evolution.

2. *Supporting the objectives of EuroHPC* as reported in ETP4HPC's Strategic Research Agenda (SRA) for open HW and SW architecture.
3. *Building over European expertise* gained through past research projects as well as through the Centers of Excellence in HPC.
4. *Opening of new usage domains*, including High Performance Data Analytics (HPDA) and High Performance Artificial Intelligence (HPC-AI) applications, alongside support for traditional HPC domains.

1.1. ***The TEXTAROSSA Consortium***

TEXTAROSSA is a three-year project co-funded by the European High Performance Computing (EuroHPC) Joint Undertaking. The project is led by ENEA (Italy) and aggregates 17 institutions and companies, including the linked third parties, located in 5 European countries: CINI, an Italian consortium grouping together three leading universities, Politecnico di Milano, Università degli studi di Torino, and Università di Pisa, Fraunhofer (Germany), INRIA (France), ATOS (France), E4 Computer Engineering (Italy), BSC (Spain), PSNC (Poland), INFN (Italy), CNR (Italy), In Quattro (Italy), Université de Bordeaux (France), CINECA (Italy) and Universitat Politècnica de Catalunya (UPC). The three Italian universities are part of the lab of CINI[4], created in 2021, that is grouping together the main academic and research entities working in the field of high-performance and Exascale computing in Italy. CINI is providing also the technical leadership of the project. More information on the activities carried out during the execution of TEXTAROSSA can be found in the project website[5].

1.2. Organization of the paper

The rest of this paper is organized as follows. In Section 2, we introduce the TEXTAROSSA co-design approach. In Section 3, we describe the key technological innovations provided by the TEXTAROSSA project, while in Section 4 we provide an overview of the application use cases. Eventually, in Section 5, we draw some conclusions and highlight future research directions.

2. TEXTAROSSA co-design approach

From a methodology point of view TEXTAROSSA adopts a co-design process as key strategy for Fast Forward and Exascale computing, considering the entire system stack from underlying technologies to applications. The co-design process concerns five layers covering the whole HPC stack: 1. *User Application*: representing a wide range of scenarios, from mathematical libraries, to miniApps and flagship codes for numerical modelling with massive parallelism

[4] https://www.consorzio-cini.it/index.php/it/laboratori-nazionali/hpc-key-technologies-and-tools (last accessed March 2022)
[5] https://textarossa.eu (last accessed March 2022)

in HPC/HPDA/AI applications. The performance of HPC applications is tied to their level of optimization and the capacity of the underlying tools they use. Research usually focuses on both facets separately: optimizing the applications on one side and improving hardware/middleware layers on the other side. This is not the case with our co-design approach: We aim at connecting both aspects by evaluating the impact of novel technologies on a set of target HPC applications, and, at the same time, studying which improvements at lower levels could be beneficial to these applications. 2. *Runtime Services*: ensuring that application requirements are dynamically satisfied and mapped onto system resources, and including execution models with workload handling, fault tolerance, and data management. These services are well established in traditional HPC data centers, and the users, mainly belonging to academic and big enterprises research, are aware enough to use the computing resources. Because there has been a growth of edge applications in the last years aimed at effectively analyzing big data in a timely manner, a cloud edge continuum enabling HPC/HPDA is becoming a business case. As the levels and fidelity of instrumentation increase and the types and volumes of available data grow, new classes of applications are being explored that seamlessly combine real-time data with complex programming models and data analytics to monitor and manage systems of interest. 3. *Programming Models*: underlying the applications, they define the toolchains and SW development tools able to implement applications in parallel architectures. The HPC community provided several programming models that have demonstrated their potential to develop efficient applications. However, parallelization models are used separately, and most of them target CPUs or combinations of CPUs/GPUs, but not FPGAs. However, FPGAs open new possibilities that can be highly beneficial to almost any HPC application, providing toolchains to handle heterogeneous architectures. 4. *System Architecture*: including the processor core's micro-architecture, the arrangement of cores within a chip, memory hierarchy, system interconnect, and storage subsystems. Future HPC platforms increasingly depend on heterogeneous node architectures to meet power and performance requirements. In the HPC landscape, two main approaches have appeared as viable solutions for a possible system architecture bridging the current gaps in terms of power and performance that are required in Exascale computing: the first approach relies on multi-core processors whose high performance is boosted by the use of GPU-based accelerators; the second approach aims at integrating FPGA-based accelerator within the host architecture. Additionally, interconnection networks featuring very low latency are going to be indispensable to support the high performance of the computation nodes. 5. *Platforms*: concerning the HW platform at node and rack levels, they need to be able to achieve performance requirements in terms of computing power and energy consumption. High-density computing power at node/rack level requires new technologies of direct cooling on the chip able to remove heat with high efficiency in order to reduce energy consumption. Direct cooling provides a more efficient method to transfer the heat from these hot components to the building chilled water loop and then outside with very little additional energy, compared to transferring the heat first to air and then to the building

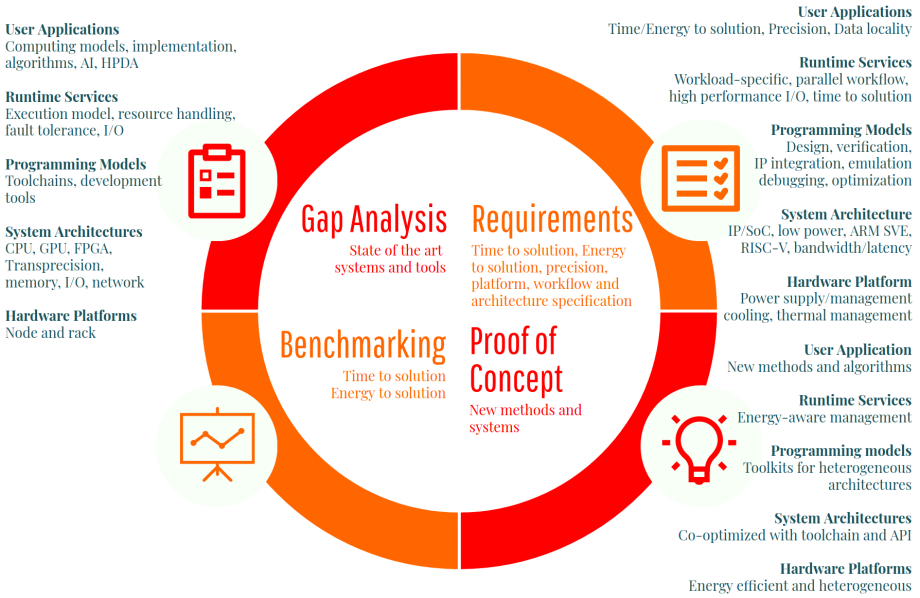

Figure 2: The TEXTAROSSA Co-Design Approach

chilled water system. In addition, in a direct cooling system, the water temperature returning after cooling the IT equipment is much higher than typically found in data centers, and provides more opportunity for heat reuse or the ability to reject this heat to the atmosphere with a dry cooler, thereby eliminating the requirement of a cooling tower or chiller plant in most climates.

Figure 2 provides an overview of the co-design approach adopted in TEXTAROSSA, showing how the five layers of the HPC stack are addressed in each of the four main stages of the co-design process:

1. *Gap Analysis*: to compare the current state-of-art of the technological assets with the objectives of the project in order to identify the gap to be filled by developments or update in co-design process.

2. *Requirements*: to define specifications and requirements of the technological solutions for designing and developing.

3. *Proof of Concept*: to develop HW/SW prototype solutions able to achieve the KPIs (Key Performance Indicators) of the project objectives.

4. *Benchmarking*: to provide performance results of the technological solutions by means of benchmark tools.

