Evaluation of Data-Intensive Accelerated Functions in Computational Storage Devices

Jiong Liu
Advisor: Charalampos Mainas, Atsushi Koshiba
Chair of Distributed Systems and Operating Systems
https://dse.in.tum.de/

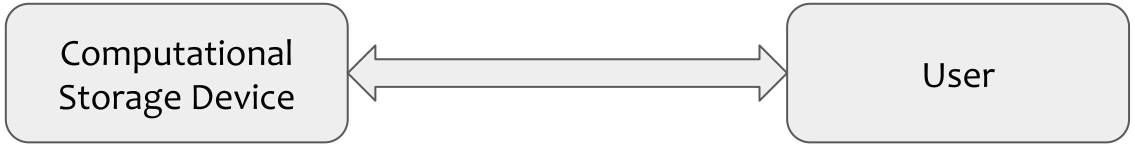
15.10.2022 – 15.02.2023

Motivation

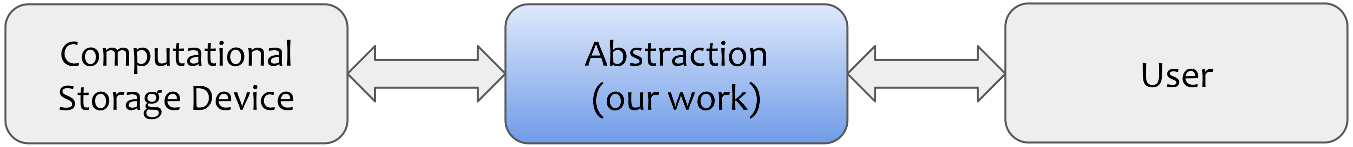
- **CPU resources:** limited and precise
- **Computational storage devices:** suitable and near data
- Idea: unload data-intensive tasks to computational storage devices

Current Issues

- Rare appropriate abstraction
- Hardware knowledge necessary -> difficult and long learning curve



Introduction



Add an abstraction between user and device

System design goals:

- High level
- Performance

System target:

Xilinx Zynq UltraScale+ MPSoC

ZCU106 Evaluation Board

Outline

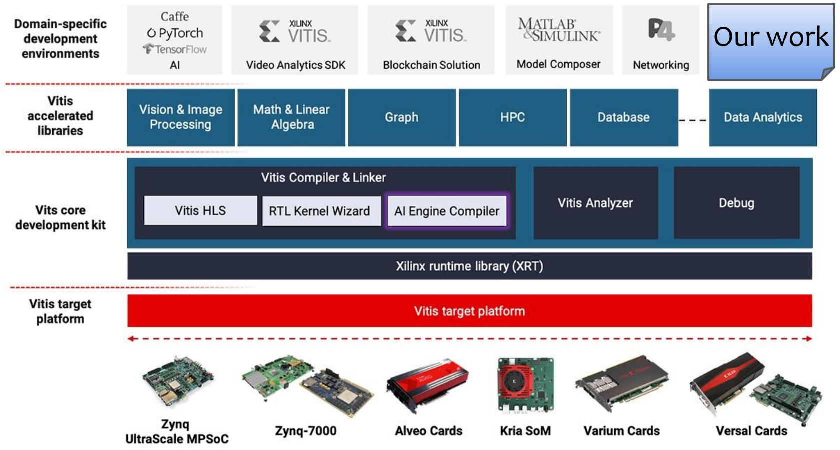
- ~~Introduction~~
- Background
    - Vitis
- Design
- Implementation
- Evaluation

Vitis

- Unified Platform for Xilinx Devices
- shares a similar design- and workflow

Vitis Accelerated Library

- sample code for different areas
- Security & Data Compression

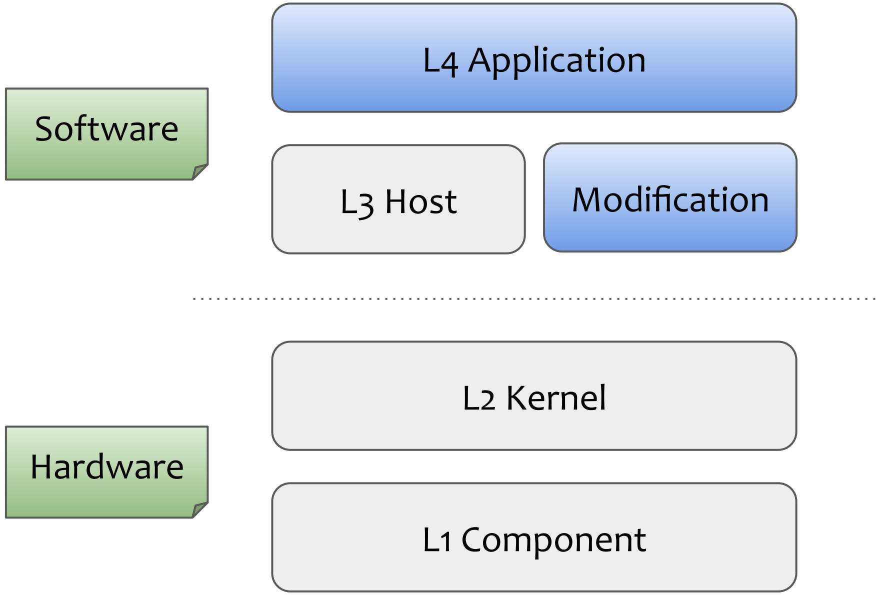


(https://www.xilinx.com/products/design-tools/vitis/vitis-platform.html)

Outline

- ~~Introduction~~
- ~~Background~~
- Design
  - System Overview
  - Design Details
- Implementation
- Evaluation

