CONTINUUM ANR Project – Summary of Project Achievements

Abdoulaye Gamatié

CONTINUUM

A FRENCH ANR PROJECT

Design Continuum for Next Generation Energy-Efficient Compute Nodes

Project Ref. Number ANR-15-CE25-0007

Summary of Project Achievements

Version 1.0
June 2019
Final

Public Distribution

Cortus, Inria, LIRMM

Project Partners: Cortus S.A.S, Inria, LIRMM

Project Partner Contact Information

Cortus S.A.S Michael Chapman 97 Rue de Freyr Le Génésis 34000 Montpellier France Tel: +33 430 967 000 E-mail: michael.chapman@cortus.com	**Inria** Erven Rohou Inria Rennes - Bretagne Atlantique Campus de Beaulieu 35042 Rennes Cedex France Tel: +33 299 847 493 E-mail: erven.rohou@inria.fr
LIRMM Abdoulaye Gamatié Rue Ada 161 34392 Montpellier France Tel: +33 4 674 19828 E-mail: abdoulaye.gamatie@lirmm.fr	

Table of Contents

Executive Summary

The CONTINUUM project aims to define a new energy-efficient compute node model, which will benefit from a suitable combination of efficient compilation techniques, emerging memory and communication technologies together with heterogeneous cores. It is a french collaborative research project, coordinated by Dr. Abdoulaye Gamatié. The consortium includes the Cortus S.A.S Company, the Inria Rennes - Bretagne Atlantique research center and the LIRMM laboratory of Montpellier. The project started in October 2015 and lasted 42 months. It benefited from ANR funding of 573 K€ for a global cost of € 781 K€, under the grant reference ANR-15-CE25-0007-01.

This report summarizes the main achievements of the project over its running period. More details will be found in the publications corresponding to the results reported here. These contributions globally concern: i) energy consumption optimization through innovative compilation approaches and cache memory system design, taking into account the integration of non-volatile memory technologies; ii) workload management (in particular, workload mapping) enabling performance and energy improvements in heterogeneous multicore systems; iii) and finally, a multicore architecture prototype designed with the core technology from the Cortus Company.

Further information about the project could be found on its dedicated website: `http://www.lirmm.fr/continuum-project/`

1 Introduction

1.1 Challenges and motivations

The current trend in the digital world suggests a paradigm shift where computations and data will be mobile and processed transparently to users. This comes with a number of major technological challenges amongst which one can underline security, reliability and energy-efficiency.

The CONTINUUM project addresses the last challenge through a multidisciplinary approach combining expertise in microelectronics and computer science to design multicore systems integrated on silicon energy efficient. Opportunities of power saving exist at different design levels of these systems: software layer, hardware architecture, technological components, etc. However, truly measurable gains will come only through a synergistic exploitation of these levels.

To achieve this goal, the CONTINUUM project brings together experts with complementary skills to foster the emergence of an approach to designing multicore systems that meet the challenge of energy-efficiency in the coming years.

1.2 Rationale and considered methodology

The holistic design flow investigated in the CONTINUUM project is originally motivated by a number of considerations. First, there is a need for a co-design approach between both compiler and architecture designers, and technology experts. This should bring all the actors to tightly cooperate towards the seamless definition of the target compute node architecture, which able to answer the energy-efficiency challenge.

Second, beyond the choice of suitable CPU cores, we also consider carefully the selection of suitable technologies that could significantly contribute to reducing the expected system power consumption without compromising the performance. In particular, we explored emerging non-volatile memory technologies as a key solution.

Finally, as a consequence, current compilation and runtime management techniques need to be revisited to adapt them to the advanced features of the designed compute node, e.g., core and memory heterogeneity, which calls for some adaptive management of workloads and data.

From a methodological point of view, this project is based on an approach combining the following ingredients:

- low-power processor technologies are adopted to build targeted multicore systems. These processors are developed by Cortus S.A.S, a leader in semiconductors and embedded systems. They offer an ideal compromise in terms of performance and dissipated power.
- since memory is a notable energy-consuming component in the systems under consideration, innovative memory technologies that reduce the dissipated power without degrading performance are taken into account. In this case, emerging non-volatile memories (NVMs) are preferred.
- a joint design approach (or co-design) able to take advantage of the technologies mentioned above is considered in order to minimize the energy consumption of the systems. It is based on new techniques for efficient program compilation and computer resource management.

To address the aforementioned challenges, the project brings together 3 partners:

- Cortus is a privately-owned, venture-backed, company based in the South of France – Montpellier, which provides processor and ancillary intellectual property (IP) to customers who are typically fabless semiconductor manufacturers. Cortus has developed a new modern RISC architecture aimed at embedded designs using C/C++ software. The current product offering of two families of processor cores spans a range of 32-embedded requirements. The cores are very silicon efficient and consume very little power. Up to 2014, over forty companies have licensed Cortus IP and more than 700 million products containing Cortus processors have been shipped. Cortus has the unique combination of expertise in compiler development, processor architecture and implementation, RTOS development and multicore hardware design which provides the competitive differentiation in its products.
- Inria is a french public research body fully dedicated to computational science. Its missions are to produce outstanding research in the computing and mathematical fields of digital sciences and to ensure its impact on the economy and society through technology transfer and innovation. Contributions to CONTINUUM project will come through the PACAP group of the Inria Rennes Bretagne Atlantique Research Center. This group focuses on computer architecture, software/-compiler optimization and real-time systems. PACAP has a proven track-record in compiler and architecture research. Participation in the CONTINUUM project is based on past experience in static and just-in-time compilation and virtualization. PACAP also has significant experience in light-weight performance monitoring and development of related tools.
- The Montpellier Laboratory of Informatics, Robotics and Microelectronics (LIRMM in French) is a cross-faculty research entity of the University of Montpellier and CNRS. The research activities concern modeling and designing hardware and software systems (e.g. robots and integrated circuits), as well as algorithmics, bioinformatics, etc. Leveraging this diversity, LIRMM reinforces its originality by combining theory, tools, experiments and applications in its expertise areas, favoring the emergence of interdisciplinary projects. Contributions to CONTINUUM will come through the ADAC group of the Microelectronics Department, which addresses computer architecture / computer science, digital hardware, and emerging technologies. with its laboratory and in contact with other laboratories and scientific fields (such as mathematics, life sciences, health, and neuroscience).

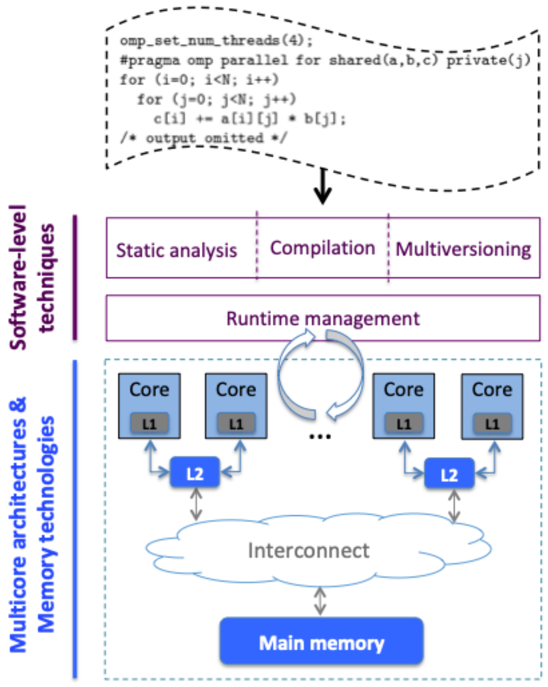
1.3 Contributions

According to the above considerations, we therefore study a system design "continuum"[1] that seamlessly goes from software level to memory technology level via hardware architecture. Fig. 1 depicts the different aspects involved in the considered design flow.

The contributions of the CONTINUUM project are multiple. First, regarding the integration of emerging NVM technologies in the memory hierarchy, a number of original results have been established on the efficient analysis and optimization of programs with data stored in NVMs. Conducted experiments showed that this provides a significant reduction in the energy consumed by the memory.

