M2DC-Modular Microserver DataCentre with heterogeneous hardware

Ariel Oleksiak, Michal Kierzynka, Wojciech Piatek, Giovanni Agosta, Alessandro Barenghi, Carlo Brandolese, William Fornaciari, Gerardo Pelosi, Mariano Cecowski, Robert Plestenjak, et al.

M2DC – Modular Microserver DataCentre with Heterogeneous Hardware

Ariel Oleksiak[a,*], Michal Kierzynka[a], Wojciech Piatek[a], Giovanni Agosta[b,*], Alessandro Barenghi[b], Carlo Brandolese[b], William Fornaciari[b], Gerardo Pelosi[b], Mariano Cecowski[c], Robert Plestenjak[c], Justin Činkelj[c], Mario Porrmann[d,*], Jens Hagemeyer[d], René Griessl[d], Jan Lachmair[d], Meysam Peykanu[d], Lennart Tigges[d], Micha vor dem Berge[e], Wolfgang Christmann[e], Stefan Krupop[e], Alexandre Carbon[f], Loïc Cudennec[f], Thierry Goubier[f], Jean-Marc Philippe[f], Sven Rosinger[g], Daniel Schlitt[g], Christian Pieper[g], Chris Adeniyi-Jones[h], Javier Setoain[h], Luca Ceva[i], Udo Janssen[j]

[a] *Poznan Supercomputing and Networking Center, ul. Noskowskiego 10, 61-704 Poznan, Poland*
[b] *DEIB – Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano, Italy*
[c] *XLAB d.o.o., Pot za Brdom 100, 1000 Ljubljana, Slovenia*
[d] *Cognitronics and Sensor Systems Group, CITEC, Bielefeld University, Germany*
[e] *christmann informationstechnik + medien GmbH & Co. KG, Ilseder Huette 10c, 31241 Ilsede, Germany*
[f] *CEA, LIST, PC 172, 91191 Gif-sur-Yvette CEDEX*
[g] *OFFIS e. V. – Institute for Information Technology*
[h] *ARM Ltd., CPC-1 Capital Park, Fulbourn, Cambridge CB21 5XE, UK*
[i] *Vodafone Telematics*
[j] *CEWE Stiftung & Co. KGaA*

Abstract

The Modular Microserver DataCentre (M2DC) project investigates, develops and demonstrates a modular, highly-efficient, cost-optimized server architecture composed of heterogeneous microserver computing resources. The resulting server architecture will be able to be tailored to meet requirements from a wide range of application domains. M2DC is built on three main pillars: a *flexible server architecture* that can be easily customised, maintained and updated; *advanced management strategies* and *system efficiency enhancements* (SEE); well-defined interfaces to the surrounding software data centre ecosystem. In this paper, we focus in particular on the thermal management strategies and on the initial benchmarking of the *Aarch64* ARM architecture.

Keywords: Microservers, Data Centres, Heterogeneous Architectures

*Corresponding author
Email addresses: ariel@man.poznan.pl (Ariel Oleksiak), agosta@acm.org (Giovanni Agosta), mporrmann@cit-ec.uni-bielefeld.de (Mario Porrmann)
URL: http://m2dc.eu (Mariano Cecowski)

1. Introduction

During the last decade, the fast development of compute-demanding applications such as Internet of Things, data analytics, media processing, and cloud platforms caused a fast growth of data centres, as illustrated by the latest Cisco Cloud Index report, which indicates that global data center IP traffic is expected to nearly triple (2.8-fold) over the next 5 years. Overall, data centre IP traffic will grow at a compound annual growth rate (CAGR) of 23 percent from 2013 to 2018 [1]. This fast growth called for large investments and increased power usage. To cope with these issues without slowing down the innovation based on the adoption of pervasive computing technologies, dramatic decreases in costs and power requirements are needed. These decreases must be, however, accompanied by assured Quality of Service (QoS), high levels of reliability and security, and by ease of configuration, integration, and application execution, even if system improvements are achieved with the use of cutting-edge and specialized technologies [2]. At the same time, new technologies and embedded computing architectures create numerous new opportunities. New architectures with high computing power to power consumption ratios are becoming widely available, such as mobile processors (e.g., ARM-based multi/many-cores), embedded SoCs (System-on-Chip) including GPU or FPGA-based accelerators, etc. Going beyond the separation between embedded and desktop/server markets, these architectures draw a continuum of computing resources, ranging from small, power-optimized microcontrollers to large, powerful many-core server chips, enabling designers to tailor a system to the exact needs of applications and workload with appropriate components. These challenges and opportunities are at the heart of the Modular Microserver DataCentre (M2DC) project [3]. M2DC will capitalize on the European strength in embedded system design and it will leverage the opportunities offered by cutting-edge computing resources and technologies so as to build specialized energy-efficient appliances aiming at meeting the needs of future high-value applications, based on intensive media processing, IoT or even HPC.

To address these emerging challenges, M2DC will investigate, develop and demonstrate as a prototype in an operational environment a modular, highly-efficient, cost-optimized server architecture composed of heterogeneous microserver computing resources, being able to be tailored to meet requirements from different application domains such as image processing, IoT, cloud computing and HPC. M2DC will develop turnkey appliances based on a microserver system enabling to build use case driven, modular, high-density efficient data centres. The idea is to provide use cases in the form of turnkey appliances that can be easily configured, produced, installed and maintained. Thus, the main M2DC goal is to deliver a new class of appliances with the following properties:

- P1 Low cost – taking into account the whole appliance life cycle (purchase, operation, maintenance and refresh cycles) and the Total Cost of Ownership (TCO) optimisation;
- P2 Low power and high energy efficiency – dramatically reducing power usage and heat dissipation while meeting Quality of Service (QoS) for key and emerging applications;

P3 Dependable by design – delivering built-in reliability and security by integrating fast and efficient monitoring and management functions,

P4 Versatile and scalable – easy to customize and update (software and hardware) to specific application types and large scales by seamless inclusion of heterogeneous and highly parallel computing resources,

P5 Easy to use and integrate with data center ecosystems – easy provisioning, monitoring and management by modern DCIM (Data Centre Infrastructure Management), cloud and HPC software;

P6 Applicable to a variety of real-life applications – facilitating application and middleware programming, deployment, and optimisation in order to use M2DC appliances for various important real-life applications such as Image Processing or Internet of Things data analytics.

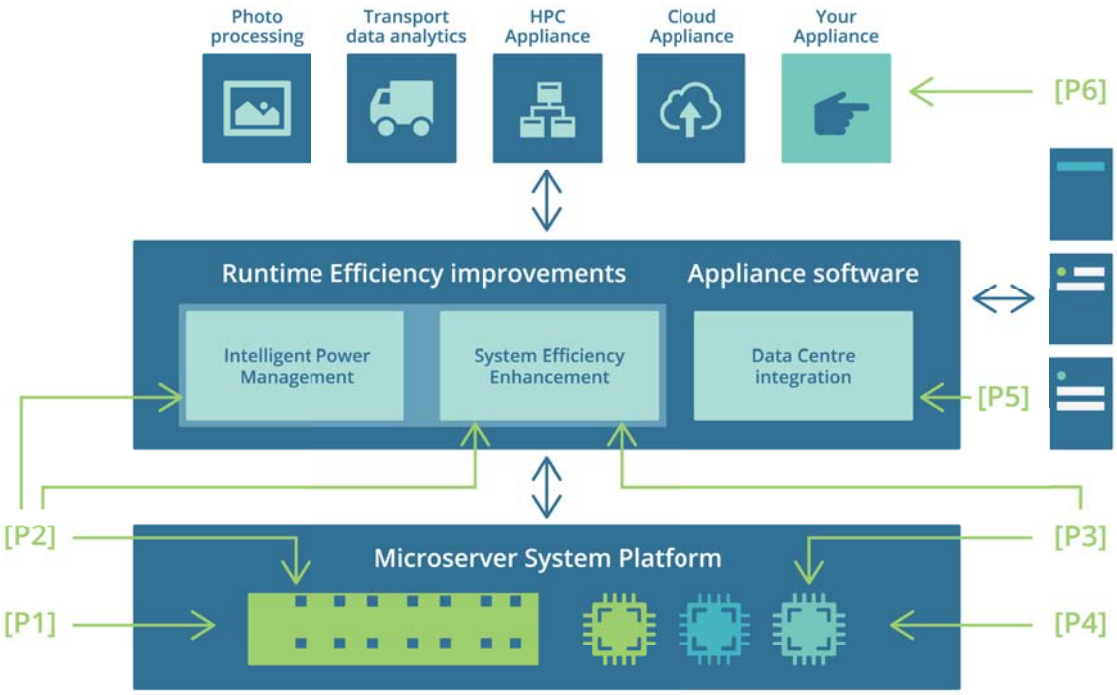

Figure 1: High-level view of the M2DC appliance with its properties and interfaces between main parts.

To develop its appliances, M2DC proposes a flexible server architecture that can be easily customized, equipped with intelligent power management and integrated with well-defined interfaces to the surrounding software ecosystem. The server architecture is based on low power System on Chip (SoC) components accompanied by built-in enhancements (e.g. for performance acceleration, efficiency, dependability) at system level, thus delivering great efficiency while minimizing the effort needed from users. M2DC appliances will enable TCO optimization for specific use cases and application areas. The overall costs will be lowered by using low cost microserver modules, decreasing of energy consumption costs, and facilitating maintenance and integration with existing computing environments. The connections between this approach and

M2DC appliance properties are illustrated in Figure 1.

Organization of the paper The rest of this paper is organized as follows. In Section 2, we provide an overview of the M2DC modular microserver hardware and software architecture, while in Section 3 we provide a more in-depth look at thermal management. In Section 4, we briefly describe the range of application scenarios on which the M2DC technology will be demonstrated, while in Section 5 we offer initial benchmarking results for the ARM *Aarch64* architecture which will serve as the general purpose processor unit. Finally, in Section 6 we review some related works in previous European projects and in Section 7 we draw our conclusions.

2. Architecture Overview

The M2DC project targets the development of a resource-efficient, highly scalable, modular microserver system that can be easily configured to fit the workload requirements of a wide variety of applications. Low-power microserver modules could be easily combined with reconfigurable and massively parallel hardware accelerators using a high-speed low-latency communication infrastructure to provide the heterogeneous mix of cutting edge technologies required by customers and applications. Efficiency in terms of performance, energy, and TCO will be demonstrated by a representative mix of turn-key appliances that are supported by an intelligent, self-optimizing management infrastructure.

