The Mont-Blanc prototype: An Alternative Approach for High-Performance Computing Systems

UPC-DAC-RR-CAP-2016-2 Technical Report, March 2016.

Nikola Rajovic[*†], Alex Ramirez[x★], Alejandro Rico[‡★], Filippo Mantovani[*], Daniel Ruiz[*†], Oriol Villarubi[*], Constantino Gomez[*†], Luna Backes[*], Diego Nieto[*], Harald Servat[*], Xavier Martorell[*†], Jesus Labarta[*†], Eduard Ayguade[*†], Mateo Valero[*†], Chris Adeniyi-Jones[‡], Said Derradji[§], Herve Gloaguen[§], Piero Lanucara[¶], Nico Sanna[¶], Jean-Francois Mehaut[‖], Kevin Pouget[‖], Brice Videau[‖], Eric Boyer[**], Momme Allalen[††], Axel Auweter[††], David Brayford[††], Daniele Tafani[††], Dirk Bröemmel[‡‡], René Halver[‡‡], Jan H. Meinke[‡‡], Ramon Beivide[xi], Mariano Benito[xi], and Enrique Vallejo[xi]

[*] Barcelona Supercomputing Center, first.last@bsc.es
[†] Computer Architecture Department, Universitat Politecnica de Catalunya - BarcelonaTech
[‡] ARM Ltd. [§] Bull/ATOS [¶] Cineca [‖] CNRS - Universite Grenoble Alpes - LIG [**] GENCI
[‡‡] Forschungszentrum Jülich GmbH [††] LRZ [x] NVIDIA [xi] Universidad de Cantabria

✦

Abstract—High-performance computing (HPC) is recognized as one of the pillars for further advance of science, industry, medicine, and education. Current HPC systems are being developed to overcome emerging challenges in order to reach Exascale level of performance, which is expected by the year 2020. The much larger embedded and mobile market allows for rapid development of IP blocks, and provides more flexibility in designing an application-specific SoC, in turn giving possibility in balancing performance, energy-efficiency and cost. In the Mont-Blanc project, we advocate for HPC systems be built from such commodity IP blocks, currently used in embedded and mobile SoCs.

As a first demonstrator of such approach, we present the Mont-Blanc prototype; the first HPC system built with commodity SoCs, memories, and NICs from the embedded and mobile domain, and off-the-shelf HPC networking, storage, cooling and integration solutions. We present the system's architecture, and evaluation including both performance and energy efficiency. Further, we compare the system's abilities against a production level supercomputer. At the end, we discuss parallel scalability, and estimate the maximum scalability point of this approach across a set of applications.

1 Introduction

High-Performance Computing (HPC) systems evolution is driven by the need of reducing time-to-solution and increasing resolution of models of problems being solved by a particular program. Important milestones from the HPC system performance perspective were achieved using commodity technology. Examples are the ASCI Red and the Roadrunner supercomputers, which broke the 1TFLOPS and 1PFLOPS barriers, respectively. These systems showed how commodity technology could be used to make the next step in HPC system architecture.

★ *The majority of the work was done while author was with Barcelona Supercomputing Center.*

The short life cycle of commodity components prevents special-purpose components to compete. Driven by a bigger market, commodity processors develop quicker and are cheaper. For this reason, RISC processors displaced vectors, and x86 displaced RISC.

Nowadays commodity is in the mobile segment. Mobile processors develop fast, and are still not at a point of diminishing returns in design and incorporate increasingly powerful processing capabilities, required for HPC. Further, the market size and customer requirements allow for constant investments into innovative designs.

The size of the market allows for testing and adoption of new technology very fast. For example, LPDDR memory technology first entered into mobile domain and recently is proposed as a solution for energy proportional servers [1]. Moreover, NVIDIA started introducing new revisions of their GPU architecture first in mobile segment, and later in the gaming workstations, and in turn into HPC.

The Mont-Blanc project aims at providing an alternative HPC system solution based on the current commodity technology: mobile chips. As a demonstrator of such approach, the project designed, built and set-up a 1080-node HPC cluster made of Samsung Exynos 5 Dual processors. The Mont-Blanc project established the following goals: to design and deploy a sufficiently large HPC prototype system based on the current mobile commodity technology; to port and optimize software stack and enable its use for HPC; to port and optimize a set of HPC applications to be run at this HPC system.

Contributions of this report are as follows:

- We reveal the architecture and the design specifics of the Mont-Blanc prototype.
- We present thorough performance and power evaluation of the prototype, and compare it to a Tier-0 production system in Europe, the MareNostrum supercomputer.
- We provide a set of recommendations for the next-generation HPC system built around the Mont-Blanc approach.

2 The Mont-Blanc Prototype

In this Section we present the architecture of the Mont-Blanc prototype. We highlight peculiarities of each building block as we introduce them.

2.1 The Mont-Blanc Compute Node

The Mont-Blanc compute node is a *Server-on-Module* architecture. Figure 2 depicts the Mont-Blanc node card (Samsung Daughter Card or SDB) and its components. Each SDB is built around a Samsung Exynos 5250 mobile SoC integrating ARM Cortex-A15 CPUs at 1.7 GHz in a dual core configuration sharing 1 MB of on-die L2 cache, and a mobile ARM Mali-T604 GPU. The SoC connects to the on-board 4 GB of LPDDR3-1600 RAM through two memory channels shared among the CPUs and GPU, providing a peak memory bandwidth of 12.8 GB/s.

The node's interconnect interface is provided through an on-die USB 3.0 interface utilizing a discrete on-board NIC (integrating USB 3.0 to 1 Gb Ethernet bridge) [2], and the additional fabric providing Ethernet PHY. An external 16 GB μSD card is connected to a μSD memory slot and provides boot-loader, OS system image and local scratch storage.

The node connects to the blade through a proprietary bus using a PCI-e 4x form factor edge connector (EMB connector).

2.2 The Mont-Blanc Blade

In Figure 1b we depict the physical view of a Mont-Blanc blade, while in Figure 3 we show the architecture of the Mont-Blanc blade (Ethernet Mother Board or EMB). The blade hosts 15 Mont-Blanc nodes which are interconnected through an on-board 1 Gb Ethernet switch fabric. The switch provides two 10 GbE up-links. In addition, the EMB provides management services, power consumption monitoring of SDBs, and blade level temperature monitoring. The EMB enclosure is air-cooled through the fans installed on the front side.

2.3 The Mont-Blanc System

The entire Mont-Blanc prototype system (see Figure 1a) fits into two standard 42U-19″ racks. Each Mont-Blanc rack hosts four bullx chassis which in turn integrate nine Mont-Blanc blades each. In addition, racks are populated each with 2U Cisco Nexus 5596UP *top-of-the-rack* switches, 1U prototype management switch and 2U storage nodes.

2.3.1 System interconnect

The Mont-Blanc prototype implements two separate networks - management and cluster network. The management network is out of the scope of this paper, thus we depict the implementation of the cluster interconnect in Figure 4.

The first level of switching is implemented inside the blades using a 1 Gb Ethernet switch fabric providing two 10 Gb Ethernet up-links. Switching between the blades occurs at the *top-of-the-rack* switches with a switching capacity of 1.92 Tbps per switch. The racks are directly connected with four 40 Gb Ethernet links. The maximum hop distance between two nodes belonging to two different racks is four.

2.3.2 Storage

The storage system is based on a Supermicro Storage Bridge Bay based on x86-64 architecture, with a total capacity of 9.6 TB providing 2-3.5 GB/s read/write bandwidth (depending on the disk zone). The storage system is connected to the top-of-the-rack switches with four 10 Gb Ethernet links. The parallel file system services are provided with Lustre.

2.3.3 Cooling

The Mont-Blanc prototype is an air-cooled cluster. Each node has a passive top-mounted heat sink. Further, every Mont-Blanc blade is a separate cooling entity having its own set of front-mounted fans and a temperature control loop. Inlet and outlet temperatures are monitored and fan speed is adjusted accordingly.

