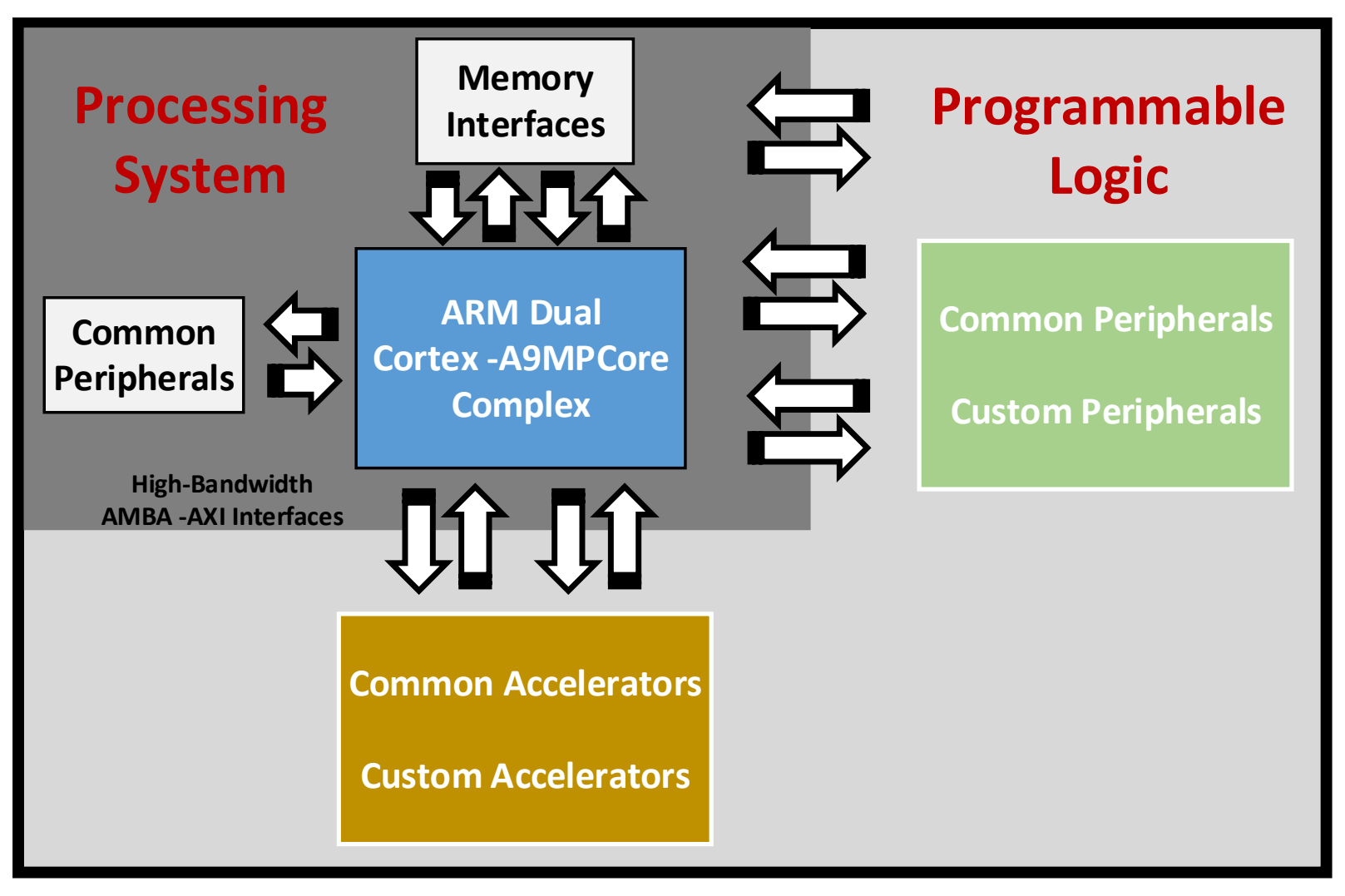
A Zynq-based Testbed for the Experimental Benchmarking of Algorithms Competing in Cryptographic Contests

