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



WP1 Specifications, Co-design & Benchmarking

D1.2 Requirements & Specifications

Part I

WP-1.5/T1.5.1: IDV-E FPGA Platform

WP-1.5/T1.5.2: IDV-A GPU Platform



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: D1.2 Requirements & Specifications

Part I – T.1.5.1/2

WP-1.5: Hardware Platforms - T1.5.1: IDV-E FPGA Platform

WP-1.5: Hardware Platforms - T1.5.2: IDV-A GPU Platform

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.

Deliverable No	D1.2 – Part I - T.1.5.1-2
WP No:	WP1
WP Leader:	ENEA
Due date:	M6 (November 30, 2021)
Delivery date:	31/05/2022

Dissemination Level:

PU	Public	х
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)	









DOCUMENT SUMMARY INFORMATION

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

WP1 Specifications, Co-design & Benchmarking

Deliverable number:	D1.2					
Deliverable title:	Requiremen	nts & Specific	ations – Part	I – T.1.5.1-2		
Due date:	M6					
Actual submission date:	31/05/2021					
Editor:	Francesco lannone					
Authors:	List of Authors					
Work package:	WP1					
Dissemination Level:	Public					
No. pages:						
Authorized (date):	31/03/2022					
Responsible person:	Francesco lannone and Berenger Bramas					
Status:	Plan	Draft	Working	Final	Submitted	Approved

Revision history:

Version	Date	Author	Comment
0.1	15/05/2022	F. lannone B. Bramas	Draft structure
0.2			

Quality Control:

Checking process	Who	Date
Checked by internal reviewer	Project Technical Committee	
Checked by Task Leader		
Checked by WP Leader	Francesco lannone	
Checked by Project Coordinator	Massimo Celino	





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 <u>http://textarossa.eu</u> 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-UNIPI); 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

Executive Summary	5
Partner Report Activity	6
List of Authors	6
List of Acronyms	6
1 Introduction	9
2 FPGA platform requirements	
2.1 FPGA system architecture	
2.2 IDV-E Target Platform	
2.2.1 IDV-E platform requirements	20
3 GPU platform requirements	27
3.1 GPU system architecture	
3.2 IDV-A Target Platform	
3.2.1 IDV-A platform requirements	

Executive Summary

This report shows the co-design process within the WP1 of Textarossa project to define requirements and specifications of IDV-E and IDV-A respectively for FPGA and GPU compute node based. As various hardware platforms with different characteristics and architectures are hitting the market, we are considering technological solutions already available at level of compute node using reconfigurable hardware integrated in the programming models more widespread able to support users' applications.

This deliverable has the subtitle *Part I T1.5.1-2*, because it'll be a living document reporting upgrade in system architecture specifications following the feedback from the other WPs. It'll contain the description of the information related to the specification of the hardware architecture of the IDV-A/E GPU/FPGA based.





Partner Report Activity

Task 1.5.1	Hardware platforms: to provide requirements and specifications on energy
TL: E4	consuptions and cooling at level of node for IDV-E and IDV-A platforms
Task 1.5.2	Participants: E4, ATOS, InQuattro, ENEA, POLIMI
TL: ATOS	
Github address	The software developed and the benchmarks carried out during the activity are
	downloadable at github at the address: <u>https://gitlab-tex.enea.it</u>
Technology	ENEA HPC CRESCO Data Centre
Technical	The technical development is performed by ENEA, INRIA, E4,ATOS, IN4
development	

List of Authors

ENEA	F. lannone, P. Palazzari
INRIA	B. Bramas
ATOS	S. Lesmanne
E4	D. Gregori
in quattro	G. Zummo

List of Acronyms

AaaS	Accelerator as Service
ABI	Application Binary Interfaces
ACP	Acceleration Coherency Port
ACPI	Advanced Configuration Power Interface
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
AXI	Advanced eXtensible Interface (Xilinx IP)
BMC	Baseboard Management Controller
C/R	Checkpointing/Restart
CAPI	Common Application Programmer's Interface
CCXI	Cache Coherent Interconnect for Accelerators
CDU	Cooling Distribution Unit
CLB	Configurable Logic Block
CNN	Convolution Neural Network
СР	Common Platform
CPU	Central Processing Unit
CRDB	Co-design Recommended Daughter Board
CU	Compute Unit
CXL	Compute Express Link
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
FSB	Front Side Bus
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
HTC	High Throughput Computing
IoT	Internet of Things
IOB	Input/Output block
IP	Intellectual Property
IPMI	Intelligent Platform Management Interface
IR	Iterative Refinement
KPN	Kahn Process Network
L2HN	L2 cache Coherence Home Node
LE	Logic Element
	Local Resource Manager
	Multi-Chip-Module
MDC /T	Molecular Dynamic
	Metadata Server/Target
	Machine Learning
	Memory Management Unit
	Message Passing Interface
MPEaC	Multi-Purpose Processing Array
IVIPSOC	Mauri-Processor System on Chip
NN NoC	Netural Network
NUC	Network On Cinp
NVIC	Nested Vectored Interrupt Controller
	Open Compute Project
	OCP Accelerator Module
	Our Auterialui Muulue Auick Path Interconnect
	Quick Fall Interconnect
	Dujeu sulage servers / larger
	Preconutioned Conjugate Gradient
	Peripheral Component Interconnect Express
	Deta Eloating Doint Operations per Second
TILOFJ	r cla noaling r onit Operations per Jecond





PGMRES	Preconditioned Generalized Minimal Residual
PSU	Power Suppy Unit
PU	Processing Unit
PUF	Power Usage Effectiveness
OCD	Quantum ChromoDynamic
OFDB	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
RoCE	RDMA over Converged Ethernet
RTC	Real Time Clock
RTRM	Runtime Resource Manager
SAS/SATA	Serial Attached SCSI/Serial ATA
SLR	Super Logic Region of FPGA
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
TOPS	Tera 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
XRT	Xilinx Runtime Library



1 Introduction

The design and configuration of HPC systems has changed over time. An effective co-design process between application users and technology developers is crucial for a successful HPC ecosystem to ensure technology is relevant for applications and can be fully exploited when available. Power consumption has become a major concern and a clear trend towards heterogeneous systems has been established: combining general purpose CPUs with different kinds of acceleration devices such as GPUs and FPGAs. Accelerators deliver a high Flop/Watt ratio but come at a price of increasing programming complexity, with application developers often required to rewrite significant parts of their codes to be executed efficiently on these devices.

The high computing density of today's HPC systems and the efforts for energy efficiency requires higher focus on socket packaging, cooling, monitoring, and power aspects of the hardware. Direct cooling techniques are seen as major trend for dense computing power with energy efficiency. For better modularity of the hardware system architectures, more disaggregation technologies based on standardized interfaces should be explored on the su-system level such as compute acceleration, processing, memories hierarchical, networking, I/O and persistent storage. This will allow each subsystem to evolve and improve at its own pace without the need to compromise with other components that otherwise are too tightly connected with and dependent on each other.

Numerous and diverse low-energy and more well-performing compute technologies (CPU, GPU or other accelerators, application- tuned programmable logic, etc.) are appearing, increasing the performance per watt ratio for a given set of applications and workloads. The expanding diversity of computing elements calls for new system architectures which should be able to orchestrate them and share them in an efficient and flexible way between applications. Several approaches exist that take advantage of the evolution of both standard and proprietary interfaces and protocols (e.g., PCIe, CXL, CCIX, GenZ, NVlink, OpenCAPI, RDMA, RoCE) in order to interconnect the system components. The traditional host-device approach, in which, within a node, one or more accelerators are attached to a host CPU that takes over booting, orchestration and network communication capabilities, has evolved in the meantime towards "island-constructions" where accelerators (in particular GPGPUs) are interconnected with each other building "very fat nodes". On the other hand, composable node-designs are best suited for creating heterogeneous nodes with the right mix of components (CPU, memories, accelerators, network) for specific problems.

