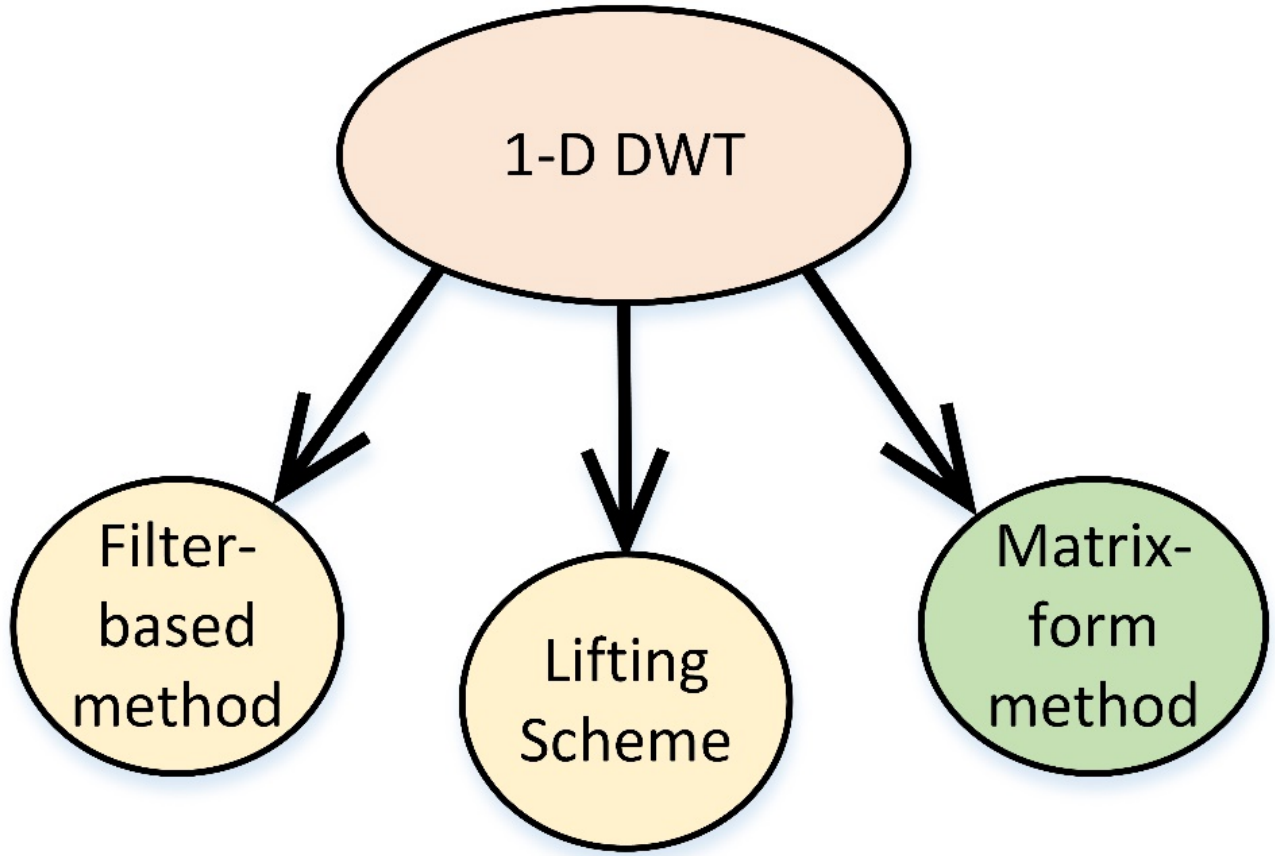
Vector FPGA Acceleration of 1-D DWT Computations using Sparse Matrix Skeletons

Sidharth Maheshwari, Gourav Modi, **Siddhartha**, Nachiket Kapre

School of Computer Science and Engineering

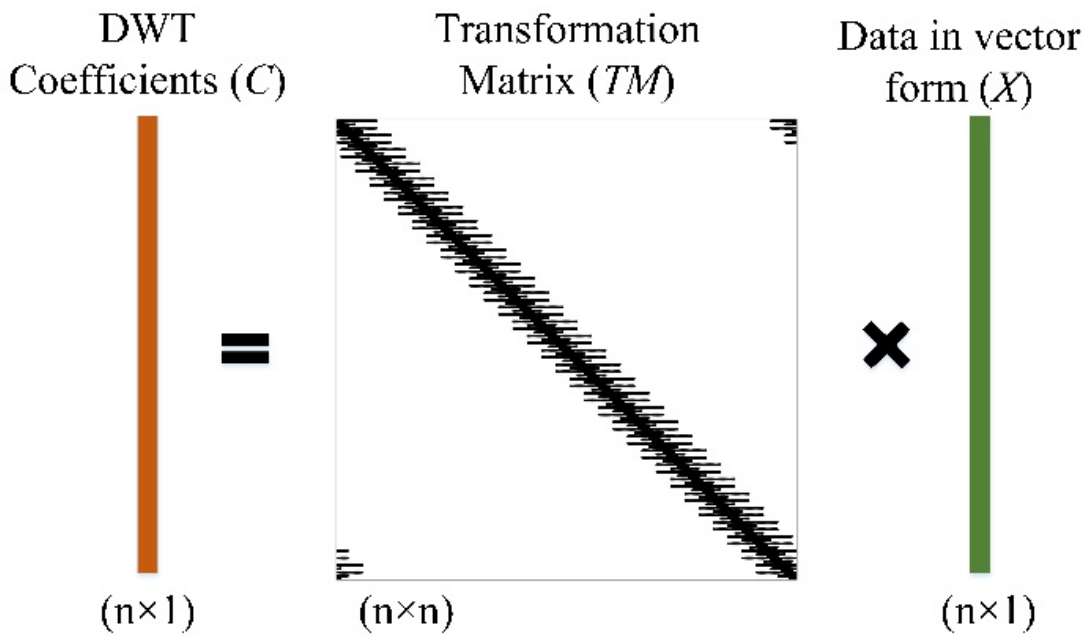
Nanyang Technological University


1-D DWT
Filter-based method
Lifting Scheme
Matrix-form method

Matrix-Form 1-D DWT

- Formulation:

