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

WP6 Applications and Use cases

D6.1 Evaluation plan

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: D6.1 Evaluation plan

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	D6.1
WP No:	WP6
WP Leader:	PSNC
Due date:	M18 (September 30, 2022)
Delivery date:	05/10/2022

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

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

WP6 Applications and Use cases

Deliverable number:	D6.1
Deliverable title:	Evaluation plan
Due date:	M18
Actual submission date:	05/10/2022
Editor:	Michał Kulczewski
Authors:	Berenger Bramas, Laura Cappelli, Sebastian Ciesielski, Pasqua D'Ambra, Daniel Jaschke, Jens Krueger, Martin Kuehn, Michał Kulczewski, Alessandro Lonardo. Ariel Oleksiak, Alice Pagano, Cristian Rossi, Sergio Saponara, Francesco Simula
Work package:	WP6
Dissemination Level:	Public
No. pages:	27
Authorized (date):	30/09/2022
Responsible person:	Michał Kulczewski
Status:	Plan Draft Working [✓ Final] Submitted Approved

Revision history:

Version	Date	Author	Comment
0.1	2022-07-05	Michał Kulczewski	Draft structure
0.2	2022-08-24	Michał Kulczewski	Executive summary, Future work, ready for partner's contribution
0.3	2022-08-29	Martin Kuehn, Pasqua D'Ambra, Michał Kulczewski	RTM, CNR contributions and tables update
0.4	2022-09-10	Sergio Saponara	CINI-UNIPI contribution
0.5	2022-09-12	WP6 partners	First version for WP1
0.6	2022-09-15	INFN	Update
0.7	2022-09-21	Alessandro Lonardo	3.1 update

0.8	2022-09-21	Michał Kulczewski	Ready for QC
0.9	2022-09-23	Pasqua D’Ambra	Internal review
1.0	2022-09-29	WP6 partners	Addressing internal review comments, final update

Quality Control:

Checking process	Who	Date
Checked by internal reviewer	Pasqua D’Ambra	23/09/2022
Checked by Task Leader	Pasqua D’Ambra	23/09/2022
Checked by WP Leader	Michał Kulczewski	29/09/2022
Checked by Project Coordinator	Massimo Celino	30/09/2022

COPYRIGHT

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-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 content

List of Tables

List of Figures

Executive Summary

This deliverable provides initial results of task T6.4 to orchestrate evaluation of TEXTAROSSA solutions. The document presents an evaluation plan using a top-down approach. First, it gives a rough overview of the TEXTAROSSA applications and their involvement in different tasks using different approaches. Heterogenous resources will be used by eight applications; mixed-precision will be applied to three use cases; dynamic runtime systems will be evaluated by one library. Important to say, these use cases are coming from many different scientific domains, representing examples of AI, HDPA and HPC codes, which brings a broad vision of the needs to the project. As of M18, seven out of nine applications reached MS2 which means that the codes are well tested and ready to apply TEXTAROSSA solutions. Going bottom, a comprehensive list of TEXTAROSSA features (hardware, software and programming models) is presented to demonstrate which application is going to use these features. Going into more details, there is a summary of KPIs for each application, followed by their explanation and an overall evaluation plan for aforementioned three different application approaches. At the bottom, a detailed evaluation plan for each application is presented.

1 Introduction

Work performed in WP6 is essential to demonstrate the TEXTAROSSA outcomes in both, hardware and software perspective. The applications need to use these for the final evaluation of the project, but far more important is to come up with conclusions if and how new hardware and software development paradigms can improve computation and energy efficiency of applications representing different domains.

In the TEXTAROSSA we focus on applications related to AI (Artificial Intelligence), HDPA (High Performance Data Analytics) and HPC (High Performance Computing). Provided software represents quite a comprehensive set of different hardware used (CPU, GPU, FPGA), programming models (MPI for distributed computing/data exchange, CUDA, IntelOne API, etc.) and problems to be solved (sparse and dense linear algebra, iterative and direct solvers, etc.). Because of that, there is a different set of computational and energy efficiency metrics defined (KPI – key performance indicator) for each of the applications, though some naturally overlaps. In order to provide a high-quality evaluation plan for these many different applications, we applied a top-down approach to describe it. First, we start with a general overview of the applications, followed by a more detailed view on the TEXTAROSSA features and their usage by individual applications. Next, we discuss the KPIs and overall evaluation plan to finalise with details of evaluation of each of the use cases.

This document is organised as follows. Section 2 provides overview of the applications, mapping each to TEXTAROSSA hardware and software solutions. Section 3 details overall evaluation plan. Section 4 describes individual evaluation plan. In Section 5 we discuss how KPIs can be extended to take the outcomes of the WP1 and how we will update the plan during the scope of the project if necessary.

2 Applications

In WP6 there are 9 main applications representing AI, HDPA and HPC classes. The mathematical libraries developed by CNR and INRIA can be divided into separate modules, however they are referred to a single mathlib to keep things simpler. A high-level overview is given in Table 1. Eight out of nine applications will benefit from heterogeneous hardware resources, three of them plan to introduce mixed-precision, and one of them will benefit from dynamic runtime systems. However, some heterogeneous applications will consider applying mixed-precision and/or dynamic runtime systems, thus the number of use cases using different approaches may change during the scope of the project. It is worth mentioning that seven out of nine proposed applications have already accomplished MS2 – prototype applications are ready, albeit not yet integrated. The missing two under development and soon the prototypes will be available. The most important feature is that there is at least one application for each functionality (task) already available.