One of the important parts of the energy budget is the data movement: moving data takes time and energy (orders of magnitude difference between an on-die transfer and transfers between chips on boards, or even worse between boards). By reducing the distance between compute nodes (general purpose processors and accelerators), networking, memory and storage (persistent and nonpersistent), it is expected to further increase efficiency of nodes. The ultimate option is the emerging field of "computing near or in memory" architectures but it is not mature enough to be used in short term production systems. Modularity and composability are also important requirements of current machines: composable nodes (comprising CPU, accelerators, DDR or HBM memories, persistent storage, network interface, interlinked by (a) (coherent) switch(es)) allow to tune the efficiency towards the different workload; composable racks (changing the ratio between compute and storage) are also emerging.



The variety of computing resources opens new possibilities for "reconfigurable computing" which offers better application efficiency by adapting the system to the needs of each individual user. Virtualization and containerization approaches provide each user with a "virtual cluster" according to specific needs. This is applied at the system level by allocating the right mix of compute components and at the node level by using programmable devices such as FPGAs. Such approaches also open HPC to Cloud usage models, enabling higher requirements in terms of security and isolation between users. In this new context, research is required to optimize scheduling, resource allocation and sharing of multiple (potentially heterogeneous) resources as well as to minimize the overhead of the resource abstraction and virtualization. Ideally, enough intelligence should be available in the system software for it to decide on which hardware to run each part of an application without transferring this burden to the user. Furthermore, standard programming interfaces, adaptive libraries and APIs are needed to facilitate programming the new devices and improve performance portability.

New memory technologies shall enable increasing the memory capacity inside the node, with new memory models employed to exploit it. Deeper memory hierarchies shall increase the effective bandwidth and reduce the effective latency. Local memories are becoming heterogeneous, with a mixture of HBM or HBM2 for high bandwidth and DDR for volume storage. This is driven, for example, by Big Data analytics and Artificial Intelligence workloads which need more data accesses (and more bandwidth) and other classical memory bound HPC workloads such as QCD. Many new workloads are memory-bound on traditional hardware, with a byte/flop ratio of 10 or higher.

Handling the evolution of individual components is amongst the first targets for system integration. The maximum possible power envelope interacts directly with component maximum performance because it is one of the major bottlenecks. The cooling techniques that are key for reliability, TCO and density become attractive for pushing up performance limits. Energy reuse and a better controlled energy supply (capping, energy consumption optimization, monitoring) are areas where active research improves HPC carbon footprint. Additionally, monitoring with access to diverse sensors should enable identifying the power-hungry components and determine how and where to execute different parts of applications and workflows to achieve the best overall energy-to-solution. Applying AI techniques on monitoring data promises progress in this area.

All the above improvements on system architectures must be implemented considering the evolving needs and characteristics of applications. The only way to make it possible is applying stringent codesign approaches in a coordinated development of hardware, middleware, and applications. As modern and future HPC systems will be used for a wide variety of fields, co-design must include realistic use-cases and datasets from all relevant fields: HPC, HTC, HPDA, AI, ML, DL, etc. These must be supported not only individually but also in combination with each other in complex, orchestrated application workflow scenarios that can also be executed concurrently.

Accordingly, computer systems should be adaptable to very diverse requirements. The decision on which parts of the heterogeneous system a given code is executed should be supported by system modelling and simulators that enable forecasting application performance on different system configurations. The deployment of codes onto the system should guarantee the security of the sources and their associated data e.g. through application containerization and isolation.



2 FPGA platform requirements

FPGA are now being used for acceleration in a wide range of applications, both in HPC systems and embedded computers. The ready availability and high-power efficiency of high-density FPGAs make them attractive in HPC landscape. Since their initial deployment in the mid-1980s, FPGA have been used to accelerate high performance applications on custom computing systems. FPGAs have historically been restricted to a small niche of HPC applications because of their relatively high cost. Over time, however, improvements in process technology have enabled vendors to manufacture chips containing multi millions of transistors. The architectural enhancements, increased locic cell count and speed contribute to an increase in FPGA logic computer performance. For instance, with an average 25% improvement in typical clock frequency for each FPGA generation, the locic compute performance (clock frequency \times logic cell count increase) has improved approximatively by 92x over the past decade while the cost of FPGAs has decreased by 90% in the same period. These developments have made it feasible to perform massive computations on a single chip at increased compute efficiency for a lower cost.

FPGAs are composed of a large array of configurable logic blocks (CLBs), digital signal processing blocks (DSPs), block RAM, and input/output blocks (IOBs). CLBs and DSPs, like a processor's arithmetic logic unit (ALU), can be programmed to perform arithmetic and logic operations like add, multiply, subtract, compare, etc. Unlike a processor, in which architecture of the ALU is fixed and designed in a general-purpose manner to execute various operations, the CLBs can be programmed with just the operations needed by the application. This results in increased compute efficiency.

Depending on the type of operators used, CLBs and DSPs can perform integer, floating point, and bitwise operations. The results of the operations are stored in the registers present in CLBs, DSPs, and block RAM. These blocks within an FPGA can be connected via flexible configurable interconnects. The output of one operator can directly flow into the input of the next operator, meaning that the FPGA's architecture lends itself to the design of data flow engines.

The FPGA architecture provides the flexibility to create a massive array of application-specific ALUs that enable both instruction and data-level parallelism. Because data flows between operators, there are no inefficiencies like processor cache misses; FPGA data can be streamed between operators. These operators can be configured to have point-to-point dedicated interconnects, thereby efficiently pipelining the execution of operators.

The parallelism offered by FPGA architecture can be easily seen in a few examples of HPC-relevant parameters:

- Internal bandwidth to move the operands and results of application specific ALUs are in order of terabytes/sec (TB/s)
- Throughput on integer operations is in the order of Tera-operations/sec (TOPS)
- Throughput on floating point operations is in the order of gigaflops/sec (GFLOPS)

The IOBs in the FPGA architecture offer several features that can be interfaced with computing system components, and in particular, are designed to support various memory and processor-interface standards. For instance, FPGAs can support multiple DDR3 memory controllers—as many as six DDR3 controllers on FPGAs with the highest densities. The higher the number of memory controllers on an



FPGA, the higher the bandwidth to the external memory. In addition to the DDR3 interface, FPGAs also provide support to interface with DDR, DDR2, RLDRAM, and QDR SRAM memories.

FPGA architecture provides support to interface and run PCle Gen1/Gen2/Gen3/Gen4, Intel's Front Side Bus (FSB), and Quick Path Interconnect (QPI) protocols. Support for these processor interfaces and protocols enables the computing applications running on FPGAs to interact with the processor and access the data required to accelerate the applications.0

A new development in recent years is the in-socket FPGA accelerator. With the ability to run FSB and QPI protocols on FPGAs, one or more processors in a multi-processor server can be replaced with FPGAs, allowing portions of the application to be accelerated using in-socket FPGA accelerators. In-socket accelerators provide the additional capability of keeping the data coherent with the processor memory space compared to PCIe-based accelerators. For instance, GPU accelerators are all PCIe-based and cannot keep the data coherent with the processor memory space; an FPGA in-socket accelerator provides this unique capability. This fact has important implications for the type of applications that can be accelerated as well as for the accelerator systems programming model.

The convergence of storage and Ethernet networking is driving the adoption of 40G and 100G Ethernet in data centers. Traditionally, data is brought into the processor memory space via a PCIe network interface card. However, there is a mismatch of bandwidth between PCIe (x8, Gen3) versus the Ethernet 40G and 100G protocols; with this bandwidth mismatch, PCIe (x8, Gen3) NICs cannot support Ethernet 40G and 100G protocols. This mismatch creates the opportunity for the QPI protocol to be used in networking systems. This adoption of QPI in networking and storage is in addition to HPC.

FPGAs tend to consume power in tens of watts, compared to other multicores and GPUs that tend to consume power in hundreds of watts. One primary reason for lower power consumption in FPGAs is that the applications typically operate between 100–300 MHz on FPGAs compared to applications on high-performance processors executing between 2–3 GHz.