$C = TM \cdot X$, where $TM = \prod_k T_{ak}^{p}$

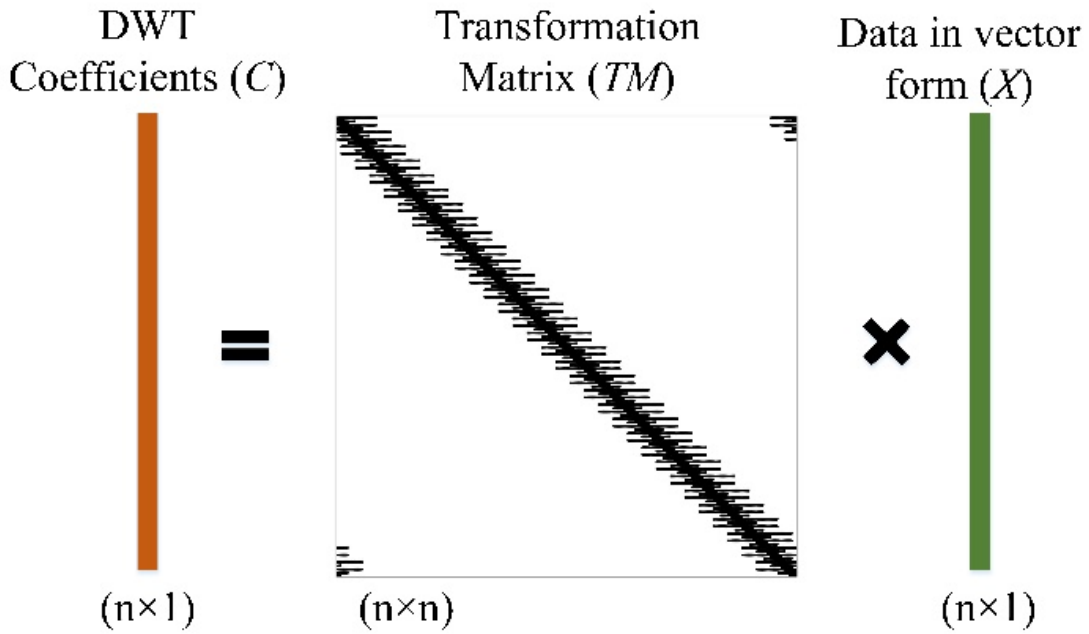


Matrix-Form 1-D DWT

- Formulation:

$$C = TM \cdot X, \text{where } TM = \prod_k T_{ak}^{p}$$

- TM matrix is highly sparse
  - Large number of multiply-by-zero operations
  - Large memory footprint consisting of zeroes

