Highly Parallel Rad-Hard RISC-V-based Manycore Accelerators

Luca Benini
ETH (CH) & UniBo (IT)
luca.benini@unibo.it

Ran Ginosar
Ramon Space (UK)
ran.ginosar@ramon.space

Frank K. Gürkaynak
ETH (CH)
frank.k.gurkaynak@gmail.com

Pablo Ghiglino
Klepsydra Technologies (CH)
pablo.ghiglino@klepsydra.com

David Steenari
ESA (NL)
David.Steenari@esa.int

ABSTRACT: A rad-hard highly parallel manycore accelerator, employing streamlined RISC-V cores as processing elements (PEs), is proposed for DSP and AI/ML applications in Space. Very small cores are required to enable integrating 1024 or more ISA-extended PEs with very large on-chip memory. The architecture should support simple and easy programmability, appropriate ranges of operators and formats, and minimal data movements on and off chip. Three alternative architectures are investigated—clustered, networked, and flat shared memory manycores, as well as the three corresponding programming models. To enable future, yet unknown DSP and AI algorithms, architectural flexibility, scalability, and streamlined programming are required. Standard software is preferred, including simulators, programming languages, compilers, analyzers, frameworks, and runtime managers. Interfaces to DRAM, HBM, high speed serial links and die-to-die links should be considered. Radiation hardening, resilience to other Space effects, and strong measures of FDIR are integrated into the architecture, considering the spectrum of requirements from short lifetime LEO to more stringent ESA Class Alpha (previously Class 1) missions. Developing a European manycore accelerator for Space calls for wide collaboration.

1. Introduction

High performance computing in space has traditionally been driven by focus on payload data processing for satcom, Earth Observation, navigation and robotics using traditional DSP (Digital Signal Processing) algorithms such as satcom signal filtering, demodulation, switching and modulation, image data pre-processing and compression (both lossless and bit-rate controlled), navigation signal tracking, and robotics applications (e.g. autonomous landing location selection).

Recently, AI/ML based algorithms have become in satellite on-board processing (OBP), implying ever growing computation and storage requirements, facilitating analytics (classification, detection, information extraction), data selection (for down-link optimization), predictive capabilities, and autonomy (decision making). In applications using AI/ML based algorithms to perform on-board analytics and data selection, there is often still a need to deploy classical algorithms for pre- and post-processing to e.g. ensure data homogeneity (i.e. due to sensor imperfections), or to perform compression on autonomously selected data.

In addition, there is an increasing demand for on-board flexibility, allowing to evolve and change the on-board processing chain with the lifetime of the mission, or to serve multiple applications (or customers) using the same on-board data, e.g. during different periods of an orbit.

Currently, on-board data processing units (DPUs) utilize multicore CPUs and FPGAs for on-board processing tasks. Existing rad-hard multicore CPUs do not meet the requirements for most demanding on-board data processing applications, and FPGAs – while providing excellent performance – require dedicated designs for each application, and cannot easily support multiple different applications simultaneously or in-flight reconfiguration.

It is evident that powerful AI & DSP accelerators are needed since traditional rad-hard space-ready computing systems cannot meet the present and future on-board computing demand in terms of both flexibility and computing performance.

Parallel processing is essential to efficiently accelerate compute-intensive DSP & AI Space workloads. General purpose multicore CPUs, designed for running operating systems and a job mix, are power-hungry and are not scalable beyond a few tens of cores. Specialized domain-specific architectures, such as those based on systolic arrays, can scale to thousands of processing elements , but their rigid and specialized interconnect and execution scheme restrict their usability and utilization [5] and limit their applicability to other domains. Furthermore, their programming model is notoriously difficult to manage [6]. Large manycore architectures such as GPU are more flexible than domain-specific hardwired accelerators, and at the same time are more scalable and more energy efficient than general-purpose processors or large DSP cores.

This paper makes some observations and lays out considerations and open questions regarding plans for Space-ready AI & DSP accelerators based on manycores, rather than proposing a specific design or a product. The rest of the paper explains design considerations, explores alternative architectures, examines required software, considers interfaces and packages, discussed radiation hardness, and reviews previous studies in the area.

2. Design Considerations

Replacing a powerful single core CPU by m tiny and simple cores that occupy the same silicon area conceptually enables in executing the same algorithm $\sqrt{m}$ faster ('speedup') while consuming only $1/\sqrt{m}$ the power and $1/m$ the energy [2][10]. This observation also applies to common multicore processors integrating a small number of powerful cores – a 1024 or larger manycore (occupying the same area as a multicore, e.g., [4]) outperforms the multicore on high-end computational challenges.