App name	Partner	Heterogeneous (T6.1)	Mixed-precision (T6.2)	Dynamic runtime (T6.3)	MS2
Smart cities	CINI	GPU, possibly FPGA (for mixed precision)	Yes (Posit)	No	Yes
Mathlib-CNR	CNR	GPU (CUDA)	Maybe	No	Yes
RTM	Fraunhofer	No	Yes (Posit)	No	Yes
HEP	INFN	GPU, possibly FPGA	No	No	No
NestGPU	INFN	GPU (CUDA)	No	Not planned	Yes
RAIDER	INFN	FPGA	Maybe	Maybe	Yes
TNM	INFN	GPU (CUDA)	No	No	No
Mathlib-INRIA	INRIA	GPU, FPGA	No	Yes (StarPU)	Yes
UrbanAir	PSNC	GPU (CUDA)	Planned	No	Yes

Table 1 Overview of WP6 applications

Before sketching an evaluation plan, it is of great importance to identify which use case is trying to benefit from proposed hardware or functionality. Table 2 provides a mapping between application and programming model, software/tools to be used and hardware to be exploited, which will be used for the final evaluation plan.

Application	Smart Cities	Mathlib-CNR	RTM	HEP	NEST-GPU	RAIDER	TNM	Mathlib-INRIA	UrbanAir
Programming models									
CUDA	Limited	Yes (MPI/CUDA tools for distributed/shared hybrid model)	Very limited	Yes	Yes (hybrid CUDA-MPI)	No	Yes	Planned to be used	Yes (MPI/CUDA)
FPGA	Yes	No	No	No	No	Yes	No	Planned to be used	No
IntelOne API	Limited	No	No	Yes	No	No	No	No	No
streaming models	No	No	No	No	No	Yes	No	No	No
task-based models	No	No	No	No	No	Maybe	No	Yes	No
Software/tools									
GPU power modelling	Yes	Planned to be used	No	No	Planned to be used	No	Planned to be used	Yes	Planned to be used
FGPA power modelling	Yes	No	No	No	No	Planned to be used	No	Yes	No
Tools for Posit	Yes	No	Yes	No	No	No	No	No	No
Tools for mixed-precision	Yes	No	Yes	No	No	Planned to be used	No	No	Planned to be used
inter-FPGA comm SW stack	No	No	No	No	No	Yes	No	Maybe	No
Hardware									
IP with mixed precision for AI	Yes	No	No	No	No	Maybe	No	No	No
Secure Crypto IP	No	No	No	No	No	No	No	No	No
IP with data compression	Yes	No	No	No	No	Maybe	No	No	No
IP with low latency FPGA	No	No	No	No	No	Yes	No	No	No
IP for fast task scheduling	No	No	No	No	No	Maybe	No	No	No

FPGA hardware counters	No	No	No	Maybe	No	Maybe	No	Yes	No
IDV-A	Maybe	Planned to be used	No	Maybe	Yes	No	Planned to be used	Planned to be used	Planned to be used
IDV-E	Maybe	No	No	Maybe	Yes (CPU version)	Planned to be used	No	Planned to be used	No

Table 2 Hardware, software tools and programming models

3 Evaluation plans

The evaluation will be based on benchmarking KPIs defined individually per each application. However, a more global approach can be applied by using a common set of indicators defined within WP1 and WP6. Some of the KPIs are easier to measure, such as time-to-iteration or time-to-solution, as they require only small integration to the code, and no external tools or access to hardware performance counters is required. Some of them, such as energy efficiency, need additional tools and sensors enabled for the underlying hardware infrastructure.

The KPIs are related to the solved problems, and some of them are common for several applications. Individual KPIs are summarised in Table 3, however an update is expected with the next D6.2 deliverable after these are liaised with WP1 outcomes.

App name	KPI - computational efficiency	KPI - energy	KPI - accuracy
Smart cities	execution time/speedup on GPU vs. scalability vs. accuracy	Power model on target GPU and on FPGA	Yes
Mathlib-CNR	execution time/speedup/strong and weak scalability; number of iterations to a fixed accuracy/time per iteration for iterative solvers	Iterations/Watt; Dofs/Watt	Yes (user's parameter dependent)
RTM	increase memory bandwidth, then maybe increase Flops	No	No
HEP	Events / s	Events / Watt	No
NestGPU	Simulated ms / s	SUPs/Watt	No
RAIDER	MEvents / s	Mevents/Watt	Yes
TNM	Qubits / s Gate / s	Qubits / Watt Gates / Watt	No
Mathlib-INRIA	Flops/s | Interactions/s	Flops/Watt, Iterations/Watt	No
UrbanAir	iterations/s, simulated time/s	Iterations/Watt	No

Table 3 Individual KPIs

The KPIs for computational efficiency are:

- Time per iteration, used by iterative solvers where performance can be judged based on how many seconds are needed per each iteration and number of iterations to a user's defined accuracy.
- Simulated time/s (or timesteps/s), used by the solvers which iterates through simulated time, the more timesteps are calculated within one second the better the performance is.
- Interactions/s, used by n-body simulations where the number of interactions between particles is representative of the performance.
- Events/s, used by trigger systems in physics experiments where we refer to "event" as the instantaneous physical situation or occurrence associated with a point in spacetime,

characterised in our systems by different information obtained through several physical detectors.

