# ZeroPoint's Ziptilion$^{\text{TM}}$ IP-Block Family for Transparent Expansion of Memory Capacity and Improvement of Effective Bandwidth

Technical Overview

ZeroPoint
TECHNOLOGIES

www.zptcorp.com

White Paper December 2021

## Table of Contents

# Executive Overview

ZeroPoint Technologies offers the Ziptilion™ IP block that is capable of expanding memory capacity and improving the effective memory bandwidth by a factor 2-3X using proprietary compression algorithms, depending on the workload. Ziptilion™ is fully transparent to the operating system, it interfaces to the memory controller and to the processor/cache subsystem (e.g., supports standard interfaces such as AXI) and consumes typically 1 mm$^2$ of silicon area (7nm). Thanks to our ultra-tuned compression/decompression accelerators and that data is compressed when fetched from memory, the memory access latency is often shorter with Ziptilion™ than without. Our customers have appreciated that we can offer RTL, simulation and FPGA models for testing Ziptilion™ in their environment. Ziptilion™ will be taped out in Q3 of 2022. This white paper is intended to provide technical details of the Ziptilion™ IP block and the status of our offering.

## Intended Audience

Technology specialists and strategists

## 1. Introduction

Main memory has a significant impact on the performance, energy consumption and cost of any computer system. In general, as compute performance doubles, main memory capacity and memory bandwidth must double as well. This essentially doubles the cost and energy consumption of memory in computerized devices.

ZeroPoint Technologies offers a unique family of IP-blocks, called Ziptilion$^{TM}$ that sits alongside the memory controllers in a compute device. The benefit of this ingenious family of IP-blocks is that memory capacity increases and memory bandwidth is improved substantially (by a factor of 2X-3X depending on the workload). This may yield a significant performance boost at a much lower cost and energy consumption than if the amount of physical memory increased correspondingly.

The Ziptilion$^{TM}$ family of IP-blocks include patented technology that manages to, on average, double memory capacity and memory bandwidth by dynamically compressing data in main memory fully transparently to user/system-level software. Ziptilion$^{TM}$ solves two challenges: 1) how to dynamically compress and decompress so fast that it does not affect the memory latency adversely and 2) how to effectively manage and expose the freed-up memory space transparently.

As for the first challenge, available hardware compression methods are simply way too slow to be useful for dynamic compression of memory data. In contrast, our proprietary lossless compression technology applies to any type of data and is so fast that it imposes virtually no impact on the memory latency. This is enabled by unique compression algorithms developed with the goal of offering a high compression level for a broad range of data types and unique technical approaches that allow for fast implementation of dynamic compression and recompression.

Concerning the second challenge, dynamic compression of memory data leads to management challenges as the size of data structures, which is fixed in a conventionally main memory, vary in compressed memory. To manage data structures in main memory that dynamically change their size is a challenge often delegated to the operating system. Our unique solution hides this completely to the user/system-level software, in a transparent fashion.

The Ziptilion$^{TM}$ IP-block comes with a software driver that expands the amount of physical memory. As data is dynamically compressed in the available physical memory, unused memory is freed up. This freed-up memory is monitored and collected by the software driver and is exposed to the system software as a *virtual page-cache* with a size as big as the available amount of physical memory. We show in this white paper that, with applications with twice as big data sets as the available amount of physical memory, we can offer nearly the same performance as a system with twice as much physical memory capacity.

Modern operating systems, such as Linux, uses ZRAM or ZSWAP (depending on the target system) to reserve part of the system's working memory to store pages, which are swapped out, in compressed form to create a memory-based swap space. Contrary to ZRAM/ZSWAP, that compress only the swapped-out pages, Ziptilion$^{TM}$ expands the **entire** memory. Ziptilion$^{TM}$ does not "steal" memory from the system's working memory, but it creates more space by compressing **all** memory pages.

This white paper describes ZeroPoint's  Ziptilion$^{TM}$ family of IP-blocks. It does so by first introducing the systems it applies to. Then it describes how the IP-blocks are integrated in these systems to provide the values. The white paper then reviews the technologies involved in detail and presents benchmark results.