3. TEXTAROSSA technologies

TEXTAROSSA develops, starting from the results of previous European research activities mentioned in Figure 1, a set of technologies to deal with

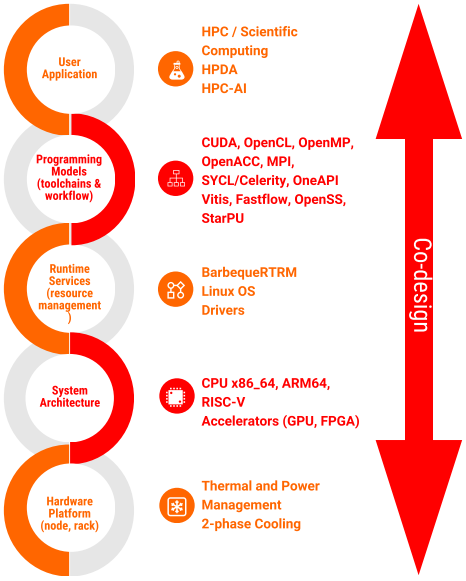

Figure 3: The TEXTAROSSA key technologies

each of the four layers of the HPC stack, as well as applications, which will be covered in full in Section 4. Figure 3 provides an overview of the primary technology bricks adopted by TEXTAROSSA. In the rest of this section, we provide insights on specific technology bricks developed within the project

3.1. Programming Models & Toolchains

Vitis based HLS flow. Vitis [52] is the HLS flow designed by Xilinx to cover all the steps required to translate an application, described through a C/C++ program, into a working bitstream running on an FPGA card and communicating with a program running on the host node. The Vitis environment allows emulating the whole system's behavior (host program and FPGA code) by running and debugging the C/C++ code in a standard IDE. Once the functional correctness has been achieved, i.e., the program produces the expected results, the HLS engine translates the pieces of C/C++ code mapped on the FPGA into equivalent, optimized, hardware implementation. The communication with the external world (host node, memory banks) is achieved through a presynthesized layer implementing the interfaces (PCIe, DDR and HBM memory banks) and the FPGA resources are accessed from the host node through the runtime and APIs developed by Xilinx. Vitis is largely customizable thanks to the open-source LLVM-based front-end[6] and many open-source accelerated libraries (e.g., math, video processing, AI, signal processing)[7]. In the TEXTAROSSA project,

[6] https://github.com/Xilinx/HLS (last accessed March 2022)
[7] https://xilinx.github.io/Vitis_Libraries (last accessed March 2022)

the Vitis environment will be extended with multi-precision arithmetic, allowing the usage of new data types based on the Posit format, and with a communication library used to perform inter-FPGA direct communications. The inter-node communication layer, namely MPI, will also be updated to transfer data without requiring memory copies between FPGAs and hosts. Furthermore, the APIs to access the FPGA will be used to build the TEXTAROSSA APIs that will be defined to access homogeneously the accelerators. Such extensions will require the integration into the Vitis flow of the new IPs defined in the project, thus we will develop the hardware modules, their C++ functional models and we will encode all the necessary info needed by the HLS to properly manage (i.e., schedule and connect) the new functionalities.

StarPU, OmpSs. Task-based programming models (like OmpSs[8] [43] and StarPU[9] [15]) address the challenges to program on heterogeneous computing nodes while achieving high productivity by providing higher-level abstractions that could help the programmer to generate high-performance code.

It has been demonstrated [23] that task-based programming models can simplify FPGA programming by making the memory allocation and data copies automatic and automating the code generation of the CPU and FPGA binaries, provided the C/C++ implementation, by transparently running open or vendor tools. Performance can be subsequently improved by using high-level tools like tracing directly from inside the FPGA [23]. In addition, these programming models allow the use of parallelism based on tasking (instead of kernel invocations) and can even provide support for data-dependent tasks and manage the execution based on such data dependencies [66]. We plan to move part of the runtime work to a fast hardware task scheduler [66], [40] to further enhance the performance of these programming models. It has already been shown [40] that these improvements can lead to obtain top performance of some applications in FPGAs directly from high-level language programs. In addition, we plan to extend the range of programs that demonstrate this optimum results with the tools used by further adapting the environment to the TEXTAROSSA platforms. New features obtained from these platforms will then be ported back to framework in order to benefit the whole range of systems targeted by the tools.

Training Deep Neural Network (DNN) is a memory-intensive operation. Indeed, the training algorithms of most DNNs require to store both the model weights and the forward activations in order to perform back-propagation. In practice, training is performed automatically and transparently to the user through autograd tools for back-propagation. Unfortunately, the memory limitation of current HW often prevents data scientists from considering larger models, larger image sizes or larger batch sizes, especially in recent NLP models [42]. Our goal is to extend StarPU to enable inference and learning, taking advantage of both the heterogeneity of the architecture to place layers and the

[8] https://github.com/bsc-pm-ompss-at-fpga (last accessed March 2022)
[9] https://starpu.gitlabpages.inria.fr (last accessed March 2022)

memory architecture to minimize transfers. In particular, we will rely on the RoToR framework[10] [16] which allows to control the memory consumption and to minimize the energy consumed by the data exchanges.

Several problems should be solved to use FPGAs more efficiently and control energy consumption. In the TEXTAROSSA project, we will extend these two task-based runtime systems, OmpSs and StarPU, which use different approaches to support FPGA. This will allow the integrated development vehicle to benefit from their existing features, but also to study their complementarity while validating the robustness of the new HW against different runtime systems.

FastFlow. Stream processing is gaining increasing industrial attention for real-time data analytics and data-driven applications [9]. FastFlow [8] is a C++ programming library targeting multi/many-cores. It offers both a set of high-level ready-to-use parallel pattern implementations and a set of mechanisms and components (called building blocks) to support low-latency and high-throughput data-flow streaming networks. FastFlow simplifies the development of parallel applications modeled as a structured, directed graph of processing nodes. The graph of concurrent nodes is constructed by the assembly of sequential and parallel building blocks and higher-level components (i.e., parallel patterns) modeling recurrent schemas of parallel computations (e.g., pipeline, task-farm, parallel-for, etc.). FastFlow efficiency stems from the optimized implementation of the base communication and synchronization mechanisms and its layered software design. Besides, stream processing is the natural paradigm for event-driven distributed applications that need to communicate with each other via message passing. Finally, some data streaming paradigms are naturally suited for implementation on reconfigurable platforms [56], e.g. the dataflow/actor paradigm. In TEXTAROSSA, we aim at exploiting reconfigurable platforms to accelerate HPDA tasks leveraging the FastFlow framework [7].

Streamflow. The StreamFlow framework [36, 37] is a container-native Workflow Management System (WMS) written in Python 3 and based on the Common Workflow Language (CWL) Standard [14]. StreamFlow has been designed around two main principles: 1. Allowing the execution of tasks in multi-container environments, in order to support concurrent execution of multiple communicating tasks in a multi-agent ecosystem; 2. Relaxing the requirement of a single shared data space, in order to allow for hybrid workflow executions on top of multi-cloud or hybrid cloud/HPC infrastructures. StreamFlow source code is available on GitHub under the LGPLv3 license. A Python package is downloadable from PyPI and Docker containers can be found on Docker Hub. More details about the tool and its applications can be found in the StreamFlow website[11].