System overview



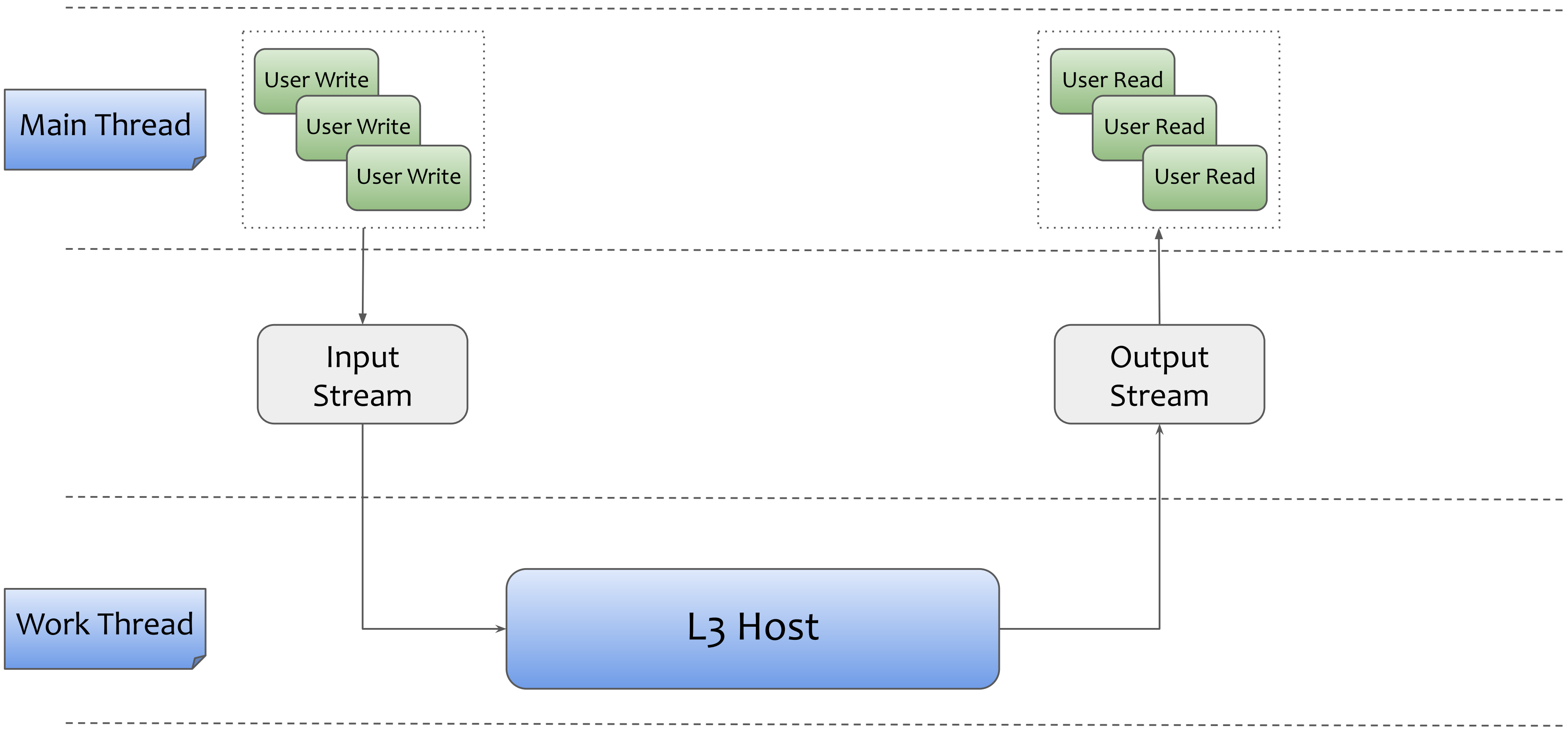
➔ Global Distribution & High Level API

➔ Data Block processing & Communication with Hardware

➔ Hardware Executable Unit & Wrapper

➔ Hardware Function Unit

Design detail: Data Compression Library



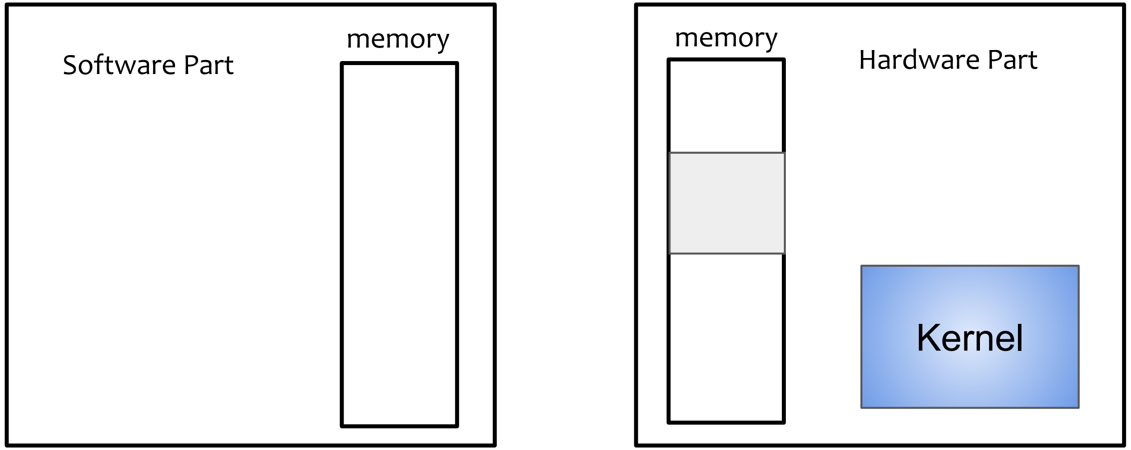
Outline

- ~~Introduction~~
- ~~Background~~
- ~~Design~~
- Implementation
- Evaluation

Implementation: Hardware Interaction

Vitis Execution Model

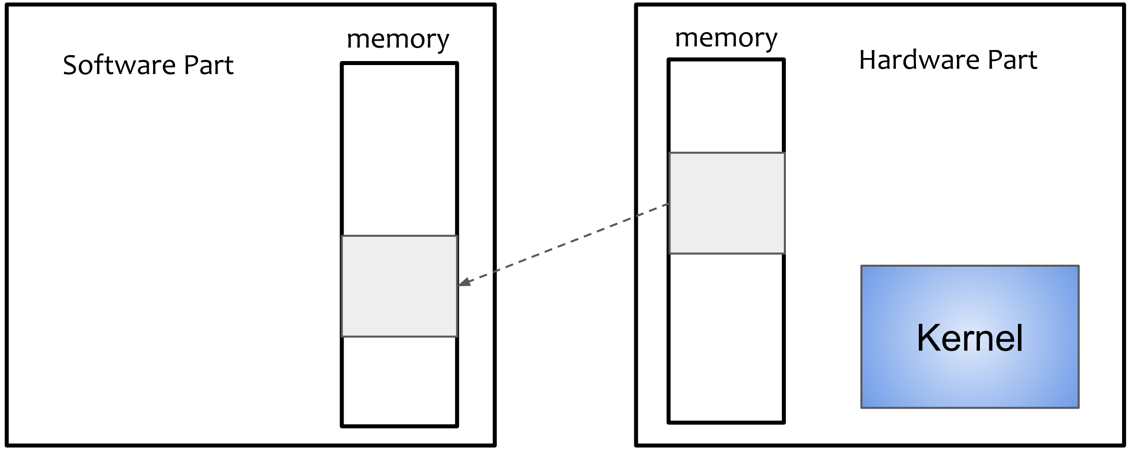
0. Buffers allocation and virtual mapping



Implementation: Hardware Interaction

Vitis Execution Model

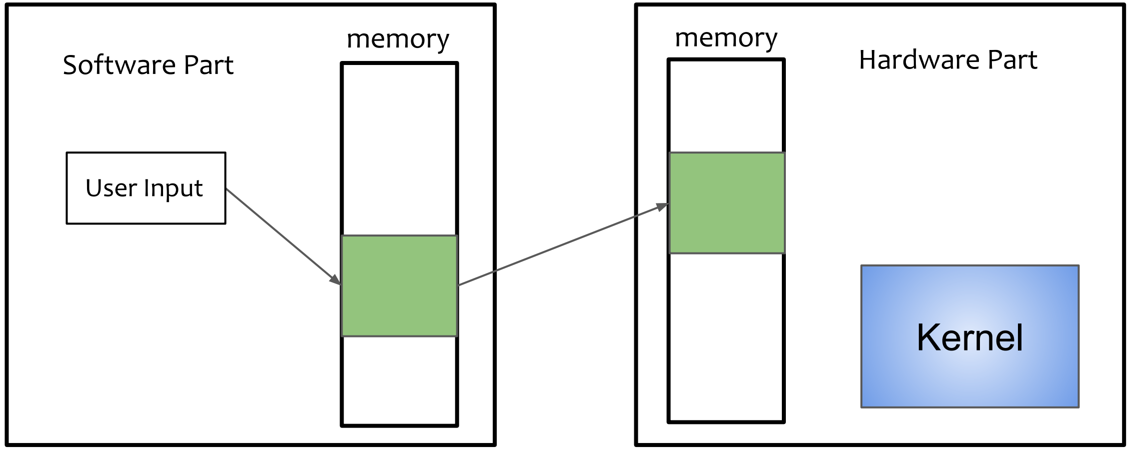
0. Buffers allocation and virtual mapping