On the other hand, several prototypes of heterogeneous multicore architectures based on the Cortus technology have been realized for the first time. This confirmed the energy-efficiency of our proposal. Intelligent resource allocation policies exploiting the characteristics of considered embedded technology enable to maximize this efficiency. In particular, machine learning techniques were used to help make allocation decisions.

[1]Hence the name of the project.



```
omp_set_num_threads(4);
#pragma omp parallel for shared(a,b,c) private(j)
for (i=0; i<N; i++)
  for (j=0; j<N; j++)
    c[i] += a[i][j] * b[j];
/* output_omitted */
```

Figure 1: A holistic design flow

Beyond the prototypes designed in the project, it is important to note that more generally a significant effort in software development made by project partners. The resulting software has been made freely available to academic and industrial actors. For example, the Multicore Architecture enerGy and Performance Evaluation Environment (MAGPIE) [20] dedicated to the modeling and evaluation of multicore systems integrating emerging NVMs has been already used by users outside of the CONTINUUM project.

The work carried out within the framework of the project was the subject of two Ph.D. theses [47, 10] successfully defended at the Universities of Montpellier and Rennes 1. It also gave rise to several publications in peer-reviewed scientific journals and conferences, as well as a few guest lectures (e.g., winter/summer schools, keynotes). A one-day tutorial dedicated to MAGPIE was organized during the COMPAS'2017 Symposium in Sophia-Antipolis (France), as well as a special session on the topics of the project during the ReCoSoC'2018 international conference in Lille (France).

1.4 Outline of the report

The remainder of this report is organized as follows: starting from a brief presentation of MAGPIE (Section 2), we deal with NVM technology integration in multicore architectures, by respectively discussing the effort achieved in the project to optimize system energy consumption through hybrid cache and main memory modeling (Section 3) and via innovative compilation techniques (Section 4); then, we cover the workload management issue (Section 5), which is central for performance and energy improvements; in addition, we describe the proposed compute node architecture prototype designed with the Cortus core technology (Section 6); finally, we give some concluding remarks (Section 7).

2 MAGPIE design exploration environment

Design space exploration can be achieved based on various supports, including typically accurate FPGA-based prototyping [62, 46], fast analytical modeling [5] or user-friendly UML-based model-driven approach [27, 65]. While each of these supports has its own advantage in terms accuracy, speed or flexibility, the CONTINUUM project combined selected simulators in order to build a suitable framework, which is capable of addressing both performance and energy in an effective and relevant manner.

The Multicore Architecture enerGy and Performance Evaluation Environment (MAGPIE[2]) [21, 20, 59], devised in the project, is a framework dedicated to an easy modeling and evaluation of multicore systems integrating emerging NVMs. It has been built based on preliminary studies in LIRMM [58, 57, 56, 15, 16], addressing NVM integration in cache memory hierarchy.

2.1 Corresponding flow

MAGPIE promotes a generic evaluation flow depicted in Fig. 2.

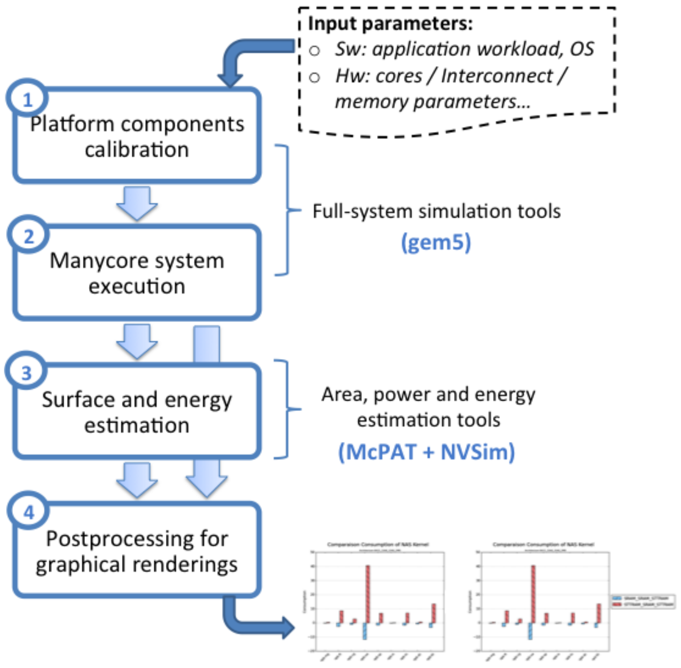

Input parameters:
Sw: application workload, OS
Hw: cores / Interconnect / memory parameters...
1
Platform components calibration
2
Manycore system execution
Full-system simulation tools
(gem5)
3
Surface and energy estimation
Area, power and energy estimation tools
(McPAT + NVSim)
4
Postprocessing for graphical renderings

Figure 2: MAGPIE evaluation flow.

The inputs of this flow comprise information related to both software and hardware components of a given system. The software-related inputs include a gem5 execution script file for each workload/application to be executed, together with the underlying operating system supported by the considered full-system simulator.

From the hardware perspective, a number of parameters of the target manycore architectures are required: types and number of cores, memory hierarchy and its technology-specific properties, and the type of interconnect. In general, the candidate full-system simulators for MAGPIE provide an IP library that significantly facilitates the instantiation of target architectures.

[2]This framework has been developed in collaboration with the GREAT European H2020 project. It is freely distributed via the following address: `http://www.lirmm.fr/continuum-project/pages/magpie.html`

Provided the above input information, MAGPIE proceeds through four main steps as follows:

1. *Platform components calibration*: the basic parameters of all hardware components are set up in the full-system simulator. Typically, the operating frequency of cores, the memory size and access latencies at different levels of the memory hierarchy are defined. For NVMs, their corresponding read/write latencies usually vary according to their characteristics such as their type [43] (e.g., RRAM, PCM, MRAM), technology node (e.g., 65nm, 45nm) and size.

2. *Manycore system execution*: the full system simulation of the user-customized system is performed in order to obtain detailed execution statistics. Note that since the inputs of the MAGPIE flow can specify at the same time different system design choices, several simulation instances can be launched in parallel. This contributes to accelerating the design space exploration with MAGPIE. Beyond the global performance metrics such as execution time, the activity events related to each hardware device are important information. These events are used for power and energy estimation.

3. *Surface and energy estimation*: based on detailed execution statistics generated by gem5, the surface, power and energy consumption of the simulated system are estimated. For instance, the dynamic energy of cache memory is determined based on its collected read/write activity events and the energy consumption of elementary memory access obtained, e.g., with CACTI.

4. *Post-processing for graphical renderings*: as a major goal of MAGPIE is to assist the user in design space exploration, the final evaluation metrics can be reported in both textual and graphical formats. Then the user can quickly gather insights from each evaluated design.

2.2 Implementation

We seamlessly combine the gem5 simulator for performance evaluation and NVSim and McPAT for estimating the energy respectively related to NVMs and the rest of the architecture. These tools are briefly described below.

Considered simulation and estimation tools. The gem5 simulator [1] provides an accurate evaluation of system performance thanks to its high configurability for a fine-grained architecture modeling. Its full-system simulation mode runs unmodified operating systems. It includes several predefined architecture component models, e.g., CPU, memory and interconnect. The simulator produces detailed execution statistics (even at micro-architecture level) for power and footprint area estimation. In the CONTINUUM project, we mainly considered ARM big.LITTLE CPU models in gem5 [16].

McPAT [31] is a power, area, and timing modeling framework for multi-threaded, multicore, and manycore architectures. It works with a variety of performance and thermal simulators via an XML template-based interface. This interface describes the micro-architecture specification and is used to communicate activity events generated by simulators. McPAT covers three simulation levels for estimation: architectural, circuit and technology (from 90nm to 22nm). NVM technologies are not addressed by McPAT. So, we use NVSim [22], which is a circuit-level estimator for NVM performance, energy, and area estimations. It supports different NVM technologies including STT-MRAM, ReRAM, and PCRAM. It uses the same modeling principles as CACTI.