- Qubits/s (and Gate/s), with an equal fixed set of convergence parameters for a quantum simulation with tensor networks method, e.g. fixed bond dimension, the performance can be evaluated looking at what is the size of the system n, in terms of number of qubits, that can be simulated within a second. In some other application, for a given n-qubit system, the performance can be evaluated looking at the number of quantum gates within a second that can be executed.
- FLOP/s, a general performance KPI to indicate how many floating-point operations per second can application achieve.

The KPIs for energy efficiency are similar to the computational ones, except that it is measured for every watt of power consumed.

Accuracy KPI: accuracy in detection and classification for the target application (i.e. accuracy = number of the correct predictions divided by total number of predictions) vs. computational complexity and vs. used arithmetic; accuracy in iterative linear solvers (i.e., number of correct digits in the solution, as required by users).

In the following subsections a common strategy for benchmarking and evaluating heterogeneous, mixed-precision and dynamic runtime systems applications is discussed.

3.1 Heterogeneous applications

The complexity of a heterogeneous computing platform such as the TEXTAROSSA project requires the use of a common methodology to perform power measurements, in order to manage a trade between computational power and energy consumption. For this purpose, a dedicated working group has been created within the project. The complete results of its activities are summarized on the technical document "Methodology for Power Measurement in the TEXTAROSSA Project". In the following we summarise the most relevant topics of the document.

3.1.1 CPU

Textarossa project will deal with two kinds of CPU architectures: x86_64 (AMD Milan/Rome, Intel Sapphire Rapids) and ARM V8.2 64 bit (AMPERE Altra Max).

3.1.1.1 x86_64 Architectures

Most modern processors, including Intel processors, provide Running Average Power Limit (RAPL) interfaces for reporting the accumulated energy consumption of various power domains of the CPU chip, attached DRAM and on-chip GPU. The update interval of the RAPL energy counters is approximately one millisecond. The RAPL energy reporting feature has been available for many generations on Intel SoC products, and energy reporting is standard practice for the industry. Intel processors utilise this energy information for internal SoC management purposes, such as control of SoC power limits in association with Intel® Turbo Boost Technology power limit settings within the SoC. This RAPL energy data is exposed to the platform via the host-software-accessible model specific registers (MSRs) such as *MSR_PKG_Energy_Status* and *MSR_PP0_Energy_Status*. This allows software to use the RAPL energy data for observation, telemetry, and/or inputs to platform-level power or thermal control algorithms [1]. The RAPL features described above are also available for AMD processors from family 17h on.

RAPL readings are highly correlated with plug power, promisingly accurate enough and have negligible performance overhead. Experimental results suggest RAPL can be a very useful tool to measure and monitor the energy consumption of servers without deploying any complex power meters. [2]

RAPL supports multiple power domains. The RAPL power domain is a physically meaningful domain (e.g., Processor Package, DRAM etc) for power management.

Figure 1 illustrates the hierarchy of the power domains graphically.

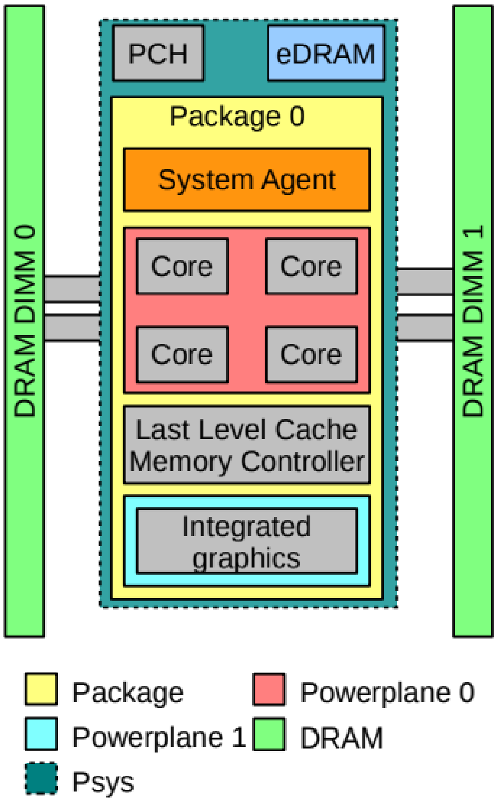

Figure 1 Power domains supported by RAPL [3]

Each power domain informs the energy consumption of the domain, allows to limit the power consumption of that domain over a specified time window, monitors the performance impact of the power limit and provides other useful information, that is, energy measurement units, minimum or maximum power supported by the domain [3].

RAPL provides the following power domains for both measuring and limiting energy consumption:

- Package: Package (PKG) domain measures the energy consumption of the entire socket. It includes the consumption of all the cores, integrated graphics and also the uncore components (last level caches, memory controller).

- Power Plane 0: Power Plane 0 (PP0) domain measures the energy consumption of all processor cores on the socket. RAPL does not support measuring the power consumption of individual CPU cores.