The efficiency in manycore architectures relies on number of cores, hence increasing the core count by using the smallest size while maintaining a simple interconnect to orchestrate them is one of the key challenges. The first key to effective manycores is simple and easy programmability. For instance, clustered manycores such as GPUs, while offering clear advantages, call for extensive programming effort in order to optimize their use. Easy programmability also depends on a simple and clear programming model, that exposes algorithmic parallelism, without requiring "heroic" compilers to re-discover it from a convoluted sequential program.

To be useful in a variety of applications, whether well-known ones as well as those that have not yet been invented, the manycore architecture should be readily programmable for a wide range of data intensive and parallel tasks. Clearly, basic RISC-V cores such as RV32IMF (integer, multiply, floating point) may fit that model (the Snitch [8] family of cores is an example).

Domain-specific ISA extensions of a simple RISC-V core for AI & DSP accelerator manycores are considered. Such extensions could support, for instance, linear algebra at varying bit precisions and non-linear operators like softmax, useful in AI applications.

An additional key challenge is to ensure that the selected architecture, as well as the adopted algorithms and their software implementation, minimize data movements, both within the manycore chip and to/from external memory. This is an orthogonal dimension to pure parallelism, but equally important, if not more, given the dominance of memory access cost in modern silicon technologies.

3. Architectural Exploration

In order to fit a large number of cores (1024 or more) into a single chip, each core had better be the smallest useful one, while still maintaining performance for key applications. The core is

studied carefully, investigating published small and efficient cores used in manycore architectures, and evaluating area cost of any extensions versus algorithmic gains in Space AI & DSP applications, simplifying the core to increase its efficiency per area. For instance, a recent study [4] demonstrates such integration using a single-stage ISA-extended Snitch [8].

While per-core acceleration is made possible by ISA extensions, it may be more efficient to create specific chip-level accelerators such as a matrix multiplier or a frequency-domain transform accelerator (Fig. 1). An IP library can be developed of domain-specific hardwired accelerators with easy plug-in logic to connect them (control-plane and data-plane) in the many-core fabric. The architects may study this trade-off in each case and for the most significant operators. However, such accelerators – being domain or application specific – have to be traded off in terms of utilized chip area, in favor of adding more cores for general purpose data processing applications.

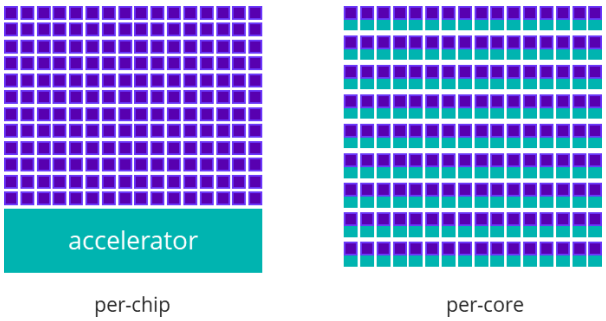

Fig. 1: Per-chip vs. per-core accelerators

Three alternative manycore architectures bring about corresponding three different programming models (Fig. 2). Clustered architectures (Occamy [12], GPU) encourage divide-and-conquer of complex algorithms into smaller units, but require complex hardware and software implementations to access data cross clusters. Networked manycores, similar to clustered architecture but using simpler interconnect, imply message-passing programming, requiring code for moving data around. Flat shared memory architectures seem simplest to program, with a model similar to that of a single core. Each core is equipped with a local cache and a scratchpad memory to facilitate locality and decouple execution from access to shared memory. Remote data accesses in flat shared memory architectures are hidden from the code and appear as cache line fetches and updates.

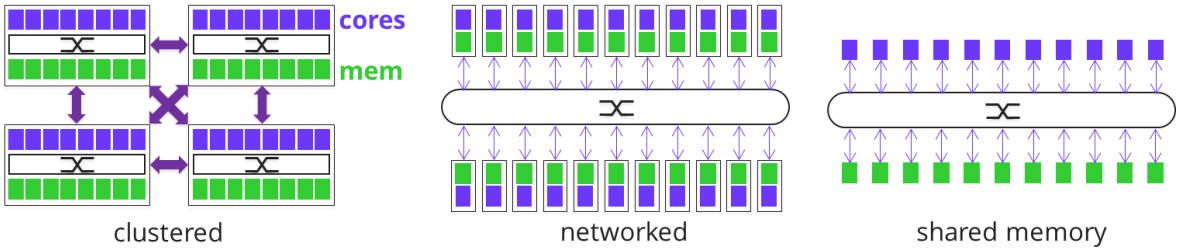

Fig. 2: Clustered, networked, and shared memory manycore architectures

Sizing on-chip memory vs. the number of cores in a manycore accelerator requires simulation studies and optimization (Fig. 3. A memory hierarchy is structured, for instance, assigning caches and scratchpad memories to cores, inserting second level caches, and a larger shared memory that is either a last-level cache (offering a window to off-chip main memory) or an on-chip main

memory, implying DMA-managed I/O access to off-chip memory. I/O-bound applications such as the autoregressive phase in LLM inference are considered in order to optimize the memory hierarchy, applying a roofline model.

