Research on Computer Architecture based on Distributed Cluster Structure

Yuwen Zheng

Shandong Women's University, Jinan, 250300, China

Keywords: computer architecture, distributed cluster structure, high performance computing (HPC), CPU and GPU.

Abstract. With the fast development of image science and embedded system technique, it is widely accepted that future HPC systems will be limited by their power consumption. The current high performance computing system is a commodity server processors, design for many years to achieve maximum performance, and then it dawned on energy efficiency. In this paper we advocate a different approach: computer architecture based on distributed cluster structure. We introduce the architecture of Tibidabo, the first large-scale HPC cluster built from ARM multicore chips, and a detailed performance and energy efficiency evaluation. We now design experience and to improve the energy efficiency of future HPC systems based on the low power core. The experimental result shows the effectiveness of our methodology, further in-depth research portion is also discussed with case studies.

1.Introduction

In High Performance Computing (HPC), there is a continued need for higher computational performance. Science major challenges such as engineering, geophysics, bioinformatics, and other types of compute-intensive applications need a large amount of computing power. On the other hand, energy is becoming one of the most expensive resources, which greatly helps to run a large total cost of super computer facilities. In some cases, the total energy cost over a few years of operation can exceed the cost of the hardware infrastructure acquisition [1]. This trend is not limited to HPC systems, can also be applied to the data centres. Energy efficiency is already a primary concern for the design of any computer system and it is unanimously recognized that reaching the next milestone in supercomputers' performance will be strongly constrained by power. The energy efficiency of a system will define the maximum achievable performance. In this article, we take the first steps in the solution by low power high performance computing system using the embedded and mobile devices. Use the CPU from the domain, however, is a challenge. Most embedded CPUs lack a vector floating-point unit and their software ecosystem is not tuned for HPC. What makes them particularly interesting is the size and power characteristics which allow for higher packaging density and lower cost. In the following three subsections we further motivate our proposal from several important aspects.

1.1 The Road to Exascale and ARM Processor

To illustrate our point about the need for low-power processors, let us reverse engineer a theoretical Exaflop supercomputer that has a power budget of 20 MW. An Exaflop machine will require 62.5 million of such cores, independently of how they are packaged together (multicore density, sockets per node). We also assume that only 30-40% of the total power will be actually spent on the cores, the rest going to power supply overhead, interconnect, storage, and memory. That leads to a power budget of 6 MW to 8 MW for 62.5 million cores, which is 0.10 W to 0.13 W per core. Current high performance processors integrating this type of cores require tens of watts at 2 GHz. However, ARM processors, designed for the embedded mobile market, consume less than 0.9 W at that frequency and thus are worth exploring even though they do not yet provide sufficient level of performance and they have a promising roadmap ahead.

There is already a significant trend towards using ARM processors in data servers and cloud computing environments [2]. Those workloads are constrained by I/O and memory subsystems, not

by CPU performance. Recently, ARM processors are also taking significant steps towards increased double-precision floating-point performance, making them competitive with state-of-the-art server performance. Previous generations of arm application processor has no function

Floating-point unit can support HPC1 required throughput and delay. The ARM architecture a9 has an optional VFPv3 floating-point unit [2] and/or neon single instruction multiple data (SIMD) floating point unit [3]. VFPv3 unit is the assembly line, each cycle is able to perform a double add operation, or a the MUL (fused multiply accumulation) every two cycles. The neon unit is the SIMD units support only integer and single precision point operand to the HPC itself less attractive. Then, use a double floating point arithmetic instructions per cycle (VFPv3), 1 GHz architecture provide 1-A9 GFLOPS peak. Recently arm architecture (A15 [4] processor has a completely pipelining double-precision floating-point unit and provide 2 GFLOPS 1 GHz per cycle (FMA). The new ARMv8 instruction set, which is being implemented in next-generation ARM cores, namely the Cortex-A50 Series [5], features a 64-bit address space, and adds double-precision to the NEON SIMD ISA, allowing for 4 operations per cycle per unit leading to 4 GFLOPS at 1 GHz.

