

**Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw  
Supercomputing Applications for exascale**



**textarossa**

## **WP1 Specifications, Co-design & Benchmarking**

---

### D1.1 Gap Analysis

Revised version

<http://textarossa.eu>



This project has received funding from the European Union's Horizon 2020 research and innovation programme, EuroHPC JU, grant agreement No 956831





## TEXTAROSSA

**Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw  
Supercomputing Applications for exascale**

**Grant Agreement No.: 956831**

**Deliverable: D1.1 Gap Analysis**

**Project Start Date:** 01/04/2021

**Duration:** 36 months

**Coordinator:** AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE, L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE - ENEA, Italy.

<b>Deliverable No</b>	D1.1 (revised version)
<b>WP No:</b>	WP1
<b>WP Leader:</b>	ENEA
<b>Due date:</b>	M6 (September 30, 2021)
<b>Delivery date:</b>	M27 (revised version)

**Dissemination Level:**

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	



This project has received funding from the European Union's Horizon 2020 research and innovation programme, EuroHPC JU, grant agreement No 956831



## DOCUMENT SUMMARY INFORMATION

<b>Project title:</b>	Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale
<b>Short project name:</b>	TEXTAROSSA
<b>Project No:</b>	956831
<b>Call Identifier:</b>	H2020-JTI-EuroHPC-2019-1
<b>Unit:</b>	EuroHPC
<b>Type of Action:</b>	EuroHPC - Research and Innovation Action (RIA)
<b>Start date of the project:</b>	01/04/2021
<b>Duration of the project:</b>	36 months
<b>Project website:</b>	textarossa.eu

## WP1 Specifications, Co-design & Benchmarking

<b>Deliverable number:</b>	D1.1					
<b>Deliverable title:</b>	Gap Analysis					
<b>Due date:</b>	M6					
<b>Actual submission date:</b>	M27 (revised version)					
<b>Editor:</b>	Francesco Iannone					
<b>Authors:</b>	List of Authors					
<b>Work package:</b>	WP1					
<b>Dissemination Level:</b>	Public					
<b>No. pages:</b>	90					
<b>Authorized (date):</b>	22/05/2023					
<b>Responsible person:</b>	Francesco Iannone					
<b>Status:</b>	Plan	Draft	Working	Final	Submitted	Approved

### Revision history:

Version	Date	Author	Comment
0.1	21/12/2021	F.Iannone	Draft structure
0.2	21/12/2021	F.Iannone	Errata Corrige
0.3	23/12/2021	P.Palazzari	Review
0.4	15/05/2023	PTC	New revised version
0.5	17/05/2023	Alessandro Lonardo	Main Revision
0.6	22/05/2023	M. Celino	Final version

### Quality Control:

Checking process	Who	Date
<b>Checked by internal reviewer</b>	Project Technical Committee	15/05/2023
<b>Checked by Task Leader</b>		
<b>Checked by WP Leader</b>	Francesco Iannone	17/05/2023
<b>Checked by Project Coordinator</b>	Massimo Celino	01/06/2023

## COPYRIGHT

© Copyright by the **TEXTAROSSA** consortium, 2021-2024

This document contains material, which is the copyright of TEXTAROSSA consortium members and the European Commission, and may not be reproduced or copied without permission, except as mandated by the European Commission Grant Agreement No. 956831 for reviewing and dissemination purposes.

## ACKNOWLEDGEMENTS

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement no 956831. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Italy, Germany, France, Spain, Poland.

Please see <http://textarossa.eu> for more information on the TEXTAROSSA project.

The partners in the project are AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE, L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE (ENEA), FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V. (FHG), CONSORZIO INTERUNIVERSITARIO NAZIONALE PER L'INFORMATICA (CINI), INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE (INRIA), BULL SAS (BULL), E4 COMPUTER ENGINEERING SPA (E4), BARCELONA SUPERCOMPUTING CENTER-CENTRO NACIONAL DE SUPERCOMPUTACION (BSC), INSTYTUT CHEMII BIOORGANICZNEJ POLSKIEJ AKADEMII NAUK (PSNC), ISTITUTO NAZIONALE DI FISICA NUCLEARE (INFN), CONSIGLIO NAZIONALE DELLE RICERCHE (CNR), IN QUATTRO SRL (in4). Linked third parties of CINI are POLITECNICO DI MILANO (CINI-POLIMI), Università di Torino (CINI-UNITO) and Università di Pisa (CINI-UNUPI); linked third party of INRIA is Université de Bordeaux; in-kind third party of ENEA is Consorzio CINECA (CINECA); in-kind third party of BSC is Universitat Politècnica de Catalunya (UPC).

The content of this document is the result of extensive discussions within the TEXTAROSSA © Consortium as a whole.

## DISCLAIMER

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

The information contained in this document is provided by the copyright holders "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the members of the TEXTAROSSA collaboration, including the copyright holders, or the European Commission be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of the information contained in this document, even if advised of the possibility of such damage.

## Table of Contents

<b>Executive Summary</b> .....	6
<b>List of Authors</b> .....	6
<b>List of Acronyms</b> .....	7
1 Introduction.....	9
2 Methodology.....	11
3 Task-1.1: User Applications.....	12
4 Task-1.1.1: New algorithm and software libraries.....	12
5 Task-1.1.2: AI and HPDA.....	16
6 Task-1.1.3: Scientific flagship codes.....	22
7 Task-1.2: Runtime Services.....	25
8 Task-1.2.1: Resources Management.....	26
9 Task-1.3: Programming Models.....	32
10 Task-1.3.1: Streaming Model.....	32
11 Task-1.3.2: Tasks Model.....	35
12 Task-1.3.3: High Level Synthesis Flow.....	37
13 Task-1.3.4: Mixed-Precision tools and alternative arithmetic-type for IP accelerators.....	40
14 Task-1.3.5: Secure Services for HPC.....	43
15 Task-1.4: System Architecture.....	45
16 Task-1.4.1: Heterogeneous Architectures.....	45
17 Task-1.4.2: Interconnection Networks.....	62
18 Task-1.5: Hardware Platforms.....	65
19 Task-1.5.1: FPGA Platform.....	68
20 Task-1.5.2: GPU Platform.....	77
21 Task-1.5.3: Power and Thermal Management.....	80
22 Summary of Actions.....	83
23 Conclusions.....	86
Annex I: Xilinx FPGA product tables.....	87
Annex II: Intel FPGA product tables.....	89
Annex III: Intel Agilex I-Series FPGA product tables.....	90

## Executive Summary

The objective of WP1 is to define and implement a co-design process steering the process towards the specification of highly energy-efficient heterogeneous computing systems for High Performance Computing, AI and HPDA. For this purpose we adopt a holistic approach involving the technology components developed in the project along with the set of selected application benchmarks. In fact a co-design process works on the whole stack from underlying hardware/software platforms to the tip of the applications with a process built around the identification of specific high-impact applications and providing custom optimization targets. The stack of the co-design process shall be deployed over the tasks matrix describing the layout topics of the WP1 objectives composed of five tasks as rows of the task matrix: 1) User Applications; 2) Runtime Services; 3) Programming Models; 4) System Architectures; 5) Hardware Platforms. This report is focused on the first column of the tasks matrix: Gap Analysis to outline the state-of-the-art in the HPC landscape in all five tasks of the co-design stack.

## Partner Report Activity

Task 1.1 TL: INRIA	<u>User Applications</u> : to provide gap analysis on flagship scientific codes, numerical libraries, AI and HPDA developed in WP6. Participants: CNR, CINI, INFN, ENEA, FHG, PSNC
Task 1.2 TL: INRIA	<u>Runtime Service</u> : to provide gap analysis on execution models of the application including workflow, resources management as well as the IO interfaces. Participants: ENEA, BSC, CINI, PSNC
Task 1.3 TL: ENEA	<u>Programming models</u> : to provide gap analysis on new toolchains and workflow shall be available for optimizing the user applications for heterogeneous architectures. Participants: ENEA, INRIA, UBx, CINI (POLIMI, UNIPI, UNITO), BSC
Task 1.4 TL: CINI	<u>System Architectures</u> : to provide gap analysis on the architectures of: CPU/FPGA/GPU, memory hierarchies, IO subsystems, networks. Participants: CINI (UNIPI, POLIMI), FHG, ENEA, E4, INFN
Task 1.5 TL: ATOS	<u>Hardware platforms</u> : to provide gap analysis on cooling at level of node, rack and data center as well as in term of Energy Reuse Effectiveness. Participants: E4, InQuattro, ENEA, PSNC
Github address	The software developed and the benchmarks carried out during the activity are downloadable at github at the address: <a href="https://gitlab-tex.enea.it">https://gitlab-tex.enea.it</a>
Technology	ENEA HPC CRESCO Data Centre
Technical development	The technical development is performed by ENEA.

## List of Authors

ENEA	F. Iannone, P. Palazzari
CINI-POLITO	M. Aldinucci
CINI-POLIMI	G. Agosta
CINI-UNIPI	S. Saponara, D. Rossi, M. Danelutto, M. Torquati, G. Mencagli
INRIA	B. Bramas
ATOS	S. Lesmanne

E4	D. Gregori
BSC	C. Alvarez, X. Martorell
PSNC	A. Oleksiak, M. Kulczewski, S. Ciesielski
INFN	A. Lonardo, A. Biagioni, T. Boccali, F. Giacomini, G. Magnifico, S. Montangero, F. Simula, P. Vicini
CNR	P. D’Ambra, M. Bernaschi
InQuattro	G. Zummo

### List of Acronyms

AaaS	Accelerator as Service
ABI	Application Binary Interfaces
ACP	Acceleration Coherency Port
ADC	Analog Digital Converter
AFS	Andrew File System
AI	Artificial Intelligence
ALU	Arithmetic Logic Unit
AMG	Algebraic MultiGrid
AMS	Analog Mixed Signal
API	Application Program Interface
ASIC	Application Specific Integrating Circuit
BMC	Baseboard Management Controller
C/R	Checkpointing/Restart
CDU	Cooling Distribution Unit
CNN	Convolution Neural Network
CP	Common Platform
CPU	Central Processing Unit
CRDB	Co-design Recommended Daughter Board
CU	Compute Unit
DAG	Data-flow Graphs
DC	Direct Cooling
DCL	Data Control Language
DDR	Double Data Rate memory
DIMMs	Dual In-line Memory Modules
DL	Deep Learning
DLC	Direct Liquid Cooling
DPSNN	Distributed Polychronous Spiking Neural
DSL	Domain Specific Language
DSP	Digital Signal Processing
DTPC	Direct Two-Phase Cooling
ECC	Elliptic Curve Cryptography
ECC	Error correction code memory
EDS	Embedded Design Suite
EFLOPS	Exa Floating Point Operations per Second
EPAC	EPI Accelerator
EPI	European Processor Initiative
FMM	Fast Multipole Method
FP	Floating Point
FPGA	Field Programmable Gate Array
FPU	Floating Point Unit
FT	Fault Tolerance
FTI	Fault Tolerance Interface
GCD	Graphics Compute Die
GIC	Generic Interrupt Controller
gpm	Gallons per minute
GPU	Graphics Processing Unit
GRM	Global Resource Manager
HBA	Host Bus Adapter
HBM	High Bandwidth Memory
HEP	High Energy Physics
HLL	High-Level Language
HLS	High Level Synthesis

HPC	High Performance Computing
HPDA	High Performance Data Analytics
HPL	High Performance Linpack
HPS	Hard Processor System
HSS	High Speed Serial
IoT	Internet of Things
IP	Intellectual Property
IPMI	Intelligent Platform Management Interface
IR	Iterative Refinement
KPN	Kahn Process Network
L2HN	L2 cache Coherence Home Node
LE	Logic Element
LRM	Local Resource Manager
MCM	Muti-Chip-Module
MD	Molecular Dynamic
MDS/T	Metadata Server/Target
ML	Machine Learning
MMU	Memory Management Unit
MPI	Message Passing Interface
MPPA	Multi-Purpose Processing Array
MPSoC	Multi-Processor System on Chip
NN	Neural Network
NoC	Network on Chip
NVIC	Nested Vectored Interrupt Controller
NVMe	Non-Volatile Memory
OAM	OCP Accelerator Module
OCP	Open Compute Project
OSS/T	Object Storage Servers /Target
PCG	Preconditioned Conjugate Gradient
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PFLOPS	Peta Floating Point Operations per Second
PGMRES	Preconditioned Generalized Minimal Residual
PSU	Power Supply Unit
PU	Processing Unit
PUE	Power Usage Effectiveness
QFDB	Quad FPGA Daughter Board
QoS	Quality of Service
RAID	Redundant Array of Independent Disks
RDMA	Remote Direct Memory Access
RISC	Reduced Instruction Set Computer
RMS	Resources Management Systems
RTC	Real Time Clock
RTRM	Runtime Resource Manager
SAS/SATA	Serial Attached SCSI/Serial ATA
SoC	System on Chip
SpMM	Sparse Matrix-sparse Matrix
SpMP	Sparse Matrix Power
SpMV	Sparse Matrix-Vector
SSD	Solid State Drive
STX	Stencil/Tensor accelerator
TCO	Total Cost of Ownership
TDAQ	Trigger & Data Acquisition
TDP	Thermal Design Power
TFLOPS	Tera Floating Point Operations per Second
ULFM	User-level Fault Mitigation
ULL	Ultra-Low Latency
UVM	Unified Virtual Memory
VCS	Virtual Compute Server
VM	Virtual Machine
VPU	Vector Processing Unit
VRP	Variable Precision co-processor



# 1 Introduction

Rising energy costs, and the increase in data center power consumption driven by an ever-increasing demand for data services, are becoming the dominating factor for the Total Cost of Ownership over the lifetime of a computing system. Additionally, current semiconductor technology will be hitting a point where downsizing and, thus, inherent reduction of power, will no longer be possible mainly due to the increase in leakage currents. Many high-profile studies show the increase of data center power demands is the main challenge that exascale and beyond exascale computing has to face. Global data center electricity demand in 2019 was around 200 TWh, or around 0.8% of global final electricity demand. If current trends in the efficiency of hardware and data centre infrastructure can be maintained, global data centre energy demand can remain nearly flat through 2022, despite a 60% increase in service demand. For HPC data centre the main challenge is the threshold of the power consumption to a 20 MW in the exascale systems, e.g. 20 mWatt per GFlops [Handbbok on Data Centers, S.U.Khan, 2015]. To better understand the challenge, Fig. 1 shows the performances in terms of mW/GFlops of the HPC systems in the first position of the top 500 list in the last 12 years.

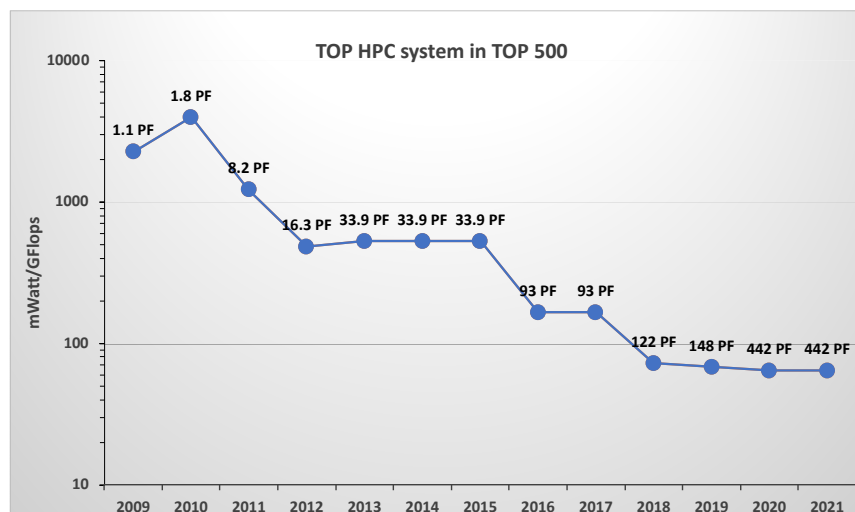


Fig.1: Electric Power consuming per GFlops of the top HPC systems in the TOP 500 lists since 2009

The increasing power density in modern post-Dennard scaling Multi-Processor System on Chips (MPSoCs) raises chip temperature and on-chip thermal gradients, leading to a wear-out of silicon devices and putting at stake the lifetime reliability of chips, defined as the long-term reliability. Together, performance and lifetime reliability issues make joint power and thermal management crucial and inevitable for MPSoCs, and also pose an important challenge from the energy efficiency perspective. The next step in higher intensity, more energy efficient cooling in data centers is the move to “on-chip” cooling, with much higher heat transfer removable rates than air cooling. A new two-phase cooling system using flow of boiling heat transfer to cool electronic devices, and compared to traditional cooling systems (liquid cooling or heat pipes), can achieve significantly higher heat transfer coefficients at significantly lower flow rates and pumping power able to achieve a PUE (Power Usage Effectiveness) close to 1, verified by monitoring all relevant information.

In the last decade, mainly because of the continuously increasing graphics processing demands of the video game industry, Graphics Processing Units (GPUs) have evolved into massively parallel computing engines. The current pre-exascale HPC systems in the last Top 10 list, are supercomputers GPU based. On the other hand, FPGA solutions can also offer high throughput to numerous data-intensive applications with critical time constraints. The following table 1 shows the main features of the GPUs and FPGA already available to co-design HPC heterogenous systems.

	Model	Processor	single precision TFlops	double precision TFlops	Power Consumption Watt	MW/GFlops (DP)
GPU	NVIDIA TESLA	A100 HGX	19.5	9.7	400	41.2
	NVIDIA TESLA	V100-SXM2	15.7	7.8	300	38.5
	AMD RADEON	INSTINCT MI60	14.7	7.4	300	40.5
FPGA	Alveo-U250-A64G-PQ-G	XLINK Virtex Ultrascale+	8.3	4.1	225	54.9
	520N-MX	INTEL STRATIX-10 GX2800	9.2	4.6	225	48.9

Tab.1: GPUs and FPGAs already available by vendors

In particular, the FPGA accelerator boards are now including 2/4 QSFP serial ports enabling the design of HPC systems based on heterogenous architectures with direct (e.g. n-dimensional torus) or indirect (switched) network topologies.

The heterogenous architectures based on FPGA got a level of flexibility able to set dynamically mixed precision methods suitable to save energy. The effect of precision on energy consumption will be studied. The energy efficiency can be further improved shifting thermal evaluation from the chip design phase to the run-time thermal management. In this context, accurate, fast and flexible thermal simulators help understanding the power dissipation requirements, tailoring the cooling to the chip requirements to best utilize HPC infrastructures while keeping cooling costs at a minimum and enabling run-time thermal management.

Starting from European Processor Initiative (EPI) activities are planned on the Stencil/tensor accelerator, boosting it using mixed-precision/trans-precision arithmetic and/or Posits and developing an accelerator with an hardware posit processing unit for HPC computation. Alternative to floats, Posits are promising to increase bandwidth and memory efficiency and hence boosting performance and saving power either for AI services (Posit8 instead of float16/32), or for scientific computation (posit16/32 instead of float64/128). IPs will be prototyped on reconfigurable technology.

In order to optimize the usage of the available resources, runtime services have to be deployed to take advantage of energy-saving features provided by the underlying hardware with respect to the application needs. The operating system plays a crucial role when it comes to energy efficiency. Most HPC systems use the Linux kernel which has an active community maintaining and updating it, thus providing further developments for new energy-saving features. The Resource Management System, which handles the system resources allocation and workload management, is highly customized for its integration with thermal management tools already available at node level. For example, many available scheduling systems can be optimized via pre-defined policies. Additionally, one could implement software-based support for specific energy saving or thermaql management

policies in the scheduling system itself or as a higher-level tool on top or aside of the scheduler. This higher-level tool could, for example, validate the workloads energy requirements to assure that certain energy driven policies are not violated before the workload is actually scheduled or to keep under control the thermal status of the computing elements.

In order to design HPC heterogeneous system running on FPGA accelerators, the availability of a High-Level Synthesis tool and the possibility to use in the flow pre-designed computation/interfaces IP is mandatory to lower the access barrier currently limiting the widespread adoption of FPGA devices.

One of the basic guidelines in energy efficient computing is the optimization and the acceleration of algorithms and software libraries that provide a reduction of the elapsed time of HPC applications and, as a consequence, 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 also for solving ever larger problems as required by high-resolution simulations and big data applications.

Within this context, new open-source high-performance algorithms and software libraries for some, among the most widely used, kernels in numerical linear algebra and graph computation, will be deployed. The library kernels will be of immediate use in a wide range of applications, ranging from simulation of phenomena driven by Partial Differential Equations to complex network analysis.

The algorithms will be designed and optimized having as target platforms clusters of hybrid nodes, with thousands of simple computing units and a memory hierarchy that is much more exposed to the developer's control with respect to the traditional multi-level cache-based systems. It is not unusual for algorithms, inefficient on traditional computing platforms, to become very much competitive on accelerators like GPU because additional computations are well tolerated and convenient when using complex memory access patterns. On the new platforms also data structures may need an in-depth revision. As for GPU, for instance, thread-locality rather than data-locality must be privileged.

## 2 Methodology

In general, a gap analysis is used to identify the discrepancies between the current state and the wished state of existing tasks in a project. As this project concern technological tasks interesting the whole co-design stack, from the top of software users' applications down to the bottom of the hardware platforms, a list of tasks gap analysis is reported as the following for each task:

- **CURRENT STATE** summarizing the current landscape of the task.
- **WISHED STATE** explaining the reasons for the improvement and advantages of it and whereas possible specifically to heterogeneous architectures of HPC systems and their own energy efficiency.
- **ACTION STATE** suggesting the steps and the solutions that can be deployed to achieve the wished states.
- **PRIORITY** prioritizing the steps in terms of their own importance.

Finally, a list with a summary of the actions for all tasks is reported to have an easy view of the activities is being to carry out in the project (section 22).

## 3 Task-1.1: User 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 targets HPC applications, and at the same time studying which improvements at lower levels could be beneficial to these applications. Task 1.1 addresses the gap analysis for the set of targets HPC applications and numerical building blocks defined in WP6 that will be ported on the project architecture and dissects the missing technologies that the project should provide to facilitate their transfer.

## 4 Task-1.1.1: New algorithm and software libraries

Nowadays main target platforms are clusters of hybrid nodes hosting accelerators and multi-core processors with wide-vector extensions therefore new algorithms and software modules for a Math library must be designed and implemented. These issues are, by now, quite common in current systems, from small and medium size servers to petascale supercomputers and fostered near-future exascale platforms. However, existing software does not exploit those computing resources at their best, especially when multiple nodes are used. Although the main target platform is made of hybrid nodes equipped with Nvidia GPUs, it's needed to start the development and benchmark, mini-app algorithms, also for the FPGA target platform.

Software modules for some of the so-called Colella's dwarves, who identified numerical methods crucial for science and engineering [Duff,2011] shall be implemented with efficiency and reliability.

Key numerical building blocks for very large (memory-bound) sparse matrix and graph computations that require multiple nodes shall be focused including:

- scalar vector products (dot operations), Sparse Matrix-Vector products (SpMV operations), Sparse Matrix Power (SpMP operation), Sparse Matrix-sparse Matrix products (SpMM operations);
- maximum weight/maximum cardinality (sparse) graph matching;
- Krylov-type iterative methods, such as Preconditioned Conjugate Gradient (PCG);
- Algebraic MultiGrid (AMG) preconditioners based on aggregation.

Furthermore, leveraging on task-based model, new parallel Math library algorithms shall be implemented as function kernels for FPGA target platform.

### CURRENT STATE

Many efforts are currently devoted to providing reusable, high-performance software components for scientific applications. These components provide efficient implementations of widely used algorithms on a variety of architectures and, at the same time, expose adaptable interfaces that enable easy integration into application codes coming from different scientific sectors. Here we limit our discussion to the main software development projects in the area of sparse matrix and graph

computations. More in details, we focus on the topics of the solution and preconditioning of sparse linear systems by iterative methods, and on matching and clustering in large and sparse graphs. Main efforts in this context are carried on in the US scientific community, within the Exascale Computing Project [ECP,2021], and include PETSc (Argonne National Laboratory) and Trilinos (Sandia National Laboratories), which provide specific solver components and infrastructures for coordinating component use; Ginkgo (University of Tennessee) and hypre (Lawrence Livermore National Laboratory), which provide key solver capabilities that can be used independently or as part of the PETSc [PETSC,2021] and Trilinos infrastructures [TRILINOS,2021]; ExaGraph [EXAGRAPH,2021], which focuses on main graph algorithms [Anztetal,2020], [Pothen,2018]. All the above-mentioned projects focus on novel algorithms and component design strategies that enable efficient utilization of the extreme degree of parallelism provided by manycore and accelerator processors via concurrency and memory system complexity. In this scenario is collocated the project of software development *PSCToolkit* by CNR, in collaboration with the University of Rome “Tor Vergata” and funded by EU within the Energy-oriented Center of Excellence [EoCoE,2021].

PSCToolkit (Parallel Sparse Computations Toolkit) [PSCTOOLKIT,2021] covers both the needs of having parallel basic algorithms for sparse matrices and graphs that can run on machines with thousands of cores and the setup of higher-level iterative solvers and preconditioners exploiting these capabilities.

The current version of PSCToolkit includes a set of different linear solvers, algebraic multigrid preconditioners, and tools for the handling of distributed linear algebra operations. It is written in modern Fortran exploiting object-oriented programming paradigm and offers a C-language interface. Some of the main kernels, such as dot and SpMV operations are already tuned for efficient exploitation of Nvidia GPU accelerators. Its potential to successfully meet the exascale challenge has been demonstrated both on model problems [D’Ambra,2021] and in real-world applications [Owen,2021], [Bertaccini,2021].

On the other hand, task-based method is well designed to parallelize applications on heterogeneous hardware composed of CPUs and GPUs [Sukkari,2017], [Mylykoski,2021], [Moustafa,2018], [Carpaye,2018]. Using dynamic schedulers that orchestrate the tasks, the runtime systems use both CPU and GPU and try to benefit from the specificities of each one. The availability of FPGAs opens new possibilities and asks several questions that should be studied to orient the design of the future HPC platforms.

## WISHED STATE

**WS-1.1.1a:** to develop a new generation of numerical algorithms and software tools in PSCToolkit which face the following computer architecture features and challenges:

- the cost of data movement dominates the cost of floating-point arithmetic; here the issue is to rethink numerical methods with the final goal to have implementations which reduce memory access and data communication among multiple processing units, and/or allow overlapping data movement and computation for latency hiding. An interesting specific use cases is to reduce movements in Krylov subspace methods, employing  $s$ -steps algorithms and/or enlarged Krylov subspaces as well as to test new approximation algorithms in graph algorithms for balancing accuracy and parallel performance.

- accelerators (GPUs, FPGAs, coprocessors) are becoming not only more powerful but also more usable, and heterogeneous architectures are increasingly prevalent; here the needs of exploiting very high levels of concurrency often require the substitution of more accurate methods with more parallel counterparts, because additional computations are well tolerated and convenient rather than using intrinsically sequential and complex memory access patterns.
- minimizing energy consumption is an increasingly important criterion for sustainability of HPC; one of the basic guidelines in energy efficient computing is the optimization and the acceleration of algorithms that provide a reduction of the elapsed time of HPC applications and, as consequence, a significant decrease in energy consumption. Here one issue is to integrate energy consumption information on specific architectures to analyze both performance and energy efficiency of different approaches using management software tools provided in this project to study possible algorithmic and implementation alternatives.
- low precision floating-point arithmetic is available in hardware on accelerators originally provided for machine learning, and offers greater throughput, albeit less accuracy; here the issue is how such lower precision computations can be exploited within an algorithm aiming for higher accuracy. Therefore, a start point is to test multiple precisions in basic sparse matrix operations and preconditioning of iterative solvers to accelerate the applications with the use of lower precision formats while maintaining the required accuracy of the output.

**WS-1.1.1b:** It shall be evaluated whether FPGAs are complementary to GPUs or could replace them and in which configurations for some Math library function kernels. Moreover, the study should be done by focusing on computing capability but also on energy consumption. While the existing studies have already demonstrated the potential of FPGAs for small test cases, there is a gap to push the validation and create a generic scheduler. Therefore, it is needed a clear description of the differences and complementarities between GPUs and FPGAs on a Math Library test case and to develop a scheduler able to obtain energy efficient executions on heterogeneous computing nodes equipped with CPUs, FPGAs and/or GPUs. With this aim, the *Heteroprio* scheduler [SCHED19] shall be implemented and use it in the Chameleon solver [Agullo,2012], [Agullo,2016], [Agullo,2011], [Agullo- Augonnet,2011] and Fast Multipole Method (FMM) application [Bramas,2019].

**ACTION STATE**

To address the wished states, the following actions shall be carried out.

	Design	Develop/implement	Test
WS.1.1.1a	Data movement in Krylov methods and (eventually) multiple precisions in basic sparse matrix operations	PoC of new algorithms in GPU and/or FPGA	test to evaluate -Time to solution -Energy to solution
WS.1.1.1b	Task based model for dense linear algebra and FMM kernels on FPGA	PoC Chameleon solver and ScalFMM on FPGA	test to evaluate -Time to solution -Energy to solution

**PRIORITY**

The highest priority is the development of efficient implementations on multiple nodes embedding GPUs of basic sparse matrix operations, such as SpMP and SpMM, to be used in communication avoiding Krylov methods and AMG preconditioners. Optimization of approximate parallel matching algorithms both for shared-memory architectures and for multiple GPUs has also a high priority to accelerate the setup of AMG preconditioners. Using mixed precision in basic kernels and preconditioning for Krylov solvers has a medium-level priority. Interactions with colleagues from other WPs are critical to experiment with software tools for FPGA implementations of some of the above operations and to analyze energy consumption of the proposed approaches, then these activities are considered with lower-level priority and will start as soon as the needed software tools from other WPs will be made available.

The most important step will be in the development of an efficient scheduler. It should be generic and remain efficient even if the hardware, the computational kernels, or even the target application change. Secondly, we should have an interface to access power monitors on FPGAs to be used by the scheduler. Finally, we will implement and optimize the BLAS routines needed by the Chameleon solver, and some FMM operators.

