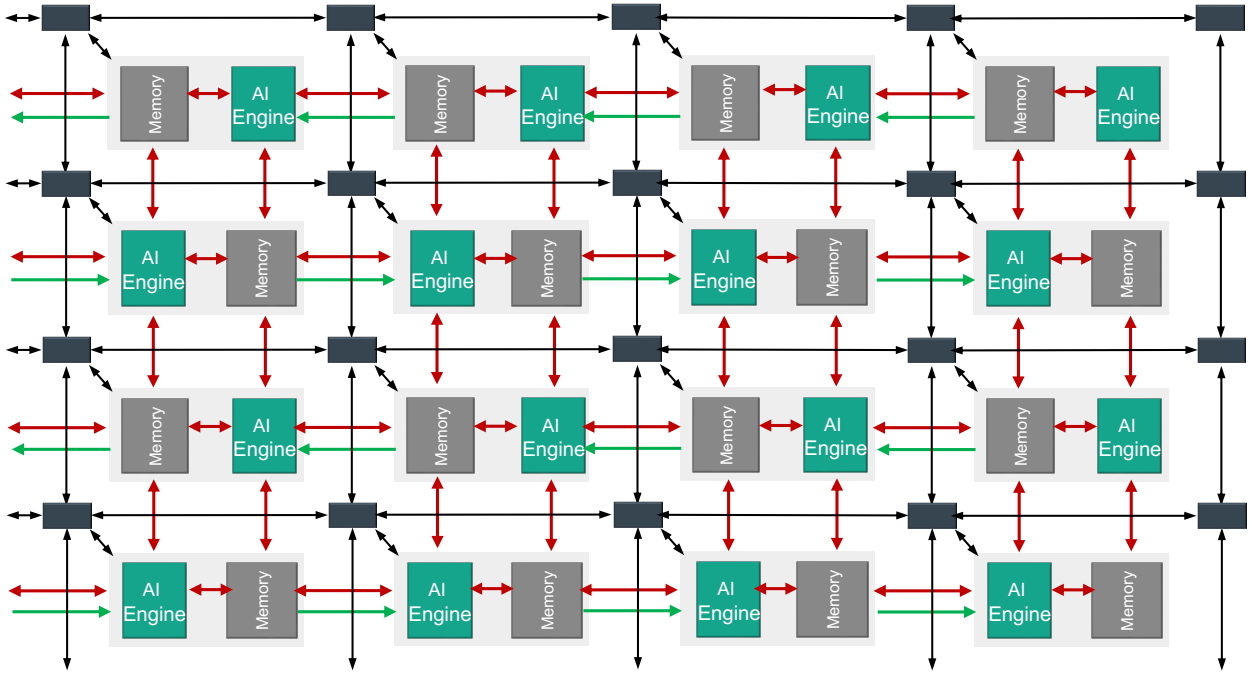
The Design and Implementation of Fast Fourier Transform (FFT) based on AIEs

Participants：Zhuoyan Bai, Yixuan Du, Longshan Xu

Advisor：Associate Professor Fei Xu

CONTENTS



1 / Overview

2 / Algorithm

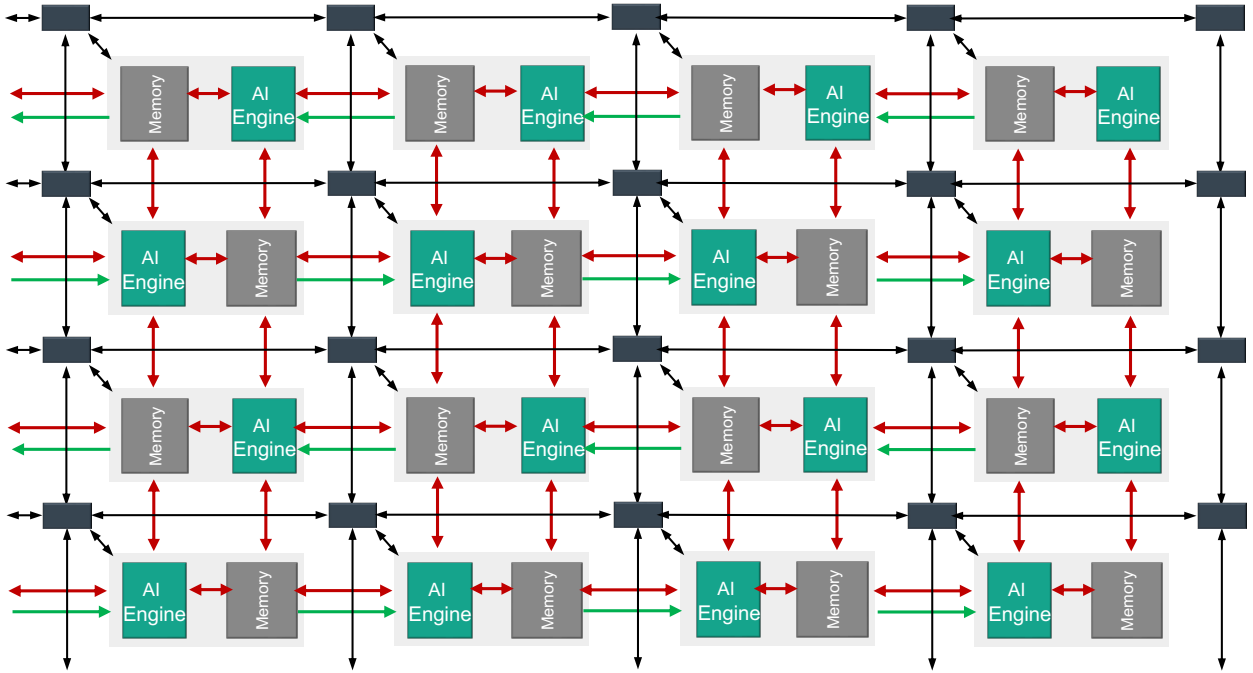
3 / Implementation

4 / Analysis

Overview

Point Size	Number of AIE	Data Type	Twiddle Type
1,024	1	cint16	cint16
		cfloat	cfloat
4,096	5	cint16	cint16
8,192	9	cint16	cint16

- Algorithm
  - ✓ **non-recursive FFT , distributed FFT**
- Kernel Design
  - ✓ **ping-pong buffer , sliding multiplication**
- Graph Design
  - ✓ **kernel connection , kernel placement**