1.2 The Bell's Law and Contribution

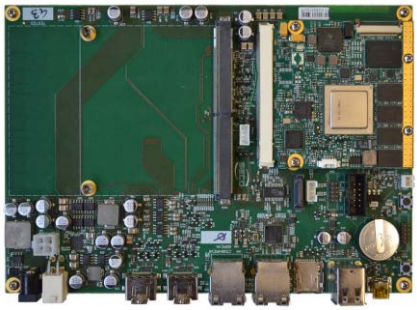
Our approach for an HPC system is novel because we argue for the use of mobile cores. We consider the improvements expected in mobile SoCs in the near future that would make them real candidates for HPC. As Bell's states [6], a new computer class is usually based on lower cost components, which continue to evolve at a roughly constant price but with increasing performance from Moore's law. This trend holds today: the class of computing systems on the rise today in HPC is those systems with large numbers of closely-coupled small cores (BlueGene/Q and Xeon Phi systems). From an architectural point of view, we suggest that the in this computing the size of the performance of the class and its growth potential and the evolution of the mobile market. In this paper, we present Tibidabo, an experimental HPC cluster that we built using NVIDIA Tegra2 chips, each featuring a performance-optimized dual-core ARM Cortex-A9 processor. We use the PCIe support in Tegra2 to connect a 1 GbE NIC, and build a tree interconnect with 48-port 1 GbE switches. We do not intend our first prototype to achieve energy efficiency competitive with today's leaders. The purpose of this prototype is to be a proof of concept to demonstrate that building such energy-efficient clusters with mobile processors is possible, and to learn from the experience. On the software side, the goal is to deploy an HPC-ready software stack for ARM-based systems, and to serve as an early application development and tuning vehicle. The contributions of this paper are: (1) Have design of the first HPC ARM-based cluster architecture, with a complete performance evaluation, energy efficiency evaluation, and comparison with state-of-the-art high-performance architectures. (2) A power distribution estimation of our ARM cluster. (3) Model-based performance and energy-efficiency projections of a theoretical HPC cluster with a higher multicore density and higher-performance ARM cores. (4) Technology challenges and design guidelines based on our experience to make ARM-based clusters a competitive alternative for HPC.

2. The ARM Cluster Architecture Analysis

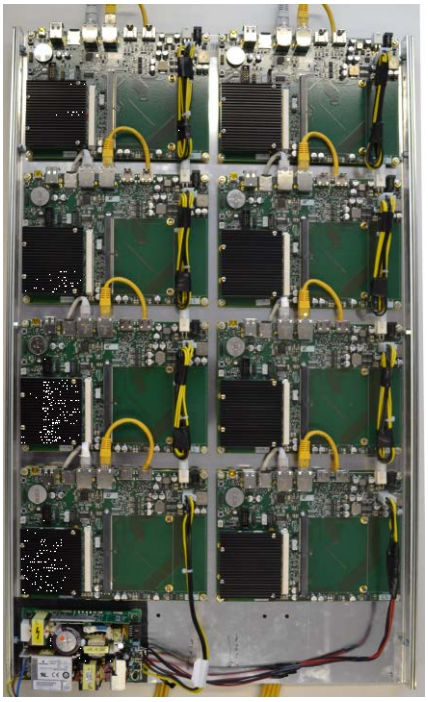
The computing chip in the Tibidabo cluster is the Nvidia Tegra2 SoC, with a dual-core ARM Cortex-A9 running at 1 GHz and implemented using TSMC's 40nm LPG performance-optimized process. Tegra2 features a number of application-specific accelerators targeted at the mobile market, such as video and audio encoder/decoder, and image signal processor, but none of these can be used for general-purpose computation and only contribute as a SoC area overhead. The GPU in Tegra2 does not support general programming models such as CUDA or OpenCL, so it cannot be used for HPC computation either. However, more advanced GPUs actually support these programming models and a variety of HPC systems use them to accelerate certain kind of workloads. Tegra2 is the central part of the Q7 module [7] (See Figure 1(a)). The module also integrates 1 GB of DDR2-667 memory, 16 GB of eMMC storage, a 100 MbE NIC (connected to Tegra2 through USB) and exposes PCIe connectivity to the carrier board. Using Q7 modules allows an easy upgrade when next generation SoCs become available, and reduces the cost of replacement in case of failure. These boards are organized into blades (See Figure 1(c)), and each blade hosts 8 nodes and a shared