## REFERENCES

- [Duff,2011]: I.S. Duff, *European Exascale Software Initiative: numerical libraries, solvers and algorithms*, RAL-TR-2011-023, RAL Library STFC Rutherford Appleton Laboratory, 2011.
- [ECP,2021]: Home page - Exascale Computing Projects: <https://exascaleproject.org>
- [PETSC,2021]: <https://petsc.org/release/>
- [TRILINOS,2021]: <https://trilinos.github.io>
- [EXAGRAPH,2021]: <https://github.com/Exa-Graph>
- [Anztetal,2020]: H. Anzt et al., *Preparing Sparse Solvers for Exascale Computing*, Phil. Trans. R. Soc. A 378: 20190053, 2020. <http://dx.doi.org/10.1098/rsta.2019.0053>.
- [Pothen,2018]: A. Pothen et al., *ExaGraph: Graph Algorithms Enabling Exascale Applications*, SIAM Annual Meeting, 2018. Available at [Pothen\\_SiamAnnualMeeting2018\\_exagraph.pdf - Google Drive](#)
- [EoCoE,2021]: <https://www.eocoe.eu>
- [PSCTOOLKIT,2021]: <https://psctoolkit.github.io>
- [D'Ambra,2021] P. D'Ambra, F. Durastante, S. Filippone, *AMG preconditioners for Linear Solvers towards Extreme Scale*. SIAM Journal on Scientific Computing, Vol. 43, N.5, 2021. <https://doi.org/10.1137/20M134914X>.
- [Owen,2021]: H. Owen , G. Houzeaux , F. Durastante, S. Filippone, P. D'Ambra, *AMG4PSBLAS Linear Algebra Package brings ALYA One Step Closer to Exascale*, 32nd Parallel Computational Fluid Dynamics Conference (ParCFD21), 2021.
- [Bertaccini,2021]: D. Bertaccini, P. D'Ambra, F. Durastante, S. Filippone, *Solving Richards Equations for Extreme Scale Applications in Hydrology*, SIAM Conference on Mathematical & Computational Issues in the Geosciences (SIAM GS21), 2021.
- [Sukkari,2017]: Sukkari, D., Ltaief, H., Faverge, M., & Keyes, D. (2017). *Asynchronous task-based polar decomposition on single node manycore architectures*. IEEE Transactions on Parallel and Distributed Systems, 29(2), 312-323.
- [Myllykoski,2021]: Myllykoski, M., & Kjelgaard Mikkelsen, C. C. (2021). *Task-based, GPU-accelerated and robust library for solving dense nonsymmetric eigenvalue problems*. Concurrency and Computation: Practice and Experience, 33(11), e5915.

- [Moustafa,2018]: Moustafa, S., Kirschenmann, W., Dupros, F., & Aochi, H. (2018, August). *Task-based programming on emerging parallel architectures for finite-differences seismic numerical kernel*. In European Conference on Parallel Processing (pp. 764-777). Springer, Cham.
- [Carpay,2018]: Carpaye, J. M. C., Roman, J., & Brenner, P. (2018). Design and analysis of a task-based parallelization over a runtime system of an explicit finite-volume CFD code with adaptive time stepping. *Journal of Computational Science*, 28, 439-454.
- [Agullo,2012]: Agullo, E., Augonnet, C., Dongarra, J., Ltaief, H., Namyst, R., Thibault, S., & Tomov, S. (2012). *A hybridization methodology for high-performance linear algebra software for GPUs*. In GPU Computing Gems Jade Edition (pp. 473-484). Morgan Kaufmann
- [Agullo,2016]: Agullo, E., Bramas, B., Coulaud, O., Darve, E., Messner, M., & Takahashi, T. (2016). *Task-based FMM for heterogeneous architectures*. *Concurrency and Computation: Practice and Experience*, 28(9), 2608-2629.
- [Agullo,2011]: Agullo, E., Augonnet, C., Dongarra, J., Faverge, M., Ltaief, H., Thibault, S., & Tomov, S. (2011, May). *QR factorization on a multicore node enhanced with multiple GPU accelerators*. In 2011 IEEE International Parallel & Distributed Processing Symposium (pp. 932-943). IEEE.
- [Agullo- Augonnet,2011]: Agullo, E., Augonnet, C., Dongarra, J., Faverge, M., Langou, J., Ltaief, H., & Tomov, S. (2011, December). *LU factorization for accelerator-based systems*. In 2011 9th IEEE/ACS International Conference on Computer Systems and Applications (AICCSA) (pp. 217-224). IEEE.
- [Bramas,2019]; Bramas, B. (2019). Impact study of data locality on task-based applications through the Heteroprio scheduler. *PeerJ Computer Science*, 5, e190.

## 5 Task-1.1.2: AI and HPDA

Heterogeneous computing with CPUs, GPUs and FPGAs is a well-established pattern in the High Performance Computing field, thanks to the huge quantity of resources available on state-of-the-art FPGAs and increasing computing power of GPUs by now an essential part of a modern artificial intelligence infrastructure.

Large experiments in High Energy Physics (HEP) have already designed heterogeneous systems GPU/FPGA based for the NA62 experiment at CERN [Ammendola,2018]. The GPU-RICH system orchestrates CPU-GPU-FPGA in a single computing node, exploiting the GPUDirect RDMA capabilities of NVIDIA Tesla GPUs, to inject an input data stream arriving from a detector front-end, directly into the GPU memory. Data from the network is received by a low-latency PCIe RDMA FPGA-based NIC, supporting different network link technologies. Future experiments, which will handle high volume of data and high costs, push the need for new techniques in TDAQs 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 can perform 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.



Neuronal simulation in HPC can benefit from heterogeneous architectures for modelling the human brain cortex. INFN is focusing three code bases: 1) the first being the Distributed Polychronous Spiking Neural Network simulator (DPSNN) [Pastorelli,2019], which was developed in-house and employed in many different ways, ranging from architecture co-design and benchmarking in the EURETILE project to studies in Slow Wave Activity in WaveScaleS, 2) while the other two are the Neural Simulation Tool (NEST) [Gewaltig,2007], a well-established name in the field of neuronal simulators, and 3) NeuronGPU [Golosio,2021], a more recent GPU-centric development that shows excellent promise and good chances of a much larger impact.

While DPSNN is a high-performance C/C++ MPI code, versatility and programmability were not foremost in its design; on the contrary, NEST is a framework that provides a feature-rich environment where a neuroscientist can build and simulate very complex electrophysiological experiments with just a few lines of Python scripting. Of course, this ease of use comes at the price of quite suboptimal efficiency when the scale of the said experiment is pushed from a single workstation to more than a few nodes of an HPC cluster. NeuronGPU takes the same approach as NEST - the experimenter describes by means of a Python script the neural system whose dynamics is then simulated by the framework - with the difference that, being designed to run on a GPU, it employs algorithms that exploit its strengths to achieve performance improvements in the range of tenfold and more when compared to NEST.

## CURRENT STATE

There is a plethora of Deep Learning (DL) frameworks available for AI algorithms applying to large dataset. Training DL networks requires efficient implementations of standard computational kernels on tensors (convolutions, multiplication, normalization, and many others), and of the derivative of these kernels. Both *TensorFlow* [TENSORFLOW,2021] and *Pytorch* [PYTORCH,2021] provide Python interfaces to such implementations (based on CUDA), and an automatic differentiation mechanism that allows to compute derivative of compositions of kernels. The training process is expressed very simply via adapted interfaces, such as: *Keras* [KERAS,2021] for TensorFlow and *Lightning* [LIGHTNING,2021] for PyTorch. These frameworks also provide automatic mixed-precision capabilities to accelerate the computations which do not require high precision. Data Parallel training is available, with a BSP-style communication pattern: all nodes perform a training step on a batch of inputs, followed by an allreduce collective operation. Finally, both frameworks provide just-in-time [XLA,2021] (Accelerated Linear Algebra) compilation, allowing to generate optimized GPU/TPU kernels specific to the user's application.

There are also high-performance deep learning inference frameworks using GPU for high throughput (e.g. TensorRT) but they are based on commercial software and it is not possible to insert code aimed to allow deterministic low-latency reconfigurable data-transport path. Furthermore for specific AI applications, FPGA devices offer better performances and reduced energy consumption using quantization on neural networks [Lammie,2016], [Jacob,2018], [Coelho,2021], [Han,2016].

Recently Microsoft has made available Project Brainwave [BRAINWAVE,2021] offering a deep learning platform for real-time AI inference in the cloud and on the edge. Unfortunately, it is based on pre-trained NN models that cannot be inserted in a custom pipeline merging computing and data-transport.

Over the last years more advanced High-Level Synthesis (HLS) tools for FPGA [Intel-HLS, Xilinx-HLS] have already established themselves as a useful design strategy for AI and Machine Learning (ML).

Besides, the HLS4ML python-based package [HLS4ML] bridges HLS Tools and TensorFlow/Keras and PyTorch libraries to create firmware implementations of machine learning algorithms using high level synthesis language (HLS). This package allows developers to translate traditional open-source machine learning package models into HLS that can be configured for the specific use case. To reduce FPGA resources used by the NN implementation, we are presently exploring the quantization of our models, using fewer bits to represent weights and biases leveraging on Qkeras, an extension to Keras. The challenge is to balance the decline in performance and a quantized version of a deep neural network model reduced in size, latency and energy consumption.

The recent aforementioned technological developments, with the availability of IP focused on low latency communication between FPGAs, create an opportunity to finalize a more flexible and powerful programming model to exploit the architectural peculiarities of these devices. Due to the intrinsic reconfigurability, modularity and scalability, such a distributed system can be employed in many different use cases.

The LOTP+ FPGA-based design hosted on a VCU118 Xilinx board is the upgraded version of the NA62 low-level trigger system developed by INFN APE Lab and is currently deployed at the experiment for validation in parallel with the previous version. We started the design and prototyping of a Neural Network (multiple Dense Layers) implemented on FPGA to count the number of tracks (rings) in the NA62 RICH detector to identify the number of charged particles in a single physics event, and to assess the number of electrons within that set of particles (partial particle identification).

NEST is a well-tested, established codebase while NeuronGPU is a new development. While NEST supports either multithreading (via OpenMP) and multi-node (with MPI) parallelization in a straightforward way, it does not support running on GPUs. On the other way, NeuronGPU is designed from scratch to run on GPUs (it employs GPU-optimized algorithms and design choices in order to do this efficiently) but, although the syntax and API for its Python scripting are mostly clones of NEST's (which is on purpose, so that the same script can be run on either simulator), they are not identical; NeuronGPU does not (yet) support the whole range of NEST functions (neuron and synapse models, probability distributions, topologies, integration schemas...) and diverges somewhat in the syntax of others, meaning that at the current moment a non-trivial script usually needs some adjusting when ported from NEST to NeuronGPU. Moreover, NeuronGPU does support running multi-node (via MPI) but the coordination among the many processes is not as integrated and transparent as it is built-in in NEST, so that using multiple GPUs on a node or GPUs on remote nodes is an added effort laid upon the script and therefore the experimenter. Even with these shortcomings, a detailed comparison designed to be fair to the differences between the codebases and spanning a range of different sizes and types for populations of neurons and their connections (see [Golosio,2021]) shows that NeuronGPU – running on an Nvidia Tesla V100 – can be from 16x to 59x faster than NEST – running on an Intel Core i9-9900K – and it is from 32% to 59% faster when compared to other GPU-centric simulators like GeNN [Yavuz,2016] and CarLSIM [Chou,2018], coming very near to real-time capability.

Finally, AI and HPDA applications can leverage on *StarPU* [STARPU,2021]. It is a task-based runtime system, written in C. StarPU can handle execution of task-based applications on heterogeneous

architectures, typically made of CPU cores and accelerators (like NVIDIA GPUs for example). StarPU performs automatic task placement thanks to efficient dynamic algorithms which allow to perform each task on the most efficient computing resource. StarPU also automatically performs all necessary data movement, even in a distributed context.

## WISHED STATE

A Proof-of-Concept shall be provided as a typical HEP (High Energy Physics) real-time AI-based data analytics on heterogeneous distributed system. This system, named RAIDER, will have a distinguishing feature in its capability of performing this distributed inference scheme with real-time constraints. Instead, neuronal simulator applications can efficiently perform on specialized hardware and efficiency in term of energy saving shall be proofed. Finally, a task-based programming model can be used as an efficient orchestrator to optimize the usage of heterogeneous hardware.

**WS-1.1.2a:** 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 will lead us to the definition of the requisites of a dedicated development framework for this class of applications. 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.

**WS-1.1.2b:** With such shrinking in runtimes, although Thermal Design Power (TDP) figures for the GPU (currently from 300W for high range to 400W for top range GPUs) are usually larger compared to those for cluster-class Intel CPUs (between 100W and 200W), significantly improved time-to-solution and energy-to-solution are expected to be easily achieved for neural simulations on such heterogeneous architectures. The margin for improvement can even be expected to increase on an HPC platform designed around low-power, high compute density GPUs and a tight integration with the network fabric.

Since the NEST Initiative Association is discussing the feasibility of integrating NeuronGPU into the NEST framework (see [Golosio,2020]), the roadmap to a neural simulation environment that

enjoys both the versatility of NEST and the speed of NeuronGPU – when GPUs are available – seems, at least in the long term and from the software development and maintenance side, already traced. In the short term, a prototype neural simulation must be devised that can run on a cluster either on NEST or NeuronGPU in order to have a baseline for meaningful comparisons between the two. This work must accommodate the discrepancies between their semantics, the greatest being the multi-node syntax for NeuronGPU, which is still a work-in-progress at the present time and only partially documented. Moreover, the precise power measurements which are required to assess energy-to-solution and power efficiency will need an instrumented setup where these simulations are to be run.

**WS-1.1.2c:** to consider the implementation a prototype of an inference phase of some NNs over StarPU independently from target architecture: CPU/GPU/FPGA, selecting dynamically the Processor Elements for the executions of the tasks using tools as Rotor [ROTOR,2021] able to mix re-materialization & offloading performing data or model parallelism.

**ACTION STATE**

The actions for achieving the wished states are as follow:

	Design	Develop/implement	Test
WS.1.1.2a	Integration INFN interconnection IPs for FPGA Xilinx and stack software	PoC of NNs in FPGA Xilinx with high level programming model offering abstraction for communications	test to evaluate -Processing throughput -Energy to solution
WS.1.1.2b	Neuronal simulation test case	PoC of Nest and NeuronGPU in GPU	test to evaluate -Time to solution -Energy to solution
WS.1.1.2c	Allocation algorithms	PoC NNs over StarPU in FPGA	test to evaluate -Time to solution -Energy to solution

**PRIORITY**

The integration and customization (e.g. the number of ports provided by our switch) of INFN interconnection IPs in the Xilinx Vitis framework for the reference Xilinx Alveo boards is the prerequisite for all subsequent developments. It will leverage the HLS framework to build mixed HLS-HDL designs to test extensively the ported IPs and to acquaint ourselves with the framework to ease the design of the HW/SW interface (i.e. the inter-HLS kernels communication primitives and the control and monitoring mechanisms to be implemented).

About neuronal simulation, at beginning a generic simulation must be defined, in type and connectivity for the neurons and in size for their network in order to obtain something wiely but with a timescale that can reflect the actual performances when a realistic simulation is up to speed. Second, this simulation must be implemented in such a way as to be able to run on either NEST and NeuronGPU with as few modifications as possible between the two, to maintain coherency. Third, an appropriate instrumentation (as hardware and/or software combination might that be) is to be defined that makes us able to give precise estimates as to the power consumption of the platform the simulation is run on.

Finally a PoC implementing a NNs over StarPU in FPGA devices is an action at high priority.

## REFERENCES

- [Ammendola,2018]: 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*, in Journal of Physics: Conf. Series 1085 (2018) 032022 doi :10.1088/1742-6596/1085/3/032022
- [Pastorelli,2019]: Pastorelli, E. et al., *Scaling of a Large-Scale Simulation of Synchronous Slow-Wave and Asynchronous Awake-Like Activity of a Cortical Model With Long-Range Interconnections*. Front. Syst. Neurosci., 23 Jul. 2019 – <https://doi.org/10.3389/fnsys.2019.00033>
- [Gewaltig,2007]: Gewaltig, M.-O. & Diesmann, M. *NEST (Neural Simulation Tool)*, Scholarpedia 2(4):1430, 2007 – <http://dx.doi.org/10.4249/scholarpedia.1430>
- [Golosio,2021]: Golosio, Bruno et al., *Fast Simulations of Highly-Connected Spiking Cortical Models Using GPUs*. Front. Comput. Neurosci., 17 Feb. 2021 - <https://doi.org/10.3389/fncom.2021.627620>
- [TENSORFLOW,2021]: <https://www.tensorflow.org/>
- [PYTORCH,2021]: <https://pytorch.org/>
- [KERAS,2021]: <https://www.tensorflow.org/guide/keras>
- [LIGHTNING,2021]: <https://www.pytorchlightning.ai/>
- [XLA,2021]: <https://www.tensorflow.org/xla>
- [ROTOR,2021]: (<https://gitlab.inria.fr/hiepac/rotor>)
- [Lammie,2016]: C. Lammie, A. Olsen, T. Carrick and M. Rahimi Azghadi, *Low-Power and High-Speed Deep FPGA Inference Engines for Weed Classification at the Edge*, in *IEEE Access*, vol. 7, pp. 51171-51184, 2019, doi:10.1109/ACCESS.2019.2911709.
- [Jacob,2018]: Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, Dmitry Kalenichenko. *Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2704-2713
- [Coelho,2021]: Coelho, C.N., Kuusela, A., Li, S. et al. *Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors*. Nat Mach Intell 3, 675–686 (2021). <https://doi.org/10.1038/s42256-021-00356-5>
- [Han,2016]: Han, S., Mao, H., & Dally, W. (2016). *Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding*. in “4th International Conference on Learning Representations, ICLR, 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings”, <https://arxiv.org/abs/1510.00149>
- [BRAINWAVE,2012] <https://www.microsoft.com/en-us/research/project/project-brainwave/>
- [Fahim,2021]: Fahim, Farah, et al. *hls4ml: An Open-Source Codesign Workflow to Empower Scientific Low-Power Machine Learning Devices*. arXiv:2103.05579 [physics], marzo 2021. [arXiv.org, http://arxiv.org/abs/2103.05579](http://arxiv.org/abs/2103.05579).
- [Yavuz,2016]: Yavuz, E. et al., *GeNN: a code generation framework for accelerated brain simulations*, Sci. Rep. 6:18854, 2016. doi: 10.1038/srep18854
- [Chou,2018]: Chou, T.-S. et al., *CARLsim 4: an open-source library for large scale, biologically detailed spiking neural network simulation using heterogeneous clusters*. International Joint Conference on Neural Networks (IJCNN) (Rio de Janeiro), 2018. doi: 10.1109/IJCNN.2018.8489326

[Goloso,2020]: Goloso, B. et al., *Toward a possible integration of NeuronGPU in NEST*, NEST Conference 2020, 7 (Aas). doi: 10.5281/zenodo.4501615

[STARPU,2021]: <https://starpu.gitlabpages.inria.fr/>

## 6 Task-1.1.3: Scientific flagship codes

A list of flag-ship codes shall be tackled into account in this task in order to evaluate their own efficiency to perform on heterogeneous systems equipped with GPU/FPGA.

### Tensor Network Methods

INFN is going to develop also efficient software modules for simulating quantum circuits via Tensor Network algorithms. In particular, we are going to explore the possibilities offered by High Performance Computing (HPC) platforms equipped with GPUs. These components could drastically reduce the computational time required by the basic tensor operations, such as tensor multiplications, contractions and singular value decompositions. With the opportunity to work on HPC clusters with multiple GPUs, we aim to push the limits of tensor networks implementations and thereby set a new benchmark for simulating quantum circuits and quantum many-body systems.

### High-Energy Physics algorithms

HEP experiments require large-scale computing systems in order to process, select and analyze the incoming data. INFN, as a major global leader in their design, realization and operations, has a vast interest in the realization of future-looking solutions, also in order to keep the cost of such systems affordable. Two main directions are under investigation in order to ease the computing of the next generation of experiments:

- The use of technical solutions with a better cost/performance ratio, such as GP-GPUs, FPGA, TPUs, down to ASICs;
- The use of solutions with a lower operational cost, with a better profile for energy per operation (ARM-based solutions, for example).

In both cases, the major obstacle comes from the huge codebase already existing (about 10 million lines of source code per experiment), which makes it difficult to fine tune algorithms for more than one specific solution (the current one being Intel CPUs).

The solution technology seems to provide it via the availability of high-level frameworks/toolkits/libraries that decouple the user-level code to the actual executable code, via a translation capability to happen either at compile or run time to provide (optimized) executable codes for multiple platforms. These tools hold the promise to provide a solution for current and future platforms, if a proper backend will be provided.

The most promising technologies on the market today are Alpaka, Kokkos and SYCL (as available, for example, in oneAPI/DPC++); each has different capabilities, support model and number of supported backends.

### Air Pollution

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 model, running at larger scale (e.g. mesoscale), with a city-scale geophysical flow solver (EULAG) for accurate prediction of contaminant (e.g. NO<sub>2</sub>, PM<sub>2.5</sub>, PM<sub>10</sub>) transportation through the street corridors, over buildings and obstacles.

## CURRENT STATE

### Tensor Network Methods

Efforts are currently devoted to explore the different strategies to implement basic tensor network operations, such as tensor multiplications and contractions, on High Performance Computing architectures equipped with GPUs. Some preliminary results for the tensor multiplications performed on GPUs have already been collected.

### High-Energy Physics algorithms

The experiments have already started (since 2017 at least) the investigation for possible long-term solutions, not implying the constant rewriting of the codebase. INFN, as a fundamental partner in the experiments, has internal activities in the same direction covering not only the HEP domain. While many of the activities are still internal to the experiments, a few have public results [Cappelli,2020], [Childers,2021].

These efforts set a framework for comparing the codes produced via manual recoding with automatically produced (pseudo) code, and to assess the difficulty to port existing codes to the high-level toolkits.

The CMS collaboration, one of the major experiments at CERN, has provisionally decided to use Alpaka for the period 2022-2025; hence, larger scale evaluations will be available shortly.

### Air Pollution

UrbanAir is a multiscale application for assessing and predicting air quality over complex urban areas. Based on the WRF, a mesoscale weather prediction community model, and EULAG, all-scale geophysical flow solver, it allows to precisely model contamination transportation (as a passive tracer) through the complex building structures. The application is parallelized using Message Passing Interface (MPI) and it is proven to scale very well. PSNC is going to focus on the latter, EULAG, model. Some preliminary tests on GPUs were conducted for the GCRK routine of EULAG, delivering promising results in terms of performance and energy efficiency, [Rosa,2014], [Ciznicki,2015]. Moreover, an uncertainty of input parameters has been studied [Wright, 2020], [Groen,2021], [Suleimenova,2021]], which increases the importance of usage of accelerators or mixed-precision techniques.

## WISHED STATE

### Tensor Network Methods

**WS-1.1.3a:** For this application a new generation of tensor networks algorithms for the simulation of quantum circuits and quantum many-body systems shall be developed exploiting the current high-performance capabilities offered by GPUs. An implementation of a low-level tensor operations in the software modules on GPU for the complete simulation of quantum circuits and quantum many-body systems shall be tested as PoC.

### High-Energy Physics algorithms

**WS-1.1.3b:** a definition of the strong / weak points for each platform, and the difficulty of its use on the user side, is urgent in the field, since a late decision would imply recoding more than once. Hence, studies in order to better understand features, pro and cons, support

model of each solution is wished. To develop the final hiring phase with a single-source programming framework like SYCL or Alpaka in order to set a coherent test bench as PoC able to compare at least CPU and GPU codes.

**Air Pollution**

**WS-1.1.3c:** EULAG scales good, albeit the modelling of passive tracers impacts the overall efficiency. Moreover, lack or poor quality of input data is a shortcoming, which may be resolved by providing uncertainty quantification of the input parameters. Then, instead of having just one simulation, it shall run a several or dozen ones called ensembles, after which the results are, for instance, averaged. The number of ensembles is related to number of sampled parameters. In this case it is crucial to provide even more efficient HPC code to be able to deliver results in a reasonable amount of time. There is an opportunity to take advantage of the accelerators, GPUs in particular, to speed up the most time-consuming parts of the code. Moreover, there is a possibility to benefit even more by utilizing FPGA and lower precision calculations, as reported in [Rojek,2017]. The most-computational intensive part of the code should be selected to be analyzed for the possibility of being adapted to GPUs and benefit from lower precision computations. The analysis should take into account data structures to identify which should be changed into single or mixed precision to bring additional speedup. The analysis of trade-off between lower precision and quality of results should be compared by means of the development and implementation of PoC. Last but not least is the analysis of other project approaches the code can benefit in terms of computational and energy efficiency.

**ACTION STATE**

The actions need to achieve an assessment of the foregoing wished states during the activities of the project is simply listed in the following table:

	Design	Develop	Implementation	Test
WS-1.1.3a	Low level tensor operations	Kernel functions	PoC on GPU	Time to solution Energy to solution
WS-1.1.3b	HEP computing algorithms	Kernel Functions in SYSCl/Alpaka	PoC on GPU	Time to solution Energy to solution
WS-1.1.3c	EULAG computing algorithms	Kernel functions	PoC on GPU	Time to solution Energy to solution

The implementation and test actions must be carried out on GPU servers already available by project partners.

**REFERENCES**

[Cappelli,2020]: <https://amslaurea.unibo.it/22171/>  
 [Childers,2021]: T.Childers, M.J.Kortelainen, M.Kwok, A.Strelchenko, Y.Wang. *Porting CMS Heterogeneous Pixel Reconstruction to Kokkos* . Submitted to 25th International Conference on Computing in High-Energy and Nuclear Physics (vCHEP 2021) <https://arxiv.org/abs/2104.06573>  
 [Rosa,2014]: B. Rosa, M. Ciznicki, K. Rojek, D. K. Wojcik, P. K. Smolarkiewicz, R. Wyrzykowski. *Porting Multiscale Fluid Model EULAG to Modern Heterogeneous Architectures*, International Journal of Applied Physics and Mathematics 01/2014; 4(3):188-195. DOI: 10.7763/IJAPM.2014.V4.281



[Ciznicki,2015]: M. Ciznicki, M. Kulczewski, P. Kopta, K. Kurowski. *Methods to Load Balance a GCR Pressure Solver Using a Stencil Framework on Multi- and Many-Core Architectures*. Scientific Programming, DOI 10.1155/2015/648752

[Rojek,2017]: Rojek K, Wyrzykowski.R. *Improving energy efficiency of MPDATA on GPU-based supercomputers using mixed precision arithmetic*. 2nd Workshop on Power-Aware Computing 2017 (PACO2017)

[Groen, 2021]: D. Groen, H. Arabnejad, V. Jancauskas, W. N. Edeling, F. Jansson, R. A. Richardson, J. Lakhilili, L. Veen, B. Bosak, P. Kopta, D. W. Wright, N. Monnier, P. Karlshoefer, D. Suleimenova, R. Sinclair, M. Vassaux, A. Nikishova, M. Bieniek, Onnie O. Luk, M. Kulczewski, E. Raffin, D. Crommelin, O. Hoenen, D. P. Coster, T. Piontek and P. V. Coveney, *VECMAtk: a scalable verification, validation and uncertainty quantification toolkit for scientific simulations*, Phil. Trans. R. Soc. A. 379, 20200221 (2021), DOI:10.1098/rsta.2020.0221

[Suleimenova, 2021]: D. Suleimenova, H. Arabnejad, W. N. Edeling, D. Coster, O. O. Luk, J. Lakhilili, V. Jancaskas, M Kulczewski, L. Veen, D. Ye, P. Zun, V. Krzhizhanovskaya, A. Hoekstra, D. Crommelin, P. V. Coveney, D. Groen, *Tutorial applications for Verification, Validation and Uncertainty Quantification using VECMA toolkit*, J. Comput. Sci. 53, 101402 (2021), DOI:10.1016/j.jocs.2021.101402

## 7 Task-1.2: Runtime Services

Usually in a HPC infrastructure the runtime services ensure that application requirements are dynamically satisfied and mapped onto system resources including execution models with workload handling resources, fault tolerance and IO management. These services are well established in traditional HPC data center and the users, mainly belonging to academic and big enterprises research, are aware enough to use the computing resources. Because in the last years there has been a growth of edge applications 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 increases and the types and volumes of available data grow, these new classes of applications, seamlessly combining real-time data with complex programming models and data analytics to monitor and manage systems of interest, are being explored. However, these applications require a malleable integration of resources at the edge, the core, and along the data path to support dynamic and data-driven application workflows, that is, they need to leverage a cloud computing continuum.

Furthermore, the proliferation of edge devices (Gartner estimates that there will be 25.1 billion Internet of Things (IoT) end points by 2021; Jones, 2018) and the need to gather and process the “rich” data being produced by smart sensors are leading to significant investment in edge computing to support timely processing close to the data sources (Shi et al., 2016), as well as a rich edge computing research agenda addressing issues such as performance, latency, fault tolerance, interoperability, security, and privacy.

## 8 Task-1.2.1: Resources Management

### CURRENT STATE

The resources management in heterogeneous architectures might use the accelerator devices equipped in a compute node resulting in unwanted behaviors. In some cases the resources management system is not able to handle the fine-grained resources compute node level. Typical case is the unpinned layout of more GPUs in a multi-socket node on which CPU and GPU coming from different sockets result in a slowdown in communication and bad performance.

Some projects as RECIPE [Fornaciari,2018] address these topics by applying runtime resource management techniques leveraging resource virtualization and disaggregation to provide the ability to partition and allocate resources at a finer grain than usually provided by job schedulers, even partitioning accelerators devices. This approach improves resource usage, capacity computing approaches allow multiple applications to coexist on the same compute node, sharing the heterogeneous resources based on each application's priority and Quality of Service (QoS) requirements. Thus the proposal of a *hierarchical approach* to resource management shall be the best practice for heterogeneous architectures, where the top level (job scheduling) keeps the same capabilities and interface as the state of the art, while enhancing it at a finer grain, without undue impact on application programming. RECIPE does that, designing a hardware abstraction layer at compute node level which enables the fine-grained management of disaggregated resources, such as deeply heterogeneous accelerators, including hardware accelerators, GPU and FPGA-based accelerators, and symmetric multiprocessors, while exposing the necessary set of hardware monitors.

The current Resources Management Systems (RMS) can be leveraged to share a collection of heterogeneous resources among the jobs in execution in a HPC cluster. However, they are not designed to handle resources such as GPU/FPGA. Concretely, although RMSs can use a generic resource plugin to manage accelerator devices, they can only be accessed by the job that is in execution on the node to which the accelerators are attached. This is a serious constraint for remote accelerators virtualization technologies able to deploy Cloud-Edge HPC capability, which aim at providing a user-transparent access to all heterogeneous hardware in HPC cluster, independently of the specific location of the node where the application is running with respect to the heterogeneous node. Nowadays the virtualization infrastructures allow to establish an abstract hardware layer between physical and virtualized resources for achieving a high flexibility dynamic system able to allocate functions as service with a centralized management that improve efficiency and reduce costs. Therefore, the key word for cloud-edge HPC services is: *Virtualize anything and everywhere*.

Up to few years ago in HPC landscape, virtualization meant additional overheads with performance loss as well as unavailability of virtualized fabric drivers to interconnect compute nodes. As these reasons, so far virtualization solutions were banned in HPC large systems confined the usage to functional services for HPC.

