Level-3 BLAS on Myriad Multi-Core Media-Processor SoC

Tomasz Szydzik[1] Marius Farcas[2] Valeriu Ohan[2] David Moloney[3]

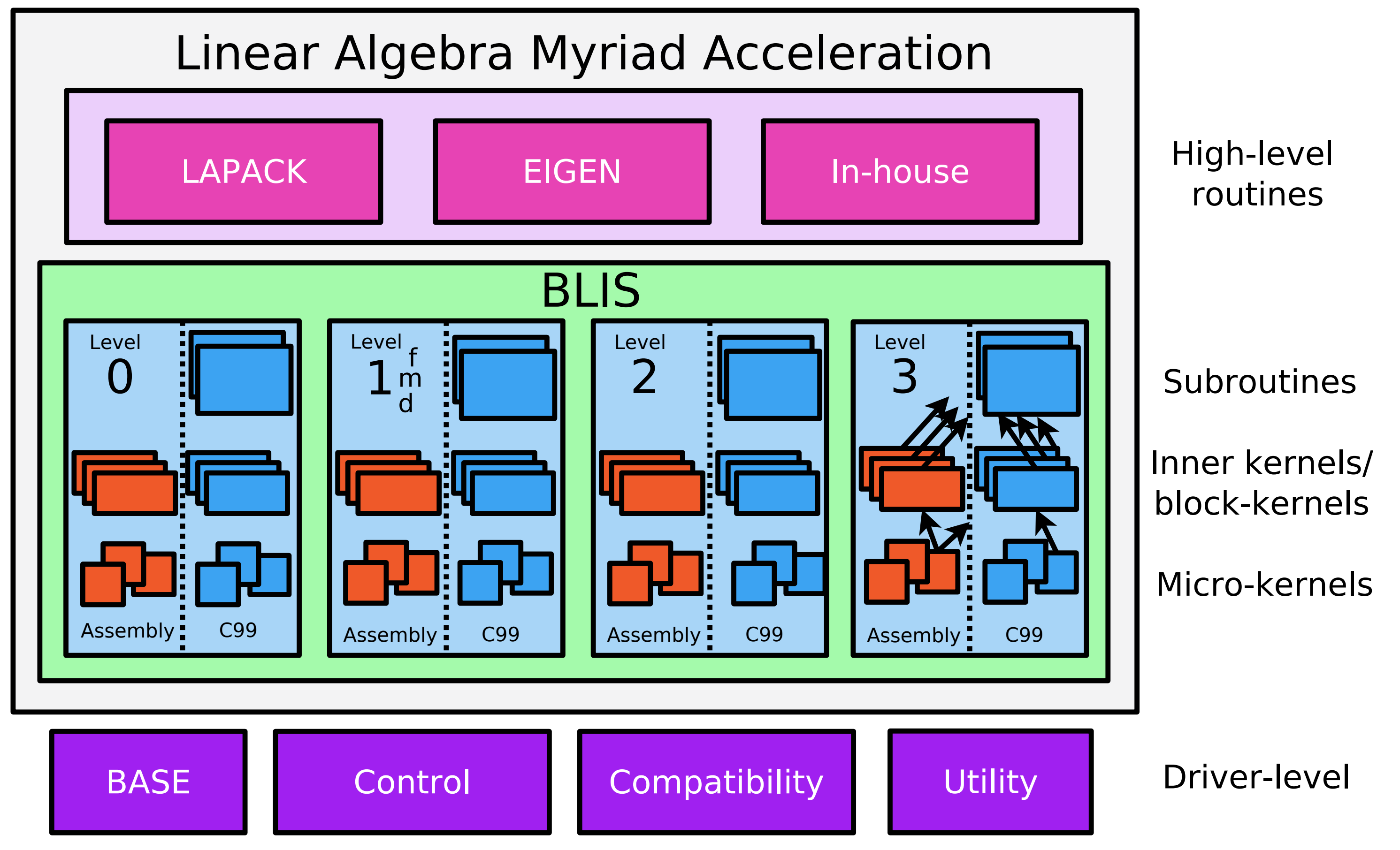
[1]Insititute for Applied Microelectronics, University of Las Palmas of Gran Canaria, [2]Codecart, [3]Movidius Ltd.

BLAS-like Library Instantiation Subprograms (*BLIS*)

The Basic Linear Algebra Subprograms (BLAS) are a set of low-level subroutines that perform common linear algebra operations. *BLIS* is a software framework for instantiating high-performance BLAS-like dense linear algebra libraries. BLIS[1] was chosen over GoTOBLAS, ATLAS, etc. due to its portable micro-kernel architecture and active user-base.