- Power Plane 1: Power Plane 1 (PP1) domain measures the energy consumption of processor graphics (GPU) on the socket (desktop models only).

- DRAM: DRAM domain measures the energy consumption of random-access memory (RAM) attached to the integrated memory controller. Deviations up to 20% from actual measurements have been reported for this particular domain, with a strong dependence on the specific processor architecture [4].

- PSys: Intel Skylake has introduced a new RAPL Domain named PSys. It monitors and controls the thermal and power specifications of the entire SoC and it is useful especially when the source of the power consumption is neither the CPU nor the GPU.

As Figure 1 suggests, PSys includes the power consumption of the package domain, System Agent, PCH, eDRAM and a few more domains on a single socket SoC. For multi-socket server systems, each socket reports its own RAPL values (for example a 2-socket computing system has two separate PKG readings for both the packages, two separate PP0 readings, etc). The support for different power domains varies according to the processor model, as energy unit used: the Sandy Bridge uses energy units of 15.3 microjoules, whereas Haswell and Skylake uses units of 61 microjoules.

RAPL measurements are accurate, the correlation coefficient between RAPL and plug AC power values has been measured using the Stream benchmark on a Haswell processor, resulting in a value of 0.99 [3].

Linux supports RAPL since from kernel 3.14, access to RAPL data is possible through several mechanisms, such as reading files under /sys/class/powercap/intel-rapl/intel-rapl:0, using the perf_event interface (e.g. sudo perf stat -a -e "power/energy-cores/" executable) or using raw-access to the underlying MSR registers provided by the msr kernel module.

Several energy profiling tools using the RAPL infrastructure are currently available, we selected likwid-powermeter as reference power measuring tool for CPU tasks. likwid-powermeter is part of the Likwid toolsuite [5] of command line applications and a library for performance-oriented programmers. It works for Intel, AMD, ARMv8 and POWER9 processors on the Linux operating system. There is additional support for Nvidia GPUs.

3.1.1.2 Ampere Altra Max

According to the technical documentation provided by the manufacturer, this implementation of the ARM V8.2 64-bit architecture does not provide any RAPL-like facility for fine-grained power measurement.

There four high power domains for Altra / AltraMax processors:

- PCP power domain for CPU cores and mesh interconnects
- SoC power domain for SoC blocks, memory and PCIe controllers
- RCA power domain for PCIe/CCIX controllers
- DDR4 power domain for memory IOs and DIMMs

The Altra Max Processor Complex (PCP) features include:

- 128 Arm v8.2+ 64-bit CPU cores at up to 3.00 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 Interconnect (CMI):

Only PCP and SoC power domains are accessible using the Linux HWMON infrastructure, either reading the corresponding /sys/class/hwmon/hwmon0/* entries of the filesystem, or using the sensors command.
An alternative method is to use the BMC infrastructure, that provides the following power data:

- PCP power domain for CPU cores and mesh interconnects
- SoC power domain for SoC blocks, memory and PCIe controllers
- DDR4 power domain for memory IOs and DIMMs

3.1.2 GPU

In the context of the project only NVIDIA GPUs will be taken into account.
To perform power monitoring on NVIDIA GPUs, a useful tool is represented by the NVIDIA Management Library (NVML): a C-based programmatic interface for monitoring and managing various states within NVIDIA GPU devices. NVML is delivered in the NVIDIA vGPU software Management SDK, which enables third party applications to monitor and control NVIDIA physical and virtual GPUS that are running on virtualisation hosts. Using the NVML APIs, we have experimented with a simple tool able to measure the power consumption of a CUDA kernel in specific points of the device code.
However, the power value obtained through the NVML APIs is updated every ~20 ms. Thus, this sampling interval is not suited for a precise evaluation of the power consumption profile of a CUDA kernel with a short execution time.

3.1.3 FPGA

TEXTAROSSA adopts the XIlinx U280 as reference platform for the project.
Referring to FPGAs' power monitoring, POLIMI developed a methodology to deploy into generic hardware design online power monitors, capable of periodic power estimate. They evaluated 2 possibilities of implemented power monitoring:

- Software power monitors: applications providing online power monitoring in cases where platform RTL description is not accessible, at the cost of a non-negligible performance overhead, low accuracy and limited temporal resolution for the power estimate.
- Hardware power monitors: dedicated hardware delivering highly accurate power estimates at high temporal resolution and without performance overhead at the cost of changing the RTL description of the computing platform.

More information about this methodology can be found in [6].

In addition to this, Xilinx® provides a suite of software tools that can assess power supply requirements of the device throughout each stage of the design cycle. For example, Vivado® power analysis feature performs power analysis through the stages of: post-synthesis, post-placement, and post-routing. Also, Xilinx Runtime library (XRT) Linux kernel driver *xclmgmt* binds to management physical function and handles the access to in-band sensors (temperature, voltage, current, power etc.). In this context, POLIMI, UNIPI and INFN have started working together in order to characterise the power consumption of the IPs developed in TEXTAROSSA project.

3.2 Mixed-precision applications

The increasing interest in complex AI and video applications involving large convolutional networks require a trade-off between the low complexity of integers and the high accuracy of floats. To this aim new arithmetic types, like Bfloat and Posits, will be considered. The KPIs will consider not only the accuracy in detection and classification for the target application (i.e. Accuracy = Number of correct predictions divided by Total number of predictions) but also computational complexity and power model.