Currently, Mellanox, the most important player in Infiniband fabric, provides *Virtual-IQ* (Virtual Intelligent Queuing technology) which is available on ConnectX® and ConnectX-2® adapters and allows I/O consolidation over Mellanox InfiniBand and 10 Gigabit Ethernet. Virtual-IQ technology

supports hardware assisted direct Virtual Machine (VM) access, VM to VM switching, advanced memory management, traffic filtering (LAN, SAN, VM migration, and console management), QoS, improved server utilization, and segregation of all traffic types across multiple virtual NICs (vNICs) and virtual HBAs (vHBAs). All these feature support includes guaranteed programmable sustained bandwidth in both transmit and receive directions for each virtual connection with no changes to data center infrastructure.

Concerning the GPU hardware, there exist several tools, reported in [Iserio,2014], for remote GPU virtualization, based either on the CUDA or OpenCL APIs. Typically, these tools provide to applications to run CUDA/OpenCL kernels are not aware that their requests are intercepted by the corresponding GPU virtualization middleware and redirected to a real GPU, which is generally located in a remote node of the cluster. Although remote GPU virtualization has demonstrated very low overhead with respect to a configuration with a local GPU [Reano,2013], due to its novelty, this technology is not yet supported by the job schedulers that are commonly encountered in production clusters. In particular, some commercial RMSs in production today, such as: PBS or LSF, only deals with real GPUs so that, when a job requests a number of nodes equipped with one (or more) GPU(s), the scheduler will try to map that job to nodes that actually own the requested number of GPUs, thus impairing the benefits of GPU virtualization. Instead with SLURM, an open-source RMS, is possible to modify the scheduler, so that it becomes aware of the fact that the assignment should no longer be constrained by the GPU kernels having to be executed in the same node where the invoking application is mapped to. Therefore, SLURM is able to handle GPU virtualization allowing applications to leverage all the cluster GPUs, independently of their location that is the efficient way to deploy HPC resources in a cloud-edge continuum.

Therefore the GPU virtualization is well supported as aforementioned. It is well supported at commercial level by GPU vendors. NVIDIA GPUs for virtualization (vGPU) is a tool providing virtual GPUs that can be shared between virtual nodes running on any physical node, anywhere. NVIDIA Virtual Compute Server (vCS) software virtualizes NVIDIA GPUs to accelerate compute-intensive workloads, including over 600 GPU-accelerated applications for AI, deep learning, data science, and HPC. vCS gives data center admins the ability to manage GPU clusters with standard server virtualization management applications, maximizing GPU utilization and ensuring security.

A profound and flexible integration of FPGAs into scalable data center infrastructures which satisfy the cloud-edge HPC as service is a task of growing importance in the field of energy efficient cloud-edge continuum. In order to achieve such an integration, the virtualization of FPGA resources is necessary. The provision of virtual FPGAs (vFPGAs) makes reconfigurable resources available to customers of the data center provider. Nowadays, FPGAs have grown in size and full utilization of the devices cannot always be achieved in practice. One possibility to increase utilization is our virtualization approach which allows for flexible design sizes and multiple hardware designs on the same physical FPGA. One challenge of this approach are the unsteady load situations of elastic clouds, which process short- and long-running acceleration services.

The items of FPGA virtualization are similar to the core objectives that resulted in the development of virtualization used in traditional CPU/software systems including Accelerator as Service (AaS) [Vaishnav,2018]. The main items are:

- Abstraction: FPGAs must be exposed to the cloud stack as a resource pool that can be actively managed, i.e. it can be requested, allocated and deallocated by a tenant. Its usage must be tracked in order to facilitate billing that is associated with the public cloud model. In addition, once provided to a tenant, the FPGA must be programmable by the tenant similar to other resources such as CPUs and GPUs. However, traditional system software stacks, i.e. operating system and hypervisor, consider FPGAs only as fixed functional acceleration devices while ignoring their nature of programmability.
- Multi-tenancy: Ability to serve multiple different users using the same FPGA fabric.
- Runtime Resource Management: Providing an abstraction/driver layer to the FPGA fabric and means of scheduling tasks to the FPGA as well as monitoring its resource usage with a high granularity level.
- Flexibility: Ability to support a wide range of acceleration workload i.e. from custom accelerators to framework specific accelerators designed in a High-Level Language (HLL) or a Domain Specific Language (DSL).
- Isolation: Providing the illusion of being a sole user of the FPGA resources for better security, fewer dependencies and correctness of the program execution.
- Sharing: FPGA resources should follow this model and enable sharing among multiple tenants and their applications in order to maximize resource utilization.
- Scalability: The system/application can scale to multiple different FPGAs or can support multiple different users at relatively low overhead.
- Performance: The impact of virtualization should be minimal on performance achievable and FPGA resources usable by the user application.
- Energy efficiency: workloads distributed in compute nodes equipped with multi-GPU/FPGA improves the energy efficiency of the whole HPC system.
- Security: Ensuring information of other tenants is not leaked and for safekeeping the infrastructure from malicious users. As FPGAs were not designed for multitenancy but for single users, security features must be introduced in order to enable FPGA usage in the cloud. FPGA accelerators typically run with full hardware access and hence a single malicious tenant can bring down a complete shared compute host. Though there are techniques to ease the impact and initially enable FPGAs in the kernel, security is best addressed by FPGA manufacturers through additional hardware changes.
- Faults Tolerance: Ability to keep the system/service running despite failures.
- Programmer's productivity: Improving the time to market and reducing the complexity of deploying a design to an FPGA from its software description. User applications have dependencies on the ecosystem (tools, libraries) that support FPGA usage. Typically there is a tight coupling between specific FPGAs, their tool chains and the applications and libraries that are written for a specific FPGA. In many cases there are no standard Application Binary Interfaces (ABI) yet released. To enable transition into the cloud, one must provide the ecosystem and the SDKs in the same manner as they are available in stand alone environments.

To guarantee effective control of QoS, performance aspects, reliability, power and thermal aspects, the RECIPE runtime resource management system proposed a novel hierarchical proactive-reactive approach. The Figure 1.2.1.1 shows a high-level overview of the full hierarchical RMS, which can be split between: *i*) the node-level (or local) runtime resource manager (LRM), and *ii*) the infrastructure-level (or global) resource manager (GRM).

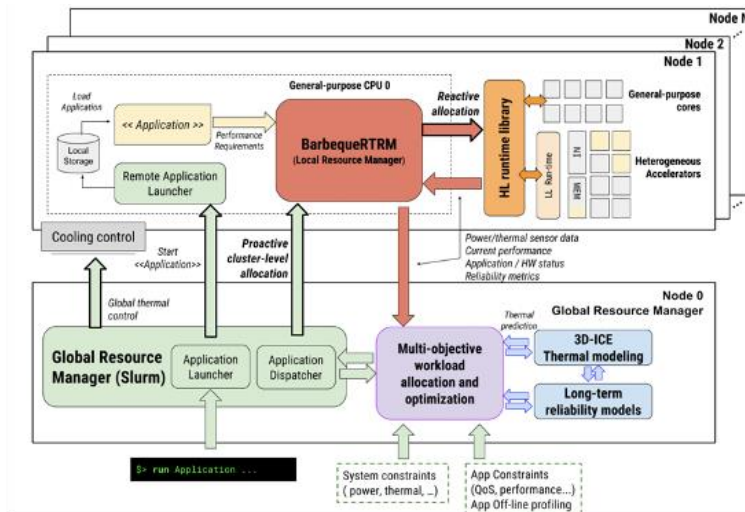


Fig.1.2.1.1 The RECIPE novel proactive/reactive hierarchical approach to resource management

The LRM is in charge for handling the computing resources assignment at node-level deploying a set of virtualized resources. It is made by means a *Runtime Resource Manager* (RTRM) covering at node level: *i*) fault tolerance; *ii*) resources usage. One instance of RTRM, (RECIPE project uses the open source tool: *BarbequeRTRM*), will run in each node to manage its heterogeneous resources. The resource manager will provide a “lightweight virtualization” mechanism allowing the applications to cope with the heterogeneity of the underlying hardware. The LRM has to be able to collect electrical and thermal data at level of node by means sensors provided by BMC (Baseboard Management Controller) for slow sampling rate (seconds) as well as other tools are able to collect electrical and thermal data for accelerator devices. The LRM framework provides Run-Time Library (RTLib) tool to properly manage the reconfigurability of the application. RTLib is integrated with the programming models, such as: OmpSS [Fernández,2014], to minimize the effort in terms of applications porting. It enables the synchronization of the application execution with the enforcement of the resource management actions. This managed execution makes also possible the construction of a runtime profile able to profile the application actual performance.

RTLib tool can be based on StarPU [Papadopoulos,2021], a software tool aiming to allow programmers to exploit the computing power of the available CPUs and GPUs, while relieving them from the need to specially adapt their programs to the target machine and processing units. At the core of StarPU is its runtime support library, which is responsible for scheduling application-provided tasks on heterogeneous CPU/GPU machines. In addition, StarPU comes with programming language support, in the form of an OpenCL front-end. StarPU's runtime and programming language extensions support a task-based programming model. Applications submit computational tasks, with CPU and/or GPU implementations, and StarPU schedules these tasks and associated data transfers on available CPUs and GPUs. The data that a task manipulates are automatically transferred among accelerators and the main memory, so that programmers are freed from the scheduling issues and technical details associated with these transfers. StarPU takes particular care of scheduling tasks efficiently, using well-known algorithms from the literature. In addition, it allows scheduling experts, such as compiler or computational library developers, to implement custom scheduling policies in a portable fashion.

The GRM is the RMS such as: SLURM. It takes care of infrastructure-level joint workload allocation and cooling control, by gathering feedback on the status of each node and by using the thermal/power/performance/reliability models. It gathers global information of the infrastructure tacking into account a complex model-based run-time multi-objective optimization, involving performance, power, reliability and temperature in high-level IT workload management (allocation and scheduling of workload to a specific node); as well as physical resource management of processors, memories, storage, network and cooling management.

## WISHED STATE

HPC is moving towards enabling access to its appliances to SME through a cloud-edge continuum approach. Initiatives for edge HPC enabling capability are being deployed more and more but on European Digital Ecosystem there is still a lack of united efforts for lowering the entry barriers for users [Koller,2015].

**WS-1.2.1a:** Cloude-edge HPC capability in exascale heterogeneous architectures requires to asses the virtualization environments for accelerator device such as GPUs and FPGAs. The usage of the GPU/FPGA for computation acceleration has made significant inroads into multiple application domains due to their ability to achieve high throughput and predictable latency, while providing programmability, low power consumption and time-to-value. Many types of workloads, e.g. databases, big data analytics, and high performance computing, can be and have been accelerated by GPUs/FPGAs. As more and more workloads are being deployed in the cloud, it is appropriate to consider how to make GPUSs/FPGAs and their capabilities available in the cloud. However, such integration is nontrivial, in particular for FPGA, due to issues related of resource abstraction and sharing, compatibility with applications and accelerator logics, and security. Therefore a PoC of a general framework for integrating FPGAs into a virtualized environment that enable the main items described above, in particular the isolation between multiple processes in multiple virtual machines, precise quantitative resources allocation and priority-based workload scheduling.

**WS-1.2.1b:** Today, a per-node power profile can be obtained by the node BMC via IPMI interface. However, this mechanism is characterized by a slow sampling rate (seconds), no time-stamping, and does not allow an accurate energy accounting. To overcome these problems, [Hackenberg,2014] proposes HDEEM, which allows power sampling up to 1 KS/s (kilo Samples per second) and accurate energy accounting, thanks to an extension of the BMC data monitoring features and a dedicated FPGA placed on each computing node. However, due to the use of the BMC as embedded monitoring system, their solution is not open and flexible to implementing new algorithms for on-board data processing, suffers from closed design, and it is limited in memory storage. However, instantaneous readings are possible only at 1 S/s and it will be wished to implement solutions that provides higher sampling rate.

**WS-1.2.1c:** As a GRM, SLURM requires having complete visibility upon the cluster, and in particular: mapping and resource usage for each application deployed in each cluster, utilization and load, and power and temperature of the available resources on the node and its accelerators. Data needs to be collected and used in (almost) real time, so that the

allocator can efficiently allocate the workload, avoiding unnecessary queues at the LRM level. Therefore it is wishful to have a plugin that collect the LRM power/thermal data at node level and insert them into SLURM database. The SLURM consumable resources plug-in already provides a template and instructions on how to create new resources. The most important point of this step is that the resources integrated in this plug-in should be synchronized with the information provided by an external data sources, such as LRM and, thus, their features standardized. Furthermore, these developments will enable to use in a fully coordinated way all the heterogeneous resources of the cluster.

**WS-1.2.1d:** A RMS should be managed automatically by the runtime, ideally without user/programmer intervention or with, at most, hints introduced in the programming model to help drive the runtime decisions. Once the program has been developed on top of a runtime system, the changes in the configurations should be transparently managed by the runtime. It means to identify key components/information that guide the programmer in the decisions about how to exploit important benchmarks in all the systems that the framework works on and - find a way to implement these strategies into the runtime model.

**ACTION STATE**

The actions need to achieve an assessment of the foregoing wished states during the activities of WP1 could be those in the following table:

	Analysis	Design	Develop	Implementation	Test
WS-1.2.1a	Virtualize solutions for GPUs/FPGAs	virtualize environment based on OpenStack	virtualize GPU/FPGA devices	PoC on GPU/FPGA	Performance -Throughput -Latency
WS-1.2.1d	LRM	LRM with power/thermal data acquisition	Monitoring	PoC on one IDV	
WS-1.2.1c	GRM/SLURM	A plugin for SLURM to store external data	Integrate LRM with SLURM	PoC on one IDV	
WS1.2.1			Integrate OmpSs/StarPU in SLURM	PoC on GPU/FPGA	
WS-1.2.1b					

**PRIORITY**

These actions should be performed in an iterative/incremental process and the implementation and test actions must be carried out on the FPGA LAB of ENEA and a GPU cluster make available by partners.

**REFERENCES**

[Fornaciari,2018]: Fornaciari, W.; Agosta, G.; Atienza, D.; Brandolese, C.; Cammoun, L.; Cremona, L.; Cilardo, A.; Farrés, A.; Flich Cardo, José; Hernández, C.; Kulchewski, M.; Libutti, S.; Martínez, J.; Massari, G.; Oleksiak, A.; Pupykina, A.; Reghenzani, F.; Tornero, R.; Zanella, M.; Zapater, M.; Zoni, D. *Reliable power and time-constraints-aware predictive management of heterogeneous exascale*

- systems. International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation. p. 187-194. (2018) DOI: 10.1145/3229631.3239368.
- [Isereto,2014]: S. Isereto et al., *SLURM Support for Remote GPU Virtualization: Implementation and Performance Study*, 2014 IEEE 26th International Symposium on Computer Architecture and High Performance Computing, 2014, pp. 318-325, doi: 10.1109/SBAC-PAD.2014.49.
- [Reano,2013]: C. Reaño, R. Mayo, E. S. Quintana-Ortí, F. Silla, J. Duato and A. J. Peña, "Influence of InfiniBand FDR on the performance of remote GPU virtualization," 2013 IEEE International Conference on Cluster Computing (CLUSTER), 2013, pp. 1-8, doi: 10.1109/CLUSTER.2013.6702662.
- [Vaishnav,2018]: A. Vaishnav, K. D. Pham and D. Koch, "A Survey on FPGA Virtualization," 2018 28th International Conference on Field Programmable Logic and Applications (FPL), 2018, pp. 131-1317, doi: 10.1109/FPL.2018.00031.
- [Fernández,2014]: Fernández A., Beltran V., Martorell X., Badia R.M., Ayguadé E., Labarta J. (2014) Task-Based Programming with OmpSs and Its Application. In: Lopes L. et al. (eds) Euro-Par 2014: Parallel Processing Workshops. Euro-Par 2014. Lecture Notes in Computer Science, vol 8806. Springer, Cham. [https://doi.org/10.1007/978-3-319-14313-2\\_51](https://doi.org/10.1007/978-3-319-14313-2_51)
- [Papadopoulos,2021]: Lazaros Papadopoulos, Dimitrios Soudris, Christoph Kessler, August Ernstsson, Johan Ahlqvist, Nikos Vasilas, Athanasios I Papadopoulos, Panos Seferlis, Charles Prouveur, Matthieu Haefele, Samuel Thibault, Athanasios Salamanis, Theodoros Ioakimidis, and Dionysios Kehagias. EXA2PRO: A Framework for High Development Productivity on Heterogeneous Computing Systems. IEEE Transactions on Parallel and Distributed Systems, August 2021
- [Koller,2015]: B. Koller, N. Struckmann, J. Buchholz, and M. Gienger, *Towards an environment to deliver high performance computing to small and medium enterprises*. in Sustained Simulation Performance 2015. Cham: Springer International Publishing, 2015, pp. 41–50.
- [Hackenberg,2014]: D. Hackenberg, T. Ilsche, J. Schuchart, R. Schöne, W. E. Nagel, M. Simon, and Y. Georgiou. *Hdeem: High definition energy efficiency monitoring*. In Energy Efficient Supercomputing Workshop (E2SC), 2014, pages 1–10, Nov 2014.

## 9 Task-1.3: Programming Models

The HPC community provided several programming models that 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. This is the main motivation of Task 1.3, which evaluates the gap to adapt programming models to the project architecture, which includes FPGAs, and describes how those models can be integrated to allow the application layer to benefit from their complementarity.

### 10 Task-1.3.1: Streaming Model

#### CURRENT STATE

Streaming applications are usually described as DATA-flow Graphs (DAGs), where each node implements some kind of operator/function/filter over the input data to produce some kind of



corresponding output data. In the recent past, several environments have been developed and adopted to process both structured and non-structured data according to the streaming programming model, mostly targeting big data and distributed architectures. Apache Storm [Storm,2021], Flink [Flink,2021], Spark Streaming [Spark,2021], just to name a few frameworks, all provide the possibility to develop stream processing applications sorting different expressive power (programmability) and performance. In general, these programming frameworks are quite efficient, but *i)* they do not provide the necessary latencies usually required within HPC computations, and *ii)* they do not provide adequate support to different kinds of accelerators. Indeed, several programming frameworks have been designed with the intent to provide better latencies besides good throughput. In some cases, the typical *continuous streaming model* is replaced by a *discretized streaming model* accumulating and processing the input data in *batches*, such that better performance can be provided mainly in terms of throughput. Recent models, such as the one provided by WindFlow [Mencagli,2021], succeed in keeping the advantages of both approaches in the simultaneous delivery of good throughput *and* limited/low latencies. Exploring different kinds of accelerators still represents an active research field, although GP-GPU exploitation has already been taken into account. At the moment being, several high level “streaming” programming models are available, providing different ways to build the operator network processing the stream of input data. As far HPC world is concerned, there is no clear evidence of which one can be adopted and, even more important, no evidence of which solutions may be used to deliver the performance typically required in this field (both in terms of throughput and in terms of latency).

## WISHED STATE

In the near future, we would like to have available stream parallel programming frameworks supporting the possibility to seamlessly exploit CPU and reconfigurable hardware accelerators in the execution of different kind of stream parallel patterns/computations. In particular to limit the amount of hardware specific knowledge required to the application programmers to achieve efficient exploitation of reconfigurable hardware accelerators. In turn, this means the compiler and/or run time support of the new stream parallel frameworks should: *i)* support development of accelerator kernels using common, existing programming languages; *ii)* provide seamless ways to execute the kernels on the accelerators (e.g. simple library/object method call); *iii)* provide seamless ways to compose accelerated kernels within other parallel patterns running on CPU cores (e.g. running an accelerated kernel as a pipeline stage or as a map worker).

**WS-1.3.1:** the deployment of a programming framework leveraging on existing frameworks efficiently supporting the streaming model on modern shared-memory multicores (FastFlow [Aldinucci,2017]) and on the possibility to offload particular streaming operators and/or entire DAGs relative to a streaming application to accelerators leveraging state-of-the-art FPGAs and the related high level synthesis tools. In this way, the application developers shall use a set of tools taking care (mostly in automatic) of all those tasks that usually are in charge of the application programmer and only related to the offloading of pre-compiled kernels to FPGA accelerators, with proper data transfer to and from accelerator memory and with the scheduling of suitable tasks on the accelerator. Ideally, the application developers shall be able to write just the high-level code modelling the operators to be deployed for acceleration, the code modelling the overall DAG of the streaming applications and the non-

accelerated operators. Then the programming tools developed shall take care of compiling and deploying the accelerated operators as proper FPGA configurations, and to provide the hooks necessary for the operators running on CPU to schedule tasks to the accelerator and process the corresponding results.

**ACTION STATE**

We have currently analyzed the problems and peculiarities of the process offloading tasks for processing by FPGA implemented operators in a streaming application where the non-accelerated operators are implemented in FastFlow. We individuated the steps necessary to generate the FPGA accelerated operators through standard FPGA toolchain(s) and those necessary to feed data to FPGA and to retrieve data from FPGA via OpenCL interfaces properly wrapped into FastFlow operator nodes deployed and run onto CPU cores. The next steps go through experiments with code developed ad hoc to expose problems and to experiment solutions, and then continue with the engineering of:

- a. the FPGA operator interface generation/operation in FastFlow
- b. the methodology to be used to compile high-level code to FPGA (configuration) code suitable to be operated via its OpenCL interface.

	Develop	Implement	Test
WS-1.3.1	FPGA operator interface generation/operation in FastFlow	Compiling high-level code to FPGA (configuration) code suitable to be operated via its OpenCL interface, according to the defined methodology	Experimental ad hoc code to expose problems and solutions

**PRIORITY**

The initial priority is in the achievement of some working, interoperable streaming operator FastFlow application/nodes (components) properly interacting with FPGA counterpart to implement accelerated streaming operators and/or portions of DAG with operator nodes. After, this priority will be moved to the engineering of the process and to the development of the proper tools/components.

**REFERENCES**

[Storm,2021] Apache Storm web page: <https://storm.apache.org/>  
 [Flink,2021] Apache Flink web page: <https://flink.apache.org/>  
 [Spark,2021] Spark Streaming programming guide: <https://spark.apache.org/docs/latest/streaming-programming-guide.html>  
 [Mencagli,2021] G. Mencagli, M. Torquati, A. Cardaci, A. Fais, L. Rinaldi, and M. Danelutto. *WindFlow: High-Speed Continuous Stream Processing with Parallel Building Blocks*, IEEE Transactions on Parallel and Distributed Systems, 2021, IEEE. ISSN: 1045-9219, DOI: 10.1109/TPDS.2017.2679197  
 [Aldinucci,2017] M. Aldinucci, M. Danelutto, P. Kilpatrick, M. Torquati “FastFlow: High-Level and

## 11 Task-1.3.2: Tasks Model

### CURRENT STATE

The task-based programming model consists in describing an algorithm as a set of tasks that are connected with dependencies, which can be represented by a Directed Acyclic Graph (DAG) in most cases. There are now several task-based runtime systems [*Duran,2011*], [*Augonnet,2011*], each with its specificities, features, and interface. These tools have been used to develop various HPC applications [*Carratalá-Sáez,2020*], [*Sukkari,2017*], [*Agullo,2016*], [*Agullo-Bramas,2016*], that demonstrated the potential of the approach on distributed heterogeneous computing nodes. However, FPGAs have been less studied than GPUs, because they rely on the stream concept and did not equip many HPC clusters.

Supporting efficiently new processing unit type impacts the task-based runtime systems on several points. There is first a technical effort to combine the different technologies to have a framework that functions and manages the data movement. In the front end, it is necessary to adapt the interface to the users to expose the specificities of the Processing Unit (PU). Finally, as some tasks can be computed on different processing units, it is the internal scheduler which decides how they are distributed. Therefore, the scheduling strategies should be evaluated and potentially improved to deal with the specific aspects of the new PU.

Even with all the recent contributions there are several shortcomings in Task Model runtimes that need to be addressed. To enumerate only some of them, no programming model (at all) is able to work with PCIe attached FPGAs using a host that is not an x86 CPU. However, new architectures (e.g. ARM or RISC-V) are envisioned to be the ones that drive the computation (like in IDV platforms) and models (and drivers) should be adapted to the new systems. In addition, FPGA support can be improved by taking advantage of the new characteristics of current FPGA platforms (like multiple Super Logic Regions or Memory Channels or even High Bandwidth Memory) and to improve power consumption control. Finally, it is a research subject to determine if task managers/schedulers should be the central control unit of manycore [*Morais,2019*] or even clusters of computing nodes [*DeHaro,2022*].

### WISHED STATE

Task-based runtime systems should support FPGAs by managing the data transfer and the execution of the tasks on the devices. The programming model should be improved to exploit the specificities of the specialized hardware such as accelerators or coprocessors that offer an interesting approach to overcome the limits encountered by conventional processor architectures. Many HPC facilities are now equipped with one or several accelerators (e.g. a GPU), in addition to the conventional processor(s). While a lot of efforts have been devoted to offload computation onto such

accelerators, very little attention has been paid to portability concerns on the one hand, and to the possibility of having heterogeneous accelerators and processors to interact on the other hand.

**WS-1.3.2a:** *OmpSs-2 [OmpSs,2021]* is a programming model composed of a set of directives and library routines that can be used in conjunction with a high-level programming language in order to develop concurrent applications. *OmpSs@FPGA [OmpSs@FPGA,2021]* is an extension of *OmpSs* programming model to support easily offloading tasks to FPGA devices. It uses the task and target directives to define the portions of application code that will become accelerators in the FPGA. The supported languages are C and C++. We propose to adapt *OmpSs@FPGA* to IDV-E platform and improve the support to obtain performance comparable or better than other state-of-the-art approaches that are less productive and/or portable (like low-level HDL programming). Also we want to research the impact of a hardware scheduler in performance of manycore systems/ clusters of IDV-E like platforms. This research requires the design of an Open Source Fast Task Scheduler IP to be integrated with the software runtime.

**WS-1.3.2b:** *StarPU [StarPU,2021]* is a runtime system that offers support for heterogeneous multicore architectures, it not only offers a unified view of the computational resources (i.e. CPUs and accelerators at the same time), but it also takes care of efficiently mapping and executing tasks onto a heterogeneous machine while transparently handling low-level issues such as data transfers in a portable fashion. At the core of *StarPU* is its runtime support library, which is responsible for scheduling application-provided tasks on heterogeneous CPU/GPU machines. In addition, *StarPU* comes with programming language support, in the form of an OpenCL front-end (SOCL OpenCL Extensions). *StarPU* task programming library, that initially targets heterogeneous architectures, like GPUs, already support FPGA [Christodoulis,2018] and it will support Xilinx and/or Intel using Vitis and Intel OneAPI toolchains.

**ACTION STATE**

The actions are as follow:

	Develop	Implement	Test
WS-1.3.2a	OmpSs@FPGA on FPGA(Xilinx Alveo)	PoC using VITIS	Performance - Time to solution - Power consumption - Programmability
WS-1.3.2b	StarPU on FPGA(Xilinx Alveo)	PoC using VITIS	Performance - Time to solution

**PRIORITY**

The first step will consist in supporting the target hardware based on FPGA Xilinx Alveo. Second, the improvement of the runtime system will be performed in an iterative/incremental process.

## REFERENCE

- [Duran,2011]: Duran, A., Ayguadé, E., Badia, R. M., Labarta, J., Martinell, L., Martorell, X., & Planas, J. (2011). Ompss: a proposal for programming heterogeneous multi-core architectures. *Parallel processing letters*, 21(02), 173-193.
- [Augonnet,2011]: Augonnet, C., Thibault, S., Namyst, R., & Wacrenier, P. A. (2011). StarPU: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2), 187-198.
- [Carratalá-Sáez,2020]: Carratalá-Sáez, R., Faverge, M., Pichon, G., Sylvand, G., & Quintana-Ortí, E. S. (2020, May). Tiled Algorithms for Efficient Task-Parallel  $\mathcal{S}$ -Matrix Solvers. In *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (pp. 757-766). IEEE.
- [Sukkari,2017]: Sukkari, D., Ltaief, H., Faverge, M., & Keyes, D. (2017). Asynchronous task-based polar decomposition on single node manycore architectures. *IEEE Transactions on Parallel and Distributed Systems*, 29(2), 312-323.
- [Agullo,2016]: Agullo, E., Giraud, L., & Nakov, S. (2016, August). Task-based sparse hybrid linear solver for distributed memory heterogeneous architectures. In *European Conference on Parallel Processing* (pp. 83-95). Springer, Cham.
- [Agullo-Bramas,2016]: Agullo, E., Bramas, B., Coulaud, O., Darve, E., Messner, M., & Takahashi, T. (2016). Task-based FMM for heterogeneous architectures. *Concurrency and Computation: Practice and Experience*, 28(9), 2608-2629.
- [Morais,2019]: Lucas Morais, Vitor Silva, Alfredo Goldman, Carlos Álvarez, Jaume Bosch, Michael Frank, Guido Araujo. Adding Tightly-Integrated Task Scheduling Acceleration to a RISC-V Multi-core Processor. *MICRO 2019*: 861-872
- [OmpSs,2021]: <https://pm.bsc.es/ompss-2>
- [OmpSs@FPGA,2021]: <https://pm.bsc.es/ompss-at-fpga>
- [StarPU,2021]: <https://files.inria.fr/starpu/doc/html/>
- [Christodoulis,2018]: G. Christodoulis, F. Broquedis, O. Muller, M. Selva and F. Desprez, "An FPGA target for the StarPU heterogeneous runtime system," 2018 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2018, pp. 1-8, doi: 10.1109/ReCoSoC.2018.8449373.
- [DeHaro,2022]: Juan Miguel De Haro Ruiz, Rubén Cano, Carlos Álvarez, Daniel Jiménez-González, Xavier Martorell, Eduard Ayguadé, Jesús Labarta, François Abel, Burkhard Ringlein, Beat Weiss. OmpSs@cloudFPGA: An FPGA Task-Based Programming Model with Message Passing. *IPDPS 2022*: 828-838

## 12 Task-1.3.3: High Level Synthesis Flow

### CURRENT STATE

For many years High-Level Synthesis (HLS) has been promising a smooth path toward working HW starting from a pure SW specification, without ever giving a satisfactory solution. In the recent period, this promise seems to be finally fulfilled thanks to the availability of many HLS flows (see [Nane,2016] for a list of HLS tools) and to the widespread diffusion of commercial tools like Vitis

[Vitis,2021] and OneAPI [Intel,2021], the former developed by Xilinx and the latter by Intel, both encompassing an HLS engine. Both these flows allow synthesizing working HW starting from specifications given as C/C++ programs, communicating with a host program through openCL APIs; the interfaces with the external environment (memory banks, PCIe, transceivers) are embedded in a pre-designed shell which communicates through the AXI4 protocol with the section containing the programmable logic, customized by the user. The instantiation of the shell, as well as the connections among shell ports and the user kernels, specified as C/C++ programs, are carried out by the HLS flow. The correctness of the whole solution, i.e. both the SW running on the host and the SW used to configure the FPGA through the HLS engine, is checked at the different levels of abstraction: it is possible to check for the correctness of the solution

- through the execution of the SW emulation of the application, which executes the C/C++ code used to program the host and the FPGA; at this level, there is only pure C/C++ code, executed in a multiprocess/multithreaded environment (functional correctness);
- through the execution of the HW emulation, where the C/C++ code running on the host interacts with the HW simulation of the HDL code produced by the HLS engine; at this level, the HW simulation gives insights about the achieved degree of parallelization and about the speed performance of the HW; the infos are bit-true and cycle-accurate for the parts not interacting with the external world, while the modeling of the interaction with the PCIe and the external memory banks is not cycle-true;
- through the execution of the C/C++ code running on the host which interacts with the FPGA which has been programmed with the configurable part of the bitstream (the shell is preloaded on the FPGA) produced by the proprietary compilation flow used to translate the HDL into an FPGA configuration file.