## 2. Intended Systems

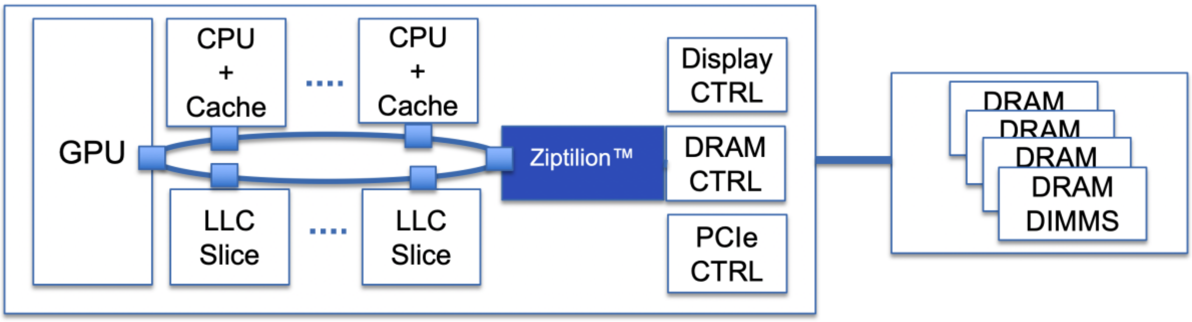


**Figure 1. Example system showing how Ziptilion™ can be integrated on a SoC.**

Figure 1 shows a generic organization of a typical microprocessor chip with three important functional blocks on-chip. Processors (cores) – denoted CPU – include core-specific functionality, such as TLBs and one or several levels of private caches. These core specific functionalities are typically interfaced to a shared, last-level cache (LLC), that interfaces to several memory controllers, denoted DRAM CTRL, that control the off-chip DRAM banks.

The shift to multi/many core microprocessor chip solutions has put an exponentially larger pressure on memory capacity and effective memory bandwidth. This is, simply put, rooted in Gene Amdahl's observation, several decades ago, that doubling compute performance requires a doubling of memory capacity and effective memory bandwidth. As demonstrated later in the white paper, **Ziptilion™ addresses this problem by substantially increasing the memory capacity as well as improving memory bandwidth.**

SoC acceleration is emerging as demonstrated by many recent developments (e.g. GPUs and accelerators for Deep Learning) in SoC designs. SoC-based accelerators face the same problem as general-purpose microprocessor chips do concerning memory capacity and effective memory bandwidth. **Ziptilion™ addresses both problems by offering substantially more memory capacity and effective memory bandwidth.**

## 3. About Ziptilion™

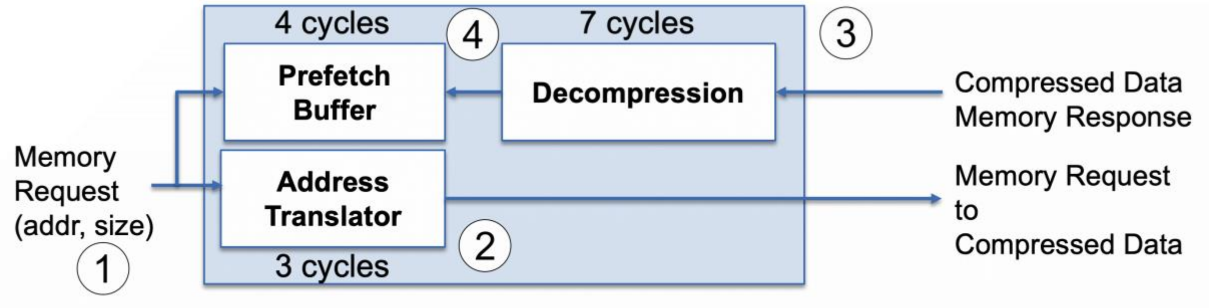

4 cycles
4
7 cycles
3
Prefetch Buffer
Decompression
Compressed Data Memory Response
Memory Request (addr, size)
Address Translator
Memory Request to Compressed Data
1
3 cycles
2

**Figure 2. The key building blocks of Ziptilion.™**

## 3.1 The Anatomy of Ziptilion$^{TM}$

As shown in Figure 2, the Ziptilion$^{TM}$ IP block comprises three building blocks denoted Locating, Compressing/Decompressing and Prefetching. When the last-level cache requests a memory block (1) and it is not available in the prefetch buffer inside the Ziptilion$^{TM}$ IP-block, to be discussed later, it will be requested from memory. Since memory blocks are compressed in memory, they will not appear on the same address as in a conventional system. The building block denoted Locating has metadata to locate where the requested block is and will issue a memory request to memory (2). Since the requested memory block and nearby memory blocks are compressed, not only the requested block but also nearby blocks will be fetched in one go. The building block Compressing/decompressing (3) will decompress and place them in the building block denoted Prefetching (4). This has the effect that if any of the nearby blocks are requested subsequently, they will typically be found in the Prefetching building block, thus cancelling that memory request. This saves memory bandwidth and shortens the memory request latency.