2.4 The Mont-Blanc Software Stack

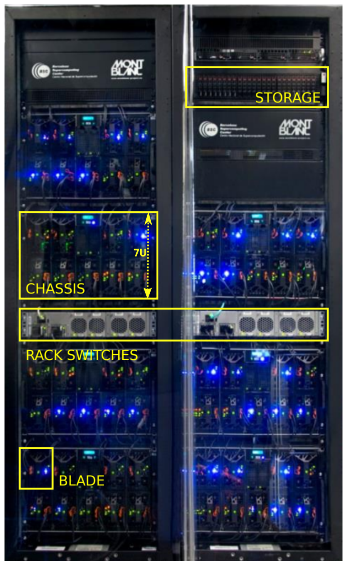
The work done during the Mont-Blanc project helped maturing the software stack on the ARM architecture. Today, working with the Mont-Blanc prototype feels like working with any other HPC cluster.

The Mont-Blanc prototype nodes are running Ubuntu 14.04.1 Linux distribution on top of the customized Kernel version 3.11.0 which enables user space driver for OpenCL programming of ARM Mali-T604 GPU. The rest of the software stack components are shown in Figure 5.

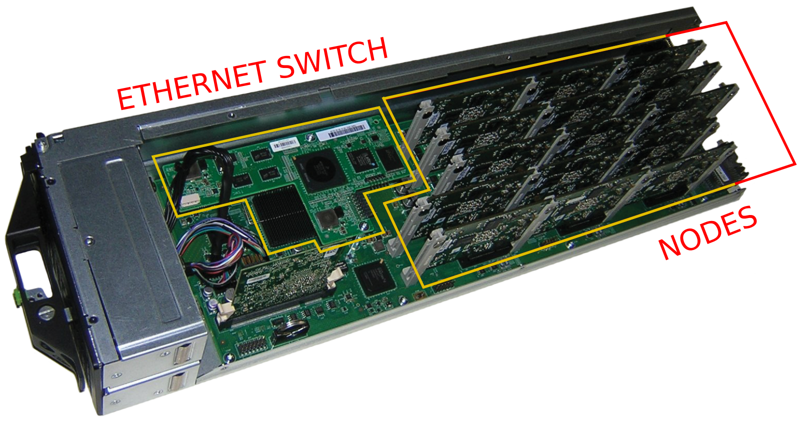
2.5 Power Monitoring Infrastructure

The Mont-Blanc system features a digital current and voltage meter in the power supply rail to each SDB. For collecting measurements at high frequencies, an FPGA on each EMB accesses all power sensors in a blade via I2C and stores their averaged value every 1120 ms in a FIFO buffer. The Board Management Controller (BMC) on the EMB communicates with the FPGA to collect the power data samples from the FIFO before storing them in its DDR2 memory along with a timestamp of the reading. User access to the data is then provided by the BMC over Ethernet through a set of custom Intelligent Platform Management Interface (IPMI) commands [3]. The concept of offloading high-frequency tasks to the FPGA and the ability to do bulk-transfers of power samples via IPMI contribute significantly to the high scalability and performance of this solution.

To provide application developers with power traces of their applications, the power measurement and acquisition



(a)



(b)

Fig. 1: Physical view of a) the Mont-Blanc prototype system, and b) the Mont-Blanc blade.

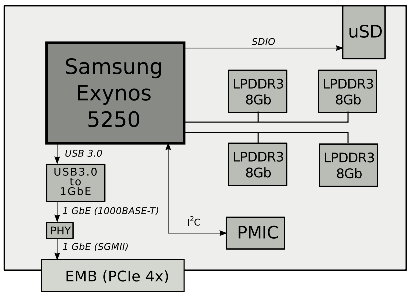


Fig. 2: The Mont-Blanc node block scheme (not to scale).

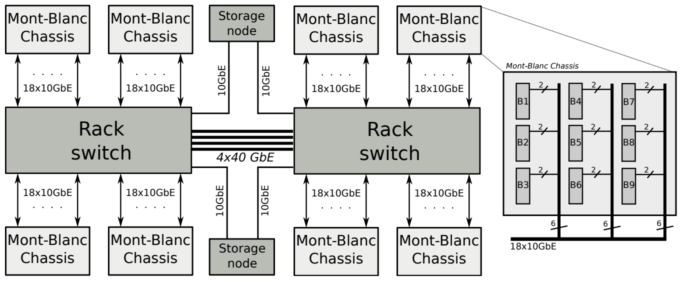


Fig. 4: The Mont-Blanc System Interconnect.

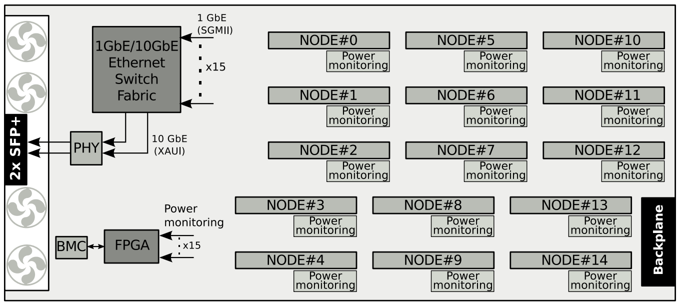


Fig. 3: The Mont-Blanc blade block scheme.

Compilers	
GNU JDK Mercurium	
Scientific libraries	
ATLAS LAPACK SCALAPACK FFTW	
BOOST clBLAS clFFT PETSc HDF5	
Performance analysis	**Debugger**
EXTRAE Paraver Scalasca	Alinea DDT
Runtime libraries	
Nanos++ OpenCL OpenMPI MPICH3	
Cluster management	
SLURM Nagios Ganglia	
Hardware support	**Storage**
Power monitor	LustreFS
Operating System	
Ubuntu	

Fig. 5: The Mont-Blanc Software Stack

process is conveniently encapsulated and automated in a custom-made system monitoring tool. The tool is developed with a focus on simplicity and scalability by respectively employing MQTT [4], for lightweight transport messaging, and Apache Cassandra [5], a scalable, distributed database for storing the acquired power data along with other time-series based monitoring data. The tool consists of three major software components: one or more instances of Pushers, Collect Agents and the Apache Cassandra distributed key-value store. The Pushers are responsible for polling sensor data (power consumption, temperatures, etc.) and forward the data using MQTT to a Collect Agent. This means that the Pusher for power data implements the necessary steps to perform the IPMI-based power acquisition process. The Collect Agents, as the name implies, collect all monitored information from one or multiple Pushers and insert it into the distributed Cassandra key-value, where all monitored information is finally stored and made available for further analysis. A set of command line tools and a special API provide users with the ability to access the raw monitoring data or to plot and correlate information from different data

TABLE 1: Mont-Blanc compute performance summary.

	Compute Node	
	CPU	**GPU**
Type	2×ARM Cortex-A15	1×ARM Mali-T604
Frequency	1.7 GHz	533 MHz
Peak performance (SP)	27.2 GFLOPS	72.5 GFLOPS
Peak performance (DP)	6.8 GFLOPS	21.3 GFLOPS
Memory (shared)	4 GB LPDDR3-800	
	Blade = 15×Node	
Peak performance (SP)	408 GFLOPS	1.08 TFLOPS
Peak performance (DP)	102 GFLOPS	319.5 GFLOPS
Memory	60 GB	
	Chassis = 9×Blade	
Peak performance (SP)	3.67 TFLOPS	9.79 TFLOPS
Peak performance (DP)	0.92 TFLOPS	2.88 TFLOPS
Memory	540 GB	
	System = 8×Chassis	
Peak performance (SP)	29.38 TFLOPS	78.3 TFLOPS
Total (SP)	*107.7 TFLOPS*	
Peak performance (DP)	7.34 TFLOPS	23 TFLOPS
Total (DP)	*30.3 TFLOPS*	
Memory	4.32 TB	

sources throughout the system. The intrinsic scalability of the Cassandra data model as well as the ability to add additional Pushers and Collect Agents as needed make this solution highly scalable and well-suited for the high-frequency power measurements of the Mont-Blanc system.

