ZeroPoint

How to accelerate memory bandwidth by 50% with ZeroPoint technology

Technical Overview
White Paper August 2023

Content

Executive Overview

Ziptilion™-BW (BW for main memory bandwidth acceleration) is a ZeroPoint technology that accelerates the effective memory bandwidth by a factor of 30-50% using proprietary compression algorithms across a broad range of server workloads yielding an improvement in performance/watt by 20%. Ziptilion™-BW interfaces to the memory controller and to the processor/cache subsystem (e.g. AXI) and consumes typically 0.4 mm^2 of silicon area (@ 5 nm).

Thanks to our ultra-tuned compression / decompression accelerators, data is already compressed when fetched from memory, thus the memory access latency is often shorter with Ziptilion™-BW than without. Our customers have appreciated that we can offer RTL, simulation and FPGA models for testing Ziptilion™-BW in their environment. This white paper is intended to provide technical details of the Ziptilion™-BW IP block and the status of our offering.

Intended Audience

Technology specialists and strategists

1. Introduction

Digitalization quickly accelerates energy consumption and is projected to stand for more than one-fifth of global electricity demand by 2030. This makes "performance per watt" critical.
ZeroPoint technology for microchips delivers up to 50% more performance per watt by removing unnecessary information, meaning that:

- Data centers can maximize performance and lower energy consumption.
- Smart device developers can maximize performance and user experience without compromising on battery life.

ZeroPoint technology combines ultra-fast data compression with real time data compaction and transparent memory management. This white paper introduces ZeroPoint Technologies' Ziptilion™-BW product.

Ziptilion™-BW is built on the foundation of the ZeroPoint technology. This is illustrated in Figure 1 and entails i) novel, general-purpose, lossless ultra-low latency compression algorithms, ii) real-time compaction techniques and iii) compressed memory management transparent to the software stack. Ziptilion™-BW accelerates memory bandwidth using three patented innovative techniques. The first technique is new proprietary data compression algorithms that manage to compress data by a factor 2-4x with ultrashort latency (a 64-byte block is compressed in a few nanoseconds in a pipelined fashion). The second innovative technique is proprietary realtime data compaction algorithms that manage to translate compression into bandwidth savings at nanoseconds speed. A third technique aims at managing the memory layout to maximize available memory bandwidth. The bottom line is that microprocessors that are equipped with Ziptilion™ can increase memory bandwidth by 30-50% yielding an improvement in performance/watt by 20%.

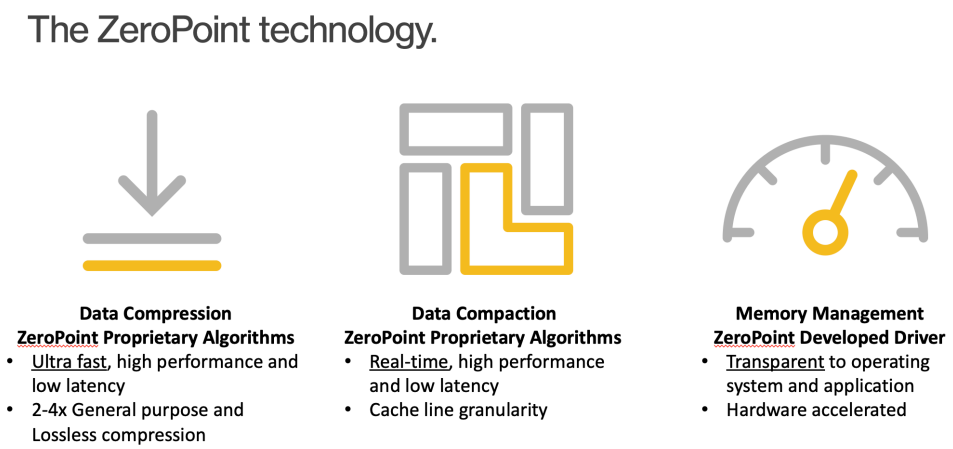

Figure 1: The ZeroPoint technology.

Available hardware compression methods are simply way too slow to be useful for dynamic compression of memory data. In contrast, our proprietary lossless compression / compaction techniques (patents pending) 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 / compaction 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.

This white paper describes ZeroPoint's product offering - Ziptilion™-BW for expansion of effective memory bandwidth. 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

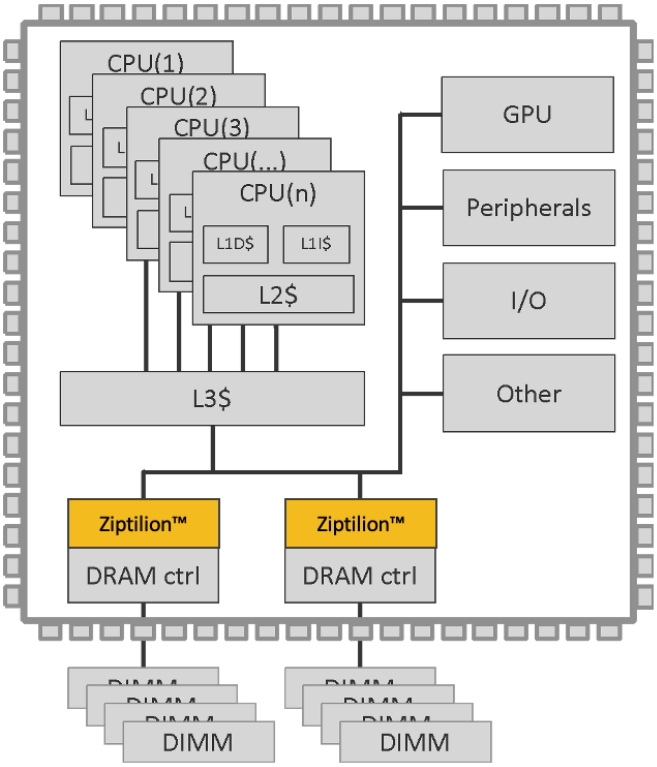

Figure 2: Example system showing how Ziptilion™-BW can be integrated on a SoC.

