System Scenario-Based Design of the Last-Level Cache in Advanced Interconnect-Dominant Technology Nodes

MAHTA MAYAHINIA, Department of Computer Science, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
TOMMASO MARINELLI, imec, Leuven, Belgium
ZHENLIN PEI, The University of Texas at Arlington, Arlington, United States
HSIAO-HSUAN LIU, Katholieke Universiteit Leuven, Leuven, Belgium and imec, Leuven, Belgium
CHENYUN PAN, The University of Texas at Arlington, Arlington, United States
ZSOLT TOKEI, imec, Leuven, Belgium
FRANCKY CATTHOOR, Microlab, National Technical University of Athens, Athens, Greece
MEHDI TAHOORI, Department of Computer Science, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany and imec, Leuven, Belgium

Feature size reduction of the front End of the Line (FEoL) and back End of the Line (BEoL) elements, i.e., transistors and interconnects, has been the main enabler of the next-generation computation systems. The decreasing trend of the cross-sectional area of the interconnect in advanced technology nodes, however, comes along with a drastic increase in the resistive parasitic, substantially impacting the overall energy efficiency and performance of the computer system. Mitigation of the high parasitic resistance within an advanced-node static RAM (SRAM)-based last-level cache (LLC) is the main target of this article. To achieve this target, we augment the LLC interconnect with some degree of reconfiguration by utilizing a dynamic segmented bus (DSB). With DSB, the interconnect segments that are most actively used for a given workload can be shortened, on average, contributing to a smaller capacitive load. Hence, the efficient reconfiguration of an LLC interconnect strongly depends on the LLC demands of the application. To account for this workload dependency, we design the required microarchitectural support in an end-to-end application-to-technology flow. By optimizing the overhead of DSB switches and additional hardware modules, the SRAM-based LLC with DSB-augmented intra-macro interconnect achieves 33% energy savings and 16% reduction in total access time across eight representative workloads, with a negligible area overhead of less than 0.4%.

CCS Concepts: • **Hardware → Static memory;**

Additional Key Words and Phrases: Interconnect dominance, Last-level cache (LLC), System technology co-optimization (STCO), system scenario-based design, Dynamic segmented bus (DSB), energy saving

Authors' Contact Information: Mahta Mayahinia, Department of Computer Science, Karlsruhe Institute of Technology (KIT), Karlsruhe, Baden Württemberg, Germany; e-mail: mahta.mayahinia@kit.edu; Tommaso Marinelli, imec, Leuven, Flanders, Belgium; e-mail: Tommaso.Marinelli@imec.be; Zhenlin Pei, The University of Texas at Arlington, Arlington, Texas, United States; e-mail: zhenlin.pei@mavs.uta.edu; Hsiao-Hsuan Liu, Katholieke Universiteit Leuven, Leuven, Flanders, Belgium and imec, Leuven, Flanders, Belgium; e-mail: Samantha.Liu@imec.be; Chenyun Pan, The University of Texas at Arlington, Arlington, Texas, United States; e-mail: chenyun.pan@uta.edu; Zsolt Tokei, imec, Leuven, Flanders, Belgium; e-mail: zsolt.tokei@imec.be; Francky Catthoor, Microlab, National Technical University of Athens, Athens, Attica, Greece; e-mail: catthoor@microlab.ntua.gr; Mehdi Tahoori, Department of Computer Science, Karlsruhe Institute of Technology (KIT), Karlsruhe, Baden Württemberg, Germany and imec, Leuven, Flanders, Belgium; e-mail: mehdi.tahoori@kit.edu.

ACM 1539-9087/2025/09-ART117
https://doi.org/10.1145/3762649

ACM Reference Format:
Mahta Mayahinia, Tommaso Marinelli, Zhenlin Pei, Hsiao-Hsuan Liu, Chenyun Pan, Zsolt Tokei, Francky Catthoor, and Mehdi Tahoori. 2025. System Scenario-Based Design of the Last-Level Cache in Advanced Interconnect-Dominant Technology Nodes. *ACM Trans. Embedd. Comput. Syst.* 24, 5s, Article 117 (September 2025), 26 pages. https://doi.org/10.1145/3762649

1 Introduction

Improving computer systems' performance and energy efficiency has always been a target in designing the new programmable processor generation [2]. Increasing the number of cores, the size of the **last-level cache (LLC)**, and the main memory are among the notable factors that improve the performance. The feasibility of these factors is mostly due to the higher integration density driven by Moore's law feature size reduction of the technology elements.

In advanced technology nodes (sub-10 nm), however, the small cross-sectional area of the **back End of the Line (BEoL)** interconnect leads to a significant increase in the resistive parasitic, impeding the higher performance and energy efficiency. In the context of the programmable processor, higher integration density leads to higher processing capabilities through the utilization of multi-core. To accommodate this, one prerequisite is to increase the size of the **static RAM (SRAM)**-based LLC, further exacerbating the interconnect dominance issue. Due to this cross-layer dependency, unlocking the scaling in advanced technology nodes is typically achieved through **system technology co-optimization (STCO)**, an end-to-end approach in which the key parameters of a system from the application all the way to the technology, are co-optimized together.

An SRAM-based LLC is comprised of multiple interconnected banks, independently functional memory units. Furthermore, each bank is comprised of multiple mats, and each mat is comprised of multiple subarrays. The impact of the intra-bank interconnect fabricated on the lower metal layers of M1–M4 is mostly on the noise margin, as comprehensively analyzed in [13, 21, 22, 30]. The inter-bank interconnect is fabricated on the highest metal layers of M5–M8 and substantially affects the overall energy consumption [26–28]. The inter-bank interconnect is the main focus of this work.

The bank-level energy consumption and access latency, dominated by the inter-bank interconnect, are asymmetric and depend on the distance between the bank and the root access pin. The high resistive parasitic in advanced technology nodes further pronounces this distance-dependent asymmetry; the closer the bank is to the root access pin, the lower the energy consumption and the shorter the access time. Breaking the high capacitive load of the inter-bank interconnection network using a **dynamic segmented bus (DSB)** is a promising approach introduced in [23]. The utilization of the DSB, however, is not exclusive to the memory structure, and its effectiveness in breaking the high interconnect's capacitive load has been reported in a vast number of functionalities, such as [5, 7, 11, 12, 14].

The core idea behind the substitution of the DSB-based interconnection for the conventional E-tree is providing the reconfiguration for the LLC. This way, based on the capacity required by the running application, only the interconnect segments corresponding to an adequate number of closest banks (to the root access pin) are activated, and the rest remain deactivated. With this reconfiguration, LLC converges to an ideal memory model; quite large with quite small energy consumption.

Design technology co-optimization (DTCO)-based incorporation of the DSB, i.e., its circuit-level implementation details, floorplanning, and control schemes, has been discussed in [23]. However, key STCO aspects remain unaddressed, which include a methodology for identifying feasible DSB reconfigurations, microarchitectural support for detecting and switching to the optimal

per-workload configuration, and strategies for estimating and mitigating the associated overhead. These are the gaps that this work aims to fill.

Identifying the optimal reconfiguration can take place with two extreme approaches, offline and online. In offline approaches, the optimal reconfiguration is determined prior to the workload execution, meaning the switching to the reconfiguration cannot take place during runtime. Although the offline reconfiguration is the most optimal, its prior determination and hence, lack of adaptability to the dynamic characteristics of a workload, induce a significant cost to the system. In contrast, online approaches allow the system configuration to adapt dynamically based on the workload through a lightweight process. Therefore, the achieved reconfiguration cannot always converge to the near Pareto-optimal reconfiguration of the offline approach.

A hybrid combination of the offline and online, also known as system scenario-based design, stands as a complementary approach [6]. In system scenario-based design, with a hybrid fashion, in the first step, the possible Pareto-optimal system scenarios are identified offline. Furthermore, during the runtime, through a lightweight approach, an optimized system scenario is detected, and finally, the configuration of the system is switched.

This article focuses on enhancing the energy efficiency and performance of an SRAM-based LLC by augmenting its inter-bank (intra-macro) interconnect with DSB switches. To achieve this objective, we adopt a system scenario-based approach. Within this approach, our article has the following key contributions:

— We apply the concept of STCO to the design of SRAM-based LLC in advanced technology nodes through end-to-end co-exploration and co-optimization of system-level abstractions, from the application down to the technology level.
— We conduct a comprehensive end-to-end offline analysis to identify Pareto-optimized system configurations and cluster them to obtain the system scenarios, i.e., the optimal activation patterns of the DSB switches, while accounting for the constraints of the LLC hardware.
— We propose a lightweight online mechanism to detect the optimized system scenario for each application phase, leveraging information from the **performance counter module (PCM)** during a limited monitoring window. This mechanism comprises: a counter to track the monitoring window, a buffer to temporarily store unserviced writes during this period, a random forest regressor to predict the optimal system scenario, and finally, a lookup table to determine the programming of the DSB switches accordingly.

By utilizing the discussed system scenario-based approach, the uniquely accessed addresses can be packed into the minimum number of innermost banks. Simultaneously, interconnect segments associated with unused outer banks can be deactivated. The proposed scenario-based approach allows the SRAM-based LLC to achieve approximately 33% energy savings and a 16% reduction in overall access time, with an area overhead of less than 0.4%.

The rest of this article is organized as follows. In Section 2, we review the motivation and existing ideas regarding the reconfigurable LLC. In Section 3, we elaborate on the core design of the system scenario-based LLC, followed by Section 4, which quantitatively assesses the effectiveness of our proposed methods and compares them against the related works. Finally, Section 5 concludes the article.

2 Reconfigurable Cache

In this chapter, we quantitatively assess the impact of the interconnect dominance in the large cache memories. Furthermore, we provide the background information on how the DSB structure enables the reconfiguration. We also provide an overview of different approaches regarding the reconfigurable cache in the literature.