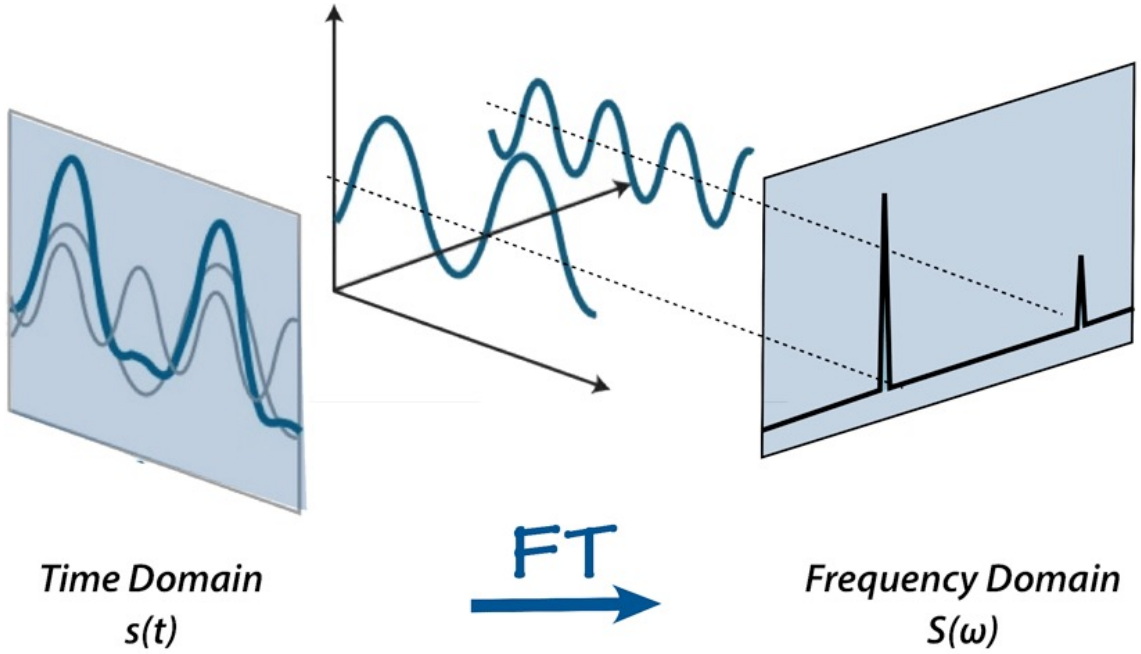
1 / Overview

2 / Algorithm

3 / Implementation

4 / Analysis

Fast Fourier Transform (FFT)



FFT : $O(N^2) \rightarrow O(NlogN)$

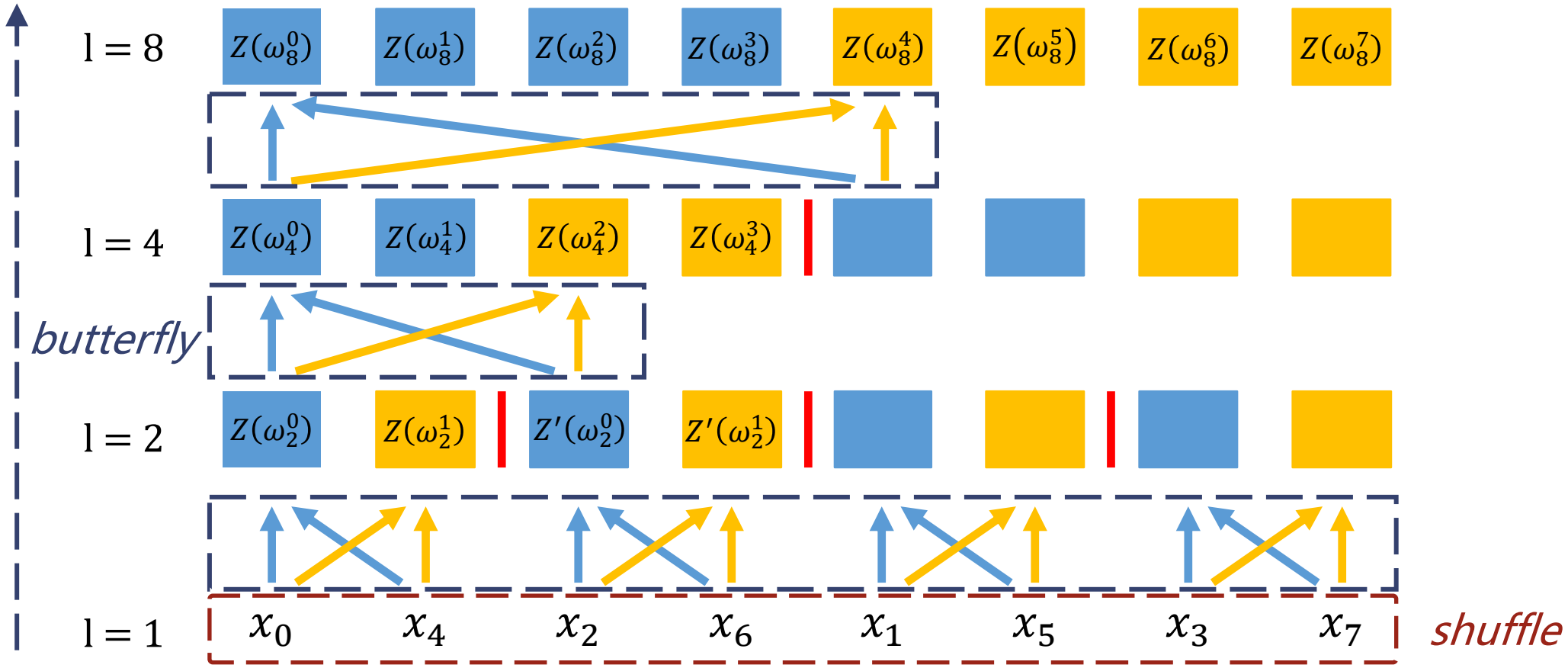
- Non-recursive FFT → 1024-point
- Distributed FFT → Expand point size