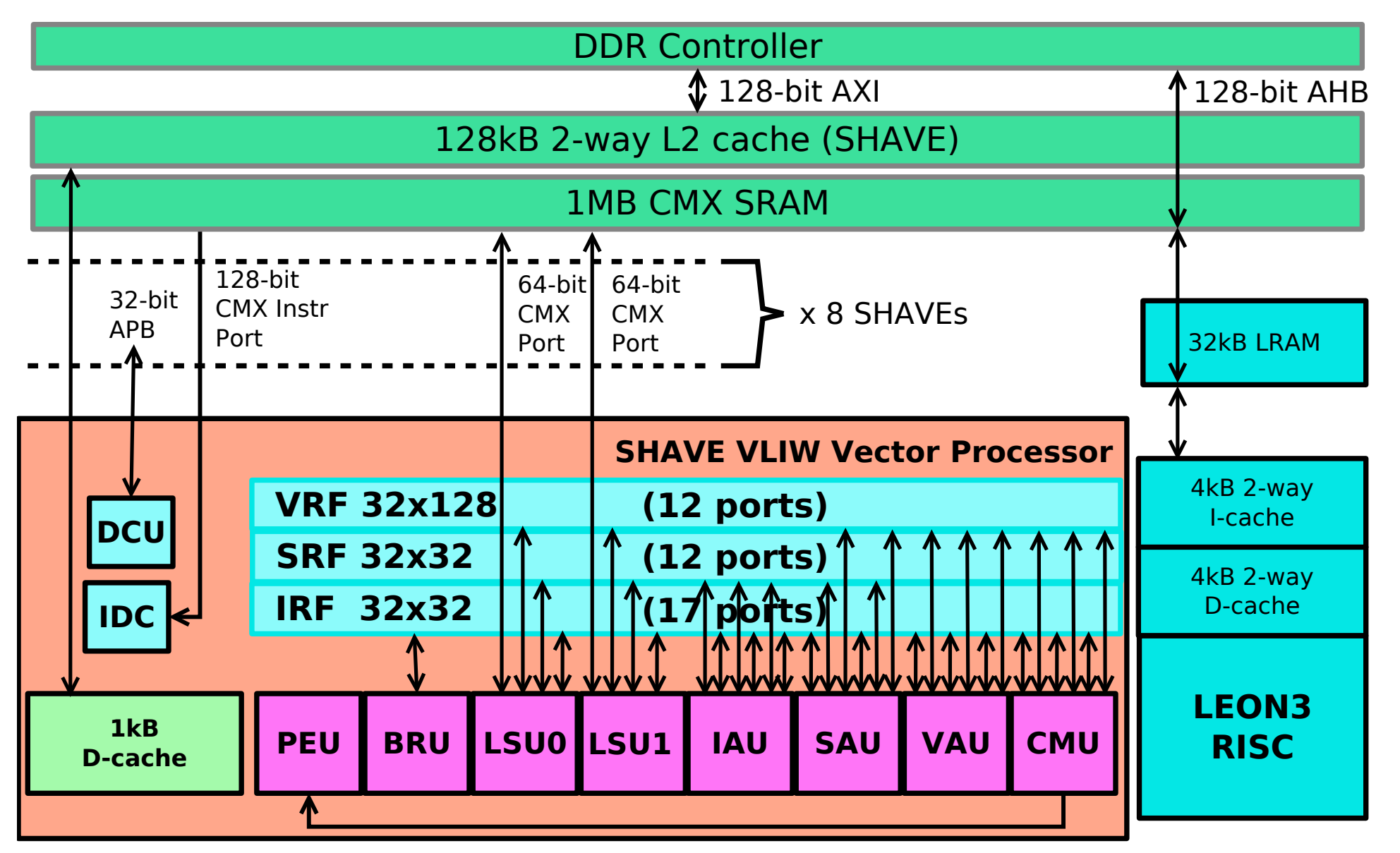
BLIS features

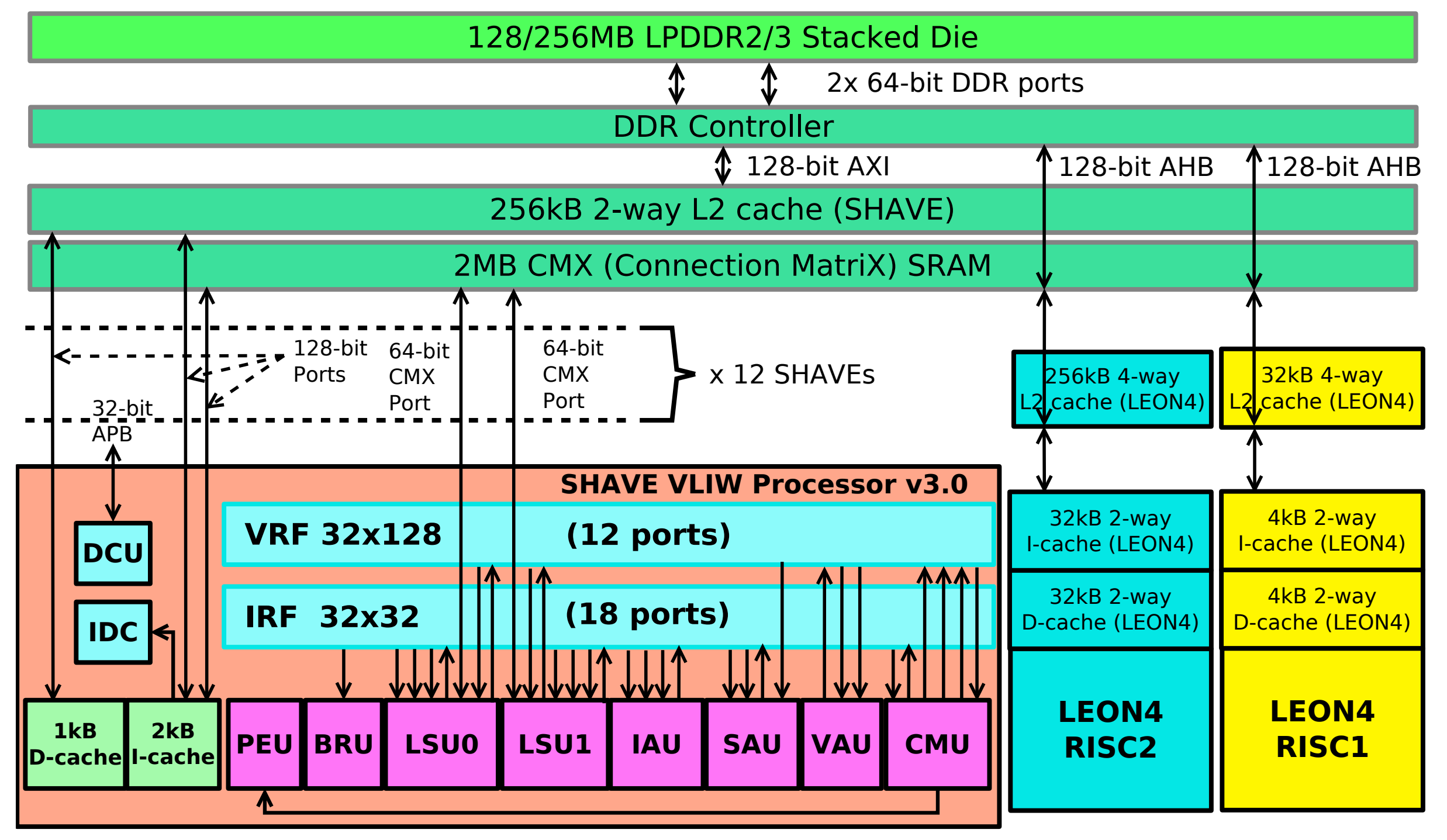
- ISO C99 code with flexible BSD license.
- Support for BLAS API calling conventions.
- Competitive performance [2].
- Multi-core friendly.
- Multi-layer API and code identifying and isolating a key set of computational kernels.
- Modularity and extensiveness.
- Portability (x86, x64, TIC66x, PowerPC, etc.) that doesn't impede high performance [3].
- Foundation for mixed precision (experimental).



The Myriad media-processor SoCs

Myriad architecture prioritises power-efficient operation and area efficiency. In order to guarantee sustained high performance and minimise power the proprietary SHAVE (Streaming Hybrid Architecture Vector Engine) processor was developed. Data and Instructions reside in a shared Connection MatriX (CMX) memory block shared by all Shave processors. Data is moved between peripherals, processors and memory via a bank of software-controlled DMA engines.




