University of Stuttgart
Germany
IPVS, Scientific Computing

# Parallel FFTW on RISC-V

## A Comparative Study including OpenMP, MPI, and HPX

June 13, 2025

**Alexander Strack**,
Christopher Taylor,
and Dirk Pflüger

# About Me

- Background in Computational Engineering
- Third-year PhD student in the Scientific Computing department at the University of Stuttgart
- Focused on asynchronous task-based parallelization of numerical algorithms across diverse hardware architectures

# Motivation


- The MILK-V Pioneer Box: first practical RISC-V desktop system
- Portability is necessary — but performance is critical

# Motivation

| Architecture | x86-64 | RISC-V |
|---|---|---|
| CPU | AMD EPYC 7742 | SOPHON SG2042 |
| Cores | 64 | 64 |
| Base clock | 2.25GHz | 2.00GHz |
| Process | 7nm/14nm | 12nm |
| L3 Cache | 256MB | 64MB |
| RAM | 2TB DDR4 | 128GB DDR4 |
| SIMD | AVX2 | RVV 0.7.1 |
| NUMA domains | 4 | 4 |
| TDP | 225W | 120W |
| Release | 2019 | 2023 |


Software Stack

# The Fastest Fourier Transform in the West (FFTW)

- De-facto standard for over 20 year
- Backend for current state-of-the-art libraries
- Performance secret: black magic

# The Fastest Fourier Transform in the West (FFTW)

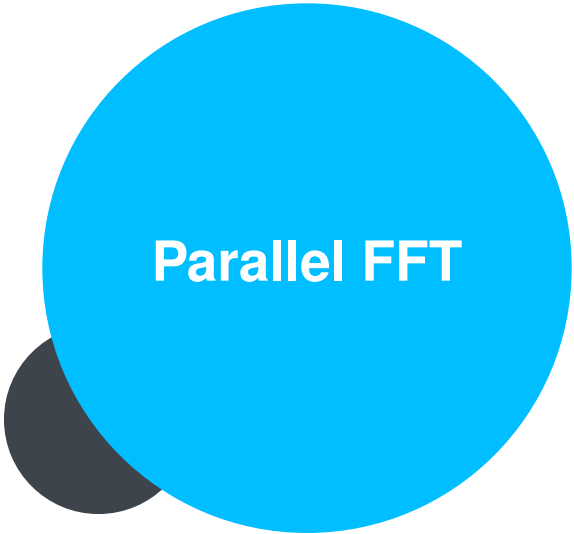
- De-facto standard for over 20 year
- Backend for current state-of-the-art libraries
- Performance secret: ~~black magic~~
  - Small FFT codelets generated at compile time
  - Combination of codelets for large transforms
  - Planning to find best combination for given system

# HPX-FFT

- Leveraging FFTW as backend for 1D FFTs
- Parallelization using the asynchronous many-task runtime HPX
- Origin: tool to evaluate different parallelization approaches with HPX

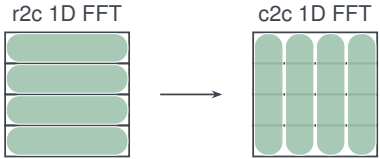
# HPX-FFT

- Leveraging FFTW as backend for 1D FFTs
- Parallelization using the asynchronous many-task runtime HPX
- Origin: tool to evaluate different parallelization approaches with HPX
- Here: performance optimization on x86 and RISC-V, baseline for evaluation of FFTWs parallel plans


Parallel FFT

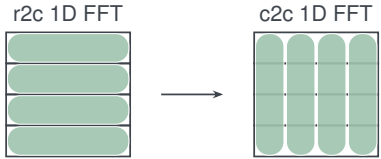
# Two-dimensional FFT

- Strided access

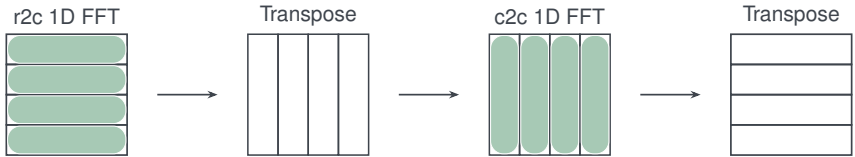


# Two-dimensional FFT

- Strided access




- Data transpose



# Parallel FFTW

Threading backends:

