Compression Acceleration Use Case

Silicom's PE316ISLBEL compression adapter, based on the Intel® C620 series chipsets, as front-loading PCIe/NVMe modules enable a surge in performance

PREFACE

The strength and efficiency of data compression solutions used at various stages of the data lifecycle – that is for data at rest, data in transition and data in use – is a key differentiator for data-handling applications and the storage solutions that serve them. Intuitively, data compression in storage applications relates to data at rest. However, in contemporary storage applications, a combined data handling model is often used: not just for data at rest or for data in transit, but rather for both, in order to achieve a specific outcome, such as a more efficient hot storage solution for servicing databases and web application workloads.

> *"Silicom's PE316ISLBEL compression adapter based on the Intel® C620 series chipset, as a front-loading PCIe/NVMe module is a great example for hardware innovation that Kaminario leverages"*
>
> *Mark Shteiman, VP Product at Kaminario*

Data compression is an elaborate discipline in itself. A long list of compression algorithms were created to serve different use cases, each with distinct requirements in the areas of compression **speed** and compression **ratio**. Generally speaking, the faster the algorithm, the less efficient its compression ratio.

The LZ77 [1] lossless compression algorithm, which serves as the basis of the DEFLATE [2] algorithm and `zlib` implementations, is widely considered to strike a fair balance between speed and compression ratio. Therefore, it is widely used not only for its common purpose – that is, for web compression – but also for a myriad of other

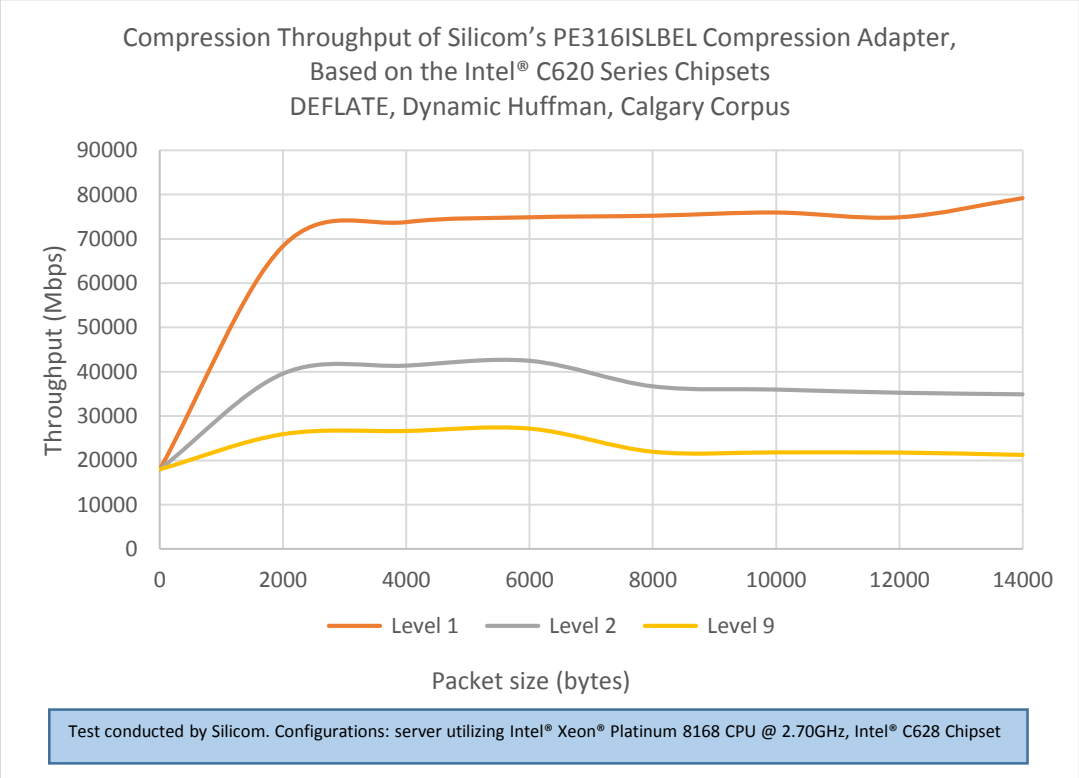

Figure 1 - Compression Throughput of Silicom's PE316ISLBEL compression adapter, based on the Intel C620 series chipsets integrating Intel® QuickAssist Technology

purposes, including tasks within big data and storage compression applications.