The possibility to check for the correctness at the functional level, being the functional behavior preserved by the successive transformations which are correct-by-construction, greatly simplifies the development cycle and makes transparent to the user the low-level details which are managed automatically by the flow.

The availability of HLS flows like Vitis on the cloud (Nimbix, AWS, Alibaba, ...), together with the possibility to execute the produced bitstreams on FPGA accelerator cards available on the cloud at prices of few \$/hour, lowers another barrier to the adoption and development of FPGA accelerators, as it is not anymore required the acquisition, the installation and the maintenance on-premise of the HW/SW infrastructure required to run accelerated applications. In order to further reduce the expertise on FPGA required, Xilinx has recently made available many libraries (Artificial Intelligence, Linear Algebra, Vision, DSP, ...) [Xilinx,2021] that can be used as building blocks to set up complex and powerful applications.

## WISHED STATE

In spite of the great advances characterizing the development flows for accelerators based on FPGAs and recalled in the previous section, there are still significant improvements that could (and should) be introduced to reduce the adoption barrier in heterogeneous HPC environments.

**WS-1.3.3a:** to develop a unique development environment which should allow the programming, within a given model of computation, of both the “standard” section of a heterogeneous HPC system and the FPGA-based accelerated section of the same system.

Today, apart from the OneAPI approach which has to be fully verified in its generality and portability of performance - there is no development flow that allows the contemporaneous and seamless programming of the parallel system as a whole: the accelerators have to be programmed in their own environment and the code has to be manually split between the conventional CPUs and the accelerators. The integration of the Vitis HLS flow with FastFlow (T4.1.1 - streaming models) and with OmpSS (T.4.2 - Task-based models) will be investigated and implemented during the lifetime of the project. These will allow the achievement of two programming environments (one for the streaming models and the other for the task-based ones) that seamlessly include the FPGA accelerators.

**WS-1.3.3b:** to implement a HW communication layer that allows seamless communication among kernels assigned to different FPGAs and the realization of a corresponding SW stack that should be integrated within the unified programming environment. It means splitting the part of the code allocated in the accelerated section among different FPGAs. This implementation will be addressed in the activities forecasted in T2.4 (IP for low-latency intra-node and inter-node communication links), where the HW infrastructure will be implemented, and in T4.4 (Inter-FPGA communication SW stack), where the SW part of the communication layer will be developed.

**WS-1.3.3c:** to develop a novel numerical representation, like posit or BFloat16, that are very well suited for domains like AI and should give important improvements both in terms of use of the HW resources and in the memory bandwidth requirements. It will be developed in T4.3 (Mixed precision technologies), where compiler extensions will be introduced to allow the adoption of new arithmetic representations and their seamless insertion in the code, having instructed the compiler about data ranges of variables and implementing policies to decide which is the best numeric representation

**ACTION STATE**

in order to achieve, at least in a prototype form, the wished states reported above, the following actions in the project are planning:

	Develop	Implement	Test
WS-1.3.3a	Unified Vitis based programming environment for: - FastFlow and OmpSS	Image Processing and lossless compression on IDV-E	Evaluate performance in terms of Energy Delay Product
WS-1.3.3b	Inter-FPGA HW/SW communication infrastructure	A HW communication layer that allows seamless communication among kernels assigned to different FPGAs and the realization of the corresponding SW stack that should be integrated within the unified programming environment.	Specific tests to measure network performance with HLS Kernels as communication endpoints: -Comm. Latency -Comm. Bandwidth
WS-1.3.3c	Multi-precision arithmetic		Performance: -Time to solution -Energy to solution

**PRIORITY**

The three wished states sketched above are listed in order of their priority: the lack of a common programming environment is a real barrier to the adoption of accelerators, so its achievement has

the highest priority. The implementation of a communication infrastructure for efficient inter-FPGA data movement could give a real benefit to many applications, allowing data to pass through a dedicated path not involving the communication network and the SW layers of the HPC system. As this improvement is transversal to many domains, it has a second highest priority. The proper management of multi-precision arithmetic is, at a first approximation, not a barrier to the adoption of the FPGA technology, because it does not extend the class of problems that can be managed, but could be classified as “nice to have” feature because, in certain classes of problems like AI, it could lead to performance gain in terms of speed, used resources, and power budget.

## REFERENCES

[Nane,2016]: R. Nane et Al. “A Survey and Evaluation of FPGA High-Level Synthesis Tools”. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 35(10), 2016.

[Vitis,2021]: [https://www.xilinx.com/html\\_docs/xilinx2021\\_1/vitis\\_doc/index.html](https://www.xilinx.com/html_docs/xilinx2021_1/vitis_doc/index.html)

[Intel,2021]: <https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fpga.html#gs.bgx9i6>

[Xilinx,2021] <https://www.xilinx.com/products/design-tools/vitis/vitis-libraries.html>

## 13 Task-1.3.4: Mixed-Precision tools and alternative arithmetic-type for IP accelerators

### CURRENT STATE

The effort in computing artificial intelligence algorithms is an open challenge in the field of computing platforms nowadays. When considering strict requirements, such as lowering the power consumption, maximizing the throughput, and minimizing the latency the computational complexity becomes more and more critical. IEEE-754 floating-point types (32 bit for single-precision and 64 bits for double-precision) have been the de facto standard for floating-point number systems for decades, but the drawbacks of this numerical representation leave much to be desired. Alternative representations are gaining traction, both in HPC and machine learning environments and many group in academia and industry are investigating alternatives.

For example, commercial accelerators used in HPC applications like the GPUs from NVIDIA, supports classic FP64 and FP32 data arithmetic plus alternative arithmetics: in the A100 GPU there is the support also for BF16 and FP16 (scaled versions on 16 bits), TF32, a custom format which uses the same 10-bit mantissa as FP16 to ensure accuracy while sporting the same range as FP32, thanks to using an 8-bit exponent, and INT8 and INT4 (i.e. integers at 8 and at 4 bits, respectively).

In <https://www.nvidia.com/en-us/data-center/a100/>, for HPC applications formats like TF32 are suggested to be used, since TF32 can target similar accuracy than FP32 but is optimized for speed for NVIDIA GPUs.

A mixed-precision approach similar to that proposed by NVIDIA is also adopted in the accelerators from the EPI SGA1: Bfloat16 and its conversion to integer (INT) is supported in <https://www.european-processor-initiative.eu/dissemination-material/data-movement-reduction-for-dnn-accelerators-enabling-dynamic-quantization-through-an-efpga/>; FP32 and FP64 are



supported in the STX accelerator, see <https://www.european-processor-initiative.eu/wp-content/uploads/2022/06/Matheus-Cavalcante-@-RISC-V-Week.pdf>

The Textarossa project, from its project proposal, aims at finding innovative solutions to computing problems, but that are complementary to the research carried out in EPI SGA1 so that the results of Textarossa may enrich the output of EPI SGA1 and be an input for further development in EPI SGA2.

Summarizing the above points, in Textarossa two main points will be addressed concerning mixed-precision and alternative arithmetics to FP for accelerator IP macrocells.

1. At SW level proposing a tool (called TAFFO), that enables precision tuning on C/C++ kernels and code fragments, using different types of arithmetics. It currently supports floating-point and fixed-point data types, but does not support heterogeneous computing.
2. At HW level proposing accelerator IP macrocells, that can be integrated with RISC-V cores, giving HW support to a new type of arithmetic type called Posit and translation from fixed and floating formats to Posit format and vice-versa.

As far as point 1 is concerned, programming mixed-precision accelerators is a complex and error-prone task that currently needs to be performed mostly manually. A recent survey of precision tuning frameworks [Cherubin,2020] highlighted the lack of industry-grade tools to automatically optimize the precision-latency-energy trade-off [Stanley-Marbell,2021]:. Particularly, there is a lack of support for High Level Synthesis toolchains, as well as for hybrid systems employing both software and hardware components.

TAFFO [Cherubin-Cattaneo,2020] is a set of plugins for the LLVM compiler framework that enables precision tuning on C/C++ kernels and code fragments, driven by programmer hints in the form of pragmas or attributes. It includes key components of value-range analysis, data type manipulation, and feedback estimation, needed to perform effectively the precision tuning at the intermediate representation level. It currently supports floating and fixed point types. At the beginning of the project, no support is available for parallel and/or heterogeneous platforms. To enable the TEXTAROSSA applications to take advantage of mixed precision while leveraging the capabilities of the TEXTAROSSA IDVs, we need to extend TAFFO to provide support for both parallel and heterogeneous programming. In particular, the majority of TEXTAROSSA applications employ CUDA to accelerate kernels on heterogeneous machines, although some of them could leverage HLS to accelerate kernels on FPGA. These features will be developed in WP4. We will report the intermediate state of the development in D4.1.

As far as point 2 is concerned, among the proposed alternatives to the IEEE 754 standard regarding floating point representation of real numbers, the recently introduced Posit [Cococcioni,2021] format has been theoretically proven to be really promising in satisfying the mentioned requirements. However, with the absence of proper hardware support for this novel type, this evaluation can be conducted only through a software emulation. Indeed, hardware support for posit numbers has been growing recently, with the development of integrated posit units inside RISC-V cores [Sharma,2021], [Tiwari,2019]. To this aim, in Textarossa will be developed Posit Processing Unit IP macrocells, that will be integrated with RISC-V core. The relevant specifications are included in the deliverable D2.1 of WP2 .

**WISHED STATE**

The possibility to have a representation of the numbers with a variable precision need of furthers development efforts and there is no unique approach for all application problems. In general a mixed precision arithmetic may use processor architectures more general purpose but the development of programming tools is needed.

**WS-1.3.4a:** to develop means to accelerate posit computations in a way that does not completely alter an existing processor architecture. Therefore, we aim at building an accelerator that can be either a co-processor (an external device connected to thye main processor via a bus), or an integrated unit in a processor architecture (such as the Floating Point Unit (FPU) or the Arithmetic Logic Unit (ALU)). The final aim of this approach is to achieve mixed precision training of Neural Networks using 16-bit and 8-bit posits for, respectively, gradient computations and weights. This requirement will be translated in the specifications contained in Section 2 of the D2.1.

**WS-1.3.4b:** to extend TAFFO to support a wider range of data types as well as to integrate TAFFO with the TEXTAROSSA HLS toolchain in order to support a range of solutions including custom hardware. Furthermore, in order to support HPC codes, there is a need to provide a smooth integration with parallel programming models, particularly for intra-node parallelism (targeting CPU and GPU nodes, depending on the needs of the applications).

**ACTION STATE**

the following actions in the project are planning:

	Develop	Implement	Test
WS-1.3.4a	Posit accelerator IPs (see specification in Section 2 od D2.1)	DNN computation with posit on FPGA	Performances on mini-apps: -Same accuracy of FP - Time to solution - Energy to solution
WS-1.3.4b	TAFFO extensions to support heterogeneous HPC nodes	TAFFO support for HLS TAFFO support for heterogeneous parallel accelerators (OpenMP, OpenCL, CUDA)	Performance on benchmarks and mini-apps : error below 3% w.r.t. floating point, performance improvement over baseline

**REFERENCES**

[Cococcioni,2021]: M. Cococcioni, F. Rossi, E. Ruffaldi, S. Saponara and B. Dupont de Dinechin, *Novel Arithmetics in Deep Neural Networks Signal Processing for Autonomous Driving: Challenges and Opportunities*. in IEEE Signal Processing Magazine, vol. 38, no. 1, pp. 97-110, Jan. 2021, doi: 10.1109/MSP.2020.2988436.

[Sharma,2021]: N.Sharma, R.Jain, M.Mohan, S.Patkar, R.Leupers, N. Rishiyur, F. Merchant. *CLARINET: A RISC-V Based Framework for Posit Arithmetic Empiricism*. 2021. <https://arxiv.org/abs/2006.00364>

[Tiwari,2019]: S. Tiwari, N. Gala, C. Rebeiro, V. Kamakoti. *PERI: A Posit Enabled RISC-V Core*. 2019. <https://arxiv.org/abs/1908.01466>

[Cococcioni,2021]: M. Cococcioni, F. Rossi, E. Ruffaldi and S. Saponara, *A Lightweight Posit Processing Unit for RISC-V Processors in Deep Neural Network Applications*. in IEEE Transactions on Emerging Topics in Computing, doi: 10.1109/TETC.2021.3120538.

[Cherubin,2020]: S.Cherubin and G.Agosta. 2020. *Tools for Reduced Precision Computation: A Survey*. ACM Comput. Surv. 53, 2, Article 33 (March 2021), 35 pages. DOI:<https://doi.org/10.1145/3381039>

[Stanley-Marbell,2021]: P.Stanley-Marbell, A.Alaghi, M.Carbin, E.Darulova, L.Dolecek, A.Gerstlauer, G.Gillani, D.Jevdjic, T.Moreau, M.Cacciotti, A.Daglis, N.E.Jerger, B.Falsafi, S.Misailovic, A.Sampson, and D.Zufferey. 2020. *Exploiting Errors for Efficiency: A Survey from Circuits to Applications*. ACM Comput. Surv. 53, 3, Article 51 (May 2021), 39 pages. DOI: <https://doi.org/10.1145/3394898>

[Cherubin-Cattaneo,2020]: S.Cherubin, D.Cattaneo, M.Chiari, and G.Agosta. 2020. *Dynamic Precision Autotuning with TAFFO*. ACM Trans. Archit. Code Optim. 17, 2, Article 10 (June 2020), 26 pages. DOI: <https://doi.org/10.1145/3388785>

## 14 Task-1.3.5: Secure Services for HPC

### CURRENT STATE

The adoption of dedicated hardware for cryptography functions and services has become common practice on platforms requiring low-latencies, high-throughput, and energy efficiency. Particularly in the context of HPC, computational power and energy-efficiency are crucial aspects. Some of these needs have been already addressed in EPI SGA1, where a cryptography hardware IP has been developed under UNIFI lead in WP9 (Stream2 of EPI1). Such cryptography IP in EPI SGA1 includes a RISC-V core plus a secure DMA and accelerators for symmetric-key cryptography based on AES 128/256 with up to 9 ciphering modes of operation, public-key cryptography based on Elliptic Curve Cryptography (ECC) with curves up to 521-bit, SHA2/SHA3 as HASH functions up to 512-bit of digest length, and on-chip random number generation.

Details on Cryptographic IPs developed in EPI 1 are public and can be found at the following link:

<https://www.european-processor-initiative.eu/dissemination-material/crypto-tile-factsheet/>

For what concerns symmetric-key cryptography and HASH functions, the cryptography IP proposed in EPI SGA1 addresses the need of a dedicated infrastructure for security functions, with long-term cryptographic protection and satisfying the requirements of Post-Quantum security. In case of public-key algorithms, ECC has been declared to be unsafe against post-quantum computation and the NIST is running a standardization process since 2016 for new public-key algorithms resistant to post-quantum computing capabilities.

In order to be complementary to the security developments in EPI1 and to cover new security features of interest, the activities in WP2 of Textarossa about security have been specified in D2.1 to cover the following cutting-edge cryptography specifications:

Textarossa will develop an IP macrocell accelerating the computing kernel of Homomorphic Encryption (HE) algorithm, which in Cloud and HPC server applications are needed when, to ensure the data privacy of different users, the Server is considered untrusted. Indeed, in standard cryptographic system only the communication edge-cloud (i.e. client-server) is encrypted and the data of different users are stored in plaintext mode in the server. Instead, in HE the data on the

server are kept encrypted but HE ensures the property that operations data in the server on the encrypted data of the users give the same results (from here the term Homomorphic where “homo” means “the same”) as doing the operations on the decrypted data.

Since WP2 of Textarossa aims at developing IPs accelerating wide-spread adopted algorithms, then we will accelerate one of the first and well-known SW implementation of the HE approach, the SEAL library from Microsoft. The latter is supported also by INTEL on its AVX512 instruction-set and by the NVIDIA GPU AI framework called CLARA.

Moreover, Textarossa will develop an IP core implementing in HW the eXtendable Output Functions (XOF) SHAKE 128/256, which is a new hashing function vs. the SHA2 and SHA3 accelerators developed in EPI1.

To be noted that SHAKE is already adopted by algorithms like Crystals-Dilithium for digital signature using Lattice LWE (Learning With Errors) Codes that have been recently standardized for Post-quantum crypto applications.

These specifications will be further discussed in Section 3 of D2.1.

**WISHED STATE**

Within this project, CINI aims at extending the capabilities of the cryptography IP proposed in EPI SGA1 and improving its functionalities. The main improvements are related to:

**WS-1.3.5a:** Extending the support to compuing intensive kernels of Homomorphic encryption library like Microsoft SEAL; by implementing in FPGA technology hardware IPs for the computation of Number Theoretic Transform (NTT).

**WS-1.3.5b:** Extending the support to Post-Quantum algorithms that have been recently standardized by NIST, focusing on hardware IPs in FPGA technology for SHAKE128/256 eXtendable Output Function (XOF) that will replace hashing functions like SHA2 and SHA3.

**ACTION STATE**

To achieve the improvements reported above, we planned the following actions:

	Analysis	Implement	Test
WS-1.3.5a	Number Theoretic Transform (NTT) computational kernel of SEAL homomorphic encryption library	Hardware IP for NTT in FPGA technology	Performances: - Computational complexity -Throughput/Latency
WS-1.3.5b	Post-Quantumr Digital Signature	Hardware IP for SHAKE128/256 eXtendable Output Function (XOF) in FPGA	Performances: - Computational complexity -Throughput/Latency

**PRIORITY**

The two improvements listed above are independent of each other and have the same priority.

## REFERENCES

[NIST Announces First Four Quantum-Resistant Cryptographic Algorithms | NIST](https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms)  
<https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms>

## 15 Task-1.4: System Architecture

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 high performance of the computation nodes.

This task addresses the gap analysis in the system architecture for HPC and identifies pathways for the development of heterogeneous architectures and interconnection networks that can fill the current gaps and pave the way towards exascale computing.

## 16 Task-1.4.1: Heterogeneous Architectures

### CURRENT STATE

Clusters of hybrid nodes hosting accelerators and multi-core processors connected by low-latency-high bandwidth networks are the main platform for High-Performance Computing nowadays. Several interesting research initiatives aim at providing alternative platforms based on completely different computing paradigms, including quantum computing. However, none of them has many chances to replace the aforementioned platform in, say, the next 5 years, at least for a wide set of applications. The main issue that hinders the scalability of current (and near-future) systems remains the cost of data movement that exceeds, by far, the cost of floating-point arithmetic. Commercial HPC solutions may be divided in two categories. The first category comprises systems using proprietary interconnection technologies that offer any-to-any connectivity with a high bandwidth (but also a non-negligible latency). These systems support a quite limited number of computing nodes (e.g., the Nvidia DGX supports up to 16 GPU) and are designed with deep-learning applications in mind. In the second category, there are nodes connected with technologies like Infiniband that support a much higher number of nodes that is limited, in some sense, only by the initial cost and the power consumption. This is the most common solution adopted in supercomputing centers in Europe and around the world.

Surely the most important news for heterogeneous systems are in the field of the integrated hybrid CPU/GPU architectures. Current compute nodes designs place the x86 processor inline between system memory and GPUs. The CPU feeds system memory with external I/O while feeding data from memory to discrete GPUs deployed within a compute node. In a system with four A100 GPUs, the

bandwidth limitation focuses on the PCIe connections between the processor and each GPU. NVIDIA assumes PCIe 3.0 x16 (16GB) support from the processor, despite the A100 offering PCIe 4.0 (perhaps a dig at Intel Xeons). Intra-GPU connectivity through NVlink (NVIDIA’s proprietary GPU interface) provides up to 600GB/s throughput and 2039 GB/s of high-bandwidth memory (HBM).

In middle 2021, **NVIDIA** announced the deployment on early 2023 of an Arm-based CPU/SoC, named Grace. The design of Grace removes the PCIe bottleneck and instead uses NVlink (4th generation, 900GB/s) in a mesh configuration to connect each processor and GPU to both HBM and LPDDR5x system memory, which from the image presented (figure 1.4.1.1a), looks to be in a “system-in-a-package” design. This arrangement is similar to the architecture of the Apple M1 processor (although that currently uses LPDDR4x). Conceptually, the change in data flow is shown in figure 1.4.1.1b.

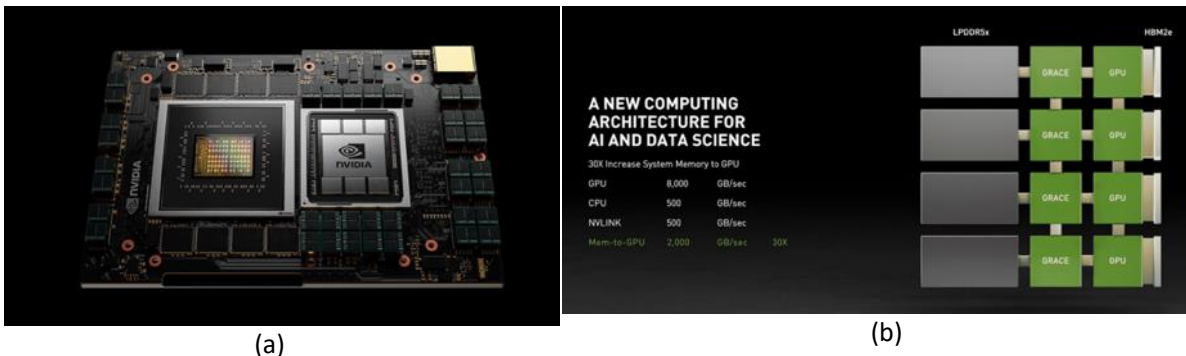


Fig.1.4.1.1: NVIDIA ARM GRACE and GPUs. (a) system-in-a-package design. (b) data flow exchange

The whole memory subsystem is cache-coherent, which simplifies programming. The use of Arm Neoverse cores and LPDDR5x memory creates a power-efficient system that NVIDIA claims will offer 10x better performance than today’s DGX-based systems running with x86 processors.

Grace will be based on a future iteration of Arm’s Neoverse cores, and it is internal product for NVIDIA, to be offered as part of their larger compute node offerings. NVIDIA isn’t directly gunning for the Intel Xeon or AMD EPYC server market, but instead they are building their own chip to complement their GPU offerings, creating a specialized chip that can directly connect to their GPUs and help handle enormous, trillion parameter AI models.

Grace is designed to fill the CPU-sized hole in NVIDIA’s AI server currently available (Table 1.4.1.1). GPUs are incredibly well-suited for certain classes of deep learning workloads, but not all workloads are purely GPU-bound, if only because a CPU is needed to keep the GPUs fed. NVIDIA’s current server offerings, in turn, typically rely on AMD’s EPYC processors, which are very fast for general compute purposes, but lack the kind of high-speed I/O and deep learning optimizations that NVIDIA is looking for. In particular, NVIDIA is currently bottlenecked by the use of PCI Express for CPU-GPU connectivity; their GPUs can talk quickly amongst themselves via NVLink, but not back to the host CPU or system RAM.

	GRACE	Xavier	Parker (Tegra X2)
--	-------	--------	----------------------

CPU Cores	?	8	2
CPU Architecture	Next-Gen Arm Neoverse (Arm v9?)	armel (Custom Arm v8.2)	Denver 2 (Custom Arm v8)
Memory Bandwidth	>500GB/sec LPDDR5X (ECC)	137GB/sec LPDDR4X	60GB/sec LPDDR4
GPU-to-CPU Interface	>900GB/sec NVLink 4	PCIe 3	PCIe 3
CPU-to-CPU Interface	>600GB/sec NVLink 4	N/A	N/A
Manufacturing Process	?	TSMC 12nm	TSMC 16nm
Release Year	2023	2018	2016

Tab.1.4.1.1; NVIDIA SoC comparison

The solution to the problem, as was the case even before Grace, is to use NVLink for CPU-GPU communications. Previously NVIDIA has worked with the OpenPOWER foundation to get NVLink into POWER9 for exactly this reason, however that relationship is seemingly on its way out, both as POWER’s popularity wanes and POWER10 is skipping NVLink. Instead, NVIDIA is going their own way by building an Arm server CPU with the necessary NVLink functionality.

The end result, according to NVIDIA, will be a high-performance and high-bandwidth CPU that is designed to work in tandem with a future generation of NVIDIA server GPUs. With NVIDIA talking about pairing each NVIDIA GPU with a Grace CPU on a single board – similar to today’s mezzanine cards – not only does CPU performance and system memory scale up with the number of GPUs, but in a roundabout way, Grace will serve as a co-processor of sorts to NVIDIA’s GPUs. This, if nothing else, is a very NVIDIA solution to the problem, not only improving their performance, but giving them a counter should the more traditionally integrated AMD or Intel try some sort of similar CPU+GPU fusion play.

By 2023 NVIDIA will be up to NVLink 4, which will offer at least 900 GB/sec of cumulative (up + down) bandwidth between the SoC and GPU, and over 600 GB/sec cumulative between Grace SoCs (Table 1.4.1.1.2.). Critically, this is greater than the memory bandwidth of the SoC, which means that NVIDIA’s GPUs will have a cache coherent link to the CPU that can access the system memory at full bandwidth, and also allowing the entire system to have a single shared memory address space. NVIDIA describes this as balancing the amount of bandwidth available in a system, and they’re not wrong, but there’s more to it. Having an on-package CPU is a major means towards increasing the amount of memory NVIDIA’s GPUs can effectively access and use, as memory capacity continues to be the primary constraining factors for large neural networks – you can only efficiently run a network as big as your local memory pool.

And this memory-focused strategy is reflected in the memory pool design of Grace, as well. Since NVIDIA is putting the CPU on a shared package with the GPU, they’re going to put the RAM down right next to it. Grace-equipped GPU modules will include a to-be-determined amount of LPDDR5x memory, with NVIDIA targeting at least 500GB/sec of memory bandwidth. Besides being what’s likely to be the highest-bandwidth non-graphics memory option in 2023, NVIDIA is touting the use of LPDDR5x as a gain for energy efficiency, owing to the technology’s mobile-focused roots and very short trace lengths. And, since this is a server part, Grace’s memory will be ECC-enabled, as well.

	GRACE	EPYC 2 + A100	EPYC 1 + V100
GPU-to-CPU Interface (Cumulative, Both Directions)	>900GB/sec NVLink 4	~64GB/sec PCIe 4 x16	~32GB/sec PCIe 3 x16

CPU-to-CPU Interface (Cumulative, Both Directions)	>600GB/sec NVLink 4	304GB/sec Infinity Fabric 2	152GB/sec Infinity Fabric
---	------------------------	--------------------------------	------------------------------

Tab.1-4-1.2; CPU & GPU Interconnect Bandwidth comparison

NVIDIA big vision goal for Grace is significantly cutting down the time required for the largest neural networking models. NVIDIA is gunning for 10x higher performance on 1 trillion parameter models, and their performance projections for a 64 module Grace+A100 system (with theoretical NVLink 4 support) would be to bring down training such a model from a month to three days. Or alternatively, being able to do real-time inference on a 500 billion parameter model on an 8 module system.

The family of custom Arm cores was never good enough, and never made it out of NVIDIA’s mobile SoCs. Grace, in contrast, is a much safer project for NVIDIA; they’re merely licensing Arm cores rather than building their own, and those cores will be in use by numerous other parties, as well. So NVIDIA’s risk is reduced to largely getting the I/O and memory plumbing right, as well as keeping the final design energy efficient.

If all goes according to plan, expect to see Grace in 2023. NVIDIA is already confirming that Grace modules will be available for use in HGX carrier boards, and by extension DGX and all the other systems that use those boards. So while we haven’t seen the full extent of NVIDIA’s Grace plans, it’s clear that they are planning to make it a core part of future server offerings.

In 2021 Intel disclosed more about its new HPC-focused GPU offering: Ponte Vecchio. It will be the heterogeneous architecture of the Aurora supercomputer at ANL [Fig. 1.4.1.2]. Aurora will be on 2 ExaFlops peak performance with compute nodes based on 2 Intel Xeon scalable “Sapphire Rapids” processors and 6 Xe-HPC arch-based PonteVecchio GPUs. The CPU-GPU interconnect is on PCIe Gen4 whilst GPU-GPU is an all-to-all Intel Xe link. The Intel Xe-HPC architecture include a new instruction set. Contains 8 vector and 8 matrix engines, alongside a large 512 KB L1 cache. Ponte Vecchio has already achieved 45 TFLOPs of single-precision compute performance in its current A0 silicon version. This data center accelerator is the first Xe-HPC-based processor featuring a multi-tile design, including Compute, Rambo, HBM, and EMIB tiles, a total of 47 tiles with 100 billion transistors. Xe-HPC Slice is the main building block, which combines 16 Xe-Cores. What might be interesting is the fact that Ponte Vecchio is equipped with Ray Tracing Units. Same as HPG, each Xe-Core is tied to a single RT unit. The purposes of those cores have been listed on the official slide as Ray Traversal, Triangle Intersection, Bounding Box Intersection. Being a server accelerator means that those cores are of course not for gaming. Ponte Vecchio will be available in 1 and 2-stack configurations. This means specs up to 8 cores, 128 Xe-Cores, and 128 Ray Tracing Units. The 2-stack configuration will have 8 memory controllers for HBM2e. Intel Ponte Vecchio GPU features 5 different process nodes, making it one of the most complex HPC accelerators on the market. This may have an impact on the supply of Ponte Vecchio GPUs, should any component see an expected shortage. Intel is comparing itself to the NVIDIA A100 accelerator by more than doubling its FP32 throughput at 45 TFLOPs. NVIDIA’s solution offers 19.5 TFLOPs.