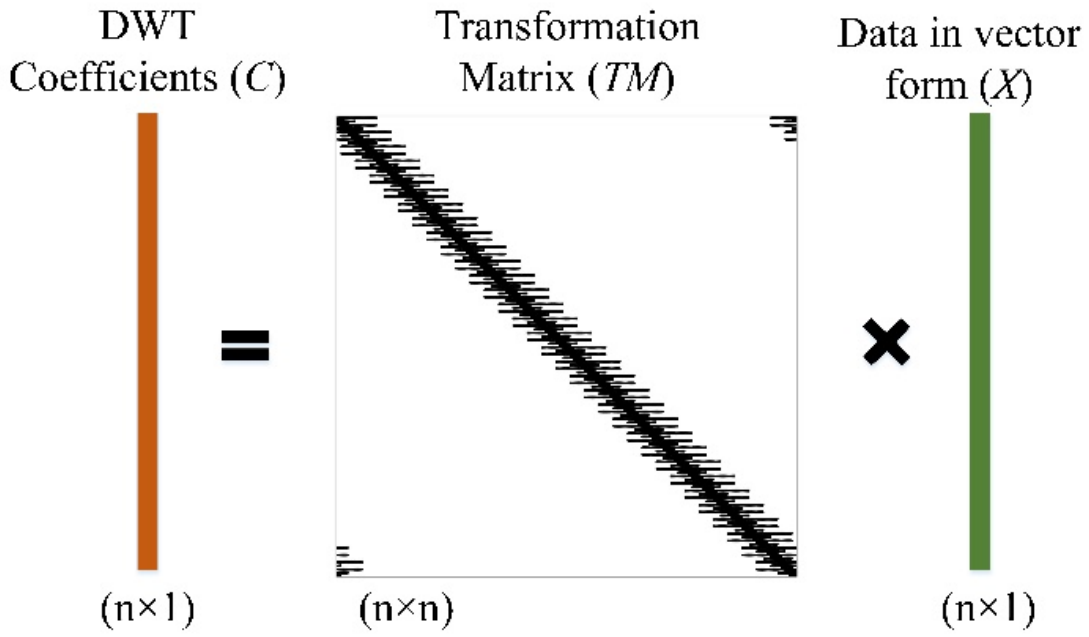


Matrix-Form 1-D DWT

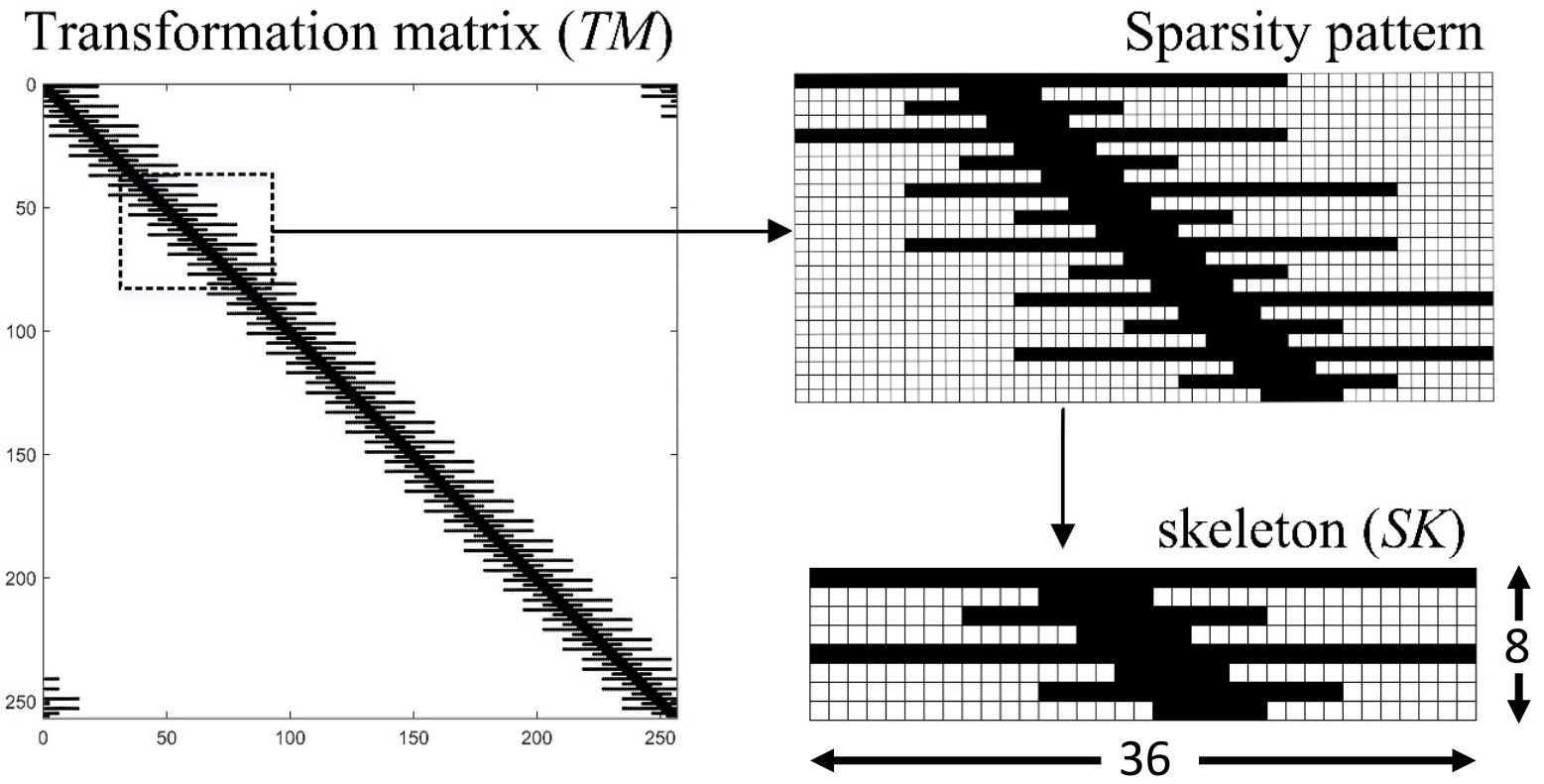
- Formulation:

$$C = TM \cdot X, \text{where } TM = \prod_k T_{ak}^{p}$$

- TM matrix is highly sparse
  - ➢ Large number of multiply-by-zero operations
  - ➢ Large memory footprint consisting of zeroes
- Goals:
  - ➢ SIMD-friendly operations on non-zero values only
  - ➢ Customized DMA routines for efficient bandwidth utilization