Implementation: Hardware Interaction

Vitis Execution Model

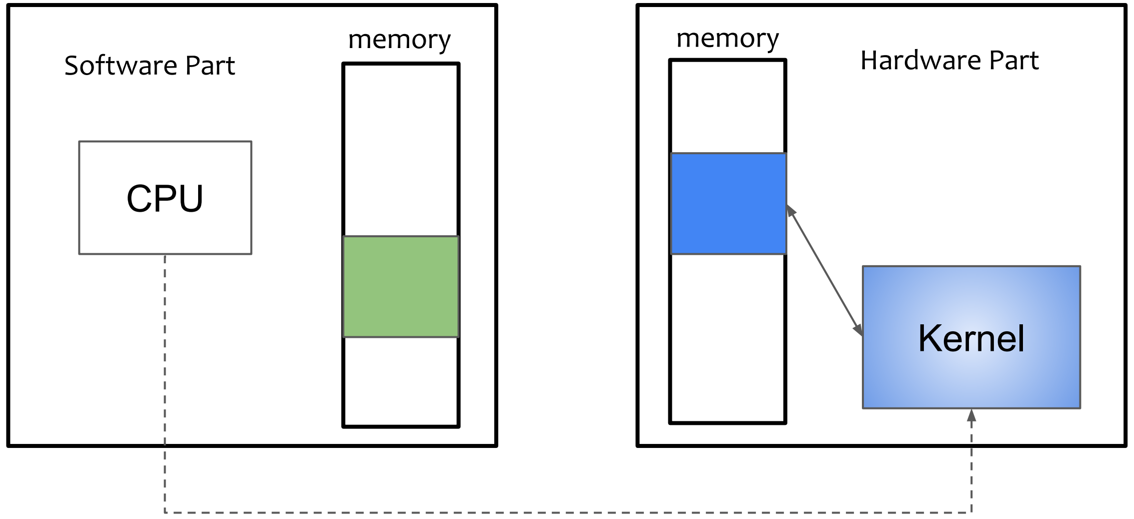
1. Migration of data to hardware



Implementation: Hardware Interaction

Vitis Execution Model

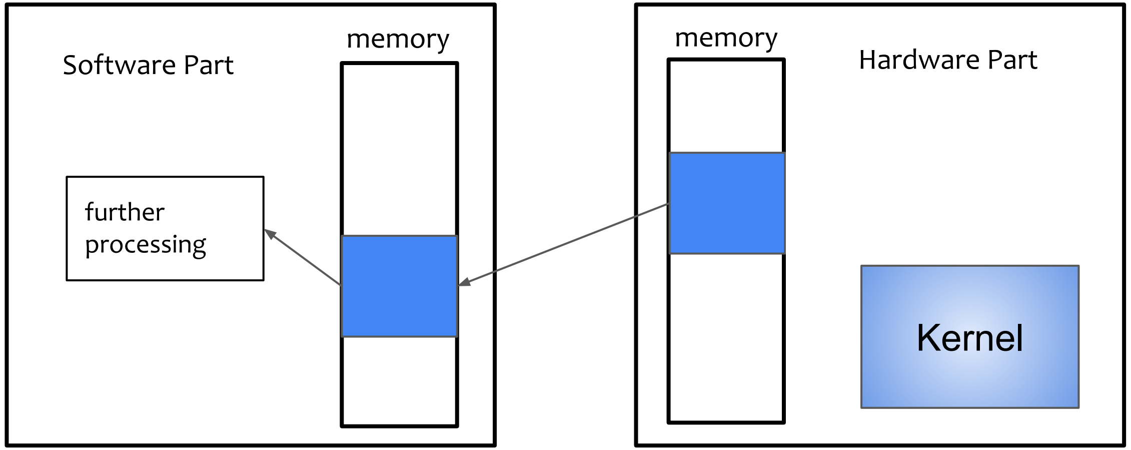
2. Kernel execution



Implementation: Hardware Interaction

Vitis Execution Model

3. Migration of data back from hardware



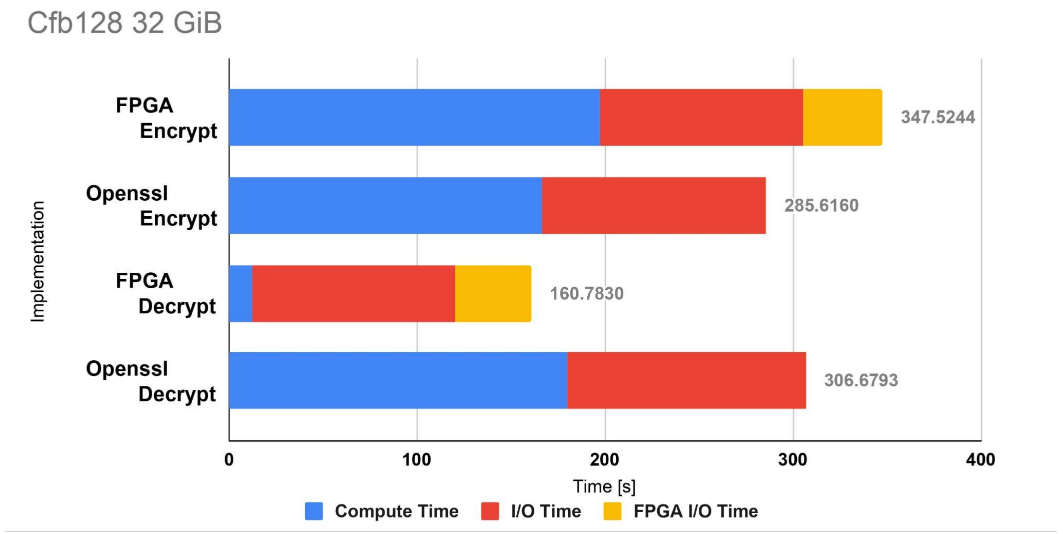
Outline

- ~~Motivation~~
- ~~Background~~
- ~~Design~~
- ~~Implementation~~
- Evaluation

Evaluation

- Test setup:
  - Intel Xeon Gold 6238R CPU (2.20 GHz, 24 cores)
  - Target Board not available -> Xilinx Alevo U50
- Benchmarks:
  - Security: OpenSSL
  - Data Compression: Linux Utility