Fig.1.4.1.2: Aurora: a 2 ExaFlops peak performance supercomputer with Intel Pontevecchio XE-HPC GPUs

In AMD house the CDNA architecture for GPUs is evolving to the next level of CDNA 2 achieving over 4x1 performance boost over the prior generation architecture, with 47.9 TFLOP/s peak double-precision vector FP64 throughput, to enable exascale levels of performance with unrivalled programmability in heterogeneous systems. It will be the GPU architecture of Frontier at ORNL (Fig.Xa) with +1.5 ExaFlops of peak computing power. The AMD CDNA 2 architecture is designed first and foremost for the most taxing scientific computing and machine learning applications and the fig.1.4.1.3 shows the performance of the new GPU MI200 vs the prior generation MI100.

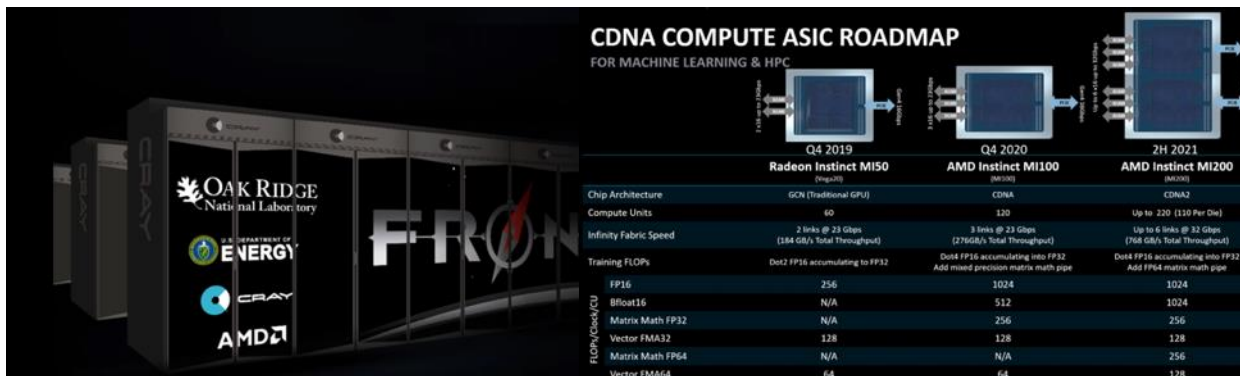


Fig.1.4.1.3: Frontier: a 1.5 ExaFlops peak power supercomputer at ORNL based on AMD CDNA 2 architecture

The AMD CDNA 2 architecture is designed first and foremost for the most taxing scientific computing and machine learning applications. It powers the new AMD Instinct™ MI200 generation of products that target solutions ranging from compact single systems all the way to the world’s largest exascale supercomputers with unique and highly differentiated programming models. The AMD Instinct MI200 Graphics Compute Die (GCD) with up to 110 Compute Units (CUs) per die, is built on advanced packaging technologies, enabling two GCDs to be integrated into a single package in the OAM (OCP Accelerator Module) form factor in the MI250 and MI250X products (Fig. 1.4.1.3).

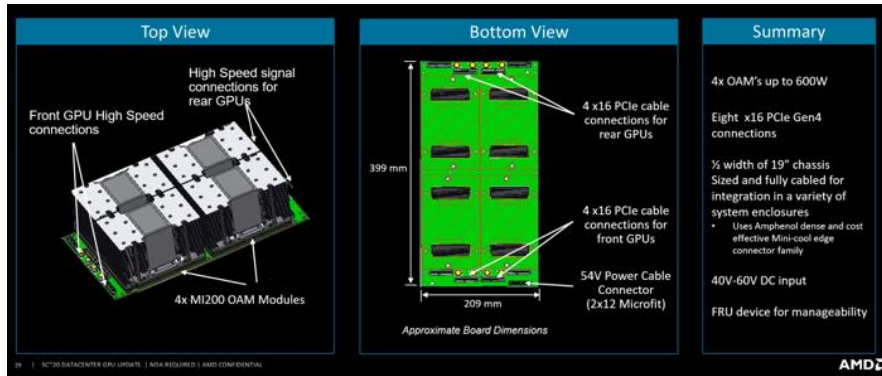


Fig.1.4.1.3: OAM Universal Baseboard for AMD Instinct MI200

Each AMD CDNA™ 2 GCD has multiple blocks dedicated to compute, memory access and communication and they are connected with a high-speed on-die fabric. One of the crucial innovations in the AMD CDNA 2 architecture is employing AMD’s unique Infinity Fabric to extend the on-die fabric across the package so that each GCD appears as a GPU in one shared memory system. Connecting two GCDs together in this fashion doubles the resources, creating a larger computational building block on top of the many other enhancements. One of the fundamental innovations in the prior generation AMD CDNA architecture was the introduction of the Matrix Core technology in the compute units (CUs) to boost computational throughput with a focus on datatypes used in machine learning. In the table 1.4.1.3 reports the Generational comparison of numerical formats and peak throughput between MI250X OAM and MI100 (PCIe).

Computation	MI100 (Flops/CLK/CU)	MI100 (Flops/CLK/CU)	MI100 (TFLOPS peak)	MI250X (TFLOPS peak)
MI200 Matrix FP64 Vs MI100 Vector FP64	64	256	11.5	95.7
MI200 Vector FP64 Vs MI100 Vector FP64	64	128	11.5	47.9
MI200 Matrix FP32 Vs MI100 Matrix FP32	256	256	46.1	95.7
MI200 Packed FP32 Vs MI100 Vector FP32	128	256	23.1	95.7
MI200 Vector FP32 Vs MI100 Vector FP32	128	128	23.1	47.9
MI200 Matrix FP16 Vs MI100 Matrix FP16	1024	1024	184.6	383
MI200 Matrix BF16 Vs MI100 Matrix BF16	512	1024	92.3	383
MI200 Matrix INT8 Vs MI100 Matrix INT8	1024	1024	184.6	383

Tab.1.4.1.3: MI250X (OAM) vs. MI100 (PCIe).

The Matrix Core technology in the AMD CDNA 2 architecture builds on this foundation and has been enhanced to support a wider range of datatypes and applications, with a particular emphasis on scientific computing with FP64 data. Additionally, similar to the prior generation, the CU array is partitioned into four shader engines that execute the compute kernels that are spawned by the command processor. The extremely high bandwidth register files and local data storage are tailored to support computational throughput. Additionally, the shared memory hierarchy outside of the CUs is critical to delivering the bandwidth necessary for real-world applications which work on large scale datasets that reside in memory. The AMD CDNA 2 memory controllers focus on efficiently accessing large working sets of data and bringing it on-die so that the

L2 cache can provide amplified the bandwidth to feed the CUs. The AMD CDNA 2 architecture boosts many different dimensions of the memory hierarchy, simultaneously improving bandwidth generationally and capacity while enhancing synchronization.

Each GCD contains an L2 cache that is physically partitioned with one slice per memory controller and shared by all the resources on a single GCD. The AMD CDNA 2 family uses a 16-way set-associative design with 32 slices with a total capacity of 8MB (per GCD). To keep pace with the computational capabilities of the CUs, the bandwidth from each L2 slice has been doubled to 128B per clock – a peak of 6.96 TB/s for the MI250, more than 2x the prior generation<sup>4</sup>. The queuing and arbitration for the distributed L2 cache have been enhanced to improve utilization of this read bandwidth over a wide range of workloads. The AMD CDNA 2 memory capabilities have been scaled up in tandem with the computational requirements to handle large exascale-class computing problems. The memory capacity has doubled to 64GB per GCD from 32GB in the prior generation, so that a multichip MI200 series accelerator can access up to 128GB of data (which is 64GB per GCD x 2 per AMD Instinct MI250/250X device) – 4x the total memory capacity than the prior generation<sup>5</sup> and comparable to the main memory of an entire server from a decade ago. The HBM2e memory interface operates at aggregate 3.2TB/s which is 2.7x the previous generation<sup>5</sup> of peak theoretical memory bandwidth taking into account the dual GCDs. To keep pace with this increase in off-chip bandwidth, the connection between the individual memory controllers and L2 cache slices has doubled to 64-bytes wide.

The most critical improvements to the AMD CDNA™ 2 architecture are in the communication capabilities of each GCD within the AMD MI200 device and especially in the unique capabilities offered by AMD Infinity Fabric™ technology. The previous generation relied on standard PCI-Express to connect to the host processor and offered three AMD Infinity Fabric™ links connecting to other GPUs. In the flagship HPC topology example show in Figure 1.4.1.4a, the AMD CDNA 2 architecture builds out the communication capabilities to an entirely different level with four different types of interfaces specialized for different purposes: in-package Infinity Fabric, inter-package Infinity Fabric links, coherent Infinity Fabric links to the host processor, and a downstream PCIe link. The in-package Infinity Fabric, coherent Infinity Fabric, and downstream PCIe links are all novel and unlock the unique system capabilities illustrated in Figure 1.4.1.4a. In the more traditional mainstream and flagship machine learning topologies, illustrated in Figure 1.4.1.4b-c, the GPUs are connected to the host processor via PCIe but still benefit from the increased number of GCD-to-GCD Infinity Fabric Links within the GPU device as well as the inter-package external Infinity Fabric links.

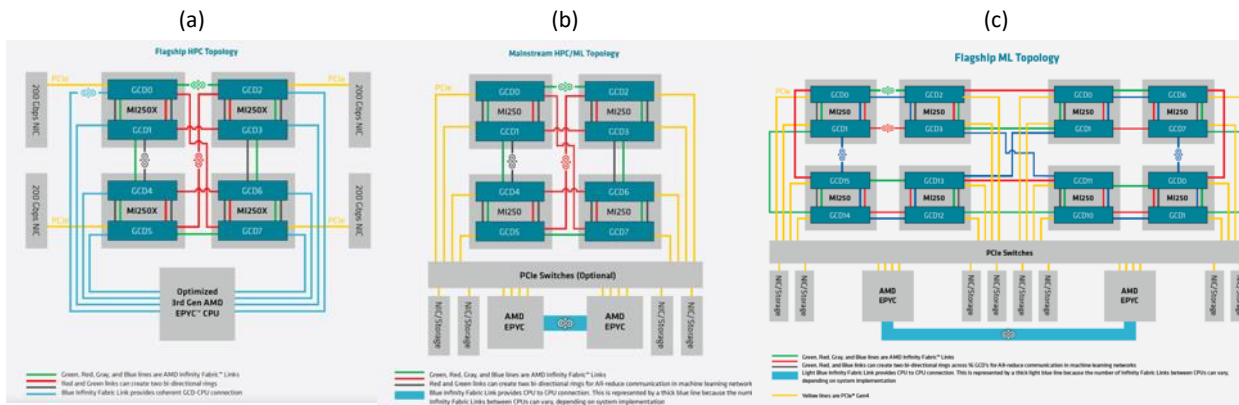


Fig.1.4.1.4. Block diagrams of a node built using the AMD Instinct™ MI250X accelerator and optimized 3<sup>rd</sup> generation AMD EPYC™ processor: a) flagship HPC, b) mainstream HPC/ML, c) ML-optimized

The key to accelerated computing for HPC and ML is a software stack and ecosystem that easily unlocks the capabilities for software developers and customers. AMD ROCm™ stack, shown in Figure 1.4.1.5, provides an open-source and easy to use set of tools that are built around industry standards and enable creating well-optimized portable software for everything from simple workstation programs to massive exascale applications. The principles behind AMD ROCm are fairly simple. First, accelerated computing requires equality between both processors and accelerators. While they focus on different workloads, they should work together effectively and have equal access to resources such as memory. Second, a rich ecosystem of software libraries and tools should enable portable and performant code that can take advantage of new capabilities. Last, an open-source approach empowers vendors, customers, and the entire community along with amplification of AMD’s own investment.

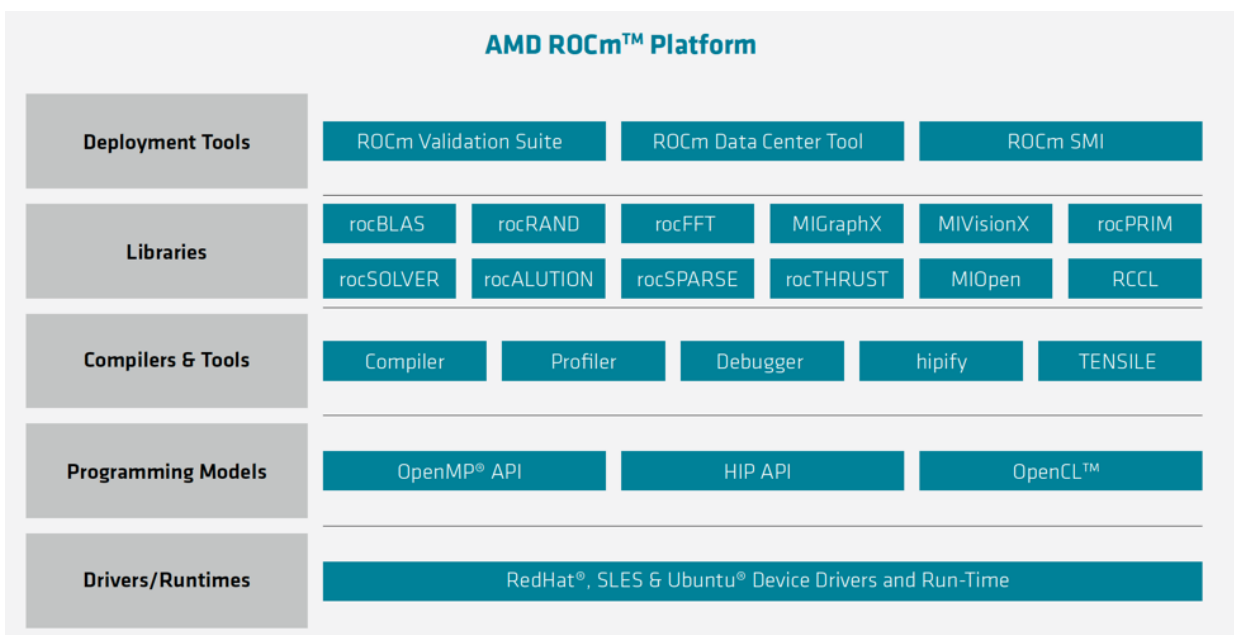
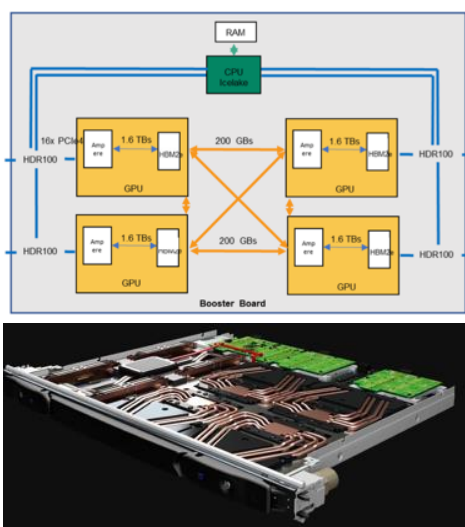


Fig.1.4.1.5: AMD’s open-source ROCm stack includes tools developers need to build high-performance

applications for scientific computing and machine learning

The AMD ROCm™ ecosystem is crucial for putting the capabilities of the AMD CDNA™ 2 architecture into the hands of developers, vendors, customers, and the entire community. For example, the rocBLAS library has incorporated the new instructions for FP64 matrix multiplication and packed FP32 vectors so that developers that work with higher-level libraries will get excellent performance from day one. At a lower level, the ROCm compiler and runtime can take advantage of these same features to generate high-performance binaries for custom code and a more diverse set of applications beyond linear algebra. At an even higher level, AMD’s Infinity Hub contains containerized HPC and ML applications that are ready to use and support the latest MI200 series accelerators. At the same time, the unique capabilities of the AMD CDNA 2 architecture - especially cache coherency, enable simplifying applications and delivering even greater performance. For example, parts of NWChemEx make use of coherent unified memory; porting this to non-coherent processors and accelerators could add complexity, introduce new bugs, and generally delay deploying the application. AMD MI250X accelerator with the optimized 3rd gen AMD EPYC™ processor in a cache-coherent configuration can greatly improve productivity.

To conclude the current deployment of heterogeneous architectures based on CPU/GPU, it has to be notify the Italian pre-exascale HPC system, named **Leonardo**, is composed of a booster module with 3456 compute nodes with a CPU Intel Xeon Ice-Lake and 4x Nvidia Ampere 100 GPUs SXM able to provide 240 PFlops of HPL (High Performance Linpack) benchmark. The system architecture of the node is shown in Fig. 1.4.1.6 and the main specification are reported in the Table 1.4.1.4. The blad is based on ad hoc board provided by Atos in BullSequana platform with CPU-GPU connection via PCIe4 16x via HDR Connect16 HCA that provides 16 PCI links towards the CPU and 16 PCI links towards GPUs with a bandwidth of 64 GB/s duplex. The GPU-GPU connection is composed of all-to-all full NVlink at 200 GB/s bi-directional.



Tab.1.4.1.4

Leonardo booster module compute node specification:

- CPU: 1 Intel Xeon Icelake, 32 cores (2.4 GHz -250 W)
- Memory: 256 GB
- CPUs:Accelerators: 1:4
- Accelerators: 4 x Nvidia Ampere 100 (SXM)
- GPU-CPU bandwidth: 400 GB/s
- Accelerator memory: 256 GB HBM2e (4x64 GB)
- Accelerator memory bandwidth: 6.5 TB/s (1.6 TB/s x 4 GPUs)

Fig.1.4.1.6: Leonardo Booster module with system architecture of the blade

Currently for CPU and FPGAs is widely used in embedded systems. Integrating the high-level management functionality of processors and the stringent, real-time operations, extreme data processing, or interface functions of an FPGA into a single device forms an even more powerful embedded computing platform.

SoC FPGA devices integrate both processor and FPGA architectures into a single device. Consequently, they provide higher integration, lower power, smaller board size, and higher bandwidth communication between the processor and FPGA. They also include a rich set of peripherals, on-chip memory, an FPGA-style logic array, and high speed transceivers.

The Table 1.4.1.5 shows the SoC FPGA products currently available on the market for three different vendors. The processors in these devices are fully dedicated, “hardened” processor subsystems (not a soft IP core implemented in the FPGA fabric). All three industry product lines employ a full-featured ARM® processor with a memory hierarchy and dedicated peripherals that largely boot, run, and act like any “normal” ARM processor.

Integrating these technologies on the same piece of silicon eliminates the cost of one of the plastic packages, and saves board space. If both the CPU and FPGA use separate external memories, it may also be possible to consolidate both into one memory device, for further savings. As the signals between the processor and the FPGA now reside on the same silicon, communication between the two consumes substantially less power compared to using separate chips. The integration of thousands of internal connections between the processor and the FPGA leads to substantially higher bandwidth and lower latency compared to a two-chip solution. Previously, the lack of an ARM processor had been a barrier to using FPGA technology for full production, but this new breed of SoC FPGAs delivers a fully-functional, fully-compatible, high-performance, dual-core ARM Cortex-A9 processor running up to 1GHz with today’s 28nm process technology.

The Intel SoC solution based on ARM Hard Processor System (HPS). It consists of a multi-core ARM Cortex MPCore applications processor, a rich set of peripherals, and multiport memory controller shared with logic in the FPGA. The HPS gives you the flexibility of programmable logic combined with the performance and cost savings of hard IP.

- Embedded peripherals eliminate the need to implement these functions in programmable logic, leaving more FPGA resources for application-specific custom logic and reducing power consumption
- The hard multiport SDRAM memory controller, shared by the processor and FPGA logic, supports DDR2, DDR3, DDR4, LPDDR2, LPDDR3, RDRAM 3, and QDR II+ SDRAM devices with an integrated ECC support for high reliability and safety-critical applications.

High-throughput data paths between the HPS and FPGA fabric provide a level of interconnect performance that is not possible in two-chip solutions. The tight integration between the HPS and FPGA fabric provides over 125 Gbps peak bandwidth in the Arria V SoC for example with integrated data coherency between the processor and the FPGA.

The flexibility offered by the FPGA logic fabric, with up to 5.5 million Logic Elements (LE) in the Stratix 10 SoC, lets you differentiate your system by implementing custom IP or off-the-shelf preconfigured IP from Intel or its partners into your designs. This allows:

- Adapt quickly to varying or changing interface and protocol standards

- Add custom hardware in the FPGA to accelerate time-critical algorithms and create a compelling competitive edge
- Reduce power consumption and FPGA resource requirements by leveraging hard logic functions within the FPGA, including PCIe ports and additional multiport memory controllers.

	Altera SoC	Xilinx Zynq 7000 EPP	Microsemi SmartFusion2
Processor	ARM Cortex-A9	ARM Cortex-A9	ARM Cortex-M3
Processor Class	Application processor	Application processor	Microcontroller
Single or Dual Core	Single or Dual	Dual	Single
Processor Max. Frequency	1.05 GHz	1.0 GHz	166 MHz
L1 Cache	Data: 32 KB Instruction: 32 KB	Data: 32 KB Instruction: 32 KB	No data cache Instruction: 8 KB
L2 Cache	Unified: 512 KB, with Error Correction Code	Unified: 512 KB	Not Available
Memory Management Unit (MMU)	Yes	Yes	Yes
Floating Point Unit/NEON Multimedia Engine	Yes	Yes	Not Available
Acceleration Coherency Port (ACP)	Yes	Yes	Not Available
Interrupt Controller	Generic (GIC)	Generic (GIC)	Nested, Vectored (NVIC)
On-Chip Processor RAM	64 KB, with ECC	256 KB, no ECC	64 KB, no ECC
Direct Memory Access Controller	8-channel ARM DMA330 32 peripheral requests (FPGA + HPS)	8-channel ARM DMA330 4 peripheral requests (FPGA only)	1-channel HPDMA 4 requests
External Memory Controller	Yes	Yes	Yes
Memory Types Supported	LPDDR2, DDR2, DDR3L, DDR3	LPDDR2, DDR2, DDR3L, DDR3	LPDDR, DDR2, DDR3
External Memory ECC	16 bit, 32 bit	16-bit	8 bit, 16 bit, 32 bit
External Memory Bus Max. Frequency	400 MHz (Cyclone V SoC), 533 MHz (Arria V SoC)	667 MHz	333 MHz
Processor Peripherals	1x Quad SPI controller 1x NAND controller 2x 10/100/1G Ethernet controller 2x USB 2.0 On the Go controller 1x SD/MMC/SDIO controller 2x UART 4x I2C controller 2x CAN controller 2x SPI master, 2x SPI slave controller 4x 32 bit general-purpose timers 2x 32 bit watchdog timers	1x Quad SPI controller 1x static memory controller (NAND, NOR, or SSRAM) 2x 10/100/1G Ethernet controller 2x USB 2.0 OTG controller 2x SD/SDIO controller 2x UART 2x I2C controller 2x CAN controller 2x SPI controllers (master or slave) 2x 16 bit triple-mode timer/counters 1x 24 bit watchdog timer	1x 10/100/1G Ethernet controller 2x USB 2.0 OTG controller 2x UART 2x I2C controller 1x CAN controller 2x SPI 2x general-purpose timers 1x watchdog timer 1x real-time clock (RTC)
FPGA Fabric	Cyclone V, Arria V	Artix-7, Kintex-7	Fusion2
FPGA Logic Density Range	25 K to 462 K LE	28K to 444 K LC	6 K to 146 K LE
Hardened Memory Controllers in FPGA	Up to 3, with ECC	Not available	Not available
High-speed Transceivers	Available at all densities	Higher-density devices only	Higher-density devices only
Analog Mixed Signal (AMS)	Not available	2 x 12-bit, 1 MSPS analog-to-digital converters (ADCs)	Not available
Boot Sequence	Processor first, FPGA first, or both simultaneous	Processor first	Processor first

Tab.1.4.1.4: The commercial FPGA devices SoC currently available

Intel offers a full-range SoC FPGA product portfolio spanning high-end, midrange, and low-end applications. In HPC data center among the most interesting there are:

- **Agilex F Series:** enable next-generation, high-performance applications via higher fabric performance, lower power, gains in Digital Signal Processing (DSP) functionality, and higher designer productivity compared to previous-generation FPGAs. The Intel Agilex FPGA meets the myriad challenges of data-centric compute while opening up new possibilities for business and industry. The FPGA fabric die at the heart of every Intel Agilex FPGA is built on Intel’s 10 nm SuperFin chip manufacturing process technology, the world’s most advanced FinFET process. The fabric die leverages the second generation of Intel® Hyperflex™ FPGA Architecture, which uses registers, called Hyper-Registers, throughout the FPGA, optimized for leading performance on 10 nm. The second generation of Intel Hyperflex FPGA Architecture, combined with Intel® Quartus® Prime Software, delivers the optimized performance and productivity required for next-generation systems. The FPGA fabric also features architecture optimizations for accelerating AI functions and DSP operations through dedicated structures for half-precision floating point (FP16) and BFLOAT16, as well as increased DSP density compared to prior generation FPGAs. Intel Agilex FPGAs can implement fixed-point and floating point DSP operations with high efficiency. The DSP blocks provide 2X the number of 9x9 multipliers compared to the prior generation. This also doubles the amount of INT8 operations that Intel Agilex FPGAs can deliver per DSP block. The addition of new modes for FP8 and FP16 supports highly efficient implementations for specific AI workloads, such as CNNs for image and object detection with a lower device utilization and lower power compared to implementation with FP32. Announced but not yet available is the Apollo Agilex SOM as SoC (Fig.1.4.1.7). It takes advantage of the latest Intel® Agilex® SoC with 1400K logic elements to obtain performance and power breakthrough (with up to 40% lower power than Stratix 10 series). Combining high-end hardware interfaces such as two high-capacity and high-bandwidth DDR4 SO-DIMM Sockets (up to 32GB DDR4), on-board QSFP28 connector, PCIe Gen 4x16 up to 25.8 Gbps/ch with carrier, on-board USB-Blaster II, and FMC/FMC+ connectors for I/O expansion.

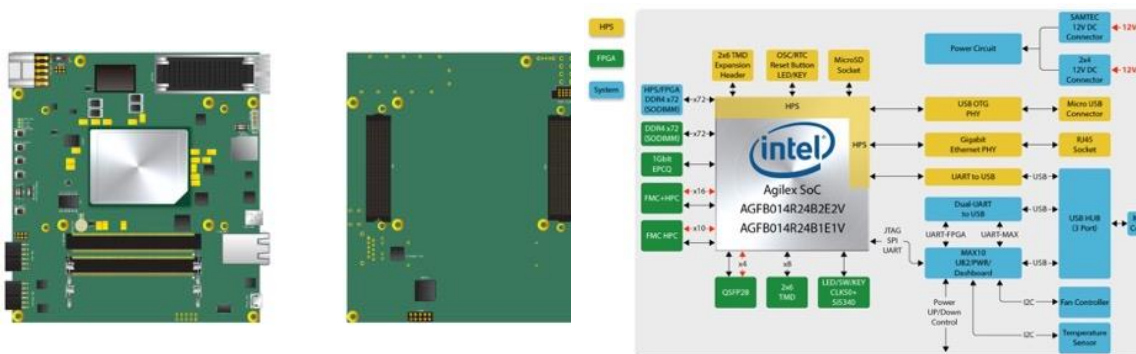


Fig.1.4.1.7; The new Intel Apollo Agilex SOM SoC FPGA and block diagram

- **Stratix 10:** offer breakthrough advantages in bandwidth and system integration, including a next-generation HPS. Stratix 10 devices feature the revolutionary HyperFlex™ architecture and are manufactured on the Intel 14 nm Tri-Gate process, delivering breakthrough levels of performance and power efficiencies that were previously unimaginable. When coupled with 64 bit quad-core ARM Cortex-A53 processor and advanced heterogeneous development and



debug tools such as the Intel FPGA SDK for OpenCL™1 and Intel SoC FPGA Embedded Design Suite (EDS), Stratix 10 devices offer the industry’s most versatile single-chip heterogeneous computing platform.

- **Arria 10:** SoCs deliver optimal performance, power efficiency, small form factor, and low cost for midrange applications. The Arria 10 SoCs (Fig. 1.4.1.8), based on TSMC’s 20 nm process technology, combine a dual-core ARM Cortex-A9 HPS with industry-leading programmable logic technology that includes hardened floating-point DSP blocks. By building on the architecture of the dual-core ARM Cortex-A9 processor from the Arria V SoC, the Arria 10 SoC offers an easy performance upgrade and software migration path for Arria V and Cyclone V SoC designs. The architectural innovation in the implementation of IEEE 754 single-precision hardened floating-point DSP blocks in Arria 10 SoCs enables processing rates up to 1.5 TFLOPs and power efficiency up to 40 GFLOPs/Watt.

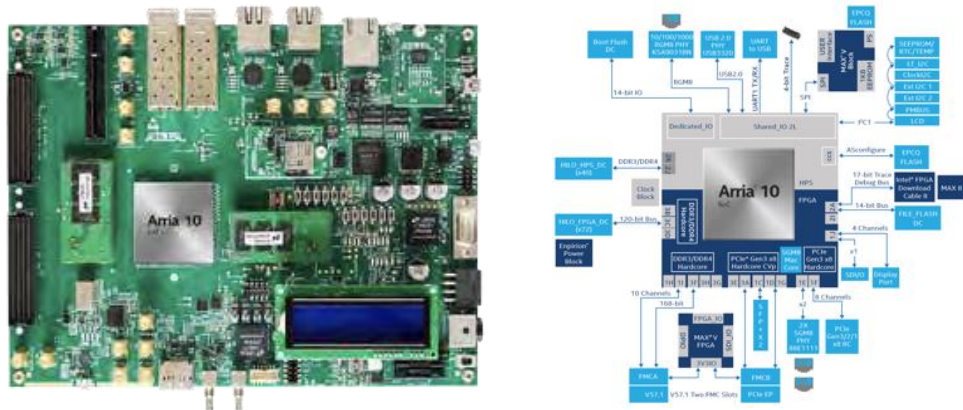


Fig.1.4.1.8; The commercial FPGA discrete currently available

Discrete FPGA devices requiring a compute host compatible with a Peripheral Component Interconnect (PCI). The following table 1.4.1.5 resumes the commercial discrete FPGA boards currently available on the market with technical specification available on line in [FPGA,2021]

Vendor	Model	FPGA	PCIe	Tranceiver
Micron	SB-851	Xilinx Virtex Ultrascale+ UV9P FPGA	x16 Gen3	2 QSFP28
Micron	EX-700	up to six single-wide AC Series modules	x16 Gen3	N/A
Xilinx	Alveo	U200, U250 or U280 Modules	x16 Gen3	2xQSFP28
Achronix	VectorPath™ S7t-VG6	AC7t150	x16 Gen4	QSFPDD QSFP56
Intel	Stratix	Stratix 10GX/DX/SX/TX/NX/MX/SX SoC	X16 Gen4	2xTQSFP56
Intel	Agilex	Agilex F/I-series/DE 10 /SoC	X16 Gen4	2xQSFPDD

Tab.1.4.1.5; The commercial FPGA discrete currently available

The Achronix FPGAs cover a small niche of the market but miss the High Level Synthesis flow required by TextaRossa. The Intel Stratix family will be soon superseded by the forthcoming Intel Agilex with FPGA OneAPI. But there are risks on their availability on the market and FPGA OneAPI development flow is still at its beginning. At the current state of the technology, the most viable way for FPGA accelerator for HPC solutions to be adopted in TextaRossa is to rely on Alveo boards for the following reasons:

- they are a mature technology and their Vitis based development flow has more than 2 years of presence in the market, so it is quite tested (the same is not still true for OneAPI)
- the boards have interesting characteristics as they can be both tailored for heavy computational loads (U250) and for more I/O demanding loads (U280 which offers gen4x8 connectivity and 2 High Bandwidth Memory (HBM) slots of HBM, each having 4GBytes of memory).