Integration within MAGPIE framework. The MAGPIE framework defines several Python script programs that automate the whole flow illustrated in Fig. 2. The inputs of the flow are first read and used for automatic calibration of the hardware architecture components in gem5. For NVMs, the NVSim tool is invoked by a script in order to calculate the corresponding read/write latencies based on the desired memory type, memory size, associativity and technology node, as specified in the inputs. Then, gem5 is automatically configured with the computed NVM access latencies. For this purpose, we modified gem5 so as to enable the configuration of memories with asymmetric read and write latencies, such as NVMs. Afterward, the specified system execution scenarios are run in parallel (according to the number of cores available on the host machine) by automatically triggering the corresponding number of gem5 simulation instances.

Each gem5 simulation instance produces the execution statistics file related to its design scenario. From these files, all data required by NVSim and McPAT (e.g., the execution time of the system, number of read/write transactions for memory blocks) are automatically extracted by another script. As these files can be huge, the script has been defined in such a way that it optimizes the reading of generated gem5 files. Then, it invokes the two estimation tools on the extracted data in order to generate the area, power and energy consumption for each captured design scenario. The results are stored in textual files.

Finally, the above textual files are post-processed by several scripts for generating various user-friendly renderings: CSV files and graphical plots that compare the performance, area, power and energy evaluation of the different scenarios. For instance, the energy breakdown of the main hardware components can be easily plotted for a fine-grained analysis.

3 NVM integration in memory system

Current design trends show that the memory speed is not growing as fast as cores computing capacity, leading to the so-called *memory-wall* issue. Caching techniques, which have been pushed in the past for mitigating the memory-wall, are facing the silicon area constraints. As the memory hierarchy capacity is increased [63], the corresponding energy consumption grows. Fig. 3 illustrates the fact that memory systems consume more than 50% of the die area. As a result, a significant part of the consumed power is devoted to memory as reported in Fig. 4.

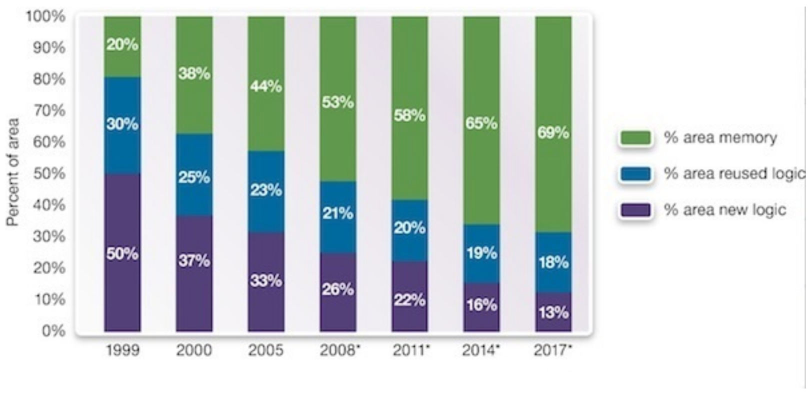


Figure 3: SoC area repartition between logic and memory (from Semico Research Corporation [55])

The current mainstream memory technology used for both cache memory and registers is SRAM thanks to the short data access latency it provides compared to other technologies. However, its potential prohibitive static energy due to the increase of the leakage current when decreasing the technology node

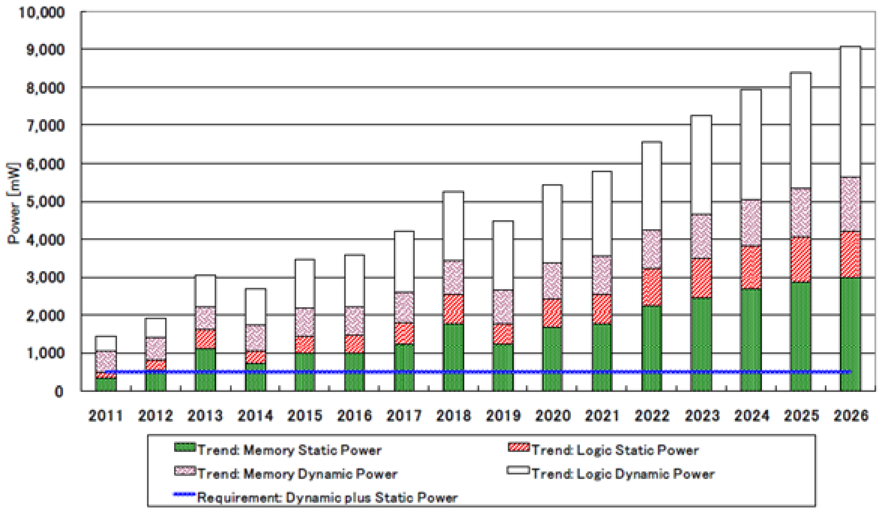


Figure 4: SoC energy repartition between logic and memory (from ITRS [32])

is a major issue to energy-efficiency. In order to address this issue, embedding volatile memories in SoCs is among the most promising trends. Emerging NVMs [43] are promising candidate technologies as they combine simultaneously high density and very low static power consumption while their performance is becoming competitive compared to SRAM and DRAM.

3.1 Cache memory level

Data accesses that occur beyond the Last-Level Cache (LLC) are usually time and energy-consuming as they have to reach the off-chip main memory. An intelligent design of the LLC reducing such accesses can save power and increase the overall performance. A usual technique adopted in the past consists in increasing the cache storage capacity so as to reduce the cache miss rate. This approach is no longer desired due to area and energy constraints. Increasing the cache size has a negative impact on the financial cost and increases the static power consumption.

In our study, we considered an emerging memory technology, the Spin-Torque Transfer Magnetic RAM (STT-RAM), a non-volatile memory teechnology that has a near-zero leakage consumption. This memory has a higher density than SRAM, providing more storage capacity for the same area. While STT-RAM read latency is close to SRAM read latency, the gap for write access is currently one obstacle to a wide STT-RAM adoption. In [48], we addressed the impact in write reduction of cache replacement policies. Each read request leading to a cache miss eventually triggers a write. Upon this cache miss, the request is forwarded to an upper level in the memory hierarchy.[3] When the response is received, the corresponding data is written into the cache. Hence, the cache replacement policy has indirectly an important impact on the number of writes that occur upon cache misses. After a careful review of existing cache replacement policies [52], we evaluated the combined use of STT-RAM and the *state-of-the-art* Hawkeye cache replacement policy [33]. Thanks to Hawkeye, the number of writes due to cache misses (on the *critical path* to main memory) is reduced, while benefiting from STT-RAM density for larger LLC.

More concretely, we observed that using Hawkeye with STT-RAM is more beneficial than with SRAM. Indeed, the read/write latency asymmetry of this technology allows a higher gap of improvement in terms

[3]The first level cache (L1), the closest to the CPU, is the lowest level.

of performance than with SRAM. However, with a large cache that drastically reduces the number of misses, the small amount of accesses does the training of the Hawkeye predictor longer. Thus, it may lead to inadequate eviction decisions. Our results showed that the global system performance can be improved up to 10% and 14% respectively for monocore and multicore platforms. This gain, combined with the drastic static energy reduction enabled by STT-RAM, leads to increased energy-efficiency, up to $26.3\%\times$ and 27.7% for monocore and multicore systems.

3.2 Main memory level

Beyond the cache level, we studied the impact of NVM integration in main memory[4]. The proposed analysis was achieved by adopting two simulation tools [36]: gem5 coupled with the NVMain simulator [50] devoted to main memory modeling and simulation.

We calibrated a DRAM model based on a Micron DDR4 datasheet. We also build new PCM and RRAM memory technology models, starting from NVMain models that are further refined based on an analysis of the existing literature on NVMs. Our evaluation targeted state-of-the-art heterogeneous ARMv8 multicore systems, envisioned for better processing efficiency of compute nodes. The explored system designs were validated on typical application workloads.