Non-recursive FFT

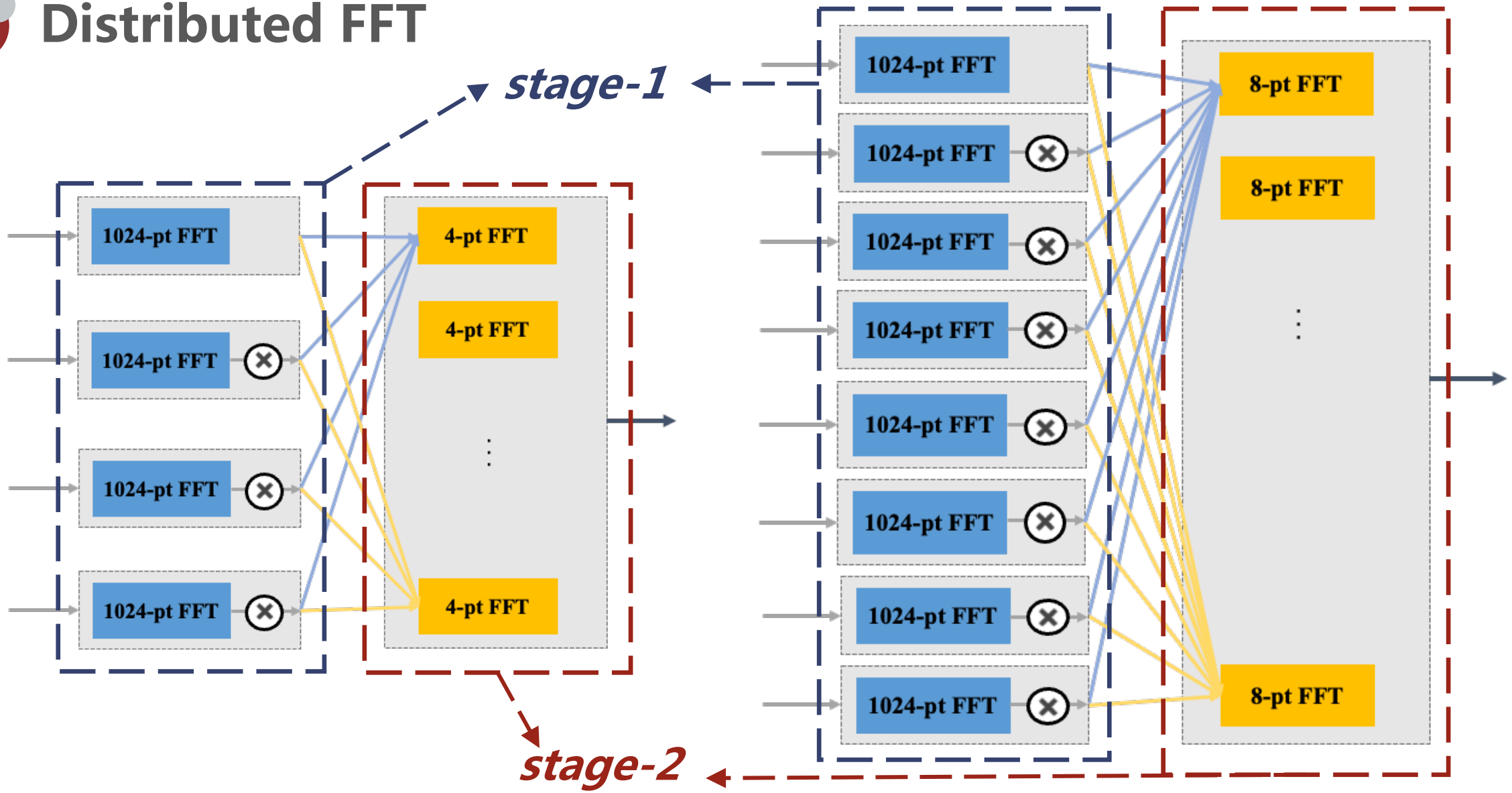
Key operations : shuffle and butterfly

$X_k = E_k + e^{-\frac{2\pi i}{N}k} O_k$

$X_{k+N/2} = E_k - e^{-\frac{2\pi i}{N}k} O_k$

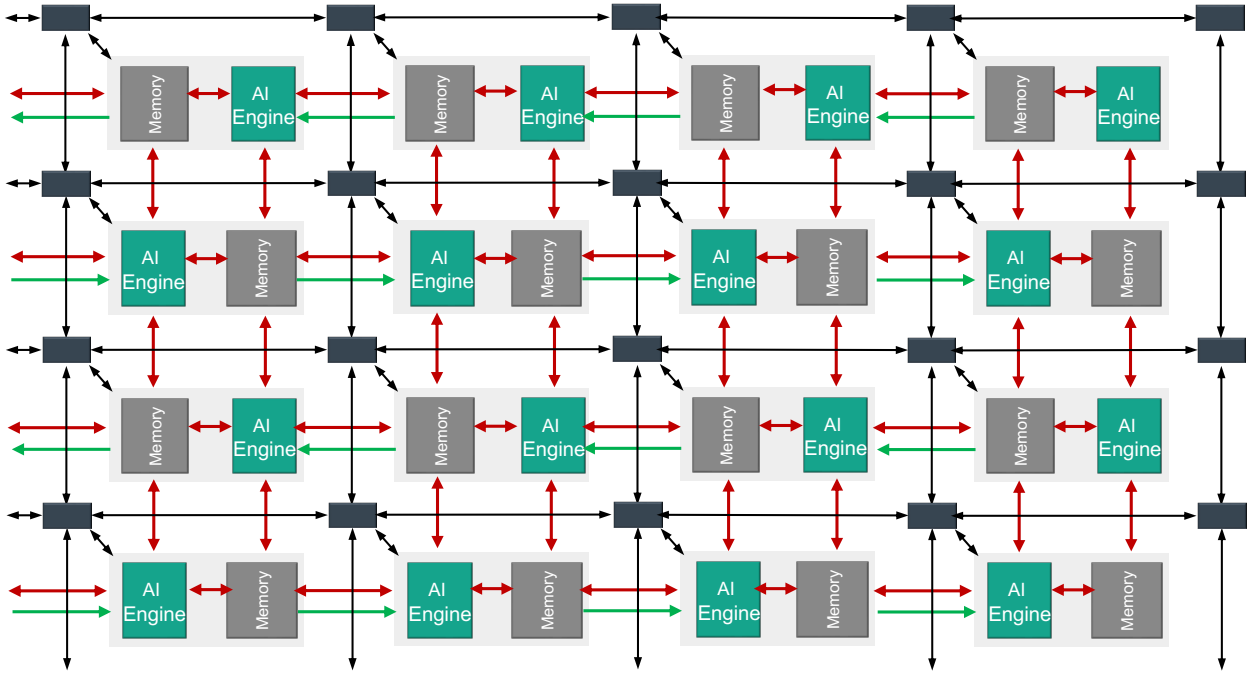


Distributed FFT



4K-point distributed FFT

8K-point distributed FFT



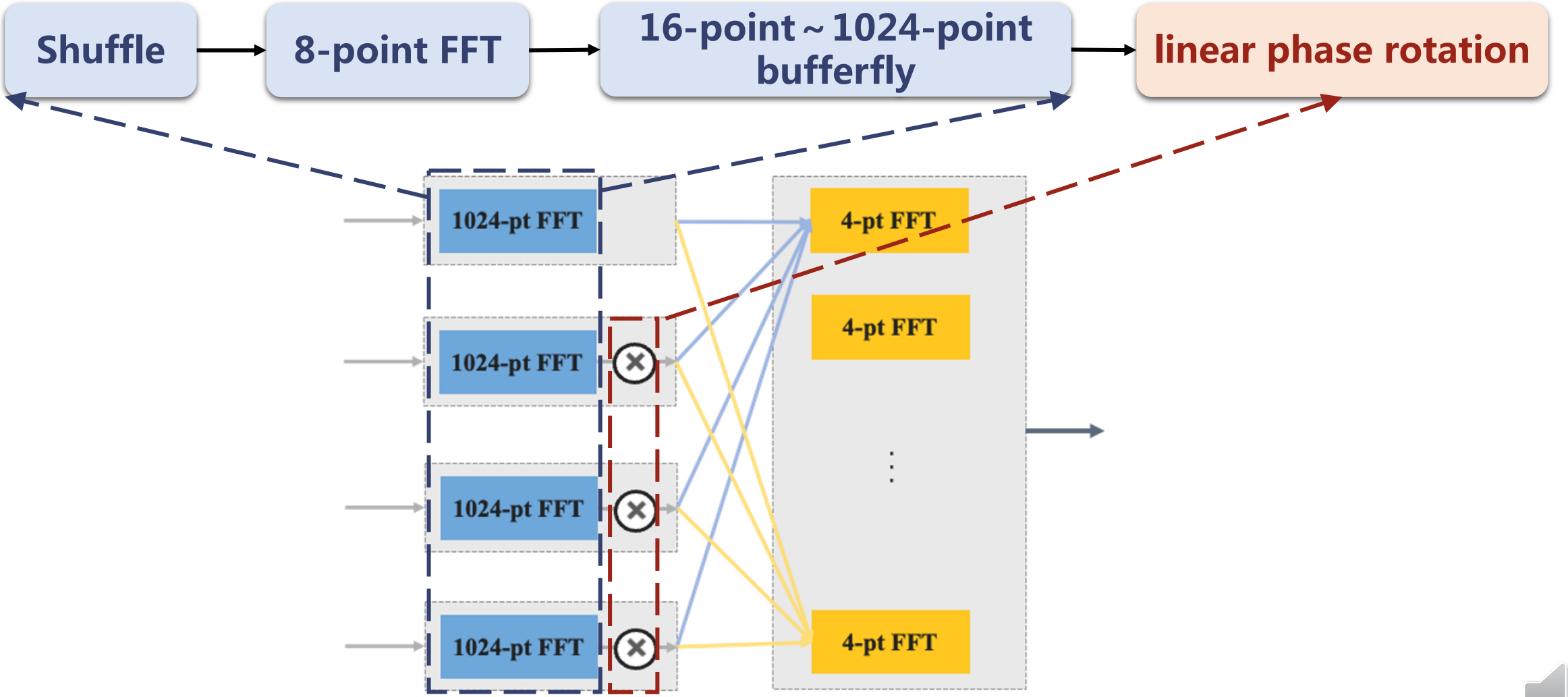
1 / Overview

2 / Algorithm

3 / Implementation

4 / Analysis

1K-point FFT (stage-1)



1K-point FFT (stage-1)

- Implement non-recursive 1K FFT based on 8-point FFT
- Shuffle to permute the order of the input signal
- Divide 1024 points into 120 sequences of length 2 and 192 sequences of length 4, and swap the data within each sequence to achieve the shuffle operation
- Use loop unroll for parallel computing