M2DC provides a versatile solution that can be easily configured to provide unparalleled density of server nodes in new data centre environments, offering the required cooling and power facilities, or to provide low-power solutions that can be easily integrated into existing data centre environments. The M2DC modular microserver system architecture is not just a research platform. On the contrary, it targets reliability and maintainability levels of a commercial product, utilizing a blade-style system approach, providing hot-swap and hot-plug capabilities. Due to its modular and scalable architecture, the system can combine arbitrary mixtures of high-performance ARM server processors, low-power ARM embedded/mobile SoCs, traditional x86 processors, GPUs and FPGAs in a heterogeneous server environment, even on chassis level. In M2DC we will concentrate on the integration of cutting-edge technologies, including the latest 64-bit ARM based server processors provided by Huawei, as well as ARM embedded/mobile SoCs, and the latest FPGA technologies.

The microserver architecture will include a dedicated communication infrastructure for monitoring and control, providing fast and easy access to the over 10,000 sensor values available in a single rack[1]. Additionally, the huge amount of sensor values gathered via the monitoring network can be efficiently pre-processed on the integrated distributed microcontroller network. The rich sensorization in combination with the capabilities for distributed data pre-processing and the potential for data mining allows for intelligent and effective power and energy management solutions. New deployment and management technologies will be combined with a proactive power management

[1] A single rack comprises up to 3360 microservers, see Section 2.1 with more than ten individual sensors, e.g., for temperature, voltage and current of the different supply rails

for providing QoS-aware dynamic performance settings, exploiting the heterogeneity of the server platform. By observing the large amount of thermal sensors in combination with machine learning approaches, M2DC will provide new methods for thermal management of heterogeneous high-density architectures that continuously adapt and optimize their behaviour at micro-server, chassis, and data centre level. By doing this, the system can be adapted at run-time to meet different requirements, e.g., for optimizing performance and minimizing hot spots.

Although the integration of energy-efficient multi-core processors will significantly increase the energy efficiency compared to today's server platforms, new approaches are required in order to reduce energy consumption by more than one order of magnitude. Massively parallel architectures have shown to provide the required performance/power ratio but typically they are integrated as hardware accelerators for a dedicated application. Within M2DC we will use cutting-edge dynamically reconfigurable FPGAs as well as GPUs for System Efficiency Enhancements (SEE) to further increase efficiency, dependability, and scalability.

In addition to providing a server platform and a variety of low-level and application-level benchmarks that prove the efficiency of the system, M2DC will deliver turnkey appliances for selected applications. The targeted appliances will be built upon different middleware layers and techniques, mostly running direct on the operating system and thus the applications directly on the hardware. The targeted "bare metal cloud" approach dynamically installs a desired operating system as well as all needed libraries and applications onto physical nodes. The advantage is that 100% of the available resources can be directly used without virtualization overheads. Nevertheless, a virtualized and containerized cloud-like approach is more flexible, so it will also be offered where appropriate. Every approach will be the basis for a pre-configured appliance, ready to be directly used through graphical dashboards. The Infrastructure-as-a-Service (IaaS) and MaaS layer itself can also be seen as a basic appliance as some users might still want to install individual applications on their own.

To ease the integration of the appliances into existing data centres with legacy management software, all relevant interfaces like the M2DC blade server, the operating system management and the appliance management, will be based on widely used standards. This will include required interfaces for smooth integration with DCIM and HPC management software allowing fine-grained monitoring and comprehensive set of power management functions. The targeted applications span from cloud computing via image processing and big data analytics to HPC applications, representing a wide variety of different requirements. The flexibility of the microserver architecture enables heterogeneous servers that are specifically tailored to each application's needs including dedicated SEEs that increase performance and energy efficiency beyond simple homogeneous platforms.

M2DC will be built on three main pillars, as shown in Figure 2. The results of these three pillars will be combined to produce Total Cost of Ownership (TCO)-optimized appliances, deployed in a real data centre environment and seamlessly interacting with existing infrastructure to run real-life applications.

2.1. M2DC Server Architecture

The M2DC next generation modular microserver integrates a wide spectrum of heterogeneous microserver technology, making it a good platform for a wide range of applications. Specifically, state-of-the-art x86 processors, 64-bit ARM mobile/embedded SoCs, 64-bit ARM server processors, FPGAs, GPUs and potentially other acceleration units could be integrated. In contrast to existing microserver platforms that support only homogeneous populations, the M2DC next generation modular microserver en-

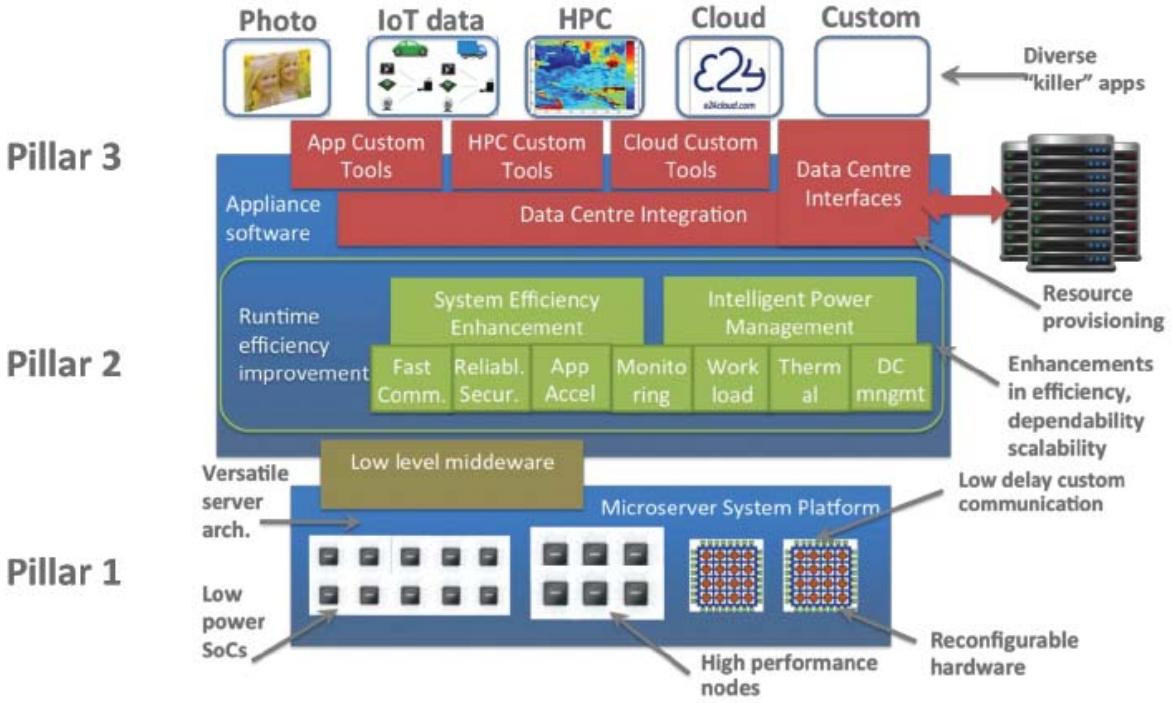

Figure 2: The pillars of the M2DC appliance concept: server architecture, system improvements, and middleware stack, all integrated into a data centre.

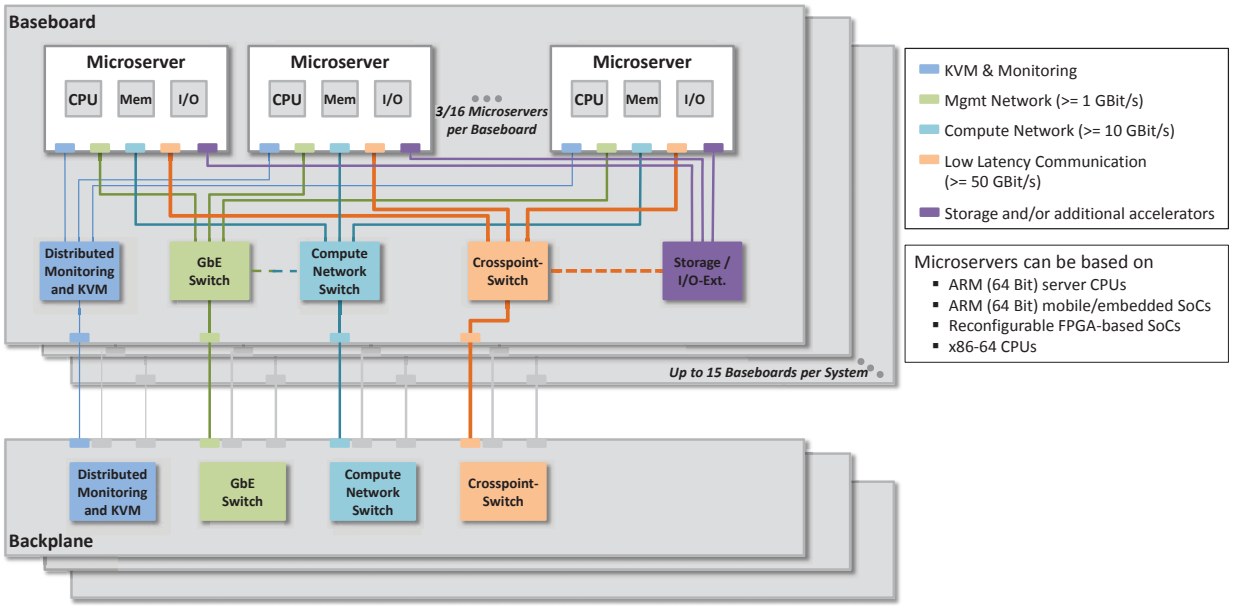

Figure 3: Overview of the M2DC server architecture.

ables a seamless combination of all these technologies in a single enclosure. This allows the fine-tuning of the platform with respect to specific applications, offering a densely coupled, highly integrated heterogeneous microserver including a scalable, high-speed, low-latency communication infrastructure. All major processing architectures (CPU, GPU and FPGA) are available in a high performance as well as a low power variant. This enables heterogeneous architectures to transfer the big-little-approach from SoC-level to system-level, combining, e.g., 64-bit ARM server processors and 64-bit ARM mobile SoCs (integrating GPUs and fixed-function accelerators). Combined with the heterogeneity of the M2DC server, the concept allows choosing e.g. a server-grade CPU combined with a small FPGA, or vice-versa.