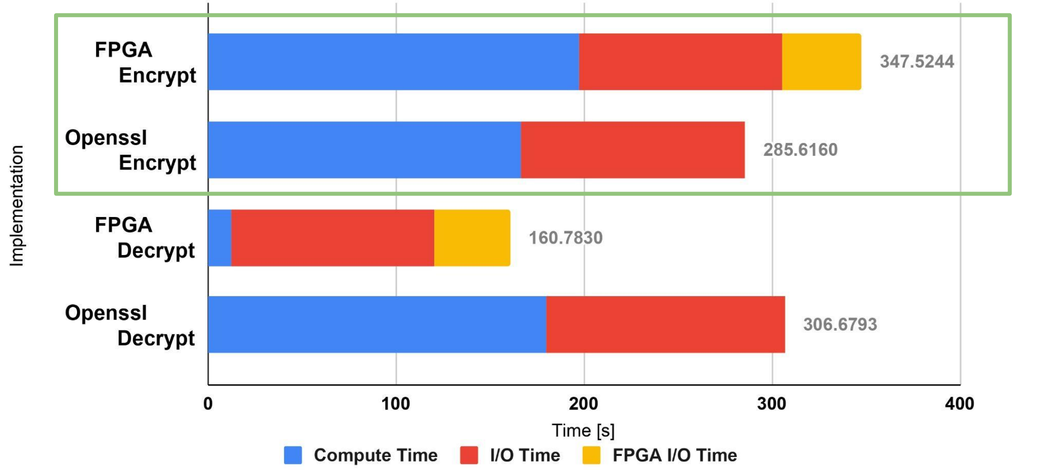
Evaluation: Security Library



Evaluation: Security Library

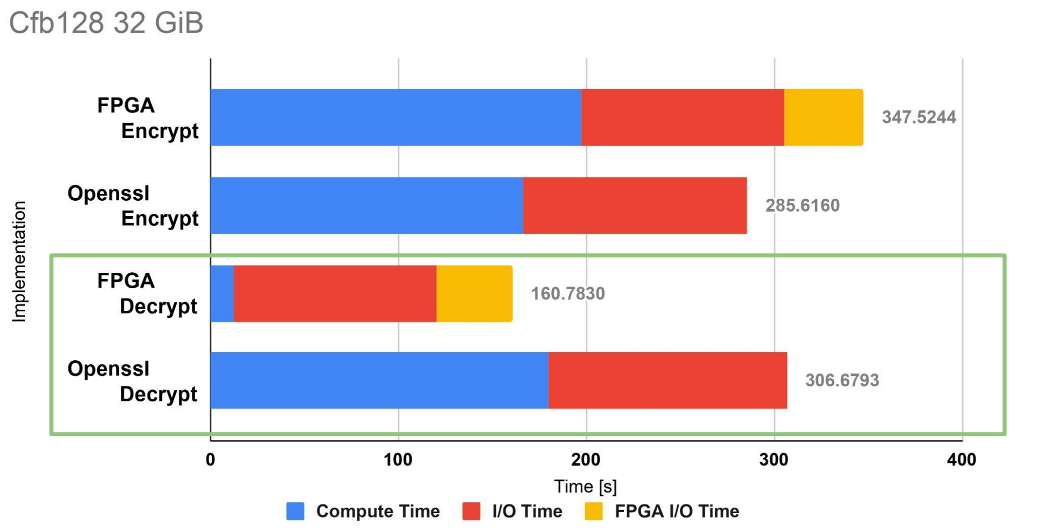
↑ 29%

Cfb128 32 GiB



Evaluation: Security Library

↓ 48%

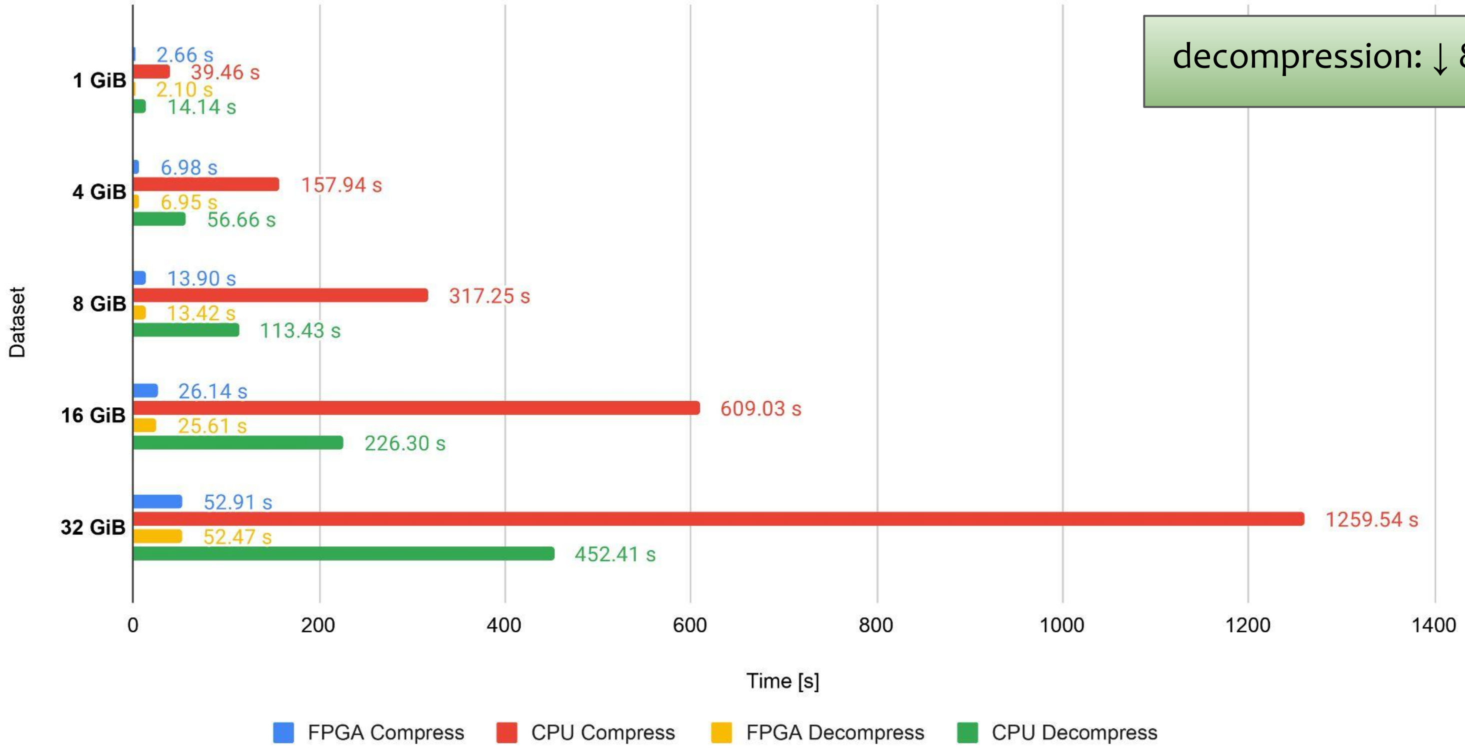


Evaluation: Data Compression Library

compression: ↓ 95%

decompression: ↓ 85%

Gzip



Summary

- Some hardware-accelerated data-intensive functions are ported
- high level abstraction
- acceptable performance, sometimes better

Future Work:

- Evaluation on the target board is still necessary

Try it out!

https://github.com/Ljiong201108/bsc-project.git

Backup

Design details: Security Library

L3 Host

- Execution of hardware encryption and decryption
- Interaction with hardware

L4 Application

- Initialization of current data block processing
- post-processing

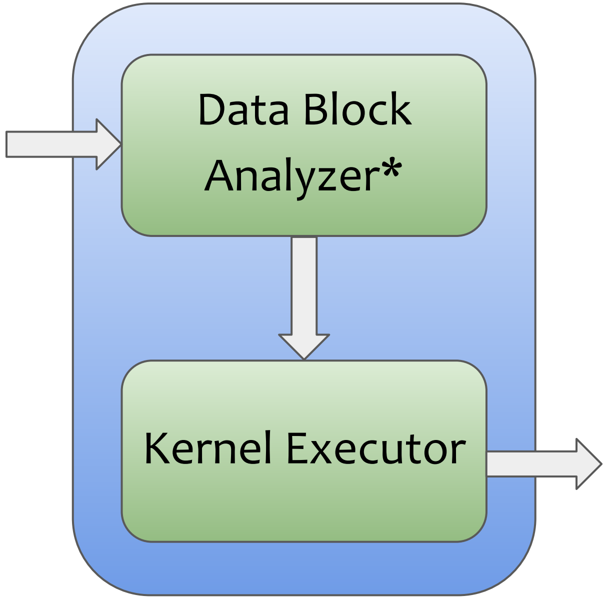
Design details: Data Compression Library(L3 Host)

Data Block Analyzer(only for decompression)

- Analyse the given data block
- Delete the meta-data

Kernel Executor

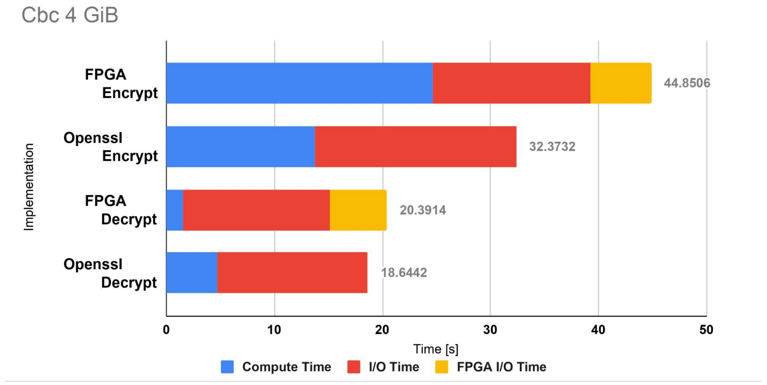
- Executes the hardware compression / decompression