Figure 2 shows a generic organization of a typical microprocessor chip with three important functional on-chip blocks. 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 – denoted L3 cache – that interfaces to several memory controllers, denoted MCTRL, that control the off-chip DRAM banks. **Ziptilion™-BW is integrated on the memory access path in close proximity to the MCTRL.**

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™-BW addresses this problem by substantially increasing the effective bandwidth.**

SoC acceleration is emerging as demonstrated by many recent developments (e.g., GPUs and accelerators for Machine Learning - ML) in SoC designs. SoC-based accelerators face the same problem as general-purpose microprocessor chips do concerning effective memory bandwidth. **Ziptilion™-BW addresses this problem by offering substantially higher effective memory bandwidth.**

3. About Ziptilion™-BW

3.1 Integration of the Ziptilion™-BW IP block

Ziptilion™-BW is integrated on the memory access path close to the MCTRL. This is illustrated in Figure 2 and in more detail in Figure 3, where all the various cores (e.g., CPUs, GPUs or other accelerators are referred to as "C"). Ziptilion™-BW is typically integrated in the memory subsystem between the MCTRL and the chip's interconnect. Ziptilion™-BW is compatible to SoC interconnection protocols such as AMBA AXI and CHI and it can be adapted to other protocols. It supports various data bus widths, typically 128b and 256b but also 512b.

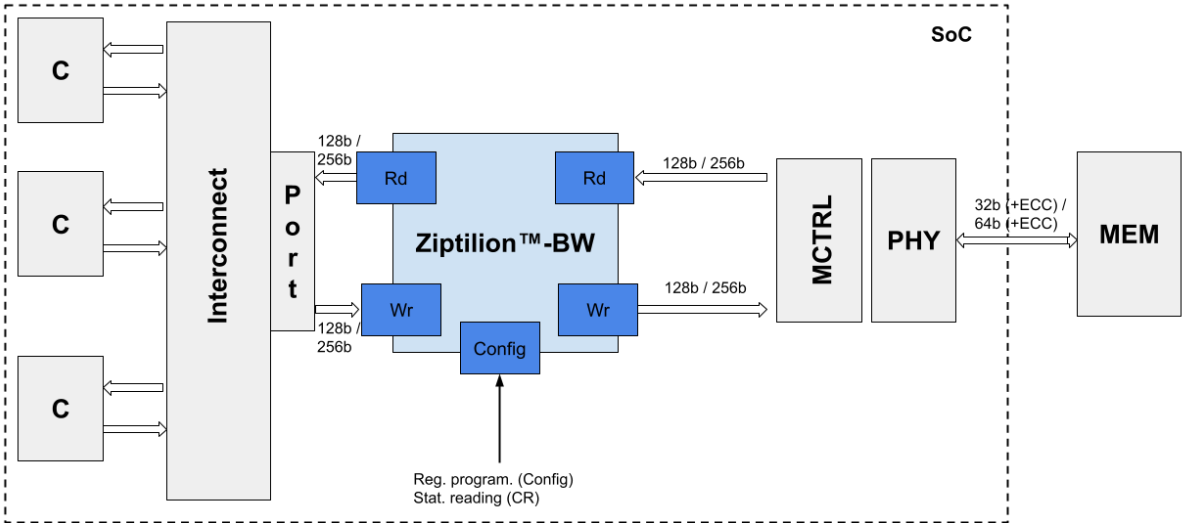

Figure 3: Integration of Ziptilion™-BW in the system.

The system on the left of the Ziptilion™-BW is agnostic to that the data memory content is compressed by Ziptilion and it continues to issue memory requests as in a system without Ziptilion. The memory subsystem on the right (MCTRL, PHY, etc.) is also agnostic to that the data transferred back and forth are also in compressed form. This is achieved by Ziptilion's transparent management of the compressed memory space, to be elaborated on in further details in the next section.

3.2 The Anatomy of the Ziptilion™-BW IP block

Figure 4 shows the anatomy of the Ziptilion™-BW IP block. It comprises five building blocks to handle memory read and write requests: A Prefetch buffer (PB), an Address translator (AT), a Decompressor, a Write buffer (WB) and a Compressor. For simplicity, but without loss of generality, we assume that only the LLC can issue memory requests. An LLC miss injects a read request into Ziptilion™-BW. The AT maintains metadata to locate a requested block in the compressed memory. On an AT hit, a memory request for this block is issued to the memory controller. As data is compressed, memory can respond with several compressed blocks that fit into the same physical block frame of typically 64B. These blocks are decompressed by the Decompressor and are stored uncompressed in the PB.