Power Supply Unit (PSU). In total, Tibidabo has 128 nodes and it occupies 42 U standard rack spaces: 32 U for compute blades, 4 U for interconnect switches and 2 U for the file server. These are the basic structure of the proposed system. At the time of writing of this paper this was the only MPI distribution that worked reliably with the SLURM job-manager in our cluster. We use ATLAS 3.9.51 [8] as our linear algebra library. This library is chosen due to the lack of a hand-optimized algebra library for our platform and its ability to auto-tune to the underlying architecture. Applications that need an FFT library rely on FFTW v3.3.1 [9] for the same reasons.

Q7 Module (a)



Carrier Board (b)



Blade with 8 Boards (c)

Tibidabo Rack (d)

Fig.1: Components of the Tibidabo System

3. The Evaluation and Validation

3.1The Methodology

For single-node energy efficiency, we have measured a single Q7 board and compared the results against a power-optimized Intel Core i7 [10] laptop (Table 1), whose processor chip has a thermal design power of 35 W. Due to the different natures of the laptop and the development board, and in order to give a fair comparison in terms of energy efficiency, we are measuring only the power of components that are necessary for executing the benchmarks, so all unused devices are disabled. On our Q7 board, we disable Ethernet during the benchmarks execution. On the Intel Core i7 platform, graphic output, sound card, touch-pad, blue-tooth, WiFi, and all USB devices are disabled, and the corresponding modules are unloaded from the kernel. The hard disk is spun down, and the Ethernet is disabled during the execution of the benchmarks. Multithreading could not be disabled, but all experiments are single-threaded and we set their logical core affinity in all cases. On both platforms benchmarks are compiled with -O3 level of optimization using GCC 4.6.2 compiler.

Table 1: Experimental Platforms

No.	Arm Platform	Intel Platform
SoC	Tegra2	Intel Core i7
Architecture	Cortex-A9	Nehalem
Core	Dual core	Dual core
Frequency	1 GHz	2.8 GHz
Cache Size	L1:32 KB	L1: 32KB
RAM	1 GB	8 GB
Compiler	GCC 4.6.2	GCC 4.6.2

3.2The Single Node Performance

We start with the evaluation of the performance and energy efficiency of a single node in our cluster, in order to have a meaningful comparison to other state-of-the-art compute node architectures. In Figure 2 we evaluate the performance of Cortex-A9 floating-point double-precision pipeline using in-house developed micro benchmarks. These benchmarks perform dense double-precision floating-point computation with accumulation on arrays of a given size (input parameter) stressing the FPADD and FPMA instructions in a loop. We exploit data reuse by executing the same instruction multiple times on the same elements within one loop iteration. This way we reduce loop condition testing overheads and keep the floating point pipeline as utilized as possible. The purpose is to evaluate if the ARM Cortex-A9 pipeline is capable of achieving the peak performance of 1 FLOP per cycle. Our results show that the Cortex-A9 core achieves the theoretical peak double-precision floating-point performance when the micro benchmark working set in the L1 cache (32 KB).

We also evaluate the effective memory bandwidth using the STREAM benchmark [10]. In this case, the memory bandwidth comparison is not just a core architecture comparison because bandwidth depends mainly on the memory subsystem. However, bandwidth efficiency, which shows the achieved bandwidth out of the theoretical peak, shows to what extent the core, cache hierarchy and on-chip memory controller are able to exploit chip memory bandwidth. We use the largest working set size that in the system. While it is true that the ARM Cortex-A9 platform takes much less power than the Core i7, it also requires a longer runtime, which results in a similar energy consumption the Cortex-A9 platform is between 5% and 18% better. Given that the Core i7 platform is faster, that makes it superior in other metrics such as Energy-Delay. Our single-node performance evaluation shows that the Cortex-A9 is 9 times slower than the Core i7 at their maximum operating frequencies, which means that we need our applications to exploit a minimum of 9 parallel processors in order to achieve competitive time-to-solution. More processing cores in the system mean more need for scalability. In this section we evaluate the performance, energy efficiency and scalability of the whole Tibidabo cluster.