2.6 Performance Summary

Table 1 shows performance figures of the Mont-Blanc prototype. The two Cortex-A15 cores provide a peak performance of 27.2 GFLOPS in single-precision (SP) and 6.8 GFLOPS in double-precision (DP). The performance disparity comes from the fact that the SIMD unit, codenamed NEON, supports only SP floating-point (FP) operations, thus DP FP are executed in a scalar unit.

The on-chip embedded Mali-T604 GPU integrates four cores providing a total of 72.5 GFLOPS SP and 21.3 GFLOPS DP [6]. The total node peak performance is thus close to 100 GFLOPS SP and 28.1 GFLOPS DP.

Given that exploiting the GPU requires OpenCL application support, Table 1 shows the peak performance at the blade, chassis and entire system levels both for just the cores and the total compute (cores+GPU).

Due to the 32-bit nature of the SoC architecture, each node integrates 4 GB of memory. The high node integration density of 1080 nodes (2160 cores) in 56U (over 19 nodes per U) adds up to 4.32 TB of memory.

3 Compute Node Evaluation

In this Section we present comparison between the Samsung Exynos 5 SoC used in the Mont-Blanc prototype, and its contemporary - 8 core Intel Xeon E5-2670 server processor used in the current MareNostrum supercomputer [7]. MareNostrum node is designed as a dual-socket solution, utilizing DDR3-1600 memory DIMMs. It is worth mentioning that both chips were introduced in the year 2012; the former in Q3 and the later in Q1.

We present and discuss both core to core, and node to node performance and power comparative figures when executing Mont-Blanc benchmark suite [8].

3.1 Core evaluation

In Figure 6 we present performance comparison on a core-to-core basis between the Mont-Blanc prototype and MareNostrum supercomputer. We report performance difference by normalizing both execution times to execution on MareNostrum.

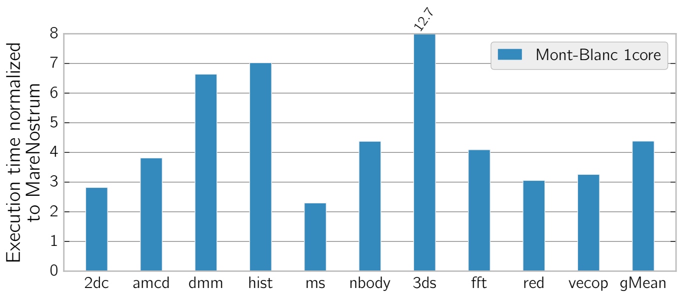


Fig. 6: Mont-Blanc benchmarks: core to core performance comparison.

Across the benchmark suite, Mont-Blanc is from 2.2 to 12.7 times slower. Particularly, Mont-Blanc core is outperformed in the workloads which mainly stress either floating-point subsystem (dmm), memory subsystem (hist), or both (3ds). On average, across the entire suite, a Mont-Blanc core is 4.3 times slower than a MareNostrum core.

3.2 Node evaluation

In the Figure 7 we present performance (7a) and energy consumption (7b) comparison on a node-to-node basis between the Mont-Blanc prototype and MareNostrum supercomputer.

Given the characteristics of the Mont-Blanc SoC, we employ three different computing scenarios: homogeneous computing with OpenMP (blue bars), heterogeneous with OpenCL (red bars), and heterogeneous with OpenCL + OmpSs [9](violet bars).

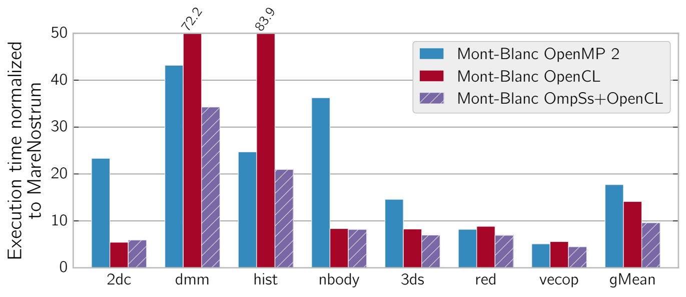
Both performance and power figures are normalized to executions on a MareNostrum node using 16 OpenMP threads.

Regarding the performance, a Mont-Blanc node is on average 18x slower than a dual-socket MareNostrum node when comparing performance achievable only with CPUs (two vs. sixteen cores). When utilizing only GPU, a Mont-Blanc node is on average 14x slower compared to a MareNostrum node. Finally, employing OmpSs to exercise both GPU and CPUs on a Mont-Blanc node, we reduce the gap and become 9x times slower on average across the workloads.

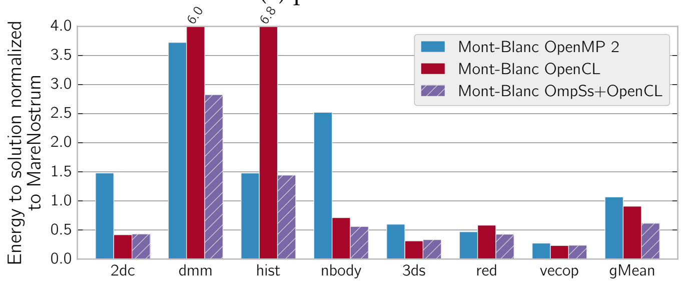
Energy wise, when using only CPUs as computational elements, a Mont-Blanc node consumes 5% more energy compared to a MareNostrum node. As we keep on improving the performance gap, Mont-Blanc node becomes more energy efficient on average: from consuming 20% less energy when using only GPU, to consuming 45% less energy when utilizing both GPU and CPU cores.

3.3 Node Power

The scheduling of computation to the Mont-Blanc node compute elements, namely CPU cores and/or GPU, has



(a) performance



(b) energy

Fig. 7: Mont-Blanc benchmarks: node to node a) performance and b) energy comparison.

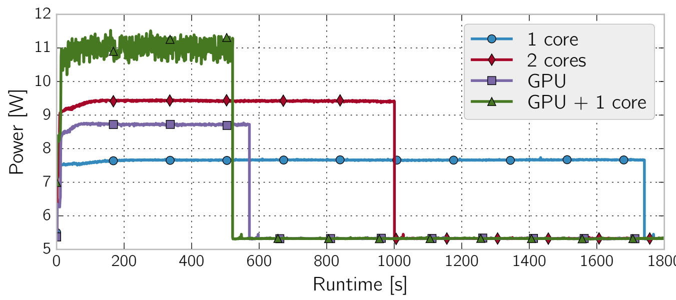


Fig. 8: Power profile of different compute to hardware mappings for 3D-stencil computation. Note: markers are only to distinguish lines, not sampling points.

a varying impact on energy efficiency, as shown in Figure 7b. The use of just the general purpose cores, just the embedded GPU or both simultaneously leads to different energy efficiency depending on the workload. Offloading computations to the GPU reduces energy consumption in most workloads, except for those not well suited for GPU, or lacking optimized GPU libraries.

Energy has two dimensions: power and time. Execution time depends on how well the application behaves on the given mappings to core and GPU. Power still depends on the application and is directly influenced by physical implementation and SoC power management.

The power monitoring infrastructure in the Mont-Blanc prototype (see Section 2.5) allows to analyze power consumption on different computation to hardware mappings. Comparing the power of different mappings, the user is able to estimate the speed-up required to compensate the power differences, and even achieve better energy efficiency.

Figure 8 shows power profile of one Mont-Blanc node for different mappings of the execution of 3D-stencil computation. The different mappings include a single CPU core, dual-core, GPU, and GPU+one CPU core.

The node idle power is 5.3W. This includes the static power of all components given that frequency scaling is disabled for benchmarking purposes. The average power consumption when running on one core and two CPU cores is 7.8W and 9.5W respectively. This includes the power consumption of the SoC, memory subsystem and network controller.