128/256MB LPDDR2/3 Stacked Die
2x 64-bit DDR ports
DDR Controller
128-bit AXI
128-bit AHB
128-bit AHB
256kB 2-way L2 cache (SHAVE)
2MB CMX (Connection MatriX) SRAM
128-bit Ports
64-bit CMX Port
64-bit CMX Port
x 12 SHAVEs
32-bit APB
256kB 4-way L2 cache (LEON4)
32kB 4-way L2 cache (LEON4)
SHAVE VLIW Processor v3.0
VRF 32x128 (12 ports)
IRF 32x32 (18 ports)
DCU
IDC
1kB D-cache
2kB I-cache
PEU BRU LSU0 LSU1 IAU SAU VAU CMU
32kB 2-way I-cache (LEON4)
32kB 2-way D-cache (LEON4)
4kB 2-way I-cache (LEON4)
4kB 2-way D-cache (LEON4)
LEON4 RISC2
LEON4 RISC1

Myriad 1 architecture highlights [4]

- 65nm ultra-low power architecture ($\leq 0.35W@180MHz$) with 11 power islands.
- Hardware support for SIMD, matrix transpose, sparse data, sqrt@fp16, predicated execution...
- Heterogeneous SoC: 1 Leon3@fp64 + 8 Shaves@fp32.
- 32KB LRAM, 1MB CMX, 16/64MB DDR, DMAs.
- Power efficiency of 1Tops/W (max 8-bit equivalent).

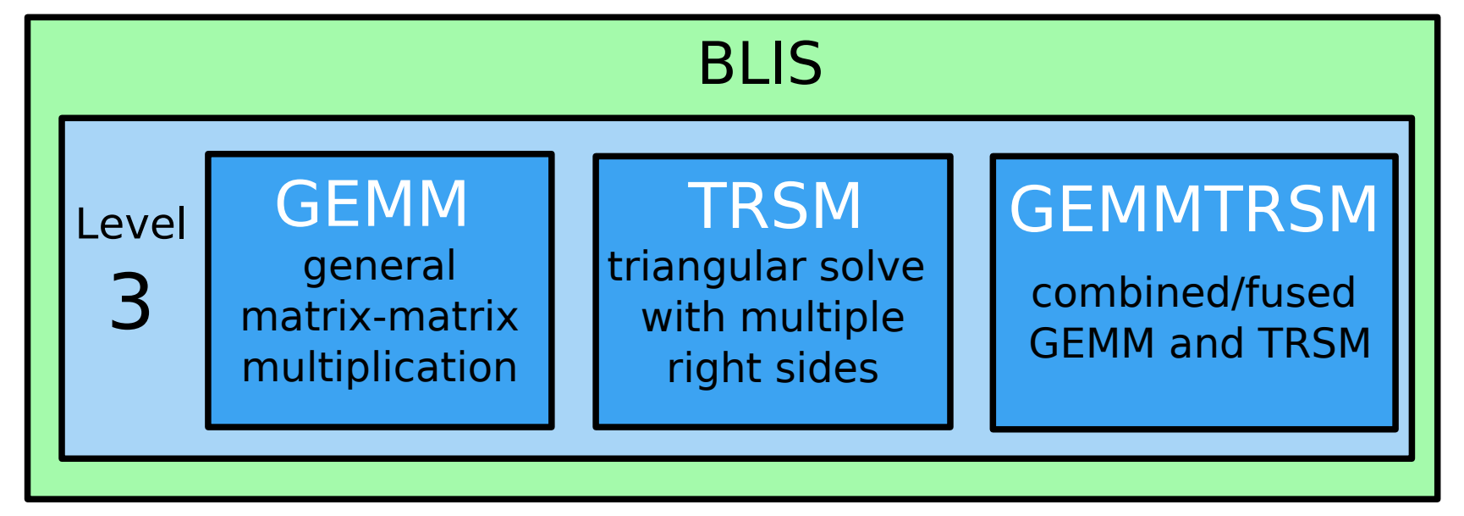
Myriad 2 architecture highlights [4]

- 28nm ultra-low power ($\leq 0.5W@600MHz$) with 17 power islands.
- Extended hardware support over Myriad 1: clock-gating, hard-wired configurable accelerators for imaging and vision, etc.
- Heterogeneous SoC: 2 Leon4@fp64 + 12 Shaves@fp32.
- 256+32KB LRAM, 2MB CMX, DDR3 support, DMAs.
- Power efficiency of 2Tops/W (max 16-bit equivalent).

