Heterogeneous Hardware and Software Alliance

Towards an Architecture

1. Background

The **Heterogeneous Hardware & Software Alliance (HH&S)** is an initiative undertaken by the members of a Research & Innovation Project TANGO (**T**ransparent heterogeneous hardware **A**rchitecture deployment for e**N**ergy **G**ain in **O**peration) funded under the EU H2020 programme. The initiative aims to join efforts of organizations interested in the development of future technologies and tools to advance and take full advantage of computing and applications using heterogeneous hardware. Heterogeneous architectures have received considerable attention, as an efficient approach to run applications and deliver services, by combining different processor types in one system to improve absolute performance, minimise power consumption and/or lower cost. The impact of such heterogeneity on all computing tasks is rapidly increasing and consequently needs consideration.

The alliance will focus on all phases of HH&S, from design time, to enhanced execution, parallel programming and optimized runtime, and consider a number of factors such as energy, performance, real-time, data locality and security. This will enable a new way of development and execution of next-generation applications.

The alliance main goal is to create an organization in which anyone interested in the technological areas that are initially part of its scope can participate and collaborate pursuing a common objective: founding a common, open-source, and an extendable set of technologies and tools around the development for heterogeneous hardware, which can be viable for mass adoption utilizing technologies created by the alliance members, and attractive, easy to use and broader in scope and value.

2. Towards an Architecture

Because the impact of heterogeneity on all computing tasks is rapidly increasing, innovative architectures, algorithms, and specialized programming environments and tools are needed to efficiently use these new and mixed/diversified parallel architectures. The transitions to multicore processors, GPU computing, Reconfigurable Acceleration, and Hardware as a Service (HaaS) cloud computing should be viewed as a single trend. As the market for heterogeneous architectures/multicore processors in embedded applications has begun to move into the product deployment stage, the need for software and the underlying programming methodologies is also increasing in parallel. Two important aspects are identified: applications and platforms on which these applications run.

Applications: the range of applications continues to grow, e.g. Cyber Physical Systems (CPS), Internet of Things (IoT), connected smart objects, High Performance Computing (HPC), mobile computing, wearable computing. For these applications to fully exploit the benefits of heterogeneous platforms, there is a need to design more flexible software abstractions and improved system architectures due to the diversity of these applications.

Platforms: incorporate multiprocessor system-on-chip (MPSoC), many-core GPUs, heterogeneous CPU+GPU chips, FPGAs, heterogeneous multi-processor clusters, and a range of additional devices into a single solution. These platforms are showing up in a wide range of environments spanning supercomputers, embedded systems, clouds, and personal smartphones.

One approach towards an architecture to support a diverse range of applications running on heterogeneous hardware is the consideration of an hour-glass like model, see Figure 1:

1. A wide top: where many applications (and high-level behaviours) can be mapped;
2. A wide bottom: where many heterogeneous hardware (and the underlying technologies and systems) are situated;
3. A thin centre: where a set of core unified models, abstractions and protocols (possibly based on existing standards) can be mapped.

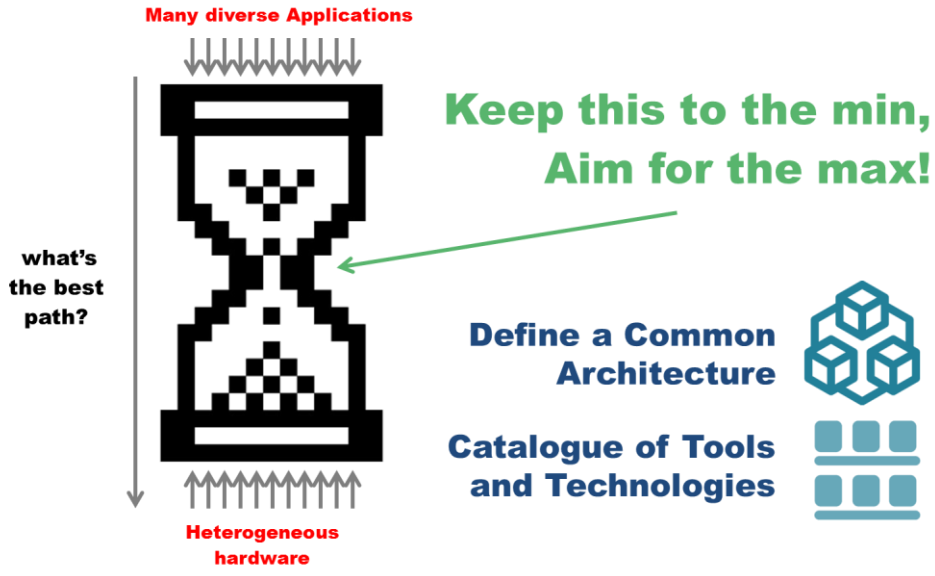

Figure 1 Overall Vision – Hourglass Model

To support such approach, the initial technological areas that the alliance may consider for supporting diverse applications running on heterogeneous hardware are divided into three layers (see Figure 2):

- **Layer 1:** includes technologies for designing and creating applications for HHW like Programming Models, design and modelling tools, code optimization tools

- **Layer 2:** includes all middleware technologies including application life-cycle, monitoring and assessment of HHW, scheduling of tasks, allocation of resources, memory and data management, optimization engines, etc.
- **Layer 3:** includes technologies, emulators, compilers and execution environments that deal with different underlying infrastructures of HHW at two levels: Macro level –Clusters, including distributed network and data management; and Micro level – Heterogeneous Parallel Devices (HPDs), including parallel processing and memory hierarchy management.