The target applications are:
- by UNIPI-CINI, some AI and video classification applications will be used for smart cities surveillance services such as man-down (detection from camera acquired images of people laying down, useful for people rescuing in case of natural disasters, wars,…) and people detection and social distancing check and covered-face detection (useful for Covid-19). These applications are further described in Section 4.1.
- by Fraunhofer supported by UNIPI-CINI, an optimised Reverse Time Migration (RTM) algorithm that is used for oil and gas exploration in seismic imaging. This application is further described in Section 4.3.

These applications will be tested on GPU (e.g. NVIDIA GPU like T4 and Jetson AGX) and FPGA. The complexity of a heterogeneous computing platform requires the use of a common methodology to perform power measurements, to manage a trade between accuracy, computational power and energy consumption. To this aim, similarly to what described in 3.1, the following tools will be considered.

For GPUs: To perform power monitoring on NVIDIA GPUs, a useful tool is represented by the NVIDIA Management Library (NVML): a C-based programmatic interface for monitoring and managing various states within NVIDIA GPU devices. NVML is delivered in the NVIDIA vGPU software Management SDK, which enables third party applications to monitor and control NVIDIA physical and virtual GPUS that are running on virtualisation hosts.

FPGAs: Referring to FPGAs' power monitoring, CINI-POLIMI has developed a methodology to deploy into generic hardware design online power monitors, capable of periodic power estimate. This methodology will be applied to the Posit Processing Unit designed by UNIPI-CINI to be integrated n FPGA technology.

3.3 Dynamic runtime system applications - INRIA

The performance of applications based on runtime systems is impacted by 1) the way the applications are parallelized, 2) the internal implementation of the runtime systems, and 3) the scheduling decisions taken at runtime to distribute the tasks overs the processing units. Concerning 1) the performance is clearly application dependent and is left aside from the current description. For 2), the internal implementation of a runtime system can be evaluated by measuring its overhead and its capacity to potentially hide data movement with computation when possible. We do not expect that including a new hardware device will change the quality of an existing runtime system, therefore we recommend performing sanity check but do not consider it relevant to include these aspects in the benchmarking in TEXTAROSSA. This is why 3) is certainly the more important criterion. To evaluate the scheduling, we propose to study the makespan and the amount of memory data transfer. With this aim, we propose to benchmark runtime system-based applications with three metrics: makespan (duration of the execution) in seconds, amount of memory transferred in GB, and occupancy of the processing units in percentage.

4 Individual evaluation plans

In Section 2 a high-level overview of applications is presented, emphasising which TEXTAROSSA hardware and software outcome will be demonstrated for each of the use cases. The application represents wide range of scientific domains and problems that are solved, and this is the reason to introduce individual evaluation plan. For each application it is explained briefly: i) the reason to improve, ii) the key performance indicators related to accuracy (if needed), computational and energy efficiency, and iii) the evaluation plan.

4.1 Smart cities – CINI

Why to improve

CINI UNIPI is working on some AI and video classification applications that will be used for smart cities surveillance services such as:

- man-down (detection from camera acquired images of people laying down, useful for people rescuing in case of natural disasters, wars,…)

- people detection and social distancing check and covered-face detection (useful for Covid-19 preventi*on*).

The need is improving the trade-off among computational complexity, frame-rate of the application, accuracy of the detection and classification.

KPIs

KPIs will be:

- the achieved frame-rate,

- accuracy of the detection and classification (i.e. Accuracy = Number of correct predictions divided by Total number of predictions

- power consumption of the application implemented on the target platform

Evaluation

The evaluation will be carried out porting the algorithms in different platforms: e.g. GPU (T4 or Jetson-AGX) and FPGA and using the tools discussed in paragraph 3.2.

4.2 MathLib – CNR

Why to improve

CNR is working on a mathematical software library for hybrid architectures, featuring NVIDIA GPUs at node level [7][8]. Mathematical software libraries provide a large resource for high-quality, reusable software components upon which applications can be rapidly constructed. They are building blocks for solving main mathematical problems, including radically new algorithms and methods at a low level, that domain scientists can transparently reuse in form of basic components with very little need of specific mathematical and computer science expertise. CNR is developing computational kernels

required in sparse matrix computations and iterative linear solvers which are widely applied in Scientific Computing and Data Analysis. Main focus is on node-level efficiency and scalability when multiple nodes are needed for computations whose dimensions largely exceed the memory resources of a single computing node, such as those stemming from leading-edge HPC applications.
The kernels include:

- Sparse matrix – vector multiplication (SpMV);
- Sparse matrix power kernel (SpMPK);
- Sparse matrix – matrix multiplication (SpMM);
- Maximum Weight Matching in undirected graphs (MWM);
- Communication Avoiding Conjugate Gradient method (CACG);
- Algebraic MultiGrid preconditioners (AMG).