Level-3 BLAS using BLIS on Myriad

BLIS Level-3 micro-kernels

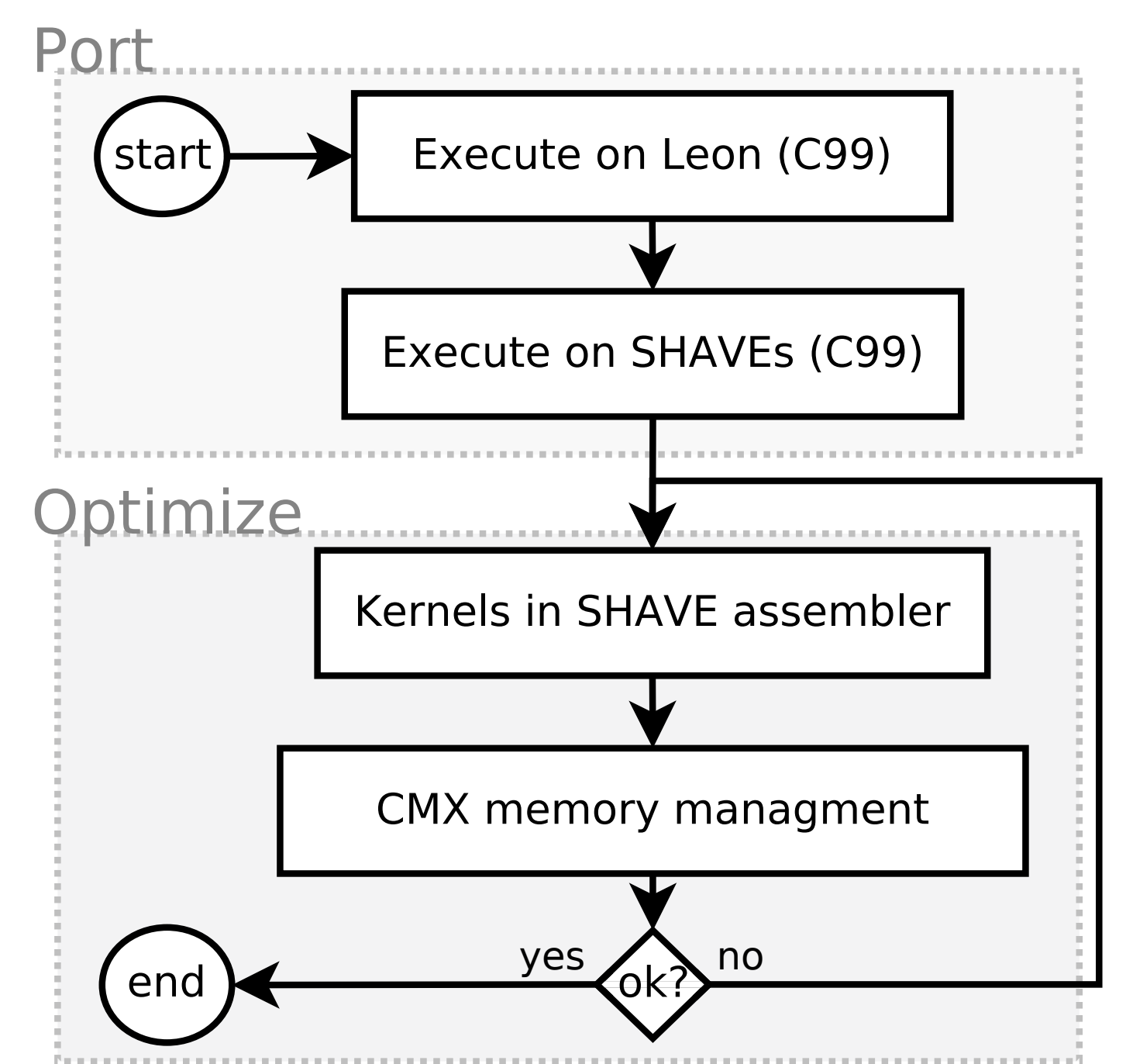
BLIS defines three Level-3 micro-kernels. Implementation of the fused GEMM-TRSM kernel is optional.


BLIS
Level 3
GEMM
general matrix-matrix multiplication
TRSM
triangular solve with multiple right sides
GEMMTRSM
combined/fused GEMM and TRSM

Level-3 BLAS on Myriad

The BLIS ISO C99 code allowed straightforward compilation for Myriad. Following optimizations consisted of:

- micro-kernels implementation in SHAVE assembler,
- and memory management/allocation



