Multi-Partner Project: Architectures and Design Methodologies to Accelerate AI Workloads. The ICSC Flagship 2 Project

Cristina Silvano*, Fabrizio Ferrandi*, Serena Curzel*, Daniele Ielmini*, Stefania Perri†, Fanny Spagnolo†, Pasquale Corsonello†, Sebastiano Fabio Schifano‡, Cristian Zambelli‡, Angelo Garofalo §, Francesco Conti §, and Luca Benini§
* Politecnico di Milano, Milano, Italy †Università della Calabria, Rende (CS), Italy
‡Università degli Studi di Ferrara, Ferrara, Italy § Università di Bologna, Bologna, Italy
Email: cristina.silvano@polimi.it

***Abstract*—Recent pre-exascale and exascale supercomputers have driven the development of increasingly sophisticated AI models for diverse applications, including image recognition and classification, natural language processing, and generative AI. These applications require specialized hardware accelerators, to handle the heavy computational demands of AI algorithms in an energy-efficient manner. Today, AI accelerators are deployed across various systems, from low-power edge devices to large-scale servers, high-performance computing (HPC) infrastructures, and data centers. The primary objective of the ICSC Flagship 2 project, discussed in this paper, is to develop heterogeneous hardware platforms optimized to accelerate HPC and big data applications. Specifically, this paper provides an overview of the key challenges addressed and the achievements realized at the current intermediate stage of the ICSC Flagship 2 project focused on architectures, technologies, and design methodologies to design efficient hardware accelerators for AI workloads, such as deep learning (DL) and transformer models.**

***Index Terms*—High-Performance Computing, Heterogeneous Architectures, AI Accelerators, RISC-V Architecture, Emerging Memory Technologies.**

I. Introduction

The ICSC Italian National Research Center for High-Performance Computing, Big Data, and Quantum Computing is a central hub for supercomputing infrastructure, supported by ten specialized research spokes, each concentrating on a specific thematic area. Among these, Spoke 1, focused on Future HPC, stands as the technological cornerstone of ICSC, committed to advancing state-of-the-art hardware and software technologies for next-generation supercomputers. Research within Spoke 1 is structured around five flagship scientific projects, involving collaboration from 15 Italian universities and 9 leading industries.

The primary objective of the Flagship 2 project is to design hardware accelerators for heterogeneous platforms to support AI workloads. This focus is driven by recent disruptive advances in AI, which have amplified the need for efficient hardware accelerators tailored to complex AI models in HPC environments. Developing these accelerators requires a multidisciplinary approach, drawing on expertise from diverse domains, including computer architecture, AI and machine learning algorithms, computational models, compiler technologies, and approximate computing.

In this context, several methodologies and tools have been proposed in recent years to design accelerators for AI, including hardware-software co-design approaches, High-Level Synthesis methods, customized compilers, and methodologies for design space exploration, modeling, and simulation. These methodologies share a common objective: to maximize computational parallelism for optimal performance while ensuring energy efficiency, thereby enabling the acceleration of AI tasks.

After a brief analysis of the state-of-the-art hardware accelerators for heterogeneous HPC platforms and perspectives on design methodologies for accelerating AI on heterogeneous architectures, this paper provides an overview of the key research challenges addressed and the main achievements reached at the current intermediate stage of the Flagship 2 project. More in detail, the paper is focused on the following research topics:

- Developing a toolchain for design space exploration and High-Level Synthesis (HLS) for AI accelerators;
- Designing efficient In-Memory Computing (IMC) architectures based on emerging technologies;
- Designing FPGA-based accelerators based on approximate computing;
- Exploiting heterogeneous CPU, GPU, ASIC, and FPGA platforms for AI and HPC applications;
- Designing accelerators based on RISC-V open-HW processor architecture.

II. State-Of-The-Art on Architectures and Design Methodologies for AI Accelerators

Stimulated by the recent AI trends that impose the use of hardware accelerators as one of the most viable solutions for several classes of applications, the ICSC Flagship 2 project mainly focused its analysis on state-of-the-art design methodologies suitable to design hardware accelerators and heterogeneous platforms for AI workloads for HPC systems, with special attention to Deep Neural Networks (DNNs) and Transformers.

This work has been supported by the Spoke 1 on *Future HPC* of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Mission 4 - Next Generation EU.

978-3-9826741-0-0/DATE25/© 2025 EDAA

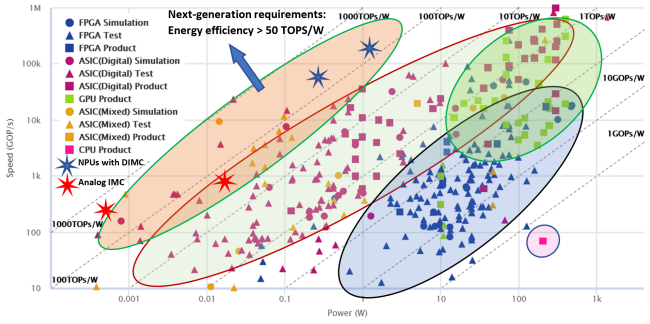


Fig. 1. Trends of state-of-the-art AI accelerators in terms of TOPs/W. Reprinted with permission from [2]. ©2024 IEEE.

One of the first outcomes of the ICSC Flagship 2 project is a survey on hardware architectures designed to accelerate AI workloads and focused on maximizing computational throughput and energy efficiency [1]. Common solutions leverage manycore architectures such as Graphics Processing Units (GPUs), Tensor Processing Units (TPUs), and Neural Processing Units (NPUs). Additional platforms include the RISC-V open-core and reconfigurable cores such as Coarse-Grained Reconfigurable Architectures (CGRAs) and Field-Programmable Gate Arrays (FPGAs). Each of these technologies can be exploited to handle the intensive, parallel nature of AI computations, offering different characteristics in terms of flexibility, cost, throughput, power consumption, and energy efficiency.