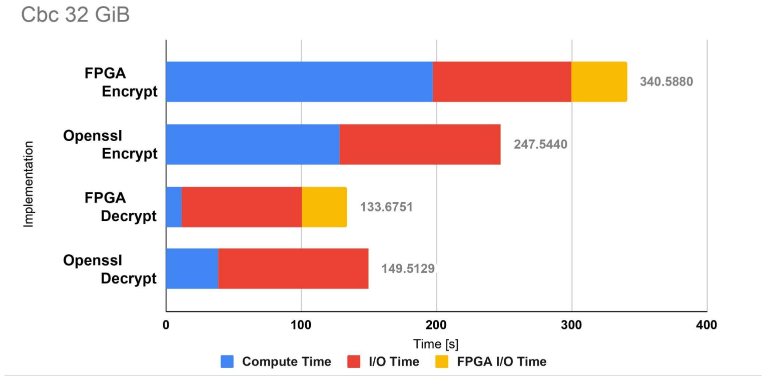


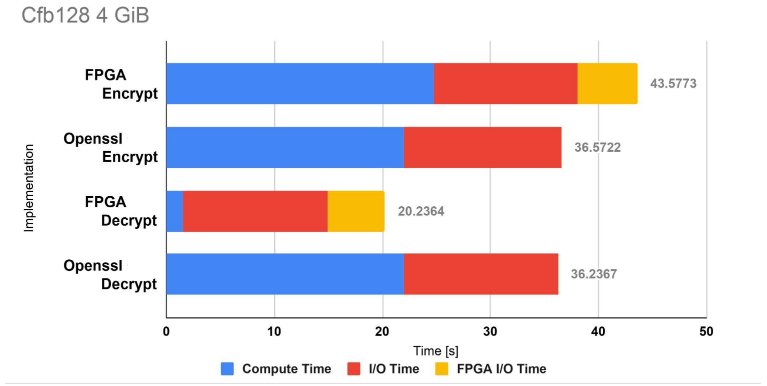
Evaluation: Security Library Overview

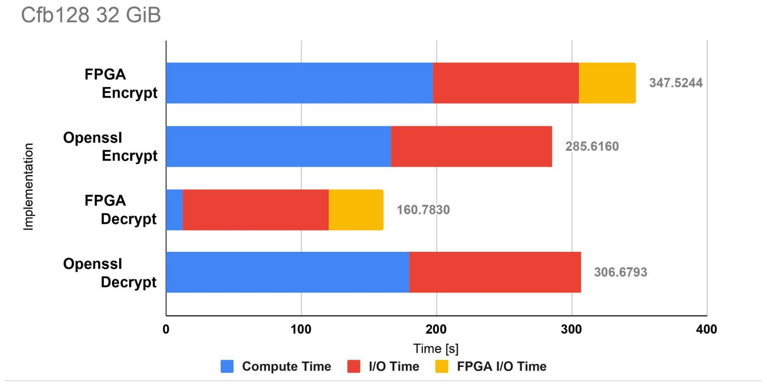
- AES
- 4 GiB & 32 GiB

Evaluation: Security Library (Block dependent)





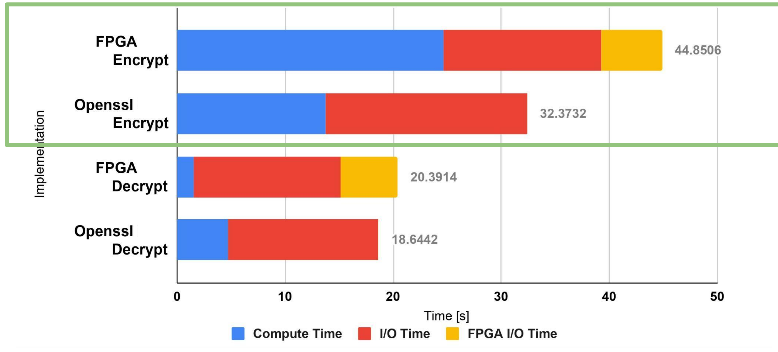




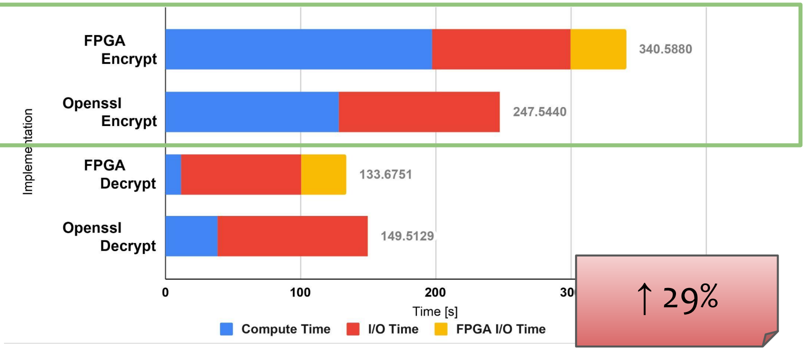
Evaluation: Security Library (Block dependent)

↑ 35%

Cbc 4 GiB

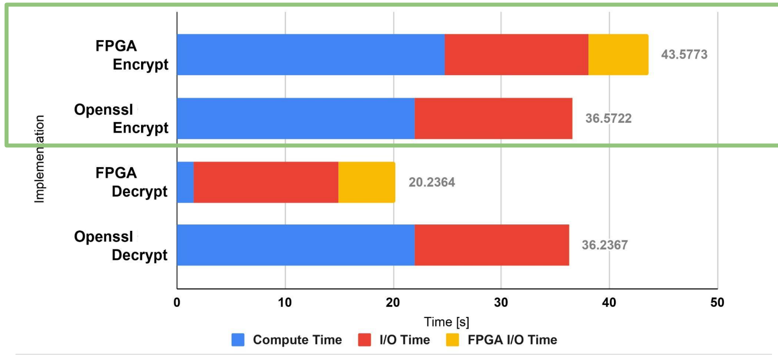


Cbc 32 GiB

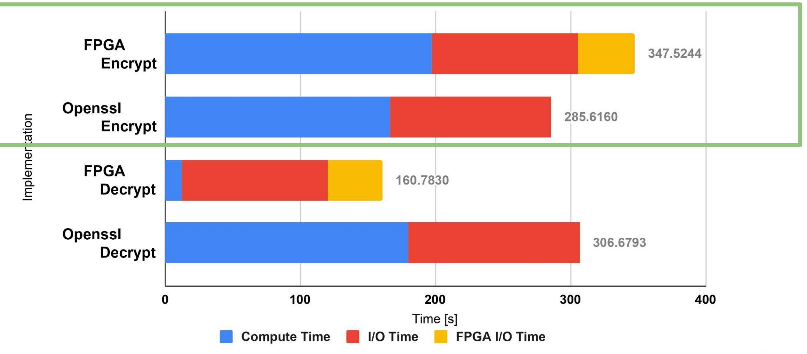


↑ 29%

Cfb128 4 GiB



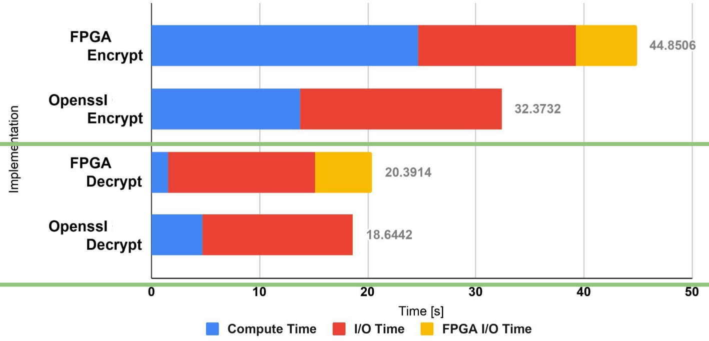
Cfb128 32 GiB



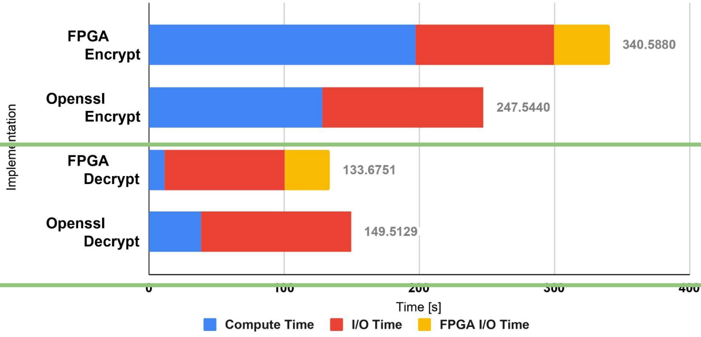
Evaluation: Security Library (Block dependent)