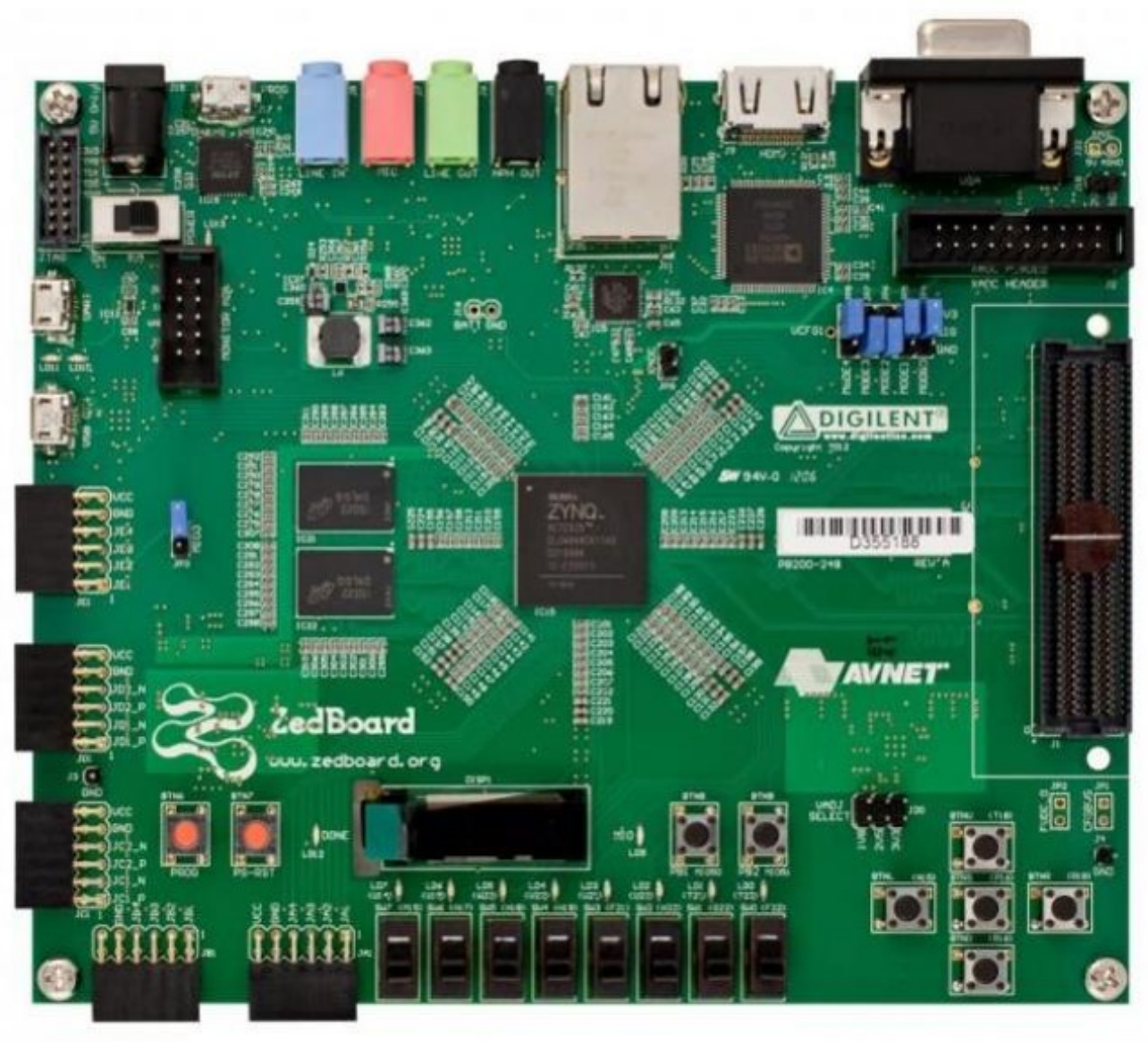
Farnoud Farahmand, Ekawat Homsirikamol, and Kris Gaj
Department of Electrical and Computer Engineering, George Mason University, Fairfax, Virginia 22030, USA

INTRODUCTION

- Hardware performance evaluation of candidates competing in cryptographic contests, such as SHA-3 and CAESAR, is very important for ranking their suitability for standardization.
- One of the most essential performance metrics is the throughput, which highly depends on the algorithm, hardware implementation architecture, coding style, and options of tools. The maximum throughput is calculated based on the maximum clock frequency supported by each algorithm.
- In this project, we have developed a universal testbed, which is capable of measuring the maximum clock frequency experimentally, using a prototyping board. We are targeting cryptographic hardware cores, such as implementations of SHA-3 candidates. Our testbed is designed using a Zynq platform and takes advantage of software/hardware co-design.