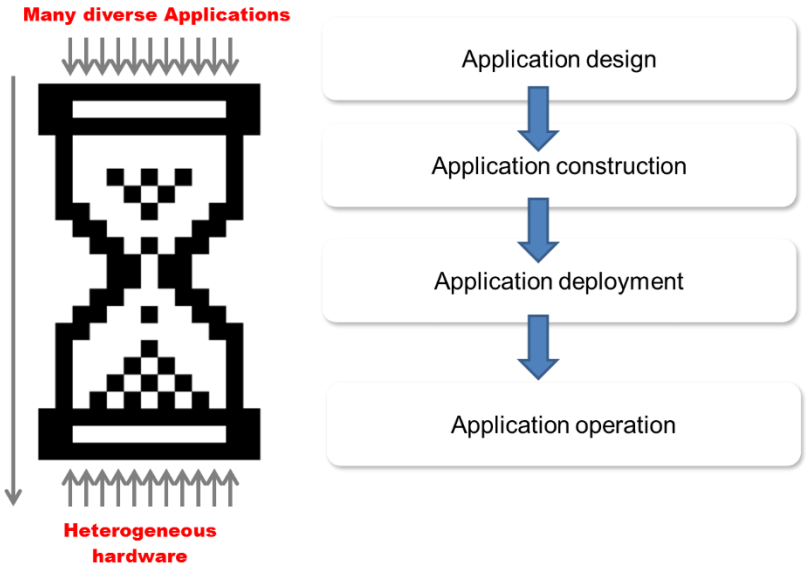

Figure 2 Overall Presentation of the Future Architecture

3. Challenges

In addition to the need to fully understand the impact of different hardware configurations on software systems (both rapidly evolving), a number of challenges lie ahead towards the proposed architecture. It is worth mentioning that the automated techniques implemented in compilers and execution environments are not designed to deal with disruptive changes in the hardware-software interfaces. Moreover, additional factors such as performance, security, time criticality, data movement and power consumption resulting from the software operating on heterogeneous hardware need careful consideration. A number of issues are summarised below.

Reference Architecture: would a three-layer architecture as introduced in section 2 be enough? Traditionally, reference architectures have been proposed in various domains/technologies (e.g. clouds) to support the requirements of families of applications that can take advantage of such architectures. Another example is the IoT for manufacturing (also known Industrial Internet (IIoT)) with its Industrial Internet Reference Architecture (IIRA) which was first published in 2015 by the Industrial Internet Consortium (IIC). The future reference architecture would not only need to provide a wide range of optimisation

criteria, e.g. energy consumption, performance, time criticality but be self-configurable as well in order to adapt and evolve constantly in response to changes in the application environment. Would a holistic approach be the solution to achieve affective and coordinated optimisation?

Programming Models: hardware architectures (such as many-cores, GPUs or FPGAs) require specific programming models. Mainstream programming languages (C/ C++/Java) have no language abstractions to express parallelism, or in other words: parallelism is *not* part of the programming model. Libraries and models for heterogeneous computing are increasingly used in the embedded domain, such as OpenMP, OpenCL and CUDA. Domain Specific Languages (DSL) are a potential approach to reduce the complexity of programming – programming models based on the concepts of the application domain are used to represent the "what", instead of detailing "how" computations are performed. The map-reduce paradigm (Hadoop as an example implementation) is a useful abstraction for parallel data processing, although limited to certain classes of computation. Serverless architectures, also known as Function as a Service (FaaS), are also gaining momentum.

Programming - Tools Support: the complexity of developing applications running on heterogeneous hardware will require extensive tool support. Traditionally, this take place during programming (e.g. parallelism as part of the programming model of the language) as well as in debugging and testing.

Virtualisation: has shown its benefits with its multiple properties and can potentially be supported by the architecture, e.g. multiple Virtual Machines and use of containers technology, in order to provide efficient resource sharing and system security. Network Function Virtualization (NFV) is also showing great potential in the future of network security and stability. Moreover low-level hardware virtualisation technologies, such as I/O virtualisation and Inter-Process Communication virtualisation, have provided energy-efficient approaches. Virtualisation hides much of how the system works in view of global optimization.