1K-point FFT (stage-1)

64 MACs —SIMD→ 8 instructions

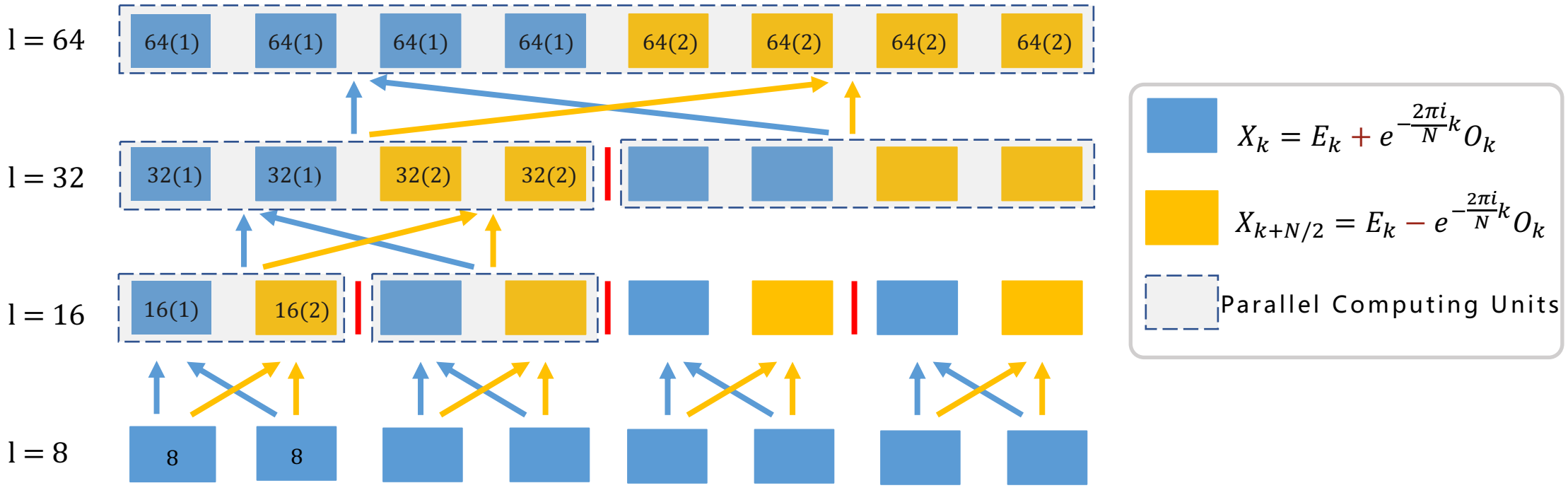
Shuffle → 8-point FFT → 16-point ~ 1024-point bufferfly → linear phase rotation

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} & \omega^{28} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} & \omega^{35} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} & \omega^{42} \\ 1 & \omega^7 & \omega^{14} & \omega^{21} & \omega^{28} & \omega^{35} & \omega^{42} & \omega^{49} \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \times x_0 + \begin{bmatrix} 1 \\ \omega^1 \\ \omega^2 \\ \omega^3 \\ \omega^4 \\ \omega^5 \\ \omega^6 \\ \omega^7 \end{bmatrix} \times x_1 + \cdots + \begin{bmatrix} 1 \\ \omega^7 \\ \omega^{14} \\ \omega^{21} \\ \omega^{28} \\ \omega^{35} \\ \omega^{42} \\ \omega^{49} \end{bmatrix} \times x_7 = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix}$$

1K-point FFT (stage-1)

Load as much data as possible to maximize parallelism

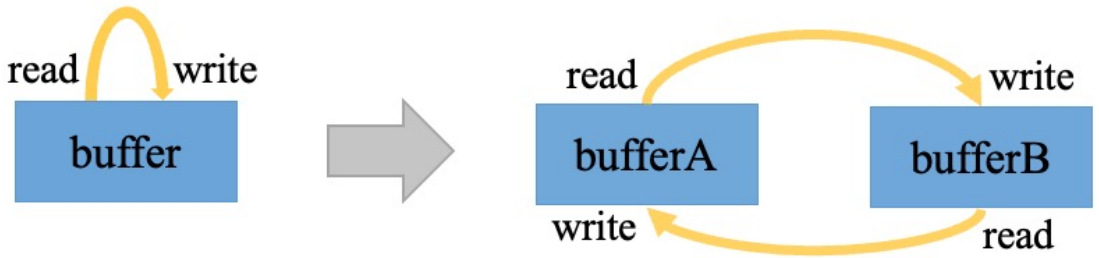
Shuffle → 8-point FFT → 16-point ~ 1024-point bufferfly → linear phase rotation



1K-point FFT (stage-1)

Use Ping-pong buffer to improve mem access efficiency

Shuffle → **8-point FFT** → **16-point ~ 1024-point bufferfly** → **linear phase rotation**



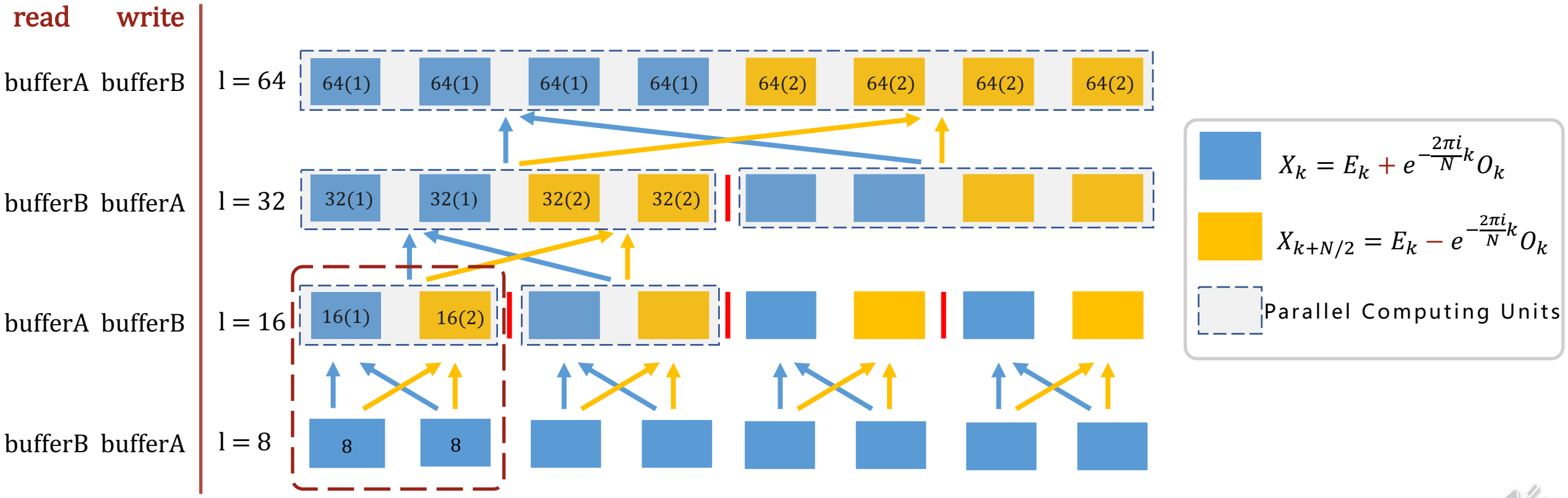
ping-pong buffer

Operation	Read Buffer	Write Buffer
Shuffle	bufferA	bufferB
8-point FFT	bufferB	bufferA
l=16 bufferfly	bufferA	bufferB
l=32 bufferfly	bufferB	bufferA
l=64 bufferfly	bufferA	bufferB
l=128 bufferfly	bufferB	bufferA
l=256 bufferfly	bufferA	bufferB
l=512 bufferfly	bufferB	bufferA
l=1024 bufferfly + linear phase rotation	bufferA	bufferB