The power when using the GPU, and the GPU plus one cores is 8.8 and 11.5W, respectively. When running on the GPU, one of the cores is a helper thread that synchronously launches kernels to the GPU and therefore blocks until it completes. For this reason, the power of using the GPU plus cores may be lower than using the cores alone. When running on the GPU plus one core, one of the cores runs a worker thread and contributes to the computation, which explains the additional power. Although we cannot confirm due to the lack of finer grain power data, the results using one core and GPU plus one core indicate that the power consumption of the Mali-T604 GPU is lower than that of the Cortex-A15 CPU.

From our results with other workloads, we found that node power varies for different applications although it remains in the same range as shown in Figure 8 for most cases. The maximum power seen for executions with two CPU cores is 14W, and for executions with GPU plus one core 13.7W.

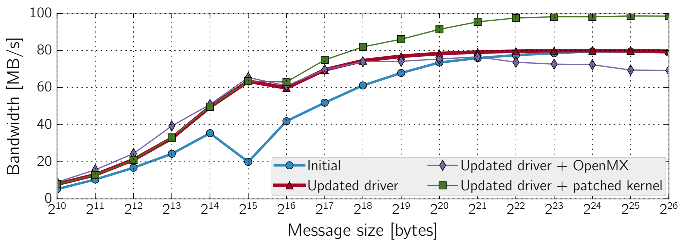
4 Interconnection Network Tuning and Evaluation

In this section, we quantify the latency and bandwidth of the Mont-Blanc node network interface. Since Mont-Blanc interconnect is implemented using a lossy Ethernet technology, it is of a paramount importance that every segment of interconnect is properly tuned. Thus, we discuss the improvements in different parts of interconnect stack which in turn affect the overall interconnect performance.

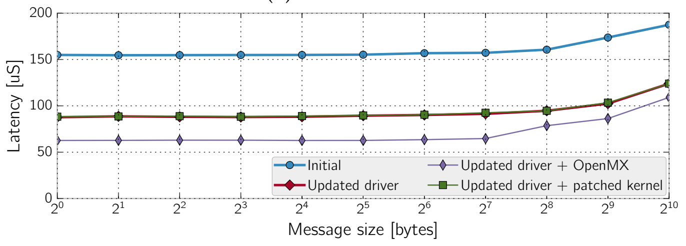
In Figure 9 we present both bandwidth (9a) and latency (9b) measurements obtained from the Mont-Blanc prototype using Intel MPI PingPong benchmark [10]. We present four curves per graph, each corresponding to incremental improvements on the side of node network interface.

After the initial deployment of the Mont-Blanc prototype we measured achievable MPI throughput and latency of 80MB/s and 156μs respectively (see blue line in the graph). The results were obtained using the NIC driver built-into the Linux Kernel.

After the update of the driver, using the proprietary one, we noticed a significant improvement in both throughput and latency for small messages - up to 3.4$\times$ better throughput for messages smaller than 64KB and reduced base MPI latency from 156 to 88μs for the zero-sized messages (see the red line in the graph). However, throughput for larger messages stayed the same as in initial configuration, $\approx$80MB/s. The main improvement comes from the feature that proprietary driver brought - configurable wait interval between the consecutive bulk transfers on the USB bus, which we reduced to the bare minimum.



(a) Bandwitdh



(b) Latency

Fig. 9: Inter-node bandwidth and latency of the Mont-Blanc prototype.

TABLE 2: MPI applications used for scalability evaluation.

Application	Domain
BigDFT [14], [15]	Electronic Structure
BQCD [16]	Quantum Chromodynamics
MP2C [17]	Multi-Particle Collision Dynamics
QuantumESPRESSO [18]	Electronic Structure and Materials Modeling
SMMP [19], [20], [21]	Molecular Thermodynamics
Alya [22], [23]	Biomedical Mechanics
COMD [24]	Proxy for Molecular Dynamics
LULESH [25], [26]	Proxy for Hydrodynamics
miniFE [27]	Proxy for Finite Element Method

Additionally, we did a back-port of a Linux Kernel patch [11] which improves throughput in USBNET [12] driver for USB3.0 compliant devices. This patch additionally improved throughput for messages larger than 64KB, with a maximum throughput of ≈100MB/s (see the green line in the graph).

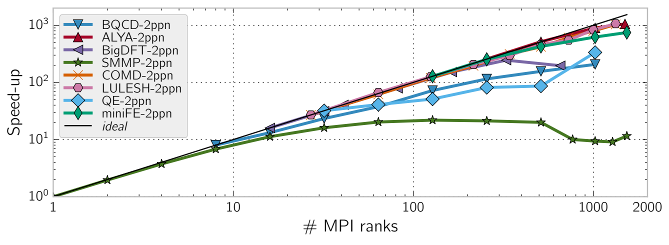
It is worth mentioning that deploying Open-MX [13], avoiding TCP/IP stack, we could further lower the base latency to 65μs which in turn would increase the throughput for small messages below 32K. Currently, this approach does not seem to show benefit for messages larger than 32K, even degrades the maximum throughput (see the violet line in the graph).

5 Overall System Evaluation

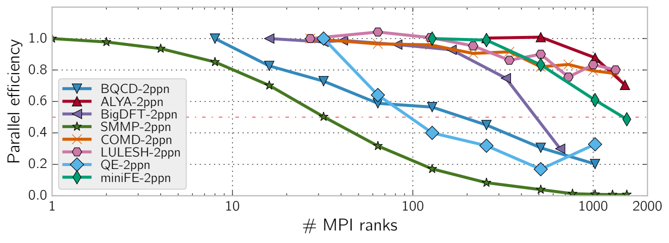
In this Section we show evaluation of parallel MPI applications listed in Table 2. These are flagship HPC production application from different domains, plus three reference mini-apps used by top US national labs. We look into weak and strong scaling figures, and further expand them with parallel efficiency graphs. Later, we compare Mont-Blanc against MareNostrum supercomputer on a subset of the applications to make a direct comparing between the two, in terms of performance and energy consumption.

5.1 Applications scalability

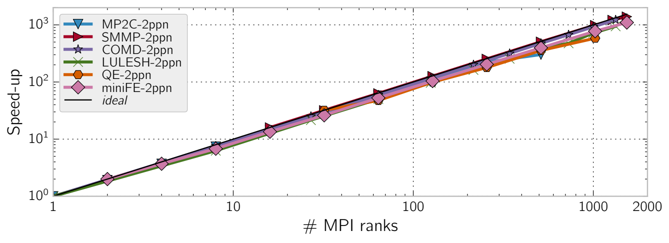
In Section 3.2 we already showed that a current Mont-Blanc node is, when using only CPU cores, 14× slower



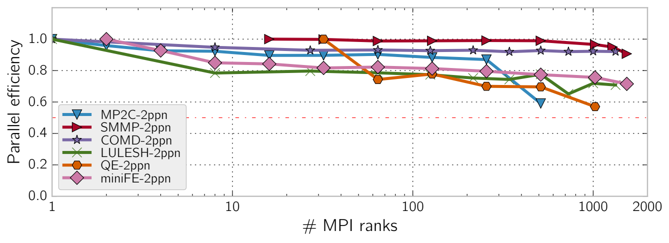
(a) Strong scaling.



(b) Strong efficiency.



(c) Weak scaling.



(d) Weak efficiency.

Fig. 10: Scalability and parallel efficiency of MPI application on the Mont-Blanc prototype.

on average compared to a MareNostrum's one. This means we should be able to scale out a workload to 14× more compute resources in order to compensate for performance disadvantage. In Figure 10 we present both *strong* and *weak* scaling figures of MPI applications when running on the Mont-Blanc prototype. Each graph is accompanied with corresponding parallel efficiency graph to provide more details about an application scalability.

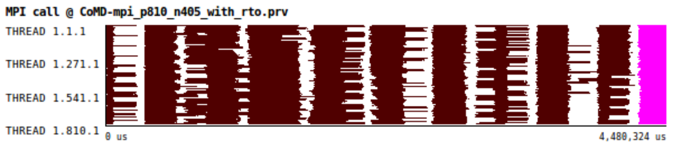
As it can be seen from Figure 10a strong scalability is bad, with only a few applications whose parallel efficiency (see Figure 10b) does not go below 0.5 for their biggest run. Weak scaling executions, revealed in Figure 10c, show better behavior. Parallel efficiency of weak scaling tests (see Figure 10d) reveals that the majority of the applications achieves parallel efficiency of 0.7 with two outliers going below 0.6.