Data collected in Fig. 1 provide an overview of state-of-the-art hardware platforms suitable for AI accelerators in terms of computational speed, power consumption, and energy efficiency in TOPs/W (Tera Operations per Second per Watt). CPUs have higher cache sizes and higher on-chip bandwidth than GPUs and reconfigurable architectures but with limited capability to process large amounts of data in parallel. Consequently, CPU-based platforms are quite inefficient compared to their GPU counterparts. In particular, GPUs are extremely efficient in terms of computational speed, as they can use thousands of parallel threads to execute math and graphics tasks. Powerful GPUs represent costly solutions typically preferred for supercomputers and data centers for heavier demanding tasks, such as training complex machine learning models built on large datasets. In contrast, FPGAs are well suited for edge AI applications to accelerate specific inference tasks on resource-constrained devices that favor energy efficiency over processing speeds. ASIC designs require long design times and high design effort, but they provide good performance and energy efficiency. A good trade-off between speed, power consumption, and design effort could be offered by CGRAs, as they exhibit near-ASIC energy efficiency with near-FPGA reconfigurability levels.

Several innovative memory architectures and technologies are emerging to design more efficient hardware accelerators. For example, Fig. 1 also includes data on NPUs combined with near-memory and In-Memory Computing devices based either on analog memory technologies, like resistive RAM (RRAM) and phase change memory (PCM), or on conventional static RAMs (SRAMs). Furthermore, digital IMC represents an effective path toward the next generation of hardware accelerators for data-intensive DNN tasks on the edge.

Designing efficient AI accelerators and heterogeneous architectures requires careful consideration across the hardware and software design phases. Even tailoring accelerators to specific algorithms and computational models, such as DNNs, presents significant challenges due to the extensive design space available to designers. This space encompasses crucial architectural choices, including the number of processing units, memory hierarchy, buffer sizes, and interconnection networks. Beyond hardware architecture design, effective dataflow management, scheduling, and mapping are critical to optimize performance. A coherent link between the algorithmic model and the design constraints — such as the target performance, resource utilization, and energy efficiency —is essential for a successful design. Introducing High-Level Synthesis (HLS) enables designers to operate at a higher level of abstraction and quickly evaluate alternative accelerator designs to meet the hardware-specific constraints. Finally, approximate computing emerged as a powerful methodology for designing efficient AI accelerators limiting power consumption and resource utilization. Such an efficient approach guarantees adequate computational complexity as well as a good trade-off between speed, energy efficiency, and cost.

III. Toolchain for Design Space Exploration and High-Level Synthesis for AI Accelerators

One of the main goals of the ICSC Flagship 2 project consists of developing a Design Space Exploration (DSE) and High-Level Synthesis (HLS) toolchain to enable designers to generate efficient reconfigurable accelerators for AI with minimal manual effort. In particular, the proposed toolchain will allow designers to explore automatically the wide space of the architectural parameters, adopt optimization strategies at a high level of abstraction through performance and resource estimations, and subsequently translate the desired design configuration into an efficient FPGA accelerator by the HLS phase.

Two HLS tools have been evaluated: the commercial tool Vitis HLS from AMD/Xilinx and the open-source tool Bambu [3]. Both tools support a set of optimization directives and standard accelerator interfaces; however, Bambu has some additional features that can result in a significant advantage when targeting AI applications. Bambu does not only support inputs written as C/C++ specifications, but also compiler intermediate representations (IRs) generated from AI frameworks [4], allowing a seamless integration between the design and training of the algorithm and its synthesis. Moreover, Bambu can target FPGAs from vendors other than AMD/Xilinx, and even ASICs through integration with the OpenROAD framework. Finally, having complete visibility of the HLS flow by using an open-source tool allows finer control of the optimization techniques applied during the process and opens up a wide range of possibilities for research into novel HLS design methodologies.

One such novel research developed within the ICSC Flagship 2 project is SPARTA [5], a methodology for the automated Synthesis of PARallel multi-Threaded Accelerators. SPARTA is integrated within Bambu, and it is triggered when the input

design contains OpenMP directives to parallelize part of the application. In this specialized HLS flow, parallel regions are first translated into calls to OpenMP runtime primitives by the front-end Clang compiler, and then implemented through corresponding low-level hardware components in the synthesis backend. Accelerators generated with SPARTA are based on a custom architecture that can exploit spatial parallelism and hide the latency of external memory accesses through context switching. Moreover, SPARTA includes a custom Network-on-Chip connecting multiple external memory channels to each accelerator, memory-side caching, and on-chip private memories for each accelerator. SPARTA has primarily been tested on graph processing kernels, to demonstrate its ability to generate efficient accelerators for irregular applications, suggesting that Graph Neural Networks could benefit from the acceleration of those kernels. However, SPARTA can be applied successfully more in general for AI applications, since many high-level design frameworks for AI provide the possibility to compile the trained model into an OpenMP application.

IV. In-Memory Computing Architectures

In-memory computing (IMC) is an emerging computing paradigm where data processing is largely executed within the memory array, thus minimizing the data movement and the associated latency and energy consumption [6]. IMC is particularly efficient for data-intensive workloads, such as training and inference of AI models. Both volatile memories, such as static random access memories (SRAMs), and emerging nonvolatile memories (NVMs), such as phase change memories (PCMs) and resistive switching memories (RRAMs), have been proposed as computational memory devices supporting IMC [7]. Recently, SRAM-based digital IMC (DIMC) has been proposed with outstanding energy-efficient characteristics [2], [8].