The ability to parallelize the applications on FPGAs, coupled with lower power consumption compared to CPUs and GPUs, results in increased performance-to- power-efficiency of FPGAs. For instance, an application that runs 10X faster than a multicore at 4X lower power results in 40X improvement in performance-to-power- efficiency on FPGAs.

2.1 FPGA system architecture

Todays there are two FPGA based architectures available for designing a compute node for HPC data center:

System-on-Chip FPGA: it integrates CPU host and FPGA architectures into a single device (a SoC integrated circuit inside the FPGA). It improves integration, lowers power and reduces board size. The FPGA can better control the board with faster communication between CPU host and FPGA and it is reprogrammable at any time. The on-chip memory, high bandwidth communication, and expansive set of possible peripherals in an SoC FPGA board can be of enormous benefit when designing a system with tightly constrained metrics for embedded use. Unlike PCIe designs, a System-on-Module (SoM) setup can offer a high capacity of logic and memory storage on a small module size.



 PCIe FPGA: it involves communication between CPU host and FPGA via PCIe bus. Vendors of FPGA devices such as: Xilinx and Intel, provide a Transaction Layer front-end IP core to use with application logic. roper DMA-based communication from the FPGA to the host requires some awareness of the specification's details, but it's otherwise straightforward in the sense that packets are formed, dispatched, and assured to arrive in the order they were sent. The other direction, host to FPGA, is somewhat trickier since the FPGA's active part is merely to issue a read request. The completions, containing the requested data, arrive in the time and format depending on the host and bus fabric. The reception logic must be prepared to react to different scenarios, so FPGA logic that worked with one host may fail with another, unless properly designed.

Compared to using a PCIe FPGA, a solution using a SoC FPGA is cheaper, uses less power consumption, and is easier to put into a design. Two components: CPU host and FPGA are replaced by one, which means less time for designing and less space on the PCB.

If external RAM is used for both the CPU host and the FPGA, these memory circuits can be consolidated into one RAM chip, saving space, cost and reducing complexity. Communication between processor and FPGA can also go much faster with both units on the same chip.

On the other hand, an architecture with PCIe FPGA is handier because SoC FPGA are somewhat limited when it comes to re-configurability. The data flows between CPU host and FPGA via PCIe can be developed better than in SoC FPGA where it can only be re-configured through a totally new design cycle where the chip's gates will be re-allocated to a new architecture. Moreover, beyond a simple daisy-chain connection, there is no easy way to redirect or reroute the data flow connections already configured on an SoC FPGA to accommodate the addition of a new functional block.

2.2 IDV-E Target Platform

The FPGA platform for IDV-E under develop in E4 shall provide a technology based on the current available solutions of PCIe FPGA. As reported in [*D1.1,2021*] the Xilinx Alveo FPGA discrete boards U250 and U280 are mature technologically with a programming toolchain, *Vitis*, developed enough and they require a compute host with a PCIe at least Gen3. Unfortunately, the Intel Agilex serie M, with features suitable for HPC data center applications, is not available yet as PCIe card.

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).

Both the U280 and U250 have got 3 Super Logic Regions (SLRs). An SLR is a physical section of the FPGA with a specific number of resources and connections. Whilst the U280 has the bottom SLR (SLR0) integrating an HBM controller to interface with the adjacent 8 GB HBM2 memory, the U250 doesn't have it. Both devices connect to 16 lanes of PCI Express[®] that can operate up to 16 GT/s (Gen4) for



U280 and 8 GT/s (Gen3) for U250. U280 device connects SLR0 and SLR1 to DDR4 16 GB, 2400 MT/s, 64-bit with error correcting code (ECC) DIMMs for a total of 32 GB, whilst U250 has 4 DDR4 16 GB 2400 MT/s for a total of 64 GB of DDR4. Both devices connect to two QSFP28 connectors with associated clocks generated on board.

The Tab.2.2.1 shows the technical specification of the U250/U280 FPGA board installed in the ENEA FPGA Lab.

	U250	U280	
Look-up tables	1,728К	1,304К	
Registers	3,456K	2,607K	
DSO Slices	12,288	9,024	
INT8 TOPs Peak	33.3	24.5	
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 with CCIX	
Network Interface	2x QSFP28	2x QSFP28	
Typical power	110 W	100 W	
Maximum power	225 W	225 W	

Tab.2.2.1: Xilinx Alveo U250 and U280 specifications

Both the FPGA cards can be equipped with active or passive airflow. Passive cards don't include a builtin fan and therefore require an external mechanism to ensure proper airflow for cooling. Passive cards should be powered with a forced airflow mechanism or a direct cooling (liquid or two-phase). The temperature specifications of the FPGA chip and board are measured by two sensors with the following limits for both U280 and U250 (Tab.2.2.2):

Sensor name	Warning limit (°C)	Critical limit (°C)	Fatal limit (°C)
FPGA chip	88	97	107
board	100	110	125

Tab.2.2.2: Xilinx Alveo U250 and U280 thermal limits specification

The difference between the two Xilinx cards is mainly in the HBM available on the U280. HBM is a novel memory architecture that enables high-performance and adaptability for memory-bound applications. HBM is a 3D-stacked DRAM that offers high-bandwidth and energy-efficient data movements. This feature is very important in HPC applications, therefore the PCIe FPGA of IDV-E will be the U280.

The Alveo U280 is built on the Xilinx 16nm UltraScale+ architecture and offers a rich set of memory solutions, as shown in Fig. 2.2.1. The Alveo U280 card features the XCU280 FPGA, which combines



three super logic regions (SLRs). The Tab 2.3, SLRO integrates an HBM controller to interface with the HBM2 subsystem through 32 pseudo-channels (PCs) each with direct access to 256 MB of storage (8 GB in total). Each 256-bit PC operates at 450 MHz, yielding a maximum bandwidth of 14.4 GB/s. The full system can thus achieve a theoretical bandwidth of 460.8 GB/s. SLRO also connects to the host via 16 lanes of the PCI Express (PCIe) interface. SLRO and SLR1 each connect to 16 GB of DDR4 each. Finally, each region has up to 8 MB of PLRAM for fast access to small data sets. In the following, we will use the term global memory for the set of memories available on the board. The host must transfer data into the device global memory before they can be accessed by the FPGA logic [*Soldavini,2022*].

		PCIe
	CU SLL	••
SLR2	SLR1	CU SLR0 CU CU CU CU CU CU CU CU CU CU CU CU CU
	t	↑
	DDR4	DDR4
Fig.2.2.1. A	rchitectur	e of XCU280

Resources	SLR0	SLR1	SLR2
HBM	32x256MB	-	-
DDR4	16GB	16GB	-
PLRAM	2x4MB	2x4MB	2x4MB
CLB LUT	369K	333K	367K
CLB Reg.	746K	675K	729K
Block RAM tile	507	468	512
UltraRAM	320	320	320
DSP	2733	2877	2880

Tab.2.2.3: Alveo U280 SLR resources

The target system for the Alveo U280 is composed of multiple compute units (CUs). Each CU is a userdefined hardware module that can be attached to any of the PCs through independent AXI interfaces, while the built-in HBM controller and switch have access to all physical channels. The CU can be described in C++ and synthesized with HLS or specified directly in RTL. Multiple CUs allow parallel execution but must be connected to different HBM channels. The system configuration file describes the connections between the CU ports and the HBM channels. The required logic is automatically generated during system synthesis.

Xilinx offers a unified software platform, called Vitis, to develop FPGA applications. Vitis includes a rich set of hardware-accelerated open-source libraries optimized for Xilinx FPGA and the Xilinx Runtime library (XRT) to facilitate communication between the host application (running on the host CPU) and the accelerator deployed on the reconfigurable portion of the card, which is connected via PCIExpress. It also includes user-space libraries and APIs, kernel drivers, and board utilities that can be used to measure performance and monitor power consumption. In this work, we aim to automate the generation of CU descriptions and the associated configuration file directly on top of the existing Xilinx libraries.