Our results [30] showed that RRAM is a very good candidate for energy issue mitigation at main memory level, while PCM would be a more promising candidate for storage level. We showed that RRAM can provide system-level performance comparable to DDR4, while memory energy consumption can be reduced by up to 50%. This is not the case of PCM.

4 Compile-time analysis for mitigating NVM integration cost

We explore compile-time analyses and optimization and software analysis, as a possible alternative to leverage the low leakage current inherent to emerging NVM technologies for energy-efficiency. A major advantage is the flexibility and portability across various hardware architectures enabled by such an approach, compared to the hardware-oriented techniques found in literature.

4.1 Empirical energy evaluation of compiler optimizations with NVM caches

As a preliminary work, we led an empirical study considering multicore system designs with NVM integrated in the cache memory hierarchy, while applying loop nest optimization to application code. Different loop optimizations have been evaluated individually and jointly, taking into account the trade-off between performance and energy with respect to SRAM cache configurations. In particular, we evaluated loop tiling and loop permutation. The former transformation splits the iteration space of a loop into smaller chunks or blocks, in such a way that the data used in the loop remains in the cache until it is reused. As a result, large arrays are split into fine grain blocks, which in turn enables the accessed array elements to fit into the cache size. The other loop transformation Loop consists in exchanging the order of different iteration variables within a loop nest. This contributes to improve the data locality by accessing the elements of a multidimensional array in an order according to which they are available in cache memory.

[4]This study has been conducted in collaboration with partners from IMEC and the H2020 Montblanc3 project.

Our experiments [51] showed that loop optimizations combined with STT-MRAM provide up to 24.2% and 31% energy reduction without performance degradation, respectively compared to multicore and monocore architectures with full SRAM cache hierarchy.

4.2 Detection and elimination of silent stores

Our proposal is inspired by some existing techniques such as the silent store elimination technique introduced by Lepak et al. [39] and worst-case execution time analysis techniques [64], as summarized in the sequel.

Profiling-based analysis. As writes on NVMs are generally more expensive than reads, we advocate a compile-time optimization by consistently reducing the number of writes on memory. Here, writes identified as redundant are eliminated, i.e.: when a strictly or approximately identical value is overwritten in the same memory location, respectively referred to as strict and relaxed silent stores.

In [11, 9], we proposed a silent store elimination technique through an implementation in the LLVM compiler [38]. Thanks to this implementation, a program is optimized once, and run on any execution platform while avoiding silent stores. This is not the case of the hardware-level implementation. We evaluated the profitability of this silent store elimination for NVM cache memories. We showed that energy gains highly depend on the silentness percentage in programs, and on the energy consumption ratio of read/write operations costs for NVMs. We validated our proposal with the Rodinia benchmark suite, while reporting up to 42% gain in energy for some applications. This validation relies on an analytic evaluation considering typical NVM parameters extracted from the literature.

Static prediction approach. While the store silentness analysis carried on in the above study is based on program profiling, we addressed a complementary approach that rather exploits static code analysis for silent store prediction [49]. We studied the influence of syntactic program features on the occurrence of silent stores. From this study, we have derived predictors which determine store instructions that tend to be silent during the execution of programs. This promising study gave a number of interesting conclusions, some of which we list below:

- Syntax that might cause the deposit of the value zero or a boolean value in memory are the most consistent sources of silentness. On the opposite direction, increments and non-zero constants tend to lead to noisy stores (i.e., non silent).
- Silent stores are very silent, and noisy stores are very noisy. A consequence of this observation is that if a store is observed to be silent or noisy, this behavior is likely to repeat if the same instruction is executed again.
- The distribution of silent stores among benchmarks seems to follow a normal distribution. This observation can be used to detect anomalies, such as performance bugs. For instance, a program can be considered suspicious if it contains a number of silent stores above a threshold P falling outside two or three standard deviations of P's distribution.
- Our choice of features leads to the construction of static predictors whose precision is significantly (in a statistical sense) superior to trivial prediction strategies.

4.3 Data lifetime analysis for efficient NVM data allocation

Given the possibility of relaxing the data retention time of NVMs, we leverage a design-time analysis on the lifetime of program variables so as to map them on NVM memory banks with customized retention capacities. This enables to accommodate NVM features with program execution requirements while favoring energy-efficiency.

We applied a partial worst-case execution time (δ-WCET) analysis to programs in order to determine the worst-case lifetimes of program variables involved in store instructions [12, 13]. This information is then used to safely allocate these variables in appropriate NVM memory banks, according to their data retention time. We validated our approach on the Mälardalen benchmark-suite, by showing a significant reduction of memory dynamic energy (up to 80%, with an average of 66%). This contributes to answer the energy-efficiency challenge faced in both embedded and high-performance computing domains.

5 Workload management in multicore systems

Application mapping on multicore platforms has been studied for decades in literature [60]. To find out near-optimal mapping solutions, many mapping techniques adopt search-based approaches combined with some analyses to evaluate considered mappings w.r.t. the design requirements. These analyses typically rely on system-level evaluations leveraging FPGA platforms [62, 4], model-driven design frameworks [27, 53] or analytical models [5, 3, 66].

Some recent approaches advocate machine learning to address the mapping problem [61]. They typically rely on *reinforcement* learning and *supervised* learning. In [8], reinforcement learning is applied through a cross-layer system approach to predict the best energy-performance trade-off in multicore embedded systems. A machine learning based approach is proposed in [41] for the optimal mapping of streaming applications described by the StreamIt formalism onto dynamic multicore processors. In [40], the authors apply machine learning to predict execution time, memory and disk consumption of two bioinformatics applications deployed on different hardware resources.

5.1 Compiler-assisted adaptive program scheduling in heterogeneous systems

Choosing the best core configuration for a running program on heterogeneous platforms is challenging, because program parts benefit differently from the same configuration. We call the task of allocating parts of a parallel program to processors the *code placement problem*. State-of-the-art approaches solve this problem dynamically (e.g., at runtime / operating system levels, or via a middleware) or statically (e.g., via compilers). A dynamic approach can use runtime information to weight the choices it makes, while a static technique reduces runtime monitoring cost and can leverage program characteristics. These two approaches can be joined, achieving a synergy that, otherwise, could not be attained by an individual approach.

To make it possible, we used the compiler to partition source code into program phases: regions whose syntactic characteristics lead to similar runtime behavior. We use reinforcement learning to map pairs formed by a program phase and a hardware state to the configuration that best fit this setup. To validate our proposal, we implemented a framework to instrument and execute applications in heterogeneous architectures: the *Astro system* [45, 44] . The framework collects syntactic characteristics from programs and instruments them using LLVM [38]. For a program region, the scheduler can take into account its corresponding characteristics collected statically, for an immediate action. An action consists in choosing an execution configuration and collecting the "reward" related to that choice. Such feedback is then used to fine-tune and improve the subsequent scheduling decisions. We rely on such a reinforcement learning technique and benefit its ability to explore a vast universe of configurations (or states), formed by different hardware setups and runtime data changing over time. However, the universe of runtime states is unbounded, and program behavior is hard to predict without looking into its source code. To speedup convergence, we resort to the compiler. The compiler gives us two benefits. First, it lets us mine program features, which we can use to train the learning algorithm. Second, it lets us instrument the program. This instrumentation allows the program itself to provide feedback to the scheduler, concerning the code region currently under execution.

The Astro system is summarized in Fig. 5. From source program P to generate an adaptive program P'', the system uses its three components as follows: *Program Instrumentation*, *Actuation* (monitoring and reward computation, learning and adapting), and *Final Code Generation*.