1K-point FFT (stage-1)

Use Ping-pong buffer to improve mem access efficiency

Shuffle → 8-point FFT → 16-point ~ 1024-point bufferfly → linear phase rotation



4/8K-point FFT (stage-2)

aie::sliding_mul_ops

x_in0	d_0	d_1	...	d_7	...
x_in1	d_8	d_9	...	d_{15}	...
x_in2	d_{16}	d_{17}	...	d_{23}	...
x_in3	d_{24}	d_{25}	...	d_{31}	...

d_0	d_1	...	d_7
d_8	d_9	...	d_{15}
d_{16}	d_{17}	...	d_{23}
d_{24}	d_{25}	...	d_{31}

4/8K-point FFT (stage-2)

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} \times \begin{bmatrix} d_0 & d_1 & \cdots & d_7 \\ d_8 & d_9 & \cdots & d_{15} \\ d_{16} & d_{17} & \cdots & d_{23} \\ d_{24} & d_{25} & \cdots & d_{31} \end{bmatrix} = \begin{bmatrix} y_0 & y_1 & \cdots & y_7 \\ y_8 & y_9 & \cdots & y_{15} \\ y_{16} & y_{17} & \cdots & y_{23} \\ y_{24} & y_{25} & \cdots & y_{31} \end{bmatrix}$$

aie::sliding_mul_ops

d_0	d_1	$\cdots$	d_7	d_8	d_9	$\cdots$	d_{15}	d_{16}	d_{17}	$\cdots$	d_{23}	d_{24}	d_{25}	$\cdots$	d_{31}
1				ω				ω^2				ω^3			
	1				ω				ω^2				ω^3		
		$\ddots$				$\ddots$				$\ddots$				$\ddots$	
			1				ω				ω^2				ω^3

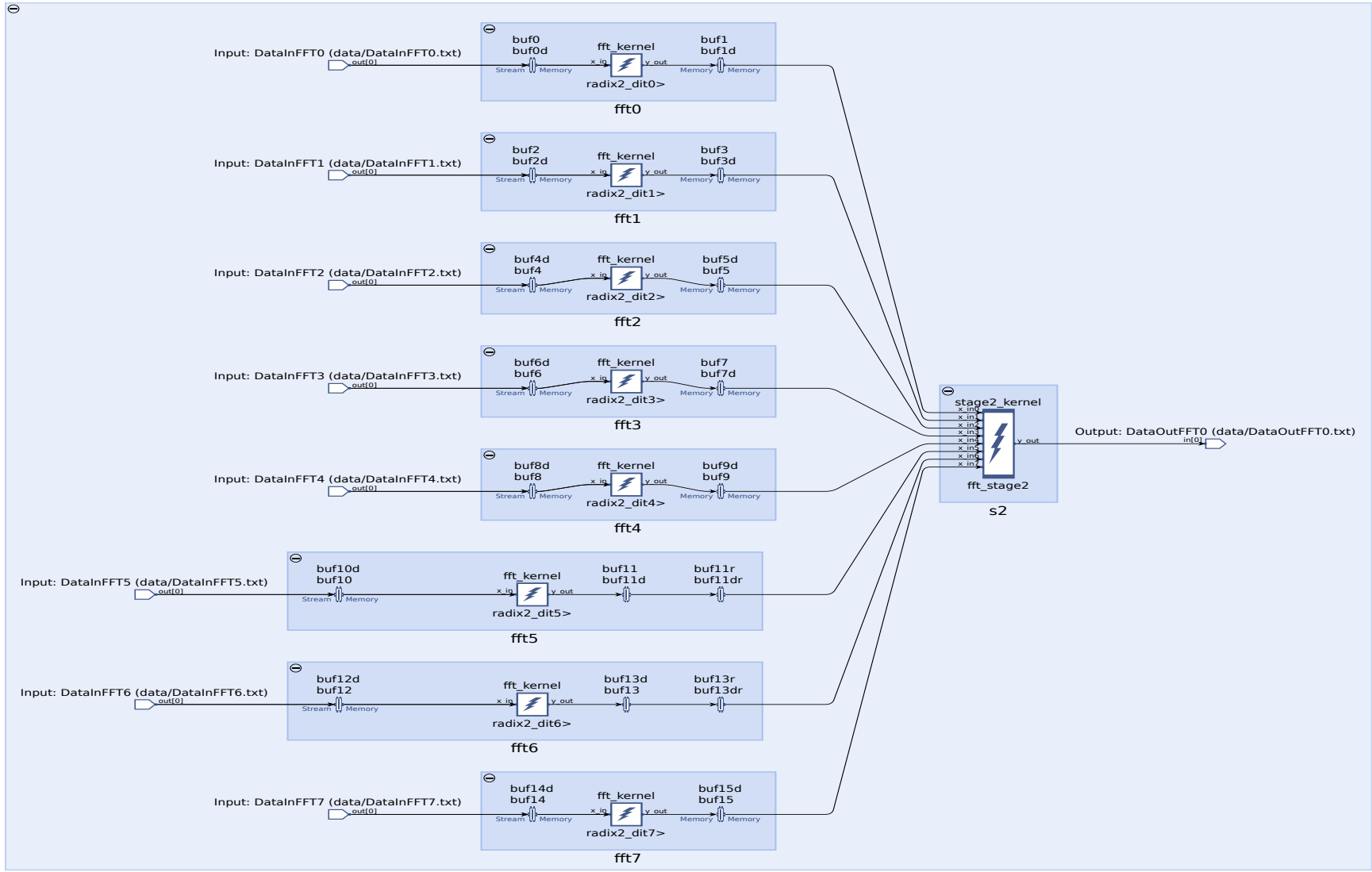
$$y_8 = 1 \cdot d_0 + \omega \cdot d_8 + \omega^2 \cdot d_{16} + \omega^3 \cdot d_{24}$$

$$y_9 = 1 \cdot d_1 + \omega \cdot d_9 + \omega^2 \cdot d_{17} + \omega^3 \cdot d_{25}$$