Within M2DC, the development of a microserver platform that combines resource efficiency with high reliability and maintainability is targeted in order to being able to compete on the server market. Therefore, we target form factors on microserver-, baseboard-, backplane-, and chassis-level that match the requirements of today's data centres, enabling hot-swapping and hot-plugging of system components as well as their smooth integration into existing data centre racks. Using existing computers on module form factors which are established in the industry allows reuse of already existing microserver developments. Fine-grained power monitoring and control within the system at the hardware level enables sophisticated high-level management of power, performance, and temperature. The power supplies of the M2DC system are foreseen to be shareable on rack level. Exploiting this feature enables improved energy efficiency as well as higher system reliability due to added redundancy.

Figure 3 shows an overview of the general M2DC server architecture. The distributed monitoring, control and maintenance infrastructure is accessible via web interface and/or RESTful API - allowing full integration into DCIM and orchestration frameworks. Apart from this infrastructure, each microserver is connected via an Ethernet-based management and compute network. These networks provide the basic communication backbone for the different microservers, offering multiple 1 GBit/s and 10 GBit/s Ethernet links to every microserver. In addition, a dedicated high-speed low-latency communication network is integrated into the M2DC next-generation modular microserver architecture, which is described in more detail in section 2.2.

Standardized computer on module form factors and interfaces are used for the microservers. This eases integration of third party microserver modules, providing a large set of commercial off-the-shelf microserver modules readily available for usage in the M2DC server. It also offers additional use cases in the embedded market, reusing the developed microserver modules. In the following, we give a short overview of the various microservers that are already available or that are currently developed for integration into the M2DC server. Based on the performance and form factors of the modules, we distinguish between high performance and low power microservers.

High Performance Microservers

The COM Express form factor [4] is used as the basis for all high performance microservers, supporting a compact form factor of just 125 mm x 95 mm. COM Express Type 6 and Type 7 modules are supported, enabling direct integration of commercial off-the-self modules, e.g., x86 modules based on Intel's Skylake/Kabylake architecture. Targeting highly resource-efficient platforms for next-generation data centres,

two new high performance microservers are developed within M2DC, utilizing FPGAs and ARM-based server processors, respectively.

The high performance ARMv8 microserver is based on an ARM 64-bit SoC, integrating 32 Cortex-A72 cores running at up to 2.1 GHz. The memory controller supports four memory channels populated with DDR4 SO-DIMMs running at 1866 MHz. In total, each microserver integrates up to 128 GByte RAM. For connectivity, the microserver provides two 10G GbE ports with RoCE support, in addition to a 1 GbE port which is mainly used for management purposes. Up to 24 high-speed serial lanes are available for peripherals connection or for high-speed, low-latency communication to other microservers in the M2DC server. Using these high-speed links, also multi-socket configurations of the ARMv8 microserver are supported. Additionally, a wide variety of fixed-function units are integrated into the SoC, providing highly resource-efficient acceleration of compression/decompression or security algorithms like asymmetric encryption.

FPGAs are becoming more and more attractive in HPC and cloud computing due to their potentially very high performance combined with moderate power requirements. High-level synthesis and OpenCL support, which is becoming more and more mature, opens additional application scenarios since programming is no longer limited to hardware specialists. The FPGA-based high performance microserver will be a full featured COM Express module, comprising an Altera Stratix 10 SoC with an integrated 64 bit quad-core ARM Cortex-A53 processor. Dedicated DDR4 memory is provided for the CPU as well as for the FPGA fabric, supporting up to four memory channels and up to 64 GByte. The high-speed transceivers integrated in the FPGAs are used for PCIe interfacing to communicate with other processor modules, and for low-latency high-bandwidth communication between High Performance FPGA-based Microserver modules.

Low Power Microservers

SoCs targeting the mobile market are promising platforms for datacenters when focusing on energy efficiency, given the large amount of integrated accelerators, including GPGPUs, fixed function units, e.g., for video transcoding, or even FPGAs. The M2DC server allows to integrate modules based on the Jetson standard from NVIDIA, which is used for the currently available Tegra SoCs (Tegra-X1) and the upcoming generations from NVIDIA. Additionally, the Apalis standard from Toradex [5] is supported. With its small form factor of just 82 x 45 mm, it allows a very high density of microservers. In addition to commercially available Apalis modules from Toradex, modules have been developed at Bielefeld University integrating Samsung Exynos5250 SoCs and Xilinx Zynq7020, respectively [6] – a single rack comprises up to 3360 microservers, with more than ten individual sensors, e.g., for temperature, voltage and current of the different supply rails.

Modularity and Scalability

A block level overview of the M2DC server architecture is provided in Figure 3. The basic concept follows a high-dense, yet modular approach. The different microservers, both the high performance and the low power ones, are plugged in blade-style

Table 1: Overview of the different chassis variants of the M2DC server, their respective integration densities and power requirements.

		Small Server	Mid-range Server	Scale-out Server
	Server Height	2 RU	3 RU	3 RU
per chassis	#LP Server	48	144	240
	#HP Server	9	27	45
per rack (42 RU)	#LP Server	672	2016	3360
	#HP Server	126	378	630
1 RU (eff.)	#LP Server	24	48	80
	#HP Server	4.5	9	15
Power	1 Chassis	1.2 kW	2.9 kW	4.9 kW
	1 RU (eff.)	0.6 kW	1 kW	1.6 kW
	1 Rack	25 kW	40 kW	68 kW

baseboards or microserver carrier, which can be slided into the M2DC server in hot-swap and hot-plug fashion. There are baseboards for every kind of microserver, e.g. a baseboard for COM Express based high performance microservers as well as Jetson based low power microservers. A high performance baseboard integrates three COM Express microservers into the M2DC server chassis; a low power baseboard supports 16 Jetson/Apalis microservers. Apart from just powering the modules, the baseboard provides the entire communication and management infrastructure required by the microservers, e.g. the 10G GbE and PCIe switching infrastructure, the KVM support, or the monitoring environment. The different baseboards of a M2DC server are then connected to a modular communication backplane, interconnecting the different baseboards, as well as providing the required external interfaces.

In order to support different data centre environments and use cases, three chassis variants of the M2DC server are supported. Table 1 gives an overview of these three different server chassis. The scale-out server generates very high integration density, targeting new hyperscale data centres which provide the required power and cooling capacities. Using the scale-out M2DC server, up to 240 low power or 45 high performance microservers can be integrated into a single 3 RU chassis. In order to support also existing data centres with limited power and cooling capacities, the mid-range chassis has been conceived. The small server chassis supports evaluation setups and special use cases, e.g., test beds or deployment in a non-data-centre environment.

2.2. High Speed Communication

The M2DC high-speed communication infrastructure is based on a dedicated high-speed low-latency communication network, which is integrated into the M2DC next-generation modular microserver architecture. It connects to the CPU-/GPU-based microservers via PCIe and to the FPGA-based microservers via their high-speed serial interfaces. Depending on the involved communication partners, it features either Host-2-Host PCIe-based packet switching or direct, circuit-based switching. In addition to direct communication between the different microservers it also supports connection to

storage or I/O-extensions. This allows for an easy integration of PCIe-based extension cards like GPGPUs or storage subsystems.

In contrast to the majority of today's implementations, where a hardware accelerator is typically physically attached to the PCIe lanes of a CPU node, our communication infrastructure can be used not only to connect CPUs to hardware accelerators, but also CPUs to CPUs or accelerators to other accelerators. Furthermore, it is possible to divide the SEE link of a certain CPU/accelerator, e.g., connecting an accelerator to both a CPU and another accelerator. Thus, accelerators can be combined to one large virtual SEE unit. At run-time, the communication topology can be reconfigured and adapted to changing application requirements. The high-speed low-latency communication infrastructure provides the basis for most of the SEEs discussed in the next section.

2.3. Advanced management strategies and system efficiency enhancements

To improve the behaviour of the system during runtime and to meet requirements from the various applications, the server architecture includes built-in enhancements at system level, such as computing acceleration, enhancements of the global efficiency thanks to data management, dependability and security, behaviour monitoring, etc. In comparison to current FPGA or GPGPU-based hardware accelerators that target specifically the applications' performance enhancement, the M2DC hardware accelerators are seamlessly integrated into the system architecture both at hardware and at software level. The accelerators are capable of providing a wide variety of mechanisms for global system efficiency enhancements ranging from application-independent system-level functions via enhancements that support a complete class of applications, to dedicated accelerators for a specific target application. For example, SEEs are envisioned for remote DMA, synchronisation, or cluster-wide available low-latency global scratchpad memory (SPM) and associated data management and transformation. Depending on the actual requirements, the accelerators can dynamically adapt their behaviour, e.g., towards performance improvements, power reduction, and dependability.

2.3.1. Neural Network

Artificial neural networks are gaining increasing attention in research as well as in commercial applications for machine learning and data mining. Hence, as a typical example for application acceleration, we target the implementation of a neural network model for unsupervised clustering of datasets, called Self Organizing Feature Map (SOM) [7]. SOMs are widely used, e.g., in bioinformatics [8] or hyperspectral image analysis [9], and well suited for hardware implementation. While the algorithm is computationally intensive, the degree of parallelism is very high, which allows for efficient computation on parallel hardware [10]. The scalability of the algorithm (in terms of the number of processing elements on which to partition the network model) makes it well suited to demonstrate the power and flexibility of the M2DC architecture. The basic SOM algorithm consists of three main phases:

Initialization of the weight vectors $\mathbf{w}_i$ of all neurons n_i, which is typically done by assigning random values to all components j of vector $\mathbf{w}_i$.

Best-Match detection: At every discrete time step t, a randomly selected input (feature) vector $\mathbf{x}(t)$ is presented and the distance between $\mathbf{x}(t)$ and all $\mathbf{w}_i(t)$ is calcu-

lated. The neuron n_c with the smallest distance to $\mathbf{x}(t)$ is selected as best-match. Typical metrics for vector spaces are Euclidean and Manhattan distance.

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha(t) \cdot h_{c,i}(t, \|\mathbf{r}_c - \mathbf{r}_i\|_p)[\mathbf{x}(t) - \mathbf{w}_i(t)] \quad (1)$$

Here $\alpha(t)$ is a scalar of range [1...0], decreasing over time. $h_{c,i}$ is the neighbourhood function, whose width decreases in space and time. The first results of the comparison between the performance and energy efficiency of FPGA and CPU implementations are presented in Section 5.