To specify the FPGA performance, floating-point operations for *Matrix-to-Matrix* multiplication are used as reference in this project since Xilinx Vitis BLAS Library is available including benchmarks on Xilinx Alveo U200. Usually, DGEMM function is relevant to stress important system properties or generate workloads that is like relevant applications. Some benchmarks were carried out in [*Meyer,2020*] implementing the GEMM matrix-matrix multiplication function of the BLAS library in a kernel function for U280 PCIe FPGA. The GEMM Cannon's Matrix Multiplication Algorithm used in the benchmark, is a 4096x4096 matrices for the calculation based on 8x8 matrices block, so the kernel can initialize a 1024 Floating-point single precision multiplications and additions per clock cycle. The board U280 with DDR and HBM usage is about 200 GFLOPS/s with a clock frequency of 250 MHz. It's difficult to establish a performance limit for DGEMM function because it depends on the algorithm. In [*de*]



Haro,2021] the implementation of DGEMM on Xilinx Alveo U200 card performed a benchmark of 350 GFLOPS/s in single precision (FP32) with a clock of 300 MHz. The best performance pf a DGEMM single precision on Xilinx U200 is 409 GFLOPS/s [*Licht,2020*]. Maybe with a DGEMM optimized algorithm developed on Xilinx Alveo U280 a performance of 0.5 TFLOPS/s can be achieved.

The IDV-E target platform based on PCIe FPGA Xilinx Alveo U280 is shown in fig.2.2.4 (a) with the block diagram in fig.2.2.4(b).



Fig.2.2.4. (a) Xilinx Alveo U280 with passive cooling. (b) U280 block diagram

The main specifications are in Tab. 2.2.4.

	Specifications
Model	Xilinx Alveo U280
processor	XLINK Virtex Ultrascale+ XCU280
resources	LUT:1304K – Register:2607K – DSP: 9024
HBM2	8GB with total bandwidth of 460 GB/s
DDR4	2x DDR4 16 GB, 2400 mega-transfers per second (MT/s), 64-bit with ECC DIMM
DGEMM single precision	0.5 TFLOPS/s
Max Power Consumption	225 W
Energy efficiency	450 mW/GFLOPS/s
PCI Express	2x Gen4 x8, Gen3 x16 with CCIX
Network Interface	2 x QSFP28 (InfiniBand IBTA EDR and IEEE 802.3bj 100 GbE)
Cooling	Passive and Direct Liquid/2-Phase
Dimensions	Height: 111.15 mm - Length: 242 mm – Thickness: 39.04 mm

Tab.2.2.4: Xilinx Alveo U280 specifications

Heterogeneous architectures combine CPUs and different accelerators: GPU/FPGA within a single compute node. Such systems are capable of processing parallel workloads very efficiently while being more energy efficient than regular systems consisting of CPUs only. Taking to extremes of the current developments in hardware specialization for performance/energy-efficiency, and chipletization for reducing manufacturing cost, it might end up with a heterogeneous node as shown in Fig. 2.2.5 on which several host CPUs equipped with several accelerators devices have got their own memory and they communicate with each other via I/O bus. There are two modes of communication between



heterogeneous processors: synchronous and asynchronous. Asynchronous communication can realize the overlap of communication and calculation and hide the communication time. For different bus communication methods, the current way for the CPUs and PCIe FPGA heterogeneous architecture, simply unloading the kernel library to the FPGA will cause a large amount of data to pass through the low-speed PCIe bus, and the communication overhead also affects the scalability. A common approach in the programming model in heterogeneous architectures. is to use the CPU/accelerator coordinated calculation, one of which is that the CPU is only responsible for managing the work of the accelerator, such as distributing data and coordinating management, while the accelerator is responsible for all calculations. The other is to let the CPU also take on part of the calculation task, and complete the calculation together with the accelerator



Fig.2.2.5. Extremized Heterogeneous architecture

While such a node design will still offer tremendous benefits to current CPU/Accelerators systems, such a design will have several disadvantages as well.

- The number of available different memory spaces and the requirement to move data/instructions back and forth between them could dominate the performance and could seriously limit the benefits from using special-purpose accelerators. For example, the latency cost of launching several thousand kernels, reconfiguring the accelerators for them and/or returning to host communicate through a NIC is not a scalable approach.
- Efficient use of all the available compute resources concurrently will require significant changes to programming model and applications. For example, applications currently explicitly use four different memory types in the CPU/Accelerator systems (CPU memory, Accelerator High Bandwidth Memory, Unified Virtual memory, and host pinned memory).



- Maintaining an application that can explicitly manage data in all different memory types will be challenging.
- The asymmetric view of memory and network where some compute units are "closer" to certain memory or network creates scaling bottlenecks.
- Coherency issues arising out of having multiple memory spaces have to be managed explicitly in hardware or software.
- Network communications become even more expensive when there are several levels of data movement from an accelerator to a host CPU and then to a smart-NIC before actually touching the wire.

An important research direction is to focus on the co-design of a node architecture that will be centred around a unified memory, where all compute units will have a "symmetric" view of the memory. The idealistic characteristics of such a node will be:

- Low latency costs in launching a kernel on an accelerator with no copies of data/instruction needed from one memory to another
- To be able to setup and communicate from any compute unit without the requirement to go to a host and do it with minimal latency
- Ability to avoid multiple coherence domains
- Ability to program all the compute units on the node seamlessly without explicitly managing multiple memory spaces

We see significant challenges and opportunities in co-designing such a heterogeneous node. The primary problem is that the right accelerators to mount on such a platform might vary significantly based on the target applications. The rise of open-source hardware designs provide a significant opportunity to participate in efforts to influence design of accelerators that serve the needs of the user applications and not just rely on industry developed solutions. A second opportunity that naturally arises out of such a co-design effort is to arrive at a node architecture with the right mix of CPUs, GPUs, FPGAs and special-purpose accelerators that is targeted towards high performance computing use cases.

The compute node currently available for HPC data centers equipped with PCIe FPGA in an extreme density layout is the Terabox 1401B/1402B (Fig.2.2.6).



Fig.2.2.6. Terabox 1401B/1402B compute node equipped with 4 PCIe FPGA

The TeraBox 1401B/1402B support up to four dual-slot PCIe FPGA cards. The Terabox gives the highest level of compute and network capability in a 1U chassis. The Terabox supports PCIe GEN4 x 16 lanes,



Intel Xeon, or AMD Epyc processors and up 1 TB of DDR4 RAM. The processors supported In the Ultra configurations are: Intel Xeon 8351N (2.4GHz, 36C/72C HThreading) with a TDP of 225W of and AMD EPYC 7702P (2.0GHz, 64C/128 Threads) with a TDP of 220W.

It is evident that an exascale HPC system composed of the aforementioned Terabox compute nodes is unsustainable. Taking into account FPGA Xilinx Alveo U280 with a DGEMM performance of 500 GFlops/s and TDP of 225 W, equipping each terabox with 4 FPGAs and 1 Intel Xeon 8351N, the DGEMM performance is about 3 TFlops/s of each node and a electrical power consumption of about 1.1 KW. It means about 375 mW/GFlops/s not suitable for exascale HPC systems with goal of 20 mW/GFlops/s.

In order to define the requirements of an exascale compute nodes based on FPGAs PCIe, bi-socket architectures with 2-4 FPGAs PCIe is taking into account (Fig.2.2.6). Both the configurations allow 1-D Torus networwk for FPGAs intra-node communication thanks to 2 QSFP28 ports currently compliant with SSF-8665 standard [*SSF-8665*] able to support the interface requirements for the operation of: InfiniBand IBTA EDR and IEEE 802.3bj 100 GbE.



Fig.2.2.6. Exascale compute node architectures based on bi-socket and (a): 2 FPGAs PCIe - (b):4 FPGAs PCIe.

