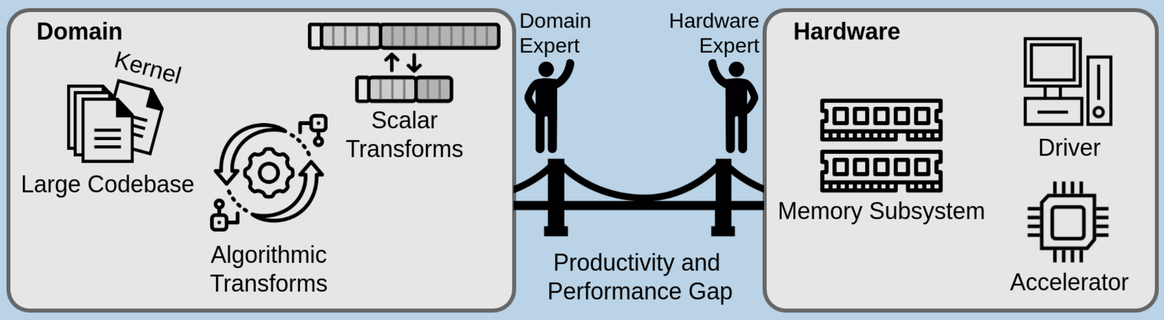
Olympus: Design Methods for Simplifying the Creation of Domain-Specific Memory Architectures

POLITECNICO MILANO 1863

Stephanie Soldavini, *Advisor:* Christian Pilato
stephanie.soldavini@polimi.it, christian.pilato@polimi.it

EVEREST

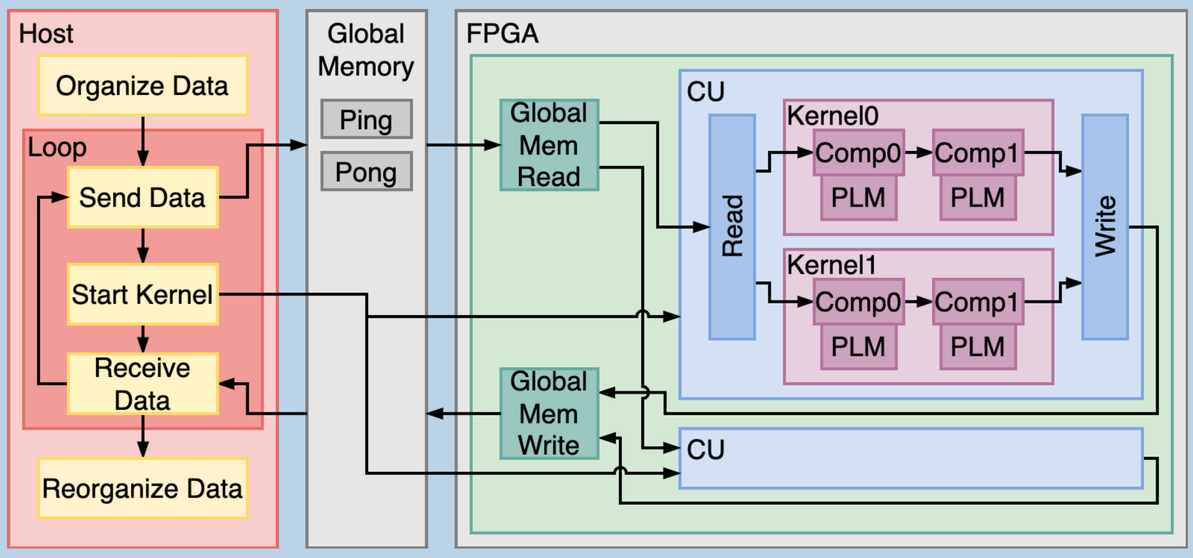
Problem



Specialized hardware accelerators can have **high performance** and **energy efficiency**, but their design are very **complex and time consuming**, especially for **big data** and **machine learning** applications [2] and on platforms with **high-bandwidth memory (HBM)**.

This complexity means the designer not only has to optimize the accelerator computation logic, but also has to **carefully craft efficient memory architectures**, which is not the case in traditional software design [5].

Olympus



Olympus automatically integrates many features targeted at solving six challenges of the hardware-software gap [4].

(1) Input Languages and Frameworks: *How can application designers exploit hardware acceleration while using high level frameworks?*

Vitis HLS C code and HDL are accepted. They can be handcrafted or be generated by domain-specific language compilers or other HLS tools.

(2) CPU-Host Communication Cost: *How to minimize Host-FPGA data transfer time so hardware acceleration is advantageous?*

While the host is exchanging data with the Ping (Pong) HBM channels, the compute units (CUs) can use the Pong (Ping) channels.