Regarding supervised approaches, another promising neural network-based model used for Cognitive Computing is the Deep Learning paradigm. For example, deep learning using Convolutional Neural Networks (CNN) is based on several layers of filters, pooling functions and classifiers, each composed of a high number of operations to execute. The learning phase is done using a learning database, composed of well-known and labelled data, enabling a learning algorithm to modify the different data of the network. From the software point of view, four main frameworks are used: TensorFlow, Torch, Caffe, and Theano. From the hardware point of view, the most used architecture for implementing both learning and feedforward phases is currently the GPU, since neural networks are regular and exhibit high parallelism. For example, Nvidia proposes optimized GPU hardware [2] and software (CuDNN). Regarding FPGAs, the acquisition of Altera by Intel is meant to provide new computing devices ever more efficient for this usage by providing large-scale FPGAs tightly coupled with Xeon server processors. From the manycore era, there are also works by Kalray for example. Several dedicated ASICs were also investigated for this purpose such as the recent solutions from the academic landscape, e.g. Neuflow from New-York University or DianNao from ICT (China) and Inria (France). From the industry point of view, one can cite the TensorFlow Processing Unit (TPU) from Google. All these architectures could be candidates to be integrated into a M2DC Deep Learning appliance. However, leveraging existing computing modules is seen to be more interesting to demonstrate the versatility of the M2DC concept. For this purpose, as a benchmark, a deep learning accelerator was proposed as a SEE to be implemented on the FPGA hardware for comparison with competitive architectures. The so-called PNeuro is a programmable clustered SIMD architecture designed to accelerate Deep Neural Network (DNN) applications. PNeuro is able to accelerate a wide range of data processings, from typical image processings to complex neural network computing chains such as CNN with possibly multiple classifier networks (Radial Basis Functions-RBF with Gaussian activation function or Multilayer Perceptron-MLP with sigmoid, tanh or rectified linear functions). This accelerator, presented in Figure 4, is composed of an instruction set of around 50 32-bit wide instructions, including both control and computing parts. It is viewed as a slave by the system, which is meant to manage the data to be processed and the results.

[2]NVidia Pascal architecture, `http://www.nvidia.com/object/deep-learning-system.htm`

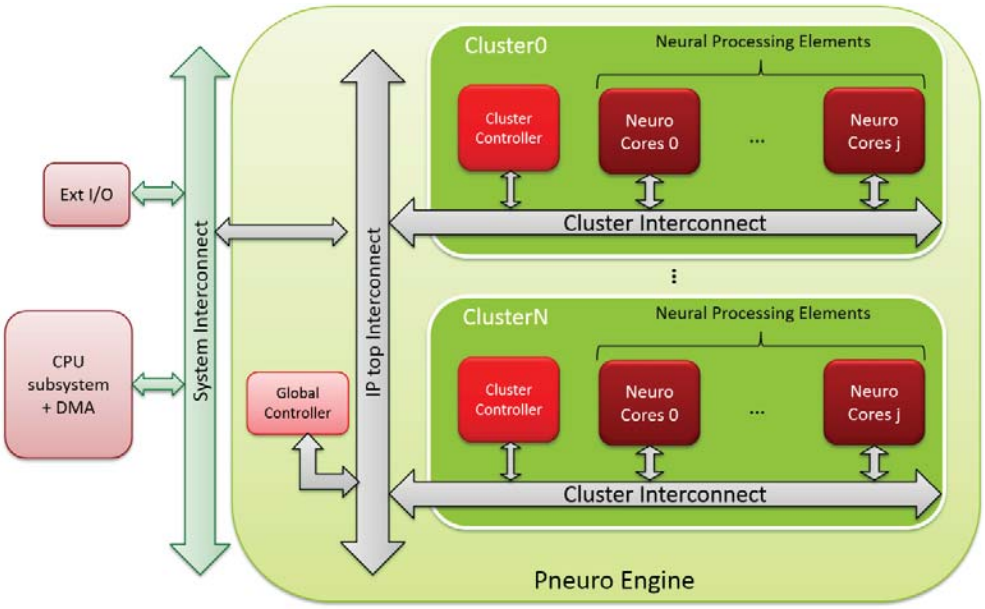

Figure 4: PNeuro architecture overview and host system

2.3.2. Security

Regarding security enhancements, in order to provide confidentiality for the transfer of input data and retrieval of the computed results, we will develop efficient OpenCL implementations of cryptographic primitives [11–13], which can be transparently deployed either on GPUs or on FPGA targets thanks to the availability of automated tools. The reasons for providing dynamically deployable, yet highly optimized cryptographic primitives are twofold. A first motivation is that the choice of the cryptographic primitives employed to provide confidentiality and integrity on bulk data, i.e., a symmetric block cipher and a cryptographic hash, is driven by both the high efficiency and security requirements, and the compliance with nation-wide regulation. Thus, although both x86-64 and ARMv8 ISAs include dedicated instructions to compute a symmetric block cipher (namely, AES) and ARmV8 also allows the integrator to include cryptographic hashes (namely, SHA-1, SHA-2-224 and SHA-2-256), providing a greater cipher suite flexibility, at a reduced efficiency impact is a definite advantage. To this end, we will provide the support for the ISO/IEC 18033-3:2010 standard block ciphers, which include Triple DEA, Misty1, CAST-128, HIGHT, all having a 64-bit input block, and Camellia and SEED which have a 128-bit input block, besides the AES block cipher. On the cryptographic hash side, we will implement the full SHA-2 cryptographic hash functions family, including the versions with 384- and 512-bit digests, which are not available as ARMv8 ISA extensions. This fulfils the large majority of future requirements for both security and efficiency. Providing legacy support for SHA-1 enhances the compatibility of the designed solution, although the algorithm is deemed to be phased out from use in the Transport Layer Security protocol (IETF Standard RFC 5246, [14]) by all the browser manufacturers due to the increasing amount of security concerns. Finally, providing support for the novel SHA-3 standard hash function (U.S. NIST FIPS-202, [15]) will further enhance the future usability of the cryptographic acceleration due to its foreseen widespread use, especially thanks to its efficient hardware implementation.

The second reason to provide dedicated accelerators to block ciphers is to free valuable CPU time, which may be better employed in loads unsuitable for accelerators. A typical case is the one of control intensive loads, which fit better to a general purpose CPU, and have significant efficiency penalties when executed on an accelerator. In these cases, regardless of the better performances achieved as a side-effect, the use of a dedicated cryptographic accelerator allows to achieve a better efficiency both thanks to a reduced power consumption in the cryptographic primitive computation, and thanks to an allocation of the computational load to better suited units.

2.4. Middleware stack

Building on the Linux operating system (e.g., Linaro for ARM-based compute modules) and other well-known software infrastructures, M2DC will also feature optimized runtime software implementations when needed to improve the efficiency of the system towards application domains such as cloud computing, big data analytics and HPC applications. At the heart of the middleware for "bare metal cloud" sits OpenStack Ironic, which provides bare metal (micro)server software deployment and lifecycle management. OpenStack Ironic will be modified and complemented by other OpenStack components for handling the dynamic and heterogeneous nature of the M2DC microserver nodes, in particular for the hardware accelerators.

3. Thermal Management

Due to the heterogeneity, power density, and thus, possible thermal imbalance of the M2DC microserver, proper energy- and thermal-aware management becomes crucial not only in achieving significant energy savings, but in assuring the high reliability of the system. However, these management policies need to consider various challenges and conditions including workload fluctuations, power leakage, managing hot spots, and finding a trade-off between workload and resource management. To deal with them resource and thermal management will take into account sensor readings and information coming from the server monitoring tools. These data will be investigated and applied to corresponding power and thermal models at the same time to determine trends in their changes. These models are supported with the benchmarking data and statistical methods increasing their accuracy. The overriding goal of thermal management is to avoid exceeding the maximum temperature allowed of the components junction. Data sheet parameters accompanied with thermal models support the estimation of the device's junction temperature and predict its changes. A general thermal model adopted within M2DC is the simple two resistors (2R) model including a junction point. Figure 5 gives a representation: when using a heat sink, the case (TOP) to ambient resistance can be decomposed as the sum of the case to heat sink resistance and the heat sink to ambient resistance, where:

- R_{cs} represents the case to heat sink thermal resistance
- R_{sa} is the heatsink to ambient thermal performance of the heatsink (depends upon the airflow)

- R_{ba}: Resistance from board to ambient (depends upon the airflow)
- R_{jc}: Resistance from junction to the top of the package (case)
- R_{jb}: Resistance from junction to the board

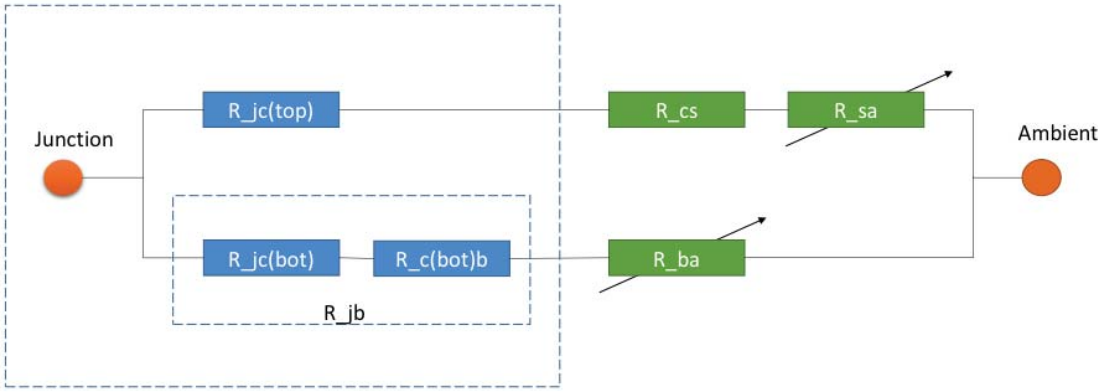

Figure 5: Thermal path for a single component.

The junction temperature is obtained from the previous parameters using the following formula:

$$T_j = T_a + \frac{(R_{jb} + R_{ba}) * (R_{jc} + R_{cs} + R_{sa})}{(R_{jb} + R_{ba} + R_{jc} + R_{cs} + R_{sa})} * Power \tag{2}$$

This generic model is a starting point to provide individual estimations for particular components of M2DC box. For instance, to reflect the thermal behaviour of a processor we use the simplified version of the above model, benefiting from Newton's law of cooling, that is presented in [16].