The specifications of exascale a compute node FPGA PCIe based require a deployment from Xilinx of a FPGA ALVEO with DGEMM performance at least of 20 TFlops/s and a max TDP of 250 W. On the other hand the current generation of CPUs multi-cores, such as: Intel Xeon and AMD Epyc, are able to perform HPL benchmarks less than of 5 TFlops/s with a TDP of about 500 W. Therefore, the architectures performances of 2 or 4 FPGAs PCIe bi-sockets compute nodes are as the following Tab.2.2.5:

Single Compute node	FPGA DGEMM [TFlops/s]	FPGA TDP [Watt]	CPU HPL [TFlops/s]	CPU TDP [Watt]	Exascale node [mW/GFlops/s]
Bi-socket 2xFPGAs	2 x 20 = 40	2 x 250 = 500	2 x 5 = 10	2 x 500 =1000	30
Bi-socket 4xFPGAs	4 x 20 = 80	4 x 250 = 1000	2 x 5 = 10	2 x 500 =1000	22

Tab.2.2.5. Exascale compute node perfomance

From point of view of the rack density, the bisocket 2xFPGAs architecture in a single node can be assembled in two single nodes in 1U rack unit, as shown in Fig.2.2.7, (a) and (b). The Direct Cooling (Liquid or Two-phase) is mandatory for 1U blade with 2 kW of power consumption. Furthermore, in order to achieve the target of 20 mW/GFlops/s in Exascale HPC system of Tab.2.2.6, low power CPUs, such as ARM, are required.





1U Blade Compute node	FPGA DGEMM [TFlops/s]	FPGA TDP [Watt]	CPU HPL [TFlops/s]	CPU TDP [Watt]	Exascale node [mW/GFlops/s]
2xBi-socket 2xFPGAs	4 x 20 = 80	4 x 250 = 1000	4 x 5 = 20	4 x 250 =1000	20

Tab.2.2.6. Exascale 1U blade performance



Fig.2.2.7. Exascale 1U blade. (a) 2xbi-socket 2xFPGA PCIe – (b) bi-socket 4xFPGA PCIe

2.2.1 IDV-E platform requirements

The IDV-E platform will be developed with the current technologies available and the bi-socket 2xFPGA PCIe configuration will be the platform on which Proof of Concepts (PoCs) and benchmark use cases will be developed.

The CPU host in a hybrid architecture of a compute node plays a crucial function since it provides to work-load balancing of computing tasks running on the accelerated cards available in the node as well as the data movement intra-node and inter-node. On the other hand the CPU has to be able to perform all its own tasks, including computational operations, with high efficiency in terms of Watt/Flops/s. The SKU list segments of Intel Xeon Sapphire Rapids, AMD Epyc Genoa and ARM Ampere Altra Max is reported in the Tab.2.2.1.1.



				cores	clock GHz	TDP Watt
			-	r		
			8490H	60	1.9	350
			8480+	corres coock Hi2 60 1.9 56 2 52 1.8 52 1.8 52 2 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.2 40 2 20 2 40 2.2 40 2.2 16 2.0-215 64 2.3-2.4 212 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.25-2.25 16 3.6-3.8 16 2.6-2.7 128 3 2	350	
			8471N		300	
			8470Q		350	
			8470N		300	
			8470	52	2	350
			8468V	48	2.4	330
INTEL	Sapphire Rapids-SP Xeon		8468H	48	2.1	330
		Platinum	8468+	48	2.1	350
			8461V	48	2.2	300
			8460Y	40	2	300
			8460H	40	2.2	330
			8458P	44	2.7	350
			8454H	32	2.1	270
			8452Y	36	2	300
			8450H	28	2	250
			8444H	16	2	270
	-		00540		20.245	200
			9654P	96	2.0-2.15	300
			9534	bit 56 2 11N 52 1.8 0Q 52 2 0N 52 2 8W 48 2.1 13V 48 2.1 14W 48 2.1 13V 48 2.1 14W 48 2.1 13V 48 2.1 14W 48 2.1 14W 48 2.1 13V 48 2.2 04H 0 2 28P 44 2.7 4H 32 2.1 27 36 2 0H 28 2 4H 16 2 4H 32 2.75.285 4H 32 2.75.285 34 32 2.75.285 34 32 2.75.285 34 32 2.45.25 34 32 3.45.6 <t></t>	280	
			5434F		250	
			9454	48	60 1.9 56 2 52 1.8 52 2 52 1.7 52 2 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 40 2.2 40 2.2 36 2 16 2 16 2 96 2.0-2.15 64 2.3-2.5 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.75-2.85 32 2.6 112 3 96 2.8	290
AMD	EDVC		9354P	32	2./5-2.85	280
AND	Line	GENOA ZEN4	0224	32	2.73-2.85	200
			02745	32	2.3-2.3	210
			5274F	32	3.4-3.0	320
			9234	24	2.4-2.3	200
		Platinum GENOA ZEN4 Mystique QuickSilver	01745	16	2.13-2.23	200
			0124	10	3.0-3.8	320
			9124	10	2.0-2.7	200
			M128-30	128	3	250
			M128-28	128	2.8	230
			M128-26	cores clock 61 60 1.9 56 2 52 1.8 52 2 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 48 2.1 40 .2 40 .2 40 .2 40 .2 40 .2 40 .2 40 .2 44 .2.7 28 .2 16 .2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32 .2.75.2 32	2.6	190
		iviystique	M112-30	112	3	240
			M96-30	96	3	220
			M96-28	96	2.8	190
			Q80-33	80	3.3	250
AKM	AMPERE ALTRA MAX		Q80-30	80	3	210
			Q80-26	80	2.6	175
			Q72-30	72	3	195
		QuickSilver	Q64-33	64	3.3	220
			Q64-30	64	3	180
			Q64-26	64	2.6	125
			Q64-24	64	2.4	95
			032-17	32	17	45

Tab.2.2.1.1. CPU sku list for Intel, AMD and ARM.

The SKU list of the CPUs shows that the ARM Ampere Altra Max can be suitable as model for IDV-E platform even if a 1 U blade with two bi-socket 2xFPGA PCIe is not available yet. Therefore, the specifications of IDV-E target platform will be based a server node on which a comparison between Direct Liquid Cooling and Direct Two-phase Cooling shall allow to evaluate the best technology solution in terms of energy performance.

IDV-E hardware specifications

The node will be a bi-socket based on Ampere Alta Max with the following specifications:

HOST Processor

CPU – ARM Ampere Altra MAX

- 128 Armv8.2+ 64-bit CPU cores up to 3.0 GHz maximum
- 64 KB L1 I-cache, 64 KB L1 D-cache per core
- 1 MB L2 cache per core
- 16 MB System Level Cache (SLC)
- 2x full-width (128b) SIMD
- Coherent mesh-based interconnect- Distributed snoop filtering

MEMORY

- 8x 72-bit DDR4-3200 channels





- ECC, Symbol-based ECC, and
- DDR4 RAS features
- Up to 16 DIMMs and 4 TB/socket

SYSTEM RESOURCES

- Full interrupt virtualization (GICv3)
- Full I/O virtualization (SMMUv3)
- Enterprise server-class RAS

CONNECTIVITY

- 128 lanes of PCIe Gen4–4 x16 PCIe + 4 x16 PCIe/CCIX with Extended Speed Mode (ESM)
- support for data transfers at 20/25 GT/s 32 controllers to support up to 32 x4 links
- 128 PCIe lanes in 1P configuration
- 192 PCIe lanes in 2P configuration
- Coherent multi-socket support

TECHNICAL SPECIFICATIONS

- Operating Junction Temperature Range 0°C to +90°C
- Power Supplies CPU: 0.75 V, DDR4: 1.2 V I/O: 3.3 V/1.8 V, SerDes PLL: 1.8 V
- Packaging 4926-Pin FCLGA

TECHNOLOGY & FUNCTIONALITY

- Armv8.2+, SBSA Level 4

- Advanced Power Management – Dynamic estimation, Voltage droop mitigation PERFORMANCE & POWER

- Est. SPECrate[®] 2017_int_base (SKU: AC-212825002): 359 at Usage Power: 178 W
- Max TDP: 250 W

PROCESS TECHNOLOGY

- TSMC 7 nm FinFET
- x16 CCIX lanes

The CPU block diagram is shown in Fig.2.2.1.1 with 128 cores operating at maximum of 3.GHz. Each core I single threaded by design with its own 64 kB L1 I-cache, 64 kB L1 D-cache, and a huge 1 MB L2 cache, delivering predictable performance 100% of the time by eliminating the noisy neighbour challenge with each core.