KPIs

Main objective is a sustained performance and scalability for solving problems at extreme scales. Therefore, main parameters will be execution times and speedup in strong and weak scalability regime. Concerning iterative linear solvers, specific parameters also include number of iterations to reach a given accuracy and execution time per iteration for increasing number of unknowns and parallel cores. Memory footprint is also an issue to efficiently face extreme scales, therefore key parameter will be also memory requirements of algorithm implementations. In the same way, parameters related to energy efficiency will include number of iterations per Watt and number of problem unknowns (dofs, degree of freedom) per Watt.

Evaluation

The plan for final evaluations on some well-known benchmarks, such as sparse linear systems coming from discretization of scalar partial differential equations of Poisson-type, includes the use of some clusters of hybrid nodes embedding Nvidia GPUs and also, when possible, the IDV based on Nvidia GPUs from WP5. Main risks are related to the usage of tools, eventually proposed in the project, for energy consumption measurements, and the need to have access, at a reasonable stage of the project, to the IDV-A based on Nvidia GPUs.

4.3 RTM – FRAUNHOFER

Why to improve

Reverse Time Migration (RTM) is used for oil and gas exploration in seismic. Migration algorithms usually need to digest input shot data up to the terabyte range to create 3D images. Days and weeks of cluster compute time are common. For this reason, the users of seismic algorithms are sensitive to compute time. Other migration algorithms approximate the wave equation to reduce the compute time, e.g., the Kirchhoff migration uses the high frequency ray approximation. However, these approximations have drawbacks, e.g. when it comes to resolve steep dips in salt domes. RTM is much better here in resolution but much more expensive in compute time as well. As TTI RTM algorithms are even more expensive than Isotropic or VTI RTM algorithms, the latter two are the most frequently used RTM algorithms in practice. Isotropic and VTI RTM algorithms are both quite memory bound.

Here calculations are done typically in 32bit floating point. Reduction of data precision (at least partial, so mixed precision) to 16 bits could half the consumed memory bandwidth and double the throughput of the kernel. So the gap in cost between RTM and cheaper migration methods is reduced.

KPIs

The drawback of using reduced or mixed precision might be a reduced image quality. As the better image quality of RTM versus cheaper methods is the main reason to use RTM in the first place, retaining an acceptable level of image quality is crucial.

This work analyses the possibility to reduce the floating-point precision at least in parts of the RTM algorithm to increase the throughput of the algorithm. Important boundary conditions that need to be kept up are the accuracy and the stability of the numerical results.

Evaluation

Simple test examples are created. From these images are computed in different reduced floating point precision formats and mixed floating point precision formats. These images are compared to images which are computed fully in single precision. Seismic experts will evaluate the images.

Further the numerical stability versus the time step size will be evaluated. Here a forward propagation of the wave signal and the total energy within the 3D volume for each time step will be computed. In the stable time step region the total energy should stay constant over simulated time. Different time steps will be evaluated to determine the stable time step region numerically.

4.4 HEP - INFN

Why to improve

The need to be able to execute scientific code on heterogeneous architectures is evident in many domains, from High Energy Physics, genomics, astrophysics to medical physics. The extrapolated needs for the next decades surpass what standard CPU evolution can allow. The most promising path to affordable computing lies in the utilization of better performance/cost computing solutions, like those offered by accelerator technologies; on top of this, the same technologies are expected to be present in most HPC centres, and available to users only if their codes can be executed efficiently. The main obstacle is the difficulty to redesign algorithms in a suitable manner for each different architecture, due to the lack of knowledge and of manpower.

For the TEXTAROSSA project, two high energy physics applications were identifies:

- a track reconstruction algorithm for the CMS detector developed by the Patatrack team [9]
- the CLUE algorithm, a cluster algorithm for high granularity calorimeters [10]

KPIs

In the kind of above applications, the measurement of the latency as the delay between the invoking an operation and getting its response is not a representative metric as much as the throughput, i.e., the number of computing tasks per time unit. Therefore, the KPI considered for the HEP applications is a throughput metric: the reconstructed events per second. In particular, for the track reconstruction algorithm, the reconstructed events are the particle tracks in the detector; instead, for the CLUE algorithm, the reconstructed events are the assignment of a cluster to each point.

Evaluation

The goal is to obtain a single heterogeneous software per application that can be run in parallel on multiple backends, taking advantage of the characteristics of each architecture. To evaluate the results, the comparison of the performance obtained using the serial code running on the CPU versus the parallel and heterogeneous code running on multiple backends (CPU, GPU, FPGA) is of main interest. The performance is measured as the time spent to reconstruct N events.

4.5 NEST-GPU - INFN

Why to improve

The main reason for developing a CUDA version of the CPU-only NEST neural simulator was to tap into the large floating point compute resources available on NVIDIA GPUs, in order to speed-up the integration of the large systems of differential equations as required by the setup and dynamics simulation of complex neural networks that an in-silico neurophysiology experiment implies. Any improvement in this regard can either push the size and the complexity of what can be achieved by such experiments on non-extreme scale HPC platforms. Moreover, shrinking the power envelope could even demonstrate the feasibility for NEST-GPU to drive an embodied agent, which would be useful for robotics applications.

A more thorough description of the NEST-GPU application can be found in [11], while an up-to-date comparison to the CPU-only sibling application NEST running on a cluster and using MPI communications can be found in [12].