Emerging NVMs (such as the RRAMs and the PCMs) have been recently explored for integration in stand-alone accelerators of deep neural networks (DNNs). NVMs offer the advantage of nonvolatile storage of the computational coefficients, *e.g.*, the weights of the DNNs, and the small area of the individual cell, which is generally in the range of 25 F^2, where F is the lithographic feature of the technology node, for typical one-transistor/one-resistor (1T1R) structure of the memory device [7]. The small area of NVMs allows for a significant cost/density benefit compared to the SRAM technology, characterized by a large-area, low-density structure consisting of six or 8 transistors for a single-bit cell. In addition, multilevel cell (MLC) operation is possible in both PCM and RRAM where the device resistance can be tuned as an analog memory with a virtually continuous distribution of weights [9]. This characteristic enables efficient matrix-vector multiplication (MVM) when RRAM and PCM are arranged in crossbar array structures by leveraging physical laws such as Ohm's law for voltage-conductance multiplication and Kirchhoff's current law (KCL) for summation of memory currents in the same bitline/wordline. The ICSC Flagship 2 project focuses on the development of RRAM-based and PCM-based IMC accelerators that address fundamental design challenges at various levels, including device, circuit, and architecture levels. Nonetheless, SRAM-based DIMC concepts are explored at the circuit and architectural levels.

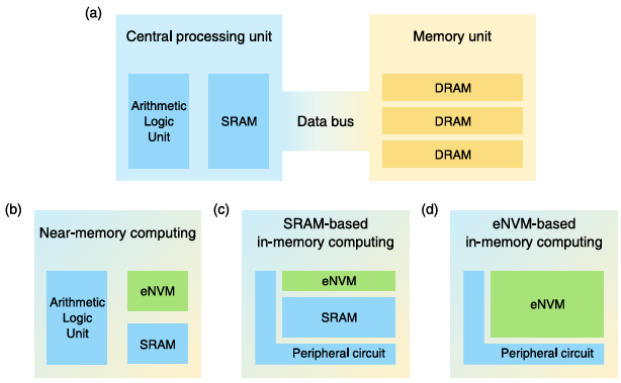


Fig. 2. Processor-memory architectures: (a) Von Neumann architecture, (b) Near-memory computing (c) SRAM-based in-memory computing, and (d) eNVM-based in-memory computing. Reprinted with permission from [7] under Creative Commons License (CC BY 4.0).

On the *device* side, both PCM and RRAM devices are characterized by non-ideal behavior in terms of variability, drift, and noise issues which severely limit the device performance. In the ICSC Flagship 2 project, we developed high-precision program-and-verify algorithms [10] to counter these non-ideal device effects, while avoiding imprecise mapping of coefficients and consequent degradation of the DNN accuracy.

On the *circuit* side, one of the key bottlenecks of NVM IMC-based accelerators is the hybrid analog/digital computation, where the analog computation in the memory enables the maximum level of parallelism. In contrast, digital computation in conventional CMOS designs is needed to perform all other types of operations (such as non-linear activation, max-pooling, and normalization). We investigated new architectures to accelerate AI tasks efficiently by minimizing the required A/D conversions with analog accumulation and neural approximated peripheral circuits [11]. The circuit design should aim at a good tradeoff between the most relevant key performance indicators (KPIs) of AI accelerators, namely performance, energy efficiency, accuracy, and cost. Performance and energy efficiency should be optimized by maximizing the parallelism of the MVM macro via analog input/activation, MLC weights, and minimum operation current. On the other hand, accuracy should be optimized by low operating current, precise MLC states, precise A/D converters, and accurate digital compensation of inaccuracies, such as drift and temperature/voltage dependence. On the other hand, DIMC relieves all the burdens described so far but introduces new challenges such as the design of fast adder trees and multipliers and the design of energy-efficient peripheral circuitry [2].

On the *architecture* side, it is essential to develop a multicore system that can harmonize and synchronize the analog MVM operations in each memory array, the digital activation and error compensation, and the data movement between the Processing Elements. This requires a suitable design of the IMC system architecture, including a proper mapping of the DNN coefficients

```
INPUT: Low-resolution image I with size H×W, filter kernel K with size t×t
OUTPUT: High-resolution image O with size 2H×2W
Initialize up and O to zero
Copy I(i, j) to up(2i, 2j)
for i = 0 to H-1 do
    for j = 0 to W-1 do
        if I(i, j) ∈ foveal region
            for u = 0 to t-1 do
                for v = 0 to t-1 do
                    O(2i, 2j) ← O(2i, 2j) + (K(u, v) × up(2i+u, 2j+v))
                    O(2i+1, 2j) ← O(2i+1, 2j) + (K(u, v) × up(2i+1+u, 2j+v))
                    O(2i, 2j+1) ← O(2i, 2j+1) + (K(u, v) × up(2i+u, 2j+1+v))
                    O(2i+1, 2j+1) ← O(2i+1, 2j+1) + (K(u, v) ×
                                    up(2i+1+u, 2j+1+v))
        else //I(i, j) ∈ peripheral region
            for u = 0 to t-1 do
                for v = 0 to t-1 do
                    O(2i, 2j) ← O(2i, 2j) + (K(u, v) × up(2i+u, 2j+v))
                    O(2i+1,2j) ← (O(2i,2j) + O(2i+2,2j)) / 2
                    O(2i, 2j+1) ← (O(2i,2j) + O(2i,2j+2)) / 2
                    O(2i+1,2j+1) ← (O(2i,2j) + O(2i, 2j+2) +
                                    O(2i+2,2j)+ O(2i+2, 2j+2)) / 4
```

Fig. 3. Pseudo-code of the proposed HTCONV layer. Reprinted with permission from [14] under Creative Commons License (CC BY 4.0).

and operations into the various tiles of the computing system and a proper communication protocol between the multiple cores. In this context, a software compiler is essential to map the DNN layers and weights to the multiple cores to maximize the KPIs.