Fig.2.2.2.1. ARM Ampere Altra MAX block diagram



Supporting eight, 2DPC, 72-bit DDR4-3200 channels, the Ampere Altra Max processor offers high bandwidth and memory capacity of up to 4 TB per socket. With 128 lanes of PCIe Gen4 per socket with support for 192 PCIe Gen4 lanes in 2P configuration that can be bifurcated down to x4, Ampere Altra Max provides maximum flexibility to interface with off-chip devices, including networking cards up to 200 GbE or more, and storage/NVMe devices. Ampere Altra Max provides industry-leading power efficiency/core, while packing 128 cores in a single-socket and 256 cores in a bi-socket platform, establishing new levels of power efficiency with scalability. Ampere Altra Max processor's advanced power management capabilities include Advanced Configuration Power Interface (ACPI) v6.2 support, Dynamic Frequency Scaling (DFS), on-die thermal monitoring, and dynamic power estimation. The Ampere Altra Max processor provides extensive enterprise server-class RAS capabilities. Data in memory is protected with advanced ECC in addition to standard DDR4 RAS features. End-to-end data poisoning ensures corrupted data is tagged and any attempt to use it is flagged as an error. The SLC is also ECC protected, and the processor supports background scrubbing of the SLC cache and DRAM to locate and correct single-bit errors before they accumulate into un-correctable errors. Ampere's Altra Max processors are supported by an extensive partner ecosystem of products and services from a wide range of leading suppliers, including industry standard providers of: Operating System (Linux RHEL, Centos) and hardware/software development tools. Several platforms: 1U, 2U and Half-Width rack unit are available for Ampere Alta Max.



Fig.2.2.1.2. ARM Altra Max 77.080 mm × 67.000 mm 4926-Pin Flip Chip Land Grid Array (FCLGA) Mechanical Data

The mechanical data and package marking of the ARM Ampere Altra Max, needs to design the cold plate for the direct cooling of the CPUs, are shown in Fig.2.2.1.2.

FPGA PCIe

The IDV-E platform based on bi-socket ARM Ampere Altra Max, will be equipped with two Xilinx FPGA Alveo U280 with passive cooling. The Tab.2.4 reports the technical specifications of the FPGA PCIe.

<u>Server Node</u>

In order to install two FPGA PCIe Xilinx Alveo U280 aided by a bi-socket ARM Ampere Altra Max with Direct Cooling, the 2U rack unit node, Mt.Collins, supports a number of PCIe slots providing the





possibility of adding FPGA boards (up to 3) and/or other boards if needed. It has the geometric space for adding the direct cooling system and it has a good match between the amount of heat to be removed and the the design point of the direct cooling system.

The Fig.2.2.1.3 shows the Mt.Collins provided by E4 (Fig.2.2.1.3(a)) and the block diagram of the node (Fig.2.2.1.3 (b)).



Fig.2.2.1.3. Ampere 2U Mt. Collins node (a). The block diagram (b)

The versatile platform offers 160 PCIe Gen4 lanes for flexible I/O connectivity via PCIe slots and another 16 PCIe Gen4 lanes for OCP 3.0 networking. Mt. Collins supports thirty-two DDR4 3200 MT/s DIMMS with a maximum memory capacity of 8 TB. It also supports OCP NIC 3.0 connector with multi-host support to capitalize on the mechanical, thermal, manageability, and security benefits. In addition, Mt. Collins includes one internal M.2 NVMe storage interface for ultra-fast reads/writes, eliminating PCIe switch adapters. Mt. Collins includes MegaRAC[®], BMC, and Aptio[®] V BIOS support. Key features include dynamic fan control, temperature monitoring, and TPM 2.0 for security. The platform includes two redundant power supplies providing the reliability required for data centers. BMC includes support for IPMI and Redfish protocols for remote management. The dimensions are: 33.36 inch (L) x 17.63 inch (W) x 3.425 (H).

IDV-E software requirements

The IDV-E stack software for ARM (AARCH64) platform is as follow:

- Sys.Op.: Linux RHEL/CentosStream/Rocky v.8.x/9.x
- Developer Toolchains: GCC (9.x/10.x/11.x), LLVM (12.x/13.x), glibc (2.x), and binutils (2.x)
- Python 3.x
- Development Libraties
- Xilinx Runtime Library (XRT)

Xilinx Runtime Library is an open-source easy to use software stack that facilitates management and usage of FPGA/ACAP devices. Users use familiar programming languages like C/C++ or Python to write host code which uses XRT to interact with FPGA/ACAP device. XRT exports well defined set of software APIs that work across PCIe based datacenter platforms and ZYNQ UltraScale+ MPSoC/Versal ACAP based embedded platforms. XRT is key component of Vitis[™] and Alveo[™] solutions.

User application is made up of host code written in C/C++/OpenCL or Python. Device code may be written in C/C++/OpenCL or VHDL/Verilog hardware description language following the workflow of Fig.2.2.1.4 (a)



Fig.2.2.1.4. (a): User application compilation and execution. (b): Alveo PCIe stack

Users use Vitis[™] compiler, v++ to compile and link device code for the target platform. Host code written in C/C++/OpenCL may be compiled with gcc/g++. Host code may also be written in Python OpenCL (using PyOpenCL) or Python XRT (using built-in python binding).

The Xilinx PCIe Alveo stack is shown in Fig.2.2.1.4 (b) and XRT supports several FPGA PCIe Alveo including the U280. PCIe based platforms are supported on x86_64, PPC64LE and AARCH64 host architectures. The platform is comprised of physical partitions called Shell and User. The Shell has two physical functions: privileged PF0 also called mgmt pf and non-privileged PF1 also called user pf. Shell provides basic infrastructure for the Alveo platform. User partition (otherwise known as PR-Region) contains user compiled binary. XRT uses Dynamic Function Exchange (DFX) to load user compiled binary to the User partition

The MGMT PF (PF0) block includes the XRT Linux kernel driver *xclmgmt* binds to management physical function. Management physical function provides access to Shell components responsible for privileged operations. *xclmgmt* driver is organized into subdevices and handles the following functionalities:

- User compiled FPGA image (*xclbin*) download which involves ICAP (bitstream download) programming, clock scaling and isolation logic management.
- Loading firmware container called xsabin which contains PLP (for 2 RP platfroms) and firmwares for embedded Microblazes. The embedded Microblazes perform the functionality of ERT and CMC.
- Access to in-band sensors: temperature, voltage, current, power, fan RPM etc.
- AXI Firewall management in data and control paths. AXI firewalls protect shell and PCIe from untrusted user partition.
- Shell upgrade by programming QSPI flash constroller.
- Device reset and recovery upon detecting AXI firewall trips or explicit request from end user.
- Communication with user pf driver xocl via hardware mailbox. The protocol is defined Mailbox Inter-domain Communication Protocol
- Interrupt handling for AXI Firewall and Mailbox HW IPs.
- Device DNA (unique ID) discovery and validation.
- DDR and HBM memory ECC handling and reporting.



The User PF (PF1) block includes the XRT Linux kernel driver *xocl* binds to user physical function. User physical function provides access to Shell components responsible for non privileged operations. It also provides access to compute units in user partition. *xocl* driver is organized into sub-devices and handles the following functionality which are exercised using well-defined APIs in *xrt.h* header file.