KPIs

As mentioned, the KPIs are those related to the size, complexity and achievability (which usually means bringing down the timeframe of a simulation to a manageable level) of a neurophysiology experiment, therefore time-to-solution (as how long it takes to simulate *e.g.* 1s of a neural network of predefined size), the synaptic activity (the ratio of synaptic events to the actual runtime) and the energy-to-solution (as the energy dissipated throughout this runtime) will be considered.

Evaluation

The evaluation of the mentioned KPIs is obtained defining some average sized network that can be representative of a sufficiently broad set of experiments and simulating via NEST-GPU 1s of activity of such network (possibly a little longer if we concede some warm-up period to let the system reach a steady state) on the reference platform while measuring the elapsed time, the number of synaptic events and the power consumption throughout.

4.6 RAIDER – INFN

Why to improve

Real-time (also called “online” in the specific context) particle identification (PID), or partial particle identification (*e.g.*, electron identification), is a critical task in High Energy Physics experiments: it enhances the suppression of background physics events, allowing to keep the bandwidth that data acquisition systems must forward to the analysis pipeline within a manageable level.

In this section, we refer to “event” as the instantaneous physical situation or occurrence associated with a point in spacetime, characterised in our systems by different information obtained through several physical detectors.

The system implementing the PID task must face two main requirements:

1. processing latency, often bounded to a few microseconds or less;
2. processing throughput, which can be in the order of 10^7 events per second.

FPGA devices are good candidates to be used as processing nodes to implement a dedicated computing architecture to perform PID, as these devices allow the design of AI algorithms through HLS tools, and the implementation of data transport (with support for a wide set of physical and transport layers protocols) and processing stages characterized by a highly predictable and low latency. A low-latency direct interconnection between FPGA nodes allows:

+ to scale the system to meet the throughput requirements, deploying multiple dedicated computational units (CUs) on several boards;
+ to gather data streams from different detectors, possibly processed according to a multi-layer architecture performing a distributed PID task.

A desirable development of such an architecture has been identified in the CERN NA62 experiment: in fact, RAIDER application seems suitable for the timing requirements of the Level 0 of the NA62 trigger, allowing a possible implementation of a PID system based on the use of neural networks trained for ring reconstruction over the events coming from the NA62 RICH detector.

A more detailed description of this workflow can be found in [13].

KPIs

Besides the processing latency per event that must be less than the experimental requirement, and so it should be considered more as a prerequisite rather than a KPI, relevant KPIs for the RAIDER application are, for given values of accuracy and purity of the (partial) PID task:

1. the number of processed events per second, *i.e.*, the throughput;
2. the number of processed events per watt.

Evaluation

The measurement of the Mevents/s KPI is trivial.

A joint activity with the POLIMI team, aimed at instrumenting the RTL code of the communication IPs and processing kernels in order to measure their power consumption, has just started. Since this methodology has already been used successfully in the past by the POLIMI team any particular risk at the moment for the measurement of the Mevents/W KPI is identified.

4.7 TNM – INFN

Why to improve

Tensor network methods consist of techniques that represent the quantum state of *N* qubits as a series of tensor contractions. By trading some accuracy, this enables for example quantum circuit simulators to handle circuits with many qubits that would not be feasible to be simulated with exact methods because of the exponential growth of the Hilbert space. However, depending on circuit topology and depth, this can also get prohibitively expensive. This highlights the need for tensor network methods to be executed on heterogeneous architectures to efficiently exploit parallel computing and powerful GPU computation. Before moving towards mixed-precision methods, the effect of running different precisions for a complete simulation are reported.

KPIs

In the kind of above applications several KPIs can be considered for estimating computational efficiency. For a quantum simulation with tensor network methods one can evaluate the performance by looking at the number of qubits that can be simulated per second with a fixed set of convergence parameters as the bond dimension between the link in the tensors. Another KPI is the number of gates per second that can be executed in a simulation of a quantum system with a given size. A possible direction to evaluate the KPI for energy consumption is to follow up on our previous work, which compares the energy consumption of a quantum circuit on a quantum processing unit against the same quantum circuit running with tensor networks [14].

Evaluation

To evaluate the results, the performance of the software executed by using the serial code running on the CPU versus the parallel and heterogeneous code running on CPU and GPU will be compared.

4.8 Chameleon (Mathlib-INRIA)

Why to improve

Chameleon is a dense linear solver based on StarPU, i.e. it is parallelised with a task-based method and relies on classical Blas functions in the tasks. Consequently, its performance is critically tied to the scheduling of the tasks and the raw performance of the Blas functions on the target processing units. Chameleon has been massively used on distributed heterogeneous computing nodes equipped with multiple GPUs. However, the study of a large-scale dense linear solver with FPGAs has never been done. This is why we want to use FPGA and see how this can improve performance and/or energy.

KPIs

Two main KPIs are FLOP per second and FLOP per watt because main interest is in evaluate speedup and energy efficiency can be obtained in running with FPGA .

Evaluation

FLOP per second can be obtained easily. The FLOPS per watt needs hardware counters.

4.9 ScalFMM (Mathlib-INRIA)

Why to improve

The fast multipole method is a well-known approach that allows for reducing the quadratic complexity when computing interactions in n-body problems. ScalFMM has been a pioneer by proving the first implementing FMM algorithm on top of StarPU, i.e. parallelised with a task-based method. ScalFMM can be executed on distributed computing nodes equipped with accelerator devices and used CPUs and GPUs concurrently. Said differently, when the GPUs are computing kernels for which they are efficient, we use CPUs at the same time for less GPU-friendly kernels. Currently, we implemented the two major FMM kernels with CUDA such that we can use the main common HPC architectures. However, we never study its energy efficiency or the use of FPGA.