The Ziptilion$^{TM}$ IP-block is designed to offer an improvement in memory performance by having a normal read path and an optimized read path. The normal read path adds some latency to the memory-access path that is typically 15 cycles @ 1 GHz. This typically includes locating and decompressing several memory blocks. In the optimized path, however, the requested block is found in the Prefetching building block and the memory request is cancelled. Here, the memory request is served in four cycles @ 1 GHz instead of a hundred cycles. We have seen that it is quite common that the hit rate in the Prefetching building block is 50%. Then, the average memory request latency is 0.5 x 115 + 0.5 x 4 = 60 cycles @ 1 GHz instead of 100 cycles @ 1 GHz as in a conventional system. This is a significant performance boost and can offer a net increase in effective memory bandwidth by 2X.

The area of the Ziptilion$^{TM}$ IP block is modest too, as will be elaborated on later.

## 3.2 What is Ziptilion's$^{TM}$ Unique Feature?

Compression technologies have been used broadly for storage and for networking. A first common type of data stored/transferred is media. Such lossy and data-type specific compression techniques have been developed that work well on e.g. compressing images, video and audio. Another common type of data stored/transferred is text. Here, lossless Lempel Ziv (LZ) based compression techniques are used. While LZ compression was used in IBMs memory expansion technology[1], it was too slow and added significant latency to memory accesses. The reason is that LZ compression algorithms need big dictionaries to work well which will lead to unacceptable overheads to decompress data.

Other proposed memory expansion technologies have used simple compression techniques where one can compress/decompress a memory block in a few cycles. One example is Base-Delta Immediate (BDI)[2] encoding, where the difference between a value and a reference value is stored instead of the value itself. Another example is pattern-based compression (PBC) techniques[3] in which a few common patterns can be detected, for example a strike of the value zero, small integers etc. BDI and PBC do not compress by much, typically only 1.4X-1.5X on average, thus the expanded memory.

[1] R.B Tremaine et al. IBM MXT in a Memory Controller Chip. *IEEE Micro.* Vol 21. Issue 2. 2001.

[2] G. Pekhimenko et al. Base-Delta-Immediate Compression. Practical Data Compression for On-Chip Caches. In *PACT,* 2012.

[3] A. Alameldeen and D. Wood. Adaptive Cache Compression for High-Performance Processors. In *ISCA,* 2004.

ZeroPoint Technologies uses entropy-based compression[4,5]. Unlike LZ-based compression, ZeroPoint's proprietary statistical compression off-loads the management of the dictionary-based encodings from the compression and decompression process, thus cutting down on the overhead in the compression and decompression process significantly. A unique feature is that, to establish efficient encodings, memory data is analyzed in the background through intelligent sampling. This takes into account analysis of inferred data types to establish highly efficient encodings.

The baseline entropy-based family of compression techniques available in our IP portfolio do well across a wide range of benchmarks and typically expands memory by 2X to 3X depending on the workload. Apart from entropy-based compression, our IP portfolio also includes a range of compression algorithms that do data-type specific optimizations and combines entropy-based compression with deduplication. With these compression algorithms in place, we anticipate a memory expansion of 3X in future releases of the Ziptilion™ family of IP blocks. This is in our roadmap.

## 3.3 Memory Expansion

Most operating systems apply compression in the main memory to create either a swap space in memory (ZRAM) or a swap cache (ZSWAP). In ZRAM, swapped out pages are compressed with software algorithms. The SW-based method used by ZRAM/ZSWAP typically compresses 2X. However, as a part of memory is reserved to store the compressed swapped out pages, this space consumes typically a small fraction of available physical memory. Assuming 20% of the available physical memory is reserved and a 2X memory expansion, the net expansion of the physical memory is only 20%.

Ziptilion™ comes with its own virtual page-cache driver that is entirely transparent to the operating system. This driver manages the memory space that is freed up through compression of the content of the entire physical memory. With 2X memory expansion, it offers a virtual page-cache that is as big as the available physical memory.