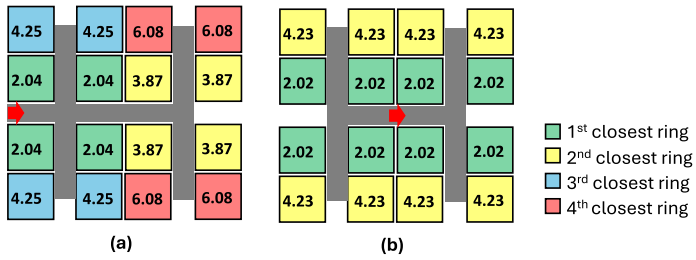

Fig. 1. A 16-bank SRAM-based memory macro fabricated in 16 nm critical wire dimension (CD), the numbers inside the banks (in mm) show the distance between the bank and the root access pin, for (a) side-pin and (b) center-pin access, the red arrow indicates the position of the root access pin and the gray line shows the E-Tree [28]

Table 1. Comparison of the Energy and Latency of a 128 MB SRAM-based Cache, Organized in 16 Banks, and a Data Width of 512 Bits

CMOS Interconnect	RC delay (Interconnect)	Optimum Latency [ns/mm]	Optimum Energy [pJ/mm]	Cycle Time [ns]	Farthest bank		Closest bank	
					Latency	Energy	Latency	Energy
- 90 nm - 90 nm CD	~20 ps/mm	0.14	392	15.56	5.39 ns 1 Cycle	59.3 nJ	43.8 ps 1 Cycle	44.1 nJ
- 40 nm - 40 nm CD	~80 ps/mm	0.17	206	10.72	2.97 ns 1 Cycle	14.1 nJ	122 ps 1 Cycle	10.7 nJ
- 32 nm - 32 nm CD	~200 ps/mm	0.19	179	9.39	2.68 ns 1 Cycle	9.82 nJ	172 ps 1 Cycle	7.94 nJ
- 22 nm - 22 nm CD	~500 ps/mm	0.22	143	6.29	2.07 ns 1 Cycle	5.32 nJ	218 ps 1 Cycle	4.14 nJ
- 3 nm - 16 nm CD	~80 ns/mm	1.11	88	0.33	6.74 ns 21 Cycles	0.57 nJ	2.26 ns 7 Cycles	0.22 nJ

2.1 Large Interconnect-dominated LLC

The capacity of the LLC is higher than faster-level caches, e.g., L1 and L2. To maintain the modularity in the design of a large SRAM-based LLC, the macro can be realized by the instantiation of multiple sub-modules known as banks. Figure 1 shows two floorplannings for interconnecting multiple banks and the position of the root access pin [28].

2.1.1 Distance Dependency of Bank-level Access. Stemmed from a high interconnect's **resistive-capacitive (RC)** delay at the technology level, the latency and energy for accessing the different banks are asymmetric and depend on the distance of the bank to the root access pin. Table 1 presents bank-level disparities across technology nodes, based on results from the CACTI framework (90–22 nm) and its extended version, CACTI++, for advanced nodes such as 3 nm [26, 28]. Until the 22 nm technology node, the asymmetry of the access latency for the closest and farthest banks was not propagated to the architecture level. Also, from the access energy point of view, until the 22 nm technology node, despite the asymmetry, the energy of the accesses to the closest and farthest banks did not show a significant difference. However, in the 3 nm CMOS node corresponding to 16 nm **critical wire dimension (CD)** interconnect technology, the access latency and energy for

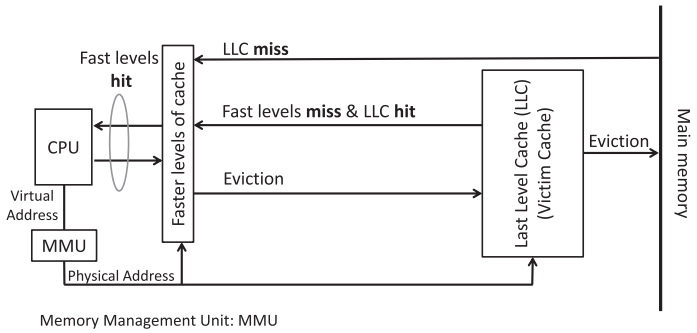

Fig. 2. LLC (as a victim cache) interactions with the processor, **memory management unit (MMU)**, faster cache levels, and the main memory, here, we assume a physically indexed, physically tagged L1 cache.

the closest and farthest banks show a significant asymmetry. This trend of the interconnect impact is also the case for beyond 3 nm technology. In these advanced technology nodes, the DSB-based reconfiguration can leverage this inherent asymmetry to reach a higher performance and energy efficiency.

2.1.2 Victim Cache. In the scope of this article, we assume that the LLC works as a victim cache, containing evicted data from the faster levels. In the event of a miss in any of the faster-level caches, the victim cache is searched first. In the case of the availability of the requested block in the victim cache, the processor can access the block faster than fetching it from the main memory. Figure 2 shows the interactions of the victim cache with the processor, MMU, faster-level caches (with an assumption of physically indexed, physically tagged L1), as well as the main memory.

2.2 Segmented Bus Utilization

The well-established concept of the segmented bus can work as a remedy for the high resistive parasitic of the interconnect in advanced nodes. The idea of the segmented bus is to divide an entire bus into multiple segments, reducing the active capacitive load of the interconnect and consequently lowering the switching energy [7]. Additionally, the role of physical floorplanning in the energy reduction of the communication systems is shown in [11, 12]. To further optimize the floorplanning, the authors of [11, 12] also suggest DSB, in which the length of the interconnect segments is not necessarily equal to each other. Moreover, for effective control of the DSB, a software-controlled fashion is proposed by [14]. More recently, the DSB utilization for the global synapse communication in a neuromorphic fabric has also been proposed by [5].

The utilization of the DSB in various domains motivates the researchers to leverage this structure as the inter-bank interconnect of a large advanced-node LLC as introduced in [23]. However, applying STCO to DSB-based reconfigurations requires the microarchitectural support of the detection and switching of the reconfiguration, as well as quantifying and optimizing the induced overhead. These are the essential details that have not been considered in [23], which focused only on the DTCO details.

2.3 Related Work on Reconfigurable Cache Memory

There are several existing ideas on the reconfigurable cache structure. These ideas can be classified into three main categories: 1. Offline methods, 2. System scenario-based methods, and 3. Online methods. The proximity to Pareto-optimal and flexibility are the two factors that vary among these three categories.

In offline methods, the cache utilization of a workload is carefully characterized in advance. Therefore, the obtained cache configuration is tailored for that workload. The primary drawback of offline cache reconfiguration approaches is their inability to adapt to dynamic workloads. On the other side of the spectrum, there are online methods. During runtime, cache configurations can be switched online. Such reconfigurations are made with limited data, so the obtained cache configuration may be far from the Pareto-optimum configuration. However, the key advantage of online methods is their flexible adaptability to dynamic workloads.

Owning the benefits of both the offline and online methods, i.e., achieving a highly optimized workload-adaptive cache configuration through a lightweight procedure, is the promise of the system scenario-based methods. In a hybrid fashion, at the design time, the optimized reconfigurations of the system, i.e., system scenarios, are identified with an end-to-end application to technology investigation approach. During the run-time of an application, the most optimal system scenario is detected through the system-level monitoring. Finally, the target module (in this article, LLC) is augmented with low-overhead switching methods. By assuming roughly a 10% distinction between the system scenarios, considering a few tens of them can sufficiently cover the entire design space [6]. Moreover, the fact that the pre-selected system scenarios are among the Pareto-optimized ones guarantees that the detected system scenario is close enough to the optimized cache configuration that would be determined using offline methods.

2.3.1 Offline LLC Reconfiguration. The work presented in [18] aimed to estimate the hit rates of the instruction cache by modeling the cache states at each point of the **control flow graph (CFG)**. The other work presented in [17] introduced the temporal usage profile to achieve instruction cache locking. Both [17, 18] can be categorized as offline methods. Since finding the workload-tailored reconfiguration typically requires exhaustive instrumentation, these offline methods cannot account for the dynamic workload behaviors.

2.3.2 System Scenario-based LLC Reconfiguration. The work presented in [25] utilized a system scenario-based approach. In [25], the dynamic instructions of an application are used as the input to a **machine learning (ML)** model, trained to predict an optimal cache configuration from a number of already defined optimal system scenarios. As another system scenario-based method, authors in [10] proposed putting the unused cache lines in low-power modes by reducing the voltage. These low-power modes are selected from already-optimized power setups.

2.3.3 Online LLC Reconfiguration. The work presented in [4] discussed an online fashion for disabling the cache ways during modest cache activities and re-enabling them in the case of more cache-intensive periods. Instead of disablement, authors in [24] suggested the concatenation of cache ways when applications show higher cache intensity. As another online method, introducing the dynamic fetch size was discussed in [16]. Proposing an adaptive looping cache was the main contribution of [29] and can be categorized as another online method for cache reconfiguration.

In addition, a closely related concept is the **non-uniform cache access (NUCA)** [15], which, similar to our work, is primarily motivated by mitigating the interconnect dominance in cache structures. The core idea in [15] is to integrate a **network on chip (NoC)** within the cache to interconnect the banks, enabling a single-level cache organization. Below, we summarize the key methods proposed in [15]:

— **Static NUCA-1 (S-NUCA-1)**: In this method, the root node connects to each cache bank via a dedicated private channel, forming a star topology. Data-to-bank mapping is determined solely by specific bits of the address. However, the star topology introduces significant interconnect overhead and inherently limits scalability, making the restricted number of banks a primary bottleneck in S-NUCA-1.

— **Static NUCA-2 (S-NUCA-2)**: In this method, the star topology used in S-NUCA-1 is replaced by a two-dimensional mesh network to enhance scalability. However, from a data mapping perspective, S-NUCA-2 adopts the same strategy as S-NUCA-1, relying on fixed address bits to determine data mapping.
— **Dynamic NUCA (D-NUCA)**: Unlike S-NUCA-1 and S-NUCA-2, D-NUCA supports dynamic data migration across banks, allowing frequently accessed addresses to be promoted to banks closer to the root, while less accessed addresses can be demoted to farther banks.

Since all NUCA approaches make decisions at runtime based on limited data, they are classified as online techniques. The primary distinction between our work and NUCA is that, to avoid the high cost of NoC utilization, we augment the inter-bank interconnect in a large LLC with DSB switches. Nevertheless, NUCA represents a key related approach and serves as an important baseline for evaluating our system scenario-based LLC design. So, a detailed comparison between the NUCA and our proposed system scenario-based design is provided in Section 4.6. In principle, although online methods possess workload adaptability, the online-achieved reconfiguration may not be near Pareto-optimal.

3 System-scenario-base LLC Design

To exploit asymmetric access energies in an SRAM-based LLC fabricated in advanced technology nodes, our proposed method follows this intuition. As shown in Figure 1, the LLC is split into multiple *rings*, each containing a group of banks at an equal distance from the root access pin [23]. Workload accesses are packed in the inner rings (closest to the root access pin) to minimize the energy consumption and overall access latency, while interconnect segments corresponding to unused outer rings are deactivated. Additionally, unlike D-NUCA, data is mapped to the appropriate ring only once, eliminating unnecessary data movement across the rings. To implement this intuition, we adopt the system scenario-based design approach, which consists of three key steps: 1. System scenario identification, 2. System scenario detection, and 3. System scenario switching. The details of each step vary depending on the design context.