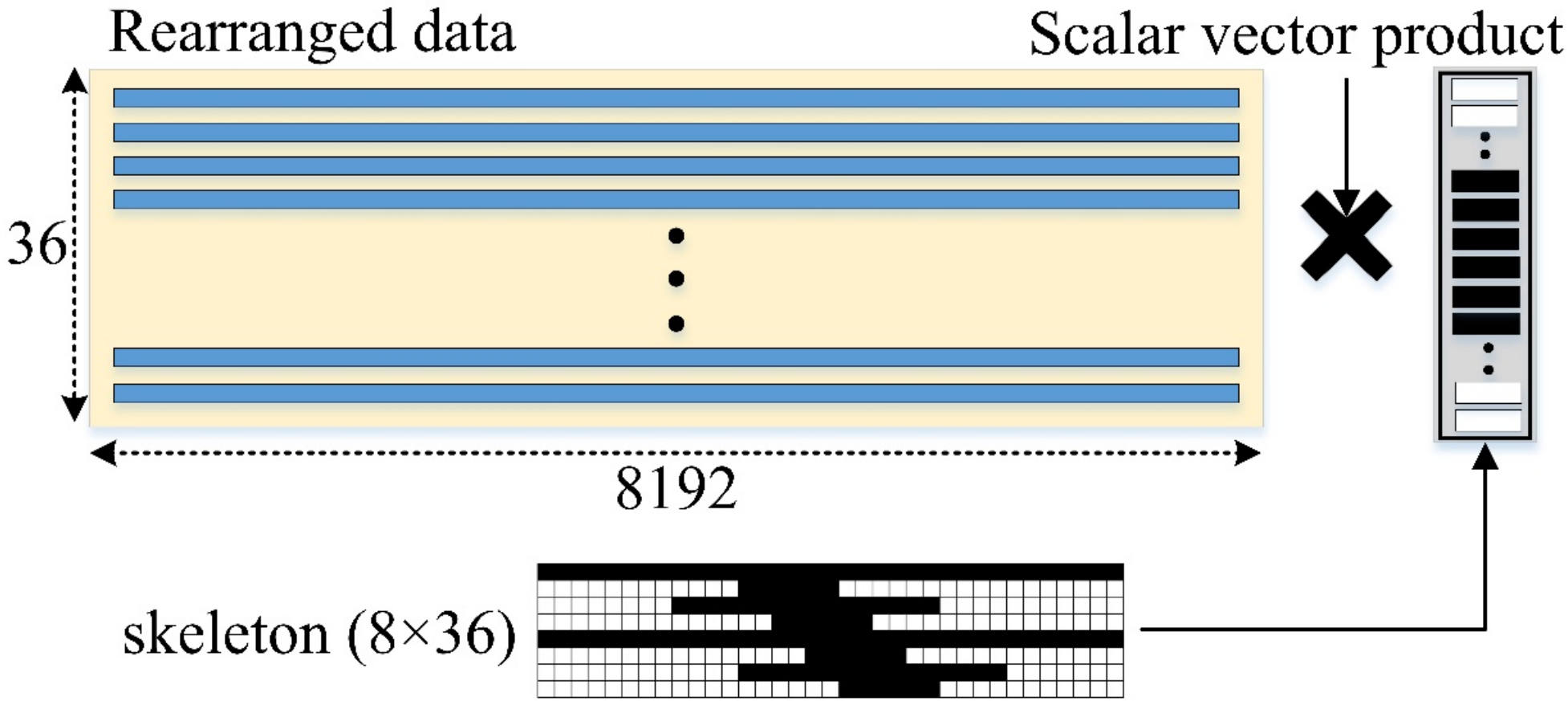
Sparse Matrix Skeleton



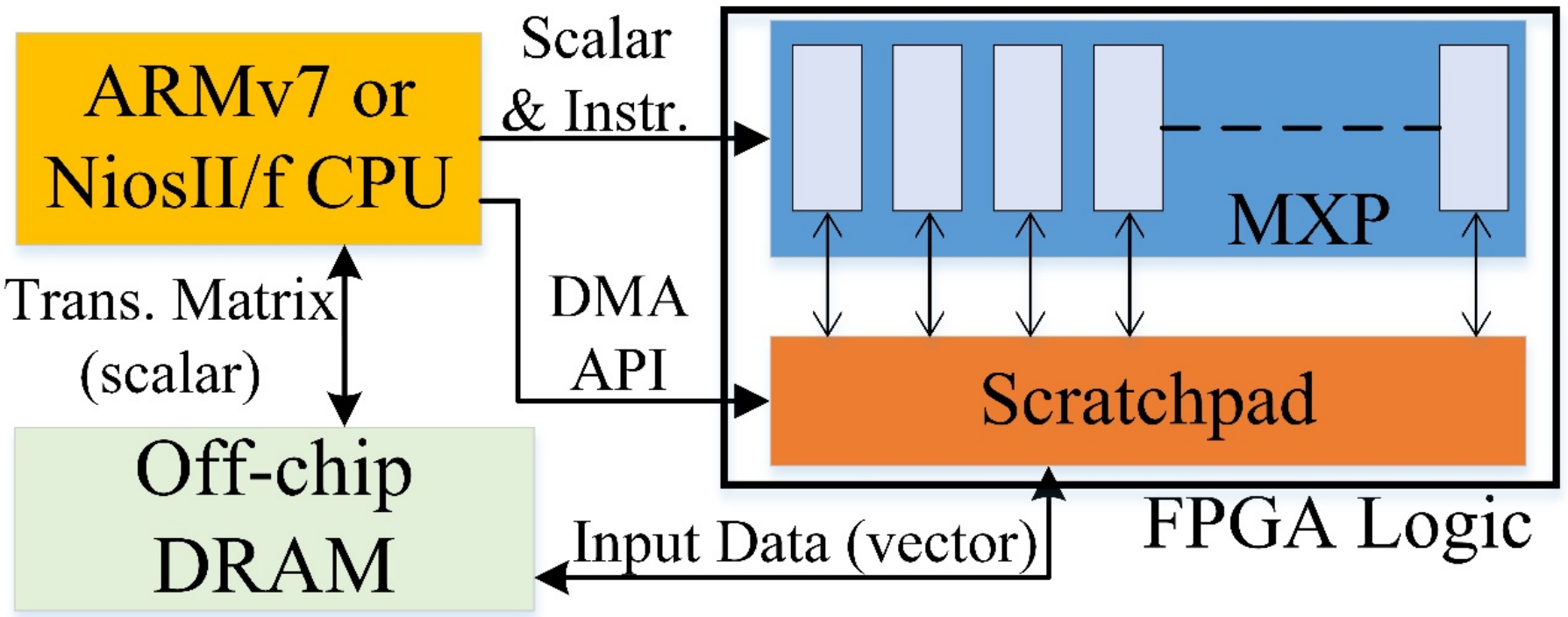
- Remove multiply-by-zero operations
- Reduction in memory footprint of *TM*.