The driver implements two key functionalities: 1) data analysis and page compression and 2) memory expansion management. In the background, compressibility of memory data is analyzed with the goal of maximally expanding the memory. Memory content is compressed at the page granularity. When a page is compressed, metadata is created to locate every single memory block in that page to be used by the Locating building block in the IP block (refer to Figure 2). This means that a compressed memory page will not occupy an entire physical page frame. The unused part of that page frame forms a fragment that will contribute to expansion of the memory by creating a virtual page-cache. The aim of the memory expansion management is to organize these fragments for maximal memory expansion.

When memory pressure builds up, pages will be swapped out to the virtual page-cache utilizing the free fragments created. Swapped-out pages can be reclaimed from the Ziptilion™ virtual page-cache driver. While the latency to reclaim a page from the virtual page-cache is a few microseconds, our analysis shows that the performance offered by the Ziptilion™ software driver is comparable to the performance obtained by a system with doubled physical memory capacity (results are shown in the next section).

Ziptilion's™ memory expansion methodology creates a *virtual page-cache* and allows memory to be expanded to host as many *passively* accessed pages as *actively* accessed pages. The true advantage of this methodology is that it is entirely transparent to operating systems. Nonetheless, ZeroPoint. Technology is underway to develop a memory expansion methodology that expands memory for actively accessed pages requiring some changes to the operating system.

[4] A. Arelakis and P. Stenström. SC$^2$: A Statistical Compression Cache Scheme. In *2014 ACM/IEEE Int. Symp. on Computer Architecture*

[5] A. Arelakis, F. Dahlgren, P. Stenstrom. HyComp: a hybrid cache compression method for selection of data-type-specific compression methods. In *IEEE MICRO,* 2015.

# 4. Results

Here, we will present results of what values our technology can offer. They are derived using the industry standard SPEC 2017 benchmark suite and include memory expansion and the effects on performance, on effective bandwidth and on memory latency.

## 4.1 Memory Expansion

Figure 3 shows the memory expansion offered by Ziptilion$^{TM}$ for the individual applications in the SPEC 2017 benchmark suite (blue bars) and for our improved proprietary compression algorithm (orange bars) on our roadmap.

A first observation is that all applications enjoy memory expansion although the expansion varies between applications for the basic Ziptilion$^{TM}$ compression algorithm. Some applications enjoy a memory expansion close to and above 3X whereas only three out of the 23 applications enjoy a memory expansion less than 30%. Overall, the basic Ziptilion$^{TM}$ algorithm offers a geometric mean expansion of 2X. Our improved offering, Ziptilion+, offers a geometric mean expansion of 2.8X and offers a memory expansion for all, except for one application of at least 30%. While the memory expansion factor is workload dependent, it offers a fairly robust and high memory expansion factor across all of the workloads.

