csDF: a double-float arithmetic library for the Cerebras CS-2

Reo Nagashima, Akeru Nakamura, Kai Murakami, Ryunosuke Matsuzaki, and Takaaki Miyajima *
takaaki.miyajima@cs.meiji.ac.jp
Department of Computer Science, School of Science and Technology, Meiji University
Kawasaki-shi, Kanagawa, Japan

ABSTRACT

Recently, there have been attempts to utilize AI accelerators for scientific computing; however, these devices generally lack hardware support for double-precision floating-point arithmetic, which is essential for many scientific applications. The Cerebras CS-2 system (CS-2) delivers extremely high single-precision performance of 1.06 PFlops/s but does not support native double-precision arithmetic. To overcome this limitation and enable scientific computations requiring double precision, a software-based approach is essential. We propose csDF, a double-float (DF) arithmetic library for the CS-2 that provides DF numeric types and arithmetic operations. To demonstrate the capability of csDF, we implemented a naive pseudo-double-precision matrix multiplication using DF addition and multiplication, and measured its strong scaling performance. Our result shows 8.09 Tera DF-Flops/s, which shows the feasibility of software-based double-precision arithmetic and enables previously infeasible scientific computations.

KEYWORDS

Cerebras CS-2, Double Float computation, matrix multiplication

ACM Reference Format:
Reo Nagashima, Akeru Nakamura, Kai Murakami, Ryunosuke Matsuzaki, and Takaaki Miyajima . 2023. csDF: a double-float arithmetic library for the Cerebras CS-2. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 CEREBRAS CS-2 AND SOFTWARE DEVELOPMENT KIT

The CS-2 was initially developed for deep learning applications using the 300 mm wafer-scale chip, the Wafer Scale Engine-2 (WSE-2) [**?**]. The chip has 745,500 user-accessible Processing Elements (PEs) arranged in a grid of 750 units along the x-axis and 994 units along the y-axis. For scientific computation, the operating frequency is 850 MHz, and the maximum power consumption of the entire CS-2 is approximately 16.5 kW. Figure 1 illustrates the architecture of the PE of WSE-2. Each PE consists of a simple compute core (CE, Compute Element) and 48 KB of local memory (Scratchpad memory), which can be accessed within a single clock cycle. The network topology connecting the PEs is a two-dimensional mesh, enabling communication with the north, west, south, and east neighboring PEs within one clock cycle. Maximum performance of single-precision matrix multiplication is 349.0 TFlops/s (matrix size: 33000×33000, using 750 × 750 PEs) and a weak scaling efficiency of 1.00 [3]. The performance study of several specific scientific applications has been studied [7][9][10][2][4][8][5][6].

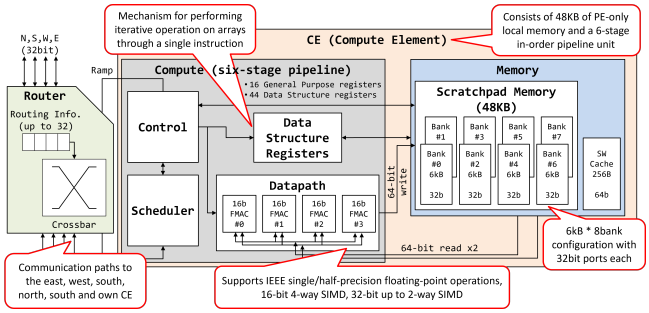

Figure 1: Architectural overview of PE in WSE-2

The Cerebras software development environment is centered around the Cerebras Hardware SDK [1], a software development kit for general-purpose computing on the CS-2. This SDK includes the CSL programming language, utility libraries, and a simulator. csDF is implemented using the SDK. The CSL can describe tasks and communication running on the WSE-2. Host-side code is written in Python and provides runtime functions such as data transfer.

2 CSDF AND PSEUDO-DOUBLE-PRECISION MATRIX MULTIPLICATION

Significant efforts have been made to emulate high-precision data types using low-precision data types. QD library [1] provides double-double and quad-double numeric types and arithmetic for CPU. They have approximately twice and four times the precision of the IEEE double-precision format, respectively. We implemented the csDF library for the CS-2, based on the QD library. It provides DF numeric types and arithmetic, which are approximately twice as precise as the IEEE single-precision format. The DF type achieves about 48 bits of precision from its fraction, while the 8-bit exponent defines the scale. This library implements fundamental arithmetic operations such as addition, subtraction, multiplication, division, and square root. It also supports commonly used transcendental functions, including exponential, logarithmic, trigonometric, and hyperbolic functions.