Modified Matrix-Form 1-D DWT

N = 65536

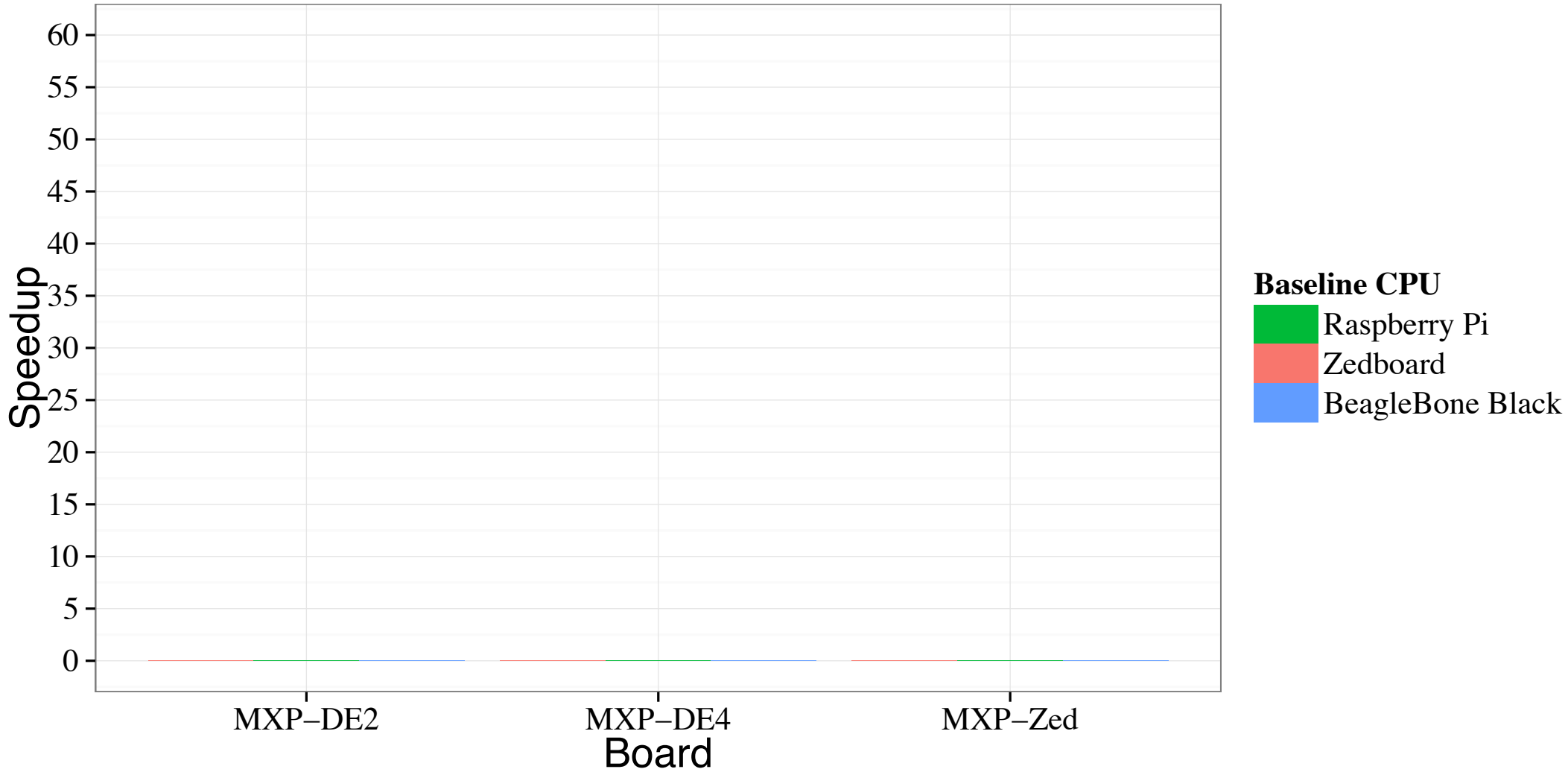


VectorBlox MXP



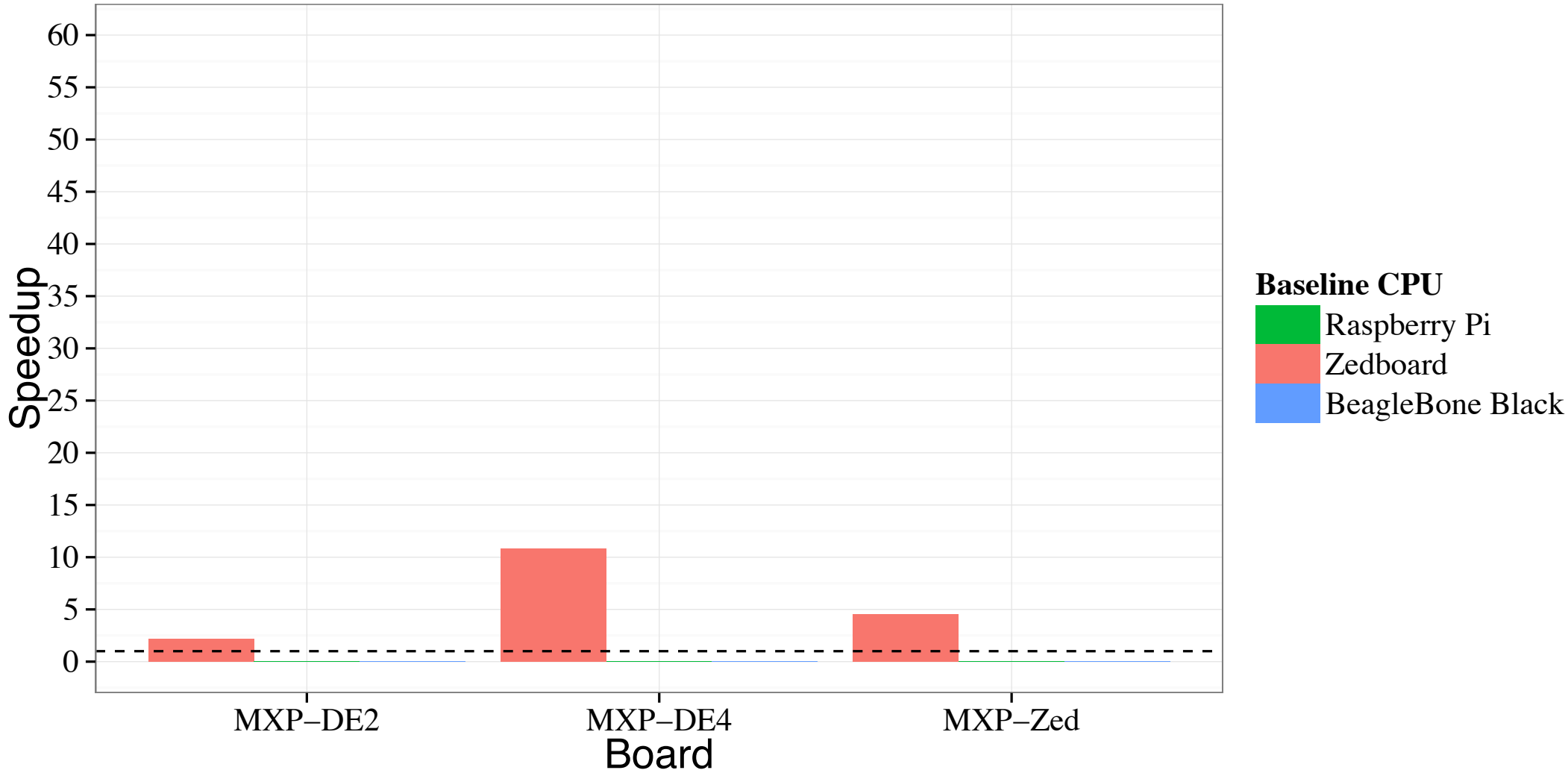
- Lanes: 16-32
- Scratchpad: 64-128 KB
- DMA bandwidth: 4-32 B/cycle