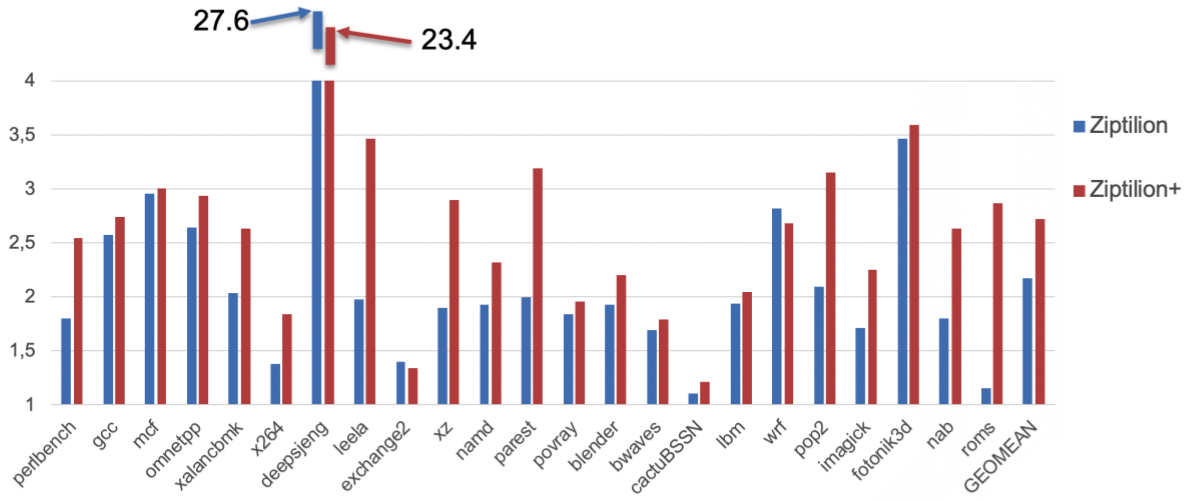


**Figure 3. Memory expansion factor with Ziptilion$^{TM}$ and Ziptilion+$^{TM}$ (on our roadmap).**

## 4.2 Effect on Performance

If a workload has a working set size that exceeds the size of physical memory, performance will quickly deteriorate. Ziptilion$^{TM}$ can counter such performance degradation by offering more memory capacity dynamically and totally transparently to the system. It does so by dynamically expanding the memory and creating a virtual page-cache with the memory freed up by compression. While ZRAM can typically only expand the size of the reserved memory space for swap storage using compression, Ziptilion$^{TM}$ expands the entire memory capacity. To see the performance effect of this, we run the SPEC 2017 applications with data sets that are twice as big as the amount of physical memory available on the baseline system.

Figure 4 shows the speedup obtained by Ziptilion$^{TM}$ and by a system with twice the amount of physical memory as the baseline system. A first observation is that, for the system with twice the amount of memory, four of the applications will get a performance boost that exceeds 2X (the orange bars). This is because these applications suffer from poor cache performance due to large working sets and benefit substantially from more physical memory. Interestingly, Ziptilion$^{TM}$ offers a performance boost that is close to that of a system with twice as much memory. Overall, Ziptilion$^{TM}$ offers, on average, a performance improvement of 30% closely tracking the performance boost of a system with 2X physical memory.

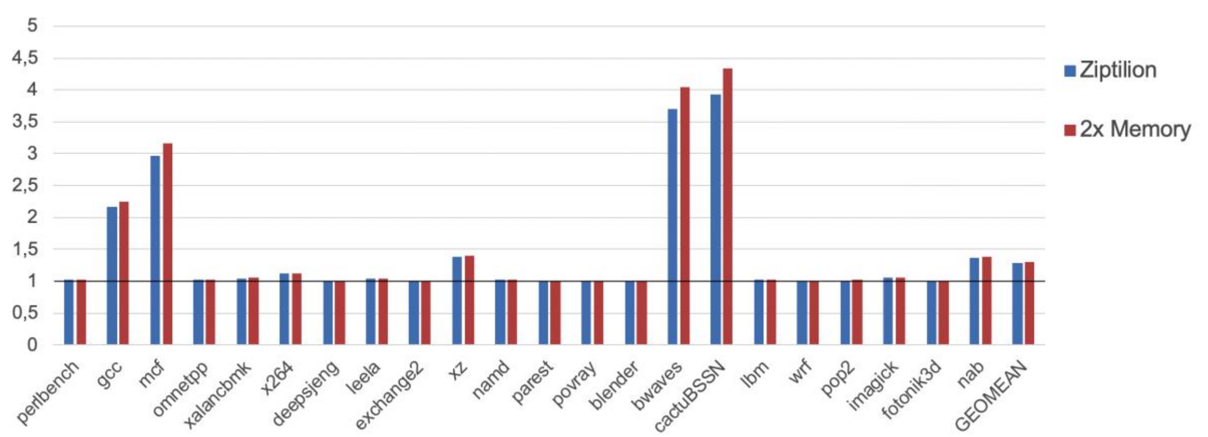


**Figure 4. Effect on performance with Ziptilion$^{TM}$. The graph shows the speedup of adding Ziptilion$^{TM}$ with same amount of physical memory as with a system with 2X amount of physical memory.**

## 4.3 Effect on Bandwidth

Apart from expanding memory capacity, Ziptilion$^{TM}$ can also increase the effective memory bandwidth. This is because compressed data is fetched from memory meaning that a request for a physical block, say 64 bytes, may bring several compressed blocks onto the chip. Figure 5 shows how memory traffic is affected by a system using Ziptilion$^{TM}$ relative to a system without Ziptilion$^{TM}$ (lower is better).

Applications with streaming behavior, where consecutive blocks are accessed, enjoy the highest reduction in memory traffic, whereas applications with random access and poor spatial locality enjoy less memory traffic reduction. Figure 5 shows the results. While some applications enjoy substantial traffic reduction, a few suffer from some slight increase in data traffic. This is because Ziptilion$^{TM}$, once in a while, must fetch compression-specific metadata that adds slightly some traffic. Overall, we can see that traffic reduction is, on average, 30% less compared to a system without Ziptilion™.