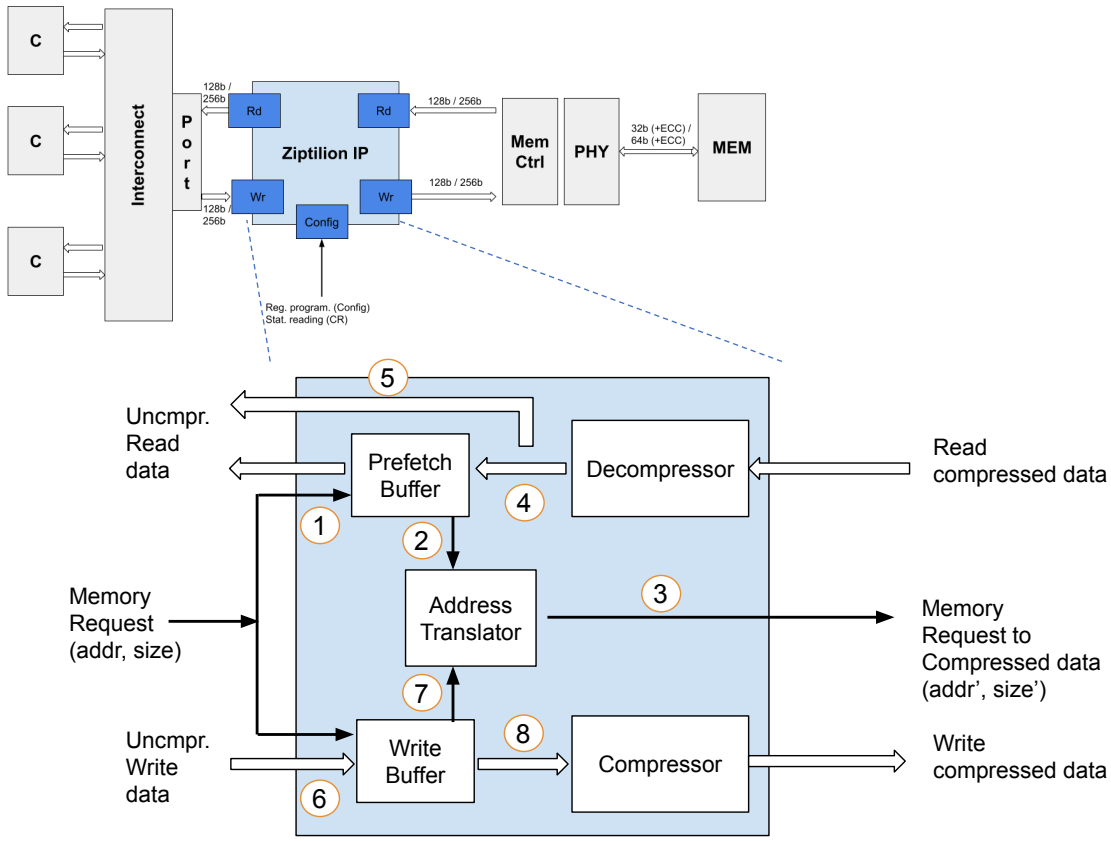

Figure 4: The key building blocks of Ziptilion^{BW}-BW

Let's follow the flow of an LLC miss through the Ziptilion™-BW IP block to see how it can improve the available memory bandwidth. The request first probes the PB whether the block was prefetched due to compression (in Figure 4). The WB is also probed in case the read block was written back recently. On a PB or WB hit, the memory request will be cancelled leading to memory traffic savings. Higher spatial locality leads to more traffic saving because more data blocks will be transferred in one go. Conversely, if the requested block is not in the PB (in Figure 4), the physical address will be translated to establish the location in memory where the compressed block resides and a request is sent to memory. The block returned from memory may contain several compressed blocks that are decompressed . All the blocks contained will be inserted in PB uncompressed and the requested block will be returned to the requestor .

The Address translator uses metadata to locate the compressed block. The Ziptilion™-BW IP block organizes memory blocks in *sectors*, where each sector contains a fixed number of neighboring compressed memory blocks, typically four. Note: a sector's blocks are consecutive in the memory space linked to a specific memory controller but can originate from different OS pages due to memory interleaving. The Ziptilion™-BW IP block is agnostic to this. The location of the first memory block of each sector corresponds to its physical address and is aligned to the DRAM block boundaries; the second memory block starts where the first compressed memory block ends, etc. Hence, the Ziptilion™ IP block stores only the memory-block-size metadata, referred to. The Ziptilion™ IP block allows a compressed memory block to be of any size in the [0,64)-byte spectrum, requiring 6b metadata/block. However, if a compressed memory block is ≥ 64 bytes, it is stored uncompressed which requires 1b/block of metadata. Since the compression metadata per compressed block is 7b, the metadata of 64 consecutive blocks (i.e., 64 x 7b = 448b) is aggregated in one metadata entry so that when a piece of metadata is fetched from memory, the metadata of neighboring compressed data

contained in that DRAM block is prefetched. All metadata is stored in main memory. However, to avoid adding an extra memory access to fetch metadata for every memory data read/write, the Ziptilion™-BW IP block caches the most recently used metadata entries on chip in the Address translator.

Memory write-back requests are intercepted by the Ziptilion™-BW IP block and complete in the Write buffer (6). The Write buffer is also organized to store sectors so that adjacent compressed memory blocks belonging to the same sector are compacted upon eviction from the write buffer. Efficient compression and compaction of written memory data is crucial for improving memory bandwidth. If a compressed memory block (evicted from the write buffer) exceeds in size compared to the current size in memory, writing the compressed memory block back to memory will result in overwriting of adjacent memory block(s) leading to data corruption, known as *compression overflow*. Ziptilion is aware of the current memory state by consulting the Address Translator (7). Hence, Ziptilion™-BW checks whether a compression overflow occurs upon a write-buffer eviction and fetches extra memory blocks to resolve it if the affected memory blocks are not present in the Write buffer or in the Prefetch buffer. This results in read-modify-write (RMW) transactions. If/When the written sector is overflow-free, it is compressed and written back to memory (8).

The Ziptilion™-BW 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 20 cycles @ 1.6 GHz. This includes checking the internal buffers, locating the compressed block in memory and decompressing the receiving compressed block into several uncompressed memory blocks. In the optimized path, however, the requested block is found in the Prefetch Buffer and the memory request is cancelled. Here, the memory request is served in 8 cycles @ 1.6 GHz instead of going further to the memory subsystem and requiring several tens of ns. We have seen that it is quite common that the hit rate in the Prefetching building block is 40%, on average. Then, the average memory request latency is 0.6 x 120 + 0.4 x 8 = 75 cycles @ 1.6 GHz instead of 100 cycles @ 1.6 GHz as in a conventional system. This is a significant performance boost and can offer a net increase in effective memory bandwidth 30-50%.