Memory focused optimizations

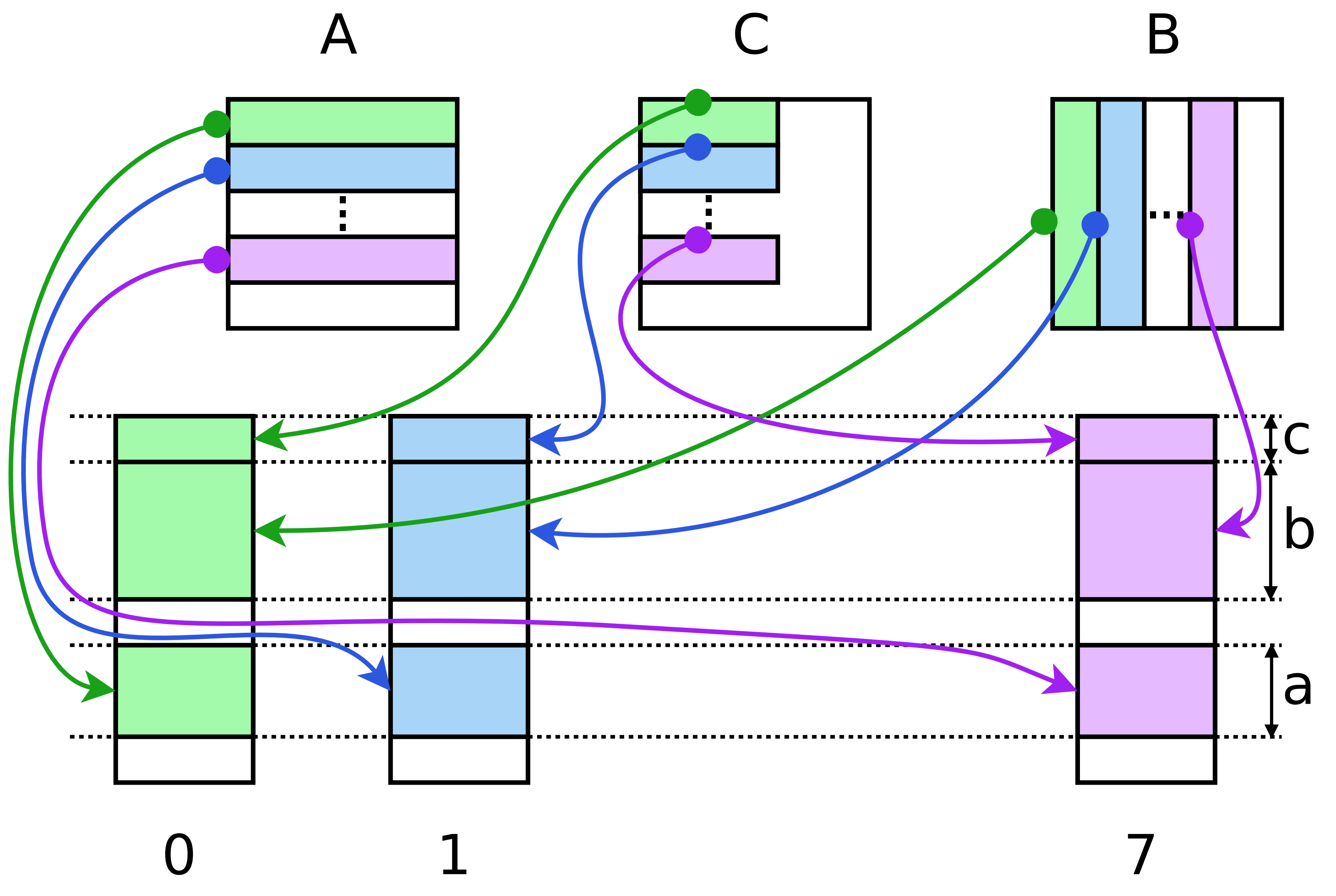
- Double/triple buffering of arguments.
- Buffers shared by all SHAVEs.
- Data passing using pointer arithmetic.
- Overlapped DMA accesses.

GEMM and TRSM operations

The xGEMM and xTRSM routines are the typical benchmarks of the Level-3 BLAS performance of an implementation. Basic information on the operations and the computational complexity of these two routines are presented below.

Routine	Operation	Flops	Comments
GEMM	$C := \alpha \cdot op(A) \cdot op(B) + \beta \cdot C$	$2mnk$	$op(X) =$
TRSM	$C := \alpha \cdot op(A^{-1})C$	nm^2	X, X^T, X^H, C
	$C := \alpha \cdot C \cdot op(A^{-1})$	mn^2	is $m \times n$

Mapping of matrix blocks on CMX (SGEMM)