Results - Speedup



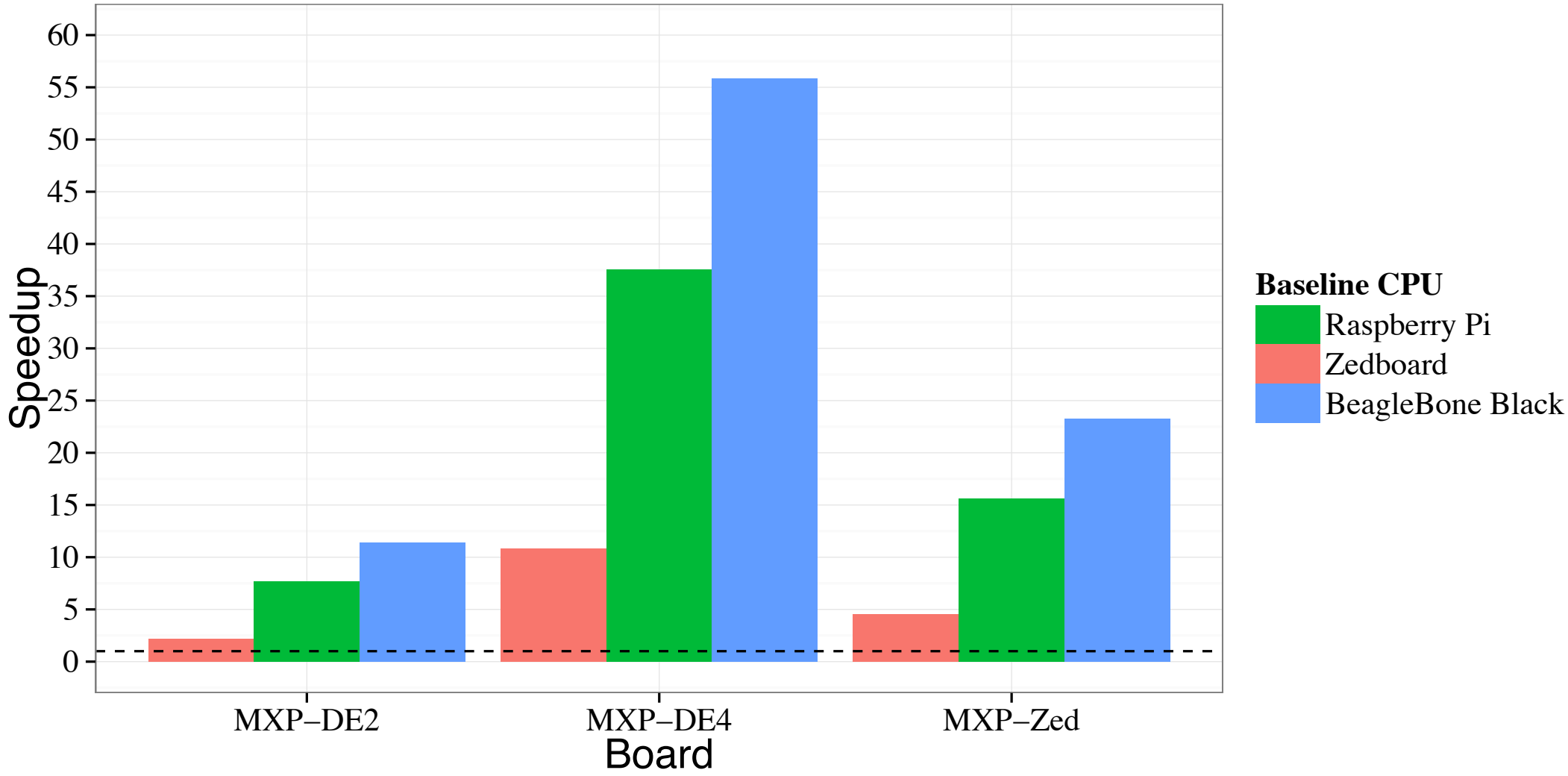
$N = 2^{16}, L = 6 \; and \; k = 3$

Results - Speedup



$N = 2^{16}, L = 6 \; and \; k = 3$

Results - Speedup



$N = 2^{16}, L = 6 \; and \; k = 3$

Summary

- We propose a **Modified Matrix-Form** scheme to unlock inherent parallelism in 1-D DWT
- We exploit the sparsity pattern in *TM* to reduce complexity from $O(n^2)$ to $O(n)$ using :
  - Skeletons to avoid wasteful multiply-by-zero operations
  - Rearrangement of input samples
- Speedups of **12-103x** over state-of-the-art in-built `signal` library in Octave (`dwt` function)

Experimental Setup

Matrix-form 1-D DWT

Sparse Matrix Skeletons

CPU

- Optimized OpenBLAS routines in Octave and C (compiled with –O3)