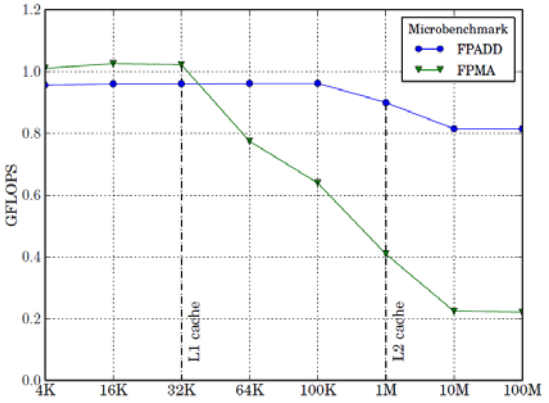


Fig.2: Performance of Double-Precision

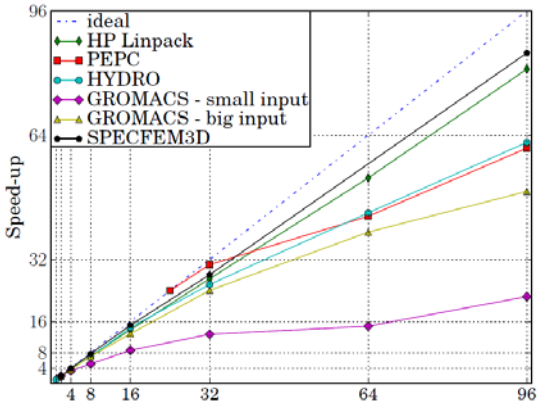


Fig.3: Scalability of HPC Applications

3.3The Cluster Energy Efficiency

For both Cortex-A9 and Cortex-A15, the CPU macro power includes the L1 caches, cache coherence unit and L2 controller [11]. Therefore, the increase in power due to a more complex L2 controller and cache coherence unit for a larger multicore are accounted when that power is factored by the number of cores. The memory power is overestimated, so the increased power due to the increased complexity of the memory controller to scale to a higher number of cores is also accounted for the same reason. Furthermore, a Cortex-A9 system cannot address more than 4 GB of memory so, strictly speaking, Cortex-A9 systems with more than 4 GB are not realistic. The remaining power in the compute node is considered to be overhead, and does not change with the number of cores. The board overhead is part of the power of a single node, to which we add the power of the cores, L2 cache and memory. However, we include configurations for higher core counts per chip to show what would be the performance and energy efficiency if Cortex-A9 included large physical address extensions as the Cortex-A15 does to address up to 1 TB of memory. The power model is summarized in these equations:

$$P_{pred} = \frac{n_{tc}}{n_{cpc}} \times \left(\frac{P_{over}}{n_{nin}} + P_{eth} + n_{cpc} \times \left(P_{A9G} + \frac{P_{L2S}}{2} \right) \right)$$

$$P_{over} = P_{tot} - n_{nin} \times \left(P_{mem} + 2P_{A9G} + P_{L2S} + P_{eth} \right)$$

There are still no enclosures announced, and no benchmark reports, but we expect a better performance than ARMv7-based enclosures, due to an improved CPU core architecture and three levels of cache hierarchy. The Calxeda ECX-1000 SoC is built for server workloads: it is a quad-core chip with Cortex-A9 cores running at 1.4 GHz, 4 MB of L2 cache with ECC protection, a 72-bit memory controller with ECC support, five 10 Gb lanes for connecting with other SoCs, support for 1 GbE and 10 GbE, and SATA 2.0 controllers with support for up to five SATA disks. Unlike ARM-based mobile SoCs, ECX-1000 does not have a power overhead in terms of unnecessary on-chip resources and, thus, it seems better suited for energy-efficient HPC. However, to the best of our knowledge, there are neither reported numbers for energy-efficiency of HPL running in a cluster environment (only single node executions) nor scientific applications scalability tests for any of the aforementioned enclosures.