In the context of an SRAM-based LLC with asymmetric access energies and latencies, system scenario identification is performed offline. By considering energy as the design objective, this process models the LLC's energy consumption across various application-to-technology design choices to determine the optimal pair of application-dependent required LLC size (S^{req}_{LLC}) and LLC energy (E_{LLC}), i.e., (S^{req}_{LLC}, E_{LLC}). These pairs are then clustered to identify sufficiently distinct configurations, i.e., system scenarios. When utilizing DSB, the activation patterns of the rings and banks provide the S^{req}_{LLC} and represent these system scenarios. System scenario detection, however, occurs online and during application runtime. Using a lightweight approach, a limited number of application features are selected and analyzed to determine which system scenario, i.e., which possible S^{req}_{LLC}, best suits the current workload. Finally, the DSB switches are programmed at runtime to activate the corresponding rings and banks, providing the S^{req}_{LLC}.

The system scenario-based approach combines offline analysis with lightweight online detection to determine the near Pareto-optimal reconfiguration. This ensures that the system converges to a tailored configuration while keeping runtime overhead minimal and efficient. Figure 3 illustrates the adaptation steps of the system scenario-based design for an SRAM-based LLC with asymmetric access energies and latencies. The following sections provide more details on each step.

3.1 Identifying the LLC System Scenarios

To obtain the optimized system scenarios, i.e., the activation patterns of the DSB switches providing the S^{req}_{LLC}, design space exploration is required. In an end-to-end approach, we have the following

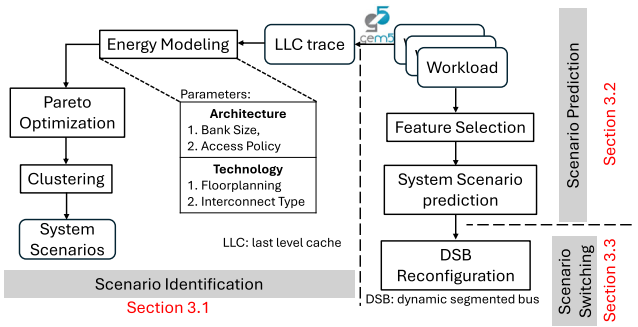

Fig. 3. Overview of the required steps for system scenario-based design.

design choices at each level of system abstraction, namely, application, architecture, as well as physical floorplanning and technology.

3.1.1 Application-level Design Choices. At the application level, we simulate several realistic workloads from benchmarks such as the SPEC CPU 2017 suite using gem5, an event-driven cycle-accurate microarchitecture simulator. Furthermore, we collect the LLC traces for each workload. The per-workload LLC trace contains the information regarding the access time, address, type (read or write), status (hit or miss), and data (in the case of write). These traces include information on the workload-dependent LLC demands that are unique to each workload.

3.1.2 Architecture-level Design Choices. At the architecture level, we consider two choices, bank size and cache access policy. For the bank size, we sweep it in a feasible range of 1 to 8 MB, and for the cache access types, we consider the following policies:

- ***Uniform bank access (UBA):*** In this policy, E-tree has been utilized as the inter-bank interconnect, equalizing the access latency and energy across the banks.
- ***Non-uniform bank access, with even address distribution (NUBAED):*** In this policy, the latency and energy of the access to each bank are not equal and show distance dependability; the closer to the root access pin, the lower the latency and energy. However, the accesses of a workload are evenly distributed among the banks.
- ***Non-uniform bank access, with non-even address distribution (NUBANED):*** In this policy, similar to the NUBAED, the latency and energy of the bank accesses are distance-dependent. However, the distribution of accesses among banks is unequal and depends on the workload. Some banks are accessed more frequently than others.
- ***Non-uniform bank access, with distance-aware address distribution (NUBADAD):*** In this policy, NUBA is still the case; however, by incorporating the impact of the workload, the distribution of the addresses among the banks is distance-aware. This means that the addresses that are accessed more frequently are in the closest banks (to the root access pin), and so on.

UBA, NUBAED, and NUBANED are access policies that treat all banks equally, without preference for any specific one. In contrast, NUBADAD prioritizes certain banks, such as those closer to the root access pin. Selecting these four policies provides a comprehensive representation of access strategies, covering both unbiased and bank-prioritizing approaches.

3.1.3 Floorplanning and Technology-level Design Choices. In the floorplanning, we consider two possibilities for root access: side-pin and center-pin access (see Figure 1). Also, for the technology level, we consider standard copper (Cu)-based and active interconnect. The active

interconnect shares similarities with the concept of three-dimensional (3D) sequential non-bonding stacking architecture, which involves a blanket active layer transfer onto prefabricated bottom devices [20].

Obtaining the Optimized System Scenarios. After obtaining the entire design space, we aim to determine the energy-optimized system scenarios, meaning the S^{req}_{LLC} provided by the activation of the rings and banks, accommodating the entire accesses of the workload. For this, we utilize the concept of Pareto optimization, and for each workload, we determine the Pareto points of the (S^{req}_{LLC}, E_{LLC}). To determine the sufficiently distinct (S^{req}_{LLC}, E_{LLC}) across various workloads, these Pareto points need to be clustered. The outputs of this clustering are associated with the system scenarios (S^{req}_{LLC}, E_{LLC}) that need to be available by reconfiguration. Methods such as the Silhouette score can be used to ensure that the number of clusters, i.e., identified system scenarios, is optimum.

In addition to the Silhouette score, which evaluates clustering from a purely mathematical perspective, hardware-imposed constraints can also influence the number of clusters. For example, the complexity and overhead of system scenario detection and switching increase proportionally with the number of system scenarios. Therefore, these factors must be considered when determining the system scenarios.

3.2 Runtime Detection of the Optimized LLC System Scenarios

The optimized system scenario is identified for each application phase. A typical application's execution consists of periods where memory (here, the LLC) is frequently accessed. We define each of these periods as an *application phase*. An application phase is considered to be changing if the number of no-access cycles exceeds a certain threshold. System scenario detection requires two tasks: 1. Feature selection, and 2. System scenario prediction. For the first task, we extract relevant features from the PCM module, and for the second task, we utilize an ML model with the already selected features as input. The PCM can be implemented as part of the CPUs.[1] Therefore, by the assumption of using such CPUs, feature extraction to detect the optimum system scenario will induce almost no overhead.

3.2.1 Feature Selection Using PCM. PCM provides visibility toward the microarchitecture. In the scope of the LLC, the relevant PCM-provided features are the number of references to the LLC as well as the number of hits and misses. As our main intuition in this work is to pack the workload accesses in the innermost rings, the number of unique accessed addresses is the key piece of information that needs to be predicted.

Feature selection through PCM happens within the *monitoring window*, as shown in Figure 4(a) and Figure 5. At the beginning of each application phase, the aforementioned features from PCM are selected in a duration equal to the monitoring window. Adjusting the monitoring window comes along with a tradeoff. On one side, expanding the monitoring window, i.e., the duration of the feature selection, results in a lower error in the prediction of the number of unique accessed addresses, directly correlating with the selection of the optimized system scenario. On the other side, however, the expansion of the monitoring window induces a latency overhead to the system, which is not desirable. As a practical example, the first phase of workload `602.gcc` (a 100 million-instruction portion of this workload, provided in the SPEC CPU 2017 benchmark suite) consists of 2,683 memory accesses, each accessing a unique address. Consequently, the total number of unique accessed addresses is also 2,683. By monitoring all 2,683 accesses by using a monitoring window of the same duration, the exact number of unique accessed addresses can be determined. Generally, a

[1]For instance, for the Intel CPUs, you can refer to https://github.com/intel/pcm

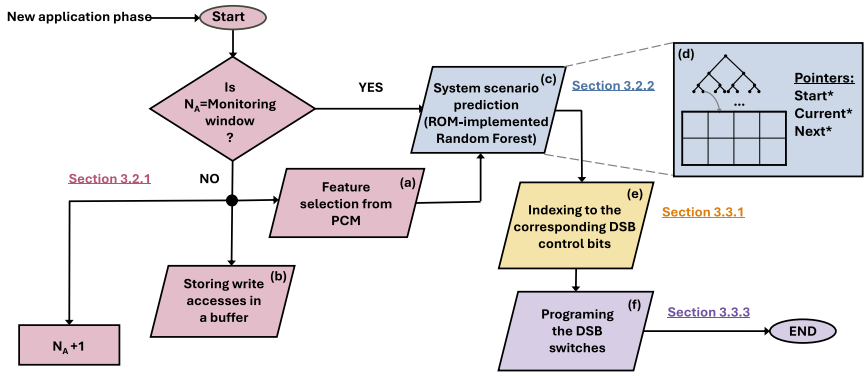

Fig. 4. Flowchart of the required action in a system scenario-based LLC design, N_A and DSB stands for the number of accesses and dynamic segmented bus, respectively.

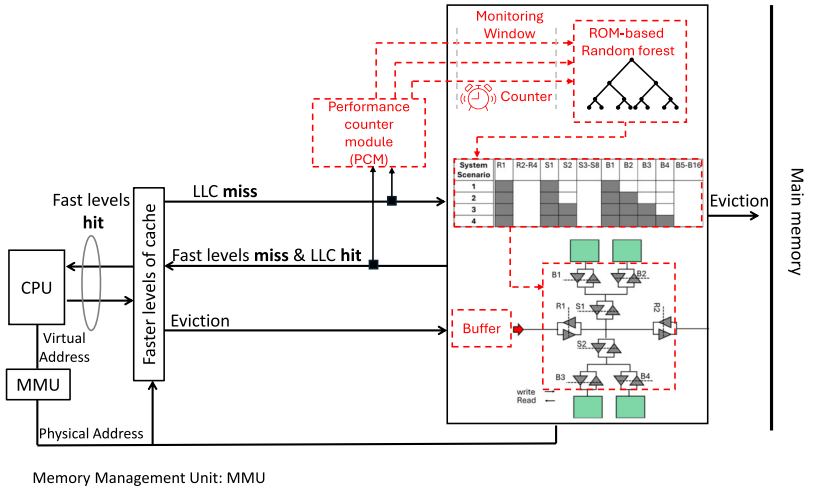

Fig. 5. Block diagram of the system scenario-based LLC design, the red dashed line shows the additional modules for the implementation of the system scenario identification, detection, and switching.

longer monitoring window reduces the prediction error. However, our goal is to achieve sufficiently low-error predictions (e.g., <10% error) of the number of unique accessed addresses using a much shorter monitoring window (e.g., <200).

To determine the optimal monitoring window duration, we set an error threshold for predicting the unique number of accessed addresses, e.g., 10% average across the total phases of various workloads. The shortest monitoring window that meets this error requirement is then selected. At the beginning of each application phase, a simple hardware-based counter is reset and starts counting the number of accesses until it reaches the pre-adjusted monitoring window.