The Appendix 1 reported the product table for Xilinx Alveo and Intel Agilex FPGA, in order to compare the specifications for the next activities.

The ENEA CRESCO FPGA Lab (Fig.1.4.1.9) has made available to the Textarossa a pool of compute nodes equipped as follow:

- n.6 Linux X86\_64 nodes with 2 x Intel Xeon Haswell CPU, 128 GB RAM
- n.2 Xilinx U280 + n.2 Xilinx U250
- Sys.Op. Linux Centos 7.4
- Development software tools: Xilinx VITIS & Intel OneAPI

The access to the FPGA Lab resources is available getting an account in ENEAGRID infrastructure following the access rules reported in ENEA CRESCO portal [CRESCO,2021]. The compute nodes equipped with FPGA boards can be access via *ssh* on front-end node: *cresco-in.portici.enea.it* and forwarding on to:

*cresco-xilinx0/1/2/3/4/5.portici.enea.it*

A graphical remote access to compute nodes of the FPGA Lab is available using ENEA F.A.R.O. (Fast Access Remote Objects) using a client ThinLIC on the front-end node: *cresco-in-gui.portici.enea.it*

For the TextaRossa project are available some Development Operations tools as follow:

A gitlab for source codes and data benchmark repositories [GITLAB,2021]

ENEA Staging Storage Sharing system based on Owncloud using AFS (Andrew File System), The geographical distributed filesystem of the ENEAGRID infrastructure as backend [E3S,2021]



Fig.1.4.1.9: The ENEA FPGA Lab, with F.A.R.O as GUI to develop with VITIS and Intel OneAPI

	U250	U280
Look-up tables	1,728K	1,304K
Registers	3,456K	2,607K
DSO Slices	12,288	9,024
DDR memory	4x 16GB 72b	2x 16GB 72b
DDR total capacity	64 GB	32 GB
DDR Max Data Rate	2400 MT/s	2400 MT/s
DDR Total BW	77GB/s	38 GB/s
HBM2 Total Capacity	-	8 GB
HBM2 Total Bandwidth	-	460 GB/s
Internal memory total capacity	57 MB	43 MB
Internal memory total BW	47 TB/s	35 TB/s
PCI Express®	Gen3 x16	2x Gen4 x8, Gen3 x16
Network Interface	2x QSFP28	2x QSFP28
Typical power	110 W	100 W
Maximum power	225 W	225 W

Tab.1.4.1.6; The commercial FPGA discrete currently available

The FPGA Xilinx U250 is more oriented for computations that require a lot of logic (intensive computations), while the U280, having more I/O bandwidth (2 HBM2 memory banks, 4 GB each, with a global bandwidth of 460 GB/s and 2x Gen4x8 PCIe channels) is more suited for applications that, while still being computationally demanding (U280 has a quite high count of logic resources - DSP, LUT, and registers), have heavy I/O requirements. Apart from the different characterization depending on the availability of more I/O logic or a larger quantity of “pure computational” logic, both the FPGA boards are well suited for HPC applications, having - both of them - a lot of logic resources (i.e. computational power) connected with internal memory (~50 MB) with a huge bandwidth (~40 TB/s). The Tab.1.4.1.6 shows the technical specification of the U250/U280 FPGA board installed in the ENEA FPGA Lab.

Finally, last but not least, it is needed to report the **European Processor Initiative (EPI)** [Kovac,2020] project, whose aim is to design and implement a roadmap for a new family of low-power European processors for exascale computing, high performance big data and a range of emerging application. The technology domain of the EPI activities in HPC landscape shall carry out:

- A first implementation of a processor platform addressing a strong set of common technologies different application domains. It deals with the selection of cutting-edge process technology, massive parallelism with multi-cores, a memory hierarchy with HBM integrated using a silicon interposer, a chiplet approach with a high-speed link between silicon dies, a low-power design approach with a low-voltage operating point and fine-grain power management, built-in security to isolate applications and resist against the new cyber threat environment. The software stack will be designed to integrate and take advantage of these features to achieve high-energy efficiency and maximize performance across a wide

range of layers from the low-level firmware, all the way up to system software and application run times.

- the development and demonstration of fully European processor IPs based on the RISC-V instruction set architecture, providing power-efficient and high-throughput accelerator tiles within the GPP chip. Using RISC-V allows for leveraging open-source resources at the hardware architecture level and software level, as well as ensuring independence from non-European patented computing technologies.

Exascale computing systems need to simultaneously meet challenges related to performance, system cost and energy efficiency. To deliver performance, a vast amount of resources is required, but the wrong choices of components, architecture or implementation might result in a system that is much too expensive and/or too power hungry. To find the right balance, global system level optimization is necessary. For this purpose, EPI will harmonize the heterogeneous computing environment by defining a common approach: the so-called Common Platform (CP) shown in Figure 1.4.1.10 It will include the global architecture (hardware and software) specification, common design methodology and the global approach for power management and security.

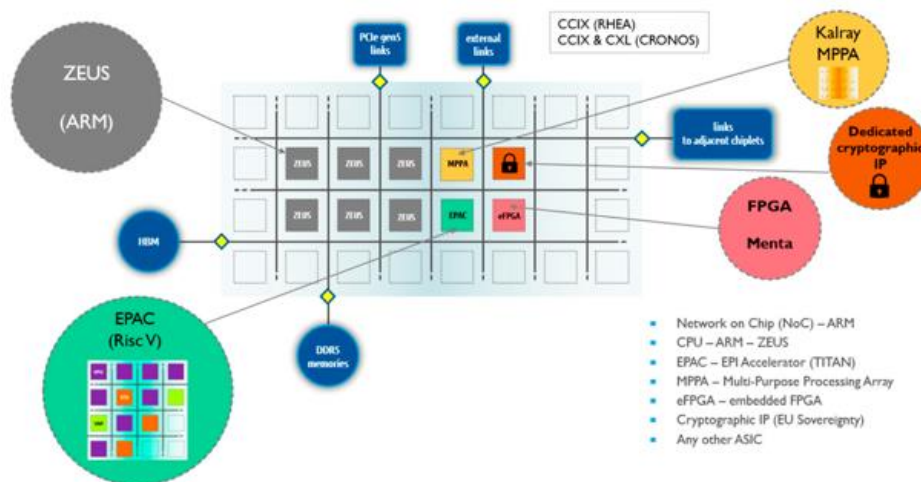


Fig.1.4.1.10; EPI Common Platform architecture

The CP is organized around a 2D mesh Network-on-Chip (NoC) connecting computing tiles based on general purpose Arm cores with high energy efficiency accelerator tiles, an RISC-V-based EPI accelerator (EPAC), a Multi-Purpose Processing Array (MPPA), cryptographic IP and embedded FPGA with different acceleration levels or any other application-specific accelerator. A common software environment between heterogeneous computing tiles will harmonize the system, as well as act as a common backbone of IP components for IO connection with the external environment, such as memories and interconnected or loosely coupled accelerators. With this CP approach, EPI will provide an environment that seamlessly integrates any computing tile. The right balance of computing resources for application matching will be defined through the ratio of the accelerator and general-purpose tiles. The EPAC accelerator tile will be a fully European processor IP accelerator, based on the RISC-V ISA and aimed at providing very low-power and

high-computing throughput accelerator to the general-purpose cores. It will include specialized tiles, such as a Vector Processing Unit (VPU), Stencil/Tensor accelerator (STX) and Variable Precision co-processor (VRP). A first silicon implementation of EPAC 1.0 test chip was announced in the middle of 2021 including of a subset of block functions synthesized on FPGA, such as: the Avispado RISC-V core, the VPU, the NoC (NoC9, the shared L2 cache with Coherence Home Node (L2HN), interrupt controllers, IO peripherals and several other components.

## WISHED STATE

There is a clear need to improve, where possible, the components of the current system architectures and to develop system software, libraries, and applications for the purpose of reaching a significant fraction of the peak performance of exascale supercomputers that will be available in the upcoming years. Only if this task is accomplished, those systems will have the chance of being sustainable, meaning that their cost and power consumption will be acceptable. Overlapping data communication and computation on this kind of platform is the most critical requirement. Support for independent streams of processing (including the computation required for data movement) and programmable network interfaces are just some examples of possible solutions to reduce the overhead introduced by inter (and intra) nodes communication.

Taking into account the CPU/GPU/FPGAs technologies suitable for HPC exascale oriented two distinct wished states shall be assessed corresponding two different system architectures of a compute node for an energy sustainable exascale HPC system.

**WS-1.4.1.a:** a system architecture currently available based on CPU and integrated GPUs able to provide energy efficiency performance less than 100 mW/GFlops at level of compute node, with a high bandwidth of GPU-GPU interconnect and a unified memory address among GPUs. The CPU host shall guarantee an efficient GPU data movement for inter-node communication. This system architecture shall allow of using programming models and toolchains for new implementations of numerical algebra kernels as well as parallel applications widely used and AI algorithms in a complete HPC infrastructure able to scale up towards an exascale size with energy consumption sustainable (< 20 MW).

**WS-1.4.1.b:** a system architecture currently available based on SoC CPU+FPGA or CPU and discrete FPGA devices able to provide energy efficiency performance less than 100 mW/GFlops at level of compute node. The FPGA-FPGA interconnect shall be developed inside the compute node and whereas needed a unified memory address among FPGAs. The CPU host shall guarantee an efficient FPGA data movement for inter-node communication. This system architecture shall allow of using programming models and toolchains for new implementations of numerical algebra kernels as well as specific parallel applications and IA algorithms in a complete HPC infrastructure able to scale up towards a exascale size with energy consumption sustainable (< 20 MW).

## ACTION STATE

It is apparent that the aforementioned system architectures as well as the associated new programming models is not enough to get a real benefit at user application level so that a re-thinking of algorithms and new software refactoring of fundamental building blocks of real-world applications are also required. The

actions needed to achieve an assessment of the foregoing wished states during the activities of WP1 is simply listed in the following table:

	Design	Test
WS.1.4.1.a	Requirements and Specifications of compute node with a system architecture CPU+GPU based	Evaluation of KPIs of TEXTAROSSA applications and use cases using GPUs: - Mathlib-CNR - Mathlib-INRIA - UrbanAir - NestGPU -HEP -TNM -Smart Cities -DNN Inference
WS.1.4.1.b	Requirements and Specifications of compute node with a system architecture CPU+FPGA based	Evaluation of KPIs of TEXTAROSSA applications and use cases using FPGAs: -RAIDER - Mathlib-INRIA

### PRIORITY

To develop the TEXTAROSSA use cases and applications in WP6 and providing feedback to WP1 in a co-design loop approach. This software development requires hardware/software resources equipped with GPUs/FPGA devices currently available and software development tools such as Vitis for Xilinx FPGAs and NVIDIA Cuda for GPUs.

### REFERENCE

[FPGA,2021]:

<https://www.xilinx.com/content/dam/xilinx/support/documentation/selection-guides/ultrascale-plus-fpga-product-selection-guide.pdf>

<https://www.micron.com/products/advanced-solutions/advanced-computing-solutions/ac-series-hpc-modules>

<https://www.xilinx.com/products/boards-and-kits/alveo/u200.html#specifications> ,

<https://www.xilinx.com/products/boards-and-kits/alveo/u250.html#specifications>

<https://www.xilinx.com/products/boards-and-kits/alveo/u280.html>

<https://www.achronix.com/product/vectorpath-accelerator-card>

<https://www.intel.com/content/www/us/en/products/details/fpga/stratix/10.html>

[FPGA,2021]: <https://www.eneagrid.enea.it/CRESCOportal/>

[GITLAB,2021]: <https://gitlab-tex.enea.it>

[E3S,2021]: <https://textarossa.enea.it>

[Kovac,2020]: Kovač, M.; Notton, P.; Hofman, D.; Knezović, J. *How Europe Is Preparing Its Core Solution for Exascale Machines and a Global, Sovereign, Advanced Computing Platform*. Math. Comput. Appl. 2020, 25, 46. <https://doi.org/10.3390/mca25030046>

## 17 Task-1.4.2: Interconnection Networks

The goal of the TEXTAROSSA project in the framework of interconnect system is to design and provide a low-latency, direct network for FPGA accelerators optimized vs. the needs of new computing paradigms like AI and HPDA.

The interconnection between nodes is one of the crucial components to achieve optimal performance. The choice of the network technology and topology will impact the effective latency of the data exchange between FPGA nodes and the power efficiency. The objective is to find the best balance between network performance and energy consumption.

The aim of the activity is to design the interconnection networks for intra-FPGA/inter-FPGA communications.

## **CURRENT STATE**

Today computing efficiency, power consumption and the system cooling have become key factors driving HPC design. In this perspective, the next generation one-million cores HPC systems have to adopt low power CPUs, co-design and integrate application-oriented computing accelerators, designing scalable, low latency, high performance interconnection architectures suitable for such as extreme scale systems.

The ExaNeSt [*Katevenis,2016*] project, funded in EU H2020 research framework (call H2020-FETHPC-2014, n. 671553), aimed to demonstrate the efficient usage of low power architectures in Exascale computing platforms.

One of the main goals within ExaNeSt has been the design of ExaNet: a novel, unified (for data and storage traffic), low latency, high throughput, RDMA-based interconnect architecture suitable for extreme scale systems. The project leaned on high end SoC FPGAs — the Xilinx Zynq UltraScale+ [*Xilinx,2021*] with 4 ARM Cortex-A53 embedded cores running at up to 1.5GHz — to integrate thousands of cores into a fully working system prototype.

ExaNet was responsible for data communication at the lower level of the network interconnect of the ExaNeSt project. The INFN APE Research group, which in the past has designed the APENet [*Ammendola,2011*] 3D-Torus network architecture, was responsible for the ExaNet Network IP [*Ammendola,2018*] that provided switching and routing features and managed the communication over the High-Speed Serial (HSS) links through different levels of the ExaNeSt interconnect hierarchy.

EuroEXA [*Euroexa,2021*] is a major European FET research initiative that leverages on previous projects results (ExaNeSt, ExaNoDe [*ExaNoDe,2021*] and ECOSCALE [*Mavroidis,2016*]) to design a medium scale but scalable, fully working HPC system prototype exploiting state-of-the-art FPGA devices that integrate compute accelerators and low-latency high-throughput network.

EuroEXA enhances the ExaNet architecture, inherited by the ExaNeSt project, and introduces a multi-tier, hybrid topology network built on top of an FPGA-integrated Custom Switch that provides high throughput and low inter-node traffic latency for the different layers of the network hierarchy.

The EuroEXA computing node, the CRDB (*Co-design Recommended Daughter Board*), is based on a module hosting Xilinx Ultrascale+ FPGAs for application code acceleration hardware, control and network implementation. The interconnect is an FPGA-based hierarchical hybrid network characterized by direct topology at blade level (16 computing nodes on a board named Blade) and a

Custom Switch [Biagioni,2020], implementing a mix of full-crossbar and Torus topology, for interconnection with the upper levels.

**WISHED STATE**

The goal is to offer support for the execution on a system of multiple FPGAs of applications developed according to the streaming programming models based on Kahn Process Networks (KPNs).

It will be carried out designing a final architecture able to allow for a directed graph of tasks built and mapped on the multiple FPGA network for the execution. It will leverage on the ExaNet interconnect IPs enhanced by intra-node and inter-node communication links for FPGAs optimized vs. the needs of new computing paradigms like AI and HPDA. A full software stack, including low-level system software and interfaces with HLS shell for programming the heterogeneous hardware system, will allow users to exploit the enhanced ExaNet IPs, programming and configuring the hardware at higher abstraction level as possible.

**WS.1.4.2.a:** HDL-based design of: (i) high speed, low-latency, high-throughput intra-node switch and inter-node serial link controller, (ii) optimized IO interface for HLS-based accelerators supporting the integration in the VITIS flow (Task 2.4)

**WS.1.4.2.b:** to design a Linux device driver able to set configuration parameters of the network (node address and topology) and monitor its status (Task 4.4).

**WS.1.4.2.c:** to design a user space library able to provide methods to the toolchains to configure and initialize the network IPs and to setup the communication channels between HLS tasks deployed either on the same FPGA or on different interconnected FPGAs (Task 4.4).

**ACTION STATE**

In order to achieve the wished goal we plan to design and implement both hardware and software components.

	Design	Development	Test
WS.1.4.2.a	HLS based design intra-node switch and serial link control	PoC on FPGA	Dedicate functional tests with HLS kernels as communication endpoints measuring: -Communication Latency -Communication Bandwidth
WS.1.4.2.b	Linux driver to set the network	PoC on server equipped with FPGA	Dedicated functional tests
WS.1.4.2.c	User Space Library to configure the network	PoC on server equipped with FPGA	Dedicated functional tests



## PRIORITY

The produced IP will be evaluated through incremental integration in the other WPs.

1. node-testbed: a single multi-task based FPGA testbed consisting of a limited number of hardware accelerated tasks and network IP implementing intra-node communication. This testbed will allow us to evaluate the node architecture, the supporting software and the programming framework.
2. small-scale testbed made of few node-testbeds interconnected through inter-node communication IPs to validate the multi-node platform, the toolchains and preliminary release of the user space library.
3. final testbed, fully integrated hardware-software system, validated running kernels selected from the TEXTAROSSA application portfolio to evaluate and characterize system performance.

## REFERENCE

[Katevenis,2016] Katevenis M et al. 2016 The ExaNeSt Project: Interconnects, Storage, and Packaging for Exascale Sys- tems 2016 Euromicro Conference on Digital System Design (DSD) pp 60–67

[Xilinx,2021] Xilinx zynq ultrascale+ MPSoC devices accessed: 30/Oct/2017 URL <https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html>

[Ammendola,2011] R. Ammendola, A. Biagioni, O. Frezza, F. Lo Cicero, A. Lonardo, P. S. Paolucci, D. Rossetti, A. Salamon, G. Salina, F. Simula, L. Tosoratto, and P. Vicini, “APENet+: high bandwidth 3D torus direct network for petaflops scale commodity clusters,” *Journal of Physics: Conference Series*, vol. 331, no. 5, p. 052029, 2011,

[Ammendola,2018] R. Ammendola, A. Biagioni, F. Capuani, P. Cretaro, G. De Bonis, F. Lo Cicero, A. Lonardo, M. Martinelli, P. Paolucci, E. Pastorelli, L. Pontisso, F. Simula, and P. Vicini, “Large scale low power computing system: Status of network design in ExaNeSt and EuroEXA projects,” *Advances in Parallel Computing*, vol. 32, pp. 750–759, 2018

[Euroexa,2021] Euroexa website, accessed: 2021-11-30, <https://euroexa.eu/>

[ExaNoDe,2021] ExaNoDe accessed: 2021-11-30, <http://exanode.eu/>

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

[Biagioni,2020] Andrea Biagioni, Paolo Cretaro, Ottorino Frezza, Francesca Lo Cicero, Alessandro Lonardo, Pier Stanislaw Paolucci, Luca Pontisso, Francesco Simula and Piero Vicini, “EuroEXA Custom Switch: an innovative FPGA-based system for extreme scale computing in Europe”, *EPJ Web of Conferences* 245, 09004, 2020

## 18 Task-1.5: Hardware Platforms

The objective of this task is to define two types of platforms/compute nodes featuring the two-phase cooling technology in order to improve the readiness of this technology. This technology will be demonstrated on two types of accelerators: first FPGA accelerators, with moderate power and second GPU with High Power (more than 300W). With the limited budget of T1.5 task, it is not

possible to develop specific nodes according to the heterogeneous architectures defined in T1.4, but the objective is to define the characteristics of these nodes that are close to this optimal definition and will provide the best demonstrators for the two-phase cooling technology. The selection and needed adaptations will be developed in WP5. This gap analysis is then focussed on cooling at level of node and rack for both FPGA and GPU platforms.

Cooling electronic components that use a closed loop liquid circuit applied directly to or near the surface of the chip is not a new technology. This approach has been used in past, and now almost exclusively on mainframe or HPC commonly found in supercomputer facilities. In recent years, economical versions of direct cooling using water have been developed and sold to the personal computer market for those customers wishing to maximize performance. Nowadays a version of the technology, with modifications, is available for the commercial server market. 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 later to the building 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.

Usually the direct cooling systems are actives requiring an external source of energy to move the fluid into the loop. The general scheme is reported in Fig.1.5.1 where the main components of the loop may be distinguished.

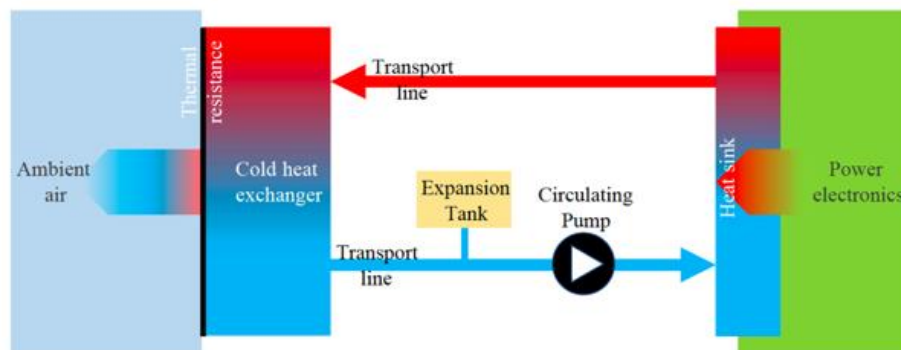


Fig.1.5.1: Schematic of Direct Liquid Cooling System

The circulating pump, or the compressor in the case of vapour compression technologies, has to be designed to counterbalance all the pressure drops in the loop and to maintain the necessary mass flow rate to evacuate the thermal load from the heatsink, where thermal load is transferred to the coolant. In the cold heat exchanger, heat is definitively evacuated to the external ambient. An expansion tank is used to prevent fluid expansion, eventual water-hammers and other instabilities to mechanically damage loop's components. Active direct cooling can be single or two-phase. Even if heat transfer mechanisms are quite different between them, the constituting components play the same role. In two-phase systems either latent and sensible heats are used: the heatsink is an evaporator and the cold source becomes the condenser (with sub-cooling). Referring to the scheme reported in Fig. 1.5.1 and for standard two-phase pumped circuits, in steady state conditions, the

sub-cooled fluid coming from the cold heat exchanger/condenser, flows to the heatsink. Hot fluid or vapour reaches the cold heat exchanger or condenser where heat is definitively evacuated to the external ambient. Through the return transport/liquid line, sub-cooled liquid return to the evaporator where the loop starts again. The expansion tank is also used to impose and regulate saturation conditions, but this point will be better approached in the following. In other cases, the pump is replaced by a compressor, the tank suppressed and an expansion valve mounted on the liquid line, obtaining a Rankine vapor compression cycle. In the case where a direct contact technology is used, such as jet-impingement/spray-cooling, the scheme reported in Fig. 1.5.1 is also valid, but a collecting tank must be provided to ensure that fluid is in liquid state before its return to the evaporator.

Single phase cooling systems, also called “liquid cooling”, are maybe the simplest and most used configuration. The physical principle of this technology is relatively simple and, in its “classical configuration”, is the best-known technology. The working fluid, heated in the heatsink, flows within the transport lines, ideally adiabatic, to the cold heat exchanger. A pump is used to move the fluid into the loop and it has to be designed to ensure the adequate mass flow rate and the necessary pressure head to counterbalance for all the pressure drops in the circuit. Some accessories, such as expansion vessel and valves have to be provided to ensure the smooth functioning. In Fig. 1.5.2a a concept scheme is reported.

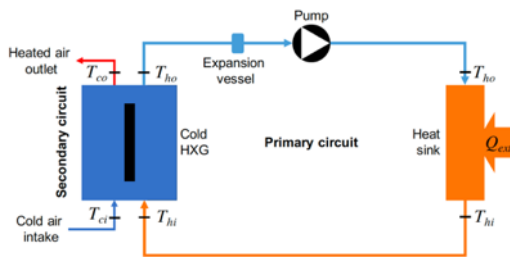


Fig.1.5.2a: scheme of single phase cooling

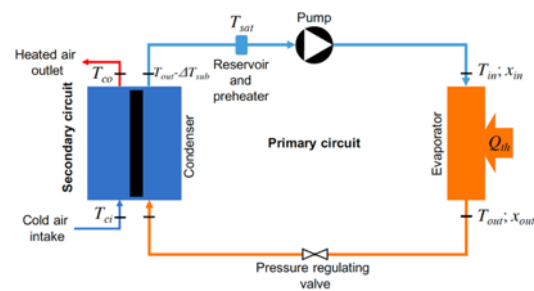


Fig.1.5.2b: scheme of two phase cooling

The active two phase cooling systems are divided in two categories with different physical working principles and thermodynamic characteristics. In pumped two-phase technology the evaporator operates at higher temperature than condenser and a pump is installed on the liquid line. In vapor compression technology, because of the use of Rankine cycle, the evaporator can operate at lower temperature than condenser and a compressor is used on the vapor line.

Even if the concept scheme for this kind of technology is similar to the one related to single-phase cooling systems, the operating principle is completely different. Here, latent heat is used to extract heat by hot source. In Fig. 1.5.2b the concept scheme of a pumped two-phase cooling system is reported. In the case where a free surface tank is used as a reservoir, where saturation conditions are imposed, subcooled liquid coming from the condenser enters in it.

## 19 Task-1.5.1: FPGA Platform

### CURRENT STATE

The state of the art of the platforms that have been developed within European projects capable of hosting discrete FPGA accelerators or SoCs is listed below. This list also contains systems available on the market or in production that have become part of the TOP500 HPC list.

- **ExaNoDE.** The principle of the project is to apply novel 3D integration and hardware design technologies, mixed with virtualization of resources and the UNIMEM memory system to deliver a prototype-level system demonstrating that those technologies are promising candidates towards the definition of a compute node for the Exascale computing. The ExaNoDe project idea is built around the following design goals:
  - **Affordability:** Ensure the solution is commercially viable and competitive in both its performance and its cost of ownership.
  - **Design Efficiency:** Using power-efficient compute elements and System design principles to ensure minimum duplication and abstractions within the infrastructure, to avoid unnecessary power consumption and latency overheads.
  - **Operational Efficiency:** power consumption proportional to activities making actual progress.
  - **Everything Close:** Leverage physical distance and data locality to design for minimum resistance and capacitance, so as to deliver the lowest power overhead associated with the required data connectivity.

ExaNoDe is also closely collaborating with the ExaNeSt [Katevenis,2016] and ECOSCALE [Mavroids,2016] H2020 projects confluence in EuroEXA [Euroexa,2020]. ExaNeSt investigated how storage, interconnections and cooling systems will have to evolve towards Exascale. ECOSCALE, instead, aims to provide a holistic approach for a novel heterogeneous energy efficient hierarchical architecture, a hybrid MPI+OpenCL programming environment and a runtime system for exascale machines. The combination of these three projects aims at covering the whole picture of an Exascale HPC machine.

The main idea of the ExaNoDe integration concept is to create a Modular Compute System partitioned into a number of chiplets stacked on a Silicon Interposer, several of them being integrated with memory devices and FPGA on a Multi-Chip-Module (MCM).

The partitioning envisioned in the ExaNoDe project prototype is:

- **Compute subsystem partition:** general purpose multicore ARMv8 CPU, plus a set of specialised units such as GPU and generic heterogeneous accelerators.
- **Memory subsystem partition:** DRAM and NVMe modules.
- **Interconnect and I/O partition:** traffic routing and scheduling, accesses to remote memory and storage, legacy I/O interfaces (e.g., PCI-e).

The FPGA components are used to implement the I/O interfaces and the actual Compute subsystems. ExaNoDe plans to use off-the-shelf ARMv8 based SoCs integrating on the same die an FPGA and a multicore ARMv8 CPU, since the design of the compute Chiplet is out of the scope of the project. The selected target for the project is the Xilinx Zynq UltraScale+ MPSoC, embedding a quad ARM Cortex-A53 processor, a Mali-400 MP2 GPU and an FPGA.

Both the ARMv8 CPU and the FPGA will be used as compute units, in order to achieve the requested level of performance. The Xilinx SoC has been selected mainly for two reasons:

- the high level of integration between heterogeneous compute units (i.e., CPU, GPU, FPGA), that goes in the direction of a higher compute density;
- the flexibility they provide for integration in a prototype demonstrator (i.e., FPGAs used not only as compute units but also to host glue logic between components).
- Multiple chiplets will lie on the Silicon Interposer which provides Chiplet-to-Chiplet, Chiplet-to-I/O and Chiplet-to-Memory communication. ExaNoDe will address Processor-to-Memory and Processor-to-Processor bandwidth versus energy efficiency by implementing new memory schemes in order to realise the benefits of the 3D integration technologies and determining the optimal trade-offs for an interposer-based 3D implementation.

The architecture proposed in Exanode/Exanest is a compute node composed of:

- Xilinx Zynq UltraScale+ MPSoC
- Partitioned global address space based on Unilogic+Unimem architecture sharing the memory that communicate through a hierarchical communication infrastructure
- Reconfigurable acceleration logic based on FPGA
- MPI is used for communication among compute nodes via CPU-based routers following the application topology.

The block diagram of the compute node is depicted in Fig.1.5.1.1.