Quality of Service (QoS): applications bring their own requirements, and therefore mechanisms are needed to deal with a desired level of QoS. Performance is, besides speed, composed of non-functional aspects such as power and energy consumption, reliability, and time requirements. Fulfilling the QoS requirements may require taking alternate executions paths depending on the favoured optimization criteria (e.g. performance vs energy). These requirements imply that optimization criteria (non-functional properties), must be exploitable holistically across all the architecture layers, including software development.

Low Power/Energy Computing: the biggest challenges to future application performance lie with not only efficient node-level execution on heterogeneous hardware but power consumption as well. CPS and IoT fields are discovering new applications where relatively high performance is required from systems which by the nature of the domain are physically constrained, by size, weight or energy dissipation, or by the reliance on battery or solar energy. Examples include vehicles where space, weight, heat dissipation and reliance on the engine for power are all constraints. Another aspect is data processing. In order to stem the flood of data from applications such as IoT, we must employ intelligent local data processing on remote devices that use minimal energy.

Predictability: for a wide class of CPS and IoT applications, the ability to reason on their behaviour, and provide guarantees on their correct response, surpasses their functional

needs and includes the requirement of temporal correction (safe and secure execution includes the time dimension). Predictability may impact performance (guaranteed performance needs different, less throughput based, approaches), or energy. At the same time, with the increasing shift to complex parallel architectures, it will be difficult, if not impossible, to provide efficient predictability. System design must consider how to build a predictable system with uncertainty at the component level.

Security: security enforcement (and privacy) are of paramount importance. Security however is a complex notion due to the specificity of environment (e.g. IoT, embedded, cloud), the context-specific meaning, the application-specific requirements or stakeholder-specific perceptions. Security threats need to be identified in the context of the applications running on heterogeneous hardware and contingency plans provided in the form of end-to-end security mechanisms applicable for applications running in such environments. Note that reducing the architecture complexity may be required to guarantee security.

Secure Update: software update, presents a conundrum: on one hand timely updates help protect the system against emerging threats (e.g. the WannaCry ransomware attack affected only computers that had not received the latest updates), but, on the other hand, unforeseen interactions between the update and the existing software, may threaten system integrity (as demonstrated by the Lexus software update that crashed the navigation electronic control unit, necessitating an unscheduled service to the repair centre). Security involves the careful consideration of the interactions between systems and subsystems and the effect these may have on the overall security posture of the system. Trust relationships need to be re-evaluated each time a system is reconfigured or modified. In this light, upgrade is an emerging area of concern as it involves both changes in the behaviour and interaction between the upgraded components or subsystems and the rest of the platform components. In addition the complexity of the interaction between entities in the embedded and IoT domains necessitates close re-evaluation of the implications of updating systems.

Platforms: as noted in section 2, applications will require efficient heterogeneous hardware, irrespective of their application domain: embedded, mobile, cloud etc. The hardware should guarantee application requirements such as performance for real-time and safety-critical applications. Heterogeneity is considered at two levels: 1) Macro level: networks of distributed computers (clouds, clusters), composed by diverse node architectures (single, multi-core), are interconnected with potentially heterogeneous networks, and 2) Micro level: deeper memory hierarchies (main, cache, disk storage, tertiary storage) and various accelerator architectures (fixed, programmable, e.g. GPUs, and reconfigurable, e.g. FPGAs).

Other: Technologies such as edge computing involve pushing intelligence and processing capabilities down closer to where the data originates, e.g. from sensors. The intelligence alongside the processing power and communication capabilities of an edge gateway or appliance are directly pushed into devices like Programmable Automation Controllers (PACs). A variety of applications will benefit from such technologies. Would the proposed architecture be extensible enough to support these technologies?

4. Next Steps

This is a draft document. It will be continually edited and updated by the alliance members following feedback. Immediate actions:

- Define a taxonomy of applications and produce the specific requirements of the architecture.
- Setup working groups
- Software repository

5. Contributors

Name	Institution	Email	Project
Karim Djemame	University of Leeds UK	K.Djemame@leeds.ac.uk	TANGO http://tango-project.eu/
Clara Pezuela	Atos Spain	clara.pezuela@atos.net	TANGO http://tango-project.eu/
Oliver Barreto	Atos Spain	oliver.barreto@atos.net	TANGO http://tango-project.eu/
Vassilis Prevelakis	TU Braunschweig Germany	prevelakis@ida.ing.tu-bs.de	SHARCS http://sharcs-project.eu/
Luis Miguel Pinho	ISEP Portugal	lmp@isep.ipp.pt	P-SOCRATES http://www.p-socrates.eu/
Marko Bertogna	UNIMORE Italy	marko.bertogna@unimore.it	HERCULES http://hercules2020.eu/
Sotiris Ioannidis	FORTH Greece	sotiris@ics.forth.gr	RAPID http://www.rapid-project.eu/
Iakovos Mavroidis	FORTH Greece	jacob@ics.forth.gr	RAPID http://www.rapid-project.eu/
Yannis Papaefstathiou	Synelixis Greece	ygp@sylexis.com	ECOSCALE http://www.ecoscale.eu/