During the monitoring window, the configuration of the LLC is not yet determined; as a result, the write accesses cannot be serviced. To hide the latency overhead due to these unserviced writes, we augment the LLC structure with a buffer as shown in Figure 4(b) and Figure 5. The number of entries for this buffer is equal to the duration of the monitoring window. Therefore, besides the latency, a shorter monitoring window is beneficial for the area and also saves energy. In fact, the duration of the monitoring window and, consequently, the number of buffer entries are

LLC design parameters determined based on the workload characteristics using the discussed end-to-end flow.

3.2.2 System Scenario Prediction Using the Random Forest Regressor Model. Our goal is to enable system scenario prediction, i.e., the number of unique accessed addresses, at application runtime, where low-latency operation is critical. To meet this requirement, the predictor module is designed for hardware implementation within the LLC. Among various ML models, we select the random forest regressor. As a supervised learning method composed of multiple decision trees (illustrated in Figure 4(c)).

The key reason for choosing a random forest regressor is its compatibility with hardware. Unlike other ML models, e.g., support vector machine and neural network, the operations of the random forest regressor are mostly simple comparisons. Additionally, this model can be efficiently implemented using **read-only memory (ROM)**, resulting in a low-cost and low-latency hardware solution. Figure 4 (d) shows the ROM-based implementation of the random forest regressor. Importantly, this approach achieves low-error prediction on par with other ML models while maintaining a feasible hardware implementation.

The number and depth of the trees in the random forest regressor correlate with the hardware cost. Please note that predicting a precise number of unique accessed addresses is not necessary. Instead, we can decrease the granularity of the prediction; i.e., predicting a range for the number of unique accessed addresses to achieve a lower-cost hardware implementation for the random forest regressor. As mandated by STCO, similar to the duration of the monitoring window, the number of trees and their depth in the random forest regressor are also determined and optimized through an end-to-end flow.

3.3 Switching Among the LLC System Scenarios Using DSB

3.3.1 Floor Planning and Control of the DSB. As elaborated in [23], to incorporate the DSB into the inter-bank interconnect network, the LLC macro is split into the following hierarchies: ring, sub-ring, and bank. Rings are a group of banks that are at equal distances from the root access pin. Each ring consists of multiple sub-rings, and consecutively, each sub-ring has multiple banks.

Each bank can be activated only if its corresponding sub-ring and ring are activated. Recursively, there is a path between the root access pin and the bank number N, if and only if there is a connection path between bank $N-1$ and the root access pin. Such a hierarchical organization brings scalability to the DSB employed as the inter-bank interconnection network. Based on the predicted system scenario, activation of the DSB switches can be obtained by having a lookup table, as illustrated in Figure 4(e) and Figure 5. Additionally, this lookup table facilitates the address mapping scheme as elaborated next.

3.3.2 Address Mapping in a System-scenario-based LLC. For each memory access, the first operation performed by the memory management unit (MMU) is the translation of the virtual address generated by the CPU into a physical address. This requires a seamless connection between the CPU and MMU, as illustrated in Figure 2. The size and associativity of the cache determine the number of index and offset bits required for addressing within the cache. Consequently, in the conventional LLC with N banks, $\lceil log_2 N \rceil$ bits of the physical address determine the bank activation. However, in the system scenario-based design, the number of accessed addresses, i.e., the corresponding system scenario, determines the bank activation and, thus, needs to affect the address mapping scheme. This way, the idea is to exploit the activation pattern (through the lookup table) as a partial substitute for the bank address.

Table 2. Simulation Assumptions and Parameters

CPU characteristics	
Instruction set architecture (ISA)	X86 Out-of-Order CPU
Clock Frequency	3 GHz
Physical address	64 b
Cache hierarchy characteristics	
Cache line size	64 B
Level	**Tech. Size Assoc. Access latency (Number of cycles) Number of banks**
L1I/D	SRAM 32 kB 8 3 –
L2	SRAM 512 kB 16 9 4
L3	SRAM 128 MB 16 33 16
CMOS technology	3 nm, nominal VDD = 700 mV
Interconnect characteristics	
Interconnect geometry	Width: 14 nm, Thickness: 28 nm
Optimum energy per unit length per bit	171.8 fJ/mm
Optimum latency per unit length	1.108 ns/mm
Via resistance (R_{via}) [8, 9]	57.6 Ω

3.3.3 Designing the DSB Switches. For the DSB switches, the transmission gate-based structure can be considered [23]. Transmission gate switches consist of a pair of an N and P-type transistors that bidirectionally transfer signals without attenuating their values. As shown in Figure 4(f) and Figure 5, these switches can be programmed to provide the S^{req}_{LLC}. Furthermore, to achieve energy efficiency, by knowing the length of each segment corresponding to the bank, sub-bank, and ring (e.g., from Figure 1), the dimensions of the transmission gate-based switches are tuned to achieve the optimized energy (for details, please refer to [23]). Eventually, Figure 5 shows the block diagram as well as the inter-module interactions of the system scenario-based LLC design.

4 Results and Discussion

In this section, we present our results on the overhead and ultimate effectiveness of the DSB incorporation as the inter-bank interconnect network of an SRAM-based LLC in advanced technology nodes using the parameters outlined in Table 2. We start by demonstrating the effectiveness of the DSB on the scalability of the memory macro. Furthermore, we identify the optimized system scenarios for LLC through an end-to-end flow and continue optimizing the monitoring window, number, and depth of the trees in the random forest regressor ML model. Finally, we show the effectiveness of the DSB incorporation on energy saving, as well as LLC access time reduction, and compare its efficiency against NUCA-based approaches.

Figure 6 illustrates the proportionality of the access energy to the memory size in the 3 nm technology node (corresponding to 16 nm CD for the BEoL interconnect). By leveraging the DSB as the inter-bank interconnection network, the frequent accesses to the memory macro are serviced from the innermost rings. Besides, this policy serves more LLC-intensive work-

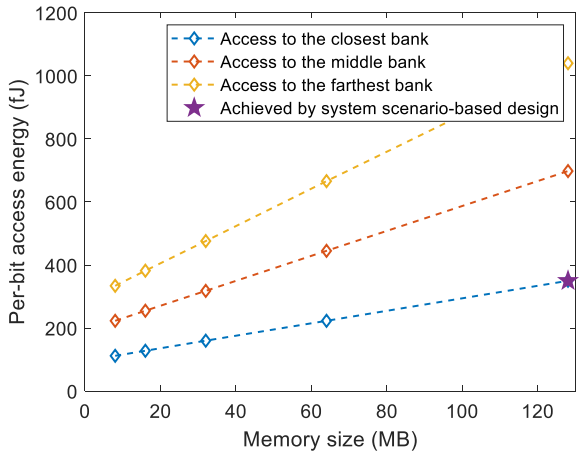

Fig. 6. The proportional increase of the access energy with the memory size in a 16-bank memory, utilizing DSB as the inter-bank interconnection network leads to unlocking the scalability by realizing a large memory size with low energy consumption.

loads by providing their required space. This way, the DSB-based inter-bank interconnection unlocks larger memory sizes at low energy consumption, standing as an effective scalability solution.

4.1 Obtaining the Optimized System Scenarios

4.1.1 End-to-end Application to Memory Floorplan Organization. To obtain the optimized system scenarios, using STCO, we consider an application-to-memory floorplanning approach. In the first step, we simulate the execution of eight representative workloads from the SPEC CPU 2017 benchmark suite, using gem5 system emulation mode with the parameters outlined in Table 2. We pursue two main objectives in the selection of these eight workloads. Firstly, we do not select the workloads that behave too similarly using the indications provided by [33]. Secondly, we aim to mix workloads that stress the memory system as well as more compute-intensive ones [32]. This workload selection methodology ensures that our analysis captures the behavior of a broad spectrum of applications. More specifically, the selection of workload `641.leela` as a deep convolutional neural network workload [1] ensures that emerging **artificial intelligence (AI)** and ML workloads are also represented in our analysis.

Additionally, for those workloads that require input data, there are typically three input data sizes: *test* and *ref*, which have the shortest and longest execution time, respectively, and *train*, which has the execution time between 'test' and 'ref' [19]. In this article, we assume the size of the input set to be 'train'. Besides, simulating these workloads from beginning to end results in an excessive time cost, especially when multiple parameters have to be explored. For this reason, only the most representative interval of 100 million instructions has been simulated for each representative workload, according to the output of the SimPoint utility [31]. A gem5 checkpoint is created for this interval and later restored for successive evaluations. Also, no cache warmup is performed due to the sufficiently large number of instructions.

The output of the gem5 simulations is the LLC access trace. Using these access traces, the total access energy can be determined for each hardware configuration, comprised of bank size, access policy, interconnect, and access type. Figure 7 shows the per-workload energy consumption in different LLC configurations. Moreover, according to Figure 7, typically, Pareto points prefer the

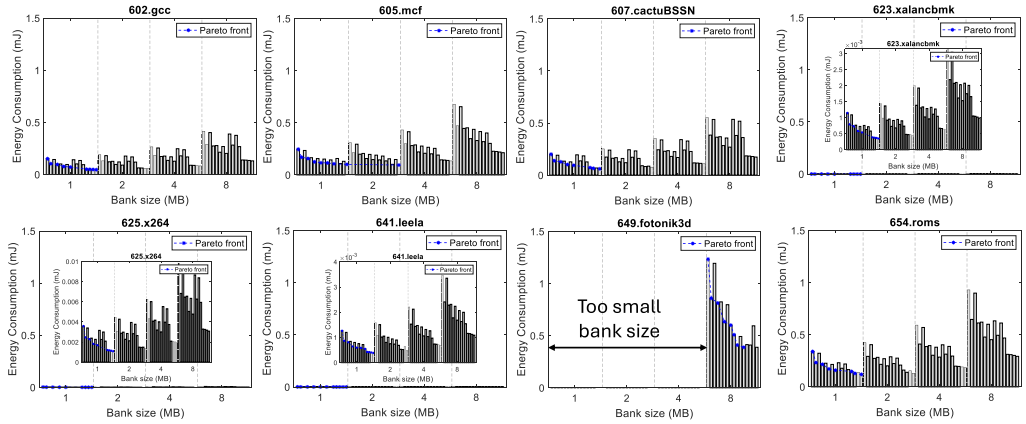

Fig. 7. Per-application energy consumption for different hardware configurations. For each bank size, from left to right, bars are associated with various access policies, interconnect, and access types, as follows: UBA-normal-side, UBA-normal-center, UBA-active-side, UBA-active-center, NUBAED-normal-side, NUBAED-normal-center, NUBAED-active-side, NUBAED-active-center, NUBANED-normal-side, NUBANED-normal-center, NUBANED-active-side, NUBANED-active-center, NUBADAD-normal-side, NUBADAD-normal-center, NUBADAD-active-side, NUBADAD-active-center.