(3) Read/Write Burst Transactions: *How can the data be reorganized to gain the maximum performance?*

The HBM channels are filled with as much data as possible and input and output data are separated each channel only moves data in a single direction.

(4) Full Bandwidth Utilization: *How can the many wide channels of emerging memory technologies be maximally and effectively leveraged?*

The memory channels are divided into "lanes" and the kernel is replicated so each uses one lane. For instance, with 64-bit data, the 256-bit bus is divided into four lanes and four kernels are instantiated.

(5) Data Allocation: *Can custom data layouts maximize area efficiency?*

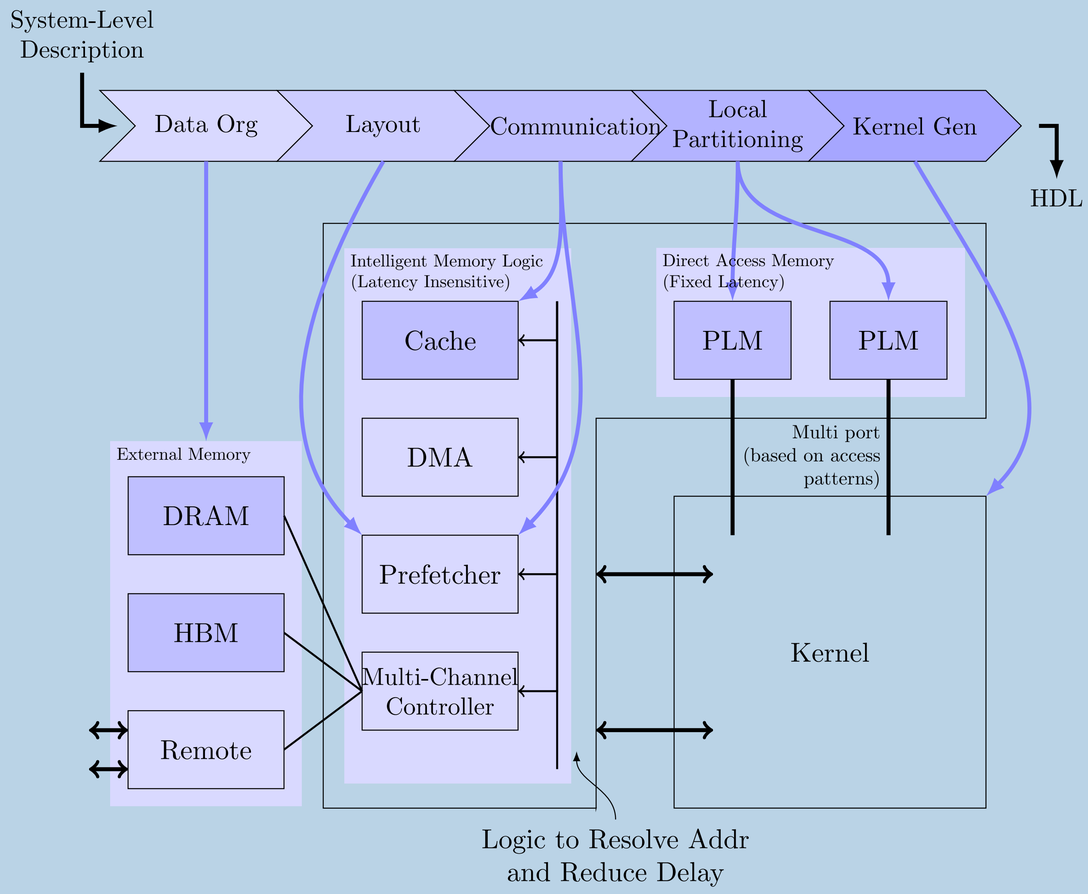
Mnemosyne [3] is used to create an efficient private local memory (PLM) architecture using up to 42% less BRAM [1].

(6) Synthesis-Related Issues: *How to trade-off between optimizing few kernels or creating many in parallel?*

A single CU performs better than several parallel CUs, due to the frequency downscaling caused by the routing congestion around the interface to HBM.