$$\vdots$$

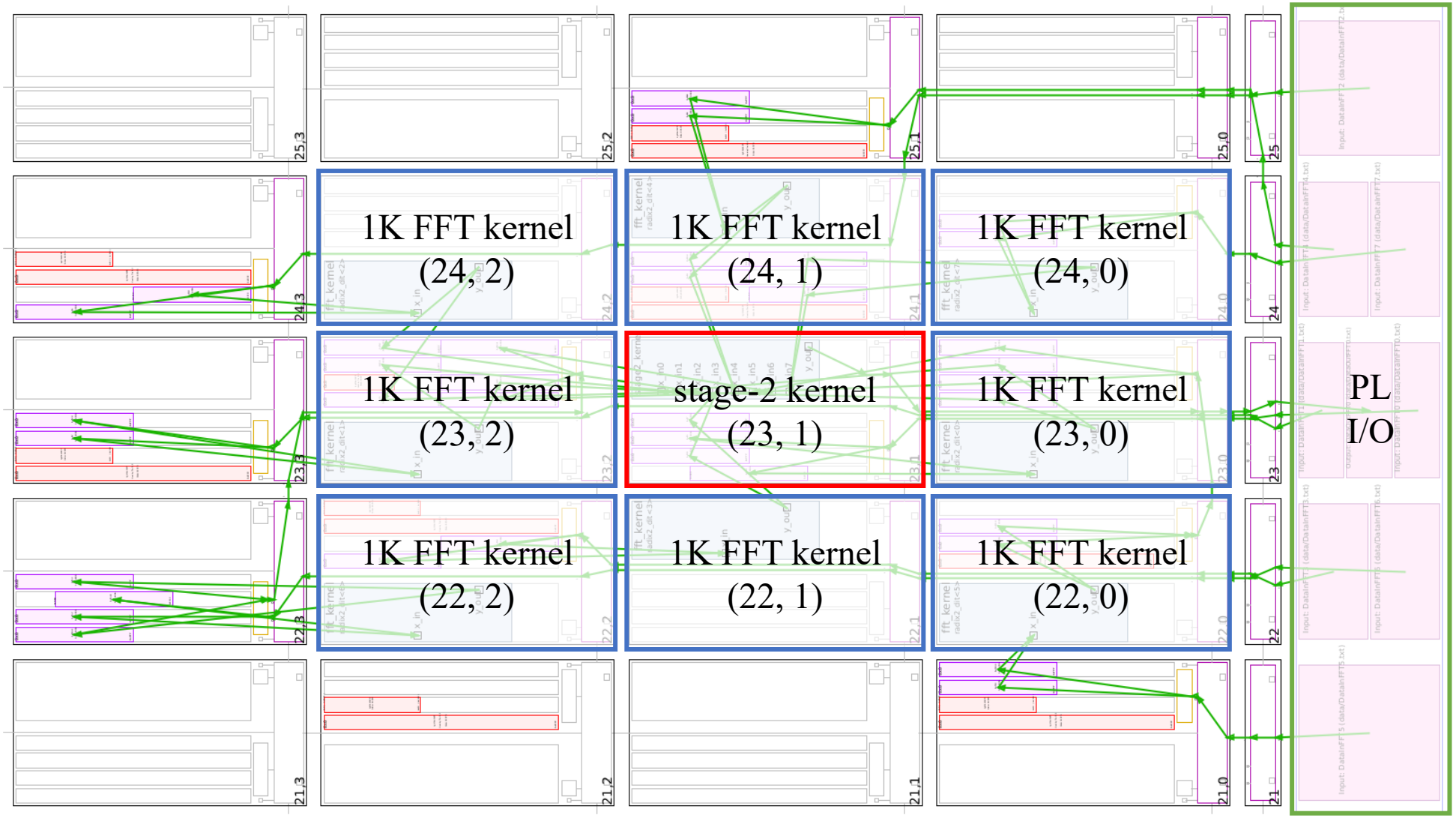
$$y_{15} = 1 \cdot d_7 + \omega \cdot d_{15} + \omega^2 \cdot d_{23} + \omega^3 \cdot d_{31}$$