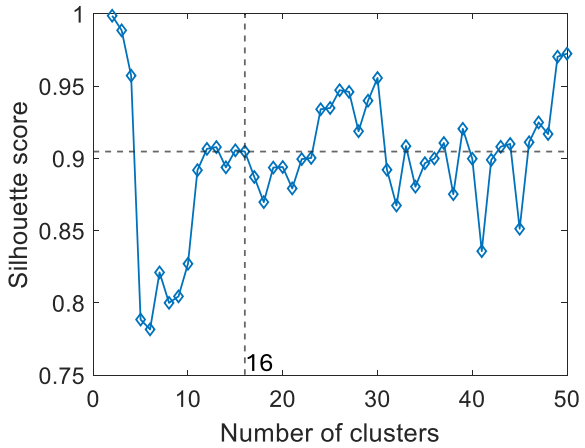

Fig. 8. Silhouette score to determine the optimum number of scenarios, i.e., the possible number of LLC reconfiguration, 16 is the optimum number of system scenarios, taking into account the low overhead scenario switching.

smaller bank sizes. However, this parameter needs to be increased based on workload with the largest number of unique accessed addresses (here, `649.fotonik3d`) to avoid any LLC conflict miss.

4.1.2 Identifying the System Scenarios. As system scenarios include the Pareto-optimum configurations, after finding the per-workload Pareto points (S^{req}_{LLC}, E_{LLC}), the final step is to obtain sufficiently distinct pairs through clustering. The output of such clustering is the system scenarios.

For clustering, we use the K-means method. Also, to evaluate the cohesion and separation of the clusters, we use the concept of the Silhouette score as shown in Figure 8. Silhouette score is a measure to assess the separation between clusters and has a range of [-1, +1]; the closer to +1, the

stronger, i.e., the more distinct the clusters. In the scope of this work, we consider a threshold of 0.9 for the obtained Silhouette score, indicating a strong enough clustering.

However, the Silhouette score is a mathematical concept, and the number of system scenarios needs to be selected by also taking into account other hardware constraints. We split the LLC into ring, sub-ring, and bank. Correspondingly, we consider a transmission gate-based switch for each hierarchy. Therefore, to maintain the hierarchical design, defining the number of system scenarios equal to the number of one of the hierarchy levels, either the number of rings, sub-rings, or banks, reduces the overhead of scenario switching. For instance, in side-pin access (Figure 1(a)), the number of rings, sub-rings, and banks is 4, 8, and 16, respectively. Please note that within the scope of this article, we assume that there is no further reconfiguration within one bank. So, increasing the number of system scenarios beyond the number of banks is impractical.

At this point, we incorporate the Silhouette score to determine the number of system scenarios. According to Figure 8, selecting the number of system scenarios as 16, i.e., equal to the number of banks, satisfies the threshold of 0.9 for the minimum silhouette score. Based on the number of unique accessed addresses, the optimum system scenarios ($\{1..16\}$) can be defined as follows. Applications whose unique accessed addresses can be accommodated by one LLC bank belong to system scenario one, all the way to the applications that require all 16 LLC banks to accommodate their unique addresses, belonging to system scenario 16. The programming of the transmission gate-based switches can be performed using a lookup table. In the case of side-pin access, for each system scenario, 28 switches need to be programmed, which adds up to an overhead of 56 B.

4.1.3 Address Mapping in System Scenario-based LLC. In an LLC with 16 banks, 4 bits of the physical address are required to select the banks, and the determined system scenario needs to be incorporated into the bank address. If an application phase belongs to system scenario N, only N LLC banks are available. To address among these N banks when N is not necessarily a power-of-two, considering $\lceil log_2 N \rceil$ of the **least significant bit (LSB)** of the bank address is sufficient. To showcase in this regard, we can consider two cases: 1. N is a power-of-two, and 2. N is not a power-of-two.

(1) If an application phase belongs to system scenario 1, the address of bank 0 (e.g., `0000`) is substituted for the entire bank address. If it belongs to system scenarios 2, 4, 8, and 16, only 1, 2, 3, and 4 LSB of the bank address are taken into account, and the rest of the **most significant bit (MSB)** become `0`.
(2) If a system scenario is non-power-of-two, referring to a virtually nonexistent bank is the issue. For instance, if the system scenario is 15, all 4 bits of the bank address need to be used. In this case, to service the references to virtually nonexistent bank 1111, we check from bank `0000` to the end (in this example, `1110`) and assign those references to the first non-full bank. Equipping each LLC bank with only a 1-bit flag indicating if the bank is full can facilitate this methodology.

This way, the address mapping procedure of the system-scenario-based LLC does not need any microarchitectural level modification and thus will not induce any further overhead.

4.2 Optimizing the Monitoring Window

To find an optimized duration for the monitoring window, the required input is the LLC access trace. For each phase of the application, we acquire the number of references, hits, and misses ($N^{\text{MW}}_{\text{reference}}$, $N^{\text{MW}}_{\text{hit}}$, $N^{\text{MW}}_{\text{miss}}$) provided by PCM in different durations of the monitoring window. The objective is to predict the number of unique accessed addresses in an entire application phase.

To achieve this objective, for different durations of the monitoring window, we train the random forest regressor model and obtain the error of the prediction. Figure 9 shows the prediction error

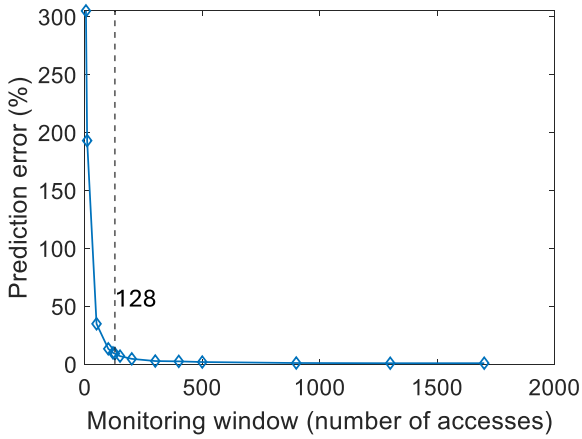

Fig. 9. Prediction accuracy versus duration of the monitoring window (number of monitored accesses), here, 128, positioned on the knee point of the graph, is the optimum duration of the monitoring window.

(average across the total phases of eight representative workloads) versus the duration of the monitoring window.

As discussed in Section 3.2.1, in the case of a longer monitoring window, lower prediction error is achieved at the cost of higher induced cost. To achieve a cost-error compromise, we determine the duration of the monitoring window at the knee point of the graph as illustrated in Figure 9. Besides, tracking the monitoring window requires a counter, for which selecting the monitoring window as a power of two is more efficient. As shown in Figure 9, selecting the monitoring window as 128 can decrease the error to 9.14%, sufficiently close to the optimum while providing a clear saving on the duration of the monitoring window. As discussed in Section 3.2.1, a buffer can be considered on the LLC write path to hide the latency of feature selection. The number of entities for this buffer is also equal to the duration of the monitoring window, and each entity has a length equal to the cache data width as outlined in Table 2.

4.3 Optimizing the Dimensions of the Random Forest Regressor Model

In hardware, the random forest regressor can be implemented using the ROM. Therefore, the number and depth of the tree are crucial factors affecting the hardware cost of the random forest regressor implementation. The depth of the decision tree is highly sensitive to the prediction granularity. Therefore, it is feasible to lower the prediction granularity to achieve hardware efficiency.

As discussed in Section 4.1.2, reconfiguration of the DSB-based inter-bank interconnection network is performed at the granularity of one bank, e.g., 8 MB. Therefore, it is not useful to predict the number of unique access addresses in the finest granularity of one. So, without compromising the prediction error, we keep lowering the granularity of the prediction and monitoring the depth of the random forest regressor model. Figure 10 shows the sensitivity of the depth of the decision tree to the granularity of the prediction. As shown in Figure 10, selecting a prediction granularity of 5k unique accessed addresses reduces the decision tree depth to eight. Notably, this combination (5k granularity, depth of eight) represents the knee point of the graph, where the sensitivity of depth to granularity significantly declines. Lowering the prediction granularity beyond this point has a negligible to no impact on the depth of the decision tree. By considering the cache line size as 64 B (outlined in Table 2), 5k as the unique accessed addresses prediction granularity corresponds to less than 4% of one bank capacity.

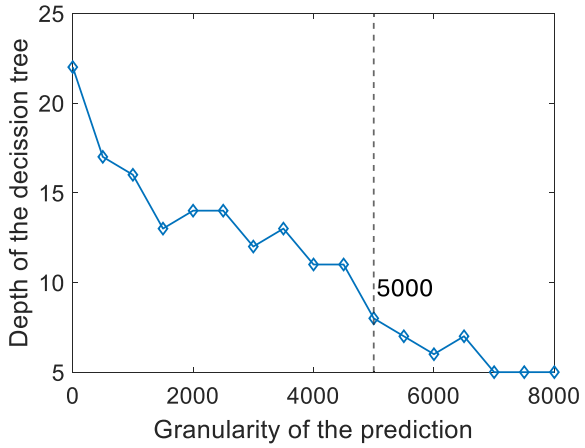

Fig. 10. Depth of the decision tree versus the granularity of the prediction, here, 5k as the granularity of the prediction results in an optimum depth of 8.

4.4 Training the Random Forest Regressor

To construct the dataset for training the optimized random forest regressor, we process per-application LLC traces generated by gem5. From these traces, we extract three key features per application phase: 1. The total number of accesses, 2. The number of misses, and 3. The number of hits. These features are used as inputs, while the number of unique accessed addresses serves as the prediction target.

Feature extraction is performed using a monitoring window of 128 LLC accesses. This constraint means that, within each application phase, the number of accesses, hits, and misses does not exceed 128. Importantly, while the monitoring window size limits the access count per phase, it does not affect the number of unique accessed addresses—the primary value we aim to predict. Since the main idea is achieving a low-enough error prediction of the number of unique accessed addresses during a limited monitoring window. In addition, the prediction granularity for the number of unique accessed addresses is set to 5k. Lastly, the final dataset includes all phases from eight representative applications. We use 80% of the dataset for training and reserve the remaining 20% for testing. Figure 11 summarizes the characteristics of the traces and the resulting training datasets. Table 3 summarizes the parameters of the random forest regressor model and their descriptions, considering 128 as the monitoring window and 5k as the prediction granularity.

4.5 Energy Consumption and LLC Access Time Gain in the Case of DSB Utilization

4.5.1 Discussion. Three workload-dependent factors influence the energy consumption and access time of the LLC when utilizing the DSB as the inter-bank interconnect: 1. The total number of accesses, 2. The number of unique accessed addresses, and 3. The number of application phases. In the following, we elaborate on the impact of these factors on the energy consumption and LLC access time.