We looked into the reasons for poor scalability. Apart from the narrow interconnect bandwidth, we discovered the following factors affecting the scalability of the prototype:

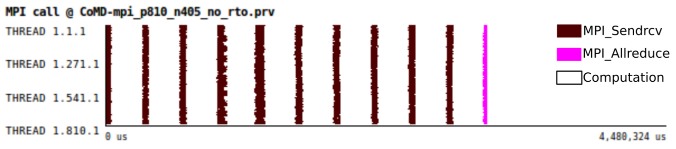
lost packets in the interconnect, each incurring a minimum penalty of one retransmission time out (RTO_min); and preemptions.

5.1.1 Lost packets

In Figure 11 we visualize execution traces of CoMD proxy application with and without lost packets. Trace of native execution which suffers lost packets is shown in Figure 11a. We simulate packet-lost-free execution by using DIMEMAS MPI simulator [28], and opt not to simulate TCP/IP layer and thus filter-out all the retransmissions (see Figure 11b).



(a) Packet loss in place.



(b) No packet loss.

Fig. 11: Illustration of lost packets effect on MPI parallel application: a) trace with, and b) without packet loss. X axis represents time, Y axis represents process number.

Both traces have the same scale, and results suggest that lost packets, and corresponding timeouts, hurt the performance 1.47×. This is of course application dependent, and depends on the communication patterns, message sizes, volume of communication, etc. In order to minimize the effects of retransmission, we lower RTO_min parameters from the default 200ms to the lowest possible - 5ms on our system.

5.1.2 Pre-emptions

In Figure 12 we present histogram of durations of computational phases during CoMD proxy application execution.

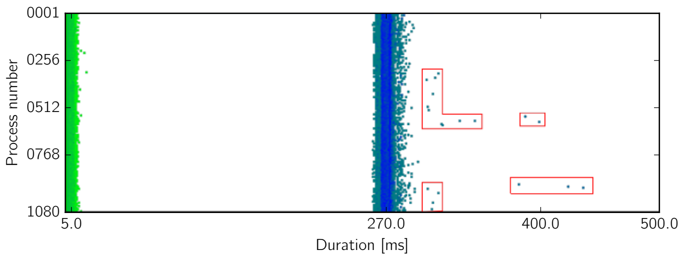


Fig. 12: 2D Histogram of computational phases duration. X axis represents bins of durations, Y axis represents process number. Gradient coloring: green-blue. Coloring function: logarithmic.

We identify two regions: with 5ms and ≈270ms durations. Regions taking 5ms are identified to be related to TCP/IP retransmissions with RTO_min = 5ms. However, most computational time is spent in regions with 270ms duration (note the blue color of those), which a duration of one inner-iteration of the application. Further, we can spot a set of outliers taking significantly more time, and we mark these with red polygons. Checking per computational phases IPC, we confirm this is not a load imbalance tied to application, but there are external factors introducing this variation. We suspect on preemptions, and from now on treat them as OS-noise in discussions to come.

TABLE 3: Mont-Blanc vs MareNostrum: same input, same execution time.

	Applications			
	CoMD		*miniFE*	
	MN[a]	MB[b]	MN	MB
# MPI ranks	64	240	64	224
Execution time [s]	70.72	68.05	71.66	72.19
Avg. power [W]	992.15	1083.25	1064.69	1033.72
Energy [Wh]	195	205	212	207
# rack units	8	7	8	7
# eff. rack units	8	6	8	6

[a] MareNostrum
[b] Mont-Blanc

5.2 Comparison with traditional HPC

In Figure 13 we depict performance and energy comparison between Mont-Blanc prototype and MareNostrum supercomputer, when using the same number of MPI ranks (cores) on both systems. For the same amount of used compute resources, Mont-Blanc prototype is 4× slower than MareNostrum when running MPI applications. In terms of energy consumption, Mont-Blanc prototype consumes 10% more on average across 7 applications, when using 257-1536 cores.

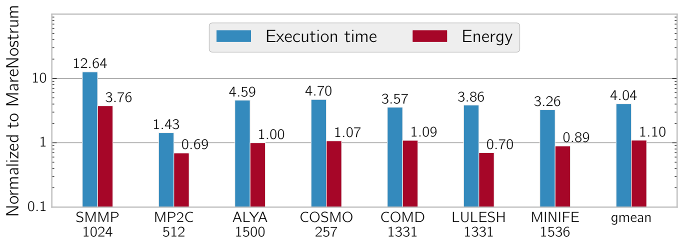


Fig. 13: Mont-Blanc vs MareNostrum: Performance and Energy comparison for a fixed number of MPI ranks.

Table 3 shows a comparison of Mont-Blanc prototype and MareNostrum when aiming to equalize their execution time. For this experiment we exercise strong-scaling capability of applications on the Mont-Blanc prototype, such that we keep input set constant and increase number of MPI ranks to get the same execution time on the MareNostrum. From the table we can observe that in order to get the same execution time, we would need between 3.5-3.75 more MPI ranks on the Mont-Blanc prototype across the applications we list in Table 3. In terms of energy consumption, both systems consume approximately the same amount of energy. Regarding the rack space, Mont-Blanc occupies one additional rack unit.

If we define utilization factor for the both machines, we can draw a conclusion such that rack space is not equally utilized: in the case of Mont-Blanc blade 90% of 7U is taken by the nodes. Furthermore, in the case of CoMD application, 240 MPI ranks occupy 88.8% of the available node resources (270 MPI ranks max). This leads to a utilization of $\sim$ 80% of 7U space. Finally, equivalent utilized rack space is 5.6$\approx$6U - which is two units less compared to MareNostrum.

To conclude, regarding system integration, Mont-Blanc prototype offers slightly denser integration, and consume about the same amount of energy. However, the peak power envelopes may vary, but it is not possible to track MareNostrum's power consumption in real time. We keep this matter open for future investigation.

6 Scalability Projection

The results in previous sections showed that the performance of the Mont-Blanc prototype is affected by interconnect technology, and potential load imbalance. These issues with the current implementation conceal the potential of the Mont-Blanc approach at scale. Further, in Section 5.2 we reveal a need for good parallel scalability in order to compensate for lower per node performance compared against MareNostrum supercomputer.

To unveil the scalability of the prototype architecture to larger systems, we employ a state-of-the-art methodology [29], [30] that allows us to project scalability of the current deployment. We look into the scalability of a certain workload through the *parallel efficiency* metric, which is a product of three fundamental factors:

$$\eta_{\|} = LB * Ser * Trf \tag{1}$$

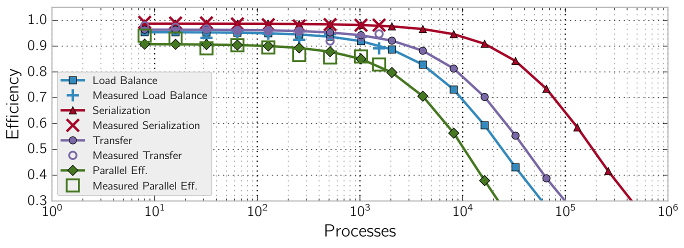
Each of the three factors (ranging from 0 to 1) reveal different aspect of an application running on a particular machine. *Load balance (LB)* defines a potential imbalance between the duration of computation phases across the running processes. Further, *Serialization (Ser)* gives an insight about potential inefficiency caused by the dependencies. Finally, *Transfer (Trf)* provides a measure of how much an actual run is affected by the interconnect.

In Figure 14 we present measured and extrapolated fundamental parallel efficiency factors from the traces obtained from the Mont-Blanc prototype. Let us define two parallel efficiency zones: from 1 to 0.8 and from 0.8 to 0.5, with the former being desirable, and the later representing low-efficiency but still acceptable until 0.5.