[10] https://gitlab.inria.fr/hiepacs/rotor (last accessed March 2022)
[11] https://streamflow.di.unito.it (last accessed March 2022)

Compiler Technology for Mixed-Precision Support. Many applications of HPC are error-tolerant, a characteristic that can be exploited – either in hardware or in software – to achieve important savings in system costs or power efficiency improvements [69]. Hardware-based proposals take advantage of inherent sensor limitations, redundant data, and reduced precision input, or introduce additional uncertainty by adopting design features that produce approximate results. On the other hand, software-based approximate-computing approaches such as floating-point optimization or loop perforation allow to trade-off algorithm exactness for a more efficient implementation [65]. To this end, high complexity of modern applications makes approximate compilers increasingly important. We aim to focus on mixed-precision compilers [29], which are a subset of approximate compilers. Recently, new data types specifically geared for mixed-precision are emerging, such as Posit [49] and BFloat16 [26]. Posit are a new compressed floating-point data format for which University of Pisa has developed a SW library called CppPosit [34, 33]. From the first results of applying the CppPosit library to AI/DNN problems, Posit can lead to the same processing accuracy of single-precision floating-point but with a data compression from a factor 2 to 4 [34, 33]. This means that applying Posit to the application cases (HPC, HPDA and AI/CNN) has the potential to reduce data storage issues and allows for fast data movement. BFloat16 (Brain Floating Point) is used in upcoming Intel AI processors (NERVANA), XEON processors, Google Cloud TPU and ARMv8.6-A, as well as in RISC-V extensions [77]. In contrast to other standardized 16-bit floating point formats, BFloat16 offers a greater dynamic range and higher compatibility with the conventional single-precision floating-point format defined by the IEEE-754 standard. In TEXTAROSSA, we aim at exploiting and extending the tools for precision tuning developed as part of the H2020 FETHPC ANTAREX project [64] collected in the TAFFO framework [30, 31]. The TAFFO framework is implemented as a set of plugins for the LLVM compiler. Based on programmer hints expressed as attributes, TAFFO performs value range analysis, data type and code conversion, and static estimation of the performance impact, automatically producing a mixed-precision application with statically-guaranteed error bounds. TAFFO is language-independent, supports data types ranging from fixed-point to standard floating-point formats, and allows the user to finely tune the performance-precision trade-off to their needs [28]. The extensions to TAFFO will allow it to cover a wider range of target platforms, such as FPGAs through integration with the TEXTAROSSA High Level Synthesis (HLS) toolchain. Additionally, we aim to include support for emerging data types such as Posit and BFloat16, expanding the use of the tools to heterogeneous systems with reconfigurable components, and to improve the performance estimation by exploiting recent analysis techniques [39] as well as a deeper understanding of the target processor pipeline.

3.2. Runtime Services: Energy/Power Management

Due to the end of the Dennard's scaling, we expect a linear increment of power consumption of the next-gen Exascale computing platforms. This would represent a non-sustainable scenario, in terms of impact on the costs of the

HPC centers and overall increase of the carbon footprint of the information technologies. The design and deployment of energy-efficient HPC infrastructures is therefore a primary concern. In this regard, the scientific community converged on the idea that providing computing nodes equipped with heterogeneous processing is the way to follow. This means that an HPC must be characterized by the presence of different processing units, such that, we can allocate the most efficient computing resources with respect to the specific application to serve. However, energy efficiency can then be maximized if we can can rely on a suitable resource management strategy at different levels. Although the state-of-the-art already includes some solutions, recent projects, like MANGO [46] and RECIPE [47], have shown that we need to take into account the platform-specific characteristics, other than develop suitable knobs to profile and monitor the execution of the workload, at runtime [72]. This enables more accurate resource management policies [3, 24]. In addition, by integrating the programming model with the resource management layers we could dynamically tune the numerical accuracy (precision) of the algorithms' implementation, with respect to the actual application requirements and power/energy constraints [62]. Similarly, this applies to the problem of guarantee real-time requirements to time-critical applications [61, 63]. In general, given the reference hardware platform and the application use-cases, the TEXTAROSSA project would represent an extremely interesting testbench for exploring novel power and energy management strategies, by operating at both software and hardware level. At the hardware level, for example, we can provide a major contribution, by instrumenting the computing architecture with ad-hoc power monitors [74] and controllers [71, 60], in order to minimize the monitoring latencies, while increasing the effectiveness of the management policies. On the software side, we start from an already existing resource management framework [17], and aim at extend it with the support for the new target hardware, the integration of the precision tuning mechanisms and the introduction of new platform-specific resource management policies. Overall, this would allow us to explore all the possibilities offered by the platform, and the application-side integration, to increase the FLOPS-per-Watt ratio, with respect to state-of-the-art HPC solutions.

3.3. Posit Hardware Accelerators

Within EPI and the European programs, accelerators based on RISC-V cores enable energy-efficiency for applications using stencil patterns or neuromorphic algorithms. Such accelerators implement floating point units able to work on classic fp32 float or new BFloat16 formats. To increase performance, one could try to reduce the number of bits in the floating point representation. But when reducing the precision of the used arithmetic, iterating over many timesteps, the derivations may become unstable and hence affect the final result of the simulations. The novel Posit binary arithmetic format can offer higher precision while using less bits than standard IEEE floating-point numbers. Recent literature shows [34, 33] that 16bit Posit can leverage comparable results like fp32 and Posit with 8bit precision outperform in terms of accuracy fp16 (for

CNN 8bit Posit can leverage comparable results like fp32). Calculations can be done even with simple bit manipulations on the Posit format without extraction, further decreasing the complexity of the operations. It is thus possible to enhance memory bandwidth, and lower level cache utilization, power footprint, and throughput of the arithmetic units.

Within TEXTAROSSA, a RISC-V unit will be extended with support for alternative data representations, including fine-grained reduced precision floating point [77] and Posit arithmetic. To complement the hardware IP developments, the LLVM compiler, also adopted in EPI, will be extended for Posit and data compression support and real-world HPC applications and CNN kernels will be ported to leverage the IP. Fast software co-design will be enabled through software implementation of Posit in the CppPosit library[12]. Two approaches will be followed: the first called light PPU (Posit Processing Unit) will add Posit to/from float and Posit to/from integer converter to a RISC-V 64b 6 stages core with ALU and FPU. This way, with minimal circuit overhead, estimated in less than 1%, and exploiting the instruction customization of RISC-V, the support of Posit8 and Posit16 is also added. Since with the light PPU, Posit processing replies on ALU or FPU back-end the advantage is mainly in the compression properties of Posits that allows a compact storage of large CNNs, while the inference part will be done using the FPU with floats. In the second approach a full PPU will be added to the RISC-V 64b core, replacing the FPU. This version will be suited to use Posits for both storage and inference. Beside the implementation of accelerators for new data formats, other analysis will be carried out to provide basic blocks to implement novel cryptosystems onto FPGAs, like multipliers for large binary polynomials [76] and decoders for post-quantum cryptography [75].

The generated IPs will be considered for the sake of the global energy and power management and optimization, thanks to the exploitation of a methodology capable to automatically identify the power model of a generic hardware module and to augment the original design with its RTL implementation [73]. Note that such power monitors are properly designed in order to prevent the creation of an entry point for side-channel attacks.

The novel IPs will be ported to FPGAs for benchmarking. Both techniques have not been implemented on top of a completely co-designed accelerator so far and will provide a huge benefit for the European IP portfolio.

3.4. Hardware Platform Optimization

HPC systems have historically always been limited by thermal considerations and computing architectures need both optimized heat dissipation solutions and runtime thermal control policies to operate reliably and efficiently. The TEXTAROSSA project will provide contributions in both areas.