Based on the models (predicting future trends) and on the analysis of collected data, the resource and thermal management module will perform energy optimization actions. The proposed solution will provide three functional modules, namely: Energy Saver Manager, Fan Manager and Power Capping Manager. The Energy Saver Manager is responsible for taking actions aimed at reduction of power consumed by M2DC box components. These include: management of power states of particular system components; dynamic addition/removal of cores; and exploitation of dynamic voltage and frequency scaling (DVFS) technology (if possible) to optimize the speed of processing units while meeting thermal constraints. It will also be responsible for keeping some of the unused resources "ready" for the upcoming load - thus it will perform selective and delayed switching. The Fan Manager is in charge of adjusting the fans speed in order to keep all components within the desired temperature range and optimize their power usage. Fan management will benefit from the possibility of managing each fan separately in a fine-grained manner. Taking into account the prediction of particular component temperatures, it will continuously adjust the speed of particular fans in a proactive way trying to maintain a particular set-point temperature and preventing the components from exceeding predefined thresholds. Finally, Power Capping Manager allows users to provide the limitation for the maximum power drawn by

the system. It will take advantage of both the Energy Saver Manager and the Fan Manager, considering management of power states, DVFS as well as fans management to reduce the power. As the actions taken by the Power Capping Manager are performed with respect to users' external requirements, they have the highest priority.

All these components take their control actions through dedicated monitoring system and OpenStack services responsible for reading sensor values and providing all the information to the management modules.

4. Use Case Scenarios

One of the key goals of the M2DC project is to deliver appliances well suited for specific classes of popular or emerging relevant applications. Thus, a crucial part of the strategy of the project is to steer the joint development of software and hardware by specific real-life use cases identified by the project partners. Such a continuous validation will enable M2DC to deliver optimized appliances customized to relevant and real-life workloads and use cases. The M2DC use cases have been carefully selected to ensure their potential wide uptake, market relevance, and special importance for future computing and data processing needs.

In this section we present two application subsets, numerical HPC applications and Neural Networks, which have been employed in the initial experimental campaign, as well as an overview of the remaining target applications.

4.1. HPC applications: EULAG

High performance computing (HPC) software tools are among the most time, energy and resource consuming type of applications. Therefore, it is crucial to address also this scenario, especially as M2DC is all about reducing energy consumption and total cost of ownership. A good representative in this area is EULAG – a numerical HPC solver offering large spectrum of application fields, such as orographic flows, urban flows, simulation of contamination dispersion, investigation of gravity waves and many others [17]. EULAG is currently used at Poznan Supercomputing and Networking Center for different scenarios, e.g. air quality monitoring and precise weather prediction. As mentioned above, EULAG is a good example of HPC application as it implements stencil-based computations, requires a low-latency network and can be efficiently parallelized on modern computational architectures [18, 19]. Therefore, it will help evaluate the computational and networking aspects of the M2DC platform. It is expected that the M2DC project will improve the performance per Watt for such complex distributed applications due to optimized reconfigurable communication and the use of hardware acceleration. The performance, energy efficiency and hardware costs will be studied in order to deeply analyse and quantify the improvements for this particular HPC scenario.

4.2. Simulation of neural networks

As a benchmarking application with potentially high parallelism, neural network models are foreseen to be ported and evaluated on the M2DC platform. Likewise HPC EULAG, these models can be used to study the scalability of the architecture, and to

evaluate the impact of the integrated low-latency communication. Both convolutional neural networks to accelerate cognitive computing as well as self-organising feature maps for data mining [10] will be implemented.

4.3. Low-cost image processing

Online Photo Services (OPS) enable customers to prepare and order photo print products on-line by using their web browsers. The OPS splits into three parts: (1) delivery of static components like web pages, (2) providing dynamic contents such as the customer design tools, and (3) providing an image subsystem processing picture data operations (e.g. scaling, cropping, rotating). These tasks are performed in data centres, independently from the customer's hardware, and they must meet strict response time constraints, requiring a fast execution of image processing tasks.

There are considerable workload fluctuations during the season or even the day, which affect the required hardware resources. A heterogeneous hardware approach with different power/performance ratios is expected to be more flexible and energy efficient than a typical setup based on x86_64 servers. In particular, the usage of FPGAs for image processing tasks seems to be especially appropriate. An advantage of M2DC will be the intelligent power management, which enables the setting of unused hardware resources to idle mode to save energy.

4.4. High Performance Data Analytics

In case of accident, several non-automotive companies provide vehicle drivers with a support service for automatic request of assistance. The architecture of such service is based on three main components: a sensing unit for acceleration measurements, a localization unit for GPS reading, and a data processing and communication unit for identification of accidents and communication of position data. This approach is completely automated, from data collection (on the car) to data processing for accident recognition and classification (in the data centre) and operator call to the driver. The goal of this use case is to optimize both the TCO of cloud servers used, and the amount of data that can be processed within a given power and energy budget [20].

To accelerate data analytics, M2DC servers will leverage a modular compilation and runtime toolchain (derived from LLVM) interfacing the R language to an heterogeneous data flow execution runtime (currently based on StarPU runtime software [21]). A dedicated version of the compilation tool-chain will be developed on top of M2DC hardware and middleware, including new transformation and analysis passes at compiler level. The existing runtime will be extended to support the hardware and to use capabilities of the hardware/middleware pair. The underlying runtime, similarly to the existing one, will also be used to build programs without any compiler support and the transformation passes will be performed at IR (Intermediate Representation) level, enabling the use of other languages to target the same tool-chain at a lower cost (only the front-end would need to be rewritten).

4.5. Energy-optimized Platform as a Service

Platform as a Service (PaaS) allow customers to run their applications on a full stack environment, and therefore they need to support the most popular programming

languages and web applications. Computing power needs to be delivered by isolated runtime containers spawned on micro servers by cloud orchestration platforms, on demand. Low power servers and precise power and thermal management are crucial for keeping low energy costs while ensuring high availability and required QoS, so M2DC appliances should be a perfect fit for such scenarios.

5. Initial Benchmarking

In order to evaluate the performance of the M2DC systems, we have collected results from several standard benchmarking suites on our *Aarch64* testbed machine and compared them against different Intel-based 64-bit servers.

With the idea of being as thorough as possible, we have selected three different benchmarking suites:

- *LMbench*[22]: A low level benchmark that measures machine characteristics such as process context switch, system call and IPC latencies as well as memory bandwidth and arithmetic performance.

- *The HPC Challenge*[23]: A high level benchmarking suite comprising seven individual benchmarks: HPL (High Performance Linpack), DGEMM (Matrix-Matrix Multiplication), STREAM (sustainable memory bandwidth), PTRANS (large array transfer rate from multiprocessors), RandomAccess (rate of random updates on memory), FFT (Discrete Fourier Transform), RandomRing (latency and bandwidth of network communication using MPI).

- *SPEC©2006*[24]: Industry standard benchmark comprising a variety of real-world applications and kernels divided in two sets – SPECint and SPECfp – depending on whether the application or kernel is integer-intensive or floating point-intensive respectively.

In all three cases, the *Aarch64* testbed machine was based on a single-socket ARM 64-bit SoC, with 32 Cortex A57 cores running at 2.1GHz. The memory controller has four memory channels fully populated with DDR3 RDIMMs running at 1866 MHz (128 GB of RAM in total).

For *LMbench*, it was compared against a Xeon E3 based server running at 3.7GHz, with 4 cores and 16GB of DDR3 RAM at 1600 MHz. Figures 6(a)-6(e) show the results of the *Aarch64* testbed normalized against the *Xeon* server, meaning that results over one indicate an advantage for the *Aarch64*-based solution and that results under one instead indicate an advantage for the *Xeon*-based server.

For *The HPC Challenge*, results on the *ARM*-based server are normalized against the results in an *i7-4700EQ*-based workstation with 4 cores running at 2.4 GHz and 16GB of DDR3 at 1600 MHz, and the results can be seen in Figure 6(f).

Finally, for *SPEC©2006*, the results on the *Aarch64* testbed are normalized against the results obtained on a 2.1GHz *Xeon*-based server with similar characteristics to those of the *ARM*-based machine (16 cores, 2 threads per core, at 2.1 GHz and 128 GB of RAM); these results can be seen in Figures 6(g) and 6(h).