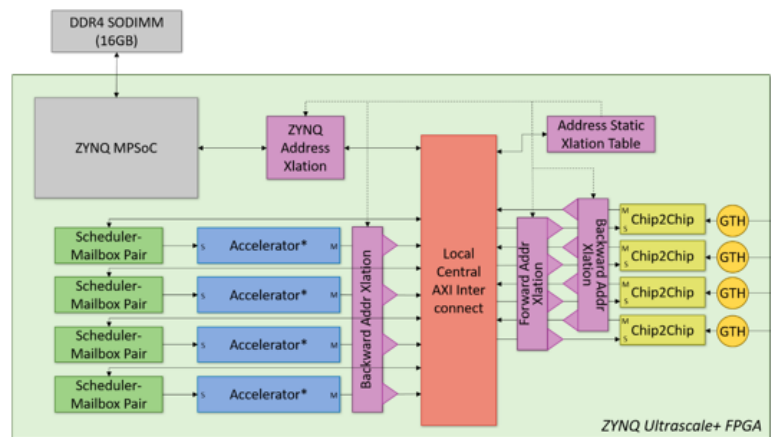


Fig.1.5.1.1: compute node including two MCMs and interposer technology

In Exanest, 4 compute nodes were integrated in a Quad FPGA Daughter Board (QFDB). The QFDB includes four Xilinx Zynq Ultrascale+ FPGAs, 64 GBytes of memory and an SSD shown in Fig.1.5.1.2. It is a energy-efficient System-On-Module, built around two state-of-the-art MCM's, each one including two Xilinx Zynq Ultrascale+FPGAs and another smaller embedded FPGA plus peripherals on an interposer (red block in the figure below). Each Ultrascale+ FPGA contains a quad-core A53 processor, significant reconfigurable logic (600K logic cells), and 40Mbits of internal memory. In addition, a significant amount of DDR4 memory is

connected to each FPGA (16 GBytes). One SSD is also connected to the daughter card, in the M.2 form factor.

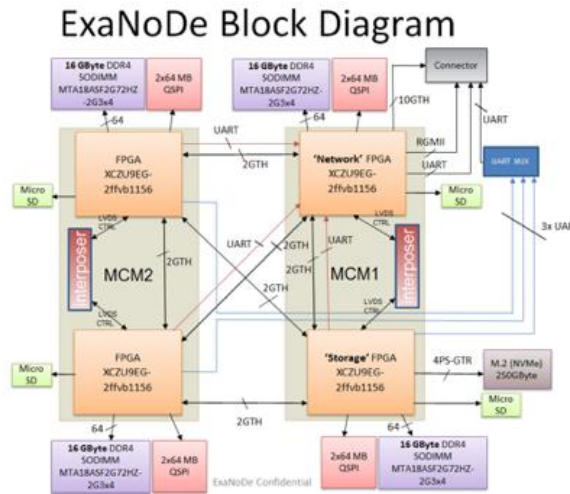


Fig.1.5.1.2: QFDB block diagram

An exanest blade were equipped with 4 QFDB, that means 16 compute nodes with Direct Liquid Cooling to assemble a rack with 12 blades that means 192 compute nodes. A HPC testbed with 8 blades is under test and a roadmap to exascale was provided estimating the maximum power for Exanest compute node technology solution as shown in the below table 1.5.1.1.

Hierarchy	Scale (compute node)	Performance	Maximum Power
Zynq Ultrascale+ FPGAs	1	1.5 Tflops	40 W
QFDB	4	6 Tflops	160 W
Blade (16 QFDB)	64	96 Tflops	2.5 kW
Chassis (6 Blades)	384	576 Tflops	20 kW+ 3 kW (cooling)
Rack (12 Chassis)	4608	6.9 Pflops	276 kW
Exascale (114 Racks)	663K	1 Eflops	40 MW

Tab.1.5.1.1: Exanest compute technology compute node scaled for Exascale

- DAVIDE.** The trade-off between computing power and energy-efficiency is becoming of crucial importance for high performance computing evolution, in particular for the race towards the exascale. In this context E4 built the DAVIDE [Bartolini,2018]: HPC cluster prototype, based on Open Power architecture, liquid cooled, with 45 computing nodes equipped with 4xGPU P100. Each node is equipped with Fine-Grain Power and Performance Monitoring Support [3]. The system was ranked #301 in TOP500 and #14 in GREEN500 in the June 2017 list. Each node of D.A.V.I.D.E. contains a dedicated programmable SoC that serves as an energy and power gateway. The energy-gateway (EG) is connected to power sensors. This allows measurement with high sampling rate the power consumption of the entire node, as well as of the main computing components in the node. At the higher level, the job scheduler features a dedicated plugin to receive the monitoring information and to correlate them with user requests and scheduling decisions. This correlation enables per user and per job energy-

accounting and profiling. In parallel, this information is recorded into a database, and computed by the management node for the training of job-to-power predictors based on the historical job request and power traces. Once available, the trained power predictors are used by the job scheduler to constrain the total power consumption of the D.A.V.I.D.E. supercomputing machine. The power cap (Limiting power and energy consumption) can be specified by the system administrator to follow infrastructure requirements. For this goal, the job scheduler and dispatcher is augmented by a dedicated engine. This is capable of using a per job power prediction to select which job should enter the supercomputing machine at each moment, in order to fulfill the specified power envelope while preserving job fairness.

- **CYGNUS.** It is an HPC cluster ranked #485 in top 500, June 2021. It is an example of GPU-FPGA hybrid cluster. Some nodes (named Albireo) are equipped with both FPGAs and GPUs, and other nodes (named Deneb) are with GPUs only. There are two types of interconnection network, 64 of FPGAs on Albireo nodes (2 FPGAs/node) are connected by 8x8 2D torus network without switch with an inter-FPGA network, for all computation nodes (Albireo and Deneb) are connected by full-bisection Fat Tree network with 4 channels of InfiniBand HDR100 (combined to HDR200 switch) for parallel processing communication such as MPI, and also used to access to Lustre shared file system.
- **AMPERE.** Ampere Altra Mt. Jade and Mt. Collins 2U Models are based on ARM, Up to 128 Armv8.2 cores with up to 3.0 GHz frequency. It supports 128 lanes of high speed PCIe Gen4 and 8x72 ECC protected DDR4 3200 memory. Description available online [*Ampere,2020*]. These systems are capable of hosting GPU or FPGA type accelerators.
- **GIGABYTE.** G242-P32 Model is a single socket ARM Ampere Altra Processor, this system is capable of hosting GPU or FPGA type accelerators. Description available online [*Gigabyte,2020*].

From point of view of the Direct Cooling In Quattro has developed a two-phase flow cooling system for thermal control of processors. Several prototypes have been developed and tested in different configurations. Configurations vary in terms of pump model, condenser types, evaporator geometry.

Part of the tests have been performed on processor simulators and part with real processors. The simulators are essentially aluminum blocks with electric cartridge heaters. The heaters are able to give the typical heat loads of the processors, and even more providing up to 1000 W of thermal power. Part of the tests have been performed on real processors like AMD Threadripper 3990 wx, one of the most powerful CPUs with its 64-Core, 128-Thread to support compute-intense workloads. Tests were performed in standard clock speed and even in overlocking configuration. The latter to simulate a real processor with high thermal power.

Tests (more than 5000 tests, actually) have shown that the two-phase flow cooling system performs quite well maintaining the processor's die temperature below its upper limit, even with the high TDP of 600 W. The new cooling system is able to cool up to 1000 W with an electric block that simulates a processor. These results are encouraging and demonstrate the validity of the new cooling technology in different conditions of thermal loads and ambient temperatures (up to 45 °C). Besides, these results demonstrate that it is possible to cool efficiently processors (CPUs, GPUs, FPGAs) with different TDP.

Recently, In Quattro, have developed the first commercial two-phase cooling systems for gaming PCs and professional workstations (Fig.1.5.1.3). The integration enables deployment of high wattage processors like AMD Ryzen Threadripper 3990x 64-Core, 128-Thread to support compute-intense workloads.



Fig.1.5.1.3 Testing Two-Phase Cooling System on AMD Threadripper 3990X at 3DWS Laboratory

The Direct Liquid Cooling (DLC) has undergone several tests in recent years. A complete test setup demonstration was carried out at ORNL [Coles,2014] to determine the thermal performance of a DLC and its potential impact on overall data center energy use. The Fig.1.5.1.4 shows the location of the system components and how they work with each other in the testbed layout. Integrated pump and cold-plate assemblies absorb heat from the CPU. The memory DIMMs are cooled by transferring heat to a manifold (in contact using heat transfer tape) carrying the cooling water. The cooling water supply and return paths are provided by a set of flexible tubes for each server. The heat collected is transferred using the tube set to the facility cooling support by means of a Cooling Distribution Unit (CDU). Some of the cooling for the server is not provided by the direct cooling system; there are a number of electronic components inside the server that still need to be air-cooled, using fans inside the server. However, the airflow requirement is reduced, and therefore the number of fans and their speed is reduced, providing a significant reduction in the energy consumed by the server for the same processing load. This server fan energy reduction accounts for most of the reduction in server energy use.

The test used a two-step approach: (1) measure the fraction of power supplied to the servers that is captured as heat by the prototype cooling system, and (2) use these fractions to calculate estimates using models comparing a server equipped with a stock cooling configuration (air cooled) to a prototype cooling system (direct-water cooled).

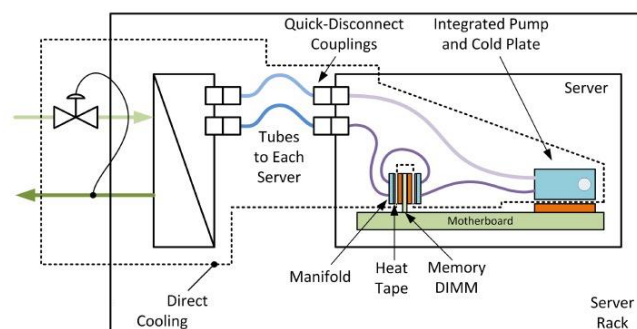


Fig.1.5.1.4: Schematic of Direct Liquid Cooling System



The parameters varied during the tests were: supply water temperature, supply water flow rate (which affects the return water temperature), and server power.

The lowest supply water temperature tested for this demonstration was 15°C. The lower supply water temperatures may produce improved results, but some data center operators will be concerned with managing condensation. Condensation is a valid concern and should be investigated with the installation of any direct cooling system. A key advantage when using direct cooling is that adequate cooling is also possible when using much higher (e.g., 45°C) supply water temperatures. In these situations, the direct cooled components are still well below critical temperature as specified by the component manufacturer or server original equipment manufacturer.

The thermal performance in this demonstration was evaluated for a range of supply water temperatures in order to investigate the potential for cooling infrastructure energy savings using atypical cooling infrastructures; for example, dry coolers or cooling towers only. The temperature range used was 15°C–45°C.

Furthermore three computing power levels were adopted: Idle (120 watt/node), Middle (270 watt/node), Full (430 watt/node) and a water supply flow rate of 4.9 Gallons per minute (gpm). The results for the maximum facility-side flow rate (~4.93 gpm) for the facility side that could be provided by the demonstration setup are shown in Fig.1.5.1.5.

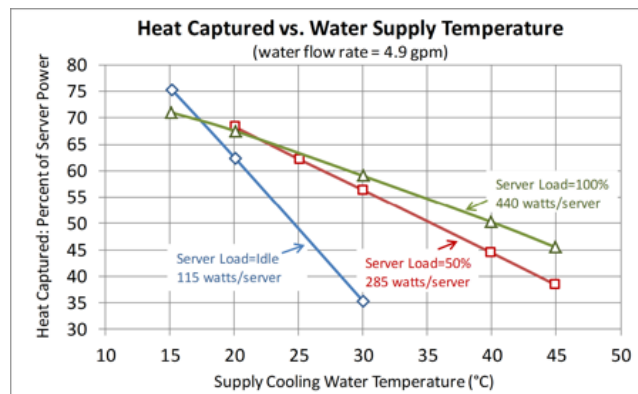


Fig.1.5.1.5: Maximum Heat Capture Vs. Water Supply Temperature For Three Server Power Levels

Taking into account the results of this tests, some observations can be done as follow:

- For all power levels, the fraction of heat captured increases as the supply water temperature is lowered.
- At higher supply water temperatures, the fraction of heat captured becomes distinctly different between the three power levels. At lower supply water temperatures, the fraction of heat captured is less different

This suggests that the server power level should be considered when investigating the supply water temperature for an optimized overall operating cost. For example, if the server power level is low, and the supply water temperature is high (30°C), the amount of heat captured is low compared to higher server power levels. It would be an advantage to be able to freely select from a wide range of supply water temperatures and capture a large fraction of the server power. However, the data show that this freedom is not available.

The Direct Two-Phase Cooling (DTPC) Two-phase pumped cooling systems are the most effective way to evacuate the thermal load by increasing the overall efficiency of the cooling system itself. Most of the authors that analysed this kind of technology focused their attention on performances, heat transfer efficiency and heat recovery requirements. When talking about electronics cooling, in fact, the attention is in particular paid to chips and data server cooling, which are characterised by quite high heat flux densities, but pretty small heat transfer surfaces. In this case, despite the really high heat flux densities, the heat rate to be evacuated is limited if compared with application analysed in this work. The effectiveness of this kind of technologies has been demonstrated in [Thome, 2010]. By comparing two-phase pumped systems with single-phase ones, they showed that, to maintain constant chip temperature, the use of two-phase technology allows sensitive pumping power saving and using a lower coolant mass flow rate. If, instead, the same pumping power than active single-phase technology is used, a temperature difference of 13K is realised in favour of two-phase one. Such solutions are currently designed to fit the thermal evacuation requirements of military aircrafts. In those cases, very high thermal power, up to hundreds of kilowatts, with peaks in the order of thousands of kilowatts, must be evacuated [Homitz,2010]. The American society ACT develops two-phase technologies to be used for civil and military applications. They ensure a lowering of overall system mass due, in particular, to the reduction of the mass flow rate and pumping power [Act,2019]. Even if the concept scheme for this kind of technology is similar to the one related to single-phase cooling systems, the operating principle is completely different. Here, latent heat is used to extract heat by hot source.

## WHISED STATE

Find a host to use as a prototype that allows you to host the accelerator card inside. This host must allow the cooling of as many components as possible through the two-phase mechanism developed by IN4. The new cooling technology, although has shown its effectiveness on single processor, needs extensive tests on a real HPC server with several FPGA and CPUs to demonstrate its potential. Another important objective is to provide an engineered solution at server and rack levels.

**WS-1.5.1** To demonstrate the potential energy-saving of the two-phase cooling technology compared to the traditional DLC by means an experimental test setup that consists to replace the conventional cooling system for some components, internal to the compute node (e.g. CPU, GPU or FPGA discrete or integrated). The conventional cooling system uses exclusively the flow of air provided by internal fans to keep electrical components within acceptable operating temperatures. The technology used in the project replaces the air cooling provided for high---heat generating components with cooling provided by a liquid in single-phase or vapour in two-phase transition. Therefore, two different compute nodes use the direct cooling technology with two heat removal solutions (single/two-phase) comparing the energy-saving performances. Estimates of the energy savings potential were performed by analyzing two scenarios: (1) a compute node retrofitted with DLC single-phase, and (2) a compute node with retrofitted DLC two-phase. Therefore the energy-savings cannot be measured, but must be a comparison of two estimates: the energy use for the DLC single-phase and the energy use for the DLC two-phase.

In the experimental test setup some parameters shall be varied, such as: supply water temperature, supply water flow rate (which affects the return water temperature), and computing power of the node. The performance associated with higher return water temperatures shall be used to investigate the possibilities of heat reuse. The computing power of the node shall be varied to investigate the effects of different levels of compute node utilization.

In the wished experimental setup in order to compare the energy-saving provided by two different direct cooling solutions at rack level, a layout shall be made deployed as in the Fig.1.5.1.6.

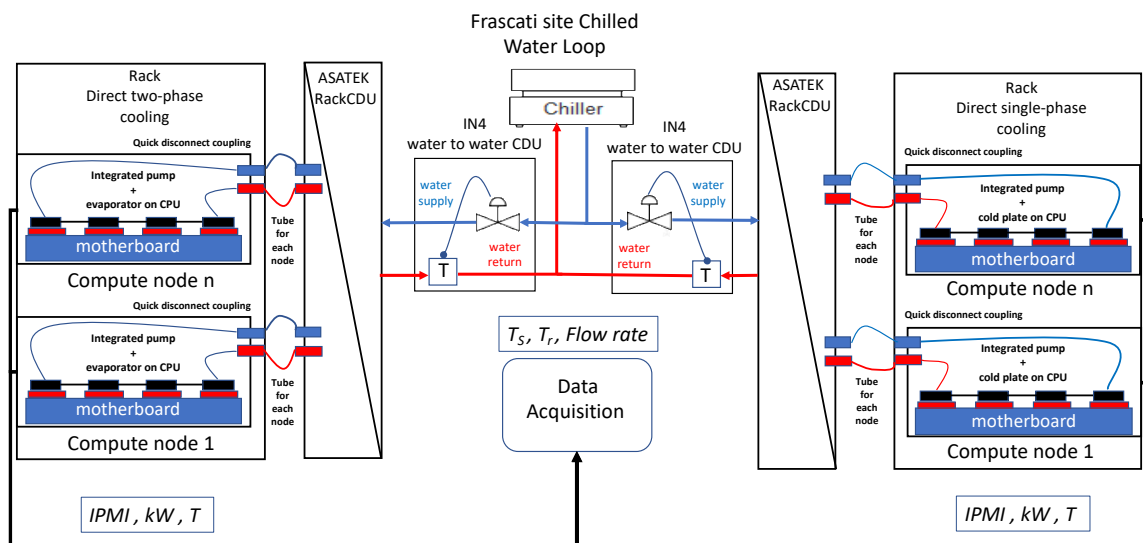


Fig.1.5.1.6 Experimental test setup schematic layout

The liquid going from the RackCDU to the servers is being provided by Asetek. This fluid is mostly water, but it also had anti-corrosion and freeze protection additives. Hereinafter, the liquid will be referred to simply as water. The parameters supply water temperature and return water temperature refer to the water flowing between a IN4 CDU and RackCDU indicated in Figure 1.5.1.6 as “cooling water supply/return.” The IN4 contains a valve controls, flow rate and temperature sensor used to adjust the supply water temperature. The Asetek direct cooling system controls the return water temperature to take advantage of potential heat reuse opportunities by adjusting the water flow rate. It should be noted that the return water temperature is an important parameter to vary, in order to understand the full range of thermal performance. The remaining parameter being varied is the computing power level. The computing power is set by running High Performance LINPACK (HPL) software on the compute nodes. Three power levels shall be used: Idle (no software applications running), 50 percent power (HPL operating at half the number of CPU cores), and Full or 100 percent power (HPL running all CPU cores).

The following parameter ranges shall be evaluated:

- Supply Water Temperature
- Water Flow Rate in order to test a range of return water temperatures

- Direct-Cooled IT Equipment Power
  - Idle
  - 50 Percent Power
  - Full or 100 Percent Power

Refer to Fig.1.5.1.6 the experimental test setup uses the following equipment:

- Compute node: at least 2 nodes in 2 distinct racks equipped with at least one GPU or FPGA discrete or integrated on the motherboard. Each node shall be equipped with direct single and two phase cooling systems. The two racks are equipped with Asetek RackCDU technology. The direct cooling shall be applied to all CPU/GPU/FPGA equipping the compute node. Each compute node provided an Intelligent Platform Management Interface (IPMI) where the front panel air-inlet temperature, fan speeds, and CPU component temperatures were provided and recorded as well as a data acquisition system for electrical energy consumption of the compute node.
- IN4 CDU: two systems provided by IN4 to supply variable water temperature and flow rates to Asetek RackCDU. The temperature set points is manually entered, and the water flow rate is manually controlled between both the building chilled water system and RackCDU as required for temperature stability and to adjust to the target RackCDU return temperature.
- Direct Cooling System: The Asetek RackCDU cooling system consists of two parts: (1) a rack-mounted CDU providing cooling water distribution to the compute node and water-o-water heat exchange between the node coolant and facilities water; and (2) cooling devices placed inside of the compute node for single phase direct cooling and external to the compute node for two phase direct cooling. These cooling devices contact temperature-sensitive components such as CPU/GPU/FPGA. For each compute node a set of flexible tubes provides a supply and return for cooling water going to the cooling devices inside the compute node.
- Power Consumption node: The electrical power of each compute node shall be acquired by means an external microcontroller with measures of precision at least of 1% and sampling rates at least of 1 second. The data acquired shall be collected by means json objects accessible via web services.
- Power Consumption rack: The electrical power of the two racks shall be acquired on the Power Distribution Units in the same way of the compute node.
- BTU meters for Asetek RackCDU and IN4 CDU for flow rates, temperatures of water on supply and return, pressure water supply for the two racks with precisions and sampling rates are being defined in the requirements and specifications deliverable.
- Compute node Thermal Stress Software: HPL benchmark software shall be used as a means of exercising the CPUs, GPU or FPGA so as to vary the power consumed, and therefore the heat generated by the compute node.

**ACTION STATE**

The actions need to achieve an assessment of the foregoing wished states during the activities of WP1 is simply listed in the following table:

	Design	Develop	Implementation	Test
WS-1.5.1	Experimental test setup	A single and two phase direct cooling	PoC on FPGA	Energy to solution

Within this action, the aim is to deploy an architectural solution FPGA technology based, discrete or SoC integrated, chosen for the development of the IDV-E prototype. This architectural solution must be able to cool through the two-phase system that will be developed in other WPs.

#### PRIORITY

The WS-1.5.1 shall be implemented and tested on IT equipment including the aforementioned on at least two compute nodes equipped at least two FPGA on each make available by partners.

#### REFERENCES

[Katevenis,2016]: M. Katevenis et al., *The ExaNeSt Project: Interconnects, Storage, and Packaging for Exascale Systems*, 2016 Euromicro Conference on Digital System Design (DSD), 2016, pp. 60-67, doi: 10.1109/DSD.2016.106.

[Mavroids,2016]: I. Mavroidis et al., *ECOSCALE: Reconfigurable computing and runtime system for future exascale systems*, 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016, pp. 696-701.

[Euroexa,2020]: <https://euroexa.eu>

[Bartolini,2018]: A.Bartolini, A.Borghesi, A.Libri, F.Beneventi, D.Gregori, S.Tinti, C.Gianfreda, and P.Altoè. 2018. *The D.A.V.I.D.E. big-data-powered fine-grain power and performance monitoring support*. In Proceedings of the 15th ACM International Conference on Computing Frontiers (CF '18). Association for Computing Machinery, New York, NY, USA, 303–308. <https://doi.org/10.1145/3203217.3205863>

[AMPERE,2021]: <https://amperecomputing.com/altra/>

[Gigabyte,2021]: <https://www.gigabyte.com/es/Enterprise/GPU-Server/G242-P32-rev-100>

[Coles,2014]: H.Coles and S.Greenberg. *Direct Liquid Cooling for Electronic Equipment*. Mar.2014. Orlando Lawrence Berkeley National Laboratory – Tech. rep (2014)

[Thome, 2010]: J.R.Thome, J.B. Marcinichen. *Refrigerated Cooling of Microprocessors with Micro-Evaporation New Novel Two-Phase Cooling Cycles: A Green Steady-State Simulation Code*. Proceedings of ENCIT 2010. 13<sup>th</sup> Brazilian Congress of Thermal Sciences and Engineering.

[Homitz,2010]: J. Homitz, R. Scaringe, G. Cole, *Evaluation of a vapor-compression thermal management system for reliability while operating under thermal transients*, Tech. rep., SAE Technical Paper (2010).

## 20 Task-1.5.2: GPU Platform

#### CURRENT STATE

ATOS has developed an entire rack infrastructure dedicated to HPC and based on its own liquid cooling technology. This infrastructure provides a common form factor for different types of compute blades. The compute nodes may be either CPU-only nodes, usually dual-socket nodes with the last generation of Intel, AMD or ARM high performant processors, or hybrid nodes with one or two CPU and up to four Nvidia or Intel GPU of last generation.

The CPU and GPU units are selected to achieve the best performance per watt. With air-cooling, the TDP (300 – 400W) would require very tall heatsinks and strong air flow. To ensure high density (up to 96 dual-socket CPU nodes in one rack), the rack is fully liquid-cooled (no air the compute or switch blades thus no fan). To achieve a power usage effectiveness (PUE) close to one, the temperature of the liquid in the customer (or primary) loop must be high enough to use external free cooling without additional energy to cool it down. Consequently, the current ATOS technology uses a primary loop temperature of up to 40°C and a rack heat exchanger to evacuate the heat from the secondary loop which cools the blades thanks to cold plates and water blocks. These features are implemented in Bull Sequana XH2000 (Fig.1.5.2.1) and used in Leonardo, ranked 4<sup>th</sup> in Top500 – November 2022 (Rmax at 174.70 PFlop/s and 5.61 MW of power) and 14<sup>th</sup> in Green500 – November 2022 (31.1 GFlop/W).

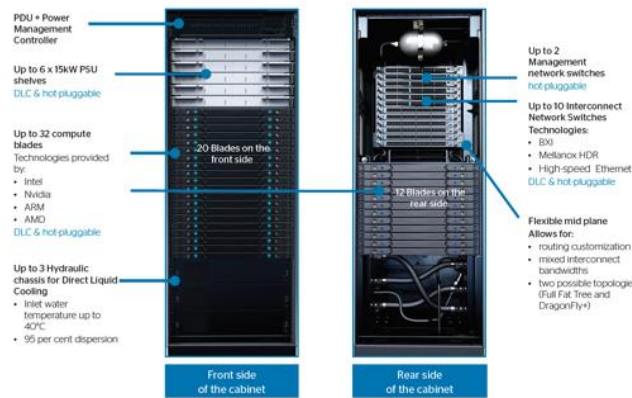


Fig.1.5.2.1: Bull Sequana XH2000



Fig.1.5.2.2a: Bull Sequana compute blade with three compute nodes

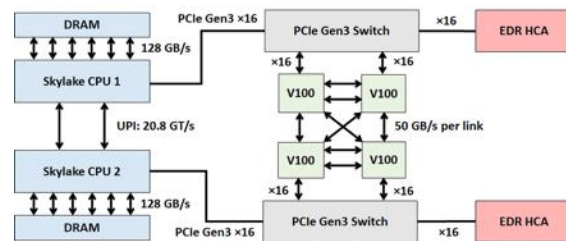


Fig.1.5.2.2b: block JUWELS booster Module Cluster compute nodes

Different types of compute blades are provided in Bull Sequana XH2000 [Bull Sequana, 2021]. The Fig.1.5.2.2a shows a compute blade with 3 dual-sockets equipped with DLC while the Fig.1.5.2.2b shows the block diagram of a compute blade with 2 CPU (dual-socket) and 4 GPU Nvidia V100 which compose the booster module of JUWELS at Julich Supercomputing Centre.

In Bull Sequana XH2000, cooling is provided by hydraulic modules located at the bottom of the rack and containing the following main components: a heat exchanger, a pump and regulation valves. The main function of the hydraulic modules is to maintain the temperature of the internal hydraulic circuit at a fixed regulation value. Liquid distribution is ensured inside each compute blade by a cold-

plate and DLC heat-spreaders which dissipate heat from CPU and GPU. Two manifolds (distribution and collection) connect the cold-plates to the rack main hydraulic circuit.

## WISHED STATE

The efficiency of the ATOS technology may become limited as the components power dissipation increases while the maximum case temperature  $T_{case}$  lowers. The current maximum dissipated power is 400W but continues to increase and might reach more than 700W in few years. Consequently, the dissipation of such component heat must become even more efficient. In parallel, the maximum temperature supported by such component (maximum  $T_{case}$ ) keeps decreasing and ensuring cooling with an inlet temperature of 40°C is getting more challenging. A few years ago, the maximum case temperature of bigger integrated circuits was around 90°C years ago while it is around 70°C in some components today. With the addition of HMB, this temperature could reach new minimums.

A minimum 40°C temperature at heat exchanger inlet is key, as in some hot European regions, free cooling would not be available all year long with a lower temperature. Meantime, with the state of Atos' current cooling technology, heat reuse is not optimized as the maximum secondary loop temperature cannot be higher than 45°C to meet the maximum  $T_{case}$  requirements. Decreasing the temperature gap between  $T_{case}$  and the maximum liquid temperature, which could reach up to 50°C, would enable better heat reuse and preserve the use of free cooling.

**WS-1.5.2:** A new generation is under development, to increase the power of the rack (up to 147 kW) and the cooling capacity. With this new generation, the blade interface characteristics will be shared under the name OpenSequana, for other HPC manufacturers to develop their specific blades and benefit from the rack infrastructure. The main characteristics wished for OpenSequana are:

- All-in-one design : Compute, Networking, Power, Cooling in one rack
- High density (~200 CPU per m<sup>2</sup>):
  - 1U blade with 3 dual-socket compute node or
  - 1U blade with single-socket accelerated compute node with 4 GPU
- Best in class PUE close to 1
  - Warm water cooling up to 40 °C inlet supply
  - No fan (DLC PSU shelves)
  - Capable of cooling suppliers highest TDP/Tcase units. Possible lower temperature inlet supply water for 350W CPU and 500W GPU (down 35 °C)
- High availability: Redundancy n+1 on PSU and Hydraulic Chassis (heat exchanger+pump)
- Easy maintenance (similar to standard air-cooled system):
  - All components are hot-swappable and can be serviced without interrupting system production
  - Processors and DIMMs can be replaced without removing the compute blade cold plate

## ACTION STATE

This task will define the characteristics of a GPU platform implementing the system architecture defined in Task 1.4 to run efficiently the user applications defined in Task 1.1. The criteria such as the number of CPU and GPU, the performance of these CPU and GPU, the capacity and bandwidth of the memory attached to the CPU, the bandwidth between CPU and GPU, the bandwidth to the highspeed interconnect network, will be analyzed taking into account the hardware feasibility.

This node must also be a good test vehicle for the two-phase cooling technology. It could consume from 3 kW to 4 kW DC and a solution with GPU in standard OAM form factor would be preferred. The selection of one OpenSequana GPU blade in the Atos portfolio with the last generations of CPU and GPU, and the adaptations will be done in WP3 and WP5.

	Design	Develop	Implementation	Test
WS-1.5.2	OpenSequana GPU node	A single and two phase direct cooling	PoC on GPU	Energy to solution

### PRIORITY

Define the requirements of this GPU node as it is the initial step before the selection and adaptation of the blade.

### REFERENCES

[*BullSequana,2021*]:

[https://atos.net/wp-content/uploads/2020/07/BullSequanaXH2000\\_Features\\_Atos\\_supercomputers.pdf](https://atos.net/wp-content/uploads/2020/07/BullSequanaXH2000_Features_Atos_supercomputers.pdf)

[https://atos.net/wp-](https://atos.net/wp-content/uploads/2020/07/BullSequanaXH2000_Features_Atos_supercomputers.pdf)

## 21 Task-1.5.3: Power and Thermal Management

### CURRENT STATE

One of the challenges in the race towards exascale computing is the steadily increasing power density of high-performance computing platforms, a fact that can be observed in CPUs, GPUs and FPGAs alike. As an example, recent CPUs reached an impressive TDP of 400W [*Intel-Xeon-9282*].