For what concerns heat dissipation improvements, the company InQuattro has developed an innovative thermal management solution (patent pending)

[12] https://github.com/eruffaldi/cppPosit (last accessed March 2022)

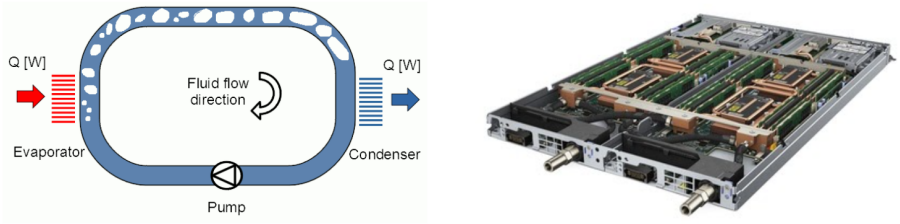

Figure 4: Schematic of the two-phase cooling system (left) and its embedding in a server (right).

based on two-phase mechanically pumped loops, which uses a flow boiling heat transfer for cooling electronics in a more efficient way. This allows the use of the latent heat of vaporization so that flow rates are significantly reduced, temperature gradients are kept small, and heat transfer coefficients are increased compared to not only air cooling but also liquid cooling systems. Two-phase cooling systems using evaporation and condensation are known to be the best way to meet demanding cooling requirements in terms of compactness, weight, and energy-consumption. A possible configuration of the two-phase cooling system for a single node is shown in Figure 4.

In TEXTAROSSA, an optimized two-phase cooling solution for HPC is being developed and customized to fit the requirements of node and rack levels for Exascale applications. The development cycle consists of an initial cooling solution design phase complemented by the design of a dynamic model to be used for simulation, as well as for the design of thermal control policies. The following step consists of the actual production of a prototype cooling solution, to be fully characterized resulting in refinements of its model through experimental data. The final step will be the testing on HPC node(s) provided by E4 and ATOS and developed with the objective to serve an entire rack. Such a testing will be performed jointly with the thermal control policies. It is foreseen to develop two solutions of the two-phase cooling system that could be patented during the project on the principal components of the cooling system (evaporator, condenser). This innovative technology is expected to improve the cooling efficiency up to 70% compared to traditional air cooling, and up to 30% compared to existing liquid cooling and will be tested on both ATOS and E4 infrastructures.

It should be stressed that both the design of cooling solutions and thermal control policies critically relies on thermal models. Thermal simulators for CPUs/MPSoCs have been proposed, but most of them can only represent a limited range of heat dissipation solutions, and are not easy to extend towards two-phase liquid cooling solutions or to encompass the simulation of an entire rack. For this reason, in TEXTAROSSA, we are taking advantage of a recently introduced paradigm for the modeling of MPSoC cooling systems [68]: the co-simulation of highly optimized IC thermal model with heat dissipation models expressed by means of equation-based object-oriented modeling languages. This

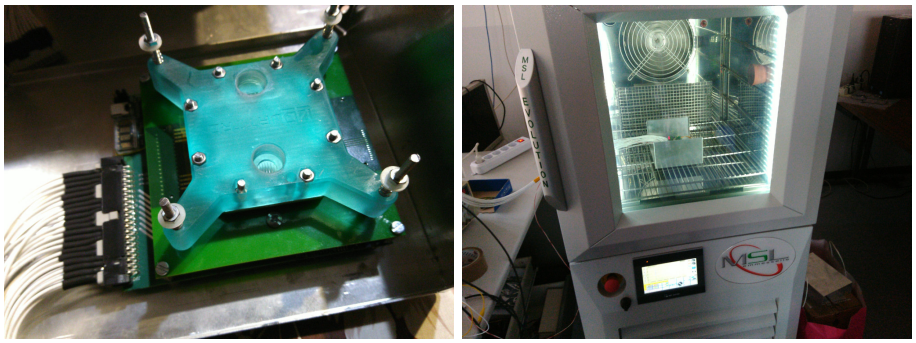

Figure 5: Two-phase evaporator prototype installed on the Thermal Test Chip (left) and thermal chamber calibration of the TTC on-die temperature sensors (right).

approach allows at the same time to leverage the performance of traditional IC thermal model, and to be able to simulate the complex, nonlinear thermal phenomena that arise when bringing evaporative cooling into play.

A key aspect of thermal modeling is the need for experimental data. As some of the the thermal phenomena to be modeled rely on empirical correlations, experiments are needed to provide correct parameter values in order to accurately model heat dissipation phenomena. Moreover, the availability of high quality experimental data allows to perform validation of thermal models. However, the direct use of MPSoCs such as processors or GPUs to perform thermal experiments is made extremely difficult by the uncertainty in the power spatial distribution across the silicon die during computational workloads in modern processors, as well as due to the insufficient number of temperature sensors to fully reconstruct the temperature spatial distribution. In TEXTAROSSA, we are overcoming these limitations by relying on a thermal test chip platform [67] developed at Politecnico di Milano. Thermal test chips are integrated circuits dedicated to thermal testing, providing an array of on-silicon heating elements and temperature sensors, allowing to capture accurate silicon thermal maps connected to the proposed heat dissipation solution, considering both spatial (e.g. hot spots) and temporal temperature variations. The integration of the two-phase cooling system with the TTC is currently underway (Figure 5). The collected experimental data will be used to improve and validate the thermal models as well as for the design of the control policies.

For what concerns thermal control policies, in TEXTAROSSA the integration of two-phase cooling will be made by means of a multilevel thermal control strategy, aiming to overcome the complexity of controlling an HPC platform from node to system level. To reduce overhead, we will exploit and extend the use of event-based control policies (patent pending) developed at Politecnico di Milano to provide effective thermal control with minimal overhead. As the fastest temperature gradients occur at the silicon active layer, we will use fast event-based control loops [53] acting on DVFS to limit the maximum operating temperature of compute elements. These inner control loop will in turn

interact with higher level control loops operating the two phase cooling infrastructure of the node, which is comparatively slower and has higher overheads but has the capability to increase the heat transfer coefficient on-demand, thus allowing to relieve the need to reduce frequency using DVFS, in turn improving performance. A further supervisory control layer will allow to set the desired temperatures at the rack level based on reliability metrics. Multilevel control allows thus to partition the system level control problem into multiple interacting control loops, each optimized for the specific thermal dynamics to control.

3.5. Low-latency Communication between FPGAs

The usage of FPGAs as accelerators is getting so widespread that even big cloud providers are now installing reconfigurable devices in their instances (e.g. on Microsoft Azure and Amazon EC2). Interaction of hundreds to thousands of FPGAs require a scalable approach to hold them together, allowing a low latency connection among them, but a definitive approach has to be found to let users make the most of their flexibility and in the meanwhile easing the usage for software developers. As an example, the latest version of the Microsoft Catapult fabric, puts a Stratix 10 device between each NIC on the x86 servers and the ToR switch, enabling a fast path for accelerators to communicate among themselves with a few microseconds latency. The Brainwave project [32] leverages this architecture to provide a deep learning platform for real-time AI inference on the cloud. While this framework offers a very friendly interface for users to deploy their models on top of this architecture, it loses the flexibility of delivering the cores as black boxes, and providing an implementation of only a few pre-trained models. Our approach, on the other hand, let users full control of the platform, allowing the implementation of custom processing tasks on FPGAs, still maintaining ease of usage by supplying a set of interfaces to integrate with the HLS tools developed in the project. This will allow developers to define a scalable application using a streaming programming model (Kahn Process Network [50]) that can be efficiently deployed on a multi-FPGAs system. TEXTAROSSA is developing a communication IP and its software stack, providing the implementation of a direct network that allows low-latency communication between processing tasks deployed on FPGAs, even hosted in different computing nodes. The communication IP, based on the ExaNet IPs (switch, router, high-speed channels [13]) developed in ExaNeSt H2020 project [51] and EuroEXA H2020 [21], can be split into two Interface Blocks — InterNode IF and IntraNode IF – which manage data flow, and a switch_component, in charge of dynamically interconnect all ports of IFs (Figure 6).

The Communication IP meets the interface requirements to be used as an RTL kernel within the Vitis IDE.