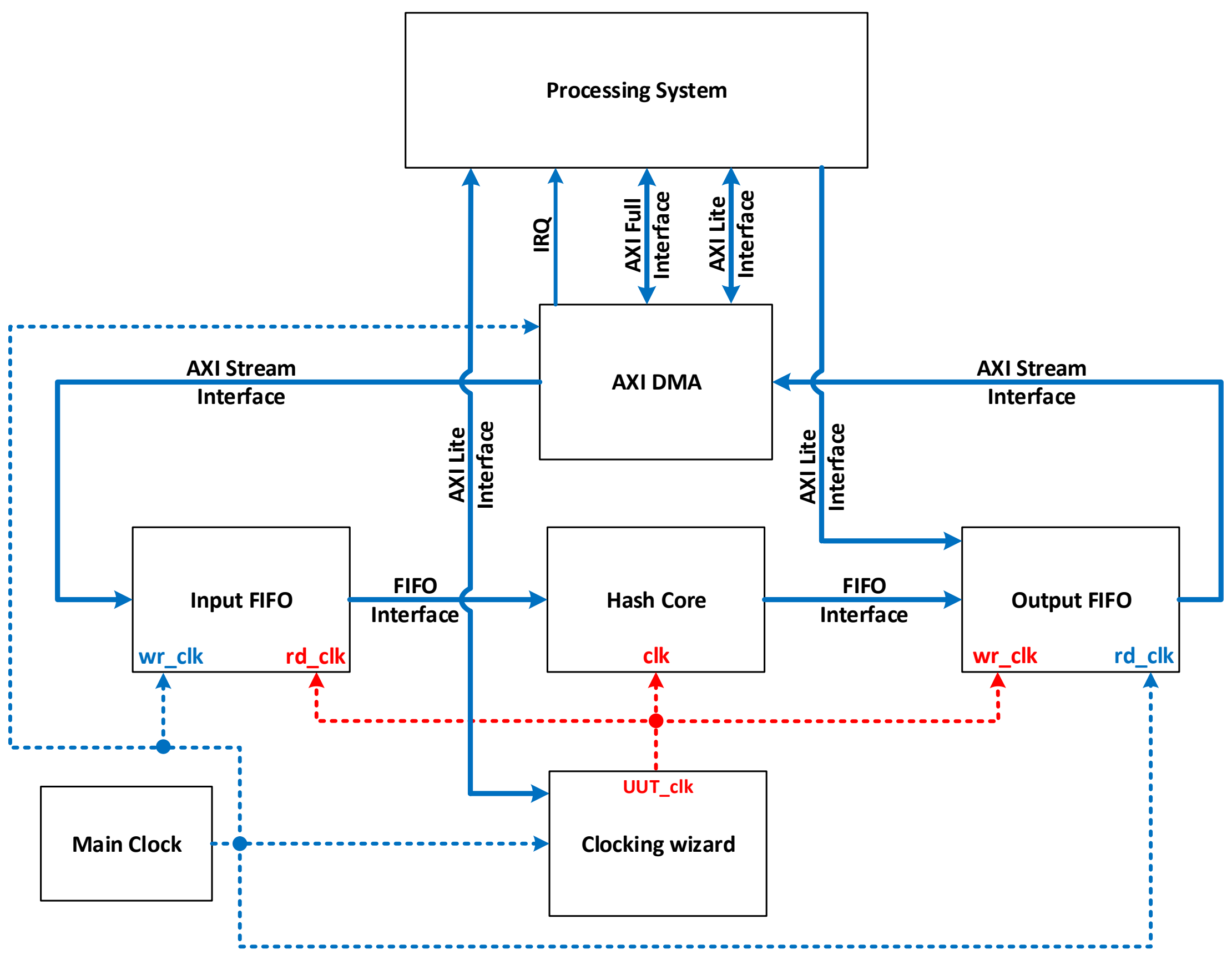
- We measured the maximum clock frequency and the execution time of 12 Round 2 SHA-3 candidates experimentally on ZedBoard and compared the results with the frequencies reported by Xilinx Vivado.