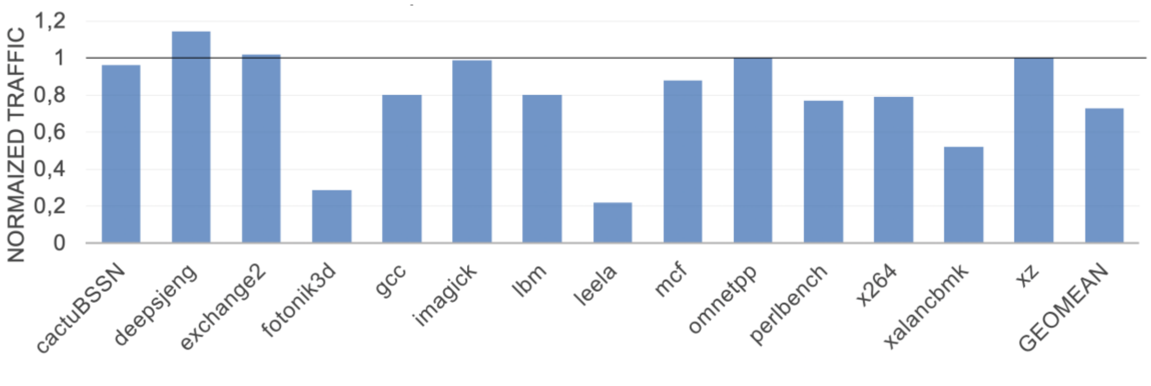
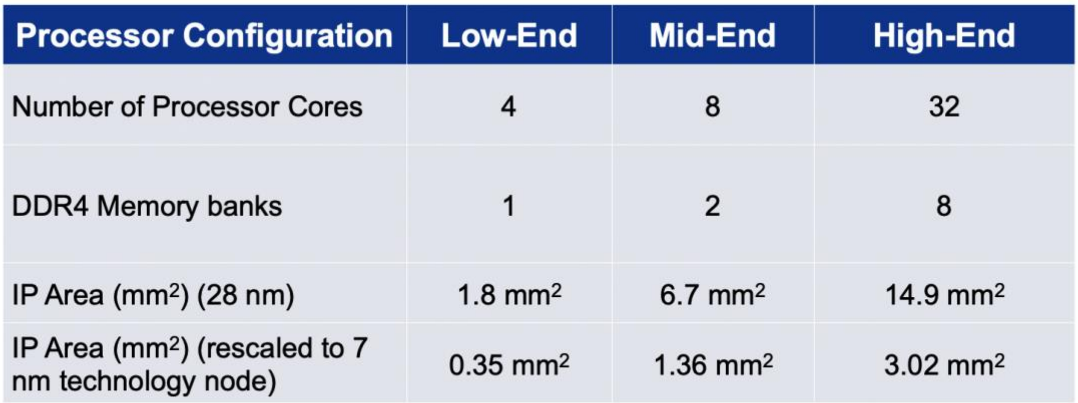


**Figure 5. Effect on bandwidth with Ziptilion$^{TM}$. The graph shows the reduction in traffic when adding Ziptilion$^{TM}$ (lower is better) compared to a system without Ziptilion$^{TM}$.**

### 4.4 IP Area

Ziptilion$^{TM}$ is RTL design ready. The area is a primarily a function of the number of cores and the number of memory controllers on a SoC. For this reason it is very important to review the particular architecture and the requirements to get accurate figures.



| Processor Configuration | Low-End | Mid-End | High-End |
|---|---|---|---|
| Number of Processor Cores | 4 | 8 | 32 |
| DDR4 Memory banks | 1 | 2 | 8 |
| IP Area (mm$^2$) (28 nm) | 1.8 mm$^2$ | 6.7 mm$^2$ | 14.9 mm$^2$ |
| IP Area (mm$^2$) (rescaled to 7 nm technology node) | 0.35 mm$^2$ | 1.36 mm$^2$ | 3.02 mm$^2$ |

**Table 1. IP area of Ziptilion$^{TM}$ for different SoC configurations assuming a 28 nm technology node and scaled results for a 7nm technology node.**

However, as an indication we consider three baseline SoC systems: 1) a low-end multicore system with 4 cores and a single memory controller 2) a mid-end multicore system with 8 cores and two memory controllers and 3) a high-end multicore system with 32 cores and eight memory controllers. Assuming a 28-nm technology node, the area is 1.8, 6.7 and 14.9 mm$^2$ for systems 1-3, respectively. The scaled area estimates are 0.35, 1.36 and 3.02 mm$^2$, respectively.