4. Conclusion and Summary

In this paper we presented Tibidabo, the world's first ARM-based HPC cluster, for which we set up an HPC-ready software stack to execute HPC applications widely used in scientific research such as SPECFEM3D and GROMACS. Tibidabo was built using commodity components that are not designed for HPC. Nevertheless, our prototype cluster achieves 120 MFLOPS/W on HPL, competitive with AMD Operton 6128 and Intel Xeon X5660-based systems. We identified a set of inefficiencies of our design given the components target mobile computing. The main inefficiency is that the power taken by the components required to integrate small low-power dual-core processors sets the high energy efficiency of the cores themselves. We perform a set of simulations to project the energy efficiency of our cluster if we could have used chips featuring higher-performance ARM cores and integrating a larger number of them together. Based on these projections, a cluster configuration with 16-core Cortex-A15 chips would be competitive with Sandy Bridge-based homogeneous systems and GPU-accelerated heterogeneous systems in the Green500 list. These encouraging industrial roadmaps, together with research initiatives such as the EU-funded Mont-Blanc project, may lead ARM-based platforms to accomplish the recommendations given in this paper in a near future. In the future, more in-depth research will be conducted and simulated.

Acknowledgements

The research work was supported by Shandong Provincial Staff Education office No. 2013-324.

References

[1] Forshaw, Matthew, A. Stephen McGough, and Nigel Thomas. "Energy-efficient Checkpointing in High-throughput Cycle-stealing Distributed Systems." Electronic Notes in Theoretical Computer Science 310 (2015).

[2] Mulfari, Davide, Antonio Celesti, and Massimo Villari. "A computer system architecture providing a user-friendly man machine interface for accessing assistive technology in cloud computing." Journal of Systems and Software 100 (2015): 129-138.

[3] Shukla, Surendra Kumar, C. N. S. Murthy, and P. K. Chande. "Parameter Trade-off And Performance Analysis of Multi-core Architecture." Progress in Systems Engineering. Springer International Publishing, 2015. 403-409.

[4] Amin, Muhammad Bilal, et al. "Profiling-Based Energy-Aware Recommendation System for Cloud Platforms." Computer Science and its Applications. Springer Berlin Heidelberg, 2015. 851-859.

[5] Bistouni, Fathollah, and Mohsen Jahanshahi. "Pars network: A multistage interconnection network with fault-tolerance capability." Journal of Parallel and Distributed Computing 75 (2015): 168-183.

[6] Kaddari, Abdelhak, et al. "A model of service–oriented architecture based on medical activities for the interoperability of health and hospital information systems." International Journal of Medical Engineering and Informatics 7.1 (2015): 80-100.

[7] Kenkre, Poonam Sinai, Anusha Pai, and Louella Colaco. "Real Time Intrusion Detection and Prevention System." Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014. Springer International Publishing, 2015.

[8] You, Simin, Jianting Zhang, and L. Gruenwald. Scalable and Efficient Spatial Data Management on Multi-Core CPU and GPU Clusters: A Preliminary Implementation based on Impala. Technical report2015.

[9] Liu, Xiaoming, and Qisheng Zhao. "Cluster Key Scheme Based on Bilinear Pairing for Wireless Sensor Networks." Proceedings of the 4th International Conference on Computer Engineering and Networks. Springer International Publishing, 2015.

[10] Singh, Balram, Shankar Singh, and Narendra Kumar Agrawal. "Mobile Agent Paradigm: A Tactic for Distributed Environment."

[11] Mezhuyev, Vitaliy. "Metamodelling Architecture for Modelling Domains with Different Mathematical Structure." Advanced Computer and Communication Engineering Technology. Springer International Publishing, 2015. 1049-1056.