- Performance measured using PAPI v5.4.3

- 32b ARMv7 on Beaglebone Black, Zedboard, and ARMv6 on Raspberry Pi

CPU + MXP

- Customized DMA routines for data transfer between host and MXP

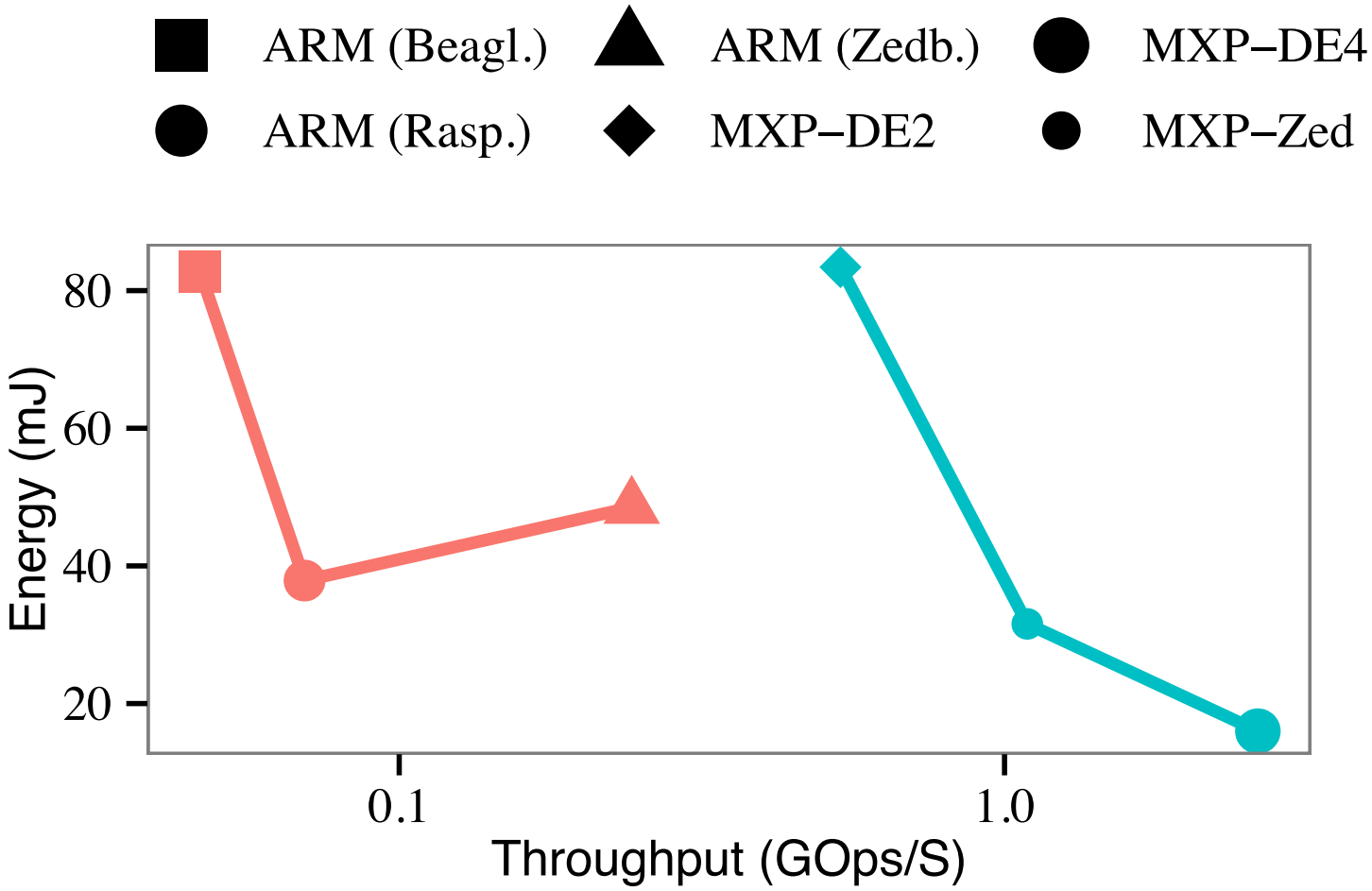
- 16-32 vector lanes

- 64-128KB scratchpad memory

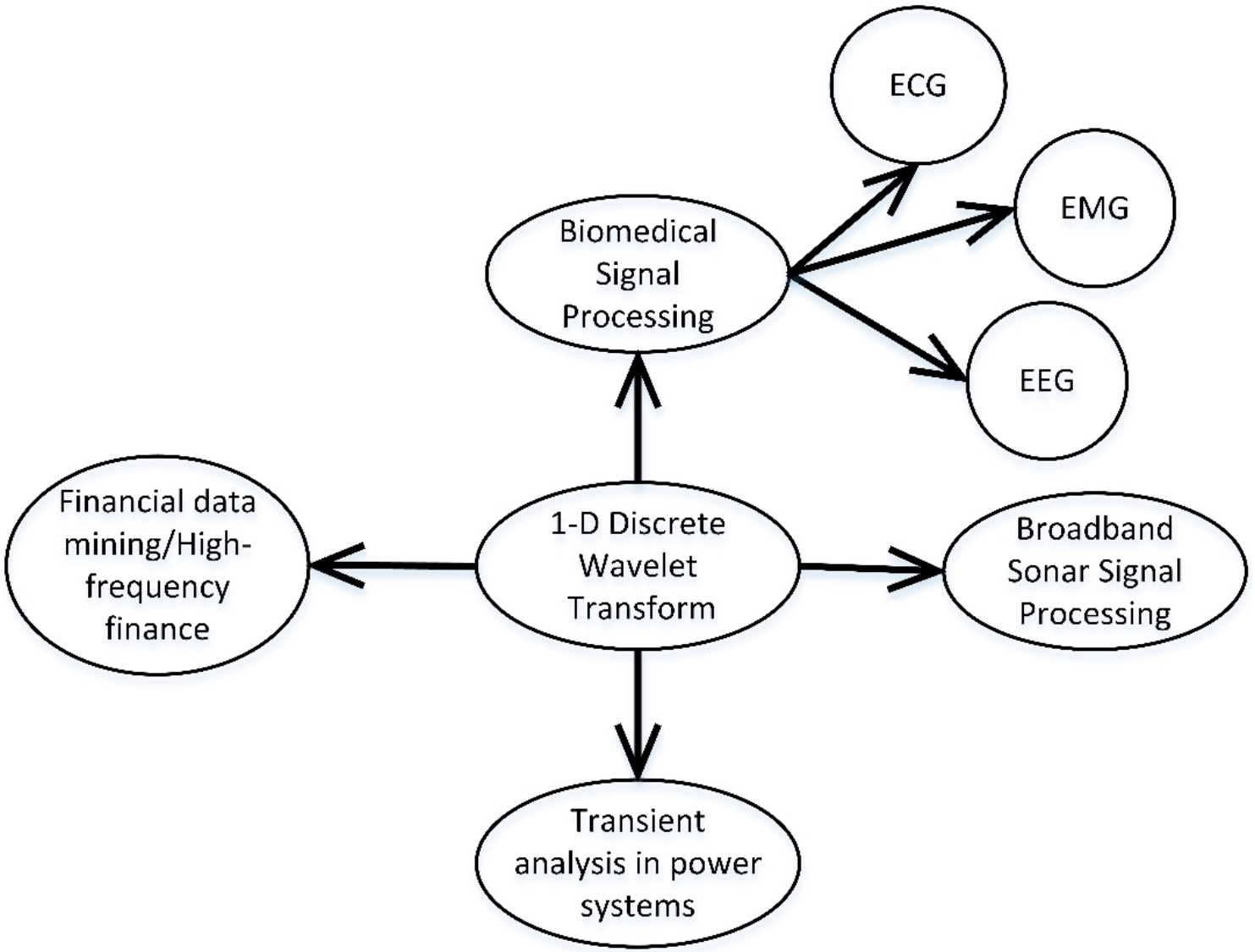
- Performance measured using MXP Timing API

- Altera DE2/DE4 and Zedboard

Results - Throughput



$N = 2^{16}, L = 6\ and\ k = 3$



CHALLENGES:

- Large volume of data
- Strict real-time processing constraints
- High accuracy demands
- Energy constraints, especially in **embedded** systems

Modified Matrix-Form 1-D DWT

Rearrangement