~ 0%

Cbc 4 GiB

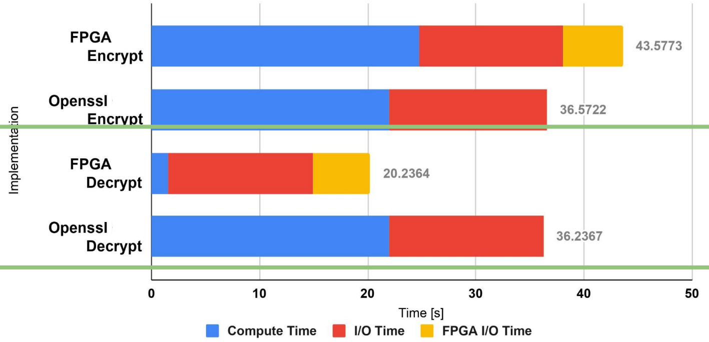


Cbc 32 GiB

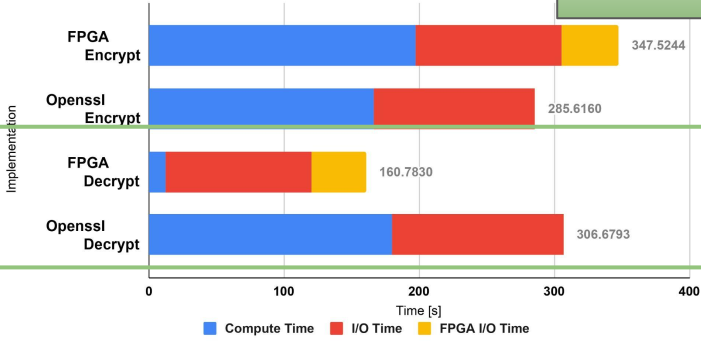


↓ 48%

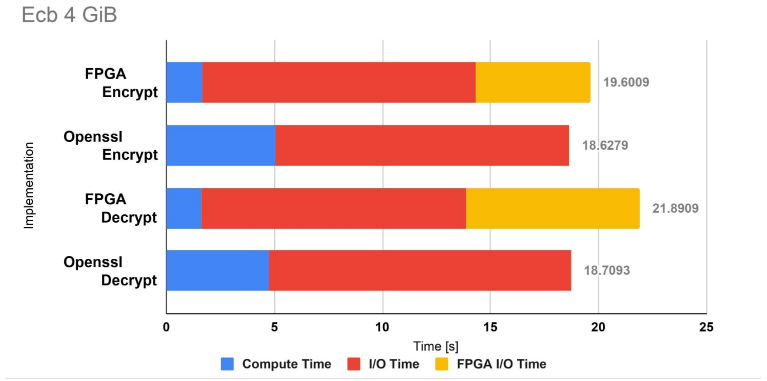
Cfb128 4 GiB

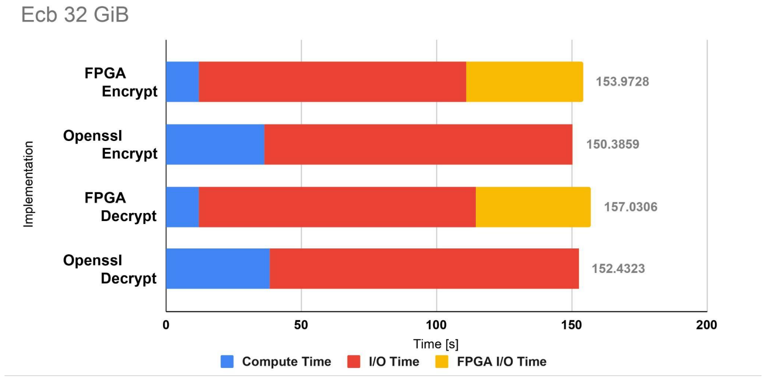


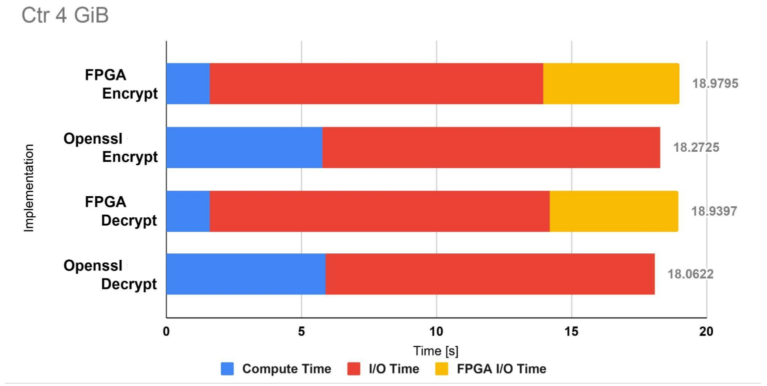
Cfb128 32 GiB

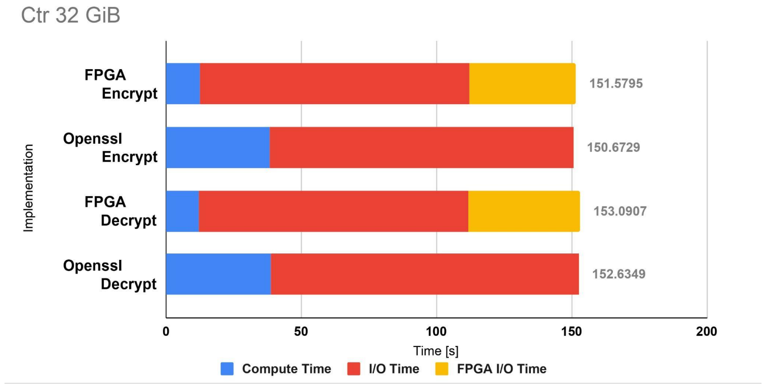


Evaluation: Security Library (Block Independent)