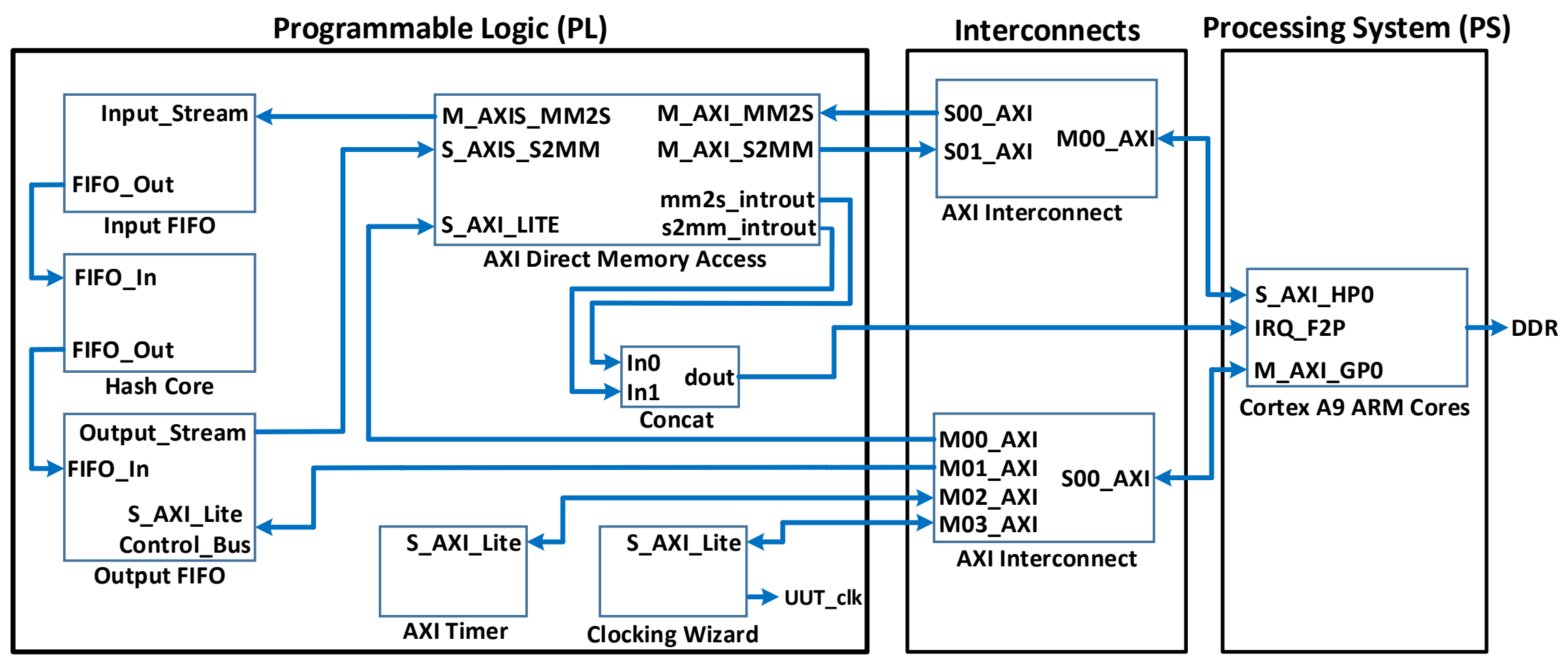
- Experimental benchmarking of cryptographic algorithms has been performed previously on different platforms other than Zynq.
  1. Maximum frequency of SHA-256 has been measured experimentally using the SLAAC-1V board based on Xilinx Virtex VCV 1000.
  2. Experimental measurement of the hardware performance of 14 round 2 SHA-3 candidates has been performed using the SASEBO-GII FPGA board.

