The Cost of Parallelism: A Measured Comparison of Sequential and Systolic INT8 GEMM on an FPGA

Boris Gordeev
Department of Electrical and Computer Engineering
Northeastern University, Boston, MA, USA
gordeev.b@northeastern.edu

Abstract

Spatial parallelism reduces computation latency, but its return depends on how effectively a workload occupies the hardware. This study compares sequential MAC reuse with a conventional output-stationary systolic array for one isolated 4×4 signed INT8 matrix multiplication on an Intel Cyclone 10 GX FPGA. The systolic engine takes 11 cycles versus 81, a 7.4× cycle speedup, while using 16 rather than one DSP block and 822 rather than 506 ALMs. Post-fit restricted maximum frequencies are similar. A schedule-derived occupancy analysis explains the sublinear return: 64 useful MACs occupy only 36.4% of the array's operation-level MAC slots.

1 Problem and Motivation

General matrix multiplication (GEMM) is central to neural-network inference, where integer arithmetic enables low-precision computation [1]. A hardware designer can reuse one multiply-accumulate (MAC) datapath over time or distribute computation across many datapaths. Systolic arrays are established architectures [2], including in FPGA inference research [4]; this work introduces no new dataflow. Its question is empirical: how much latency does spatial execution save for small INT8 GEMM, and what does that saving cost in resources, frequency, and occupancy?

2 Approach

Two SystemVerilog engines implement identical 4×4 products with signed INT8 operands and INT32 accumulation. They share a logical start/busy/done interface and receive identical operands. Both target the Cyclone 10 GX 10CX220YF780E5G using Quartus Prime Pro 26.1, a 100 MHz constraint, and the slow 900 mV, 0 °C timing corner. Separate core builds with virtual matrix pins isolate each engine's resource and timing results from the combined verification design.

The sequential engine reuses one MAC under FSM control: one initialization cycle, 64 computation cycles, and 16 result-store cycles total 81. The systolic engine contains sixteen processing elements (PEs). Skewed inputs move A values east and B values south; each PE retains one INT32 output sum. One load cycle followed by ten enabled feed cycles gives 11 cycles overall. All accumulators connect directly to the result bus, avoiding a separate store phase.

3 Results and Analysis

Post-fit resource counts are 506 versus 822 ALMs, 819 versus 909 registers, and one versus sixteen DSP blocks for sequential and systolic execution, respectively. Neither core uses M20K memory. Restricted maximum frequency is 162.6 versus 165.8 MHz. Thus, the measured builds exchange 16× DSPs and 1.62× ALMs for approximately 7.4× fewer cycles without a timing penalty in this comparison. One compilation per engine cannot establish that the small frequency difference is systematic.

Dividing cycle counts by each core's restricted maximum frequency yields estimated latencies of 498.2 and 66.3 ns, or approximately 7.5× estimated time speedup. Useful throughput is 0.79 versus 5.82 MACs per cycle. Counting multiplication and addition separately gives derived core estimates of 0.26 and 1.93 GOPS. These are calculations, not directly timed board performance or sustained system throughput.

ALM growth is smaller than DSP growth because the baseline already requires operand multiplexing, counters, and result-selection logic, whereas the mesh maps multiplication to dedicated DSPs and distributes accumulation locally.

The wavefront's useful occupancy rises, peaks at twelve of sixteen PEs, and falls; it never reaches full useful occupancy for this operation. The 64 useful MACs span 176 available slots across sixteen PEs and eleven cycles, giving 36.4% utilization, or 40% over the ten feed cycles alone. The sequential baseline uses its multiplier in 64 of 81 cycles, giving 79.0%. These workload-specific arithmetic measures do not measure switching activity or energy efficiency.

4 Verification

Verification combines an independent Python reference, directed and randomized self-checking RTL tests, and bit-exact comparison between structurally different engines. The top-level suite passed five sequential, five systolic, and four engine-agreement checks. Both engines passed fixed-vector FPGA self-tests. Signal Tap confirmed the systolic engine's eleven-cycle busy window; its pass indicator asserted later after the comparison stage registered the result. The sequential latency remains an RTL measurement. Directed cases include positive, negative, zero, and identity operands; seeded random vectors broaden coverage. A separate Python verifier also checked sixteen output elements against the golden model outside the simulation environment.

5 Limitations and Significance

Only one matrix size and one device were evaluated, with one compilation per engine. The wide parallel interface excludes external transfers; no DRAM system, streaming, tiling, requantization, power, energy, or neural-network accuracy was evaluated. Larger configurations remain analytical predictions rather than experimental results. For the documented square workload and equally sized array, operation-level utilization is $N/(3N-1)$, approaching one-third as N grows; increasing array size alone therefore does not improve this utilization. Longer workloads on a fixed array require separate evaluation. The contribution is a controlled, hardware-validated resource/latency comparison: spatial execution substantially reduces this isolated operation's latency, while its schedule leaves much of the installed arithmetic capacity unused. Future work should independently vary workload and array dimensions and evaluate placement variation and energy.

Acknowledgements

OpenAI ChatGPT [3] provided writing assistance throughout; experimental measurements originate in the author’s research report.

References

[1] B. Jacob et al. 2018. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. Proceedings of CVPR.

[2] H. T. Kung and C. E. Leiserson. 1979. Systolic Arrays (for VLSI). Sparse Matrix Proceedings.

[3] OpenAI. 2026. ChatGPT. https://chatgpt.com/. Writing assistance used September 30, 2026.

[4] X. Wei, C. H. Yu, P. Zhang, Y. Chen, Y. Wang, H. Hu, Y. Liang, and J. Cong. 2017. Automated Systolic Array Architecture Synthesis for High Throughput CNN Inference on FPGAs. Proceedings of DAC, 1–6.