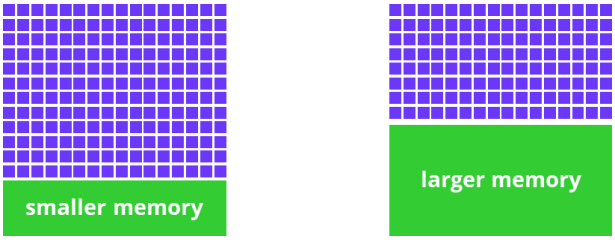

Fig. 3: Memory size vs. number of cores trade-off

I/O to both data streams and large memories are structured as streams and handled by DMA controllers or smarter I/O processors. Data access patterns in accelerators are reasonably predictable and can benefit from large volume transactions at high data rates.

An architecture study, in anticipation of designing a high end space-ready manycore accelerator on UDSM (ultra-deep sub-micron) technology will be preceded by intensive simulations, considering relevant applications as in [4] and including programming models and appropriate software tools, in order to drive design choices with a quantitative methodology.

4. Software

Minimizing the need for specific, proprietary software tools and methods is essential. ISA-extended RISC-V cores are supported by freely available compilers and profilers, while manycore architectural simulators are accessible from the ETH PULP project [9] and other sources. Parallel languages like OpenMP (for homogeneous multicores) and OpenCL (for heterogeneous systems) require special, non-standardized compilers and struggle with large manycores. In contrast, simpler models, such as the task-oriented RC64 model [7], ease development, support progressive optimization, and better suit manycore accelerators.

Beyond telecommunication applications [4][6][7], AI/ML applications are expected to be the most demanding application driver for manycore accelerators in space [1]. On-board data processing AI/ML spans analytics (e.g., ship or fire detection in EO data, cloud-screening), predictive analysis (e.g., fault estimation), and autonomy (e.g., preventive maintenance and constellation self-management). Early deep-learning models are giving way to large language models, transformers, and Retrieval-Augmented Generation (RAG). The space sector is likely to adopt AI methodologies from terrestrial applications, reducing the need for custom solutions in this niche market. However, AI is in tumultuous development, thus architectural flexibility, scalability, and streamlined programming are key requirements for future space architectures.

Efficient software frameworks are needed to manage end-to-end data flow in both classical and AI-based applications, from sensor data acquisition to processing and transmission. While tools like TensorFlow Lite and TensorFlow Micro support AI on compatible hardware, comprehensive solutions remain limited. Klepsydra AI and Streaming framework addresses this gap by including the support for multiple hardware accelerators and optimizing AI and data processing orchestration [10].

One challenge is that for ESA Class Alpha missions rigorous software verification and validation is required following ECSS standards. For a rad-hard space-qualified DSP & AI accelerator, the full software stack, including tools and libraries used in the on-board software have to be designed with such standards in mind – and any required on-board software libraries be provided as pre-qualified software packages, to ensure adoption into such programs.

5. Interfaces, Packaging, Integration

The primary I/O interfaces of a manycore accelerators employ High Speed Serial Links utilizing multi-gigabit SERDES, supporting both data streams (which include chip-to-chip links in multi-accelerator scale-out applications) and external DRAM access. In terms of interconnect to external devices, such as host CPUs or FPGAs (for expanding interface capabilities, e.g. to sensors with non-standard interfaces), both standard terrestrial high-speed interface protocols such as PCIe and multi-gigabit Ethernet standards (as such protocols are already available in some space qualified up-screened COTS devices), as well as space-specific protocols (such as SpaceFibre, which has been adopted in the European ADHA (Advanced Data Handling Architecture) as the high-speed data interface over backplanes) should be considered and traded-off, maximizing the possible board- and system-level architectures employing one or multiple DSP & AI manycore accelerators.

For external memory, if HBM (High Bandwidth Memory) becomes usable in space, it should be supported as well by the HSSL interfaces. As a minimum, DDR4 should be supported – as space-qualified DDR4 devices are already available on the market.

Chiplet support, including D2D interfaces such as UCIe, may be considered for two purposes: First, homogeneous integration of several manycore dies may facilitate higher yield fabrication of smaller dies. Second, the manycore accelerator may be integrated heterogeneously with CPU, FPGA, I/O and other types of chiplets for easy build-up of specific mission solutions and platform derivatives. If a network-centric approach is selected for such systems, they are readily expandable and scalable over multiple packages.