SYSTEM DESIGN

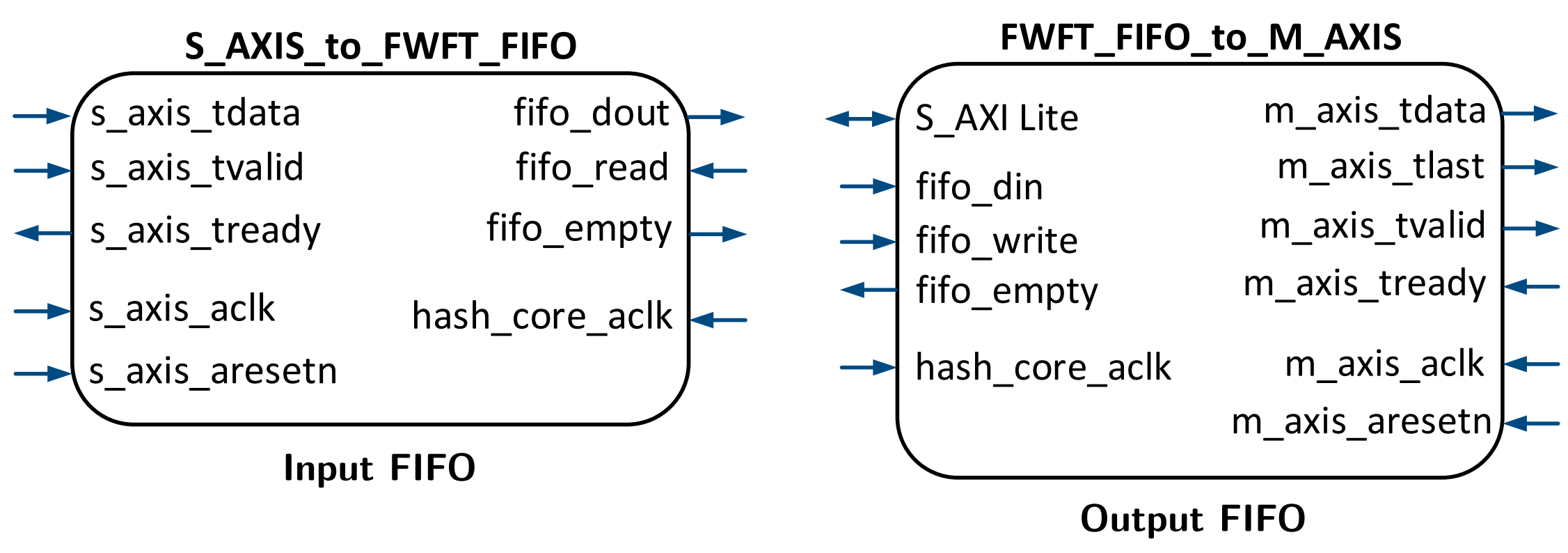
- **Simplified block diagram of the PL side with the indication of two independent clocks:**



- **Block Diagram of the Testbed with the division into Programmable logic (PL), Interconnects, and Processing System (PS):**