- Device memory topology discovery and device memory management. The driver provides well-defined abstraction of buffer objects to the clients.
- XDMA/QDMA memory mapped PCIe DMA engine programming and with easy-to-use buffer migration API.
- Multi-process aware context management with concurrent access to device by multiple processes.
- Compute unit execution pipeline management with the help of hardware scheduler ERT. If ERT is not available, then scheduling is completely handled by xocl driver in software.
- Interrupt handling for PCIe DMA, Compute unit completion and Mailbox messages.
- Setting up of Address-remapper tables for direct access to host memory by kernels compiled into user partition. Direct access to host memory is enabled by Slave Bridge (SB) in the shell.
- Buffer import and export via Linux DMA-BUF infrastructure.
- PCIe peer-to-peer buffer mapping and sharing over PCIe bus.
- Secure communication infrastructure for exchanging messages with xclmgmt driver.
- Memory-to-memory (M2M) programming for moving data between device DDR, PL-RAM and HBM.

<u>WARNING</u>: ARM server is not an official supported use case. The XRT AARCH64 supports means embedded acceleration support, not Alveo support. *XOCL* can be cross compiled on arch64. Technically it's possible to control Alveo U50 on ARM servers. But adding this to an official support feature needs extra efforts to design and validate it.

REFERENCE

[D1.1,2021] Textarossa Deliverable: Gap Analysis (2021)

[*Soldavini,2022*] Soldavini S., Friebel K., Tibaldi M., Hempel G., Castrillon J., Pilato C. "Automatic Creation of High-Bandwidth Memory Architectures from Domain-Specific Languages: The Case of Computational Fluid Dynamics". ACM Transactions on Reconfigurable Technology and Systems. Accepted on August 2022 DOI: 10.1145/3563553.

[*Meyer,2020*] M. Meyer, T. Kenter and C. Plessl, "Evaluating FPGA Accelerator Performance with a Parameterized OpenCL Adaptation of Selected Benchmarks of the HPC Challenge Benchmark Suite," 2020 IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC), 2020, pp. 10-18, doi: 10.1109/H2RC51942.2020.00007.

[*de Haro*, 2021] J. M. de Haro et al., "OmpSs@FPGA Framework for High Performance FPGA Computing," in IEEE Transactions on Computers, vol. 70, no. 12, pp. 2029-2042, 1 Dec. 2021, doi: 10.1109/TC.2021.3086106.

[*Licht,2020*] Licht, Johannes & Kwasniewski, Grzegorz & Hoefler, Torsten. (2020). "Flexible Communication Avoiding Matrix Multiplication on FPGA with High-Level Synthesis". 244-254. 10.1145/3373087.3375296.

[SSF-8665] http://www.snia.org/sff/specifications



3 GPU platform requirements

The GPU platform for IDV-A under development in ATOS shall provide a new technology solution able to increase the power of the rack for HPC data center up to 147 kW, including DLC.

Obviously, the specifications of the IDV-A platforms depend on the GPUs available in the next few months of 2022 (Q4 2022), already reported in the D1.1 deliverable [*D1.1,2021*] of this project and reported as follow Tab.3.1:

	NVIDIA H100 SXM	AMD Instinct	Intel Ponte Vecchio
	110.000		
Architecture	Hopper (HG100)	CDNA2	XE-HPC
GPU Clusters	132 (SMs)	220 (CUs)	128 XE-Cores
FP32 (TFlops)	60	95.7	45
Memory size	80 GB HBM3	128 GB HBM2e	128 GB HBM2
TDP (Watt)	700	560	600
mW/GFlops (peak rate)	11.6	5.8	13.3
Programming Toolchains	CUDA	ROCm	OneAPI

Tab.3.1. GPUs specifications.

Today, NVIDIA GPU can be considered the dominant accelerator and CUDA is the most popular programming language for it [*Costanzo*, 2021]. One effort to face some of the programming issues related to heterogeneous computing is SYCL 3, a new open standard from Khronos Group. SYCL is a domain-specific embedded language that allows the developer to write single-source C++ host code including accelerated code expressed as functors. In addition, SYCL features asynchronous task graphs, buffers defining location-independent storage, automatic overlapping kernels and communications, interoperability with OpenCL, among other characteristics. Recently, Intel announced the oneAPI programming toolchain that provides a unified programming model for a wide range of hardware architectures. At the core of the oneAPI environment is the Data Parallel C++ (DPC++) program-ming language, which can be summarized as C++ with SYCL.

To tackle CUDA-based legacy codes, oneAPI provides a compatibility tool (*dpct*) that facilitates the migration to the SYCL-based DPC++ programming language. Recent experiences for migrating CUDA code to DPC++ using Intel OneAPI have been carried out in [*Costanzo, Rucci,2021*], reporting test the effectiveness of the compatibility tool for the selected test cases. Despite not translating 100% of the code, the tool does most of the work, reporting the developer of possible pending adaptations. Second, it was possible to verify the functional portability of the obtained DPC++ code, by successfully executing it on different CPU and GPU architectures. This study targets the last GPU generation, with the form factor optimized for liquid cooling and High-Performance Computing or Artificial Intelligence. Indeed, the standard PCIe (double slot Full Height Full Length) is limited to air cooled servers and then limited in power envelop (350W for Nvidia Hopper and 300W for Intel Ponte Vecchio).



3.1 GPU system architecture

For all GPU providers, the current architecture for water cooled GPU is based on a carrier board or baseboard with 4 GPU fully connected with high speed links (Nvlink for Nvidia, Infinity fabric for AMD, Xe link for Intel). Each GPU is connected to the host system running the operating system with 16 PCIe lanes (PCIe Gen4 for MI250X, PCI Gen5 for H100 and Ponte Vecchio).

The Open Compute Project has defined a new form factor for accelerators named OCP Accelerator Module (OAM), including the interfaces and the Baseboard as define in [OAM Spec.1.5].

AMD and Intel follow this standard, while Nvidia keeps its proprietary SXM form factor, using the Fifth generation SXM5 described in [*NVIDIA H100, 2022*].

Even if AMD MI250X is a generation before Nvidia Hopper and Intel Ponte Vecchio (launched end of 2021 with PCI Gen4), AMD has reached Linpack Exascale in TOP500 for the first time in June 2022. The Frontier system at Oak Ridge National Laboratory achieved 1.102 Linpack Exaflop/s in 21.1MW (52.2 GFlops/W). Each node contains one 3rd Gen AMD EPYC[™] CPUs (Trento) as host and four AMD Instinct[™] 250X accelerators. The architecture is described in the following Fig.3.1 with details in [*Frontier,2022*]



Fig.3.1. System architecture with AMD Instinct MI250X (NIC connected to GPU) and 1U blade

The specificity of this architecture is the direct connection from GPU to the slingshot interconnect, which does not exist in H100 or Ponte Vecchio. For H100 and Ponte Vecchio GPUs, there are several possible architectures to connect the node to the high-speed interconnect with at least 100Gb/s per







Fig.3.2. System architecture with CPU between GPU and NIC

GPU. One architecture provides direct connections of GPU and NIC (Network Interface Controller) to CPU, as described in the following Fig.3.2.

The number of NICs depends on the bandwidth required per processor and on the technology of the NIC. For example, four NICs are required when the NIC speed is @ 100Gb/s and the GPU requires 100Gb/s. But only two NICs, one per CPU, is possible with NIC @200Gb/s. The advantage of such architecture is that it works with all types of GPU but the CPU PCIe root complex must be efficient enough to support GPU and CPU communication without penalty. This architecture is the architecture of the Aurora supercomputer that Intel is currently installing in Argonne National Laboratory [*Argonne,2022*]. A second option is to use some NIC with one embedded PCIe switch to provide connections between CPU, GPU and NIC as described in the following Fig.3.3.



Fig.3.3. System architecture with NIC between CPU and GPU

In this architecture, the main constraint is on NIC with two x16 PCie slots, one to CPU and one to GPU, while x16 PCie slots are limited to one in CPU and GPU. The Nvidia/Mellanox ConnectX-7 SmartNIC provide such connection @PCIe Gen5 speed, to achieve 400Gb/s



Fig.3.4. System architecture with PCIe switches





per interconnect link. This configuration optimizes GPU direct communications, but requires one NIC per GPU, even if 400Gb/s is not required per GPU. A last option is to add PCIe switches to provide direct connection between GPU and NIC as described in the following Fig.3.4.