To better take advantage of the framework and to easily allow the interconnection of computational kernels, we are developing a tool to simplify the inclusion and the configuration of the interconnection IP without the need to add an ad-hoc custom project. The user can specify the number of the ports as parameter, as well as how to connect the different kernels to the available ports in a custom configuration file. This configuration file is then used to generate

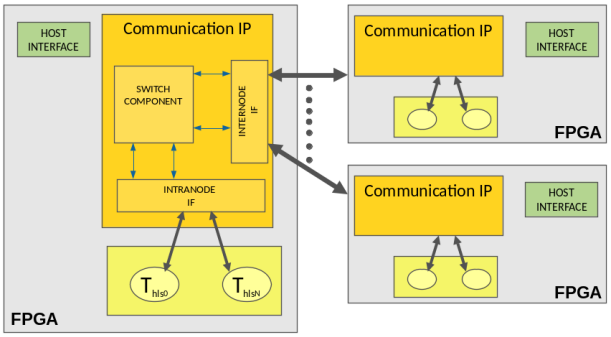

Figure 6: Architectural partition of Communication IP

the project files. This way the interconnection logic is built automatically by the configuration tool based on the application needs, allowing the end user to focus on the processing / computing kernels. The direct communication between tasks deployed on FPGAs avoid the involvement of the CPUs and system bus resources in the data transfers, improving the platform's energy efficiency and reducing communication latency.

4. TEXTAROSSA applications

In this section, we briefly provide an overview of the use case applications adopted in TEXTAROSSA. To address the variety of application domains of future Exascale systems, TEXTAROSSA applications include basic mathematical building blocks (*MathLib*), traditional HPC applications (*UrbanAir*, *TNM*, *HEP*, and *RTM*), and applications from emerging domains (*RAIDER*, *DPSNN*, *Danger Detection*).

4.1. MathLib

One of the main requirements for an energy efficient computing is the optimization and the acceleration of algorithms and SW libraries to provide a reduction of the elapsed time of HPC applications and thus a significant cut in energy consumption. The new power-to-solution metrics requires a rethinking of many computational kernels of HPC applications looking for a trade-off between the reduction of the total energy and the minimization of the time-to-solution, promoting scalability. Within this context, extensions and improvements of high-performance algorithms and SW libraries for kernels in numerical linear algebra [6, 35] and graph computation, such as iterative [20, 18, 19, 38] and direct

linear solvers, edge weighted graph matching, and fast multipole methods [5] will be deployed. In more details, we will provide new high-performance algorithms and software modules for some of the so-called Colella's dwarves, that classified numerical methods crucial for science and engineering. In particular, we will focus on algorithms and software for sparse linear algebra, where data sets include many zero-values and are usually stored in compressed data structures to reduce storage and memory bandwidth requirements. Those data structures are generally accessed with indexed loads and stores, and main computational kernels are communication bound. We intend to develop and benchmark a library of kernels optimized for hybrid nodes embedding NVDIA GPUs in order to pursue node-level performance further than multi-node scalability. Scalability will be improved by leveraging overlap between communication and computation. We will also investigate the chance of using mixed-precision floating-point arithmetic, which offers many advantages in terms of memory footprint and computational efficiency, by applying strategies for preserving robustness and correctness, such as iterative refinement to meet user's accuracy and reproducibility needs. The library kernels will be of immediate use in a wide range of applications, ranging from classical scientific simulation to AI techniques, including automatic pattern recognition in complex systems, and will be tested in some of the applications proposed as use cases in this project.

4.2. HPDA and HPC-AI Applications

Real-time AI-based Data analytics on hetERogeneous distributed systems (RAIDER). A Proof-of-Concept shall be provided as a typical High Energy Physics (HEP) real-time AI-based data analytics on heterogeneous distributed system, a processing paradigm that our research group has already successfully adopted in the context of the NA62 experiment at CERN where we have developed the GPU-RICH system [11, 10]. GPU-RICH integrates one CPU, one FPGA and one GPU in a single computing node and implements a real-time track reconstruction pipeline in which the FPGA-based NIC receives eight data streams from the RICH Cherenkov detector, processes them and injects the output stream directly into the GPU memory through the PCIe bus (GPUDirect RDMA) for the track reconstruction parallel processing. Future experiments, which will handle high volume of data and high costs, push the need for new techniques in Trigger and Data Acquisition (TDAQ) systems to improve particle identification and further suppress background events in trigger systems, or to perform an efficient online data reduction for trigger-less ones. Architecturally, data streams from different channels/sources/detectors can be recombined through some processing layers using a low-latency, modular and scalable network infrastructure (configurable in number of channels, topology and size). Each processing layer performs feature extraction through machine learning leveraging Deep Neural Networks implemented on heterogeneous devices. This implementation must take in account the limited memory and floating point resources of some of the devices (usually those located at the edge) deploying resource-demanding Neural Network layers (e.g. CNN) in subsequent processing layers and studying reduced precision and/or compression techniques

for input data. These concepts define the RAIDER software application and its reference hardware platform, whose combination yields a distinguishing feature that is the capability of performing this distributed inference scheme with real-time constraints. FPGA devices are the key architectural elements enabling the implementation of the general RAIDER architecture in application scenarios characterized by very low-latency classification requirements. In fact, these devices allow the implementation of data transport and processing stages characterized by a highly predictable and low latency. Wrapping up overview and studies concerning available low-latency communication IPs and frameworks for Neural Networks (NNs) deployment on FPGA to finalize a preliminary testbed for a distributed data-analytics system. Henceforth having the possibility to evaluate pros and cons of a programming model focused on the realisation of a real-time AI-based data analytics heterogeneous distributed system. For this purpose, a distributed HLS development framework is required, hence we will work toward the extension of one of those frameworks, namely Xilinx Vitis, in order to be able to deploy in a straightforward manner multi-FPGA distributed low-latency applications, such as RAIDER. It shall be done integrating INFN set of Interconnection IPs (switch, low-latency data channels) in the Xilinx Vitis framework also through the definition and implementation of the full software stack supporting those IPs for the very low-latency data transfer between processing tasks deployed on the same FPGA (intra-node communication) and on different FPGAs (inter-node communication), in order to offer hardware support for the execution of applications developed according to the streaming programming models on a system made of multiple interconnected FPGAs. The careful design and implementation of the HW/SW interface, through the Vitis HLS flow to define the communication protocol (HLS communication primitives) and to map the I/O ports of the Vitis HLS flow on those provided by the communication IPs will be of great importance to fully exploit the potential of the hardware.

Brain Simulation (DPSNN). INFN started tackling neural simulations with its own engine, the Distributed and Plastic Spiking Neural Network (DPSNN), which is a scalable C++/MPI code for HPC platforms at extreme scales simulating the spiking dynamics of a brain cortex modeled as a grid of cortical columns populated with neurons and their interconnecting synapses. It has been used to first model brain cortex behaviour – with a special focus on sleep-like states [58] – and to gauge compute and power efficiency on different architectures [12]. INFN has now transitioned to another, more versatile tool, the NEST Simulator [41]. This is a C++/MPI/OpenMP code by the NEST Initiative[13] that empowers an user with a domain-specific language to design a virtual neurophysiology experiment, from the equations driving the dynamics of the components of interest in the cortex (with a rich library of many types of either neurons and synapses ready to be used) to the topology of their intercon-

[13] https://nest-initiative.org (last accessed March 2022)

nections – the so-called *connectome* – all the way to the necessary supporting tools, like probing or stimulating electrodes that read or inject electrical currents into the simulated cortex. NEST offers to the experimenter an intuitive Python interface to easily setup a detailed and complex protocol of interaction between a simulated cortex and a set of external stimuli. Coupled with the huge set of tools for analysis, visualization and data transformation available to the Python user, NEST allows for a compact yet expressive way to perform even very involved neural simulations. INFN has used NEST to implement a biologically-inspired thalamo-cortical model which is able to be trained in classification of handwritten digits from the MNIST dataset and then mimic the wake-sleep cycle, in order to test the enhancing effects of sleep on the quality of learning and recognition [27], even in noisy environments [48].