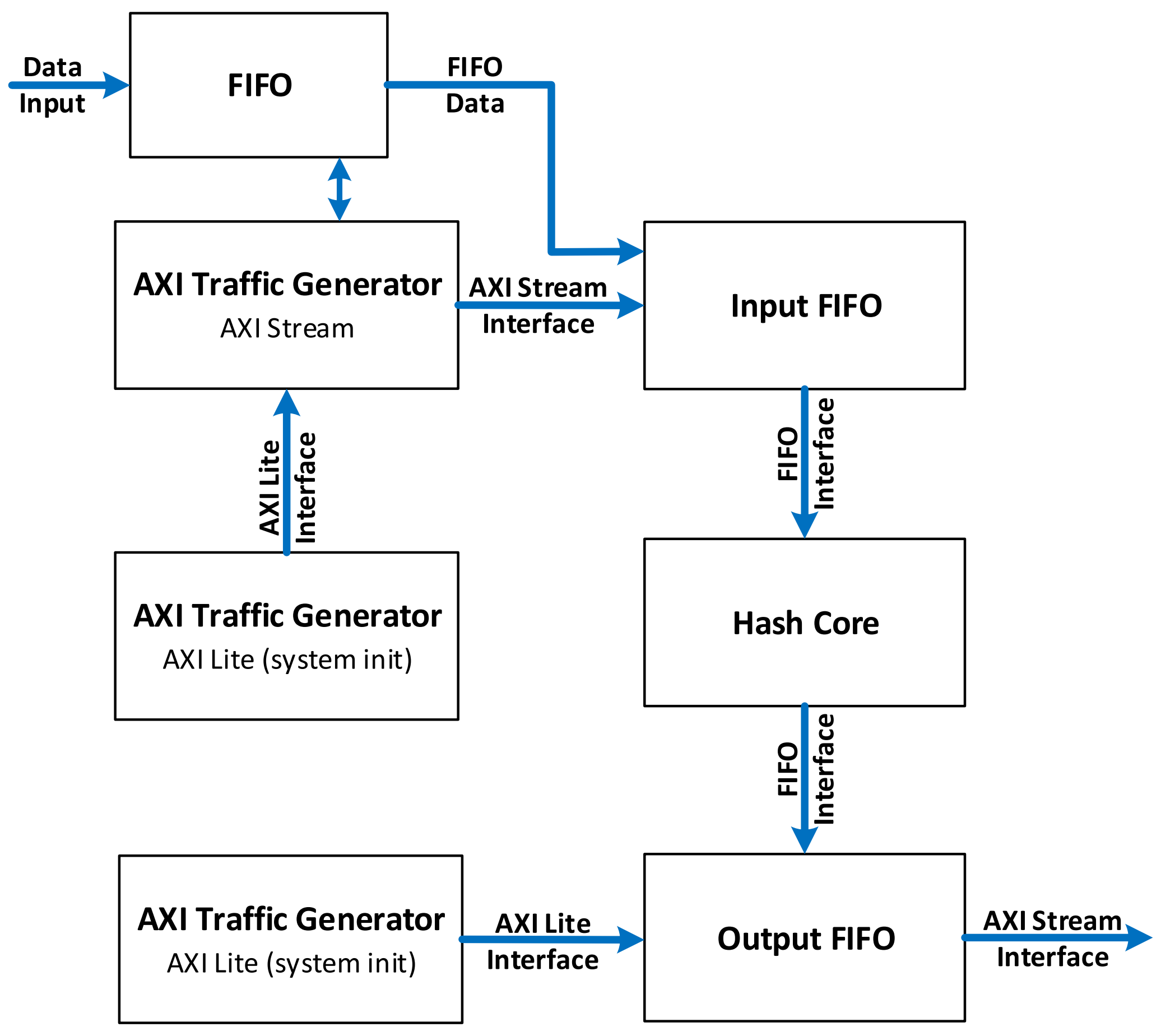
- **Custom IPs:**



VERIFICATION METHODOLOGY

- A universal testbench has been developed in the Vivado environment to verify the operation of our testbed using simulation.
- ATG (AXI Traffic Gen) IP has limitation in case of generating specific data in AXI stream mode through tdata port. As a result, we used a separate FIFO which is already filled with our desired data and AXI stream ATG only provides control signals.
- AXI Lite ATGs are used to configure Output FIFO and AXI Stream ATG.

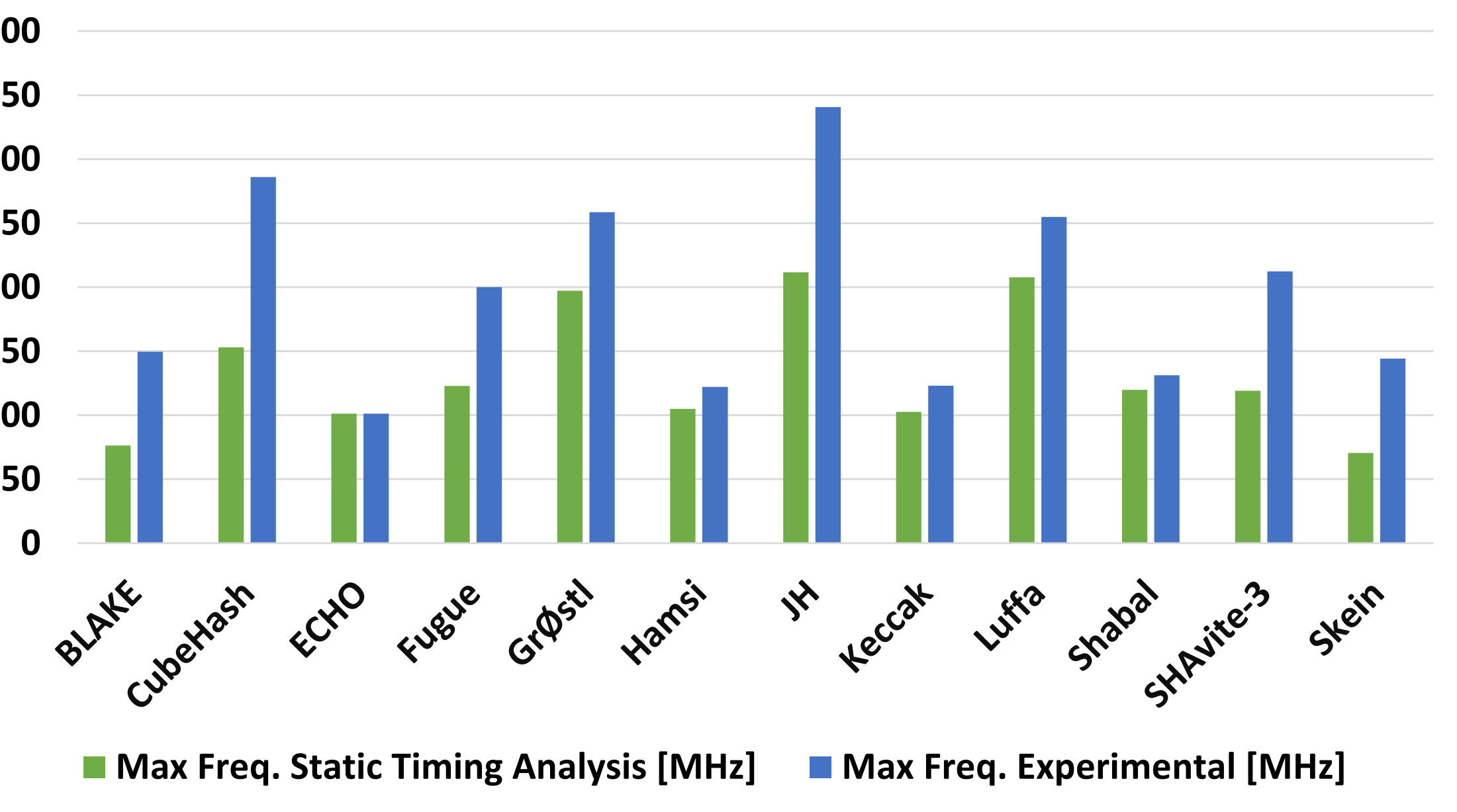
Universal testbench for Vivado environment