To demonstrate the capability of csDF, we implemented a naive pseudo-double-precision matrix multiplication and measured the strong scaling on the CS-2. List 1 shows a pseudo-code of the DF matrix multiplication. DF multiplication (`df_real.mul`) and addition (`df_real.add`) are used to perform the dot product and accumulate.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

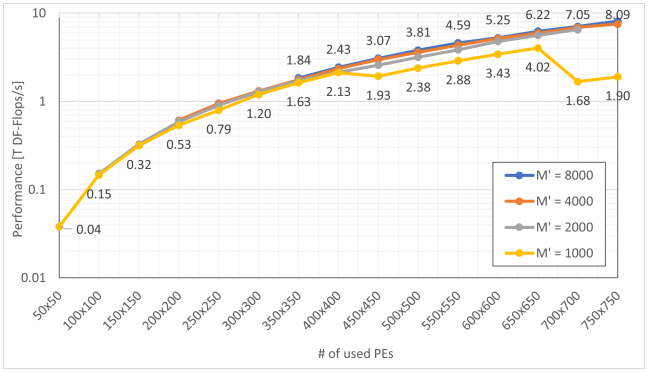

Figure 2: Strong scaling: Trends of computational performance at $M' = 1000$ to 8000.

Listing 1: Pseudo-code of double-float matrix multiplication.

```
// Import csDF library
const csdf = @import_module("csdf/csdf.csl");
// Loop over each row of DF matrix A
for (@range(i16, Kt)) |k|{
  // Loop over each column of DF matrix B
  for (@range(i16, Nt)) |j|{
    // Make a DF type (df_real) from two 32-bit values
    B[0] = csdf.get_df_real(B[j*Kt+k,0], B[j*Kt+k,1]);
    // Compute the dot product of A and B
    for (@range(i16, Mt)) |i|{
      // Accumulate the product of corresponding elements
      product[i] = csdf.mul(A[i+z], B[0]);
      C[i+x] = csdf.add(C[i+x], product[i]);
    x = x + Mt;}
  z = z + Mt;}
}
```

For the strong scaling experiment, the number of PEs used varies from 50×50 to 750×750 in increments of 50. The overall matrix size is $M \times M$, and each PE computes a submatrix of size $M_t \times M_t$. Since M may not be divisible by the number of PEs, we define a target matrix size M', which differs slightly from M. Here, M' is calculated as $M' = M_t \times P$, where P is the number of PEs along the x-axis. The hardware cycle counter was used for the time measurement. The elapsed time was calculated as the elapsed cycles divided by the operating frequency of 850 MHz. The computational performance was calculated as the number of DF floating point operations (DF-Flop), $2 \times M^3$, divided by the elapsed time. Figure 2 shows the computational performance trends for each matrix size. A maximum of 8.09 T DF-Flops/s was achieved, which is approximately 1/43 of the maximum performance of SGEMM [3].

3 SUMMARY

In this paper, we presented csDF, a double-float arithmetic library designed for the CS-2, which lacks native hardware support for double-precision floating-point operations. Based on the Quad Double Computation Package, csDF provides double-float numeric types and arithmetic operations, enabling pseudo-double-precision computations on the CS-2 platform. To demonstrate the practical capability of csDF, we implemented a naive pseudo-double-precision matrix multiplication and evaluated its strong scaling performance on the CS-2. Our results show that csDF achieves a maximum performance of 8.09 Tera DF-Flops/s. This work highlights the feasibility of software-based high-precision arithmetic on the CS-2, paving the way for further exploration of scientific computing.

ACKNOWLEDGEMENT

This work was supported by Japan Science and Technology Agency (JST) as part of Adopting Sustainable Partnerships for Innovative Research Ecosystem (ASPIRE), Grant Number JPMJAP2341. This work was supported by JSPS KAKENHI Grant Number 24K14972. This research used resources of the Argonne Leadership Computing Facility, a U.S. Department of Energy (DOE) Office of Science user facility at Argonne National Laboratory and is based on research supported by the U.S. DOE Office of Science-Advanced Scientific Computing Research Program, under Contract No. DE-AC02-06CH11357.