— Energy

In a system scenario-based LLC design, the contributions of various energy elements are as follows:

(1) Energy savings result from transferring each bit of data over a shorter distance during each LLC access.

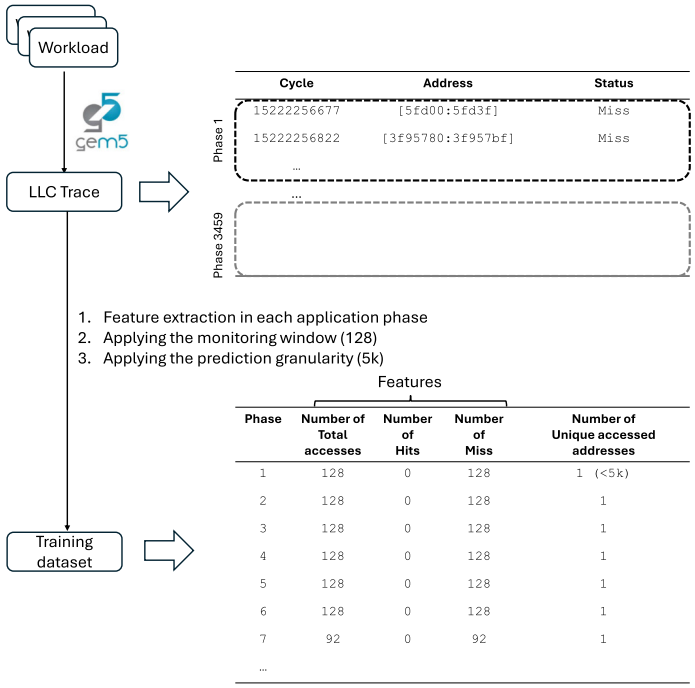

Fig. 11. Details on the generation of the training dataset for the random forest regressor, in this figure, the workload `602.gcc` is selected for showcase.

Table 3. Model Parameters for the Random Forest Regressor, with 128 as the Monitoring Window and 5k as the Prediction Granularity

Parameter	Description	Value
`criterion`	Function used to evaluate the quality of a split	Squared error
`n_estimator`	Number of trees	1
`max_depth`	Maximum tree depth	8
`max_features`	Number of input features	3
`min_samples_split`	Minimum number of samples required to split an internal node	2
`min_samples_leaf`	Minimum number of samples required to be at a leaf node	1
Error	Relative error between the real test value and its prediction: $\lvert\frac{y_{pred}-y_{test}}{y_{test}}\rvert \times 100$	0.14%

(2) The energy overhead arises from **I**. Transferring each bit of data through the DSB switches during every access, and **II**. Detecting system scenarios at the beginning of each application phase.

Among the contributing factors, those that apply to each access have a higher impact. Therefore, among various supporting hardware components—including transmission gate-based DSB switches, a counter, a buffer, a decision tree regressor (implemented as ROM), and a lookup table—the transmission gate DSB switches have the most significant impact due to their frequent per-access activation.

The number of unique accessed addresses determines S^{req}_{LLC}, which is provided by activating the inner rings. A lower number of unique accessed addresses reduces the activated interconnect load, leading to lower energy consumption. By definition, the number of unique accessed addresses is always less than or equal to the total number of accesses.

Table 4. The Overhead Corresponding to the Required Hardware Supporting the Proposed System Scenario-based LLC

Hardware module	Area	Normalized Area (relative to 128 MB cache macro area*)	Energy	Latency
Counter	4.31 um^2	1.93E-07	28.9 fJ	35.4 ps
Buffer	4.15e4 um^2	1.86E-03	4.09 pJ	33.4 ps
Decision Tree	7.76 um^2	3.48E-07	612 fJ	163 ps
Lookup table	1.28 um^2	5.75E-08	38.5 fJ	56.2 ps
Each transmission gate-based switch	2.52 um^2	1.13E-07	133.5 fJ**	2.56 ns

*Normalized to the most area-efficient design, consisting of subarrays with 256 rows and 256 columns, taking into account both data and tag array [26, 27]

**This overhead corresponds to the interconnect length of 2.04 mm as the load

Another key factor is the number of application phases. The switching of the system scenario, if necessary, takes place at the beginning of each application phase. This way, at the beginning of each phase, the optimum system scenario is detected by leveraging the counter, buffer, decision tree regressor, and lookup table. Unlike the transmission gate-based DSB switches, which activate on every access, the other four modules are only triggered once per application phase, reducing their overall energy impact of system scenario switching. In other words, the overall energy consumption is influenced more by the total number of accesses and the number of unique accessed addresses than by the number of application phases.

— LLC access time

Similar to energy consumption, each LLC access incurs latency due to data transfer across interconnect segments and the activation of DSB switches. This latency is directly proportional to the length of the interconnect paths. By placing frequently accessed addresses in the innermost rings, access latency can be minimized, thereby reducing the overall LLC access time. As discussed in Section 3.2.1, a buffer is integrated into the LLC hardware to temporarily store write accesses during the monitoring window dedicated to feature extraction. As a result, the latencies associated with the buffer, the decision tree regressor, and the lookup table are incurred only once per application phase.

4.5.2 Quantitative Analysis of the Area, LLC Access Time, and Energy Overheads. Table 4 presents the hardware overhead associated with implementing the proposed system scenario-based LLC, depicted in Figure 5. As reported in Table 4, the area overhead of the supporting hardware is less than 0.4% of the total LLC macro and thus considered negligible. Furthermore, as quantified by Figure 1(a) and Table 1, the access asymmetry among the banks is more pronounced in the case of side-pin access. Therefore, to showcase the effectiveness of our proposed system scenario-based approach, we consider a bank size of 8 MB in a side-pin LLC structure with a normal (not active) interconnect technology. A bank size of 8 MB provides a sufficiently large LLC size for accommodating all the accesses of an LLC-intensive application, as shown in Figure 7, e.g., for workload `649.fotonik3d`. This way, the number of conflict misses reduces, further enhancing the energy saving and latency reduction. Please note that these parameters are fixed at design time and cannot be changed during runtime. Additionally, extending our methodology to center-pin floorplanning is straightforward. Without altering the overall methodology's outcome, this extension only affects a few modeling parameters, including ring size and bank-level access energy and latency (due to the different bank-root access pin distances).

Table 5 shows the number of unique accessed addresses and corresponding system scenarios for eight representative workloads. Regarding addressability, for the workloads that occupy one or two banks, the address mapping is straightforward since the number of occupied banks is a power of

Table 5. Variety of the Number of Unique Accessed Addresses and Application Phase Characteristics in LLC Traces of Eight Workloads from the SPEC CPU 2017 Benchmark Suite

Workload	Unique accessed addresses (MB)	System Scenario (number of occupied banks)	Number of phases	[Min, Max] length of phase (number of accesses)
602.gcc	3.72	1	3,459	[1,57093]
605.mcf	15.71	2	18,419	[1, 10820]
607.cactusBSSN	8.65	2	32,899	[1, 410]
623.xalanbmk	0.36	1	305	[1, 1223]
625.x264	1.03	1	501	[1, 2219]
641.leela	0.37	1	647	[1, 2057]
649.fotonik3d	112.07	15	47,677	[1, 318]
654.roms	11.50	2	8,607	[1, 24835]

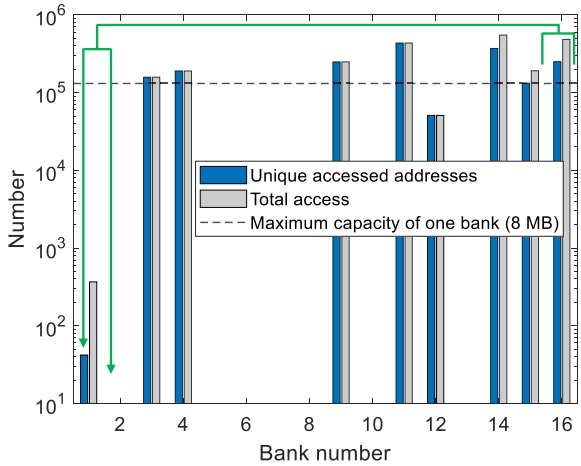

Fig. 12. Distribution of the unique and total accessed addresses among 16-bank LLC, each 8 MB for workload 649.fotonik3d, occupying 15 banks, references to (virtually) nonexistent bank 16 can be accommodated by banks 1 and 2.

two. However, for workload 649.fotonik3d, bank 16 does not (virtually) exist, meaning there is no path between this bank and the root access pin. Therefore, its references need to be mapped to the first non-full bank. Figure 12 shows the per-bank accesses of workload 649.fotonik3d throughout its entire representative execution time. As shown in Figure 12, the accesses to bank 16 can be entirely accommodated by banks 1 and 2. This is because banks 1 and 2 have always had sufficient capacity to accommodate the reference to nonexistent bank 16. For these references, in the worst case, the LLC management needs to check two flag bits to determine the destination bank, which induces negligible overhead.

Ultimately, Figure 13 shows the energy consumption and LLC access time using the conventional E-tree structure and DSB in side-pin LLC. To report the energy and latency overhead induced by our methodology, Figure 13 illustrates the energy consumption and LLC access time in the case of DSB incorporation with two bars. The first bar shows the ideal energy consumption and access time, in which the energy and latency overhead of the supporting hardware, as well as DSB switches, are zero, whereas the second bar shows the realistic energy consumption and access time in which the overhead of the supporting hardware is also considered.

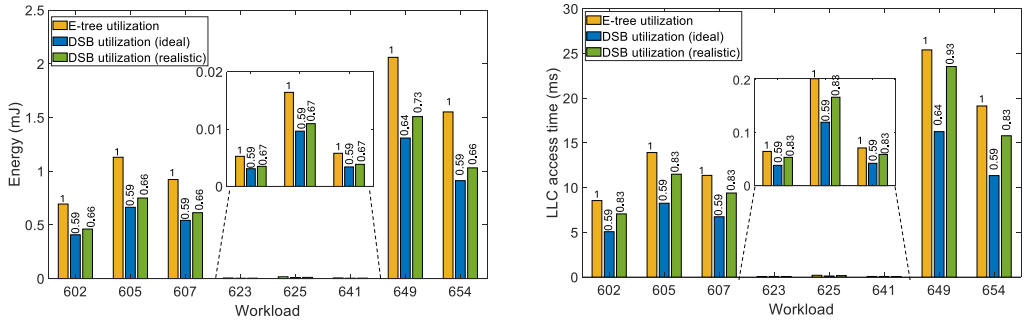

Fig. 13. Per-workload energy (left) and LLC access time (right) gains by utilizing the DSB as the LLC's inter-bank interconnect.