8K-point FFT Graph

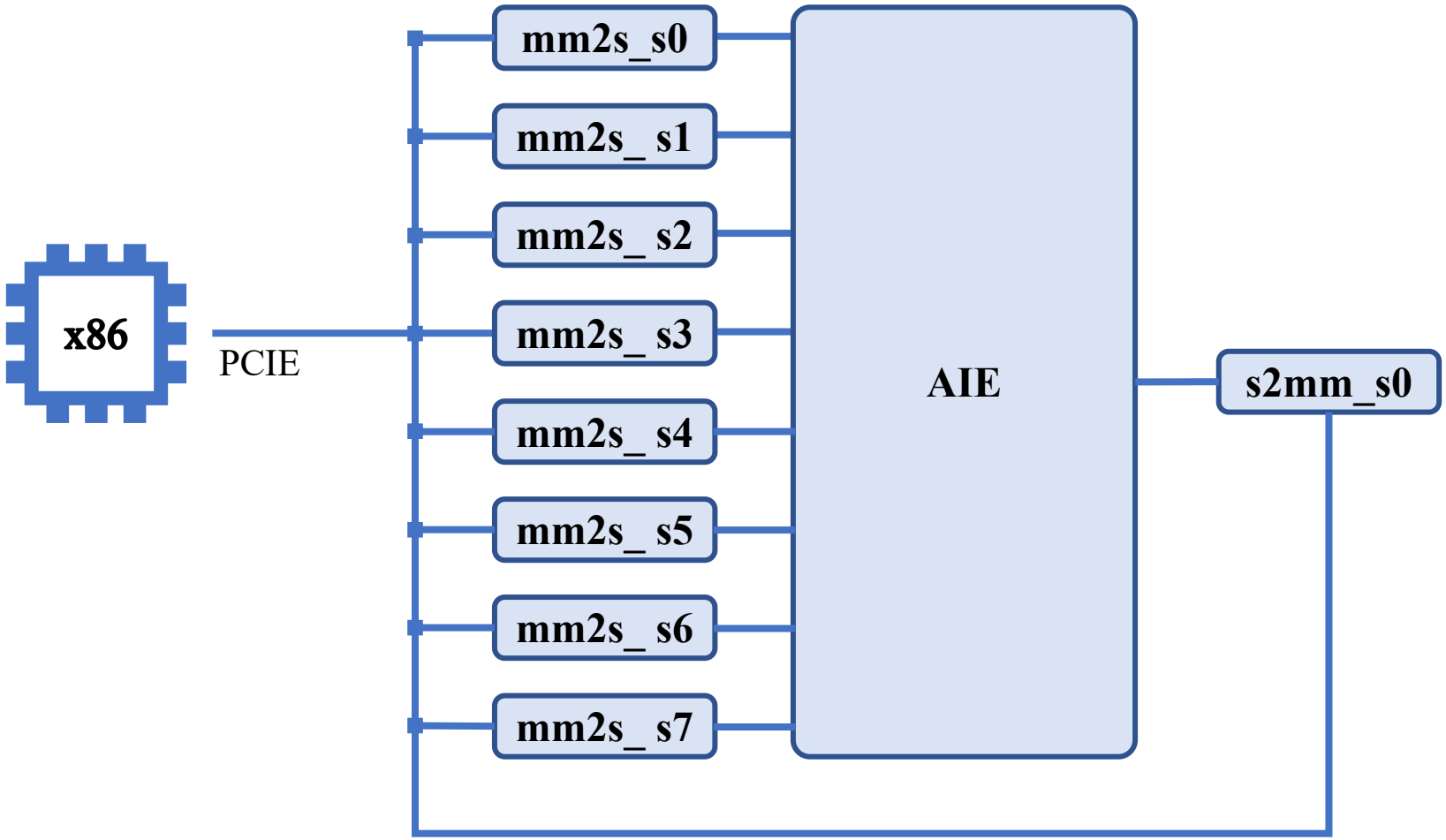


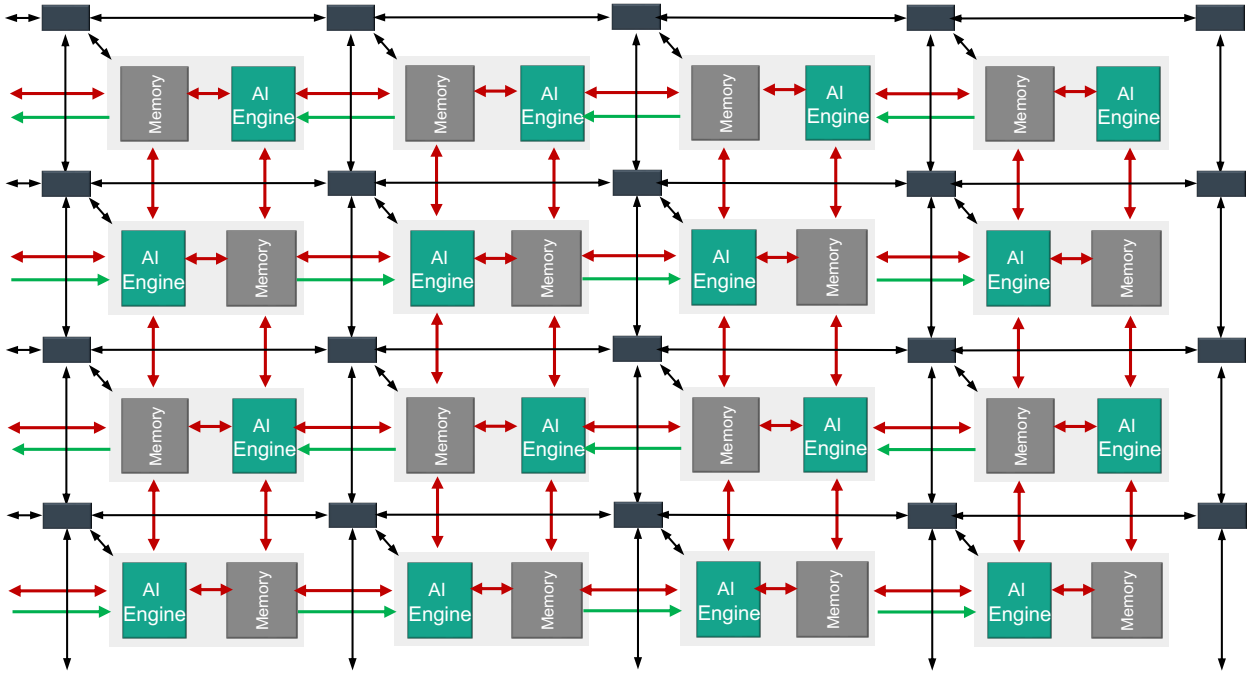
g

8K-point FFT Array Layout (Kernel Placement Optimization)