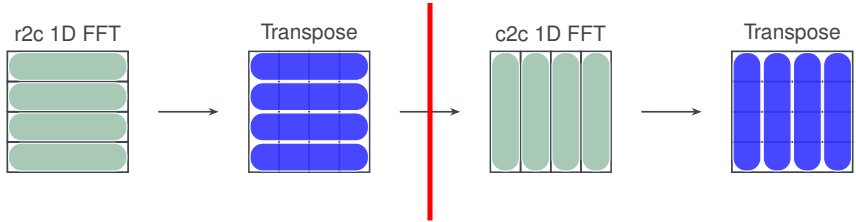
- OpenMP and pthreads
- Choose the block size and number of threads in order to minimize the critical path

MPI backend:

- All-to-all collectives
- Slab decomposition
- Compatible with threading backends to run MPI+X

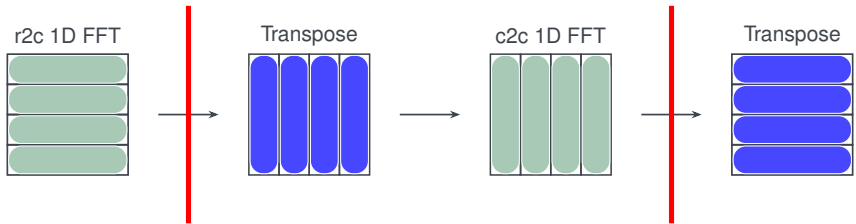
# HPX-FFT variants

- HPX-FFT naive



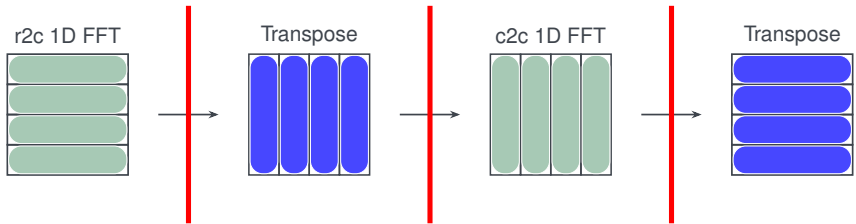
# HPX-FFT variants

- HPX-FFT opt



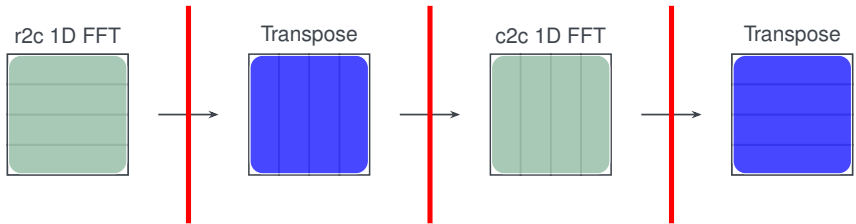
# HPX-FFT variants

- HPX-FFT sync



# HPX-FFT variants

- HPX-FFT for_loop



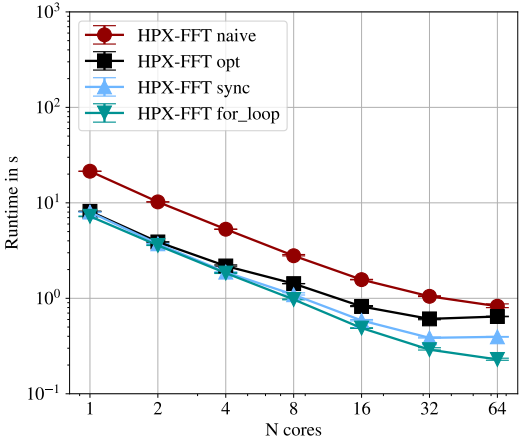

Results

# Benchmark

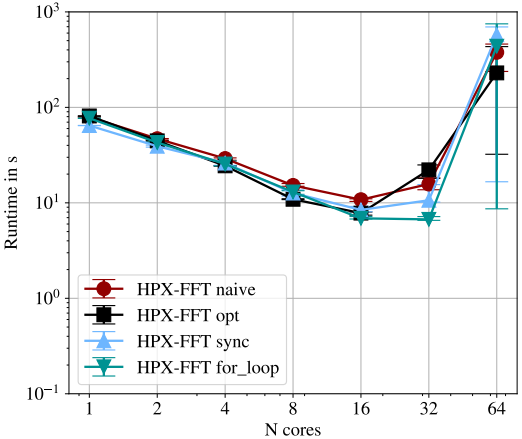
Strong scaling from one to 64 cores:

- Two-dimensional real-to-complex FFT
- No vectorization
- Problem size: $2^{14} \times 2^{14}$
- Data points: median out of 10 runs
- Errorbars: min/max out of 10 runs