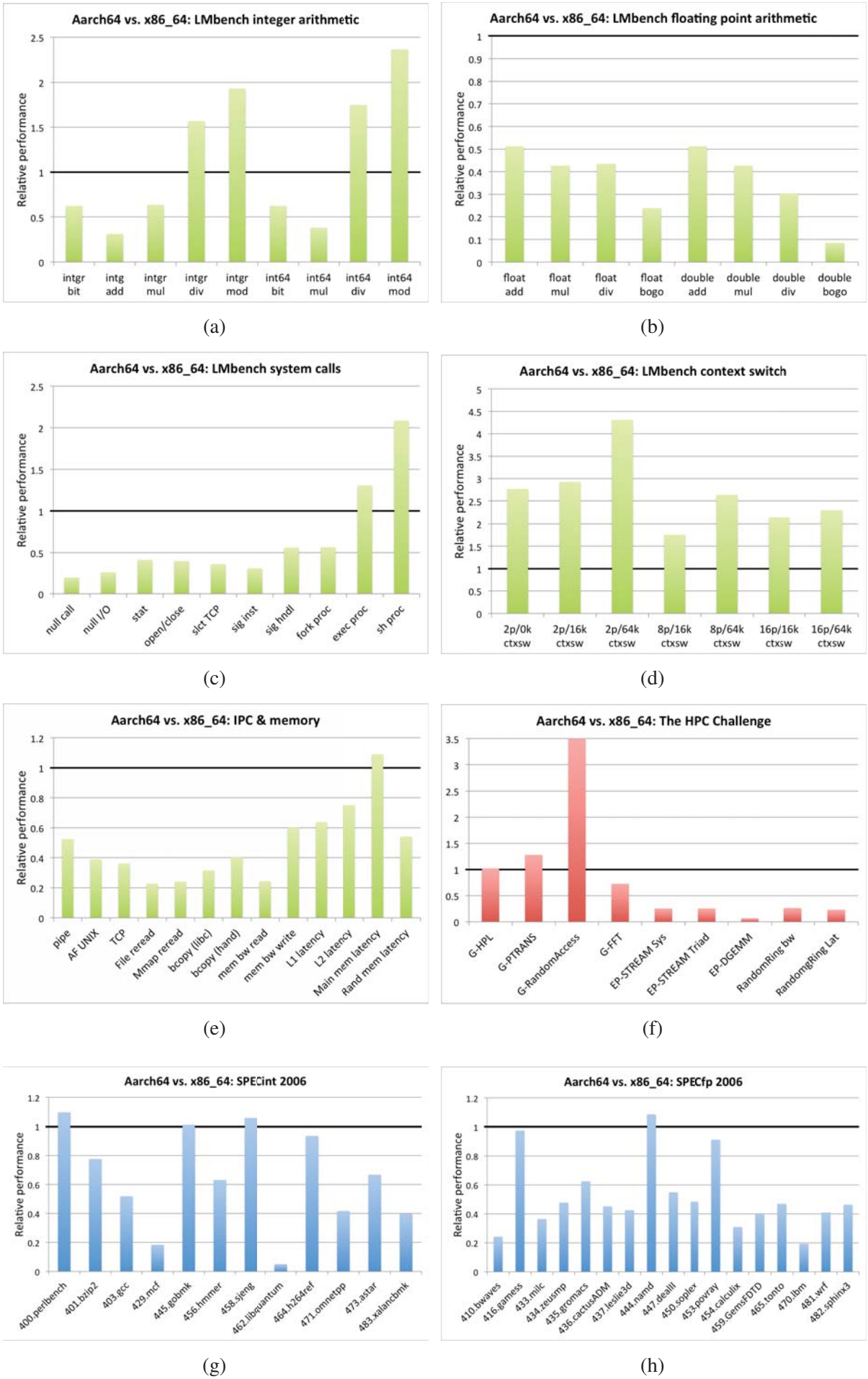

Figure 6: Relative performance for Aarch64 vs. x86_64 for LMbench (6(a)-6(e)), The HPC Challenge (6(f)), SPECint©2006 (6(g)) and SPECfp©2006 (6(h))

At this point, in raw arithmetic performance the Aarch64-based platform is still in clear disadvantage with respect to the current x86_64-based servers. Although integer division is significantly faster, Figures 6(a) and 6(b) show that the Aarch64 platform lags behind in all other integer operations and floating point operations in general. The SPEC©benchmarks (Figures 6(g) and 6(h)) confirm those results showing, on average, half of the whole performance of the x86_64 platform.

We can see a consistent superiority on the Aarch64 side regarding process maintenance (Figure 6(d)), although the behaviour for other system operations seems to be noticeably worse(6(c)).

Finally, in *The HPC Challenge* benchmark (Figure 6(f)), we can see that Aarch64 shows similar results for Linpack (G-HPL) and data transfer rates between processors (PTRANS), as well as a clear superiority in random memory updates (RandomAccess). On the other hard, x86_64 shows superiority in the double precision based tests (DGEMM and FFT) and the memory bandwidth and latency tests (STREAM and RandomRing).

In the future, these results will serve as a guide to drive the design efforts as well as a reference to gauge the advances achieved by M2DC.

5.1. High performance simulations with EULAG

EULAG is a numerical solver for all scale flow problems, as described in Section 4.1. In this section we focus on its performance and energy efficiency. We run the software on two hardware platforms, namely Intel Xeon and *Aarch64*. Our Intel platform consists of two dual-socket nodes IBM iDataPlex dx360 M4, each equipped with two Intel Xeon E5-2670 2.60 GHz (HT disabled, 2×8 cores), 16×32GB DDR3 running at 1333 MHz (512GB), and 10 Gbps Ethernet. For the *Aarch64* platform the specification was exactly the same as in Section 5, i.e. one node equipped with single-socket ARM 64-bit SoC, 32 Cortex A57 cores running at 2.1 GHz and 128 GB of DDR3 memory running at 1866 MHz.

For benchmarking purposes we ran EULAG using a standard test case for incompressible flow solvers, i.e. simulation of decaying turbulence of a homogeneous incompressible fluid. Three different grid resolutions were tested: $128 \times 128 \times 128$, $256 \times 256 \times 256$ and $512 \times 256 \times 256$. We focus here on the GCR pressure solver which is one of the most important element of the simulation. The performance was measured in Giga FLOPS, i.e. the number of floating point operations per second. All operations were performed in double precision. Note that, due to NDA, only relative values are presented in this paper. Consequently, the presented relative energy efficiency was originally captured in GFLOPJ, that is the number of floating point operations per Joule. Note that this is equivalent for GFLOPS/Watt. The electrical power (DC) used by the entire systems was read using IBM IMM II and using Huawei iBMC interfaces, for Intel and *Aarch64* servers, respectively.

Figure 7 presents the performance of *Aarch64* platform relative to the Intel-based system, depending on the number of cores used and the domain size. We can see that the Intel system outperforms *Aarch64* by a large margin. One interesting observation that may be made is the increasing relative performance of *Aarch64* up to 16 cores followed by a sudden decrease. This is caused by the fact that the Intel system has

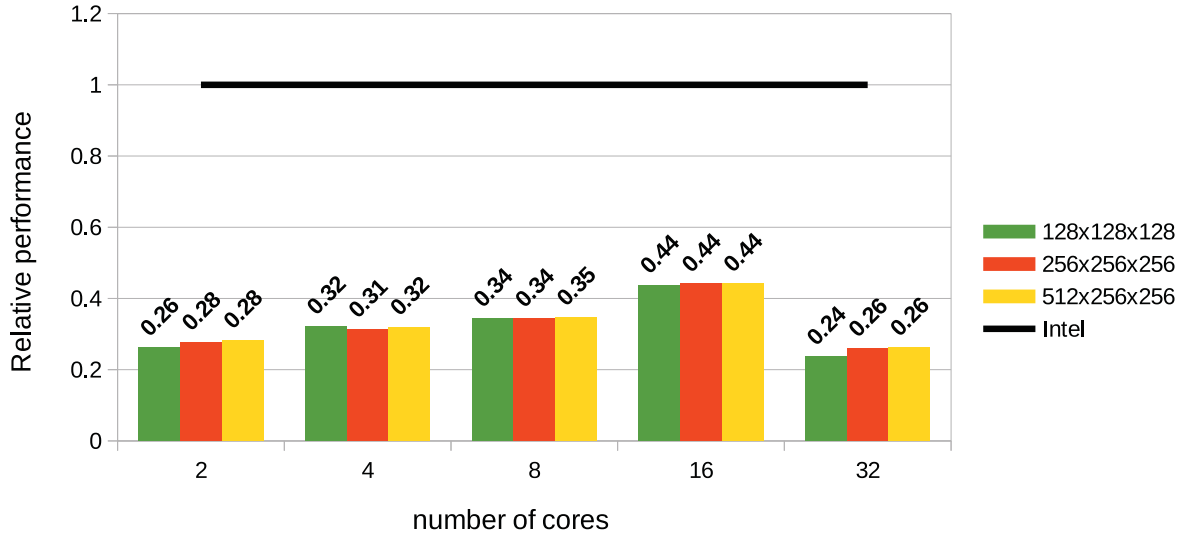

Figure 7: Performance of EULAG running on *Aarch64* relative to Intel platform. Relative performance depends on the number of cores used and the domain size.

slower memory modules which get saturated earlier than on *Aarch64*. However, the cumulative memory throughput doubles as EULAG starts to run on two Intel servers, i.e. on 32 cores.

Figure 8 presents the energy efficiency of *Aarch64* platform relative to the Intel-based system, again depending on the number of cores used and the domain size. With high level of parallelism the *Aarch64* platform performs better compared to the Intel system, by around 15-30%. This means that in this case *Aarch64* used less energy to perform the whole simulation. For the distributed run (32 cores) the domain size starts to make slight fluctuations. This may be partially caused by a large total amount of Intel CPU cache (here: 80 MB) starting to favor the smallest domain, in which case fewer memory transactions are needed. On the other hand, for 2-8 cores the Intel platform still has sufficient memory throughput to perform better than *Aarch64*, even on the energy efficiency front. Overall, we can conclude that *Aarch64* is an interesting architecture to look at, mainly when energy is the main concern.

5.2. Neural Network Simulation

5.2.1. Self-Organizing Maps

For early benchmarking of the targeted neural network implementation (Self-Organizing Maps – SOM), x86 CPU implementations and FPGA implementations based on Xilinx Virtex 5/7 FPGAs will be used as baseline until the new ARMv8 and Altera Stratix-10 based microservers become available. Early evaluation of FPGAs is seen crucial to motivate further optimizations on the M2DC server. For the x86 CPU, an Intel Core i7-4770K quad-core CPU is selected. OpenMP 2.0 is used to realize an optimized multi-threaded implementation and the Streaming SIMD Extensions (SSE) of the Intel CPU are added using SSE2 intrinsics.

RAPTOR-XPress, a modular FPGA-based hardware accelerator, is used for the FPGA implementation, equipped with Xilinx Virtex 5 (XC5VFX100T-2) or Virtex 7

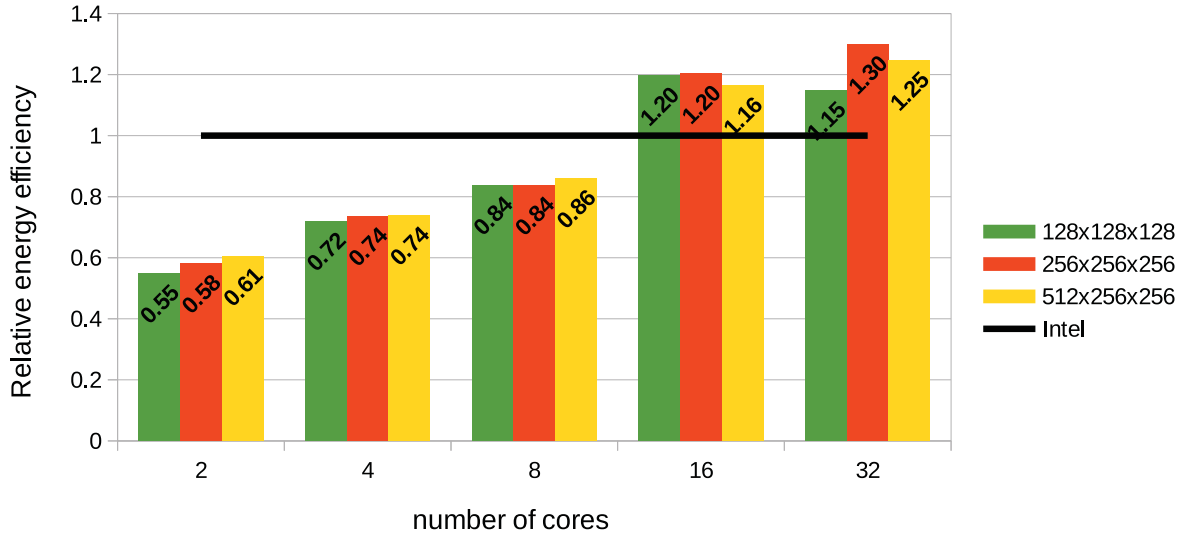

Figure 8: Energy efficiency of EULAG running on *Aarch64* relative to Intel platform. Relative energy efficiency depends on the number of cores used and the domain size.