Proposed Flow

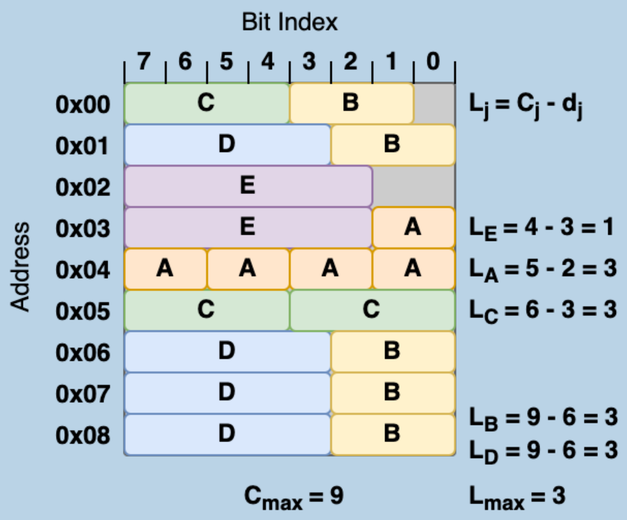
We aim to create a multi-level compilation flow [6] that specializes a domain-specific memory template to match data, application, and technology requirements in order to simplify the hardware accelerator development process.



Iris

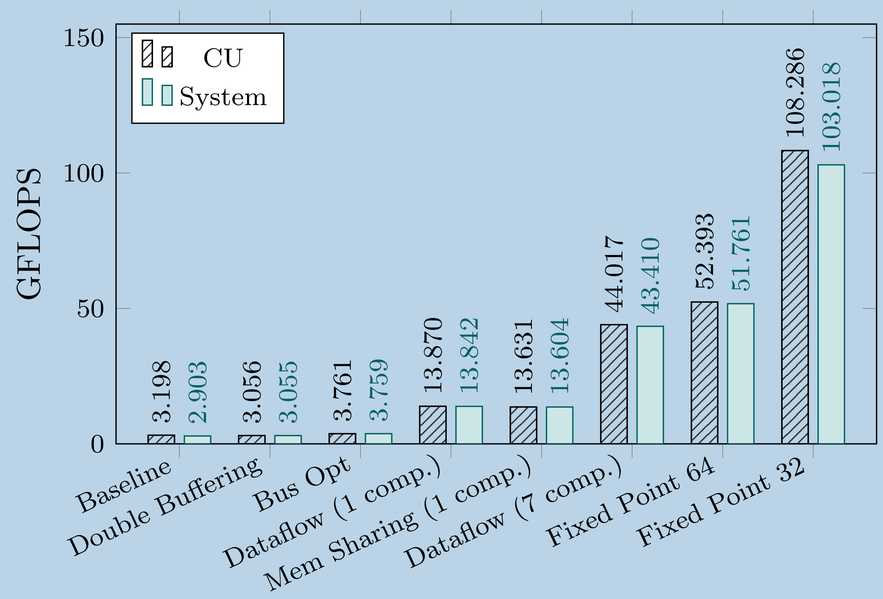
Iris [7] is a method for creating a data layout for **unconventional data widths** which **minimizes the data transfer time** between global memory and the accelerator.

- Inspired by processor scheduling
- Arrays are "preemptible tasks"
- Can be split and interleaved in a data transfer "schedule"
- Optimizes so array is available to its compute unit as soon as possible after it is needed



- Schedules 6.4% more efficient than the naive method, which over millions of iterations can be significant
- FPGA logic uses up to 33.3% smaller data FIFOs than the naive method

Results



Performance of the Inverse Helmholtz operator with Olympus optimizations (applied incrementally) [4].

References

[1] F. A. Karl Friebel, S. Soldavini, et al. "From Domain-Specific Languages to Memory-Optimized Accelerators for Fluid Dynamics". In: *CLUSTER*. 2021.

[2] C. Pilato, S. Bohm, et al. "EVEREST: A design environment for extreme-scale big data analytics on heterogeneous platforms". In: *DATE*. 2021.

[3] C. Pilato, P. Mantovani, et al. "System-Level Optimization of Accelerator Local Memory for Heterogeneous Systems-on-Chip". In: *TCAD* 36.3 (2017).

[4] S. Soldavini, K. F. A. Friebel, et al. "Automatic Creation of High-Bandwidth Memory Architectures from Domain-Specific Languages: The Case of Computational Fluid Dynamics". In: *ACM TRETS* (Sept. 2022).

[5] S. Soldavini and C. Pilato. "A Survey on Domain-Specific Memory Architectures". In: *JICS* 16.2 (Aug. 2021).

[6] S. Soldavini and C. Pilato. *Compiler Infrastructure for Specializing Domain-Specific Memory Templates*. 2021.

[7] S. Soldavini, D. Sciuto, et al. "Iris: Automatic Generation of Efficient Data Layouts for High Bandwidth Utilization". In: *ASPDAC*. Accepted. Tokyo, Japan: ACM, 2023.

Acknowledgements

This work is partially funded by the EU Horizon 2020 Programme under grant agreement No 957269 (EVEREST).