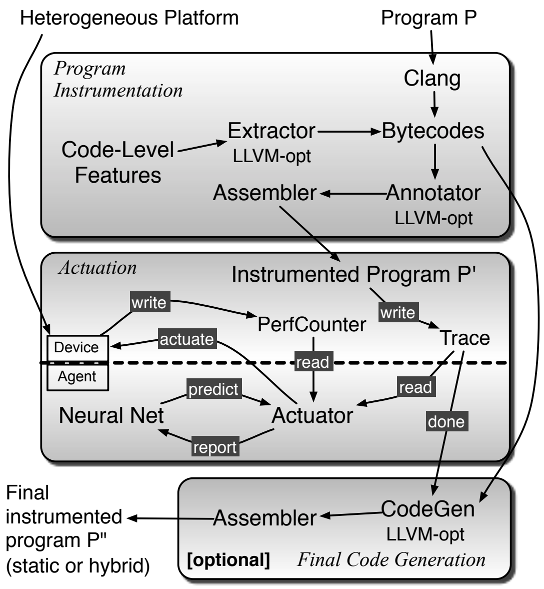


Figure 5: The Astro System.

We evaluated Astro over parallel benchmarks running on a big.LITTLE system. Experiments on Parsec [7] and Rodinia [17] benchmarks running on an Odroid XU4 show that we can obtain speedups of more than 10% over the default GTS scheduler used in ARM-based systems.

5.2 Performance prediction for application mapping in multicore systems

In [28], we addressed the application mapping problem[5] on multicore architectures based on two off-line supervised machine learning approaches: on the one hand, classification through Support Vector Machine (SVM) [18] and Adaptive Boosting (AdaBoost) [25] techniques, and on the other hand, regression by using Artificial Neural Networks (ANNs) [26]. SVM has been very popular in machine learning thanks to its ability to apply in both classification or regression problems, even though it is often used in the former.

[5]This work has been carried out as a follow-up of an earlier work, and in collaboration the Hefei University of Technology in China.

AdaBoost provides an original vision combining different learners that enable accurate classifications while acting together. On the other hand, ANNs have been proved powerful enough to solve various regression problems. Compared to classification techniques, finding a good compromise between accuracy and training cost is however more challenging with ANNs due to their tedious parameterization.

Our obtained results showed that, under some conditions, AdaBoost and ANNs can achieve very promising prediction accuracy with up to 84.8% and 89.05% respectively, which confirms the effectiveness of these two models. SVM gave less precised prediction score, possibly due to its inability to correctly deal with imbalanced distribution of mapping data, which is more tractable with Decision Trees supported in AdaBoost.

6 Multicore asymmetric architecture prototype

Heterogeneous computing usually refers to systems including various processing elements so as to meet both performance and power-efficiency requirements. Typical heterogeneous architectures combine CPUs and compute accelerators such as Graphical Processing Units (GPUs). While the former are well-suited for executing sequential workloads and the operating system, the latter are rather devoted to massively regular parallel workloads, e.g., data-parallel algorithms. Further examples are the Cell multiprocessor [34] developed by Sony, Toshiba and IBM, and the Llano processor [14] proposed by AMD. Cell combines a general-purpose core with streamlined co-processing elements that accelerate multimedia and vector processing applications, whereas Llano combines a quad-core CPU with a GPU.

The more recent ARM big.LITTLE technology [6], which has been used in the CONTINUUM project for validation, considers two different clusters: a big cluster composed of high-performance application processors used to execute heavy workloads; and a LITTLE cluster composed of low power application processors that are used for lightweight workload to save energy. By exploiting this feature, a suitable runtime can provide workloads with required performance while reducing the power consumption whenever possible.

The aforementioned architectures are also referred to as asymmetric multicore architectures due to their core heterogeneity in terms of performance and power consumption [37] [42]. Despite their very attractive features for providing energy-efficiency, asymmetric multicore chips are still not mature and robust in real-world commercial solutions [42].

6.1 Overview

We devised a novel original asymmetric multicore architecture [29], comprising two execution islands: parallel and sequential. While the former is devoted to highly parallelizable workloads for high throughput, the latter addresses weakly parallelizable workloads. Accordingly, the parallel island is composed of many low power cores and the sequential island is composed of a small number of high-performance cores. An originality of our proposal is the usage of the cost-effective and inherently low power core technology provided by Cortus [2], one of the world-leading semiconductor IP companies in the embedded domain. These cores are highly energy (MIPS/μW) and silicon efficient (MIPS/mm^2) compared to existing technologies. We believe the massive usage of such embedded cores deserves attention to achieve the energy-efficient architectures required for high-performance embedded computing.

Our architectural solution is somehow similar to the CPU/GPU heterogeneous design paradigm. However, an important difference is that the parallel island, which plays the same role as the GPU, can deal with both regular and irregular parallel workloads. In addition, both sequential and parallel islands support the same programming model, facilitating the job of programmers. GPUs require specific APIs such as OpenCL and CUDA, which are not necessarily supported by CPUs, requiring extensive software support. Compared to big.LITTLE, which considers only application processors, here we combine application processors on the "big" side and micro-controllers on the "LITTLE" side. Such compact "LITTLE" cores (which are not intended to support a full operating system) are key for aggressive energy optimization. Our proposal considered a task-based programming model. A recent work addressed task mining in sequential programs [54], from which suitable task-oriented programs could be defined.

Another trade-off considered in our solution is the support of floating point arithmetic, which benefits certain operations in embedded applications, e.g., matrix inversion required for Multiple Input / Multiple Output (MIMO), Fast Fourier Transforms (FFTs) which often suffer from scaling problems in fixed point. As floating point units (FPUs) can be expensive in terms of area and power in the very low power cores being considered, it will be supported only by a subset of these cores.

Multi-cluster architectures. Fig. 6 depicts the final prototype architecture, which consists of seven CPU cores. One high performance CPU Core is implemented with Cortus APSX2 core IP. There are then two clusters of lower performance cores:

1. Cluster 1: two lower performance CPU Cores with hardware floating point support; and one lower performance integer only CPU Cores.
2. Cluster 2: one lower performance CPU Core with hardware floating point support; and two lower performance integer only CPU Cores.

Each CPU core has a subset of private peripherals, which generate interrupts only for that CPU core - for example each CPU core has its own counter peripheral to generate periodic interrupts (which is essential for the scheduler).

A message box peripheral is used to pass messages between the CPU cores, this peripheral can generate interrupts for all the CPU cores. The task scheduler uses the message box peripheral to manage the tasks on each of the CPU cores. The use of an interrupt driven scheme ensures maximum efficiency and responsiveness with a minimum of software overhead in the tasks being scheduled. Each CPU core can also use the message box peripheral to pass a message to the controlling CPU Core.

Concretely the controlling CPU core writes a message into the message box and generates an interrupt for the chosen CPU core. To respond the CPU Core writes its response into the message box peripheral and an interrupt is generated for the controlling CPU Core.

Here, the communication interconnect is implemented by a hierarchical crossbar. Note that initially, the project aimed at investigating advanced interconnect infrastructures such as 3D Networks-on-Chip (NoCs) [24, 35, 23]. But, the hierarchical crossbar advocated by the Cortus partner proved adequate-enough for the prototype implementation.

Memory architecture. All CPU cores share both program and data memory but have local data and instruction caches. The caches are not coherent. Each CPU core has a reserved address range (which is cached by the CPU's local caches) and there is a system wide shared memory reserved for inter CPU communication (this memory region is marked as non-cacheable).

A level 2 cache caches all the memory access to the external DDR memory, reducing the memory access latency.