Smart Cities video surveillance. Implementing distributed video surveillance systems in the context of smart cities provides a huge amount of data for processing, using also AI techniques for detection and classification. Possible applications are related to safety versus Covid-19, e.g., by detecting in real-time people, their body temperature, the respect of social distancing, or if they are wearing protective facial masks. A design activity will carried out for a real-time danger alarm system composed of a network of smart cameras where a pre-processing stage plus an AI algorithm (e.g. a Yolo based CNN) is implemented on an EDGE server.

4.3. Traditional HPC Applications

Air Pollution (UrbanAir). The UrbanAir concerns the modelling and forecasting of the concentration and dispersion of pollutants. It is a 3D multiscale model that combines a numerical weather prediction (NWP) model, running at larger scale (e.g. mesoscale), with a city-scale geophysical flow solver (EULAG) for accurate prediction of contaminant (e.g. NO2, PM2.5, PM10) transportation through the street corridors, over buildings and obstacles. A design activity shall be carried out to use mixed-precision computing and energy-efficient accelerators for faster response while preserving results accuracy. The EULAG is a Fortran application which exploits message passing parallelization in two (2P) or three (3P) dimensions. In each version, the computational domain is balanced between CPU workers - horizontally for the former version, horizontally and vertically for the latter. Some of the kernels of 3P version have been already adapted to GPU to reduce processing time. The computational domain can be divided between GPU accelerators only or between CPUs and GPUs. In the latter case the subdomain sizes are carefully chosen to avoid unbalanced computations. Moreover, the 3P version has been adapted with the BarbequeRTRM runtime resource manager to manage computing resources allocation, to dynamically switch between CPUs and GPUs to increase energy efficiency, and to increase reliability with checkpoint/restore mechanism [2].

Quantum Simulation (TNM). Tensor Network Methods (TNM) are a class of powerful numerical methods developed to study the equilibrium and out-of-equilibrium properties of strongly correlated many-body quantum systems [57].

TNM are complementary to Monte Carlo methods as they do not suffer from the sign problem. A design activity shall be performed to overcome current TNM limitations studying how to push the simulation boundaries towards high-dimensional systems with the support of HPC infrastructure. As important steps in this direction, the first tensor network simulations of lattice gauge theories in (2+1) and (3+1) dimensions have been performed [44, 55]. In particular, lattice Quantum Electrodynamics (QED) in the Hamiltonian formulation including dynamical matter has been considered and, by using the sign-problem-free TNM, it has been possible to compute the ground states of the model at zero and finite charge densities, and to address fundamental questions such as the characterization of collective phases of the model, the presence of a confining phase at large gauge coupling, and the study of charge-screening effects. These simulations have been performed on computer clusters by taking advantage only of OpenMP parallelization on single multi-core nodes. Further developments shall be carried out to simulate larger system sizes, by exploiting large-scale parallelization of TNM by means of MPI or GPU acceleration.

High Energy Physics (HEP). The HEP community has the transition to heterogeneous computing on its roadmap for the next decade, in order to profit from the large investments in HPC systems, and in general to access resources with a better cost/performance ratio [22]. A number of solutions have appeared in the last 4-5 years in the market; they, with their own peculiarities and strong and weak points, promise a seamless utilization of a variety of platforms, with a clear separation between a frontend part (seen by the user/programmer) and a backend part (taken care by the framework via a toolset of libraries and compilers). The HEP community, and in particular the LHC experiments, have started an experimentation with a few products, like Alpaka [70], Kokkos [59], Intel OneApi and SYCL. In TEXTAROSSA, we plan to select a representative set of applications, mission critical in their scientific domains, and evaluate the porting to the frameworks. Whenever successful, a thorough benchmarking will follow on the main target platform, and auxiliary ones. A first target for migration is certainly the collection of high-level software libraries used by the LHC experiments for simulation and data analysis. They include Geant4 [4] and Fluka [45] for particle-matter simulation, the use of high-level analysis tools like those in ROOT [25], and simulation packages of high-energy collisions. A design activity will be focused to optimize these software frameworks for the realization of code bases able to execute on multiple architectures, including the next generations of pre-Exascale and Exascale European HPC systems.

Biomedical Application (HPC-Drugs). Ligand binding affinity predictions carried out with Molecular Dynamics simulation is one of the main research focus in computational chemistry today due its potential impact in industrial drug discovery. A design activity shall be carried out for HPC-backed pharmaceutical applications based on n-body kernel functions running in specialized cores of GPU and FPGA, relying on recently discovered non-equilibrium thermodynamics theorems and capable of delivering absolute binding free energies of drug-size

__kernel void kforce(global double i-th part.);
for *each particle j-th* **do**
| evaluate the force on j-th part. by i-th part.;
end
end __kernel kforce;

Algorithm 1: n-body kernel function algorithm.

molecules in a predictable wall-clock time with a credible confidence interval, hence bypassing the limitations of the traditional equilibrium-based Free Energy Perturbation (FEP) alchemical approaches [54].

Heterogenous architecture based on GPU/FPGA is already used to reduce the huge processing time in MD simulations. Its programming model provides a top-level abstraction for low-level hardware routines as well as consistent memory and execution models for dealing with massively-parallel code execution. A standard programming model is OpenCL, which is composed of one CPU-based "Host" controlling multiple "Compute Devices" such as GPUs and FPGAs. Each of these coarse-grained compute devices consists of multiple "Compute Units", and within these are multiple "Processing Elements". At the lowest level, these processing elements all execute OpenCL "Kernel Functions". These kernel functions compute the forces by involving in an n-body interaction a particle system on which all coordinates are fixed during the force computation that can be parallelized for the different values of each particle. In terms of streams and kernels, this can be expressed as the kernel function shown in the pseudo-code of Algorithm 1.

Reverse Time Migration (RTM). The Reverse Time Migration application and mini-kernels are used within EPI to co-design the STX Accelerator and have been ported to FPGAs within the EuroEXA project. The respective kernels will be analysed to which extent they can leverage the new, energy-efficient, capabilities like Posit arithmetic and lossy compression to enhance performance and energy efficiency. The RTM kernels are stencil-based kernels. Hence, they provide conclusions on many stencil based applications. Reverse Time Migration by FHG for HPC applications to Oil & Gas and Geo-Services.

5. Conclusions

The EuroHPC TEXTAROSSA project addresses technology gaps towards pre-Exascale and Exascale scenarios, developing new IPs, algorithms, methods and software components for HPC-AI, HPC and HPDA applications. The majority of the TEXTAROSSA tools will be open-source and able to be adopted as standalone building blocks or to interoperate with other Exascale-ready components developed within the EuroHPC initiative.

Acknowledgements

This work is supported by the TEXTAROSSA project (G.A. n. 956831), as part of the EuroHPC initiative.

References

[1] Giovanni Agosta, Daniele Cattaneo, William Fornaciari, Andrea Galimberti, Giuseppe Massari, Federico Reghenzani, Federico Terraneo, Davide Zoni, Carlo Brandolese, Massimo Celino, et al. Textarossa: Towards extreme scale technologies and accelerators for eurohpc hw/sw supercomputing applications for exascale. In *2021 24th Euromicro Conference on Digital System Design (DSD)*, pages 286–294. IEEE, 2021.

[2] Giovanni Agosta, William Fornaciari, David Atienza, Ramon Canal, Alessandro Cilardo, José Flich Cardo, Carles Hernandez Luz, Michal Kulczewski, Giuseppe Massari, Rafael Tornero Gavilá, and Marina Zapater. The recipe approach to challenges in deeply heterogeneous high performance systems. *Microprocessors and Microsystems*, 77:103185, 2020.