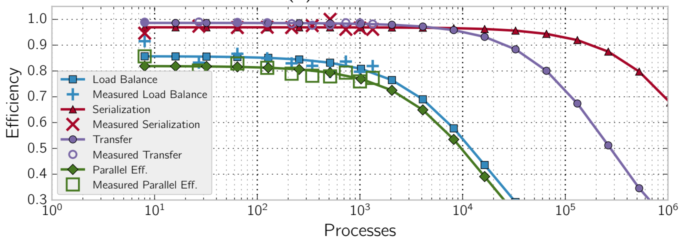
Looking into the figure, we observe none of the application would achieve efficiency in the desirable zone for the process count larger than 2.000. The most inefficient one, LULESH (see Figure 14b), enters the low-efficiency zone at the 256 processes count. Looking into the extracted fundamental factors, we see that all three applications are mostly limited by the load imbalance and transfer, while the former is the one affecting the performance the most.

In the Figure 15 we depict the improvement in parallel efficiency if we were able to completely remove retransmissions from the Mont-Blanc prototype.

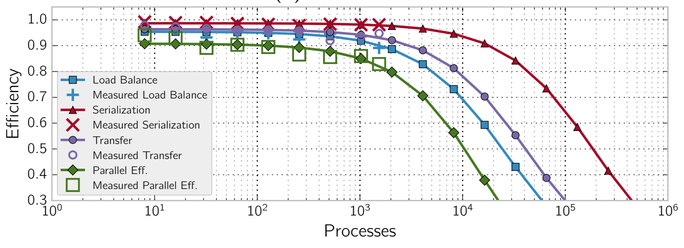
Both CoMD and LULESH benefit from removing the retransmissions related to lost-packets, such that they experience increase in the number of processes at which they reach 0.8 efficiency. The former improves from $\approx$2K to 9K, while the later from $\approx$256 to 0.5K. On the other side, miniFE does not seem to be affected with packet-loss issues (comparing Figure 14c and Figure 15c).



(a) CoMD.



(b) LULESH.



(c) miniFE.

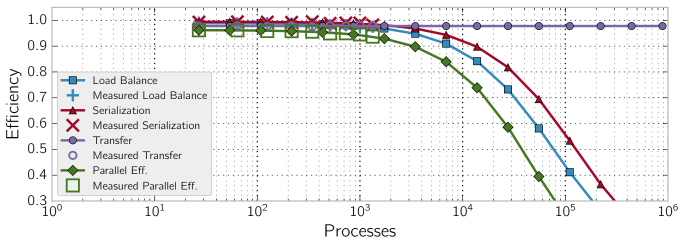
Fig. 14: Parallel efficiency - base case. Lines represent the model, while the lone marker ticks represent measurements points.

Further, in Figure 16 we depict the effect of removed preemptions and packet-loss altogether. From the graphs, only CoMD experience significant performance degradation due to preemptions, as it now reaches 0.8 parallel efficiency with 10K cores (see Figure 16a). It also shows slight improvement for 0.5 parallel efficiency, and now we consider to stop scaling at 40K cores with CoMD. Scalability of LULESH (see Figure 16b is still dominated by the load-balance inefficiencies and we achieve 0.8 efficiency with 256 cores and 0.5 with 1K. On the other side, miniFE (see Figure 16c) is affected by both load-balance and transfer inefficiencies with the root of the problem not being packet-loss. Removing preemptions slightly improves Serialization efficiency but it is the least contributing factor to the final parallel efficiency in this case (being close to 1.0 in a wide region). miniFE would scale up to 10K cores with the efficiency of 0.5 on the Mont-Blanc prototype.

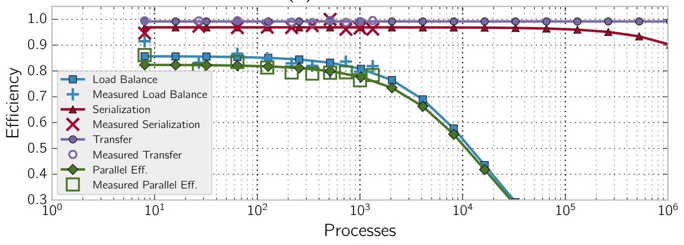
7 Related Work

ASCI Red was a landmark supercomputing machine deployed in 1997. It was the first supercomputer to break the TFLOPS barrier [31], and it remained at number one on the TOP500 list for three years [32]. It was the first top-tier supercomputer that, instead of powerful HPC-optimized vector processors, integrated 7,246 Pentium Pro processors,[1]

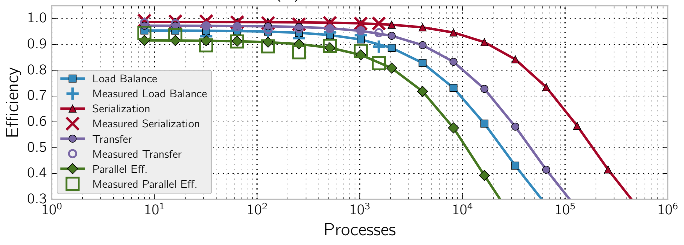
1. Later the number of processors was increased to 9,632 and upgraded to Pentium II processors



(a) CoMD.

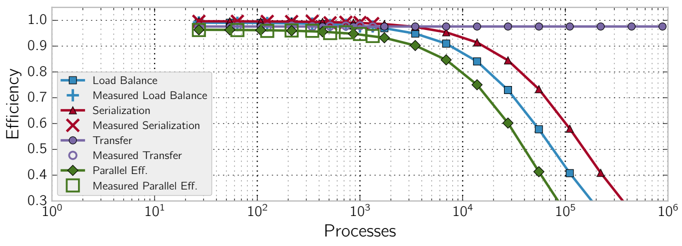


(b) LULESH.

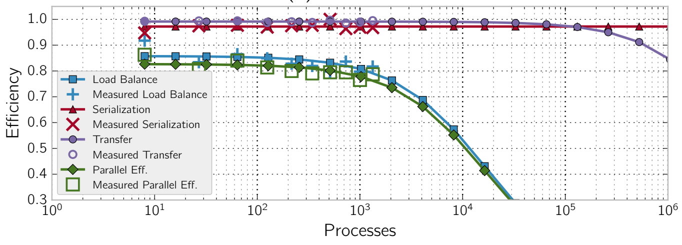


(c) miniFE.

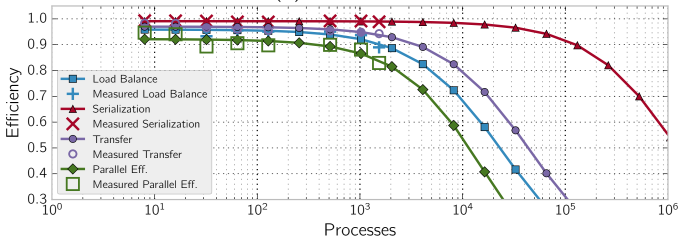
Fig. 15: Parallel efficiency - no lost packets. Lines represent the model, while the lone marker ticks represent measurements points.



(a) CoMD.



(b) LULESH.



(c) miniFE.

Fig. 16: Parallel efficiency - no lost packets, no pre-emptions. Lines represent the model, while the lone marker ticks represent measurements points.

the first x86 commodity processors to support on-chip double-precision floating-point computation. It opened the door to more designs based on the same principle: clusters of commodity PC processors; and the beginning of the end of vector processors.

GreenDestiny [33] represents an early attempt to use low-power processors in HPC, using the Transmeta TM5600 processor. MegaProto Systems [34] was another prototype in the same direction, based on later versions of Transmeta's processors, namely TM5800 and TM8820. Unfortunately, none of the systems entered the commercialization for HPC. However, some architectural principles found in the aforementioned processors, are now incorporated into NVIDIA Denver CPU architecture, meant for mobile and automotive market [35].

The three generations of BlueGene family of supercomputers [36], [37], [38], first introduced in 2004, introduced a new approach for energy-efficient HPC. The BlueGene processors were based on IBM cores used in the embedded market and extended with SIMD capabilities. System software and interconnect fabrics are however architecture-specific.