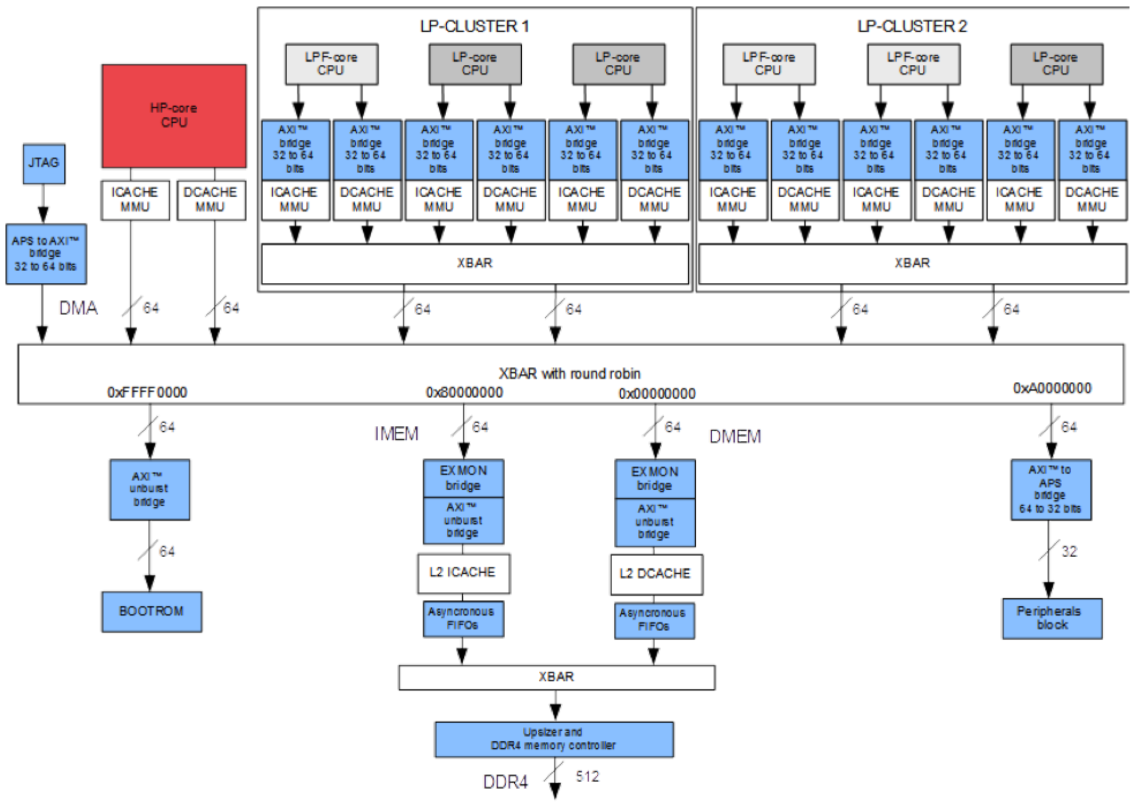


Figure 6: Overview of the compute node architecture (syntesized on a Xilinx Virtex Ultrascale VCU108 FPGA board).

Workload management system A tailored lightweight and flexible multi-thread execution model is also defined in order to enable the management of programs executed on the proposed architecture, which is synthesized on FPGA prototypes. This workload management exploits the nature of programs, which is analyzable statically during compilation, e.g., computation versus memory intensiveness, floating point intensive or not.

6.2 Evaluation

The above asymmetric multicore architecture has been comprehensively evaluated in terms of performance and energy efficiency. A tailored energy measurement infrastructure, inspired by [19], has been deployed for the comparison.

Different prototype versions have been devised on FPGA: quadcore versus heptacore architectures [29]. Our experiments showed that an adequate multi-benchmark workload management on the heterogeneous cores can provide about 22% energy gain on average, compared to a reference design. This makes our solution a very promising candidate for typical embedded systems such as edge compute nodes where energy-efficiency is key.

7 Conclusions

This report presented a brief description of the main achievements realized during the CONTINUUM project, which aims to design energy-efficient compute node systems.

On the one hand, a number of results have been presented regarding the integration of emerging NVM technologies in memory hierarchy. These comprise the efficient analysis and optimization of programs with data stored in NVMs. Conducted experiments showed that a non negligible energy reduction can be reached for the memory.

On the other hand, several prototypes of heterogeneous multicore architectures based on the Cortus technology have been realized for the first time. Adapted resource allocation strategies exploiting the characteristics of considered embedded technology enable to maximize the overall energy-efficiency.

Beyond the architecture prototypes designed in the project, a significant effort in software development has been made by involved partners. The resulting software is made freely available for experimental usage. For example, the MAGPIE environment dedicated to the modeling and evaluation of multicore systems integrating emerging NVMs is already adopted by several users beyond the CONTINUUM project.

References

[1] The gem5 simulator. http://www.gem5.org, 2019.

[2] Cortus SAS – Advanced Processing Solutions. `http://www.cortus.com`, July 2017.

[3] An, X., Boumedien, S., Gamatié, A., and Rutten, E. Classy: A clock analysis system for rapid prototyping of embedded applications on mpsocs. In *Proceedings of the 15th International Workshop on Software and Compilers for Embedded Systems*, SCOPES '12, pages 3–12, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1336-0. doi: 10.1145/2236576.2236577. URL `http://doi.acm.org/10.1145/2236576.2236577`.

[4] An, X., Rutten, E., Diguet, J.-P., Le Griguer, N., and Gamatié, A. Autonomic Management of Reconfigurable Embedded Systems using Discrete Control: Application to FPGA. Research Report RR-8308, INRIA, May 2013. URL `https://hal.inria.fr/hal-00824225`.

[5] An, X., Gamatié, A., and Rutten, É. High-level design space exploration for adaptive applications on multiprocessor systems-on-chip. *Journal of Systems Architecture - Embedded Systems Design*, 61(3-4):172–184, 2015.

[6] ARM Ltd. big.little technology: The future of mobile. 2013. URL `https://www.arm.com/`.

[7] Bienia, C., Kumar, S., Singh, J. P., and Li, K. The PARSEC benchmark suite: Characterization and architectural implications. In *PACT'08*, pages 72–81. ACM, 2008.

[8] Biswas, D., Balagopal, V., Shafik, R., Al-Hashimi, B. M., and Merrett, G. V. Machine learning for run-time energy optimisation in many-core systems. In *Proceedings of the Conference on Design, Automation & Test in Europe*, DATE '17, pages 1592–1596, 3001 Leuven, Belgium, Belgium, 2017. European Design and Automation Association. URL `http://dl.acm.org/citation.cfm?id=3130379.3130749`.

[9] Bouziane, R., Rohou, E., and Gamatié, A. How could compile-time program analysis help leveraging emerging nvm features? In *2017 First International Conference on Embedded Distributed Systems (EDiS)*, pages 1–6, Dec 2017. doi: 10.1109/EDIS.2017.8284031.

[10] Bouziane, R. *Software-level analysis and optimization to mitigate the cost of write operations on non-volatile memories*. Theses, Université Rennes 1, December 2018. URL `https://tel.archives-ouvertes.fr/tel-02089718`.

[11] Bouziane, R., Rohou, E., and Gamatié, A. Compile-time silent-store elimination for energy efficiency: an analytic evaluation for non-volatile cache memory. In Chillet, D., editor, *Proceedings of the RAPIDO 2018 Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools, Manchester, UK, January 22-24, 2018.*, pages 5:1–5:8. ACM, 2018. ISBN 978-1-4503-6417-1. doi: 10.1145/3180665.3180666. URL `https://doi.org/10.1145/3180665.3180666`.

[12] Bouziane, R., Rohou, E., and Gamatié, A. Energy-efficient memory mappings based on partial WCET analysis and multi-retention time STT-RAM. In Ouhammou, Y., Ridouard, F., Grolleau, E., Jan, M., and Behnam, M., editors, *Proceedings of the 26th International Conference on Real-Time Networks and Systems, RTNS 2018, Chasseneuil-du-Poitou, France, October 10-12, 2018*, pages 148–158. ACM, 2018. doi: 10.1145/3273905.3273908. URL `https://doi.org/10.1145/3273905.3273908`.

[13] Bouziane, R., Rohou, E., and Gamatié, A. Partial Worst-Case Execution Time Analysis. In *ComPAS: Conférence en Parallélisme, Architecture et Système*, pages 1–8, Toulouse, France, July 2018. URL `https://hal.inria.fr/hal-01803006`.

[14] Branover, A., Foley, D., and Steinman, M. Amd fusion apu: Llano. *IEEE Micro*, 32(2):28–37, March 2012. ISSN 0272-1732. doi: 10.1109/MM.2012.2.