FPGAs (XC7VX690T-2) [25]. The implementation is an extended version of the gNBXe-System presented in [10], especially utilizing the high-speed interconnect of the FPGAs to enable scalability beyond single accelerator systems. A single master FPGA performs all required control and synchronization as well as communication with the host system while an arbitrary number of processing FPGAs is dedicated to the main computation tasks. At design time, the architecture can be flexibly parameterized in terms of number of FPGAs and processing elements to optimally fulfill the application requirements.

To cover a wide range of up-to-date SOM applications, four different synthetic datasets are defined and trained for SOMs with various numbers of neurons. For representation of lower dimensional data (e.g., for multi-spectral image analysis) a dataset with 16-dimensional feature vectors is used (dat_{16}). To benchmark the implementations for medium to high-dimensional data (e.g, hyperspectral image analysis or bioinformatics applications) two datasets are selected, using 104-dimensional and 200-dimensional vectors (dat_{104} and dat_{200}) respectively. To further evaluate how efficient vectors with hundreds of features can be processed, a 1000-dimensional dataset is generated (dat_{1000}).

The classification performances of the evaluated platforms depend on the size of the SOM (see Table 2). For Core i7 based SOM processing, the maximum classification performance is achieved with $1,000$-dimensional feature vectors for $1,400$ neurons, resulting in about 26 GCPS (Giga Connections per Second) for *i7-sse* implementation. Using the *FPGA-V7* SOM implementation, a maximum classification performance of 5.7 TCPS (Tera Connections per Second) is achieved with $84,000$ neurons and $1,000$ dimensional feature vectors, clearly outperforming the other implementations.

The classification efficiency of the different implementations is shown in Figure 9. In the left column, the efficiencies of the Core i7 implementations (*i7*, *i7-sse*, and *i7-sse-int*) for SOMs with 100 to $84,000$ neurons (N_N) and datasets with 16 to 1000

vector components are shown. For low-dimensional data, the *i7-sse-int* implementation is most efficient and about 1.4 to 1.6 times more efficient than the *i7-sse* and the *i7* version respectively. With increasing dimensionality of the datasets, the most-efficient implementation depends on the required size of the SOM. For smaller SOM sizes *i7-sse* benefits more from SSE instructions; with growing numbers of neurons, the effects of cache size limitations diminish the advantage of streaming SIMD operation with floating point accuracy and *i7-sse-int* is about 2.3 times more efficient.

Using the modular RAPTOR-XPress platform, a large number of processing FPGAs – and thus parallelism – can be utilized. The energy efficiency of the FPGA implementations is presented in the right column of Figure 9. Due to the small size of the Virtex-5 FPGA, and the therefore required large number of FPGAs, the *FPGA-V5* efficiency gain compared to the *i7-sse-int* implementation is relatively small. It starts at about two for dat_{16} and increases to three for dat_{1000}. Using the *FPGA-V7* implementation, the achieved gain increases to 11 for dat_{16} and 28 for dat_{1000}.

5.2.2. Deep Learning

Concerning SEEs for Deep Learning, the current work consists in porting PNeuro (presented in Section 2.3.1) on an FPGA microserver to be evaluated in a datacentre infrastructure. Existing benchmarks show interesting results, even when PNeuro is implemented on high-end FPGAs (Xilinx Kintex-7 K480T). For example, Table 3 shows a comparison of different implementations of a sample CNN structure categorizing pictures of faces, planes, motorbikes, and cars on a quad ARM A7 and a quad ARM A15 using OpenMP as well as a Tegra K1 platform using OpenCL. The PNeuro implementation is composed of one cluster of four NeuroCores which is far from filling the entire FPGA space. The expected performance on M2DC hardware is meant to be at least the same since PNeuro is not on the critical path of the FPGA system. From the available results, one can see that PNeuro on FPGA is approximately five times more power efficient than a Tegra K1 in batch mode.

6. Related Works

To achieve its objectives, M2DC takes as baseline the results of different EU FP7 projects, to which some of the members of the M2DC participate.

Table 2: Maximum performance (CP) in GCPS (Giga Connections per Second) for SOMs with 100 to 84,000 neurons (N_N) and 16 (dat_{16}) to 1000 (dat_{1000}) vector components.

		dat_{16}	dat_{104}	dat_{200}	dat_{1000}
i7-sse	CP [GCPS]	8.5	22.3	23.9	25.9
	N_N	84k	14k	7k	1.4k
i7-sse-int	CP [GCPS]	11.8	18.5	19.1	19.9
	N_N	84k	28k	18k	3k
FPGA-V7	CP [GCPS]	178	776	1,235	5,711
	N_N	84k	84k	84k	84k

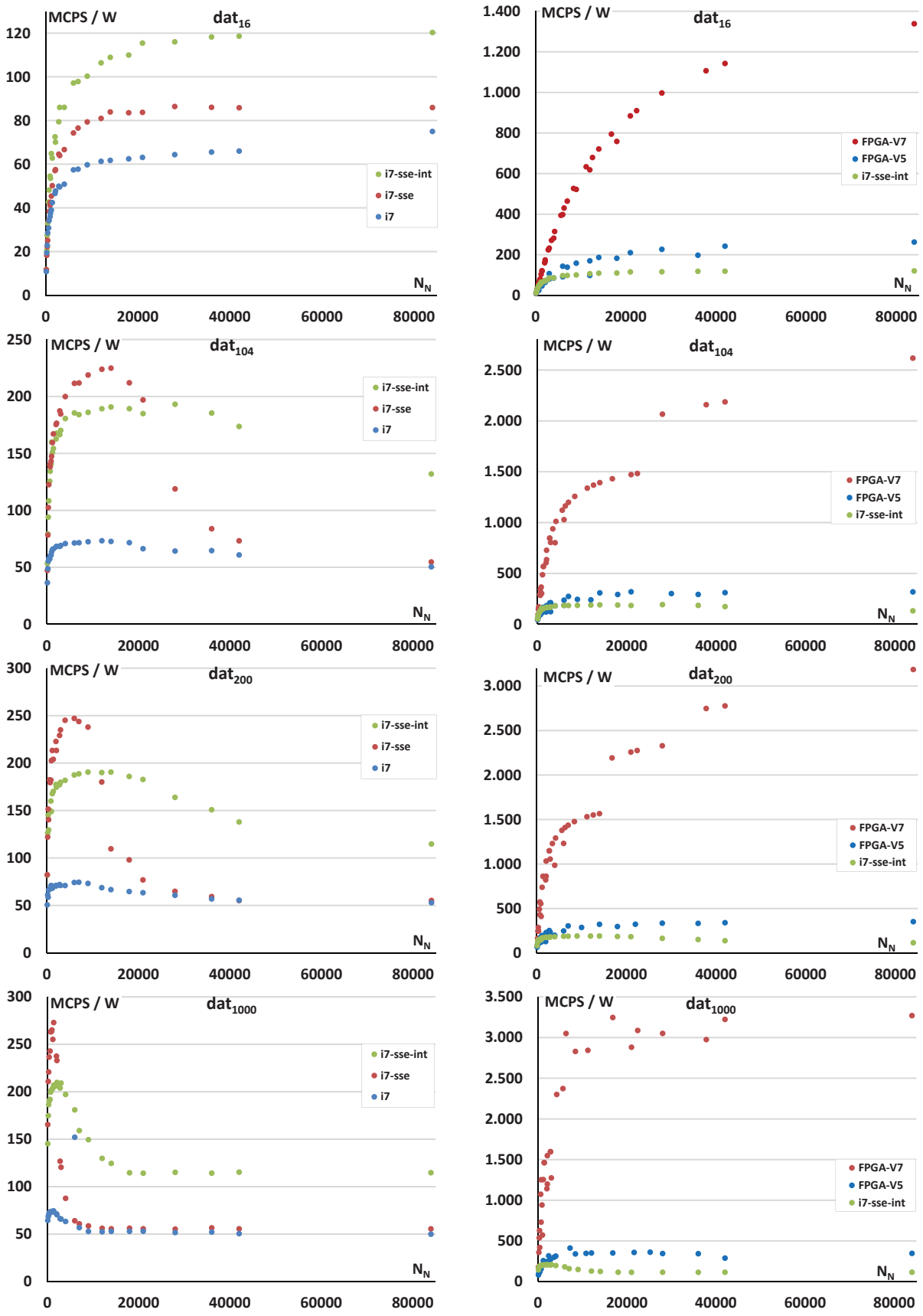

Figure 9: Classification efficiency in MCPS/W of Core i7 and FPGA based SOM implementations for synthetic datasets with different number of features per vector and varying number of neurons. Left column is Core i7, right column is FPGA, datasets from top to bottom are dat_{16}, dat_{104}, dat_{200} and dat_{1000}.

Table 3: Performance of PNeuro FPGA CNN implementation

Target	Frequency (MHz)	Performance (images/s)	Power Efficiency (images/s/W – SoC only)
Quad ARM A7	900	480	380
Quad ARM A15	2000	870	350
Tegra K1	850	3550	600
PNeuro (FPGA)	100	7000	2800

FiPS [6, 26] results will be reused for the management of heterogeneity in a server infrastructure. In particular M2DC will take the RECS3.0 prototype currently under development as a direct basis for further enhancements, and the application mapping methodology will also be developed towards the goal of turnkey appliances. RECS3.0 is in turn based on RECS2.0, a result of CoolEmAll [27].

The Mont-Blanc project [28] used commodity energy-efficient embedded technology to build a prototype High Performance Computing system based on ARMv7 (32-bit) System-on-Chips. A next-generation system architecture design based on ARMv8 (64-bit) technology is being explored in the Mont-Blanc 2 project. These projects have been pivotal in building up the software ecosystem required for ARM-based HPC: system software, networking and communication libraries, support for heterogeneous processing and compute acceleration, much of which is also required for server applications.

