Acceleration by Separate-Process Cache for Memory-Intensive Algorithms on FPGA via High-Level Synthesis

Master's Degree Thesis

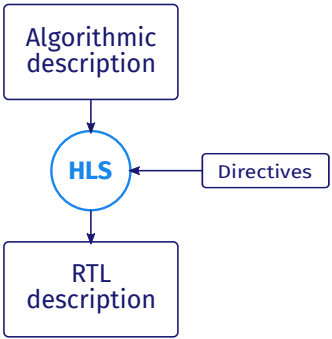
Brignone Giovanni

Supervisor: Prof. Lavagno Luciano

October 21, 2021

Politecnico di Torino

High-Level Synthesis



Productivity:

- Functionality
- Debugging
- Design space exploration

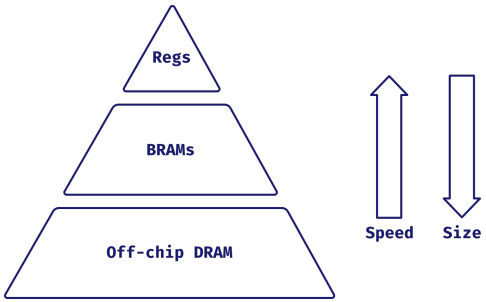
Memory management

Scratchpad:

- Manual memory selection
- HLS state of the art

Cache:

- Automatically exploit whole hierarchy
- Thesis work objective



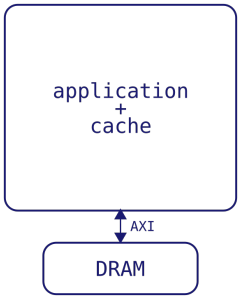
Previous work

Architecture:

- Cache inlined in application
- One cache per DRAM array

Implementation:

- `C++` class
- User friendliness
  - *Integrability*: `operator[]` overload
  - *Configurability*: template parameters
  - *Observability*: hit ratio reports
- Application logic cluttering



Thesis work

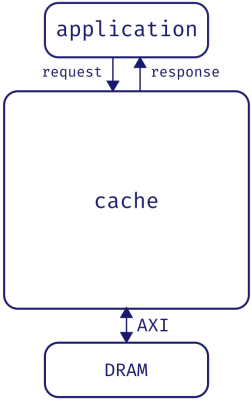
Basic architecture

Objective:

- Limit application cluttering

Proposed solution:

- *Cache:* separate process
  - Modeling: threads (SW), dataflow (HW)
  - Communication: FIFOs
- *Application:*
  1. Send *request*
  2. Receive *response*



Basic architecture — Implementation

Objective:

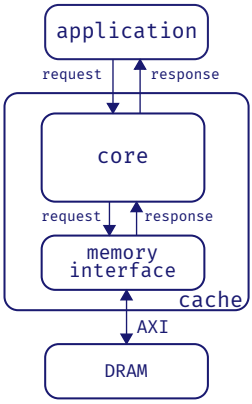
- Optimally pipeline cache hits (II=1)

Problem:

- Dependencies on DRAM interface

Proposed solution:

- Multi-process architecture:
  - *Core*: cache functionality
  - *Memory interface*: DRAM accesses (miss only)



Optimizations

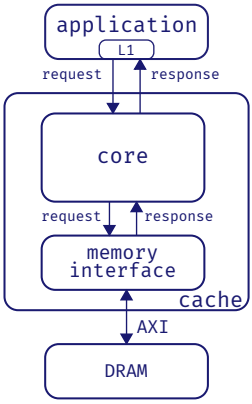
Multi-levels architecture

Objective:

- Reduce read latency

Proposed solution:

- L1 cache inlined in application
  - Fast: no communication overhead
  - Simple: direct mapped, write-through



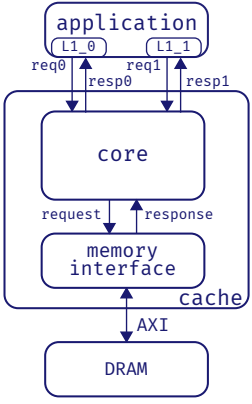
Multi-ports architecture

Objective:

- Execute multiple reads in parallel

Proposed solution:

- Single L2 cache (with multiple ports)
- Multiple L1 caches (one per L2 port)



Results

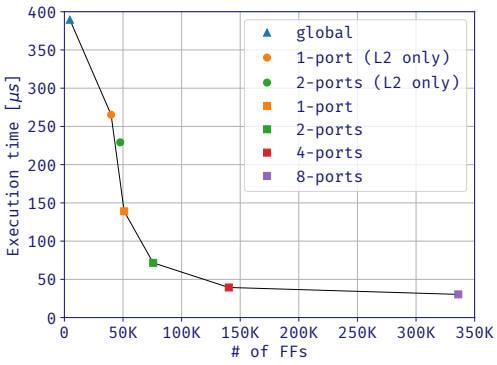
Matrix multiplication

Algorithm:

$$C = A \times B, \quad A, B, C \in \mathbb{R}^{32 \times 32}$$

Caches configuration:

- A: 1 line of 32 elements
- B: 32 lines of 32 elements
- C: 1 line of 32 elements



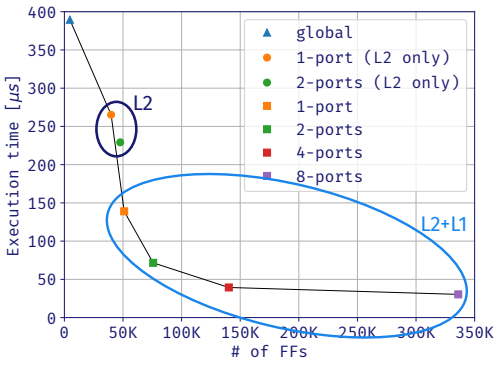
Matrix multiplication

Algorithm:

$$C = A \times B, \quad A, B, C \in \mathbb{R}^{32 \times 32}$$

Caches configuration:

- *A*: 1 line of 32 elements
- *B*: 32 lines of 32 elements
- *C*: 1 line of 32 elements



Summary

Summary

Achieved results:

- Multi-process modeling for HLS
- Design space extended by cache

Future work:

- RTL implementation
- Pre-fetching

References

Ma, L., Lavagno, L., Lazarescu, M., & Arif, A. (2017). Acceleration by inline cache for memory-intensive algorithms on fpga via high-level synthesis. *IEEE Access*, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2017.2750923

Xilinx Inc. (2021). *Vitis high-level synthesis user guide*. https://www.xilinx.com/support/documentation/sw_manuals/xilinx2021_1/ug1399-vitis-hls.pdf

Interface issues

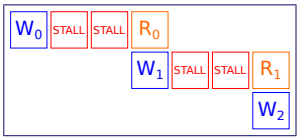
Cache process:

- Optimal pipeline (II=1)

Interface:

- Scheduler unaware of latency between *request* and *response*
- Workaround: forced clock cycles

Original scheduling



Scheduling with workaround