6. Space Worthiness and Radiation Hardness

Radiation hardness can be judiciously embedded in the manycore. TID resilience is mostly inherent in UDSM processes, as long as certain structures (PLL, etc.) are treated carefully. Destructive latch-up (SEL) should be avoided, by proper selection of the cell library and operating voltage. Upsets (SEU and SEFI) are mitigated by ECCs, EDACs, monitors, and other well-known circuit and architecture approaches. While a large portion of the Space market requires only a modest level of protection, appropriate for LEO missions shorter than 10 years, the manycore accelerator should also be suitable for ESA Class Alpha and Beta (previously Class 1-3) missions (e.g. for longer LEO missions with high reliability/availability requirements, GEO or Moon/Mars missions) with more stringent requirements.

The manycore accelerator must also be resilient to environmental stress such as thermal cycles, board-level reliability challenges, and mechanical vibrations. There could be more than one type of package, serving the different market sectors.

FDIR (Fault Detection, Isolation and Recovery), as well as more advanced methods for AI-based self-healing, are included in the architecture. These means are inserted at multiple levels—cells, circuits, architecture (e.g., [13]), power and clock networks, and software.

7. Previous Studies

ETH (Zurich) and the University of Bologna have studied various manycore architectures [3][4]. Ramon Space has designed and manufactured RC64 [7]. Different programming models have been investigated and could be considered: BareC, OpenMP, Halide domain specific language, and the task-oriented model developed for RC64. Klepsydra has undertaken several projects to validate its software development and run-time solutions for Space AI & DSP applications, including testing on space-qualified hardware and software, and has recently conducted an in-orbit demonstration using AI on a GPU-based computer [10]. ESA has studied the performance of key on-board processing applications, both classical and AI/ML based, in multicore and parallel architectures in the OBPMark and GPU4S studies [14].

Acknowledgement
Comments by Peleg Aviely are highly appreciated.

References
[1] Feruglio, Lorenzo, et al. "Future-ready space missions enabled by end-to-end AI adoption." In Artificial Intelligence for Space: AI4SPACE. CRC Press, pp. 303-360, 2023.
[2] Ginosar, R., “Theoretical complexity analysis of many-cores on a single chip,” https://arxiv.org/abs/2502.00153, 2011.
[3] Riedel, S., Cavalcante, M., Andri, R., & Benini, L. “MemPool: A scalable manycore architecture with a low-latency shared L1 memory.” IEEE Transactions on Computers 72(12), pp. 3561-3575, 2023.
[4] Zhang, Y., Bertuletti, M., Zhang, C., Riedel, S., Vanelli-Coralli, A., & Benini, L. “A 1024 RV-Cores Shared-L1 Cluster with High Bandwidth Memory Link for Low-Latency 6G-SDR.” arXiv:2408.08882, 2024 (and https://arxiv.org/abs/2405.04988).
[5] M. Petry, A. Koch, M. Werner, U. Hoch, T. Helfers, R. Wiest, “Machine Learning on Telecommunication Satellite,” DASIA 2023.
[6] A. Mège, “SMOS HR Processing Study,” DASIA 2023.
[7] Ginosar, Ran, Peleg Aviely *et al*. "RC64: High performance rad-hard manycore." IEEE Aerospace Conf., 2016.
[8] Zaruba, Florian, *et al*. "Snitch: A tiny pseudo dual-issue processor for area and energy efficient execution of floating-point intensive workloads." IEEE Trans. Comp. 70(11), pp. 1845-1860, 2020.
[9] Riedel, Stefan, *et al*. "Banshee: A Fast LLVM-Based RISC-V Binary Translator," IEEE/ACM Int. Conf. Comp. Aided Design (ICCAD) 2021.
[10] P. Ghiglino, M. Harshe, D. Steenari and L. Mansilla, "AI/ML Inference Engine Software for High-Reliability applications on Space Qualifiable Hardware," EDHPC, 2023.
[11] Luca Benini, private communication, 2025.
[12] Scheffler, P., Benz, T., Potocnik, V., Fischer, T., Colagrande, L., Wistoff, N., ... & Benini, L. Occamy: A 432-Core Dual-Chiplet Dual-HBM2E 768-DP-GFLOP/s RISC-V System for 8-to-64-bit Dense and Sparse Computing in 12-nm FinFET. IEEE Journal of Solid-State Circuits, 2025.
[13] Rogenmoser, M., Tortorella, Y., Rossi, D., Conti, F., & Benini, L., Hybrid modular redundancy: Exploring modular redundancy approaches in RISC-V multi-core computing clusters for reliable processing in space. ACM Trans. Cyber-Physical Systems, 9(1), 1-29, 2025.
[14] Steenari, D., Kosmidis, L., et al, OBPMark (on-board processing benchmarks)-open source computational performance benchmarks for space applications. 2nd European Workshop on On-Board Data Processing (OBDP), 2021.