[3] Giovanni Agosta, William Fornaciari, Giuseppe Massari, Anna Pupykina, Federico Reghenzani, and Michele Zanella. Managing heterogeneous resources in hpc systems. In *Proceedings of the 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms*, PARMA-DITAM '18, page 7–12, New York, NY, USA, 2018. Association for Computing Machinery.

[4] S. Agostinelli et al. GEANT4: A simulation toolkit. *Nucl. Instrum. Meth.*, A506:250–303, 2003.

[5] Emmanuel Agullo et al. Task-based fmm for heterogeneous architectures. *Concurrency and Computation: Practice and Experience*, 28(9):2608–2629, 2016.

[6] Emmanuel Agullo et al. Achieving high performance on supercomputers with a sequential task-based programming model. *IEEE TPDS*, 2017.

[7] Marco Aldinucci et al. Design patterns percolating to parallel programming framework implementation. *International Journal of Parallel Programming*, 42(6):1012–1031, 2014.

[8] Marco Aldinucci et al. Fastflow: high-level and efficient streaming on multicore. In *Programming Multi-core and Many-core Computing Systems*, Parallel and Distributed Computing, chapter 13. 2017.

[9] Marco Aldinucci, Salvatore Ruggieri, and Massimo Torquati. Porting decision tree algorithms to multicore using FastFlow. In *Conference in Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, volume 6321 of *LNCS*, pages 7–23, September 2010.

[10] R Ammendola, M Barbanera, A Biagioni, P Cretaro, O Frezza, G Lamanna, F Lo Cicero, A Lonardo, M Martinelli, E Pastorelli, P S Paolucci, R Piandani, L Pontisso, D Rossetti, F Simula, M Sozzi, P Valente, and P Vicini. Real-time heterogeneous stream processing with NaNet in the NA62 experiment. *Journal of Physics: Conference Series*, 1085:032022, sep 2018.

[11] Roberto Ammendola et al. NaNet: a flexible and configurable low-latency NIC for real-time trigger systems based on GPUs. *Journal of Instrumentation*, 9(02):C02023, 2014.

[12] Roberto Ammendola et al. The brain on low power architectures-efficient simulation of cortical slow waves and asynchronous states. *Advances in Parallel Computing*, 32:760=769, 2018.

[13] Roberto Ammendola et al. Large scale low power computing system: Status of network design in exanest and euroexa projects. *Advances in Parallel Computing*, 32:750–759, 2018.

[14] Peter Amstutz et al. Common workflow language, v1. 0. 2016.

[15] Cédric Augonnet et al. Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2):187–198, 2011.

[16] Olivier Beaumont et al. Optimal checkpointing for heterogeneous chains: how to train deep neural networks with limited memory. Report RR-9302, Inria Bordeaux Sud-Ouest, November 2019.

[17] Patrick Bellasi, Giuseppe Massari, and William Fornaciari. Effective runtime resource management using linux control groups with the barbequertrm framework. *ACM Trans. Embed. Comput. Syst.*, 14(2):39:1–39:17, March 2015.

[18] Massimo Bernaschi, Pasqua D'Ambra, and Dario Pasquini. AMG based on compatible weighted matching for GPUs. *Parallel Computing*, 92:102599, 2020.

[19] Massimo Bernaschi, Pasqua D'Ambra, and Dario Pasquini. BootCMatchG: An adaptive algebraic multigrid linear solver for GPUs. *Software Impacts*, 6:100041, 2020.

[20] Massimo Bernaschi et al. A factored sparse approximate inverse preconditioned conjugate gradient solver on graphics processing units. *SIAM Journal on Scientific Computing*, 38(1):C53–C72, 2016.

[21] Biagioni, Andrea et al. Euroexa custom switch: an innovative fpga-based system for extreme scale computing in europe. *EPJ Web Conf.*, 245:09004, 2020.

[22] Tommaso Boccali. Computing models in high energy physics. *Reviews in Physics*, 4:100034, 2019.

[23] Jaume Bosch et al. Application acceleration on fpgas with ompss@fpga. In *FPT 2018, Naha, Okinawa, Japan, December 10-14, 2018*, pages 70–77, 2018.

[24] Carlo Brandolese, Simone Corbetta, and William Fornaciari. Software energy estimation based on statistical characterization of intermediate compilation code. In *IEEE/ACM International Symposium on Low Power Electronics and Design*, pages 333–338, 2011.

[25] Rene Brun and Fons Rademakers. Root - an object oriented data analysis framework. In *AIHENP'96 Workshop, Lausane*, volume 389, pages 81–86, 1996.

[26] Neil Burgess et al. Bfloat16 processing for neural networks. In *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*, pages 88–91. IEEE, 2019.

[27] Cristiano Capone et al. Sleep-like slow oscillations improve visual classification through synaptic homeostasis and memory association in a thalamocortical model. *Scientific reports*, 9(1):1–11, 2019.

[28] Daniele Cattaneo, Michele Chiari, Nicola Fossati, Stefano Cherubin, and Giovanni Agosta. Architecture-aware precision tuning with multiple number representation systems. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 673–678, 2021.

[29] Stefano Cherubin and Giovanni Agosta. Tools for reduced precision computation: a survey. *ACM Computing Surveys*, 53(2), Apr 2020.

[30] Stefano Cherubin, Daniele Cattaneo, Michele Chiari, Antonio Di Bello, and Giovanni Agosta. TAFFO: Tuning assistant for floating to fixed point optimization. *IEEE Embedded Systems Letters*, 2019.

[31] Stefano Cherubin et al. Dynamic precision autotuning with taffo. *ACM Trans. Archit. Code Optim.*, 17(2), May 2020.

[32] Eric Chung and Alii. Serving dnns in real time at datacenter scale with project brainwave. *IEEE Micro*, 38(2):8–20, 2018.

[33] Marco Cococcioni et al. A fast approximation of the hyperbolic tangent when using posit numbers and its application to deep neural networks. In *International Conference on Applications in Electronics Pervading Industry, Environment and Society*, pages 213–221, 2019.

[34] Marco Cococcioni et al. Vectorizing posit operations on risc-v for faster deep neural networks: experiments and comparison with arm sve. *Neural Computing and Applications*, pages 1–11, 2021.

[35] Terry Cojean et al. Resource aggregation for task-based cholesky factorization on top of modern architectures. *Parallel Computing*, 83:73–92, 2019.

[36] Iacopo Colonnelli et al. Streamflow: cross-breeding cloud with HPC. *IEEE Transactions on Emerging Topics in Computing*, 2020.

[37] Iacopo Colonnelli et al. HPC Application Cloudification: The StreamFlow Toolkit. In *PARMA-DITAM HiPEAC workshop, 2021)*, volume 88 of *OASIcs*, pages 5:1–5:13, Dagstuhl, Germany, 2021.

[38] Pasqua D'Ambra and Salvatore Filippone. A parallel generalized relaxation method for high-performance image segmentation on GPUs. *J. of Computational and Applied Mathematics*, 293:35–44, 2016.

[39] Eva Darulova and Viktor Kuncak. Towards a compiler for reals. *ACM Trans. Program. Lang. Syst.*, 39(2):8:1–8:28, March 2017.

[40] Juan Miguel De Haro et al. Ompss@fpga framework for high performance FPGA computing. *IEEE Trans. Computers*, 70(12):2029–2042, 2021.

[41] Robin de Schepper, Jochen Martin Eppler, Anno Kurth, Pooja Nagendra Babu, Rajalekshmi Deepu, Sebastian Spreizer, Guido Trensch, Jari Pronold, Stine Brekke Vennemo, Steffen Graber, Aitor Morales-Gregorio, Charl Linssen, Mohamed Ayssar Benelhedi, Håkon Mørk, Abigail Morrison, Dennis Terhorst, Jessica Mitchell, Sandra Diaz, Itaru Kitayama, Mahdi Enan, Nilton Liuji Kamiji, and Hans Ekkehard Plesser. Nest 3.2, January 2022.