The area of the Ziptilion™-BW IP block is modest too. As will be elaborated on later, for a SoC with 8 cores and one memory controller, the IP block consumes 0.4 mm^2 assuming a 5-nm technology.

3.3 The Ziptilion™ Unique Features

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 prohibitive 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 expand memory bandwidth by much – the best to hope for is 10% bandwidth savings.[4,6]

ZeroPoint Technologies uses entropy-based compression[4,6,7](patents pending). Unlike LZ-based compression, ZeroPoint's proprietary statistical-based compression off-loads the management of the dictionary-based encodings to software 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 compresses data 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, the anticipated compression ratio goes beyond 3x.

---

[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.

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

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

[7] A. Angerd, A. Arelakis, V. Spiliopoulos, E. Sintorn, P. Stenstrom. GBDI: Going Beyond Base-Delta-Immediate Compression with Global Bases. I*n Proc. of the 28th IEEE Symposium on High-Performance Computer Architecture, 2022*

4. Results

In this section we present results of the values provided by the Ziptilion™-BW product. They are derived using the industry standard SPEC2017[5] benchmark suite, the MonetDB[6] database benchmark and HPC applications such as the HPCG[7] benchmark, which is an industry standard for HPC applications, and real applications such as SMURFF[8] and NEMO[9] The metrics include the memory expansion potential and its effects on effective memory bandwidth and performance measured as Instructions Per Cycle (IPC) improvement.

4.1 Memory expansion

This section presents the evaluation of Ziptilion in terms of memory expansion and memory bandwidth acceleration potential. This is evaluated using the compression-ratio metric. This metric measures the efficiency of the online compression engines by which they compress on the cache line granularity. The compression ratio is measured by compressing each individual cache line in the memory and then calculating the uncompressed memory size over the compressed memory size.

Figures 5 - 8 plot the compression ratio for the SPEC2017INT benchmarks, TPC-H database queries, SPEC2017FP and HPC applications, respectively.

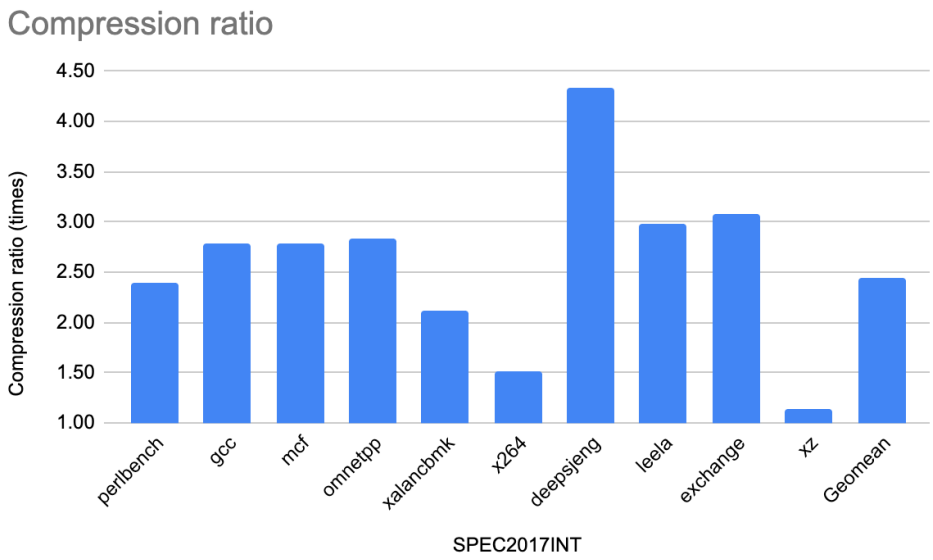

Figure 5: Compression ratio for the SPEC2017INT applications.

[5] SPEC2017: https://www.spec.org/cpu2017/

[6] MonetDB: https://www.monetdb.org/

[7] HPCG: https://www.hpcg-benchmark.org/

[8] SMURFF: A high-performance implementation of Bayesian matrix factorization with limited communication, Tom Vander Aa et al., International Conference on Computational Science, https://github.com/ExaScience/smurff

[9] NEMO Ocean Engine: Gurvan Madec and NEMO System Team, Scientific Notes of Climate Modelling Center, 2019, http://www.nemo-ocean.eu/.

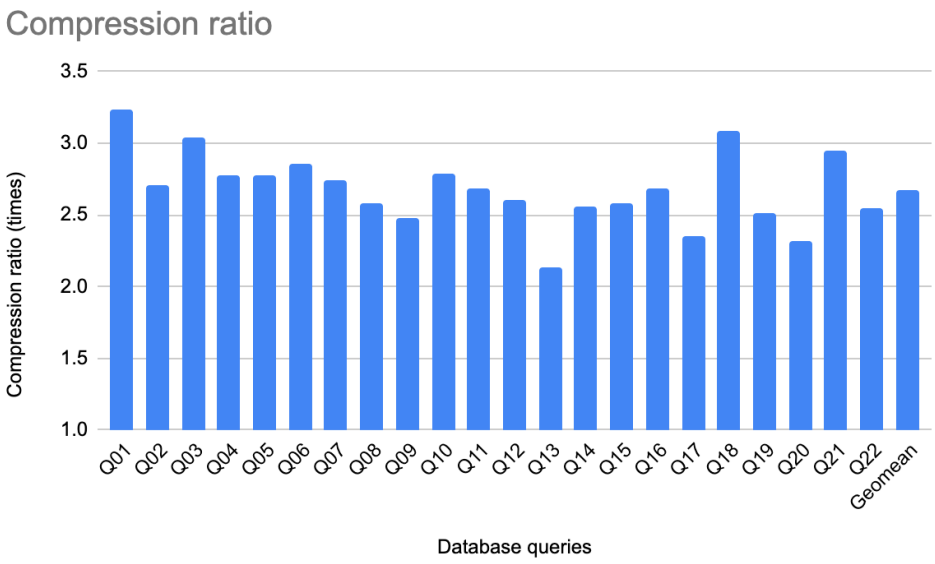

Figure 6: Compression ratio for TPC-H queries when running MonetDB.