[15] Butko, A., Gamatié, A., Sassatelli, G., Torres, L., and Robert, M. Design exploration for next generation high-performance manycore on-chip systems: Application to big.little architectures. In *2015 IEEE Computer Society Annual Symposium on VLSI*, pages 551–556, July 2015. doi: 10.1109/ISVLSI.2015.28.

[16] Butko, A., Bruguier, F., Gamatié, A., Sassatelli, G., Novo, D., Torres, L., and Robert, M. Full-system simulation of big.little multicore architecture for performance and energy exploration. In *2016 IEEE 10th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSOC)*, pages 201–208, Sep. 2016. doi: 10.1109/MCSoC.2016.20.

[17] Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J. W., Lee, S.-H., and Skadron, K. Rodinia: A benchmark suite for heterogeneous computing. In *IISWC*, pages 44–54. IEEE, 2009.

[18] Cortes, C. and Vapnik, V. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995.

[19] da Silva, J. C. R., Pereira, F. M. Q., Frank, M., and Gamatié, A. A compiler-centric infrastructure for whole-board energy measurement on heterogeneous android systems. In Niar, S. and Saghir, M. A. R., editors, *13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip, ReCoSoC 2018, Lille, France, July 9-11, 2018*, pages 1–8. IEEE, 2018. ISBN 978-1-5386-7957-9. doi: 10.1109/ReCoSoC.2018.8449378. URL `https://doi.org/10.1109/ReCoSoC.2018.8449378`.

[20] Delobelle, T., Péneau, P., Gamatié, A., Bruguier, F., Senni, S., Sassatelli, G., and Torres, L. MAGPIE: System-level Evaluation of Manycore Systems with Emerging Memory Technologies. In *Workshop on Emerging Memory Solutions - Technology, Manufacturing, Architectures, Design and Test at Design Automation and Test in Europe (DATE)*, 2017.

[21] Delobelle, T., Péneau, P.-Y., Senni, S., Bruguier, F., Gamatié, A., Sassatelli, G., and Torres, L. Flot automatique d'évaluation pour l'exploration d'architectures à base de mémoires non volatiles. In *ComPAS: Conférence en Parallélisme, Architecture et Système*, Compas'2016 : Parallélisme/ Architecture / Système, Lorient, France, July 2016. URL `https://hal-lirmm.ccsd.cnrs.fr/lirmm-01345975`.

[22] Dong, X., Xu, C., Xie, Y., and Jouppi, N. P. NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory. *IEEE Trans. on Computer-Aided Design of Integ. Circ. and Sys.*, 31(7):994–1007, 2012.

[23] Effiong, C., Lapotre, V., Gamatie, A., Sassatelli, G., Todri-Sanial, A., and Latif, K. On the performance exploration of 3d nocs with resistive-open tsvs. In *2015 IEEE Computer Society Annual Symposium on VLSI*, pages 579–584, July 2015. doi: 10.1109/ISVLSI.2015.49.

[24] Effiong, C., Gamatié, A., and Sassatelli, G. Design Exploration Framework for 3D-NoC Multicore Systems under Process Variability at RTL level. Research report, LIRMM (UM, CNRS), September 2018. URL `https://hal-lirmm.ccsd.cnrs.fr/lirmm-01870671`.

[25] Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, August 1997. ISSN 0022-0000.

[26] Fyfe, C. Artificial neural networks. In Gabrys, B., Leiviskä, K., and Strackeljan, J., editors, *Do Smart Adaptive Systems Exist?*, volume 173 of *Studies in Fuzziness and Soft Computing*, pages 57–79. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-24077-8.

[27] Gamatié, A., Le Beux, S., Piel, E., Ben Atitallah, R., Etien, A., Marquet, P., and Dekeyser, J.-L. A model-driven design framework for massively parallel embedded systems. *ACM Trans. Embed. Comput. Syst.*, 10(4):39:1–39:36, November 2011. ISSN 1539-9087. doi: 10.1145/2043662.2043663. URL `http://doi.acm.org/10.1145/2043662.2043663`.

[28] Gamatié, A., An, X., Zhang, Y., Kang, A., and Sassatelli, G. Empirical Model-Based Performance Prediction for Application Mapping on Multicore Architectures. *Journal of Systems Architecture*, June 2019. doi: 10.1016/j.sysarc.2019.06.001. URL `https://hal-lirmm.ccsd.cnrs.fr/lirmm-02151502`.

[29] Gamatié, A., Devic, G., Sassatelli, G., Bernabovi, S., Naudin, P., and Chapman, M. Towards energy-efficient heterogeneous multicore architectures for edge computing. *IEEE Access*, 7:49474–49491, 2019. doi: 10.1109/ACCESS.2019.2910932. URL `https://doi.org/10.1109/ACCESS.2019.2910932`.

[30] Gamatié, A., Nocua, A., Weloli, J. W., Sassatelli, G., Torres, L., Novo, D., and Robert, M. Emerging NVM Technologies in Main Memory for Energy-Efficient HPC: an Empirical Study. working paper or preprint, May 2019. URL `https://hal-lirmm.ccsd.cnrs.fr/lirmm-02135043`.

[31] HP Labs. McPAT Tool. http://www.hpl.hp.com/research/mcpat/, 2008.

[32] ITRS. International technology roadmap for semiconductors. URL `http://www.itrs.net/`.

[33] Jain, A. and Lin, C. Back to the Future: Leveraging Belady's Algorithm for Improved Cache Replacement. In *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*, pages 78–89. IEEE, 2016.

[34] Kahle, J. A., Day, M. N., Hofstee, H. P., Johns, C. R., Maeurer, T. R., and Shippy, D. Introduction to the cell multiprocessor. *IBM J. Res. Dev.*, 49(4/5):589–604, July 2005. ISSN 0018-8646.

[35] Kologeski, A., Kastensmidt, F. L., Lapotre, V., Gamatié, A., Sassatelli, G., and Todri-Sanial, A. Performance exploration of partially connected 3d nocs under manufacturing variability. In *2014 IEEE 12th International New Circuits and Systems Conference (NEWCAS)*, pages 61–64, June 2014. doi: 10.1109/NEWCAS.2014.6933985.

[36] Komalan, M., Rock, O. H., Hartmann, M., Sakhare, S., Tenllado, C., Gómez, J. I., Kar, G. S., Furnemont, A., Catthoor, F., Senni, S., Novo, D., Gamatié, A., and Torres, L. Main memory organization trade-offs with DRAM and STT-MRAM options based on gem5-nvmain simulation frameworks. In *2018 Design, Automation & Test in Europe Conference & Exhibition, DATE 2018, Dresden, Germany, March 19-23, 2018*, pages 103–108. IEEE, 2018. ISBN 978-3-9819263-0-9. doi: 10.23919/DATE.2018.8341987. URL `https://doi.org/10.23919/DATE.2018.8341987`.

[37] Kumar, R., Tullsen, D. M., Jouppi, N. P., and Ranganathan, P. Heterogeneous chip multiprocessors. *Computer*, 38(11):32–38, November 2005. ISSN 0018-9162. doi: 10.1109/MC.2005.379.

[38] Lattner, C. and Adve, V. S. LLVM: A compilation framework for lifelong program analysis & transf. In *CGO'04*, pages 75–88, 2004.

[39] Lepak, K. M. and Lipasti, M. H. Silent stores for free. In *Proceedings of the 33rd Annual ACM/IEEE International Symposium on Microarchitecture*, MICRO 33, pages 22–31, New York, NY, USA, 2000. ACM. ISBN 1-58113-196-8. doi: 10.1145/360128.360133. URL `http://doi.acm.org/10.1145/360128.360133`.

[40] Matsunaga, A. and Fortes, J. A. B. On the use of machine learning to predict the time and resources consumed by applications. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, CCGRID '10, pages 495–504, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4039-9. doi: 10.1109/CCGRID.2010.98. URL `https://doi.org/10.1109/CCGRID.2010.98`.