[42] Jacob Devlin et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[43] Alejandro Duran et al. Ompss: a proposal for programming heterogeneous multi-core architectures. *Parallel processing letters*, 21(02):173–193, 2011.

[44] Timo Felser, Pietro Silvi, Mario Collura, and Simone Montangero. Two-dimensional quantum-link lattice quantum electrodynamics at finite density. *Physical Review X*, 10(4), Nov 2020.

[45] A Ferrari, Paola R Sala, A Fassò, and Johannes Ranft. *FLUKA: A multi-particle transport code (program version 2005)*. CERN Yellow Reports: Monographs. CERN, Geneva, 2005.

[46] J. Flich et al. Exploring manycore architectures for next-generation HPC systems through the MANGO approach. *Microprocessors and Microsystems*, 61:154 – 170, 2018.

[47] William Fornaciari, Giovanni Agosta, David Atienza, Carlo Brandolese, Leila Cammoun, Luca Cremona, Alessandro Cilardo, Albert Farres, José Flich, Carles Hernandez, Michal Kulchewski, Simone Libutti, José Maria Martínez, Giuseppe Massari, Ariel Oleksiak, Anna Pupykina, Federico

Reghenzani, Rafael Tornero, Michele Zanella, Marina Zapater, and Davide Zoni. Reliable power and time-constraints-aware predictive management of heterogeneous exascale systems. In *Proceedings of the 18th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation*, SAMOS '18, page 187–194, New York, NY, USA, 2018. Association for Computing Machinery.

[48] Bruno Golosio, Chiara De Luca, Cristiano Capone, Elena Pastorelli, Giovanni Stegel, Gianmarco Tiddia, Giulia De Bonis, and Pier Stanislao Paolucci. Thalamo-cortical spiking model of incremental learning combining perception, context and NREM-sleep. *PLOS Computational Biology*, 17(6):e1009045, Jun 2021.

[49] John L Gustafson and Isaac T Yonemoto. Beating floating point at its own game: Posit arithmetic. *Supercomputing Frontiers and Innovations*, 4(2):71–86, 2017.

[50] Gilles Kahn. The semantics of a simple language for parallel programming. In *Information Processing, 6th IFIP Congress 1974*, pages 471–475, 1974.

[51] Manolis Katevenis, Roberto Ammendola, et al. Next generation of exascale-class systems: Exanest project and the status of its interconnect and storage development. *Microprocessors and Microsystems*, 61:58 – 71, 2018.

[52] Vinod Kathail. Xilinx vitis unified software platform. In *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 173–174, 2020.

[53] Alberto Leva et al. Event-based power/performance-aware thermal management for high-density microprocessors. *IEEE Trans on Control Systems Technology*, 26(2):535–550, 2018.

[54] Marina Macchiagodena et al. Virtual double-system single-box: A nonequilibrium alchemical technique for absolute binding free energy calculations: Application to ligands of the sars-cov-2 main protease. *Journal of Chemical Theory and Computation*, 16(11), 2020.

[55] Giuseppe Magnifico, Timo Felser, Pietro Silvi, and Simone Montangero. Lattice quantum electrodynamics in (3+1)-dimensions at finite density with tensor networks. *Nature Communications*, 12(1), Jun 2021.

[56] Stephen Neuendorffer and Kees Vissers. Streaming systems in fpgas. In *Embedded Computer Systems: Architectures, Modeling, and Simulation*, pages 147–156, 2008.

[57] Ahmed Omran et al. Generation and manipulation of schrödinger cat states in rydberg atom arrays. *Science*, 365(6453):570–574, 2019.

[58] Elena Pastorelli et al. Gaussian and exponential lateral connectivity on distributed spiking neural network simulation. In *26th Euromicro PDP*, pages 658–665. IEEE, 2018.

[59] Sivasankaran Rajamanickam, Seher Acer, Luc Berger-Vergiat, Vinh Q. Dang, Nathan D. Ellingwood, Evan Harvey, Brian Kelley, Christian R. Trott, Jeremiah J. Wilke, and Ichitaro Yamazaki. Kokkos kernels: Performance portable sparse/dense linear algebra and graph kernels. *CoRR*, abs/2103.11991, 2021.

[60] Federico Reghenzani, Simone Formentin, Giuseppe Massari, and William Fornaciari. A constrained extremum-seeking control for cpu thermal management. In *Proceedings of the 15th ACM International Conference on Computing Frontiers*, CF '18, page 320–325, New York, NY, USA, 2018. Association for Computing Machinery.

[61] Federico Reghenzani, Giuseppe Massari, and William Fornaciari. The misconception of exponential tail upper-bounding in probabilistic real time. *IEEE Embedded Systems Letters*, 11(3):77–80, 2019.

[62] Federico Reghenzani, Giuseppe Massari, and William Fornaciari. A probabilistic approach to energy-constrained mixed-criticality systems. In *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 1–6, 2019.

[63] Federico Reghenzani, Giuseppe Massari, and William Fornaciari. Timing predictability in high-performance computing with probabilistic real-time. *IEEE Access*, 8:208566–208582, 2020.

[64] Cristina Silvano et al. The antarex domain specific language for high performance computing. *Microprocessors and Microsystems*, 68:58–73, 2019.

[65] Phillip Stanley-Marbell et al. Exploiting errors for efficiency: A survey from circuits to applications. *ACM Comp Surveys*, 53(3), 2020.

[66] Xubin Tan et al. A hardware runtime for task-based programming models. *IEEE Trans. Par. Distributed Syst.*, 30(9):1932–1946, 2019.

[67] Federico Terraneo et al. An Open-Hardware Platform for MPSoC Thermal Modeling. In *SAMOS'19*, pages 184–196, 2019.

[68] Federico Terraneo et al. 3D-ICE 3.0: efficient nonlinear MPSoC thermal simulation with pluggable heat sink models. *IEEE Trans on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2021.

[69] Swagath Venkataramani et al. Approximate computing and the quest for computing efficiency. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2015.

[70] Erik Zenker, Benjamin Worpitz, René Widera, Axel Huebl, Guido Juckeland, Andreas Knüpfer, Wolfgang E. Nagel, and Michael Bussmann. Alpaka - an abstraction library for parallel kernel acceleration. *CoRR*, abs/1602.08477, 2016.

[71] Davide Zoni, Luca Cremona, and William Fornaciari. All-digital control-theoretic scheme to optimize energy budget and allocation in multi-cores. *IEEE Transactions on Computers*, 69(5):706–721, 2020.

[72] Davide Zoni, Luca Cremona, and William Fornaciari. All-digital energy-constrained controller for general-purpose accelerators and cpus. *IEEE Embedded Systems Letters*, 12(1):17–20, 2020.

[73] Davide Zoni, Luca Cremona, and William Fornaciari. Design of side-channel resistant power monitors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2021.

[74] Davide Zoni et al. Powertap: All-digital power meter modeling for run-time power monitoring. *Microprocessors and Microsystems*, 63:128–139, 2018.

[75] Davide Zoni, Andrea Galimberti, and William Fornaciari. Efficient and scalable fpga-oriented design of qc-ldpc bit-flipping decoders for post-quantum cryptography. *IEEE Access*, 8:163419–163433, 2020.

[76] Davide Zoni, Andrea Galimberti, and William Fornaciari. Flexible and scalable fpga-oriented design of multipliers for large binary polynomials. *IEEE Access*, 8:75809–75821, 2020.

[77] Davide Zoni, Andrea Galimberti, and William Fornaciari. An fpu design template to optimize the accuracy-efficiency-area trade-off. *Sustainable Computing: Informatics and Systems*, 29:100450, 2021.