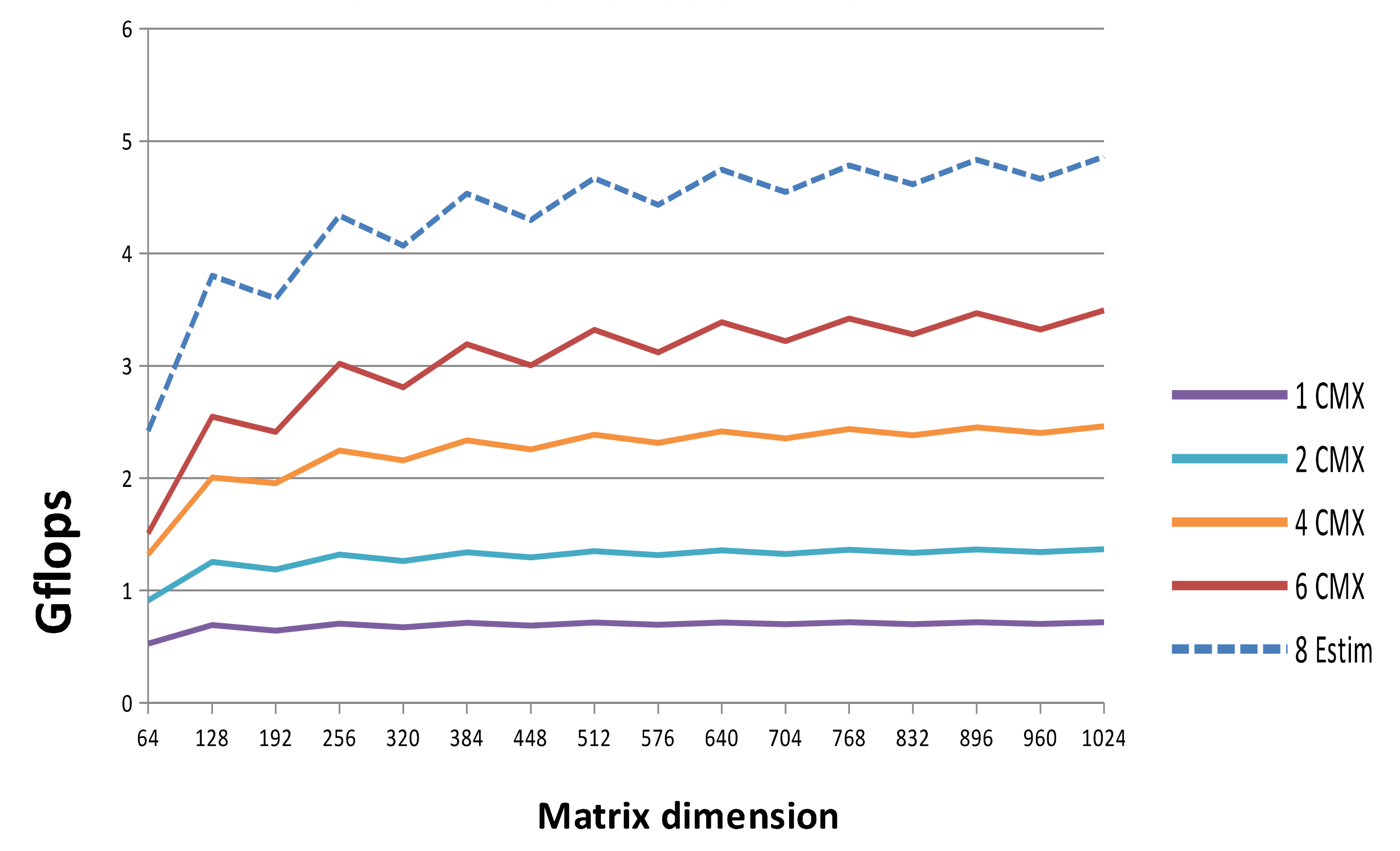


where,

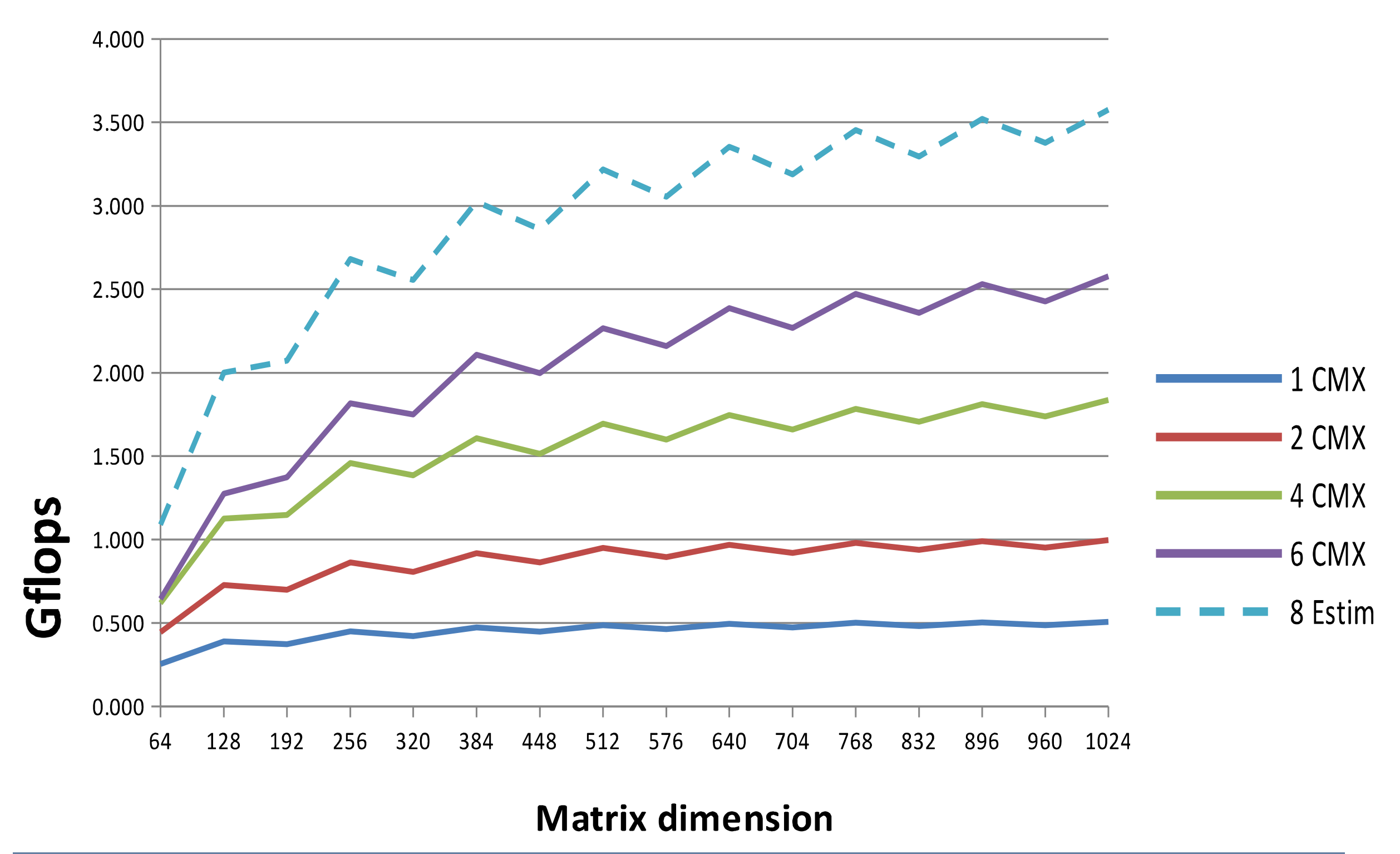
- the numbers represent SHAVE cores,
- the boxes above them, their local CMX memory slice,
- buffering of A, B and C is not shown in order to preserve clarity.

SGEMM and STRSM performance

SGEMM vs #cores and matrix width:



STRSM vs #cores and matrix width:



Results

Architecture	Power [W]	Performance [GFLOPS] core	Performance [GFLOPS] system	[GFLOPS/W]
SGEMM				
Cell [5]	20	23.01	184.8	9.22
C6678 [3]	10	10.3	79.6	7.96
Myriad 1 [4]	0.35	0.75	4.92	14.06
STRSM				
Cell [5]	20	16.47	131.8	6.59
C6678 [3]	10	8.7	59.5	5.95
Myriad 1 [4]	0.35	0.5	3.57	10.2

Conclusions

- High-performance required hand optimized micro-kernels and memory management.
- Most effort spent on tuning memory management.
- Port to Myriad 2 (expected 10× more efficient).

[1] F. G. Van Zee and R. A. van de Geijn, *"BLIS: A framework for rapidly instantiating BLAS functionality,"* ACM Transactions on Mathematical Software, 2013, accepted.