V. FPGA-based Accelerators for Approximate Computing

In recent years, approximate computing has gained popularity as a powerful methodology to design efficient hardware accelerators with limited power consumption and resource utilization [12], [13]. AI models can take advantage of sophisticated approximation strategies that allow the fine-tuning of the power-delay-accuracy tradeoffs. Our accelerators exploit approximate computing within critical layers typically employed in Deep Learning models, such as convolutions (CONVs), transposed convolutions (TCONVs) ([14], [15], [16], [17]), pooling, fully connected operations, and SoftMax function [18]. To get reasonable flexibility, our accelerators were designed to be integrated as IPs within heterogeneous FPGA-based hardware platforms.

Among the various approximate accelerators developed in the ICSC Flagship 2 project, we focused on the new approximate TCONV layer as one of the most common techniques used in Super Resolution (SR) CNNs to upscale low-resolution images. A TCONV layer has a computational complexity significantly higher than a traditional CONV layer and requires more complex data access policies [14]. Our approach reduces the computational complexity of TCONV layers by exploiting the concept of foveated rendering of the human visual system: it has high visual acuity in a very small region, called the *fovea*, whereas outside this area it has relatively lower visual acuity, color sensitivity, and stereo depth discrimination. Based on this consideration, we developed a hybrid computational scheme (namely, HTCONV) that performs mixed accurate and inaccurate elaborations according to the foveal region.

Fig. 3 shows the pseudo-code of the proposed HTCONV method, whereas Fig. 4 shows its hardware schematic. Tests

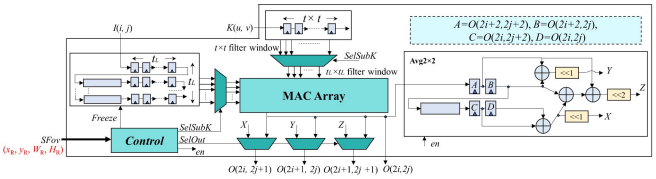


Fig. 4. Hardware architecture of the proposed HTCONV layer. Reprinted with permission from [14] under Creative Commons License (CC BY 4.0).

were performed using the pre-trained FSRCNN(25,5,1) model [19] quantized at 16-bit fixed-point and customized by including our HTCONV layer. Our approximate model was compared to the baseline FSRCNN(56,12,4), in its floating point (FP) and 16-bit quantized configurations [19], and to the 16-bit FSRCNN(25,5,1) model using the conventional TCONV layer. Our approximation strategy saves more than 80% of MACs, with a PSNR reduction lower than 10%. Table I collects some experimental results and shows the advantages offered by the new HTCONV layer over SotA solutions.

VI. Heterogeneous CPU-GPU-ASIC-FPGA platforms for AI and HPC applications

Recent advancements in technology and architecture have led to the integration of an increasing number of high-performance parallel processors and co-processors like graphics processing units (GPUs) and tensor processing units (TPUs) within HPC system nodes. This formidable computing power has accelerated the automatic training phase of AI models and their subsequent inference phases in various application scenarios. Specialized hardware accelerators such as FPGAs are essential to effectively manage the substantial computational requirements of AI algorithms while achieving better energy efficiency [20]. However, the overall performance and energy efficiency of typical AI applications, like the DNN illustrated in Fig.5, are contingent on optimizations applied across the complete software/hardware stack, as well as on the refinement of the end-to-end data flow between the data host and the accelerator. In the ICSC Flagship 2, we conducted a benchmarking campaign on a relevant DL model for medical image segmentation by using the most appropriate profiling tools for CPU, GPU, and FPGA architectures in different stages of the DL pipeline (i.e., mainly during training and inference) to extract the performance characteristics or any other suitable metric to expose the accelerators energy efficiency [21], [22]. The results are a reference point for future optimization and tradeoff analysis. After the performance bottleneck identification, we started improving the end-to-end performance in DL by addressing the I/O path with the adoption of custom solutions such as the one in [23] based on the Computational Storage paradigm and even prospecting the use of advanced memory devices such as Persistent Memory modules or low-latency SSDs. We obtained a training time reduction of up to 10% and inference throughput improvement of up to 10%.

Another application that necessitates HPC computational capabilities is the deoxyribonucleic acid (DNA) based data storage pipeline simulation. DNA is a promising approach

TABLE I
COMPARISON RESULTS TO FPGA-BASED SOTA SOLUTIONS. ADAPTED WITH PERMISSION FROM [14] UNDER CREATIVE COMMONS LICENSE (CC BY 4.0).