KPIs

Two main KPIs are n-body interactions per second and n-body interactions per watt because main interest is in evaluate speedup and energy efficiency in running with FPGA. Energy efficiency will be also evaluated for the existing GPU version.

Evaluation

N-body interactions or FLOP per second is obvious. Whereas interactions or FLOP per watt needs hardware counters. Once these counters will be available, FPGA kernels for the P2P operators and benchmark will be evaluated. Comparisons when using GPUs or FPGAs to highlight the situations where one is better than the other will be carried out.

4.10 UrbanAir – PSNC

Why to improve

In the UrbanAir we deal with weather forecasting which then influences how pollutants are transported and dispersed within the cities. One of the challenges is to efficiently and effectively represent complex building structures which affect contaminants flow. To model the problem accurately, there is a need for vast of computational resources. In order to be able to simulate larger domains, we need to improve. CPU+GPU realisation on multiple nodes is considered to shorten execution time, and to increase in energy efficiency. Additionally, we want to investigate whether implying mixed precision can lead to increased efficiency by minimising communication time.

KPIs

The main part to be adapted to heterogeneous resources is an iterative solver, with the aim of dividing it into smaller kernels. Iterations/s and iterations/watt for respectively computational and energy efficiency will be considered.

Evaluation

The kernels will be benchmarked on currently available hardware for the baseline measurements. Iterations/s will be collected programmatically, while iterations/Watt need some energy measurement tools developed on WP4. The progress will be measured on a regular basis, on the available testbed, and at the end of the project it will be compared against IDV-A and project tools.

5 Future work

In this deliverable we discuss general and individual evaluation plan of the Textarossa features and uses cases. The next step is to benchmark each application with defined KPI, which will be the baseline measurements to compare with at the end of the project. The outcomes of this task will be described in the following deliverable – D6.2 Initial application benchmarks and results. The baseline measurements from WP1 benchmarking task shall serve for comparison and calculation of achieved improvements. Moreover, remaining outcomes of WP1 shall be taken into account to extend the proposed evaluation metrics. It is planned to derived such discussion in the next deliverable. It may be the case that the details of evaluation plan may require an update, such will be provided with the next deliverables.

6 References

[1] https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/advisory-guidance/running-average-power-limit-energy-reporting.html

[2] Khan, Kashif & Hirki, Mikael & Niemi, Tapio & Nurminen, Jukka & Ou, Zhonghong. (2018). RAPL in Action: Experiences in Using RAPL for Power Measurements. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS). 3. 10.1145/3177754.

[3] Intel Corporation 2015. Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3, System Programming Guide. Intel Corporation

[4] Spencer Desrochers, Chad Paradis, and Vincent M. Weaver. 2016. A Validation of DRAM RAPL Power Measurements. In Proceedings of the Second International Symposium on Memory Systems (MEMSYS '16). ACM, New York, NY, USA, 455–470. https://doi.org/10.1145/2989081.2989088

[5] https://hpc.fau.de/research/tools/likwid/

[6] Cremona et al., Automatic identification and hardware implementation of a resource-constrained power model for embedded systems, Sustainable Computing: Informatics and Systems, Volume 29, Part B, 2021, 100467, ISSN 2210-5379, https://doi.org/10.1016/j.suscom.2020.100467.

[7] M. Bernaschi, P. D'Ambra, D. Pasquini, BootCMatchG: An adaptive Algebraic MultiGrid linear solver for GPUs, Software Impacts (Invited Paper). Vol. 6, 2020. https://doi.org/10.1016/j.simpa.2020.100041

[8] M. Bernaschi, P. D'Ambra, D. Pasquini, AMG based on Compatible Weighted Matching on GPUs, Parallel Computing. Vol. 92, 2020. https://doi.org/10.1016/j.parco.2019.102599

[9] Bocci A. et al, Heterogeneous Reconstruction of Tracks and Primary Vertices With the CMS Pixel Tracker, Frontiers in Big Data Journal, 2020

[10] Rovere M. et al, A Fast Parallel Clustering Algorithm for High Granularity Calorimeters in High-Energy Physics, Frontiers in Big Data Journal, 2020

[11] B. Golosio et al, Fast Simulations of Highly-Connected Spiking Cortical Models Using GPUs, Frontiers in Computational Neuroscience February 2021 (doi.org/10.3389/fncom.2021.627620)

[12] G. Tiddia et al, Fast Simulation of a Multi-Area Spiking Network Model of Macaque Cortex on an MPI-GPU Cluster, Frontiers in Neuroinformatics July 2022 (doi.org/10.3389/fninf.2022.883333)

[13] R. Ammendola et al, Progress report on the online processing upgrade at the NA62 experiment, 2022 JINST 17 C04002 (iopscience.iop.org/article/10.1088/1748-0221/17/04/C04002)

[14] Daniel Jaschke, Simone Montanegro, Is quantum computing green? An estimate for an energy-efficiency quantum advantage