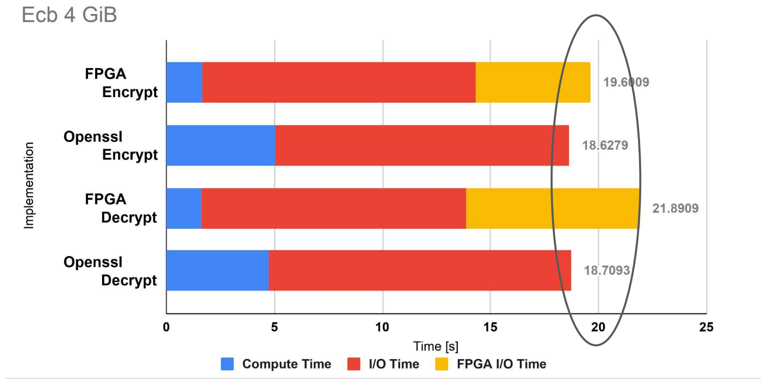


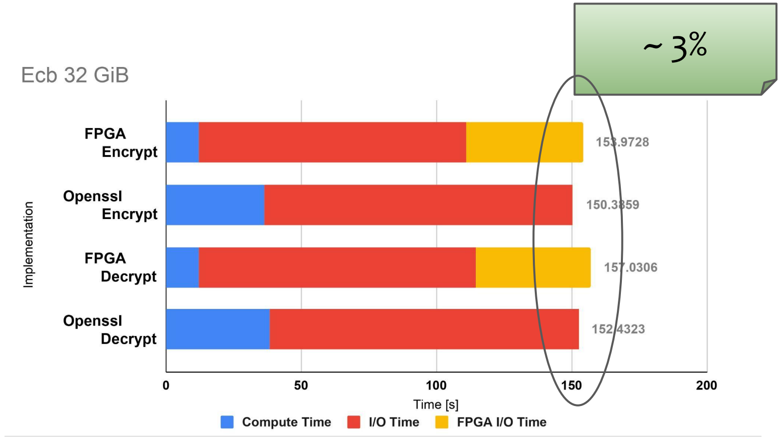


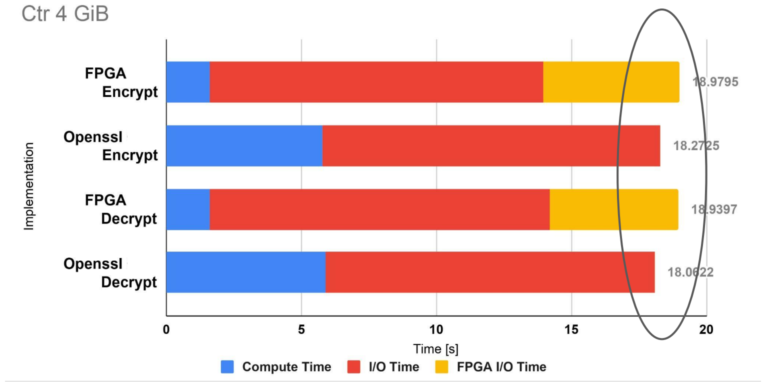


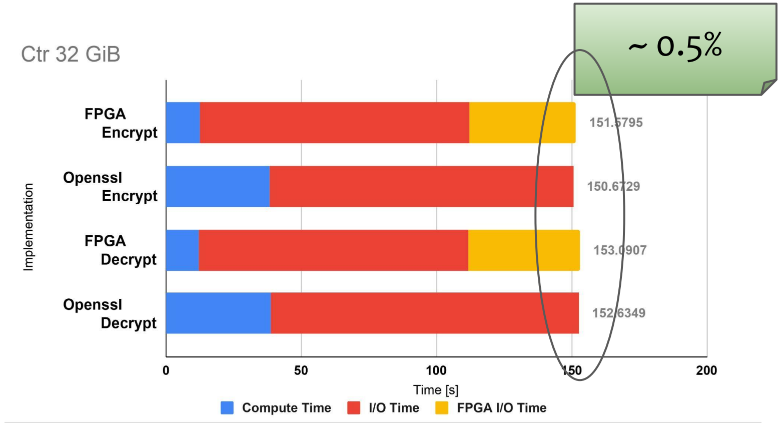


Evaluation: Security Library (Block Independent)









Evaluation: Data Compression Library Overview

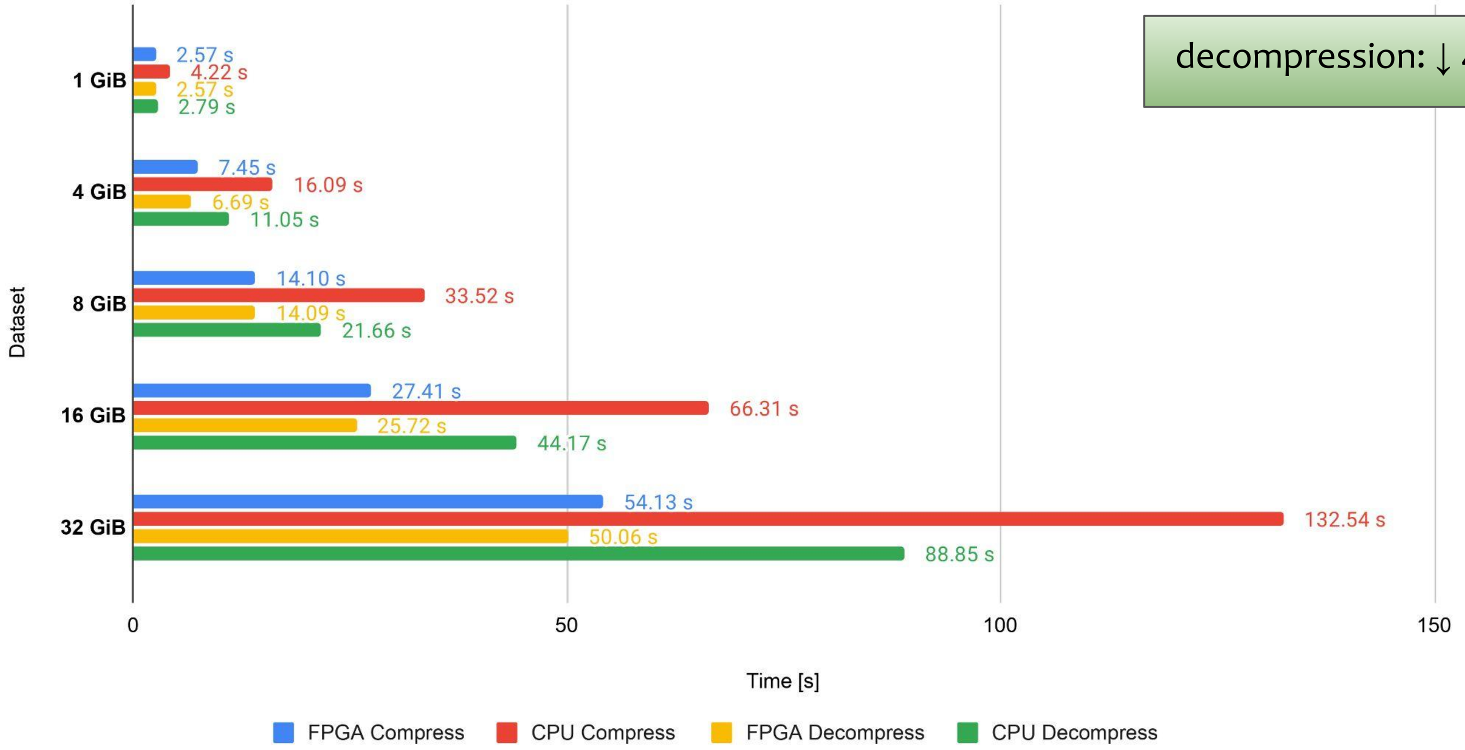
- Gzip & Lz4 & Snappy & Zstd
- 1 GiB to 32 GiB