This architecture removes the constraints on CPU, GPU and NIC and optimizes GPU direct communications as well. The number of NICs also depends on the bandwidth required per processor and on the technology of the NIC. It is a good compromise for performance, but the development cost and the shop cost are higher due to the development of the board for PCIe switch.

3.2 IDV-A Target Platform

IDV-A Target Platform In the current Textarossa project definition, Atos no longer develops of a specific GPU blade for IDV-A but will adapt one GPU blade under development for the two-phase cooling solution. The selected platform is the last generation BullSequana XH3000, which is compatible with OpenSequana blades (Fig.3.2.1). This cabinet offers 38 front flexible slots for compute and switch blades, with flexible cabling between compute nodes and interconnect switches and up to 147 kW DC available per rack. Indeed, introducing a new cooling technology on the market is considered only with new generation. It is interesting because the rack provides 7l/min flowrate per blade and can cool 600W GPU with water @40°C at rack input, but cooling GPU over 700W becomes a challenge with same input temperature and lower GPU max Tcase.



Fig.3.2.1. BullSequana XH3000 cabinet

Two GPU blades are under development and candidate for Textarossa IDV-A. As the cooling adaptation will require more than 1U, the new blade will use several slots and embed the heat exchanger between the InQuattro two-phase liquid and the BullSequana XH3000 native liquid.

Both candidates use a host motherboard with two Intel Sapphire Rapids processors (8 DDR5 DIMMs per processor, three x16 Gen5 PCIe lanes per processor, one Baseboard Management Controller (BMC) with Redfish interface and sharing one 1GbEthernet link with the host.





The first candidate connects 4 Intel Ponte Vecchio to this motherboard, according to the system architecture with CPU between GPU and NIC as described before. The next figure details this architecture (Fig.3,2,3).



Fig.3.2.3. Architecture of GPU blade with Intel Ponte Vecchio

This blade is connected to interconnect with different types of NICs: BXIv2 (100Gb/s), HDR (200Gb/s) or NDR (400Gb/s). As IDV-A consists in a single blade, there is no need of access to interconnect.

The second candidate connects 4 Nvidia H100 to this motherboard, according to the system architecture with CPU between GPU and NIC as described before. The next figure details this architecture (Fig.3.2.4).



Fig.3.2.4. Architecture of GPU blade with Nvidia Hopper

This blade is connected to interconnect only with Nvidia/Mellanox ConnectX7, and the NIC must be present in the configuration even when the high-speed interconnect is unused, because this component is mandatory for PCIe switching between CPU and GPU. As these two candidates are developed internally for Atos roadmap, it is not possible to change boards and add additional sensors. Then the power management software tool must run on an extra microcontroller to get access to the



sensors added by InQuattro. By default, the BMC can read the consumption of the entire node with a Redfish command. The consumption of GPU is accessed Infiband. The selection of the best candidate has been discussed during co-design meetings.

Considering the hardware part, the blade with Intel GPU is preferred because it is simpler, with free samples from Intel for CPU and GPU, and from Atos for BXIv2 NIC (option of the preliminary cost estimation in the project proposal). The blade with Nvidia GPU is more complex and more expensive, with GPU and NIC samples to be bought from Nvidia).

From point of view of exscale compute node the following Tab.3.2.1 shows the performance of the two GPU system architectures based on GPU Nvidia H100 and Intel Ponte Vecchio (PV), in terms of energy efficiency.

Single Compute node	GPU FP32 [TFlops/s]	GPU TDP [Watt]	CPU HPL [TFlops/s]	CPU TDP [Watt]	Exascale node [mW/GFlops/s]
Bi-socket-4xH100	4 x 60 = 240	4 x 700 = 2800	2 x 5 = 10	2 x 500 =1000	15
Bi-socket - 4xPV	4 x 45 = 180	4 x 600 = 2400	2 x 5 = 10	2 x 500 =1000	17

The two GPUs, Nvidia and Intel, show an energy efficiency suitable for exscale systems and both are candidate for the IDV-A platform. Considering the software and application users, most partners have already ported their application to CUDA, and the final budget of Textarossa budget is not compatible with porting the application to Intel OneAPI. After deliberations and verification that the extra cost can be funded, we decided to select the Nvidia blade.

There is no deviation for this analysis of GPU platform requirements, but the selection of Nvidia blade instead of Intel blade has an impact on the development of the prototype in T5.1 of WP5.

3.2.1 IDV-A platform requirements *IDV-A hardware specifications*

The IDV-A 1U blade enclosure corresponds to a 675 mm x 668 mm x 43.7 mm (LxWxH) cuboid (Fig.3.2.1.1 1). It includes:

- One Nvidia Redstone-Next board with 4 SXM5 H100 GPUs.
- One Atos C4E board with 2 x Intel Sapphire Rapids CPUs (sku TBD)
- RAM of 4 GB/core DDR5-4800
- 4 x Nvidia/Mellanox ConnectX7 for intra-node GPU interconnection
- 1 NIC for inter-node communication.







Fig.3.2.1.1. Top of view of the IDV-A blade (a). Front view (b)

IDV-A software specifications

The IDV-A stack software for Intel Xeon (X86_6464) platform is as follow:

- Sys.Op.: Linux RHEL/CentosStream/Rocky v.8.x/9.x
- Developer Toolchains: GCC (9.x/10.x/11.x), LLVM (12.x/13.x), glibc (2.x), and binutils (2.x)
- Python 3.x
- Development Libraries
- NVIDIA CUDA 11
- Intel OneAPI toolkits

Intel OneAPI Toolkit is oneAPI is an open, free, and standards-based programming system that provides portability and performance across accelerators and generations of hardware. oneAPI consists of a language and libraries for creating parallel applications:

- SYCL: oneAPI's core language for programming accelerators and multiprocessors. SYCL allows developers to reuse code across hardware targets (CPUs and accelerators such as GPUs and FPGAs) and tune for a specific architecture
- oneDPL: A companion to the DPC++ Compiler for programming oneAPI devices with APIs from C++ standard library, Parallel STL, and extensions.
- oneDNN: High performance implementations of primitives for deep learning frameworks
- oneCCL: Communication primitives for scaling deep learning frameworks across multiple devices
- Level Zero: System interface for oneAPI languages and libraries
- oneDAL: Algorithms for accelerated data science
- oneTBB: Library for adding thread-based parallelism to complex applications on multiprocessors
- oneVPL: Algorithms for accelerated video processing
- oneMKL: High performance math routines for science, engineering, and financial applications
- Ray Tracing: A set of advanced ray tracing and high-fidelity rendering and computation routines for use in a wide variety of 3D graphics uses including, film and television photorealistic visual effects and animation rendering, scientific visualization, high-performance computing computations, gaming, and more.





REFERENCE

[D1.1,2021] Textarossa Deliverable: Gap Analysis (2021)

[*Costanzo*,2021] Manuel Costanzo et al. "Comparison of HPC Architectures for Computing All-Pairs Shortest Paths. Intel Xeon Phi KNL vs NVIDIA Pascal". In: Computer Science {CACIC 2020. Vol. 1409. 2021, pp. 37{48. doi:10.1007/978-3-030-75836-3_3.

[*Costanzo,Rucci,2021*] Costanzo, Manuel & Rucci, Enzo & Garcia, Carlos & Naiouf, Marcelo. (2021). "Early Experiences Migrating CUDA codes to oneAPI". <u>https://doi.org/10.48550/arXiv.2105.13489</u>.

[OAM Spec 1.5] <u>https://www.opencompute.org/documents/ocp-accelerator-module-design-</u> specification-v1p5-final-20220223-docx-1-pdf

[NVIDIA H100, 2022] NVIDIA, "NVIDIA H100 Tensor Core GPU Architecture", 2022.

[*Frontier,2022*] <u>https://www.nextplatform.com/2021/10/04/first-look-at-oak-ridges-frontier-</u> exascaler-contrasted-to-argonnes-aurora/).

[*Argonne,2022*] <u>https://www.hpcwire.com/2021/09/08/how-argonne-is-preparing-for-exascale-in-2022/</u>.