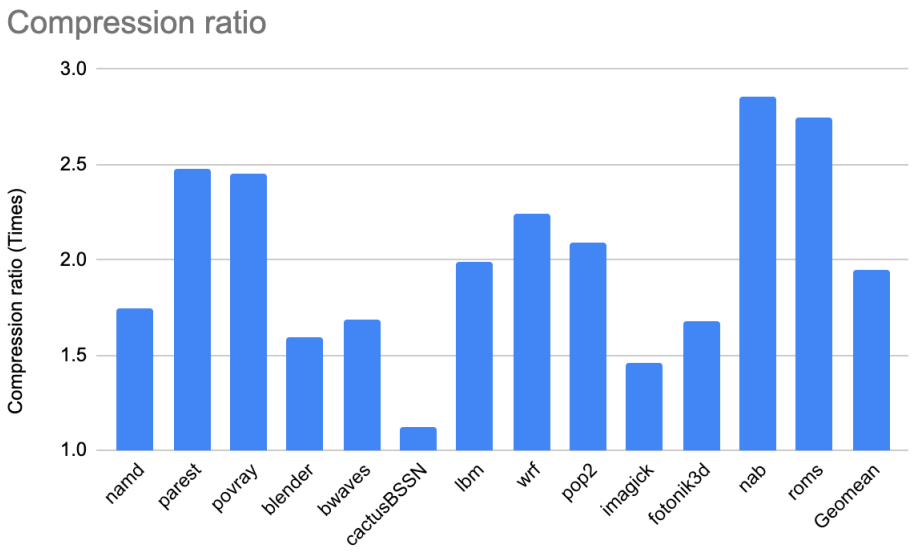

Figure 7: Compression ratio for the SPEC2017FP applications.

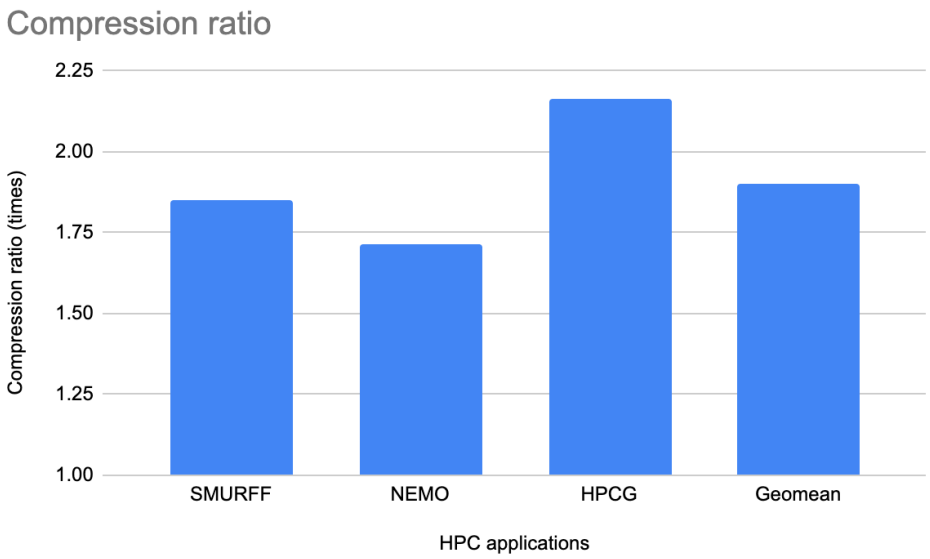

Figure 8: Compression ratio for HPC applications.

A first observation is that all applications enjoy a substantial compression ratio although the ratio varies between applications. This is to be expected because ZeroPoint's algorithms are lossless. Some applications enjoy a compression ratio close to and above 3x whereas only four out of the 27 applications[10] enjoy a memory expansion less than 1.5x.

The compression ratio for the database queries is more than 2.5x in more than 80% of the queries (18 out of 22) queries. Moreover, we observe that Ziptilion offers a geometric mean compression ratio of 2.5x for integer-based workloads and 2.7x for database-based workloads, while for floating-point applications the geometric mean compression ratio is close to 2x (1.95x for SPEC2017FP and 1.9x for the evaluated HPC applications).

Bottom line is that while the ratio is data dependent, it offers a fairly robust and high compression ratio (typically 2x-3x) across all of the workloads.

4.2 Effect on Bandwidth

Apart from expanding memory capacity, our product Ziptilion™-BW improves 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 in one go.

Figures 9 - 12 present how much more memory bandwidth is gained by a system using Ziptilion™-BW relative to a system without Ziptilion™-BW (higher is better). The evaluation is presented for the SPEC2017INT, the database queries, SPEC2017FP and HPC applications.

[10] We consider all the database queries as one application despite their various characteristics.

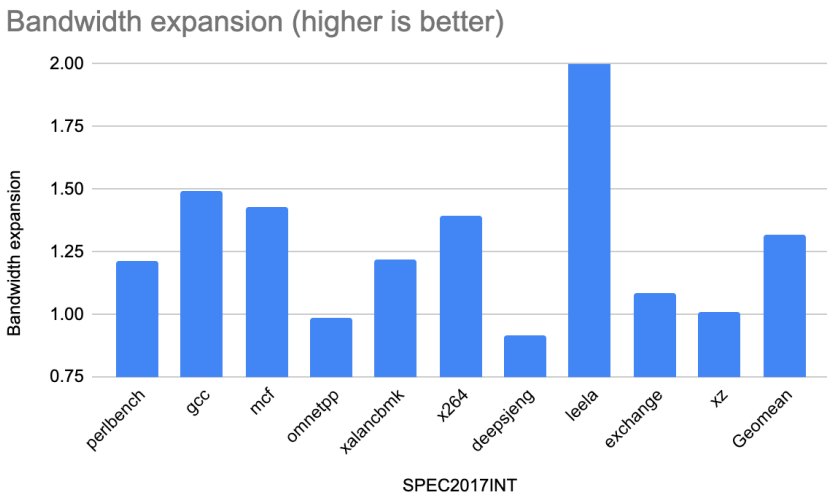

Figure 9: Effect on bandwidth with Ziptilion™-BW for the SPEC2017INT benchmarks.