[2] F. G. Van Zee, T. Smith, F. D. Igual, M. Smelyanskiy, X. Zhang, M. Kistler, V. Austel, J. Gunnels, T. M. Low, B. Marker, L. Killough, and R. A. van de Geijn, *"The BLIS framework: Experiments in portability,"* ACM Transactions on Mathematical Software, 2013, accepted.

[3] M. Ali, E. Stotzer, F. Igual, and R. van de Geijn, *"Level-3 blas on the TIC6678 multi-core DSP,"* in Computer Architecture and High Performance Computing (SBAC-PAD), 2012 IEEE 24th International Symposium on, Oct 2012, pp. 179–186.

[4] David Moloney, *1TOPS/W Software Programmable Media Processor*, HotChips 2011 (HC23), Stanford, California, August 23rd, 2011.

[5] J. Kurzak, A. Buttari, and J. Dongarra, *"Solving systems of linear equations on the cell processor using cholesky factorization,"* Parallel and Distributed Systems, IEEE Transactions on, vol. 19, no. 9, pp. 1175–1186, Sept 2008

Acknowledgement

This work has been carried out under the HiPEAC PhD industrial internship programme and Excess project (www.excess-project.eu) while being co-founded by Movidius and the European Union under the grant EU ICT-287759 and 611183.