# HPX-FFT optimization

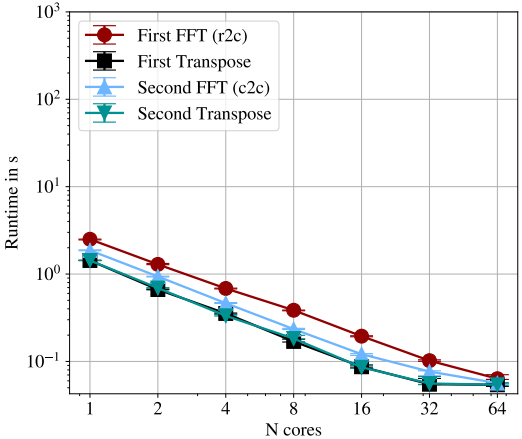



AMD EPYC 7742

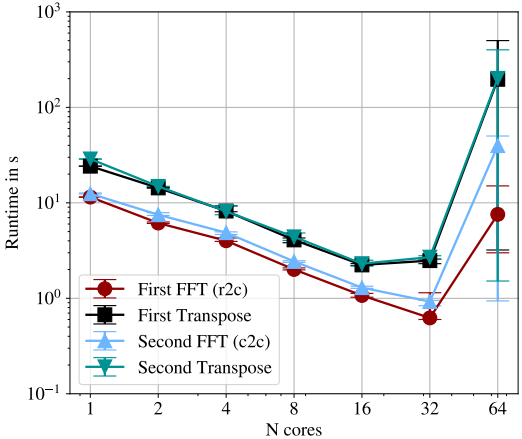



SOPHON SG2042

# HPX-FFT partial (for_loop)

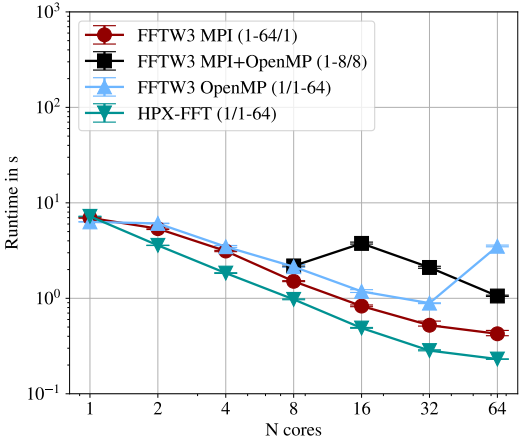



AMD EPYC 7742

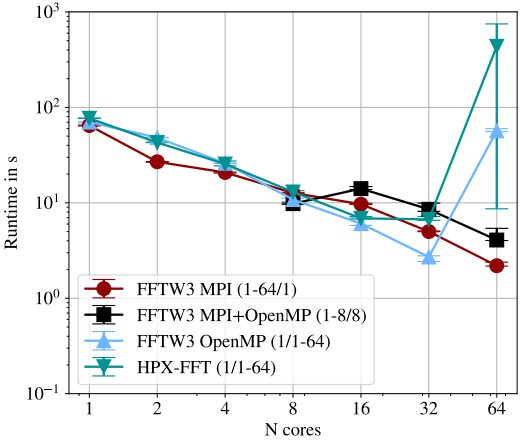



SOPHON SG2042

# FFTW estimated planning

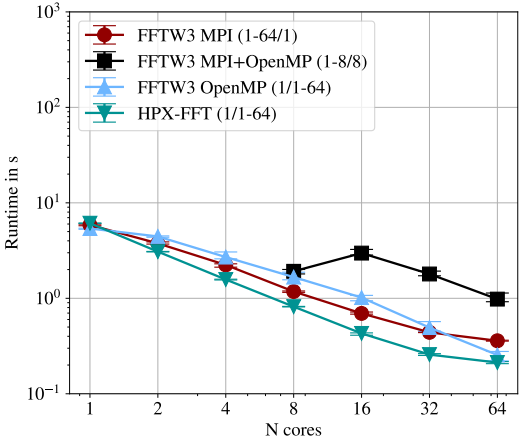



AMD EPYC 7742

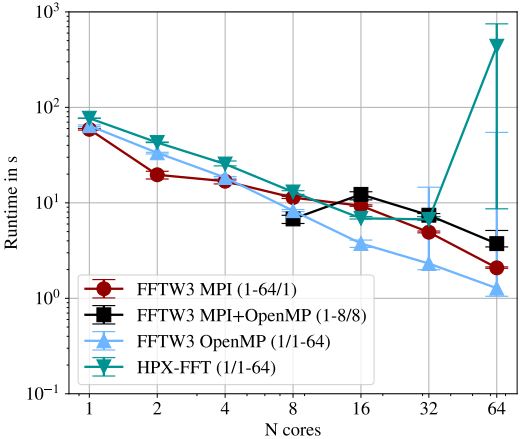



SOPHON SG2042

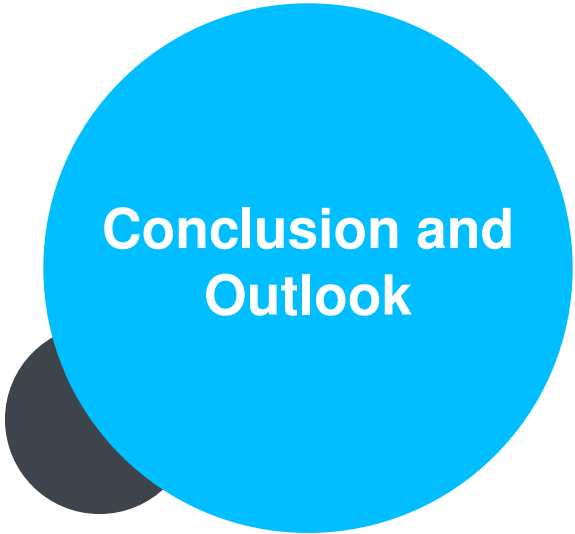
# FFTW measured planning




AMD EPYC 7742




SOPHON SG2042


Conclusion and
Outlook

# Conclusion

- Removing synchronization barriers does not guarantee better performance
- Across different backends performance delta between factor four to eight
- Memory bottleneck on SG2042 chip

# Outlook

- Distributed FFTW and HPX-FFT
- FFTW with RVV1.0
- Comparison against ARM architecture

University of Stuttgart
Germany

# Thank You!

**Alexander Strack, Christopher Taylor, and Dirk Pflüger**
IPVS

Mail alexander.strack@ipvs.uni-stuttgart.de
Phone +49 711 685 88308
Internet https://www.ipvs.uni-stuttgart.de