REFERENCES

[1] DavidH. Bailey, XiaoyeS. Li, and Yozo Hida. 2003. QD: A Double-Double/Quad-Double Package. https://doi.org/10.11578/dc.20210416.14

[2] Piotr Luczynski, Lukas Gianinazzi, Patrick Iff, Leighton Wilson, Daniele De Sensi, and Torsten Hoefler. 2024. Near-Optimal Wafer-Scale Reduce. In *Proceedings of the 33rd International Symposium on High-Performance Parallel and Distributed Computing* (Pisa, Italy) *(HPDC '24)*. Association for Computing Machinery, New York, NY, USA, 334–347. https://doi.org/10.1145/3625549.3658693

[3] Ryunosuke Matsuzaki, Daichi Mukunoki, and Takaaki Miyajima. 2025. Performance evaluation and modelling of single-precision matrix multiplication on Cerebras CS-2. In *Proceedings of the SC '24 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis* (Atlanta, GA, USA) *(SC-W '24)*. IEEE Press, 727–731. https://doi.org/10.1109/SCW63240.2024.00101

[4] Matthew Andres Moreno, Connor Yang, Emily Dolson, and Luis Zaman. 2024. Trackable Island-model Genetic Algorithms at Wafer Scale. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (Melbourne, VIC, Australia) *(GECCO '24 Companion)*. Association for Computing Machinery, New York, NY, USA, 101–102. https://doi.org/10.1145/3638530.3664090

[5] Marcelo Orenes-Vera, Ilya Sharapov, Robert Schreiber, Mathias Jacquelin, Philippe Vandermersch, and Sharan Chetlur. 2023. Wafer-Scale Fast Fourier Transforms. In *Proceedings of the 37th ACM International Conference on Supercomputing* (Orlando, FL, USA) *(ICS '23)*. Association for Computing Machinery, New York, NY, USA, 180–191. https://doi.org/10.1145/3577193.3593708

[6] Kamil Rocki, Dirk Van Essendelft, Ilya Sharapov, Robert Schreiber, Michael Morrison, Vladimir Kibardin, Andrey Portnoy, Jean Francois Dietiker, Madhava Syamlal, and Michael James. 2020. Fast Stencil-Code Computation on a Wafer-Scale Processor. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Atlanta, Georgia) *(SC '20)*. IEEE Press, Article 58, 14 pages.

[7] Ryuichi Sai, François P. Hamon, John Mellor-Crummey, and Mauricio Araya-Polo. 2024. Matrix-Free Finite Volume Kernels on a Dataflow Architecture. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis* (Atlanta, GA, USA) *(SC '24)*. IEEE Press, Article 28, 11 pages. https://doi.org/10.1109/SC41406.2024.00034

[8] Ryuichi Sai, Mathias Jacquelin, Francois Hamon, Mauricio Araya-Polo, and Randolph R. Settgast. 2023. Massively Distributed Finite-Volume Flux Computation. In *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis* (Denver, CO, USA) *(SC-W '23)*. Association for Computing Machinery, New York, NY, USA, 1713–1720. https://doi.org/10.1145/3624062.3624252

[9] Ryuichi Sai, John Mellor-Crummey, Jinfan Xu, and Mauricio Araya-Polo. 2024. Automated Code Generation of High-Order Stencils for a Dataflow Architecture. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis* (Atlanta, GA, USA) *(SC '24)*. IEEE Press, Article 19, 13 pages. https://doi.org/10.1109/SC41406.2024.00025

[10] Shihui Song, Yafan Huang, Peng Jiang, Xiaodong Yu, Weijian Zheng, Sheng Di, Qinglei Cao, Yunhe Feng, Zhen Xie, and Franck Cappello. 2024. CereSZ: Enabling and Scaling Error-bounded Lossy Compression on Cerebras CS-2. In *Proceedings of the 33rd International Symposium on High-Performance Parallel and Distributed Computing* (Pisa, Italy) *(HPDC '24)*. Association for Computing Machinery, New York, NY, USA, 309–321. https://doi.org/10.1145/3625549.3658691