Thermal management is thus increasingly becoming a vital part of the optimization process of an HPC infrastructure. Insufficient cooling could lead to reduced computing performance due to the need to throttle operating frequencies for extended periods of time, and may lead to reliability issues including increased fault rates. On the other hand, energy efficiency concerns demand efficient cooling technologies capable of reducing cooling energy requirements in order to meet the exascale challenge [*Zhabelova,2018*] [*Iranfar,2019*].

On a more detailed level, the power consumed by computing devices is highly variable for reasons including different code execution patterns, cache miss patterns and time-varying use of computational resources. These high and frequent power variations, coupled with the small thermal capacitance of the active silicon layer and the non-negligible thermal resistance towards the heat dissipation stack give rise to fast thermal transients causing the temperature of active silicon to vary



by tens of degrees in tens of milliseconds [Terraneo,2019] requiring control at a timescale that cooling systems are structurally incapable of meeting, evidencing a strong need for innovative cooling solutions, thermal modeling and control methodologies.

For what concerns thermal modeling, in the current state-of-the-art we can find the 3D-ICE 3.0 extensible thermal simulator [Terraneo,2021]. While this thermal simulator is currently capable of simulating air- and water-cooled scenarios, at present it lacks support for two-phase cooling.

For what concerns innovative cooling solutions, two-phase cooling technology has been explored in the state-of-the-art in its gravity-driven thermosiphon configuration for increased heat dissipation capability and energy efficiency of computing units [Iranfar,2019]. Alternative solutions, such pump-driven two-phase cooling received far less attention despite its potential to deliver increased heat removal capacity.

For what concerns thermal control policies, a prominent solution in the state-of-the-art is the use of event-based control resulting in thermal policies with a low computational overhead [Leva,2018]. This approach has been verified on actual hardware using DVFS, but its extension to handle hierarchical scenarios as well as the control of two-phase cooling solutions remains to date an open challenge.

**WISHED STATE**

The current scenario outlined by the state of the art evidences the strong need to move in two complementary directions:

**WS-1.5.3a:** To bring efficient cooling as near as possible to the active silicon. The use of pump-driven two-phase cooling solutions is expected to bring both higher cooling capacity to the advantage of increased computing performance due to the possibility to keep computing units at an average higher operating frequency, to increase reliability due to a lowering of operating temperatures, as well as to improve energy efficiency.

**WS-1.5.3b:** To employ hierarchical thermal control policies to overcome the problem posed by fast thermal dynamics using fast actuators such as DVFS, while increasing the cooling capacity at a timescale compatible with cooling systems in order to optimize operating frequencies to the advantage of both computing power and power efficiency.

**ACTION STATE**

To address the evidenced research directions our main activities will be devoted to:

- Designing thermal models for pump-driven two-phase cooling solutions, using validation data provided by the prototype developed by IN4.
- Designing hierarchical thermal control policies taking advantage of pump-driven two-phase cooling solution.

	Design & Development	Implementation	Test
WS-1.5.3a	thermal models for pump-driven two-phase cooling solutions	PoC on one IDV	Thermal model validation
WS-1.5.3b	thermal control policies models for pump-driven two-phase cooling solutions	PoC on one IDV	Control policies

**REFERENCES**

- [Intel-Xeon-9282]: <https://ark.intel.com/content/www/it/it/ark/products/194146/intel-xeon-platinum-9282-processor-77m-cache-2-60-ghz.html>
- [Zhabelova,2018]: Gulnara Zhabelova, Mattias Vesterlund, Sascha Eschmann, Yulia Berezovskaya, Valeriy Vyatkin, Damien Flieller. *A Comprehensive Model of Data Center: From CPU to Cooling Tower*. IEEE Access, <https://doi.org/10.1109/ACCESS.2018.2875623>
- [Iranfar,2019]: Arman Iranfar, Federico Terraneo, Gabor Csordas, Marina Zapater, William Fornaciari and David Atienza. *Dynamic thermal management with proactive fan speed control through reinforcement learning*. In: Design, Automation and Test in Europe Conference (DATE). 2020. doi:10.23919/DATE48585.2020.9116510
- [Terraneo,2019]: Terraneo Federico, Alberto Leva, William Fornaciari. *Event-Based Thermal Control for High Power Density Microprocessors*. In: *Harnessing Performance Variability in Embedded and High-performance Many/Multi-core Platforms: A Cross-layer Approach*. Springer International Publishing, 2019. isbn: 978-3-319-91962-1. doi: 10.1007/978-3-319-91962-1\_5.
- [Terraneo,2021]: Federico Terraneo, Alberto Leva, William Fornaciari, Marina Zapater, David Atienza. *3D-ICE 3.0: efficient nonlinear MPSoC thermal simulation with pluggable heat sink models*. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2021)*. doi: 10.1109/TCAD.2021.307461.
- [Leva,2018]: Alberto Leva, Federico Terraneo, Irene Giacomello, William Fornaciari. *Event-based power/performance-aware thermal management for high-density microprocessors*. In: *IEEE Transactions on Control Systems Technology (2017)*. doi: 10.1109/TCST.2017.2675841.

## 22 Summary of Actions

Here we report the design and development actions needed to reach the identified wished states for any of the co-design tasks.

### TASK-1.1: User Applications

<b>T-1.1.1</b>		<b>New algorithm and software libraries</b>	<b>INRIA, UBx, CNR, PSNC, ENEA</b>	
	Design		Develop/implement	Test
WS.1.1.1a	Data movement in Krylov methods and multiple precisions in basic sparse matrix operations		PoC of new algorithms in GPU and/or FPGA	test to evaluate -Time to solution -Energy to solution
WS.1.1.1b	Task based model for dense linear algebra and FMM kernels on FPGA		PoC Chamelon solver and Heteroprio on FPGA	test to evaluate -Time to solution -Energy to solution

<b>T-1.1.2</b>		<b>AI and HPDA</b>	<b>INRIA, CINI, INFN, PSNC</b>	
	Design		Develop/implement	Test
WS.1.1.2a	Integration INFN interconnection IPs for FPGA Xilinx and stack software		PoC of NNs in FPGA Xilinx with high level programming model offering abstraction for communications	test to evaluate -Processing throughput -Energy to solution
WS.1.1.2b	Neuronal simulation test case		PoC of Nest and NeuronGPU in GPU	test to evaluate -Time to solution -Energy to solution
WS.1.1.2c	Allocation algorithms		PoC NNs over StarPU in FPGA	test to evaluate -Time to solution -Energy to solution

<b>T-1.1.3</b>		<b>Scientific flagship codes</b>	<b>FHG, INRIA, UBx, ENEA, INFN, PSNC</b>		
	Design		Develop	Implementation	Test
WS-1.1.3a	Low level tensor operations		Kernel functions	PoC on GPU	Time to solution Energy to solution
WS-1.1.3b	HEP computing algorithms		Kernel Functions in SYCL/Alpaka	PoC on GPU	Time to solution Energy to solution
WS-1.1.3c	EULAG computing algorithms		Kernel functions	PoC on GPU	Time to solution Energy to solution

### TASK-1.2: Runtime Services

<b>T-1.2.1</b>		<b>Resources Management</b>		<b>ENEA</b>		
	Analysis	Design		Develop	Implementation	Test
WS-1.2.1a	Virtualize solutions for GPUs/FPGAs	virtualize environment based on OpenStack		Virtualize GPU/FPGA devices	PoC on GPU/FPGA	Performance -Throughput -Latency
WS-1.2.1b	LRM	LRM with power/thermal data acquisition		Monitoring	PoC on GPU/FPGA	
WS-1.2.1c	GRM/SLURM	A plugin for SLURM to store external data		Integrate LRM with SLURM	PoC on GPU/FPGA	
WS1.2.1d				Integrate OmpSs/StarPU in SLURM	PoC on GPU/FPGA	

### TASK-1.3: Programming Models

T-1.3.1	Streaming Model	CINI	
	Develop	Implement	Test
WS-1.3.1	FPGA operator interface generation/operation in FastFlow	compiling high-level code to FPGA (configuration) code suitable to be operated via its OpenCL interface, according to the defined methodology	Experimental ad hoc code to expose problems and solutions

T-1.3.2	Tasks Model	CINI, BSC, INRIA, UBx	
	Develop	Implement	Test
WS-1.3.2a	OmpSs@FPGA on FPGA(Xilinx Alveo)	PoC using VITIS	Performance <ul style="list-style-type: none"> <li>- Time to solution</li> <li>- Power consumption</li> <li>- Programmability</li> </ul>
WS-1.3.2b	StarPU on FPGA(Xilinx Alveo)	PoC using VITIS	Performance <ul style="list-style-type: none"> <li>- Time to solution</li> </ul>

T-1.3.3	High Level Synthesis Flow	ENEA	
	Develop	Implement	Test
WS-1.3.3a	Unified Vitis based programming environments for: <ul style="list-style-type: none"> <li>- FastFlow and OmpSS</li> </ul>	Image Processing and lossless compression on IDV-E	Evaluate performance in terms of Energy Delay Product
WS-1.3.3b	Inter-FPGA HW/SW communication infrastructure	A HW communication layer that allows seamless communication among kernels assigned to different FPGAs and the realization of the corresponding SW stack that should be integrated within the unified programming environment.	Specific tests to measure network performance with HLS Kernels as communication endpoints: <ul style="list-style-type: none"> <li>- Comm. Latency</li> <li>-Comm. Bandwidth</li> </ul>
WS-1.3.3c	Multi-precision arithmetic		Performance: <ul style="list-style-type: none"> <li>- Time to solution</li> <li>- Energy to solution</li> </ul>

T-1.3.4	Mixed precision for new accelerators	CINI UNIPI-POLIMI	
	Develop	Implement	Test
WS-1.3.4a	Posit accelerator IPs (see specification in Section 2 of D2.1)	DNN computation with posit on FPGA	Performances on mini-apps: <ul style="list-style-type: none"> <li>-Same accuracy of FP</li> <li>- Time to solution</li> <li>- Energy to solution</li> </ul>
WS-1.3.4b	TAFFO extensions to support heterogeneous HPC nodes	TAFFO support for HLS TAFFO support for heterogeneous parallel accelerators (OpenMP, OpenCL, CUDA)	Performance on benchmarks and mini-apps : error below 3% w.r.t. floating point, performance improvement over baseline

T-1.3.5	Secure services for HPC	CINI	
	Analysis	Implement	Test
WS-1.3.5a	Number Theoretic Transform (NTT) computational kernel of SEAL homomorphic encryption library	Hardware IP for NTT in FPGA technology	Performances: <ul style="list-style-type: none"> <li>-Computational complexity</li> <li>-Throughput/Latency</li> </ul>
WS-1.3.5b	Post-Quantumr Digital Signature	Hardware IP for SHAKE128/256 eXtendable Output Function (XOF) in FPGA	Performances: <ul style="list-style-type: none"> <li>- Computational complexity</li> <li>-Throughput/Latency</li> </ul>

### Task-1.4: System Architecture

<b>T-1.4.1</b>		<b>Heterogeneous Architectures</b>	<b>E4</b>
		Design	Test
WS.1.4.1.a	Requirements and Specifications of compute node with a system architecture CPU+GPU based		Evaluation of KPIs of TEXTAROSSA applications and use cases using GPUs: - Mathlib-CNR - Mathlib-INRIA - UrbanAir - NestGPU -HEP -TNM -Smart Cities -DNN Inference
WS.1.4.1.b	Requirements and Specifications of compute node with a system architecture CPU+FPGA based		Evaluation of KPIs of TEXTAROSSA applications and use cases using FPGAs: -RAIDER - Mathlib-INRIA

<b>T-1.4.2</b>		<b>Interconnection Networks</b>	<b>INFN</b>	
		Design	Development	Test
WS.1.4.2.a	HLS based design intra-node switch and serial link control		PoC on FPGA	Dedicate functional tests with HLS kernels as communication endpoints measuring: -Communication Latency -Communication Bandwidth
WS.1.4.2.b	Linux driver to set the network		PoC on server equipped with FPGA	Dedicated functional tests
WS.1.4.2.c	User Space Library to configure the network		PoC on server equipped with FPGA	Dedicated functional tests

### TASK-1.5: Hardware Platforms

<b>T-1.5.1</b>		<b>FPGA platform</b>	<b>E4, InQuattro, ENEA</b>	
		Design	Develop	Implementation      Test
WS-1.5.1	Experimental test setup	A single and two phase direct cooling	PoC on FPGA	Energy to solution

<b>T-1.5.2</b>		<b>GPU platform</b>	<b>ATOS</b>	
		Design	Develop	Implementation      Test
WS-1.5.2	OpenSequana GPU node	A single and two phase direct cooling	PoC on GPU	Energy to solution

<b>T-1.5.3</b>		<b>Power and Thermal Management</b>	<b>CINI-POLIMI , PSNC</b>	
		Design & Development	Implementation	Test
WS-1.5.3a	thermal models for pump-driven two-phase cooling solutions		PoC on FPGA platform	Thermal model validation
WS-1.5.3b	thermal control policies models for pump-driven two-phase cooling solutions		PoC on FPGA platform	Control policies

## 23 Conclusions

We described the methodology adopted for performing the gap analysis along the five dimensions of TEXTAROSSA macro-activities (tasks):

- i) User Applications,
- ii) Runtime Services,
- iii) Programming Models,
- iv) System Architectures,
- v) Hardware Platforms.

For any sub-task, after providing an overview of its state-of-the-art, we identified a set of possible improvements with respect to TEXTAROSSA objectives (wished states), along with the associated co-design actions to be carried out in the project and eventually providing a priority between them.

Results for the gap analysis are detailed in the corresponding sections and, for the reader's convenience, summarized in section 22 – Summary of Actions. They show that the orchestration of different tasks is mandatory to reach the goal of a high performance and energy efficient HPC node: from the development of new numerical methods capable of exploiting to the maximum extent the computing capabilities offered by accelerators to the improvement of key enabling IPs (e.g. advanced cooling techniques) and programming models allowing the seamless integration of heterogeneous accelerators in the development workflow.

## Annex I: Xilinx FPGA product tables

	Feature	Alveo U200	Alveo U250	Alveo U280	Alveo U50
Dimensions	Width	Dual Slot	Dual Slot	Dual Slot	Single Slot
	Form Factor, Passive Form Factor, Active	Full Height, ¼ Length Full Height, Full Length	Full Height, ¼ Length Full Height, Full Length	Full Height, ¼ Length Full Height, Full Length	Half Height, ½ Length
Logic Resources <sup>1</sup>	Look-Up Tables	1,182K	1,728K	1,304K	872K
	Registers	2,364K	3,456K	2,607K	1,743K
	DSP Slices	6,840	12,288	9,024	5,952
DRAM Memory	DDR Format	4x 16GB 72b DIMM DDR4	4x 16GB 72b DIMM DDR4	2x 16GB 72b DIMM DDR4	–
	DDR Total Capacity	64GB	64GB	32GB	–
	DDR Max Data Rate	2400MT/s	2400MT/s	2400MT/s	–
	DDR Total Bandwidth	77GB/s	77GB/s	38GB/s	–
	HBM2 Total Capacity	–	–	8GB	8GB
	HBM2 Total Bandwidth	–	–	460GB/s	316GB/s <sup>4</sup>
Internal SHA1	Total Capacity	43MB	57MB	43MB	28MB
	Total Bandwidth	37TB/s	47TB/s	35TB/s	24TB/s
Interfaces	PCI Express®	Gen3 x16	Gen3 x16	Gen3 x16, 2xGen4 x8, CCIX	Gen3 x16, 2xGen4 x8, CCIX
	Network Interface	2x QSFP28	2x QSFP28	2x QSFP28	U50 <sup>2</sup> - 1x QSFP28 U50DD <sup>3</sup> - 2x SFP-DD
Power and Thermal	Thermal Cooling	Passive, Active	Passive, Active	Passive, Active	Passive
	Typical Power	100W	110W	100W	50W
	Maximum Power	225W	225W	225W	75W
Time Stamp	Clock Precision	–	–	–	IEEE Std 1588
Tool Support	Vitis™ Developer Environment	Yes	Yes	Yes	Yes
Solutions	Solutions & Libraries	<a href="#">Acceleration Application Libraries and Solutions</a>			

Alveo™ Data Center Accelerator Cards

Notes

- Logic resources shown without platform usage; refer to card user guides for platform resource usage.
- U50 is a production qualified card for volume deployment.
- U50DD is an engineering sample card not for volume production.
- A-U50DD-P00G-ESS3-G and A-U50-P00G-PQ-G measured 316GB/s peak HBM2 bandwidth, 201 GB/s nominal.



	Feature	Alveo U25	Alveo SN1022
Dimensions	Width	Single Slot	Single Slot
	Form Factor	Half Height, ½ Length	Full Height, ½ Length
Logic Resources	Look-Up Tables	523K	1,030K
	Registers	1,045K	2,059K
DRAM Memory	DDR Format	- 1x 2GB x 40 DDR4-2400 - 1x 4GB x 72 DDR4-2400	- 1x 4GB x 72 DDR4-2400 (Arm® Processor) - 2x 4GB x 72 DDR4-2400 (FPGA)
Interfaces	PCI Express®	Gen3 x16, 2xGen3 x8	Gen 3 x16, Gen 4 x8
	Link Speeds	10/25GbE	100GbE
	Network Interface	2x SFP28	2x QSFP28
	Arm Processor	Integrated Quad-core Cortex®-A53 Arm Processor	Discrete 16-core Cortex-A72 Processor
Power and Thermal	Thermal Cooling	Passive	Passive
	Thermal Design Power	40W	70W
	Total Power	75W	75W
Networking	Stateless Offloads	Yes	Yes
	Tunneling Offloads	VXLAN, NVGRE, Geneve, Custom	VXLAN, NVGRE, Custom
	SR-IOV	Yes	Yes
	Advanced Packet Filtering	Yes	Yes
	Acceleration / Offloads	DPDK, Onload®	DPDK, Onload®, Open Virtual Switch (OVS), Virtio-net, Virtio-blk, vDPA, Ceph RBD Client offload
	Manageability	PMCI Protocols	NC-SI, PLDM Monitoring and Control, PLDM MCTP
PMCI Transports		MCTP SMBus, MCTP PCIe VDM	MCTP SMBus, MCTP PCIe VDM
Boot Support		PXE and UEFI	UEFI
Software Plugins	Software and FPGA Extensibility via Dynamically Loadable Plugins	No	Yes
Tool Support	Vitis™ Developer Environment	Yes	Yes

Alveo™ SmartNIC Data Center Accelerator Cards

Notes:

- Logic resources shown without platform usage; refer to card user guides for platform resource usage.





Feature		Alveo U30 <sup>1</sup>
Dimensions	Width	Single Slot
	Form Factor, Passive Form Factor, Active	Half Height, ½ Length
Logic Resources <sup>2</sup>	Look-Up Tables	460K
	Registers	920K
	DSP Slices	3,456
Video support	Video Codec Unit (VCU)	Hardened
	Codec	H.264 and H.265
	Video Transcodes per Card	2 x 4kp60, 8 x 1080p60, 16 x 1080p30, 32 x 720p30
DRAM Memory	DDR Format	2 x 4GB 72b DDR4
	DDR Total Capacity	8GB
	DDR Max Data Rate	2400MT/s
	DDR Total Bandwidth	38GB/s
Internal SRAM	Total Capacity	11MB
	Total Bandwidth	10TB/s
Internal Interfaces	PCI Express®	Gen3 x8, 2 x Gen3 x4 <sup>3</sup>
	Network Interface	-
Power and Thermal	Thermal Cooling	Passive
	Typical Power	40W
	Maximum Power	75W
Tool Support	Vivado® Design Suite	No
	FFmpeg with Xilinx Video Plugins	Yes
	Vitis™ Developer Environment	No
Solutions	Solutions & Libraries	<a href="#">Video Transcoding Solutions</a>

Alveo™ Media Data Center Accelerator Cards

Notes

1. The Alveo U30 Media Accelerator Card is designed for use in Xilinx real-time evaluation servers or turn-key high-density video transcoding appliances offered by Xilinx VARs.
2. Logic resources shown are without platform usage. User does not have access to these resources on the Alveo U30 card.
3. Servers need to support bifurcation for users to take advantage of both Zynq® UltraScale+™ devices on the Alveo U30 card.





Annex II: Intel FPGA product tables

# Intel® Agilex™ F-Series FPGA and SoC FPGA Product Table



PRODUCT LINE	AGF 006	AGF 008	AGF 012	AGF 014	AGF 019	AGF 023	AGF 022	AGF 027
Logic elements (LEs)	573,480	764,640	1,178,525	1,437,240	1,918,975	2,308,080	2,208,075	2,692,760
Adaptive logic modules (ALMs)	194,400	259,200	399,500	487,200	650,500	782,400	748,500	912,800
ALM registers	777,600	1,036,800	1,598,000	1,948,800	2,602,000	3,129,600	2,994,000	3,651,200
High-performance crypto blocks	0	0	0	0	2	2	0	0
eSRAM memory blocks	0	0	2	2	1	1	0	0
eSRAM memory size (Mb)	0	0	36	36	18	18	0	0
M20K memory blocks	2,844	3,792	5,900	7,110	8,500	10,464	10,900	13,272
M20K memory size (Mb)	56	74	115	139	166	204	212	259
MLAB memory count	9,720	12,960	19,975	24,360	32,525	39,120	37,425	45,640
MLAB memory size (Mb)	6	8	12	15	20	24	23	28
Fabric PLL	6	6	8	8	5	5	12	12
I/O PLL	12	12	16	16	10	10	16	16
Variable-precision digital signal processing (DSP) blocks	1,640	2,296	3,743	4,510	1,354	1,640	6,250	8,528
18 x 19 multipliers	3,280	4,592	7,486	9,020	2,708	3,280	12,500	17,056
Single-precision or half-precision floating point operations per second (TFLOPS)	2.5 / 5.0	3.5 / 6.9	6.0 / 12.0	6.8 / 13.6	2.0 / 4.0	2.5 / 5.0	9.4 / 18.8	12.8 / 25.6
Maximum EMIF x72 <sup>2</sup>	4	4	4	4	3	3	4	4
Maximum differential (RX or TX) pairs	192	288	384	384	240	240	384	384
AIB interfaces	2	2	2	2	4	4	4	4
Memory devices supported	DDR4 and QDR IV							
Secure data manager	AES-256/SHA-256 bitstream encryption/authentication, physically unclonable function (PUF), ECDSA 256/384 boot code authentication, side channel attack protection							
Hard processor system	Quad-core 64 bit Arm Cortex-A53 up to 1.50 GHz with 32 KB I/D cache, NEON coprocessor, 1 MB L2 cache, direct memory access (DMA), system memory management unit, cache coherency unit, hard memory controllers, USB 2.0x2, 1G EMAC x3, UART x2, serial peripheral interface (SPI) x4, I2C x5, general purpose timers x7, watchdog timer x4							
F-Tile	PCI Express (PCIe) hard IP block (Gen4 x16) or Bifurcateable 2x PCIe Gen4 x8 (EP) or 4x Gen4 x4 (RP) Transceiver channel count : 16 channels at 32 Gbps (NRZ) / 12 channels at 58 Gbps (PAM4) - RS & KP FEC Advanced networking support: - Bifurcateable 400 GbE hard IP block (10/25/50/100/200/400 GbE FEC/PCS/MAC) - Bifurcateable 200 GbE hard IP block (10/25/50/100/200 Gbps FEC/PCS) IEEE 1588 v2 support PMA direct							
E-Tile	Transceiver channel count : Up to 24 channels at 28.9 Gbps (NRZ) / 12 channels at 57.8 Gbps (PAM4) - RS & KP FEC <sup>1</sup> Networking support : - 400GbE (4 x 100GbE hard IP blocks (10/25 GbE FEC/PCS/MAC)) IEEE 1588 v2 support PMA direct							
P-Tile	PCIe hard IP block (Gen4 x16) or Bifurcateable 2x PCIe Gen4 x8 (EP) or 4x Gen4 x4 (RP) SR-IOV 8PF / 2kVF VirtIO support Scalable IOV							

<sup>1</sup> Only 4 instances of KP-FEC are supported when using 100GE MAC  
<sup>2</sup> Max EMIF count achieved using AVST x8 mode Compact - Address/Cmd lane [3 lanes] configuration

PRODUCT LINE	AGF 006	AGF 008	AGF 012	AGF 014	AGF 019	AGF 023	AGF 022	AGF 027
<b>F-Tile - Package Options and I/O Pins</b>								
GPIO (LVDS) / F-Tile 32G NRZ (58G PAM4)								
1546A (37.5 mm x 34 mm, 0.92 mm Hex)	384(192)/32(24)	384(192)/32(24)						
2340A (45 mm x 42 mm, 0.92 mm Hex)	576(288)/32(24)	576(288)/32(24)	744(372)/32(24)	744(372)/32(24)	480(240)/32(24)	480(240)/32(24)	744(372)/32(24)	744(372)/32(24)
3184C (56 mm x 45 mm, 0.92 mm Hex)					480(240)/64(48)	480(240)/64(48)	720(360)/64(48)	720(360)/64(48)
<b>E-Tile and P-Tile - Package Options and I/O Pins</b>								
GPIO (LVDS) / E-Tile 28.9G NRZ (57.8G PAM4) / P-Tile 16G PCIe								
2486A (55 mm x 42.5 mm, 1.0 mm Hex)			768(384)/16(8)/16	768(384)/16(8)/16				
2581A (52.5 mm x 40.5 mm, 0.92/0.94 mm Hex) <sup>2</sup>					480(240)/24(12)/32	480(240)/24(12)/32	624(312)/24(12)/32	624(312)/24(12)/32

<sup>2</sup> Conditional migration path from AGF 019/023 to AGF 022/027 devices

# Intel® Agilex™ I-Series FPGA and SoC FPGA Product Table



PRODUCT LINE	AGI 019	AGI 023	AGI 022	AGI 027	AGI 035	AGI 040
Logic elements (LEs)	1,918,975	2,308,080	2,208,075	2,692,760	3,540,000	4,047,400
Adaptive logic modules (ALMs)	650,500	782,400	748,500	912,800	1,200,000	1,372,000
ALM registers	2,602,000	3,129,600	2,994,000	3,651,200	4,800,000	5,488,000
High-performance crypto blocks	2	2	0	0	4	4
eSRAM memory blocks	1	1	0	0	3	3
eSRAM memory size (Mb)	18	18	0	0	54	54
M20K memory blocks	8,500	10,464	10,900	13,272	14,931	19,908
M20K memory size (Mb)	166	204	212	259	292	389
MLAB memory count	32,525	39,120	37,425	45,640	60,000	68,600
MLAB memory size (Mb)	20	24	23	28	37	42
Fabric PLL	5	5	12	12	6	6
I/O PLL	10	10	16	16	12	12
Variable-precision digital signal processing (DSP) blocks	1,354	1,640	6,250	8,528	9,594	12,792
18 x 19 multipliers	2,708	3,280	12,500	17,056	19,188	25,584
Single-precision or half-precision tera floating point operations per second (TFLOPS)	2.4 / 4.9	2.4 / 4.9	9.4 / 18.8	12.8 / 25.6	14.3 / 28.7	19.1 / 38.3
Maximum EMIF x72 <sup>1</sup>	3	3	4	4	4	4
Maximum differential (RX or TX) pairs	240	240	360	360	576	576
AIB interfaces	4	4	4	4	6	6
Memory devices supported	DDR4 and QDR IV					
Secure data manager	AES-256/SHA-256 bitstream encryption or authentication, physically unclonable function (PUF), ECDSA 256/384 boot code authentication, side channel attack protection					
Hard processor system	Quad-core 64 bit Arm Cortex-A53 up to 1.50 GHz with 32 KB I/D cache, NEON coprocessor, 1 MB L2 cache, direct memory access (DMA), system memory management unit, cache coherency unit, hard memory controllers, USB 2.0x3, 1G EMAC x3, UART x2, serial peripheral interface (SPI) x4, I2C x5, general purpose timers x7, watchdog timer x4					n/a
F-Tile	PCI Express(PCIe) hard IP block (Gen4 x16 ) or Bifurcateable 2x PCIe Gen4 x8 (EP) or 4x Gen4 x4 (RP) Transceiver channel count : - 4 channels at 116 Gbps (PAM4) / 58 Gbps (NRZ) - 16 channels at 32 Gbps (NRZ) / 12 channels at 58 Gbps (PAM4) - RS & KP FEC Advanced networking support: - Bifurcateable 400 GbE hard IP block (10/25/50/100/200/400 GbE FEC/PCS/MAC) - Bifurcateable 200 Gb hard IP block (10/25/50/100/200 GbE FEC/PCS) IEEE 1588 support PMA direct					
R-Tile	Compute Express Interface (CXL) - Link width x16 lanes, x8 lanes PCIe hard IP block (Gen5 x16) or Bifurcateable 2x PCIe Gen5 x8 (EP) or 4x Gen5 x4 (RP) Virtualization (SR-IOV) supporting 8 PFs/2k VFes Scalable IOV VirtIO support Precise time management PIPE direct					

<sup>1</sup> Max EMIF count achieved using AVST x8 mode Compact - Address/Cmd lane [3 lanes] configuration

## Annex III: Intel Agilex I-Series FPGA product tables

PRODUCT LINE	AGI 019	AGI 023	AGI 022	AGI 027	AGI 035	AGI 040
<b>F-Tile - Package Options and I/O Pins</b>	<b>GPIO (LVDS) / F-Tile 32G NRZ(58G PAM4) / High-Speed Transceiver 58G NRZ (116G PAM4) Channels</b>					
3184B (56 mm x 45 mm, 0.92 mm Hex)	F-Tile x4 480(240) / 64(48) / 8(8)	480(240) / 64(48) / 8(8)	720(360) / 64(48) / 8(8)	720(360) / 64(48) / 8(8)		
3948A (56mm x 56mm, 0.92 mm Hex)					576(288) 96(72) / 24(24)	576(288) 96(72) / 24(24)
<b>F-Tile and R-Tile - Package Options and I/O Pins</b>	<b>GPIO (LVDS) / F-Tile 32G NRZ(58G PAM4) / High-Speed Transceiver 58G NRZ (116G PAM4) Channels / R-Tile 32G PCIe (CXL) lanes</b>					
2957A (56 mm x 45 mm, 1.0 / 0.92 mm Hex)	F-Tile x1 & R-Tile x3		720(360)/16(12)/4(4)/48(32)	720(360)/16(12)/4(4)/48(32)		
3184A (56 mm x 45 mm, 0.92 mm Hex)	F-Tile x3 & R-Tile x1		720(360)/48(36)/8(8)/16(16)	720(360)/48(36)/8(8)/16(16)		
1805A (42.5mm x 42.5mm, 1.025 mm Hex)	F-Tile x1 & R-Tile x1	480(240)/16(12)/0(0)/16(0)	480(240)/16(12)/0(0)/16(0)			