PL Implementation





1 / Overview

2 / Algorithm

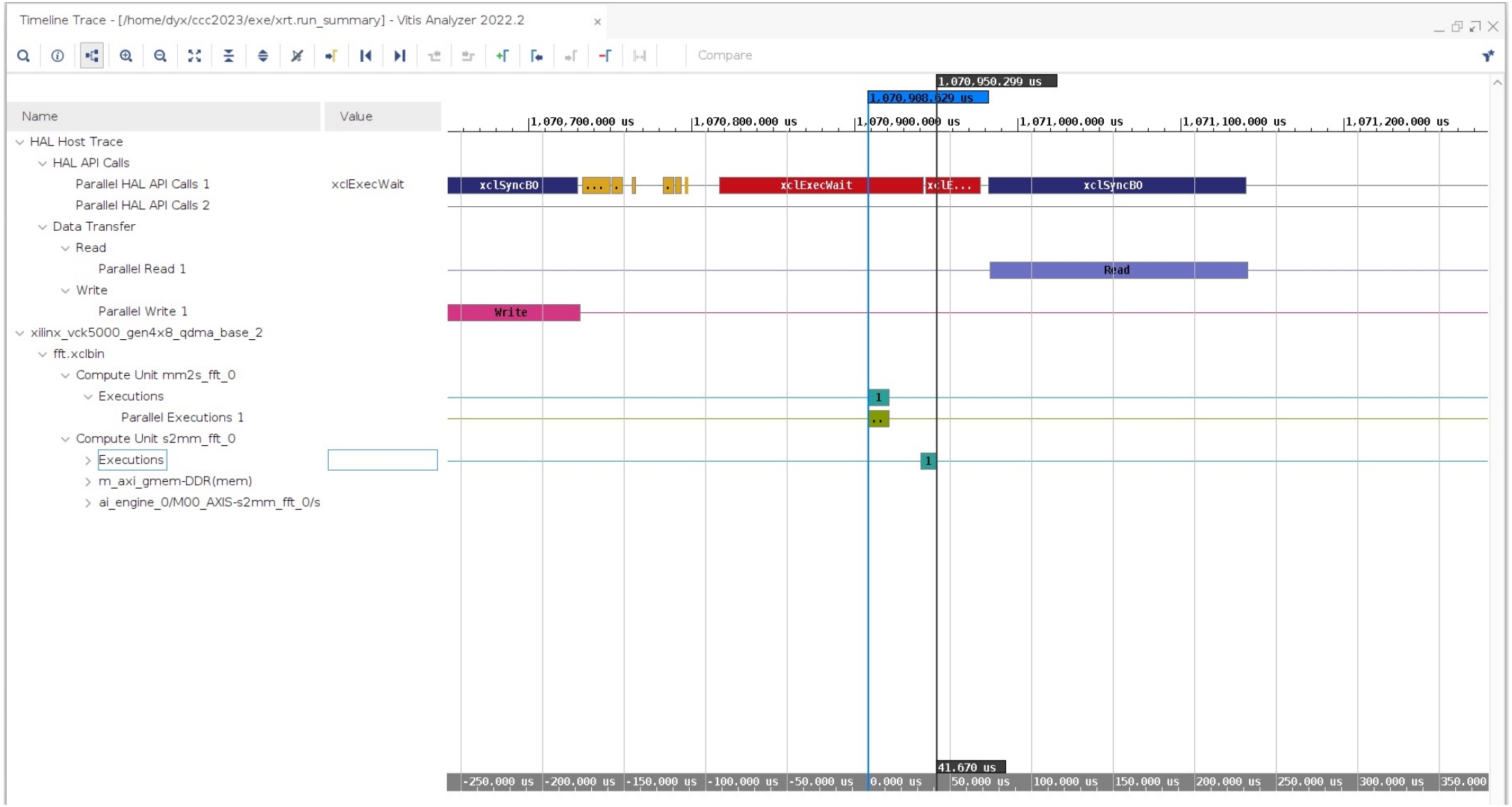
3 / Implementation

4 / Analysis

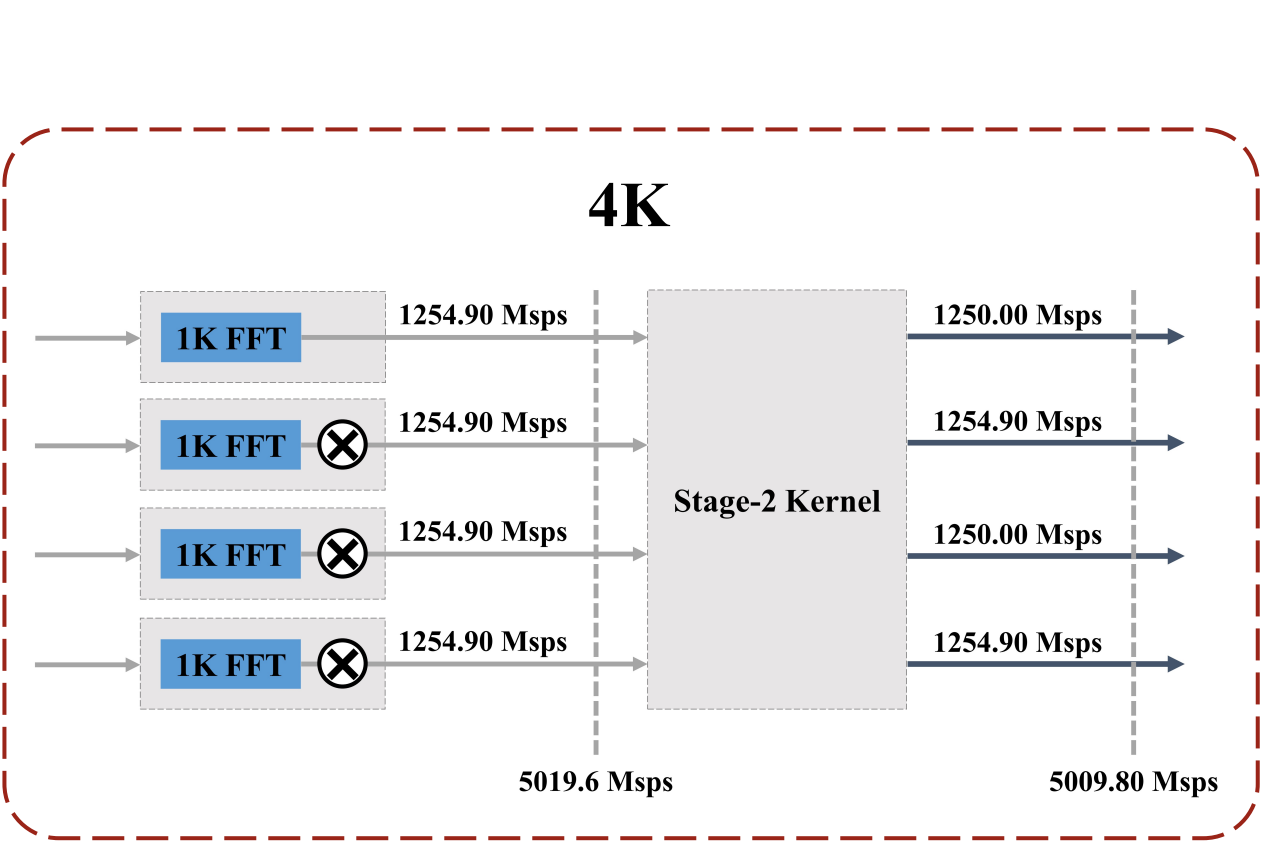
Performance Analysis

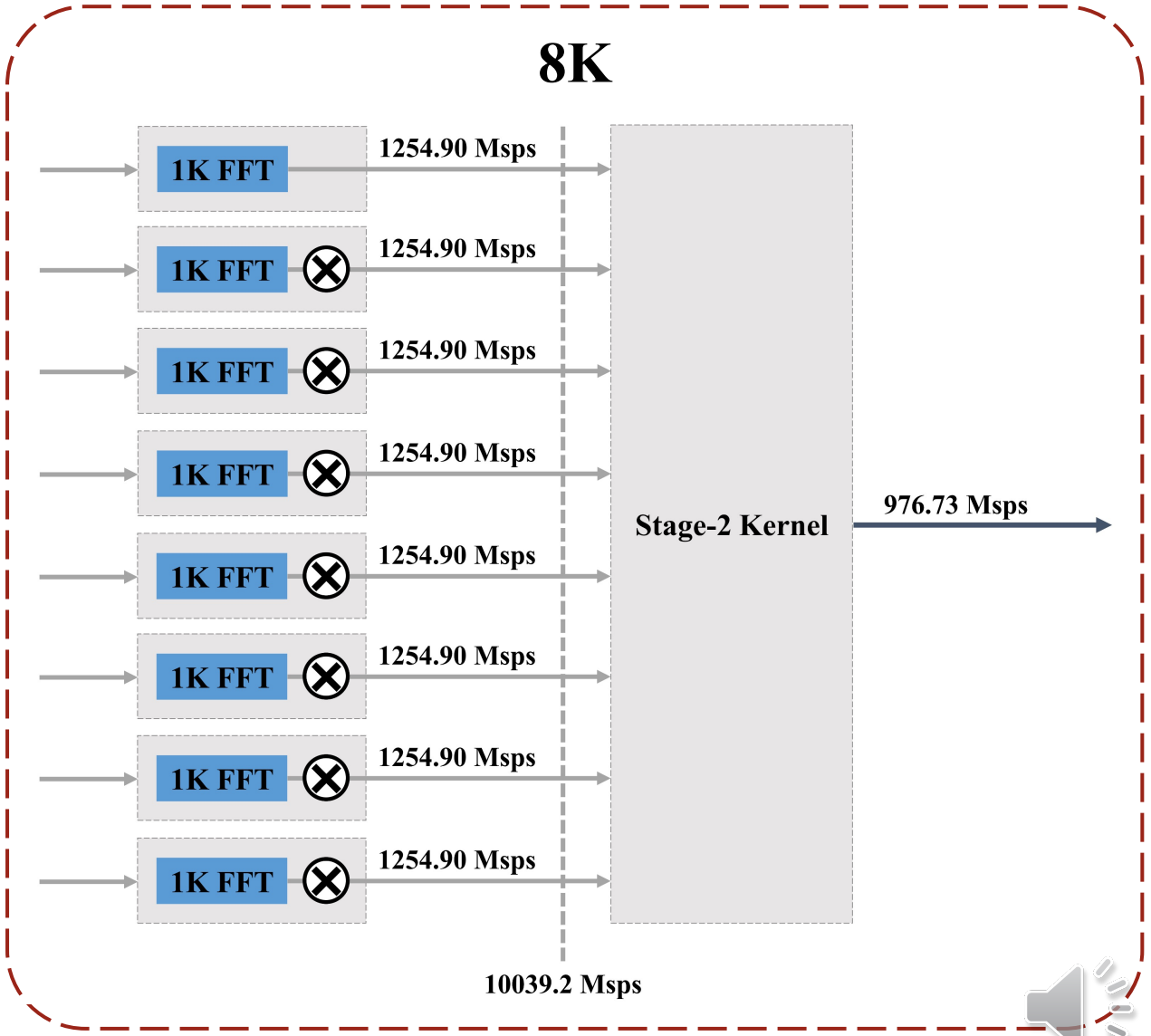
Point Size	Number of AIE	Data Type	Twiddle Type	AIE Runtime (us)	Throughput (Msps)
1,024	1	cint16	cint16	6.588	1254.90
		cfloat	cfloat	29.840	626.22
4,096	5	cint16	cint16	9.764	5009.80
8,192	9	cint16	cint16	16.174	976.73

Trace


Timeline Trace - [/home/dyx/ccc2023/exe/xrt.run_summary] - Vitis Analyzer 2022.2
Compare
Name
Value
HAL Host Trace
HAL API Calls
Parallel HAL API Calls 1
xclExecWait
Parallel HAL API Calls 2
Data Transfer
Read
Parallel Read 1
Write
Parallel Write 1
xilinx_vck5000_gen4x8_qdma_base_2
fft.xclbin
Compute Unit mm2s_fft_0
Executions
Parallel Executions 1
Compute Unit s2mm_fft_0
Executions
m_axi_gmem-DDR(mem)
ai_engine_0/M00_AXIS-s2mm_fft_0/s
1,070,950.299 us
1,070,908.629 us
1,070,700.000 us
1,070,800.000 us
1,070,900.000 us
1,071,000.000 us
1,071,100.000 us
1,071,200.000 us
xclSyncBO
xclExecWait
xclE...
xclSyncBO
Read
Write
41.670 us
-250.000 us
-200.000 us
-150.000 us
-100.000 us
-50.000 us
0.000 us
50.000 us
100.000 us
150.000 us
200.000 us
250.000 us
300.000 us
350.000

Throughput


4K
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
Stage-2 Kernel
1250.00 Msps
1254.90 Msps
1250.00 Msps
1254.90 Msps
5019.6 Msps
5009.80 Msps


8K
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
1K FFT
1254.90 Msps
Stage-2 Kernel
976.73 Msps
10039.2 Msps

Summary

- We implemented **non-recursive 1K FFT** and **4K/8K FFT** based on a distributed FFT algorithm.
- We implemented PL and host so that this project can run on VCK5000.

Key Optimization

- **Loop opt:** loop unroll and pipelining
- **Parallel computing:** sliding multiplication
- **Ping-Pong Buffer:** improve memory access efficiency

Future Improvement

- **Precision:** adapt to larger point size
- **Resource allocation:** explore better PL (currently used only for data transmission) and AIE resource allocation

華東師範大學
EAST CHINA NORMAL UNIVERSITY

THANKS FOR ALL

2023.08.23

CCFSys Customized Computing Challenge 2023