FAWN was a proposal to use Intel Atom processors in clusters [39]. The cluster was built and tested with a range of workloads, but evaluation did not include a suite of true HPC workloads. The authors did a thorough study to determine the type of workloads where Intel Atom can offer a competitive energy-efficiency compared to an Intel Core i7 processor. A follow-up of this work found that a homogeneous cluster of low-power Intel Atom processors is not suited to complex database workloads [40].

The Apple TV cluster [41] was the first cluster built from ARM-based consumer devices. Composed of four AppleTV devices for a total of four ARM Cortex-A8 cores, was the first platform allowing parallel processing on processors targeted to the mobile market. Authors pointed out the lack of fully pipelined floating-point unit which is nowadays unquestionable resource in mobile SoCs. Also, they report the latency of the network interface to be ≈200μs which is 2.27× bigger compared to the Mont-Blanc prototype.

The Tibidabo cluster prototype, deployed in 2011, was the first large scale HPC cluster based on mobile processors [42]. Built with a total of 256 developer kits (nodes) featuring dual-core Cortex-A9 processors, it was the first cluster with a full HPC software stack including cluster management, job scheduler, scientific libraries and HPC performance analysis tools for the ARM architecture. The ARM Cortex-A9 was the first mobile processor with built-in fully-pipelined double-precision floating-point unit. Tibidabo also demonstrated a case for scale-out parallel processing with real scientific applications on mobile processors.

Several commercial solutions targeting the server market have founded their design on mobile core IP. Calxeda EnergyCore ECX-1000 [43] was an ARM-based SoC with four ARM Cortex-A9 cores, five 10 GbE links, and SATA. AMD SeaMicro SM10000-64 was a 10U chassis with 256 dual-core

Intel Atom N570 processors. Quanta Computer S900-X31A is a 3U microserver of 24 nodes with Intel Atom S1200 "Centerton" at 10W per node. AMD Opteron A1100 is a micro-server SoC with eight ARM Cortex-A57 cores, two memory controllers and two 10GbE network controllers.

Other companies have developed custom processors based on the ARM architecture. Applied Micro (APM) X-Gene [44] is a server-class SoC with eight 64-bit ARMv8 cores and four 10 GbE links. Cavium, with large experience in networking processors, designed ThunderX [45], another server-class SoC with 48 ARMv8 cores and multiple 10/40GbE interfaces. Qualcomm and Phytium also announced ARMv8 server SoCs with 24 [46] and 64 [47] cores, respectively.

Some successful deployments of some of these SoCs are already in place. CERN has published a comparison of APM X-Gene compared to Intel Xeon and IBM Power8 chips [48]. PayPal has deployed HP Moonshot servers with APM X-Gene processors claiming half the price, one seventh of the power consumption and 10x more nodes per rack compared to their traditional data center infrastructure [49].

These efforts, however, target the server marker and there are still no large demonstrators of such mobile-technology-based processors for HPC. The Mont-Blanc prototype is thus the first demonstrator of an HPC cluster with full HPC software stack running real scientific application, commodity networking, and standard system integration. Our experiments demonstrate the feasibility of the proposed alternative approach, assess system software maturity and project its scalability at larger scale.

8 Conclusions

In this report, we have presented the architecture of the Mont-Blanc prototype, from both hardware and software perspectives. Comparing its performance against a production level supercomputer we have shown that it is $\approx 4\times$ slower then its contemporary supercomputer based on the Intel SandyBridge Xeon processor, when using the same number of MPI ranks (cores). Energy-efficiency wise, Mont-Blanc prototype is on pair. However, if we favor latency-oriented computing, aiming towards faster runtime over other metrics, we have to put $4\times$ more computing resources in order to compensate for lower per node performance.

Direct comparison of the cost of the Mont-Blanc prototype against a production level supercomputer is impossible. The price of the former is dictated by the prototyping process, while the cost of the later is always formed after negotiations and competition required by the procurements. However, we could speculate that using the same SoCs, as the one used in modern mobile, embedded and automotive devices, Mont-Blanc approach could offer more affordable High-Performance Computing machines. Though, given the current limitations of the aforementioned SoCs, it probably could not compete at scale in near future, unless vendors take another direction and start incorporating minimum required HPC features.

Prototyping a new HPC platform requires freezing design specification at a given moment of the design cycle. The Mont-Blanc prototype, presented in this report, is built on the technology dating from 2012. From the moment of freezing design specification there have been many advances in the mobile, embedded and automotive SoCs - chips integrating more cores, heterogeneous CPUs, shift towards 64-bit architecture, advances in memory technology, advances in process nodes etc. - while staying within the same power and thermal budget dictated by the target application of the SoCs. Here we leave open the question for future research - *how would an upgraded revision of the Mont-Blanc prototype perform?*

References

[1] K. T. Malladi, B. C. Lee, F. A. Nothaft, C. Kozyrakis, K. Periyathambi, and M. Horowitz, "Towards energy-proportional datacenter memory with mobile DRAM," in *ACM SIGARCH Computer Architecture News*, vol. 40, pp. 37–48, IEEE Computer Society, 2012.

[2] ASIX, "AX88179 - USB3.0 to 10/100/1000M Gigabit Ethernet Controller." http://www.asix.com.tw/FrootAttach/brief/AX88179_Brief.pdf, 2015.

[3] "Intelligent Platform Management Interface." http://en.wikipedia.org/wiki/Intelligent_Platform_Management_Interface, 2015.

[4] "The MQTT Protocol." http://mqtt.org, 2014.

[5] "The Apache Cassandra Project." http://cassandra.apache.org/, 2014.

[6] "ARM Connected Community Forums." https://web.archive.org/web/20160303143357/https://community.arm.com/message/18218, 4 2014.

[7] Barcelona Supercomputing Center, "MareNostrum III (2013) System Architecture." https://web.archive.org/web/20160303114630/https://www.bsc.es/marenostrum-support-services/mn3.

[8] N. Rajovic, A. Rico, J. Vipond, I. Gelado, N. Puzovic, and A. Ramirez, "Experiences with mobile processors for energy efficient HPC," in *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 464–468, EDA Consortium, 2013.

[9] A. Duran, E. Ayguadé, R. M. Badia, J. Labarta, L. Martinell, X. Martorell, and J. Planas, "Ompss: a proposal for programming heterogeneous multi-core architectures," *Parallel Processing Letters*, vol. 21, no. 02, pp. 173–193, 2011.

[10] Intel, "MPI Benchmarks v4.1." https://software.intel.com/en-us/articles/intel-mpi-benchmarks.

[11] Ming Lei, "USBNET: increase max rx/tx qlen for improving USB3 throuput." https://web.archive.org/web/20160308154732/https://github.com/torvalds/linux/commit/452c447a497dce3c9faeb9ac7f2e1ff39232876b, 2013.

[12] David Brownell, "The GNU/Linux "usbnet" Driver Framework." https://web.archive.org/web/20160304233120/http://www.linux-usb.org/usbnet/, 2005.

[13] B. Goglin, "Design and implementation of Open-MX: High-performance message passing over generic Ethernet hardware," in *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, pp. 1–7, IEEE, 2008.

[14] "The BigDFT Scientific Application." http://bigdft.org/, 2015.

[15] L. Genovese, B. Videau, M. Ospici, T. Deutsch, S. Goedecker, and J.-F. Méhaut, "Daubechies Wavelets for High Performance Electronic Structure Calculations: the BigDFT Project.," in *Compte-Rendu de l'Académie des Sciences, Calcul Intensif.*, Académie des Sciences, 2010.

[16] Y. Nakamura and H. Stüben, "BQCD-Berlin quantum chromodynamics program," *arXiv preprint arXiv:1011.0199*, 2010.

[17] G. Sutmann, L. Westphal, and M. Bolten, "Particle based simulations of complex systems with mp2c : Hydrodynamics and electrostatics," *AIP Conference Proceedings*, vol. 1281, no. 1, pp. 1768–1772, 2010.