FP7 project EUROSERVER [29] advocates the use of state-of-the-art low-power ARM processors in a new server system-architecture that uses 3D integration to scale with both the numbers of cores, and the memory and I/O. While M2DC does not design new chips, the chiplet technology used in EUROSERVER and the chips themselves could be seen as interesting opportunities to add another level of flexibility by designing fully heterogeneous low cost and low power chips to be integrated in the M2DC server.

Results from the FP7 project DEEP-ER [30] concerning benefits of NVM memory could be of high interest on high-end versions of future M2DC appliances, especially those aiming to execute HPC applications. Additionally, advanced checkpointing techniques could improve the reliability of M2DC appliances, if applicable to microserver-based systems.

7. Conclusions

We have presented an overview of M2DC, an EU H2020 project started in January 2016 and aimed at defining a modular microserver architecture for future data centres. M2DC aims at reducing TCO for selected applications and use cases by 50% compared to other servers. Depending on selected system configuration and application, it also improves energy-efficiency by 10-100 times, compared to 2013 typical servers. .

To this end, M2DC proposes a flexible, high-density, cost-optimised server architecture able to benefit from the wide variety of available computing architectures, from low power computing resources to accelerators. Maintainability and modularity will be ensured by proper design, based on established standards, while advanced power and

thermal management techniques, based on numerous sensors and fine-grain resource provisioning as well as system efficiency enhancements, will dramatically improve the general efficiency of the system, from the performance to energy consumption points of view. Finally, seamless integration into existing data centre infrastructures will be guaranteed by the compatibility with de-facto standards from the hardware and software points of view.

We also proposed an initial performance analysis of the *Aarch64* ARM architecture, which will serve as the low-power general purpose processor for the microservers.

Acknowledgements

This work was supported in part by the European Union's Horizon 2020 research and innovation programme, under grant 688201, Modular Microserver DataCentre (M2DC).

References

[1] Cisco. Cisco Global Cloud Index: Forecast and Methodology 2013-2018 White Paper. `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.pdf`, 2014.

[2] M. Duranton, K. De Bosschere, A. Cohen, J. Maebe, and H. Munk. The HiPEAC Vision 2015. http://www.hipeac.net/roadmap, January 2015.

[3] M. Cecowski, G. Agosta, A. Oleksiak, M. Kierzynka, M. v. d. Berge, W. Christmann, S. Krupop, M. Porrmann, J. Hagemeyer, R. Griessl, M. Peykanu, L. Tigges, S. Rosinger, D. Schlitt, C. Pieper, C. Brandolese, W. Fornaciari, G. Pelosi, R. Plestenjak, J. Cinkelj, L. Cudennec, T. Goubier, J. M. Philippe, U. Janssen, and C. Adeniyi-Jones. The m2dc project: Modular microserver datacentre. In *2016 Euromicro Conference on Digital System Design (DSD)*, pages 68–74, Aug 2016.

[4] PICMG. PICMG COM.0 R2.1 - Com Express Module Base Specification. *Available at* `http://www.picmg.org`. Accessed: 13-August-2015.

[5] Toradex. Apalis Computer Module - Module Specification. *Available at* `http://developer.toradex.com/hardware-resources/arm-family/apalis-module-architecture`. Accessed: 13-August-2015.

[6] René Griessl, Meysam Peykanu, Jens Hagemeyer, Mario Porrmann, Stefan Krupop, Micha vor dem Berge, Thomas Kiesel, and Wolfgang Christmann. A scalable server architecture for next-generation heterogeneous compute clusters. In *Proceedings of the 2014 12th IEEE International Conference on Embedded and Ubiquitous Computing*, EUC '14, pages 146–153, Washington, DC, USA, 2014. IEEE Computer Society.

[7] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[8] Marc Weber, Hanno Teeling, Sixing Huang, Jost Waldmann, Mariette Kassabgy, Bernhard M Fuchs, Anna Klindworth, Christine Klockow, Antje Wichels, Gunnar Gerdts, et al. Practical application of self-organizing maps to interrelate biodiversity and functional data in NGS-based metagenomics. *The ISME journal*, 5(5):918–928, 2011.

[9] U. Siripatrawan and Y. Makino. Monitoring fungal growth on brown rice grains using rapid and non-destructive hyperspectral imaging. *International Journal of Food Microbiology*, 199:93 – 100, 2015.

[10] Jan Lachmair, E. Merényi, Mario Porrmann, and Ulrich Rückert. A reconfigurable neuroprocessor for self-organizing feature maps. *Neurocomputing*, 112(SI), 2013.

[11] Giovanni Agosta, Alessandro Barenghi, Alessandro Di Federico, and Gerardo Pelosi. Opencl performance portability for general-purpose computation on graphics processor units: an exploration on cryptographic primitives. *Concurrency and Computation: Practice and Experience*, 27(14):3633–3660, 2015.

[12] Giovanni Agosta, Alessandro Barenghi, and Gerardo Pelosi. Exploiting Bit-level Parallelism in GPGPUs: a Case Study on KeeLoq Exhaustive Search Attacks. In Gero Mühl, Jan Richling, and Andreas Herkersdorf, editors, *ARCS 2012 Workshops, 28 Feb.- 2 Mar. 2012, München Germany*, volume 200 of *LNI*, pages 385–396. GI, 2012.

[13] Giovanni Agosta, Alessandro Barenghi, Fabrizio De Santis, and Gerardo Pelosi. Record Setting Software Implementation of DES Using CUDA. In Shahram Latifi, editor, *Seventh International Conference on Information Technology: New Generations, ITNG 2010, Las Vegas, Nevada, USA, 12-14 April 2010*, pages 748–755. IEEE Computer Society, 2010.

[14] Tim Dierks. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246, August 2008.

[15] NIST Computer Security Division. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. FIPS Publication 202, National Institute of Standards and Technology, U.S. Department of Commerce, May 2014.

[16] Wojciech Piatek, Ariel Oleksiak, and Georges Da Costa. Energy and thermal models for simulation of workload and resource management in computing systems. *Simulation Modelling Practice and Theory*, 2015.

[17] Milosz Ciznicki, Piotr Kopta, Michal Kulczewski, Krzysztof Kurowski, and Pawel Gepner. Elliptic solver performance evaluation on modern hardware architectures. In *Parallel Processing and Applied Mathematics*, pages 155–165. Springer, 2013.

[18] Krzysztof Andrzej Rojek, Milosz Ciznicki, Bogdan Rosa, Piotr Kopta, Michal Kulczewski, Krzysztof Kurowski, Zbigniew Pawel Piotrowski, Lukasz Szustak, Damian Karol Wojcik, and Roman Wyrzykowski. Adaptation of fluid model eulag to graphics processing unit architecture. *Concurrency and Computation: Practice and Experience*, 27(4):937–957, 2015.

[19] Milosz Ciznicki, Michal Kulczewski, Piotr Kopta, and Krzysztof Kurowski. Scaling the gcr solver using a high-level stencil framework on multi-and many-core architectures. In *Parallel Processing and Applied Mathematics*, pages 594–606. Springer, 2016.

[20] Ariel Oleksiak, Michal Kierzynka, Giovanni Agosta, Carlo Brandolese, William Fornaciari, and Gerardo Pelosi, et. al. Data Centres for IoT applications: the M2DC Approach (Invited Paper). In *IEEE International Conference on Embedded Computer Systems: Architectures, MOdeling, and Simulation (IC-SAMOS 2016)*. IEEE, July 2016.

[21] Cédric Augonnet, Samuel Thibault, Raymond Namyst, and Pierre-André Wacrenier. StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures. *Concurr. Comput. : Pract. Exper.*, 23(2):187–198, February 2011.

[22] L. McVoy and C. Staelin. lmbench: Portable tools for performance analysis. In *Proceedings of the USENIX 1996 Annual Technical Conference*, pages 279–284, San Diego, CA, USA, Jan 1996.

[23] Piotr R Luszczek, David H Bailey, Jack J Dongarra, Jeremy Kepner, Robert F Lucas, Rolf Rabenseifner, and Daisuke Takahashi. The HPC Challenge (HPCC) Benchmark Suite. In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, SC '06, New York, NY, USA, 2006. ACM.

[24] John L. Henning. SPEC CPU2006 Benchmark Descriptions. *SIGARCH Comput. Archit. News*, 34(4):1–17, September 2006.

[25] Mario Porrmann, Jens Hagemeyer, Christopher Pohl, Johannes Romoth, and Manuel Strugholtz. RAPTOR - A Scalable Platform for Rapid Prototyping and FPGA-based Cluster Computing. In Chapman, B. et al., editor, *Parallel Computing: From Multicores and GPUs to Petascale*, pages 592–599, Lyon, France, 2010. IOS Press.

[26] Yves Lhuillier, Jean Marc Philippe, Alexandre Guerre, Michal Kierzynka, and Ariel Oleksiak. Parallel architecture benchmarking: From embedded computing to hpc, a fips project perspective. In *Proceedings of the 2014 12th IEEE International Conference on Embedded and Ubiquitous Computing*, EUC '14, pages 154–161, Washington, DC, USA, 2014. IEEE Computer Society.

[27] M. vor dem Berge, W. Christmann, E. Volk, S. Wesner, A. Oleksiak, T. Piontek, G. Da Costa, and J. M. Pierson. CoolEmAll - Models and tools for optimization of data center energy-efficiency. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2012*, pages 1–5, Oct 2012.

[28] Nikola Rajovic, Paul M. Carpenter, Isaac Gelado, Nikola Puzovic, Alex Ramirez, and Mateo Valero. Supercomputing with Commodity CPUs: Are Mobile SoCs Ready for HPC? In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 40:1–40:12, New York, NY, USA, 2013. ACM.

[29] Y. Durand, P. M. Carpenter, S. Adami, A. Bilas, D. Dutoit, A. Farcy, G. Gaydadjiev, J. Goodacre, M. Katevenis, M. Marazakis, E. Matus, I. Mavroidis, and J. Thomson. EUROSERVER: Energy Efficient Node for European Micro-Servers. In *Digital System Design (DSD), 2014 17th Euromicro Conference on*, pages 206–213, Aug 2014.

[30] Norbert Eicker. Taming Heterogeneity by Segregation – The DEEP and DEEP-ER take on Heterogeneous Cluster Architectures. `http://hdl.handle.net/2128/9379`, 2015.