Method	In (Out) resolution	Bitwidth (data, weights)	Technology	Fmax (MHz)	Out Thr. (Mpixels/s)	Resources	BRAM (kB)	Power (W)	Energy eff. (Mpixels/s/W)
[15]	1440×640 (2880×1280)	(13, 13)	XC7K410T	130	495.7	171008 LUTs 161792 FFs, 1512 DSPs	922	5.38	92.13
[17]	1920×1080 (3840×2160)	(12, 12)	XC7VX485T	200	762.53	107520 LUTs 125592 FFs, 1558 DSPs	1118	NA	NA
New	1920×1080 (3840×2160)	(16, 16)	XC7K410T	222	753.04	28080 LUTs 81791 FFs, 1750 DSPs	542.25	3.7	203.5

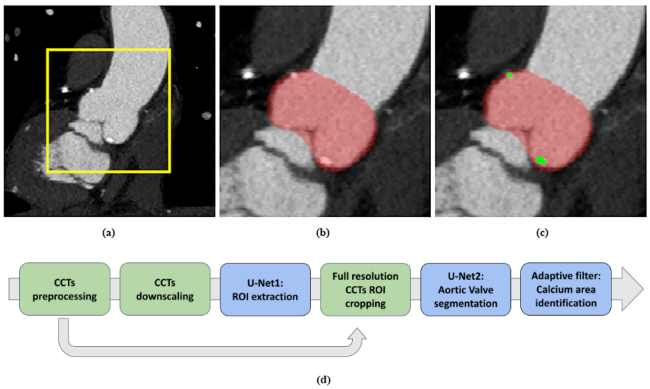


Fig. 5. The processing steps (end-to-end flow) in a typical DNN application for medical image segmentation. Reprinted with permission from [21] under Creative Commons License (CC BY 4.0).

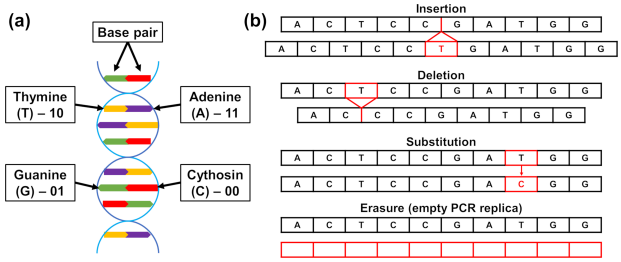


Fig. 6. (a) The double helix structure of the DNA with the base pairs evidenced along with the digital encoding of the bases. (b) Typical channel of a DNA-based data storage application. Adapted with permission from [26] under Creative Commons License (CC BY 4.0).

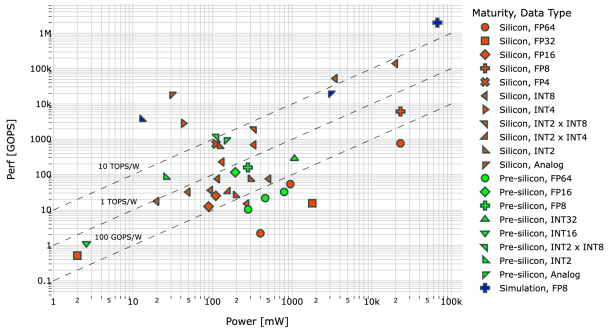


Fig. 7. RISC-V acceleration State-of-the-Art (data from [1]).

to lowering the total cost of ownership (TCO) for storage, addressing challenges in the cloud archival tier, and future AI applications [24]. This method allows encoding the digital information—composed of '1's and '0's—in a synthetic molecule made up of two polymer chains that contain a sequential arrangement of nucleotide monomers (bases) (refer to Fig. 6). Data stored in DNA can endure for thousands of years with minimal power consumption to enhance retention and achieve a density of approximately 100 PB per gram [25]. The DNA storage process encompasses several stages typically modeled by a software framework [26] designed to capture the unique aspects of encoding and decoding information. The most crucial element of the model involves the DNA channel noise characteristics and the decoding phase of the information. A distinctive feature of the DNA channel is that the input consists of numerous strings of similar lengths that share a certain degree of similarity, characterized by a specific distance. The similarity index is determined using the edit distance, also known as the Levenshtein distance [27]. Since the computations are in the context of bitwise operations without the need for floating-point or fast tensor operations, there is a surge of interest in FPGA accelerators for edit distance [28]. GPUs [29] and ASIC accelerators such as the Intel Xeon Phi [30] have provided relevant performance, although their architectures are unsuitable for energy-efficient designs. In literature, previous works demonstrated the feasibility of FPGA designs [26], [28], [31]. However, most results do not meet the requirements dictated by the fast decoding process used in DNA-based storage applications [32]. Alternative solutions are based on approximated distance techniques between strings [33], [34], although struggling in terms of edit/s figure of merit. In the ICSC Flagship 2 project, we developed a custom FPGA accelerator based on the AMD-Xilinx Alveo U50 Data Center Accelerator Card [35]. Our solution uses nearly 90% of FPGA basic-block hardware resources, achieving about 90% computing efficiency while delivering a maximum throughput of 16.8 TCUPS and an energy efficiency of 46 Mpair/Joule, enabling the use of FPGAs as a new class of accelerators for High-Performance Computing in DNA applications.