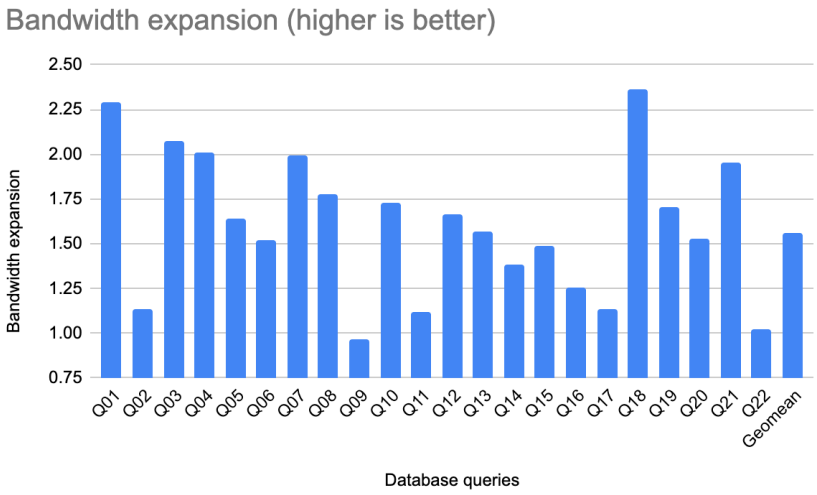

Figure 10: Effect on bandwidth with Ziptilion™-BW for TPC-H database queries

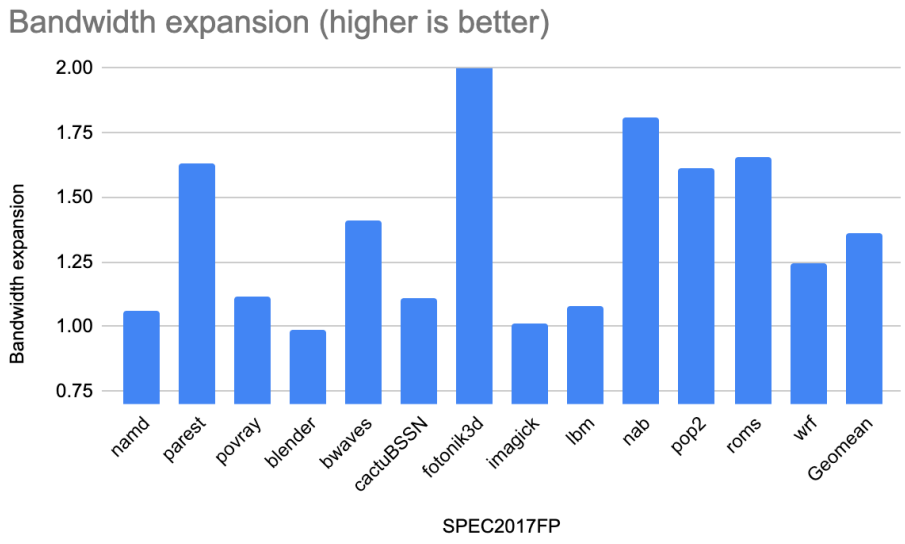

Figure 11: Effect on bandwidth with Ziptilion™-BW for SPEC2017FP benchmarks.

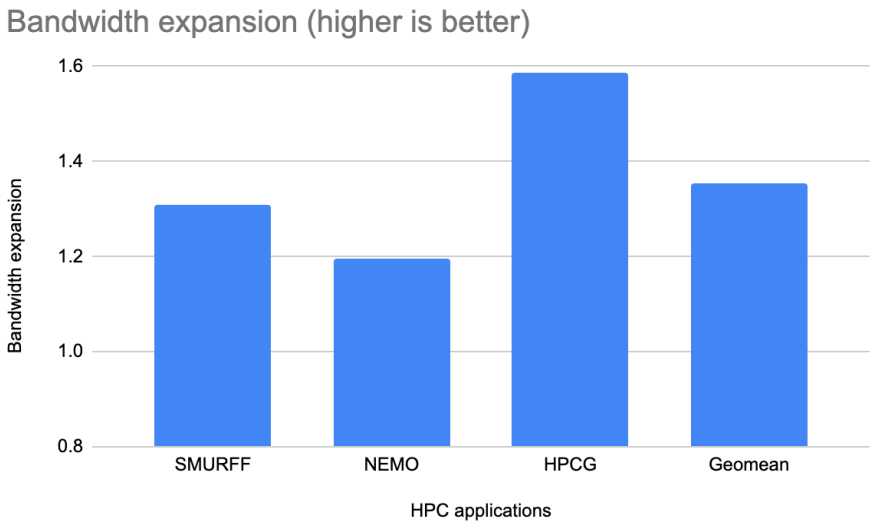

Figure 12: Effect on bandwidth with Ziptilion™-BW for HPC applications

Applications with streaming behavior, where consecutive blocks are accessed, and with high compression ratio such as several database queries (e.g., Q01, Q03, Q18 - Figure 10), fotonik3d (Figure 11), HPCG (Figure 12), enjoy the highest expansion in memory bandwidth, whereas applications with random access and poor spatial locality enjoy less memory bandwidth expansion (i.e., omnetpp, xz and deepsjeng -- Figure 9).