[41] Micolet, P.-J., Smith, A., and Dubach, C. A machine learning approach to mapping streaming workloads to dynamic multicore processors. *SIGPLAN Not.*, 51(5):113–122, June 2016. ISSN 0362-1340. doi: 10.1145/2980930.2907951. URL `http://doi.acm.org/10.1145/2980930.2907951`.

[42] Mittal, S. A survey of techniques for architecting and managing asymmetric multicore processors. *ACM Comput. Surv.*, 48(3):45:1–45:38, February 2016. ISSN 0360-0300. doi: 10.1145/2856125.

[43] Mittal, S., Vetter, J. S., and Li, D. A survey of architectural approaches for managing embedded dram and non-volatile on-chip caches. *IEEE TPDS*, 26(6):1524 – 1537, June 2015.

[44] Novaes, M., Petrucci, V., Gamatié, A., and Pereira, F. M. Q. Compiler-assisted adaptive program scheduling in big.little systems. *CoRR*, abs/1903.07038, 2019. URL `http://arxiv.org/abs/1903.07038`.

[45] Novaes, M., Petrucci, V., Gamatié, A., and Pereira, F. Poster: Compiler-assisted adaptive program scheduling in big.little systems. In *24rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '19. ACM, 2019.

[46] Ohba, N. and Takano, K. An soc design methodology using fpgas and embedded microprocessors. In *Proceedings of the 41st Annual Design Automation Conference*, DAC '04, pages 747–752, New York, NY, USA, 2004. ACM. ISBN 1-58113-828-8. doi: 10.1145/996566.996769. URL `http://doi.acm.org/10.1145/996566.996769`.

[47] Péneau, P.-Y. *Integration of emerging non volatile memory in the cache hierarchy for energy-efficiency improvement*. Theses, Université de Montpellier, October 2018. URL `https://tel.archives-ouvertes.fr/tel-01957250`.

[48] Péneau, P.-Y., Novo, D., Bruguier, F., Torres, L., Sassatelli, G., and Gamatié, A. Improving the performance of STT-MRAM LLC through enhanced cache replacement policy. In Berekovic, M., Buchty, R., Hamann, H., Koch, D., and Pionteck, T., editors, *Architecture of Computing Systems - ARCS 2018 - 31st International Conference, Braunschweig, Germany, April 9-12, 2018, Proceedings*, volume 10793 of *Lecture Notes in Computer Science*, pages 168–180. Springer, 2018. ISBN 978-3-319-77609-5. doi: 10.1007/978-3-319-77610-1_13. URL `https://doi.org/10.1007/978-3-319-77610-1_13`.

[49] Pereira, F. M. Q., Leobas, G. V., and Gamatié, A. Static prediction of silent stores. *TACO*, 15(4): 44:1–44:26, 2019. URL `https://dl.acm.org/citation.cfm?id=3280848`.

[50] Poremba, M., Zhang, T., and Xie, Y. Nvmain 2.0: A user-friendly memory simulator to model (non-)volatile memory systems. *IEEE Computer Architecture Letters*, 14(2):140–143, July 2015. ISSN 1556-6056. doi: 10.1109/LCA.2015.2402435.

[51] Péneau, P., Bouziane, R., Gamatié, A., Rohou, E., Bruguier, F., Sassatelli, G., Torres, L., and Senni, S. Loop optimization in presence of stt-mram caches: A study of performance-energy tradeoffs. In *2016 26th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, pages 162–169, Sep. 2016. doi: 10.1109/PATMOS.2016.7833682.

[52] Péneau, P.-Y., Novo, D., Bruguier, F., Sassatelli, G., and Gamatié, A. Performance and energy assessment of last-level cache replacement policies. In *2017 First International Conference on Embedded Distributed Systems (EDiS)*, pages 1–6, Dec 2017. doi: 10.1109/EDIS.2017.8284032.

[53] Quadri, I. R., Gamatié, A., Boulet, P., and Dekeyser, J.-L. Modeling of Configurations for Embedded System Implementations in MARTE. In *1st workshop on Model Based Engineering for Embedded Systems Design - Design, Automation and Test in Europe (DATE 2010)*, Dresden, Germany, March 2010. URL `https://hal.inria.fr/inria-00486845`.

[54] Ramos, P., Mendonca, G. S. D., Soares, D., Araújo, G., and Pereira, F. M. Q. Automatic annotation of tasks in structured code. In Evripidou, S., Stenström, P., and O'Boyle, M. F. P., editors, *Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques, PACT 2018, Limassol, Cyprus, November 01-04, 2018*, pages 31:1–31:13. ACM, 2018. doi: 10.1145/3243176.3243200. URL `https://doi.org/10.1145/3243176.3243200`.

[55] SEMICO. Semico research corporation. URL `http://www.semico.com/`.

[56] Senni, S., Brum, R. M., Torres, L., Sassatelli, G., Gamatié, A., and Mussard, B. Potential applications based on nvm emerging technologies. In *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1012–1017, March 2015. doi: 10.7873/DATE.2015.1120.

[57] Senni, S., Torres, L., Sassatelli, G., Gamatié, A., and Mussard, B. Emerging non-volatile memory technologies exploration flow for processor architecture. In *2015 IEEE Computer Society Annual Symposium on VLSI*, pages 460–460, July 2015. doi: 10.1109/ISVLSI.2015.126.

[58] Senni, S., Torres, L., Sassatelli, G., Gamatié, A., and Mussard, B. Exploring mram technologies for energy efficient systems-on-chip. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 6(3):279–292, Sep. 2016. ISSN 2156-3357. doi: 10.1109/JETCAS.2016.2547680.

[59] Senni, S., Delobelle, T., Coi, O., Peneau, P., Torres, L., Gamatié, A., Benoit, P., and Sassatelli, G. Embedded systems to high performance computing using stt-mram. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, pages 536–541, March 2017. doi: 10.23919/DATE.2017.7927046.

[60] Singh, A. K., Shafique, M., Kumar, A., and Henkel, J. Mapping on multi/many-core systems: Survey of current and emerging trends. In *Design Automation Conference*, pages 1:1–1:10, 2013. ISBN 978-1-4503-2071-9.

[61] Singh, A. K., Leech, C., Reddy, B. K., Al-Hashimi, B. M., and Merrett, G. V. Learning-based run-time power and energy management of multi/many-core systems: Current and future trends. *J. Low Power Electronics*, 13(3):310–325, 2017.

[62] Stefanov, T., Pimentel, A., and Nikolov, H. Daedalus: System-level design methodology for streaming multiprocessor embedded systems on chips. *Handbook of Hardware/Software Codesign*, pages 1–36, 2017.

[63] Vardhan, K. A. and Srikant, Y. Exploiting Critical Data Regions to Reduce Data Cache Energy Consumption. In *Proceedings of the 17th International Workshop on Software and Compilers for Embedded Systems*, pages 69–78. ACM, 2014.

[64] Wilhelm, R., Engblom, J., Ermedahl, A., Holsti, N., Thesing, S., Whalley, D., Bernat, G., Ferdinand, C., Heckmann, R., Mitra, T., Mueller, F., Puaut, I., Puschner, P., Staschulat, J., and Stenström, P. The worst-case execution-time problem&mdash;overview of methods and survey of tools. *ACM Trans. Embed. Comput. Syst.*, 7(3):36:1–36:53, May 2008. ISSN 1539-9087. doi: 10.1145/1347375.1347389. URL `http://doi.acm.org/10.1145/1347375.1347389`.

[65] Yu, H., Gamatié, A., Rutten, E., and Dekeyser, J.-L. Safe design of high-performance embedded systems in an mde framework. *Innovations in Systems and Software Engineering (ISSE), A NASA Journa*, 4(3), 2008.

[66] Zhu, X. Y., Geilen, M., Basten, T., and Stuijk, S. Multiconstraint static scheduling of synchronous dataflow graphs via retiming and unfolding. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(6):905–918, June 2016.