## 5. Status

Ziptilion$^{TM}$ is now in the final phase of evaluation. It is RTL ready and will be taped out in Q3 2022. We are engaged in customer projects and can offer a wide range of models for evaluation including GEM5 system models, RTL code for FPGA testbeds etc.

## 6. Conclusion

ZeroPoint Technologies offers the Ziptilion™ IP block that is capable of expanding memory capacity and effective memory bandwidth by a factor 2-3X using proprietary compression algorithms, depending on the workload. Ziptilion™ is fully transparent to the operating system, it interfaces to the memory controller and to the processor/cache subsystem (e.g. AXI) and consumes typically 1 mm$^2$ of silicon area (7nm). Thanks to our ultra-tuned compression/decompression accelerators and that data is compressed when fetched from memory, the memory access latency is often shorter with Ziptilion™ than without. Our customers have appreciated that we can offer RTL, simulation and FPGA models for testing Ziptilion™ in their environment. Ziptilion™ will be taped out in Q3 of 2022.

# 7. Appendix

## Patents

**Granted patents**

1. Cache System and a Method of Operating a Cache Memory, US9330001 (B2)
2. Methods, devices and systems for semantic-value data compression and decompression, US10268380 (B2)
3. Methods, Devices and Systems for Compressing and Decompressing Data, SE540178 (C2)
4. Methods, Devices and Systems for Hybrid Data Compression and Decompression, US10476520 (B2)
5. Variable-sized symbol entropy-based data compression, SE542507 (C2)
6. Variable-sized symbol entropy-based data compression, US10862507 (B2)
7. Systems, methods and devices for eliminating duplicates and value redundancy in computer memories, SE543186 (C2)
8. Methods, devices and systems for hybrid data compression and decompression, US10819369 (B2)
9. Methods, devices and systems for compressing and decompressing data, US10831655 (B2)
10. Methods, devices and systems for compressing and decompressing data, US10846218 (B2)
11. Methods, devices and systems for hybrid data compression and decompression, EP3304746 (B1)
12. Managing free space in a compressed memory system, SE543649 (C2)

## Contact ZeroPoint Technologies AB

### DR. PER STENSTRÖM

per.stenstrom@zptcorp.com

CHIEF SCIENTIST OFFICER (CSO)
FOUNDER AND CO-INVENTOR

Dr. Per Stenström, Chief Scientist Officer (CSO), Founder and co-inventor – Professor of Chalmers University of Technology and an internationally renowned expert in computer architecture, especially for his contributions to high-performance memory systems (ACM and IEEE Fellow). He has authored/co-authored more than 150 papers and 14 approved patents on a broad range of topics in computer architecture. He has industrial experience as a senior staff engineer at Sun Microsystems and several board-level and executive positions in startup companies. He is member of the Royal Swedish Academy of Engineering Science, the Academia European and the Spanish Royal Academy of Engineering.

### DR. ANGELOS ARELAKIS

angelos.arelakis@zptcorp.com

CHIEF TECHNOLOGY OFFICER (CTO)
FOUNDER AND CO-INVENTOR

Dr. Angelos Arelakis, Chief Technology Officer (CTO), Founder and co-inventor. He is an expert in memory system architectures and ultra-fast data compression. He holds a Ph.D. degree in Computer Architecture from the Computer Science and Engineering department of Chalmers University of Technology, Sweden. He has published in flagship computer architecture conferences as well as a book. He has earned two paper awards from the European Network of Excellence on High Performance and Embedded Architecture and Compilation (HiPEAC). He has received a scholarship from the King Carl XVI Gustaf Foundation’s 50th anniversary fund for science, technology and the environment. He is a member of HiPEAC and holds 14 patents.

### KLAS MOREAU

klas.moreau@zptcorp.com

CEO

Klas is a determined and results focused industry leader, with 20 years of progressive leadership experience in global enterprise environments including Ericsson, Sweden’s world-leading telecommunications corporation. Klas’ understanding of technology, combined with his business acumen makes him a great fit now that ZeroPoint Technologies are moving into a phase of stronger focus on business development and company growth.

www.zptcorp.com