VII. ACCELERATORS BASED ON RISC-V ARCHITECTURE

AI applications are expected to drive the next generation of HPC applications, as outlined by AMD CEO Lisa Su in the Plenary Session of ISSCC'23 [36]. AI will be used as a companion to conventional HPC applications to speed up and boost their energy efficiency, for example by providing approximate but widely usable first solutions using low-bit-precision (e.g., FP16, FP8, or even FP4) to be refined by con-

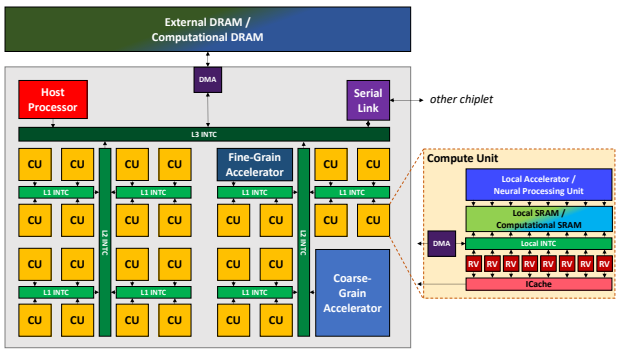
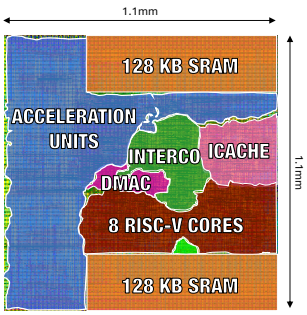


Fig. 8. *Scalable Compute Fabric* architecture template.



Technology	GlobalFoundries 12nm
Maturity	Post-layout
Area	1.21 mm^2
Frequency	460 MHz @ 0.55V
Performance (BF16)	150 GFLOPS
Efficiency (BF16)	1.5 TFLOPS/W

Fig. 9. A prototype *Compute Unit* for AI acceleration.

ventional high-accuracy algorithms employing highly accurate data representation (FP64 or FP128).

To feed these developments, novel hardware accelerator architectures are required that focus on highly energy-efficient inference and not on training, like current-generation data center GPUs [37], [38]. Most developments in this direction are undergoing in North America or East Asia [39]–[43], with the only notable exception of AxeleraAI's Metis architecture [44]. It is therefore highly desirable to increase Europe's independence and know-how in the specification, design, and fabrication of accelerator systems for this particular niche. The RISC-V open instruction set architecture (ISA) and its vast IP ecosystem, with a strong presence of European academia and industry, offer a solid starting point for this effort.

Fig. 7 shows how current RISC-V-based architectures for Deep Learning and Transformers acceleration, many of which are EU-based, are "clustered", especially in the 100mW-1W power range. The ICSC Flagship 2 project aims to extend this range of interest to architectures in the >1W range for HPC deep learning inference. Therefore, one of the target research topics is the architecture design, simulation framework, and overall validation of the system architecture of a *Scalable Compute Fabric* (SCF) exploiting the RISC-V open processor.

Fig. 8 shows the target template of the architecture of the SCF. The template includes, on a single silicon chip/chiplet, a heterogeneous acceleration system with a host/controller Linux capable processor (e.g., based on the CVA6 design) and an acceleration fabric composed of a collection of *Compute Units* (CUs) and accelerators of various granularity, connected using a scalable interconnect, such as a hierarchical AXI [45], [46] or a Network-on-Chip [47]. CUs are based on (not necessarily identical) clusters of one or more RISC-V cores oriented on computation, such as Snitch[1] or CV32E40P[2], sharing a local L1 SRAM to enable coordinated computation. Each CU can further be augmented with special purpose units, such as vector processing units tightly-coupled to the cores [48]; local neural processing units (NPUs) [49]; tensor cores [50]; digital in-memory-computing augmented SRAM [8].

Figure 9 shows a prototype *Compute Unit* developed within the ICSC Flagship 2 for the acceleration of DNN and Transformer units. The CU, laid out in GlobalFoundries 12nm technology, occupies $\sim$1.21mm^2, which is comparable to the size of a cluster in an already-fabricated precursor to the SCF architecture, Occamy [46]. Thanks to accelerators using the BFloat16 precision for all major Transformer blocks, the CU achieves up to 150 GFLOPS and 1.5 TFLOPS/W at 460 MHz, 0.55 V. The next steps of the Flagship 2 activities include using this and other similar CUs to build a scaled-up SCF.

VIII. Conclusions

This paper describes an overview of the main research challenges and related intermediate outcomes realized at the current stage of the ICSC Flagship 2 project focused on design methodologies for energy-efficient hardware accelerators for AI and HPC workloads.

References

[1] C. Silvano *et al.*, "A Survey on Deep Learning Hardware Accelerators for Heterogeneous HPC Platforms," 2024. arXiv, cs.AR, 2306.15552.

[2] S. Perri, C. Zambelli, D. Ielmini, and C. Silvano, "Digital In-Memory Computing to Accelerate Deep Learning Inference on the Edge," in *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 130–133, 2024.

[3] F. Ferrandi, V. G. Castellana, S. Curzel, P. Fezzardi, M. Fiorito, M. Lattuada, *et al.*, "Bambu: an Open-Source Research Framework for the High-Level Synthesis of Complex Applications," in *DAC 2021: 58th ACM/IEEE Design Automation Conference*, 2021.

[4] N. Bohm Agostini, S. Curzel, J. J. Zhang, A. Limaye, C. Tan, V. Amatya, M. Minutoli, V. G. Castellana, J. Manzano, D. Brooks, G.-Y. Wei, and A. Tumeo, "Bridging Python to Silicon: The SODA Toolchain," *IEEE Micro*, vol. 42, no. 5, pp. 78–88, 2022.

[5] G. Gozzi, M. Fiorito, S. Curzel, C. Barone, V. G. Castellana, M. Minutoli, A. Tumeo, and F. Ferrandi, "Sparta: High-level synthesis of parallel multi-threaded accelerators," *ACM Trans. Reconfigurable Technol. Syst.*, July 2024. Just Accepted.

[6] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, pp. 333–343, 2018.

[7] N. Lepri, A. Glukhov, L. Cattaneo, M. Farronato, P. Mannocci, and D. Ielmini, "In-Memory Computing for Machine Learning and Deep Learning," *IEEE Journal of the Electron Devices Society*, vol. 11, pp. 587–601, 2023.