Ziptilion™-BW works robustly offering significant bandwidth improvement for memory access behaviors that saturate the memory bandwidth while turning off compression for random memory access patterns[11] reducing latency and traffic overheads. This is done by employing a novel and proprietary mechanism that identifies, in realtime, which memory areas that benefit from compression and and only compresses data in these memory areas. The mechanism works transparently from the rest of the system and allows Ziptilion™-BW to have compression on always.

While the bandwidth expansion varies across applications, 21 memory access scenarios (out of 48) enjoy a memory bandwidth expansion of more than 50% (1.5x). Overall, the bandwidth expansion is, on average, 25% (1.25x), 50% (1.5x), 35% (1.35x) and 35% (1.35x) for SPEC2017INT, database queries, SPEC2017FP and HPC applications, respectively.

4.3 Effect on Performance

This section evaluates the impact of Ziptilion™-BW on overall computer system performance. The metric used is Instructions Per Cycle (IPC). Figures 13-16 summarize the results.

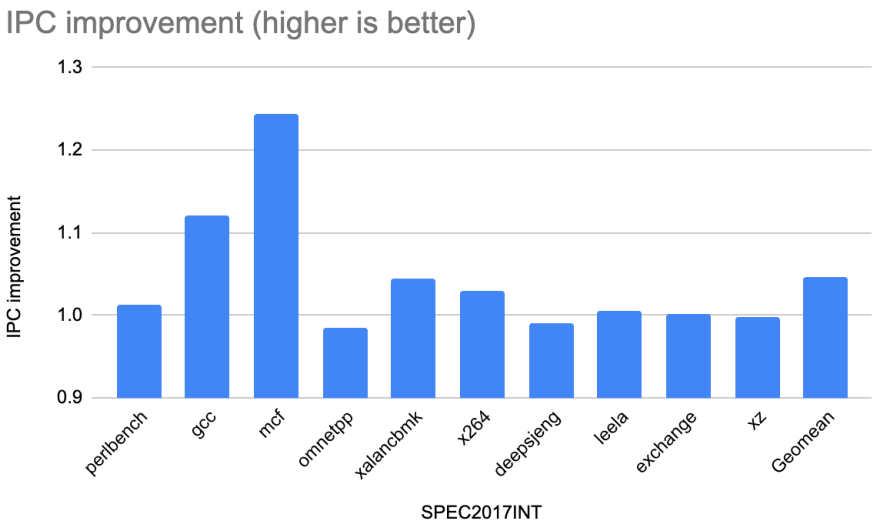

Figure 13: IPC improvement with Ziptilion™-BW for the SPEC2017INT benchmarks.

[11] Data sets with random memory behavior is a general problem to memory access efficiency. ZeroPoint technology have embedded intelligence that detects this behavior and turn off data compression to inhibit additional penalty to memory accesses. Examples of these applications are some SPEC2017 applications (omnetpp, deepsjeng, xz, blender) and a few database queries (Q02, Q11, Q17, Q22).

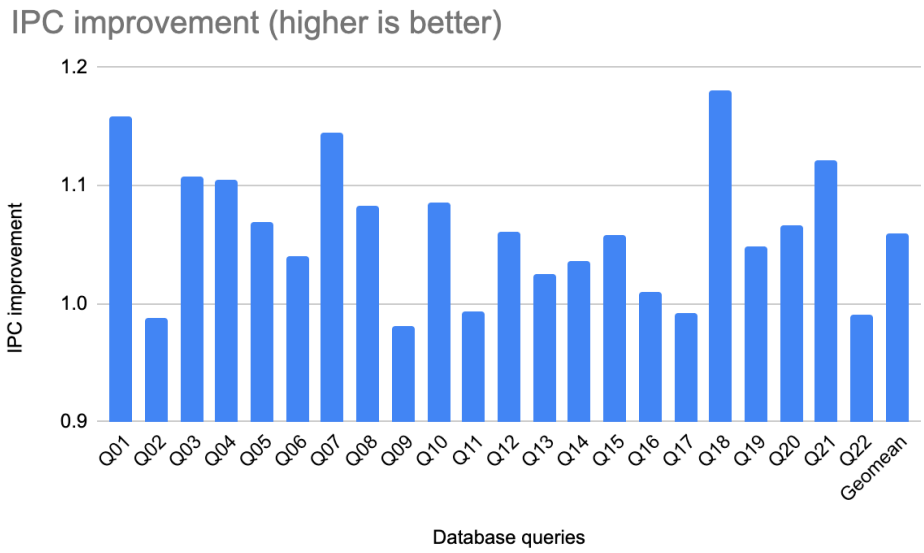

Figure 14: IPC improvement with Ziptilion™-BW for TPC-H database queries.

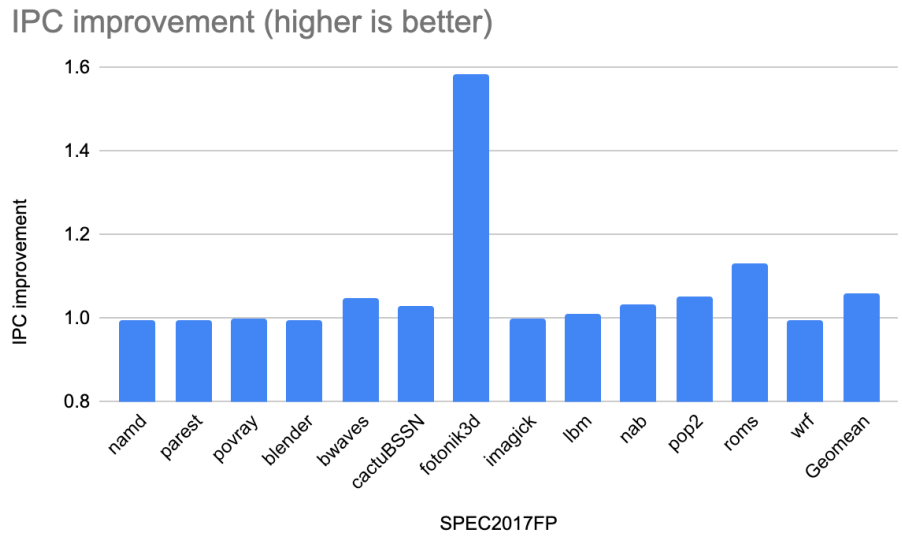

Figure 15: IPC improvement with Ziptilion™-BW for SPEC2017FP benchmarks.