Evaluation: Data Compression Library

compression: ↓ 59%

decompression: ↓ 41%

Lz4

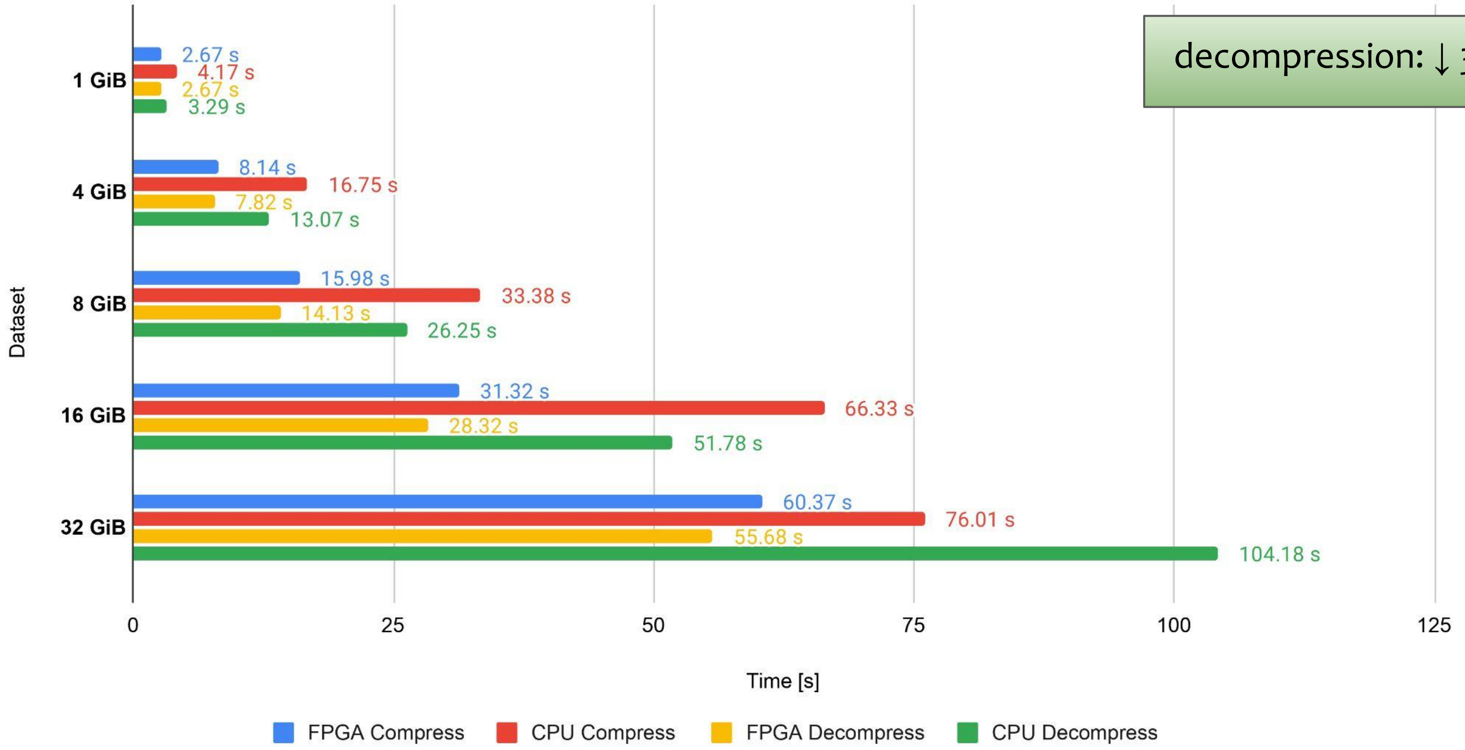


Evaluation: Data Compression Library

compression: ↓ 40%

decompression: ↓ 37%

Snappy

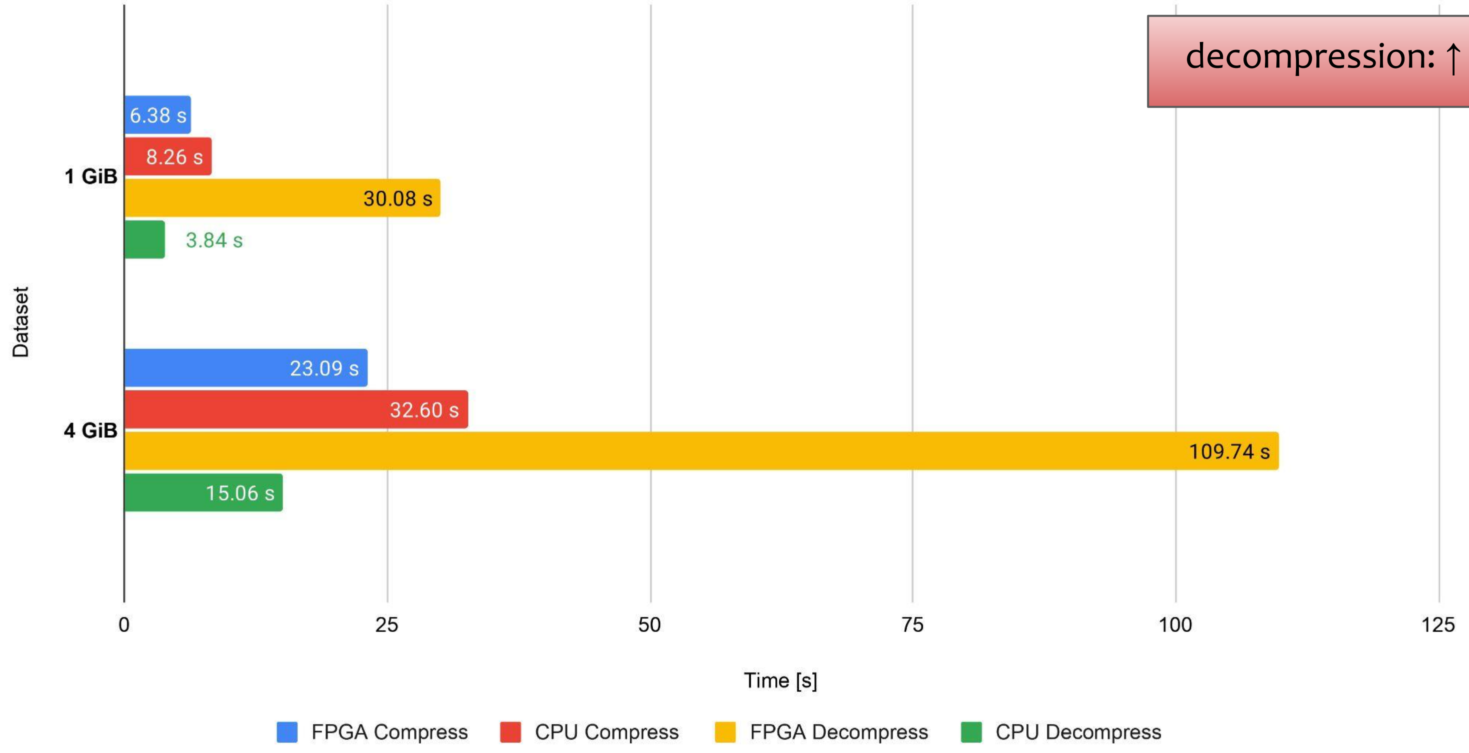


Evaluation: Data Compression Library

compression: ↓ 27%

decompression: ↑ 6x

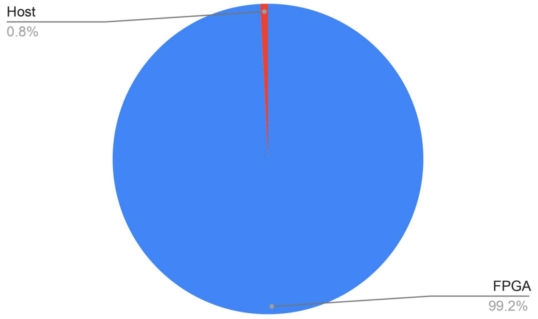
Zstd



Evaluation: Data Compression Library

Zstd: Further Investigation

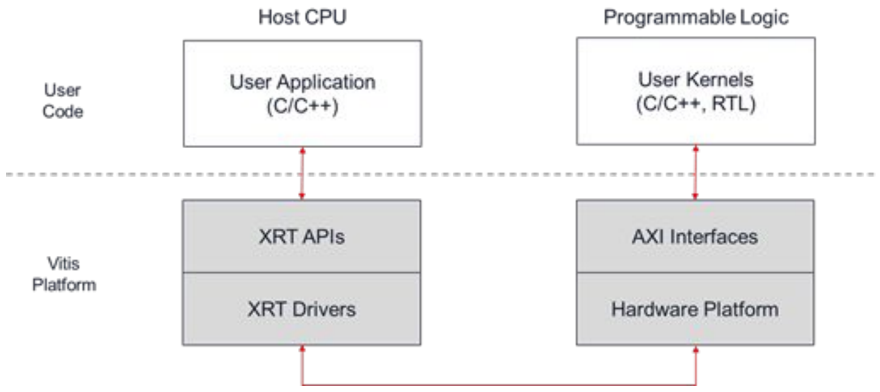
- 99% FPGA time
- Hardware optimization necessary



Implementation

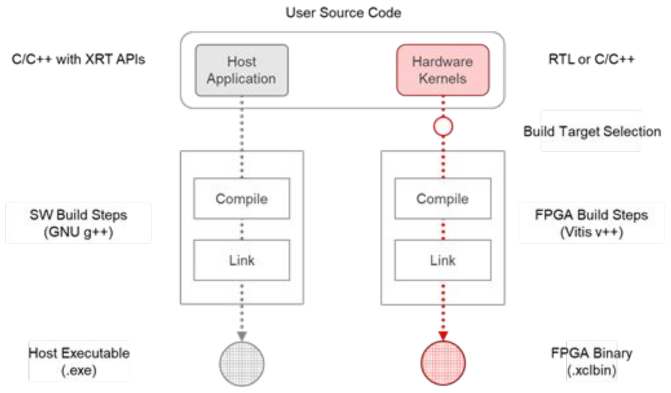
- Software part follows the design guideline
- Security: straight-forward single thread
- Data Compression: multi-threading -> synchronization
- **Key Point:** Interaction with Hardware follows Vitis Execution Model

Vitis Communication



https://github.com/Xilinx/Vitis-Tutorials/blob/2022.2/Getting_Started/Vitis/images/part1_execution_model.png

Vitis Compilation



https://github.com/Xilinx/Vitis-Tutorials/blob/2022.2/Getting_Started/Vitis/images/part1_build_flow.png