Within the investigated workloads, except for workload `649.fotonik3d`, the rest occupy the innermost ring. Therefore, the achieved energy and performance gain for these workloads is roughly similar, around 34% and 17%, respectively. However, workload `649.fotonik3d` accesses a larger number of unique addresses and requires 15 LLC banks. Due to the utilization of 15 banks, most of the DSB segments are active, which results in a modest energy and performance gain of 27% and 7%, respectively.

Our results demonstrate that integrating the DSB as the inter-bank interconnection network, combined with the system scenario-based design approach, leads to an average energy and LLC access time improvement of approximately 41% across eight representative workloads. While the supporting hardware introduces an energy and latency overhead of about 8% and 25%, the net energy saving and performance gain remain substantial at around 33% and 16%, respectively.

The reason behind this unbalanced overhead lies in an inherent tradeoff between energy and latency. In the context of this work, we initially consider energy as the optimization objective during the system scenario detection. This assumption primarily affects the design of DSB switches. Since they are involved in every LLC access, and consequently, their induced latency and energy overhead are impactful. As the optimization objective in designing the DSB switches is energy [23], the inherent tradeoff between these two system merits implies a higher overhead for the latency. Furthermore, as previously discussed in this Section, the area overhead is less than 0.4%, highlighting the practicality and the efficiency of the proposed solution.

4.5.3 Generalizability of the Proposed System Scenario-based LLC Design. Another piece of information provided by Table 5 is the number, minimum, and maximum length of the application phase. This information clearly shows that the length of application phases is heavily dynamic; it varies significantly not only across the workloads but also within one workload. For instance, workload `649.fotonik3d` consists of 47,677 short-lived phases of $\leq$318. In contrast, though the number of phases for workload `602.gcc` is 3,459, significantly less compared to `649.fotonik3d`, the length of the application phase in this workload can be considerably longer, that is, $\leq$57,093. However, the application phase only concerns the periods of intense burst LLC requests. So, without losing the generality and in the case of parallel execution of multiple applications, such periods of intense burst requests can still be considered as an application phase. This way, the idea of system scenario-based LLC design can be applied to a shared LLC structure across the emerging chiplets and multi-core clusters.

4.5.4 Scalability. The proposed system scenario-based LLC design is scalable. As shown in Table 4, the dominant area contributor is the buffer. The number of buffer entries equals the length

Table 6. The Normalized Area Overhead for Various Macro Sizes

Hardware module	Area	Normalized Area to macro size of*				
		64 MB	128 MB	256 MB	512 MB	1024 MB
Counter	4.31 um^2	3.42E-07	1.93E-07	8.74E-08	4.90E-08	2.22E-08
Buffer	4.15e4 um^2	3.30E-03	1.86E-03	8.41E-04	4.71E-04	2.14E-04
Decision Tree	7.76 um^2	6.17E-07	3.48E-07	1.57E-07	8.82E-08	3.99E-08
Lookup table	1.28 um^2	1.02E-07	5.75E-08	2.60E-08	1.46E-08	6.60E-09
Each transmission gate-based switch	2.52 um^2	2.00E-07	1.13E-07	5.11E-08	2.86E-8	1.30E-8

*Normalized to the most area-efficient design, consisting of subarrays with 256 rows and 256 columns, taking into account both data and tag arrays [26, 27]

of the monitoring window. Also, as discussed in Section 4.2, selecting the length of the monitoring window as a power of two is more hardware compatible. Therefore, even for the larger cache macros, increasing the length of the monitoring window beyond 128, particularly in powers of two, degrades the energy, performance, and area efficiency. Limiting the monitoring window to 128 indicates that the extra area introduced by the system scenario-based design is roughly constant, leading to a reduction of overall area overhead, enabling scalability. Table 6 quantitatively reports the reducing trend of the area overhead in various cache macro sizes, using the area results provided in [27].

4.6 System Scenario-based LLC Design Versus Other Existing Methodologies

Eventually, in this section, we present a comprehensive comparison between the proposed system scenario-based design and several related works discussed in Section 2.3. Given that both our approach and NUCA share the common objective of mitigating interconnect overhead in cache hardware, we place particular emphasis on a detailed comparison between these two methodologies.

4.6.1 Comparison with Offline Methods. The primary distinction between offline and system scenario-based approaches lies in the overhead associated with determining the optimal reconfiguration. Any offline profiling technique capable of estimating the number of uniquely accessed addresses can, in principle, be applied. Furthermore, in our proposed system scenario-based LLC design, we predict this number with a granularity of 5k addresses, striking a balance between error and runtime overhead. Even if offline methods achieve finer-grained predictions, they offer limited practical benefit, since the number of active banks must be configured at the granularity of a full bank.

4.6.2 Comparison with System Scenario-based Methods. The concept of placing non-accessed cache banks into low-power modes, as proposed in [10], can be effective in reducing energy consumption. However, the practicality of such techniques is highly dependent on the underlying technology. For example, in 3 nm technology nodes, lowering the supply voltage (VDD) below the nominal level of 700 mV can lead to circuit instability, making voltage scaling-based low-power modes impractical.

4.6.3 Comparison with Online Methods. The original NUCA [15] is an online cache configuration method designed to mitigate interconnect overhead by interconnecting smaller memory macros through a NoC. To ensure a fair comparison between our proposed system scenario-based approach and NUCA, we first incorporate distance-awareness into the NoC routing algorithm wherever applicable. This adaptation allows for a more meaningful comparison, as it aligns the baseline with our methodology. Consequently, rather than comparing against the original NUCA, which performs online reconfiguration, we evaluate a system scenario-based variant of NUCA that benefits from the determination of the Parto-optimized system scenarios and distance-aware data placement.

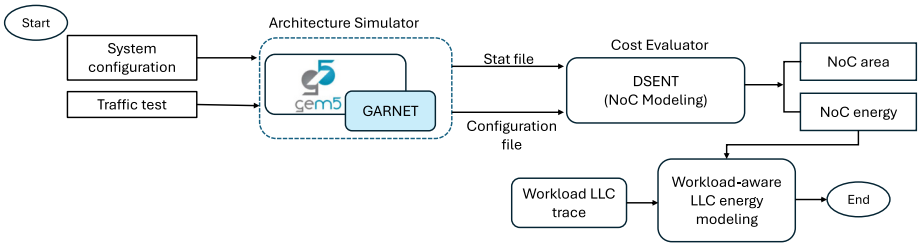

Fig. 14. The flowchart of obtaining the LLC energy in the case of NoC interconnection.

Secondly, we employ the detailed 5-stage GARNET network [3] within the gem5 simulator and generate random traffic using the `ruby_random_test.py` benchmark. The system configuration, including the **instruction set architecture (ISA)**, clock frequency, and multi-level cache sizes, is summarized in Table 2. In our experiments, both the number of NoC nodes and the network topology are treated as variable parameters. Increasing the number of memory macros reduces the size of each macro, affecting overall cache behavior. In terms of topology, we evaluate not only the mesh network originally proposed in [15] but also **point-to-point** (**P2P**) and crossbar configurations. For mesh-based experiments, the number of rows is considered an additional variable to explore different NoC configurations.

After completing microarchitecture-level simulations in gem5 with the various NoC configurations discussed, we feed both the resulting statistics and the full system setup into the DSENT simulator to estimate the energy cost of the NoC-based interconnect [34]. Using the extracted LLC traces from representative workloads in the SPEC CPU 2017 benchmark suite, we model the workload-dependent energy consumption.

Figure 14 illustrates the flowchart for energy modeling when memory macros are interconnected via a NoC. To showcase, the total energy consumption for workload `602.gcc` under different NoC topologies and configurations is presented in Figure 15. The key insights from this analysis are as follows:

— P2P and crossbar topologies result in the highest energy consumption among DSB and mesh-based NoC incorporation.

This is primarily due to their constant latency behavior of O (1), which prevents the use of distance-aware data mapping policies. In other words, in these cases, the topology does not support the integration of a system scenario-based approach. As a result, distance-aware data placement becomes infeasible, leading to increased energy consumption.

— The most energy-efficient configuration is a mesh topology with 16 memory macros (each 8 MB), arranged in 16 rows.

This topology enables effective exploitation of distance-aware mapping, made possible by the adoption of a system scenario-based approach, leading to reduced energy consumption. Furthermore, from the area point of view, in a NoC organization, there is one router per node consisting of a buffer, crossbar, and switch allocator. Using the DSENT framework, the overall area overhead of the aforementioned energy-efficient NoC organization sums up to 8.44e4 μm^2, 8.76% more than the area overhead in the case of DSB incorporation as reported in Table 4. Among multiple per-router elements, the area contribution of the buffer is maximum.

Finally, based on our results and analysis, integrating a NoC offers higher efficiency compared to the conventional E-tree architecture. However, in addition to the higher area overhead associated with the NoC organization due to the per-router buffer, crossbar, and switch allocator, it also falls short of the energy benefits achieved by the proposed DSB-based method, driven by a system

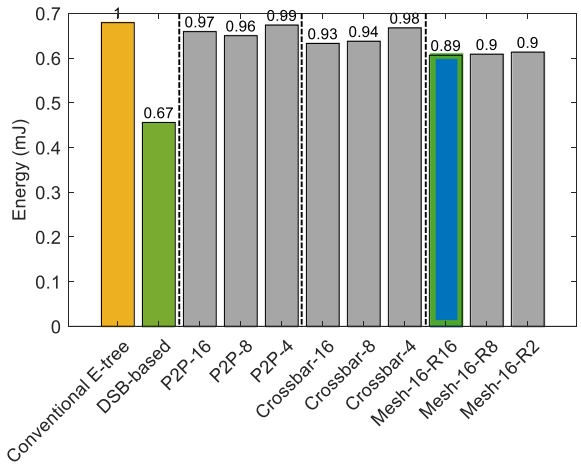

Fig. 15. Energy consumption in different LLC interconnection showcases for workload `602.gcc`, the highest energy efficiency happens for DSB-based design, P2P stands for point-to-point, and the number after the topology refers to the number of nodes, in the mesh topology, the number after 'R' refers to the number of rows, the length of the link for the NoC designs is 3 mm, in the NoC organizations, the highest energy efficiency happened for a 16-node mesh topology with 16 rows.

scenario-based approach. This establishes the DSB-enabled, system scenario-based LLC as a high-performance, energy-efficient solution for building large cache macros in advanced technology nodes.