[8] G. Desoli, N. Chawla, T. Boesch, M. Avodhyawasi, H. Rawat, H. Chawla, V. Abhijith, P. Zambotti, A. Sharma, C. Cappetta, M. Rossi, A. De Vita, and F. Girardi, "A 40-310TOPS/W SRAM-Based All-Digital Up to 4b In-Memory Computing Multi-Tiled NN Accelerator in FD-SOI 18nm for Deep-Learning Edge Applications," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 260–262, 2023.

[9] P. Mannocci, M. Farronato, N. Lepri, L. Cattaneo, A. Glukhov, Z. Sun, and D. Ielmini, "In-memory computing with emerging memory devices: status and outlook," *APL Machine Learning*, vol. 1, p. 010902, 2023.

[1] https://github.com/pulp-platform/snitch

[2] https://github.com/openhwgroup/cv32e40p

[10] V. Milo *et al.*, "Accurate program/verify schemes of resistive switching memory (RRAM) for in-memory neural network circuits," *IEEE Transactions on Electron Devices*, vol. 68, pp. 3832–3837, 2021.

[11] W. Cao, Y. Zhao, A. Boloor, Y. Han, X. Zhang, and L. Jiang, "Neural-PIM: Efficient Processing-In-Memory With Neural Approximation of Peripherals," *IEEE Transactions on Computers*, vol. 71, no. 9, pp. 2142–2155, 2022.

[12] M. Alioto, "Energy-quality scalable adaptive vlsi circuits and systems beyond approximate computing," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pp. 127–132, 2017.

[13] M. Alioto, V. De, and A. Marongiu, "Guest Editorial for the Special Issue on Energy-Quality Scalable Circuits and Systems for Sensing and Computing: from Approximate, to Communication-Inspired and Learning-Based," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, pp. 1–1, 2018.

[14] F. Spagnolo, P. Corsonello, F. Frustaci, and S. Perri, "Design of a Low-Power Super-Resolution Architecture for Virtual Reality Wearable Devices," *IEEE SENSORS JOURNAL*, vol. 23, no. 8, pp. 9009–9016, 2023.

[15] W. Chang, K.-W. Kang, and S.-J. Kang, "An Energy-Efficient FPGA-Based Deconvolutional Neural Networks Accelerator for Single Image Super-Resolution," *IEEE Trans. Circ. Syst. Video Tech.*, vol. 30, no. 1, p. 281–295, 2020.

[16] C. Sestito, F. Spagnolo, and S. Perri, "Design of Flexible Hardware Accelerators for Image Convolutions and Transposed Convolutions," *Journal of Imaging*, vol. 7, no. 10, 2021.

[17] L. Chang, X. Zhao, and J. Zhou, "ADAS: A High Computational Utilization Dynamic Reconfigurable Hardware Accelerator for Super Resolution," *ACM Trans. Reconf. Techn. Syst*, vol. Early Access, 2022.

[18] F. Spagnolo, S. Perri, and P. Corsonello, "Aggressive Approximation of the SoftMax Function for Power-Efficient Hardware Implementations," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, p. 1652–1656, 2022.

[19] C. Dong, C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision, 2016*, 2016.

[20] T. Nguyen, S. Williams, M. Siracusa, C. MacLean, D. Doerfler, and N. J. Wright, "The Performance and Energy Efficiency Potential of FPGAs in Scientific Computing," in *2020 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*, pp. 8–19, 2020.

[21] G. Minghini, A. U. Cavallo, A. Miola, V. Sisini, E. Calore, F. Fortini, R. Micheloni, P. Rizzo, S. F. Schifano, F. V. D. Sega, and C. Zambelli, "An HPC Pipeline for Calcium Quantification of Aortic Root From Contrast-Enhanced CCT Scans," *IEEE Access*, vol. 11, pp. 101309–101319, 2023.

[22] V. Sisini, A. Miola, G. Minghini, E. Calore, A. U. Cavallo, S. F. Schifano, and C. Zambelli, "Segmentation of Aortic Valve Calcium Lesions using FPGA Accelerators," in *Proc. of 15th International Conference on Parallel Processing & Applied Mathematics (PPAM)*, 2024.

[23] C. Zambelli, R. Bertaggia, L. Zuolo, R. Micheloni, and P. Olivo, "Enabling computational storage through fpga neural network accelerator for enterprise ssd," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 10, pp. 1738–1742, 2019.

[24] "Preserving our digital legacy: an introduction to DNA Data Storage." White paper, 6 2021.

[25] R. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. Stark, "Robust Chemical Preservation of Digital Information on DNA in Silica with Error-Correcting Codes," *Angewandte Chemie International Edition*, vol. 54, p. 2552, 1 2015.

[26] A. Marelli, T. Chiozzi, N. Battistini, L. Zuolo, R. Micheloni, and C. Zambelli, "Integrating FPGA Acceleration in the DNAssim Framework for Faster DNA-Based Data Storage Simulations," *Electronics*, vol. 12, no. 12, 2023.

[27] B. Berger, M. S. Waterman, and Y. W. Yu, "Levenshtein Distance, Sequence Comparison and Biological Database Search," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 3287–3294, 2021.

[28] D. Castells-Rufas, S. Marco-Sola, J. C. Moure, Q. Aguado, and A. Espinosa, "FPGA Acceleration of Pre-Alignment Filters for Short Read Mapping With HLS," *IEEE Access*, vol. 10, pp. 22079–22100, 2022.