The LZ77 compression ratio is affected mostly by compression level [3] settings (represented by the 1-9 range, determining the length of string hash chains that form the vocabulary of the algorithm) as well as by the Huffman coding mode [4] (either static or dynamic) used. Nonetheless, the corpora themselves, meaning the nature of the data to be compressed, have a significant effect on the final compressed size.

Compression speed, on the other hand, is more related to the specific machine running the algorithm. As such, the selection of the right machine can result in an increase in the compression speed achievable at any given ratio, consequently improving the overall behavior and performance of the application.

compression adapter, based on the Intel C620 series chipsets featuring Intel QuickAssist Technology implemented by Silicom as a front-loading PCIe/NVMe module, is a great example for hardware innovation that Kaminario leverages to be one step ahead of the competition," said Mark Shteiman, VP Product at Kaminario. According to Shteiman, acceleration of inline compression via a dedicated compression engine delivers major benefits.

When Kaminario customers use LZ77 compression and Huffman coding (DEFLATE), they are able to achieve a 30% increase in the compression ratio with no impact on latency. In addition, they get industry leading, hard guarantees on total data-reduction while freeing up CPU resources to handle additional data services and customer-initiated processes.

Packet Size	Huffman Type	Compress	Decompress	Compress	Decompress
		Comp. level 1, ratio 0.41		Comp. level 2, ratio 0.40	
8192 bytes	Dynamic	**75223** Gbps	**108890** Gbps	**43014** Gbps	**109657** Gbps
		Comp. level 1, ratio 0.52		Comp. level 2, ratio 0.49	
8192 bytes	Static	**101112** Gbps	**108134** Gbps	**43986** Gbps	**108890** Gbps

Table 1- Compress/Decompress Capabilities of Silicom's PE316ISLBEL compression adapter, based on the Intel C620 series chipsets integrating Intel QuickAssist Technology

SILICOM'S PE316ISLBEL

With an impressive rate of up to 100Gbps with DEFLATE (both compression and decompression), Silicom's PE316ISLBEL compression adapter, based on the Intel® C620 series chipsets integrating Intel® QuickAssist Technology, LZ77 compression accelerator brings these algorithms to a new level of efficiency. As exhibited in Figure 1 above, tests were carried out to evaluate the performance of an Intel® Xeon® Platinum 8168 CPU @ 2.70GHz server with Silicom's PE316ISLBEL compression adapter, based on the Intel C620 series chipsets integrating Intel QuickAssist Technology. Using no more than 6 CPU cores, the compression result surged to nearly 80Gbps, while the decompression processing rate surpassed 100Gbps.

Table 1 further demonstrates how compression and decompression actions are accelerated per given packet size (8192 bytes) by Silicom's PE316ISLBEL compression adapter, with different compression levels yielding impressive compression ratios with unmatched speed.

KAMINARIO K2

Kaminario's K2 all-flash array Gen6 storage controllers are equipped with front-loading, hot-swappable, PCIe/NVMe slots that can be leveraged for innovation. "Silicom's M20E3ISLBT

SUMMARY

By freeing up of CPU cycles for an application's business logic without compromising on overall performance, the right compression solution can contribute significant value for any data handling application – and especially, following Kaminario's successful use case, for storage applications.

The extensive range of compression modes supported by Silicom's PE316ISLBEL and M20E3ISLBT compression adapters, with thier unmatched performance, makes them suitable solution for any type of data handling workload, including storage, web, big data, and database.

As such, Silicom can offer its customers the ability to reach an impressive 0.4 compression ratio at high data rates (close to 80Gbps) and to reach 100Gbps with a still-impressive 0.52 ratio – all bundled with full support for open source tools.

REFERENCES

[1] https://en.wikipedia.org/wiki/LZ77_and_LZ78
[2] IETF RFC 1951
[3] http://www.gzip.org/algorithm.txt
[4] https://en.wikipedia.org/wiki/Huffman_coding