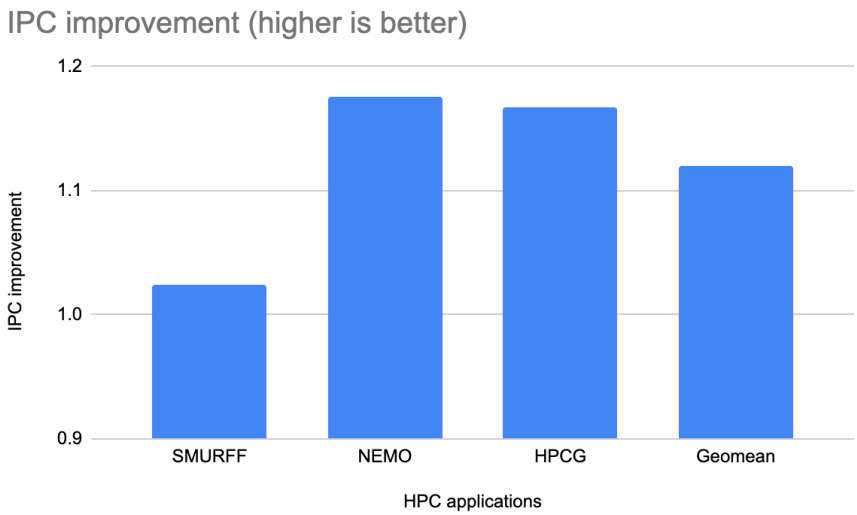

Figure 16: IPC improvement with Ziptilion™-BW HPC applications.

Memory-bandwidth intensive applications, such as gcc and mcf (from the SPEC2017INT), fotonik3d and roms (from SPEC2017FP) as well as most of the evaluated HPC applications and database queries (8 out of 21) enjoy significant improvement in IPC.

Specifically, from the SPEC2017INT applications (Figure 13), the IPC is improved by 12% and 25% for gcc and mcf, respectively; in the database queries (Figure 14) Q01, Q03, Q04, Q07, Q18 and Q21 the IPC is improved by more than 10% and up to 18%; in the SPEC2017FP applications , (Figure 15), the IPC is improved by 60% and 13% for fotonik3d and roms, respectively. Finally, the IPC improvement is 18% and 17% for NEMO and HPCG, respectively.

About 12 applications and database queries (out of 48) have an IPC improvement in the range of 4 to 9%. 7 out of 48 evaluated applications and database queries, show a small overhead in performance. Specifically, omnetpp and deepsjeng have an IPC penalty of 1.5% and 0.8% respectively, 4 database queries (Q02, Q11, Q17 and Q22) have an IPC penalty of <1% and Q09 an IPC penalty of 1.9%.

Bottomline is that the Ziptilion™-BW can improve the system's performance on average by 12% for the evaluated benchmarks, close to 20% for traditional memory-bandwidth intensive applications and we have even seen up to 60% in some cases.

4.4 Effect on Power and Performance/Watt

Ziptilion™-BW yields 12% higher performance, on average, across a wide range of applications from different domains as shown in the previous section. Reducing memory traffic also reduces the energy expended. In Section 4.2 we have seen that the memory traffic goes down by 35%. This will cut the dynamic energy by the same amount.

Typically, at least 30% of the power in a computer system is dissipated in the memory system. The power is typically evenly dissipated statically in DRAM controlled by its size and dynamically due to memory transfers. As Ziptilion™-BW can reduce the traffic by 35%, it means that dynamic power of a computer system goes down by 5%.

The impact of Ziptilion™-BW on performance per watt is established by the ratio of improvement in performance over the reduction in power dissipation: 1.12/0.95 = 1.2x improved performance / watt according to the findings in Sections 4.2 and 4.3. **That is 20% more performance per watt.**

20% more performance per watt is a significant improvement and can be achieved with the product available today.

4.5 IP Area

The Ziptilion area is measured to 0.4mm^2 per memory channel for a 5nm technology node. More than 70% of the Ziptilion's real estate is SRAM to accommodate the prefetch buffer, write buffer and address translation. A significant fraction of this area can be removed or mitigated by a co-design between ZeroPoint Technologies and the SoC manufacturer.

5. Status

Ziptilion™-BW is now in the final phase of evaluation. We are engaged in customer projects due to be taped out and can offer a wide range of models for evaluation including architecture/system models (such as GEM5-based), RTL code for FPGA testbeds etc.

6. Conclusion

Ziptilion™ is a ZeroPoint technology that is capable of expanding memory capacity by a factor 2-3x and effective memory bandwidth 30-50% using proprietary compression algorithms, depending on the workload leading to an improvement in performance per watt by 50%. Ziptilion™ is fully transparent to the operating system and the software stack. It interfaces to the memory controller and to the processor / cache subsystem with standard interfaces (e.g. AXI or CHI) and consumes typically 0.4 mm^2 of silicon area (@ 5nm). Thanks to our ultra-tuned compression/decompression accelerators and that data is compressed when fetched from memory, the memory access latency is reduced with Ziptilion™. Our customers have appreciated that we can offer RTL, simulation and FPGA models for testing Ziptilion™ in their environment.

ZeroPoint

Remove the waste.
Release the power.