[18] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, A. D. Corso, S. de Gironcoli, S. Fabris, G. Fratesi, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A. P. Seitsonen, A. Smogunov, P. Umari, and R. M. Wentzcovitch, "QUANTUM

ESPRESSO: a modular and open-source software project for quantum simulations of materials," *Journal of Physics: Condensed Matter*, vol. 21, no. 39, p. 395502, 2009.

[19] F. Eisenmenger, U. H. E. Hansmann, S. Hayryan, and C.-K. Hu, "[SMMP] A modern package for simulation of proteins," *Computer Physics Communications*, vol. 138, no. 2, pp. 192–212, 2001.

[20] F. Eisenmenger, U. H. E. Hansmann, S. Hayryan, and C.-K. Hu, "An enhanced version of SMMP—open-source software package for simulation of proteins," *Computer Physics Communications*, vol. 174, no. 5, pp. 422–429, 2006.

[21] J. H. Meinke, S. Mohanty, F. Eisenmenger, and U. H. E. Hansmann, "[SMMP] v. 3.0—Simulating proteins and protein interactions in Python and Fortran," *Computer Physics Communications*, vol. 178, no. 6, pp. 459–470, 2008.

[22] M. Vazquez, G. Houzeaux, S. Koric, A. Artigues, J. Aguado-Sierra, R. Aris, D. Mira, H. Calmet, F. Cucchietti, H. Owen, *et al.*, "Alya: towards exascale for engineering simulation codes," *arXiv preprint arXiv:1404.4881*, 2014.

[23] M. Vázquez, R. Arís, J. Aguado-Sierra, G. Houzeaux, A. Santiago, M. López, P. Córdoba, M. Rivero, and J. C. Cajas, *Selected Topics of Computational and Experimental Fluid Mechanics*, ch. Alya Red CCM: HPC-Based Cardiac Computational Modelling, pp. 189–207. Cham: Springer International Publishing, 2015.

[24] ExMatEx, "Comd proxy application."

[25] I. Karlin, J. Keasler, and R. Neely, "Lulesh 2.0 updates and changes," Tech. Rep. LLNL-TR-641973, August 2013.

[26] I. Karlin, A. Bhatele, J. Keasler, B. L. Chamberlain, J. Cohen, Z. DeVito, R. Haque, D. Laney, E. Luke, F. Wang, D. Richards, M. Schulz, and C. Still, "Exploring traditional and emerging parallel programming models using a proxy application," in *27th IEEE International Parallel & Distributed Processing Symposium (IEEE IPDPS 2013)*, (Boston, USA), May 2013.

[27] M. A. Heroux, D. W. Doerfler, P. S. Crozier, J. M. Willenbring, H. C. Edwards, A. Williams, M. Rajan, E. R. Keiter, H. K. Thornquist, and R. W. Numrich, "Improving performance via mini-applications," *Sandia National Laboratories, Tech. Rep. SAND2009-5574*, vol. 3, 2009.

[28] R. M. Badia, J. Labarta, J. Gimenez, and F. Escale, "Dimemas: Predicting mpi applications behavior in grid environments," in *Workshop on Grid Applications and Programming Tools (GGF8)*, vol. 86, pp. 52–62, 2003.

[29] M. Casas, R. M. Badia, and J. Labarta, "Automatic analysis of speedup of MPI applications," in *Proceedings of the 22nd Annual International Conference on Supercomputing, ICS 2008*, pp. 349–358, 2008.

[30] C. Rosas, J. Giménez, and J. Labarta, "Scalability prediction for fundamental performance factors," *Supercomputing frontiers and innovations*, vol. 1, no. 2, 2014.

[31] T. Mattson and G. Henry, "An Overview of the Intel TFLOPS Supercomputer," *Intel Technology Journal*, vol. 2, no. 1, 1998.

[32] TOP500, "Top500®supercomputer cites." http://www.top500.org/.

[33] M. Warren, E. Weigle, and W. Feng, "High-density computing: A 240-processor Beowulf in one cubic meter," in *Supercomputing, ACM/IEEE 2002 Conference*, pp. 61–61, IEEE, 2002.

[34] H. Nakashima, H. Nakamura, M. Sato, T. Boku, S. Matsuoka, D. Takahashi, and Y. Hotta, "Megaproto: 1 TFlops/10kW rack is feasible even with only commodity technology," in *Proceedings of the ACM/IEEE SC 2005 Conference on Supercomputing*, IEEE, 2005.

[35] D. Boggs, G. Brown, B. Rozas, N. Tuck, and K. S. Venkatraman, "NVIDIA's Denver Processor," in *Hot Chips 2014*.

[36] N. R. Adiga, G. Almási, G. S. Almasi, Y. Aridor, R. Barik, D. Beece, R. Bellofatto, G. Bhanot, *et al.*, "An overview of the BlueGene/L supercomputer," in *ACM/IEEE 2002 Conference on Supercomputing*, IEEE Computer Society, 2002.

[37] S. Alam, R. Barrett, M. Bast, M. R. Fahey, J. Kuehn, C. McCurdy, J. Rogers, P. Roth, R. Sankaran, J. S. Vetter, P. Worley, and W. Yu, "Early evaluation of IBM BlueGene/P," in *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, (Piscataway, NJ, USA), pp. 23:1–23:12, IEEE Press, 2008.

[38] IBM Systems and Technology, "IBM System Blue Gene/Q Data Sheet," November 2011.

[39] V. Vasudevan, D. Andersen, M. Kaminsky, L. Tan, J. Franklin, and I. Moraru, "Energy-efficient cluster computing with fawn: Workloads and implications," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, pp. 195–204, ACM, 2010.

[40] W. Lang, J. Patel, and S. Shankar, "Wimpy node clusters: What about non-wimpy workloads?," in *Proceedings of the Sixth International Workshop on Data Management on New Hardware*, pp. 47–55, ACM, 2010.

[41] K. Fürlinger, C. Klausecker, and D. Kranzlmüller, "Towards energy efficient parallel computing on consumer electronic devices," in *Information and Communication on Technology for the Fight against Global Warming*, pp. 1–9, Springer, 2011.

[42] N. Rajovic, A. Rico, N. Puzovic, C. Adeniyi-Jones, and A. Ramirez, "Tibidabo: Making the case for an ARM-based HPC system," *Future Generation Computer Systems*, vol. 36, pp. 322–334, 2014.

[43] Calxeda, "Calxeda EnergyCore ECX-1000 Series." http://www.calxeda.com/wp-content/uploads/2012/06/ECX1000-Product-Brief-612.pdf, 2012.

[44] Applied Micro, "APM "X-Gene" Launch Press Briefing." https://web.archive.org/web/20120813151248/http://www.apm.com/global/x-gene/docs/X-GeneOverview.pdf, 2012.

[45] Cavium, "ThunderX$^{\text{TM}}$." https://web.archive.org/web/20160310114848/http://www.cavium.com/pdfFiles/ThunderX_PB_p12_Rev1.pdf, 2013.

[46] PCWorld, "Qualcomm enters server CPU market with 24-core ARM chip." http://www.pcworld.com/article/2990868/qualcomm-enters-server-cpu-market-with-24-core-arm-chip.html, 2015.

[47] Charles Zhang, Phytium Technology Co., Ltd, "Mars: A 64-core ARMv8 Processor." https://web.archive.org/web/20160310155325/http://www.hotchips.org/wp-content/uploads/hc_archives/hc27/HC27.24-Monday-Epub/HC27.24.30-HP-Cloud-Comm-Epub/HC27.24.321-64core-Zhang-phytium-v1.0.pdf, 2015.

[48] D. Abdurachmanov, B. Bockelman, P. Elmer, G. Eulisse, R. Knight, and S. Muzaffar, "Heterogeneous high throughput scientific computing with apm x-gene and intel xeon phi," in *Journal of Physics: Conference Series*, vol. 608, p. 012033, IOP Publishing, 2015.

[49] Data Center Knowledge, "PayPal Deploys ARM Servers in Data Centers." https://web.archive.org/web/20160310160416/http://www.datacenterknowledge.com/archives/2015/04/29/paypal-deploys-arm-servers-in-data-centers/, 2015.