[29] A. Chacón, S. Marco-Sola, A. Espinosa, P. Ribeca, and J. C. Moure, "Thread-Cooperative, Bit-Parallel Computation of Levenshtein Distance on GPU," in *Proceedings of the 28th ACM International Conference on Supercomputing*, p. 103–112, 2014.

[30] E. Rucci, C. García, G. Botella, A. De Giusti, M. Naiouf, and M. Prieto-Matías, "An energy-aware performance analysis of SWIMM: Smith–Waterman implementation on Intel's Multicore and Manycore architectures," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 18, pp. 5517–5537, 2015.

[31] D. Castells-Rufas, S. Marco-Sola, Q. Aguado-Puig, A. Espinosa-Morales, J. C. Moure, L. Alvarez, and M. Moretó, "OpenCL-based FPGA Accelerator for Semi-Global Approximate String Matching Using Diagonal Bit-Vectors," in *2021 31st International Conference on Field-Programmable Logic and Applications (FPL)*, pp. 174–178, 2021.

[32] C. Rashtchian, K. Makarychev, M. Racz, S. Ang, D. Jevdjic, S. Yekhanin, L. Ceze, and K. Strauss, "Clustering Billions of Reads for DNA Data Storage," in *Advances in Neural Information Processing Systems 30*, 12 2017.

[33] M. Alser, H. Hassan, A. Kumar, O. Mutlu, and C. Alkan, "Shouji: a fast and efficient pre-alignment filter for sequence alignment," *Bioinformatics*, vol. 35, no. 21, pp. 4255–4263, 2019.

[34] M. Alser, T. Shahroodi, J. Gómez-Luna, C. Alkan, and O. Mutlu, "SneakySnake: a fast and accurate universal genome pre-alignment filter for CPUs, GPUs and FPGAs," *Bioinformatics*, vol. 36, no. 22-23, pp. 5282–5290, 2020.

[35] S. F. Schifano, M. Reggiani, E. Calore, R. Micheloni, A. Marelli, and C. Zambelli, "High throughput edit distance computation on fpga-based accelerators using hls," *Future Generation Computer Systems*, vol. 164, p. 107591, 2025.

[36] "Session 1 Overview: Plenary," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 6–7, Feb. 2023.

[37] A. C. Elster and T. A. Haugdahl, "Nvidia Hopper GPU and Grace CPU Highlights," *Computing in Science & Engineering*, vol. 24, pp. 95–100, Mar. 2022.

[38] J. Choquette, "NVIDIA Hopper H100 GPU: Scaling Performance," *IEEE Micro*, pp. 1–13, 2023.

[39] S. Ward-Foxton, "Axelera Demos AI Test Chip After Taping Out in Four Months," 2022.

[40] D. R. Ditzel and t. E. team, "Accelerating ML Recommendation With Over 1,000 RISC-V/Tensor Processors on Esperanto's ET-SoC-1 Chip," *IEEE Micro*, vol. 42, pp. 31–38, May 2022.

[41] S. Knowles, "Graphcore," in *2021 IEEE Hot Chips 33 Symposium (HCS)*, pp. 1–25, Aug. 2021.

[42] S. Davidson *et al.*, "The Celerity Open-Source 511-Core RISC-V Tiered Accelerator Fabric: Fast Architectures and Design Methodologies for Fast Chips," *IEEE Micro*, vol. 38, pp. 30–41, Mar. 2018.

[43] J. Vasiljevic *et al.*, "Compute Substrate for Software 2.0," *IEEE Micro*, vol. 41, pp. 50–55, Mar. 2021.

[44] P. A. Hager *et al.*, "11.3 Metis AIPU: A 12nm 15TOPS/W 209.6TOPS SoC for Cost- and Energy-Efficient Inference at the Edge," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, pp. 212–214, Feb. 2024.

[45] N. Bruschi, G. Tagliavini, A. Garofalo, F. Conti, I. Boybat, L. Benini, and D. Rossi, "End-to-End DNN Inference on a Massively Parallel Analog In Memory Computing Architecture," Nov. 2022.

[46] G. Paulin *et al.*, "Occamy: A 432-Core 28.1 DP-GFLOP/s/W 83% FPU Utilization Dual-Chiplet, Dual-HBM2E RISC-V-Based Accelerator for Stencil and Sparse Linear Algebra Computations with 8-to-64-bit Floating-Point Support in 12nm FinFET," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, pp. 1–2, June 2024.

[47] T. Fischer, M. Rogenmoser, M. Cavalcante, F. K. Gürkaynak, and L. Benini, "FlooNoC: A Multi-Tb/s Wide NoC for Heterogeneous AXI4 Traffic," *IEEE Design & Test*, vol. 40, pp. 7–17, Dec. 2023.

[48] M. Cavalcante, D. Wüthrich, M. Perotti, S. Riedel, and L. Benini, "Spatz: A Compact Vector Processing Unit for High-Performance and Energy-Efficient Shared-L1 Clusters," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, (San Diego California), pp. 1–9, ACM, Oct. 2022.

[49] A. Prasad, L. Benini, and F. Conti, "Specialization meets Flexibility: A Heterogeneous Architecture for High-Efficiency, High-flexibility AR/VR Processing," in *Proceedings of the 2023 Design Automation Conference (DAC 2023), to Appear*, 2023.

[50] Y. Tortorella, L. Bertaccini, L. Benini, D. Rossi, and F. Conti, "RedMule: A Mixed-Precision Matrix-Matrix Operation Engine for Flexible and Energy-Efficient On-Chip Linear Algebra and TinyML Training Acceleration," Jan. 2023.