RESULTS: Maximum Frequency

- ZedBoard and Vivado 2015.4 have been used for result generation. All options of Vivado design suite including synthesis and implementation settings are set to default mode.
- On the software side, the bare metal environment and Xilinx SDK are used for running the C code on the ARM core of Zynq.

Maximum clock frequencies obtained using static timing analysis and the experimental measurement, respectively



Maximum frequencies and throughputs

Algorithm	Max Freq. Static Timing Analysis [MHz]	Max Freq. Experimental [MHz]	Throughput Based on Formula and Max Exp. Freq. [Gb/s]	Throughput Based on Exp. HW Exe. Time [Gb/s]
BLAKE	76.4	145.4	3.546	3.544
CubeHash	152.9	275.8	4.413	4.399
ECHO	100.1	101.1	5.999	6.000
Fugue	122.9	200.0	3.200	3.191
Grøstl	197.2	258.6	6.305	5.821
Hamsi	105.0	124.9	1.333	1.332
JH	211.6	333.3	4.740	4.726
Keccak	102.6	123.1	5.292	5.314
Luffa	152.5	247.4	7.037	7.213
Shabal	119.7	122.5	0.981	0.983
SHAvite-3	119.0	205.7	2.846	2.828
Skein	70.6	140.3	3.782	3.772