5 Conclusion

The size reduction of circuit elements driven by Moore's Law has stagnated in advanced technology nodes, limiting further energy efficiency and performance gains. This limitation arises primarily from the drastic increase in resistive parasitic, caused by the reduced cross-sectional interconnect area. However, further energy efficiency and performance in advanced technology nodes can be achieved by utilizing STCO through an application-to-technology co-exploration approach. In this article, we have focused on enhancing the energy efficiency and access time of the SRAM-based LLC by mitigating the high resistive parasitics of its interconnection network using a DSB structure. Inspired by system scenario-based design, we have considered an end-to-end approach to determine the energy-optimized LLC reconfigurations by selectively activating the corresponding banks. Furthermore, in the granularity of an application phase, we have predicted its required LLC size (S_{LLC}^{req}) and, by leveraging the DSB structure, activate only the minimum number of innermost banks, packing all accesses within them. Finally, by optimizing the supporting hardware in an end-to-end flow, we have demonstrated that the proposed system scenario-based LLC achieves approximately 33% of energy improvements and 16% of overall LLC access time reduction across various workloads, with an area overhead of less than 0.4%. The proposed system scenario-based design is scalable and can be applied to heavily dynamic access patterns. Furthermore, this approach can also be adapted to shared LLC across chiplets or multi-core clusters.

References

[1] [n. d.]. Leela. Retrieved from https://www.sjeng.org/leela.html. Accessed: 2025-06-12.
[2] Ravi Bhargava and Kai Troester. 2024. AMD Next-Generation "Zen 4" Core and 4th Gen AMD EPYC Server CPUs. *IEEE Micro* 44, 3 (2024), 8–17.

[3] Niket Agarwal, Tushar Krishna, Li-Shiuan Peh, and Niraj K. Jha. 2009. GARNET: A detailed on-chip network model inside a full-system simulator. In *Proceedings of the 2009 IEEE International Symposium on Performance Analysis of Systems and Software*. 33–42.

[4] David H. Albonesi. 1999. Selective cache ways: On-demand cache resource allocation. In *Proceedings of the 32nd Annual ACM/IEEE International Symposium on Microarchitecture MICRO-32.* IEEE, 248–259.

[5] Adarsha Balaji, Yuefeng Wu, Anup Das, Francky Catthoor, and Siebren Schaafsma. 2019. Exploration of segmented bus as scalable global interconnect for neuromorphic computing. In *Proceedings of the 2019 on Great Lakes Symposium on VLSI*. 495–499.

[6] Francky Catthoor, Twan Basten, Nikolaos Zompakis, Marc Geilen, and Per Gunnar Kjeldsberg. 2020. *System-scenario-based Design Principles and Applications*. Springer.

[7] J. Y. Chen, Wen-Ben Jone, Jinn-Shyan Wang, H-I Lu, and Tien-Fu Chen. 1999. Segmented bus design for low-power systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 7, 1 (1999), 25–29.

[8] Ivan Ciofi, Antonino Contino, Philippe J. Roussel, Rogier Baert, Victor-H Vega-Gonzalez, Kristof Croes, Mustafa Badaroglu, Christopher J. Wilson, Praveen Raghavan, Abdelkarim Mercha, et al. 2016. Impact of wire geometry on interconnect RC and circuit delay. *IEEE Transactions on Electron Devices* 63, 6 (2016), 2488–2496.

[9] Ivan Ciofi, Philippe J. Roussel, Yves Saad, Victor Moroz, Chia-Ying Hu, Rogier Baert, Kristof Croes, Antonino Contino, Kevin Vandersmissen, Weimin Gao, et al. 2017. Modeling of via resistance for advanced technology nodes. *IEEE Transactions on Electron Devices* 64, 5 (2017), 2306–2313.

[10] Krisztián Flautner, Nam Sung Kim, Steve Martin, David Blaauw, and Trevor Mudge. 2002. Drowsy caches: Simple techniques for reducing leakage power. *ACM SIGARCH Computer Architecture News* 30, 2 (2002), 148–157.

[11] J. Guo, A. Papanikolaou, P. Marchal, and F. Catthoor. 2006. Physical design implementation of segmented buses to reduce communication energy. In *Proceedings of the Asia and South Pacific Conference on Design Automation, 2006.*

[12] Jin Guo, Antonis Papanikolaou, Hao Zhang, and Francky Catthoor. 2007. Energy/area/delay tradeoffs in the physical design of on-chip segmented bus architecture. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 15, 8 (2007), 941–944.

[13] Mohit Kumar Gupta, Pieter Weckx, Manu Perumkunnil Komalan, and Julien Ryckaert. 2023. Impact of interconnects enhancement on SRAM design beyond 5nm technology node. In *Proceedings of the 2023 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 1–5.

[14] Kris Heyrman, Antonis Papanikolaou, Francky Catthoor, Peter Veelaert, Koen Debosschere, and Wilfried Philips. 2006. Energy consumption for transport of control information on a segmented software-controlled communication architecture. In *Proceedings of the Reconfigurable Computing: Architectures and Applications: 2nd International Workshop, ARC 2006, Delft, The Netherlands, March 1-3, 2006, Revised Selected Papers 2*. Springer, 52–58.

[15] Changkyu Kim, Doug Burger, and Stephen W Keckler. 2002. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*. 211–222.

[16] Jung-Hoon Lee, Shin-Dug Kim, and Charles Weems. 2002. Application-adaptive intelligent cache memory system. *ACM Transactions on Embedded Computing Systems* 1, 1 (2002), 56–78.

[17] Yun Liang and Tulika Mitra. 2010. Instruction cache locking using temporal reuse profile. In *Proceedings of the 47th Design Automation Conference*. 344–349.

[18] Yun Liang and Tulika Mitra. 2013. An analytical approach for fast and accurate design space exploration of instruction caches. *ACM Transactions on Embedded Computing Systems* 13, 3 (2013), 1–29.

[19] Ankur Limaye and Tosiron Adegbija. 2018. A workload characterization of the spec cpu2017 benchmark suite. In *Proceedings of the 2018 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 149–158.

[20] Hsiao-Hsuan Liu, Carlo Gilardi, Shairfe M. Salahuddin, Zhenlin Pei, Pieter Schuddinck, Yang Xiang, Pieter Weckx, Geert Hellings, Marie Garcia Bardon, Julien Ryckaert, et al. 2024. Future design direction for SRAM data array: Hierarchical subarray with active interconnect. *IEEE Transactions on Circuits and Systems I: Regular Papers* 71, 12 (2024), 6495–6506.

[21] Hsiao-Hsuan Liu, Shairfe M. Salahuddin, Dawit Abdi, Rongmei Chen, Pieter Weckx, Philippe Matagne, and Francky Catthoor. 2022. Extended methodology to determine SRAM write margin in resistance-dominated technology node. *IEEE Transactions on Electron Devices* 69, 6 (2022), 3113–3117.

[22] Mahta Mayahinia, Hsiao-Hsuan Liu, Subrat Mishra, Zsolt Tokei, Francky Catthoor, and Mehdi Tahoori. 2023. Electromigration-aware design technology co-optimization for SRAM in advanced technology nodes. In *Proceedings of the 2023 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. IEEE, 1–6.

[23] Mahta Mayahinia, Tommaso Marinelli, Zhenlin Pei, Hsiao-Hsuan Liu, Chenyun Pan, Zsolt Tokei, Francky Catthoor, and Mehdi Tahoori. 2024. Dynamic segmented bus for energy-efficient last-level cache in advanced interconnect-dominant nodes. *IEEE Embedded Systems Letters* 16, 4 (2024), 321–324.

[24] Osvaldo Navarro and Michael Hübner. 2018. Runtime adaptive cache for the leon3 processor. In *Proceedings of the Applied Reconfigurable Computing. Architectures, Tools, and Applications: 14th International Symposium, ARC 2018, Santorini, Greece, May 2-4, 2018*. Springer, 343–354.

[25] Osvaldo Navarro, Jones Yudi, Javier Hoffmann, Hector Gerardo Muñoz Hernandez, and Michael Hübner. 2020. A machine learning methodology for cache memory design based on dynamic instructions. *ACM Transactions on Embedded Computing Systems* 19, 2 (2020), 1–20.

[26] Zhenlin Pei, Hsiao-Hsuan Liu, Mahta Mayahinia, Mehdi Tahoori, Francky Catthoor, Zsolt Tokei, Prashant Dubey, and Chenyun Pan. 2025. Interconnect/memory co-design and co-optimization using differential transmission lines. In *Proceedings of the IEEE Transactions on Very Large Scale Integration (VLSI) Systems*.

[27] Zhenlin Pei, Hsiao-Hsuan Liu, Mahta Mayahinia, Mehdi B. Tahoori, Francky Catthoor, Zsolt Tőkei, Dawit Burusie Abdi, James Myers, and Chenyun Pan. 2024. Ultra-scaled e-tree-based SRAM design and optimization with interconnect focus. *IEEE Transactions on Circuits and Systems I: Regular Papers* 71, 10 (2024), 4597–4610.

[28] Zhenlin Pei, Mahta Mayahinia, Hsiao-Hsuan Liu, Mehdi Tahoori, Francky Catthoor, Zsolt Tokei, and Chenyun Pan. 2023. Technology/memory co-design and co-optimization using E-Tree interconnect. In *Proceedings of the Great Lakes Symposium on VLSI 2023*. 159–162.

[29] Marisha Rawlins and Ann Gordon-Ross. 2013. Adaptive loop caching using lightweight runtime control flow analysis. *ACM Transactions on Embedded Computing Systems* 12, 1s (2013), 1–23.

[30] S. Salahuddin, M. Perumkunnil, E. Dentoni Litta, A. Gupta, P. Weckx, J. Ryckaert, M. H. Na, and A. Spessot. 2020. Buried power SRAM DTCO and system-level benchmarking in N3. In *Proceedings of the 2020 IEEE Symposium on VLSI Technology*. IEEE, 1–2.

[31] Timothy Sherwood, Erez Perelman, Greg Hamerly, and Brad Calder. 2002. Automatically characterizing large scale program behavior. *ACM SIGPLAN Notices* 37, 10 (2002), 45–57.

[32] Sarabjeet Singh and Manu Awasthi. 2019. Memory centric characterization and analysis of spec cpu2017 suite. In *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*. 285–292.

[33] Shuang Song, Qinzhe Wu, Steven Flolid, Joseph Dean, Reena Panda, Junyong Deng, and Lizy K. John. 2018. *Experiments with Spec Cpu 2017: Similarity, Balance, Phase Behavior, and Simpoints*. Technical Report.

[34] Chen Sun, Chia-Hsin Owen Chen, George Kurian, Lan Wei, Jason Miller, Anant Agarwal, Li-Shiuan Peh, and Vladimir Stojanovic. 2012. DSENT—a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In *Proceedings of the 2012 IEEE/ACM 6th International Symposium on Networks-on-Chip*. 201–210.

Received 11 August 2025; revised 11 August 2025; accepted 12 August 2025