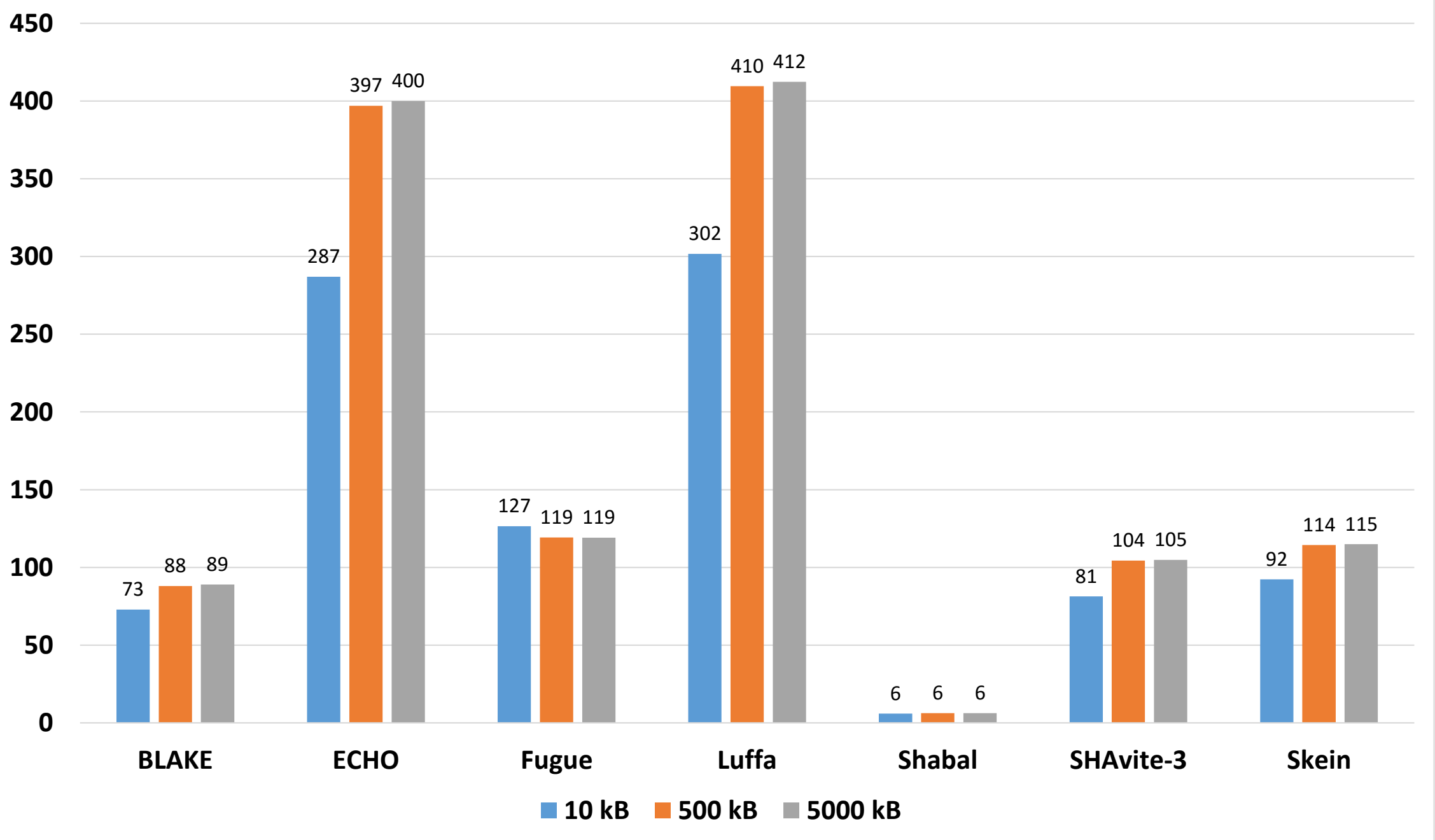
- Max Freq. Experimental was determined as a worst case value across all investigated input sizes from 10 to 5000 kB.
- Throughput Based on Exp. HW Exe. Time was obtained by dividing the message input size by the actual execution time of hashing in hardware, measured using AXI Timer for the input size equal to 1000 kB.

Formulas for the execution time and throughput
Notation: T - clock period in ns, N - number of input blocks

Algorithm	**I/O Bus width**	**Hash Time [cycles]**	**Throughput [Gbit/s]**
BLAKE	64	$2+8+21\cdot N+4$	$512/(21\cdot T)$
CubeHash	64	$2+4+16\cdot N+160+4$	$256/(16\cdot T)$
ECHO	64	$3+24+26\cdot N+1+4$	$1536/(26\cdot T)$
Fugue	32	$2+2\cdot N+37+8$	$32/(2\cdot T)$
Grøstl	64	$3+21\cdot N+4$	$512/(21\cdot T)$
Hamsi	32	$3+1+3\cdot(N-1)+6+8$	$32/(3\cdot T)$
JH	64	$3+8+36\cdot N+4$	$512/(36\cdot T)$
Keccak	64	$3+17+24\cdot N+4$	$1088/(24\cdot T)$
Luffa	64	$3+4+9\cdot N+9+1+4$	$256/(9\cdot T)$
Shabal	64	$2+64\cdot N+64\cdot 3+16$	$512/(64\cdot T)$
SHAvite-3	64	$3+8+37\cdot N+4$	$512/(37\cdot T)$
Skein	64	$2+8+19\cdot N+4$	$512/(19\cdot T)$

RESULTS: SpeedUp vs. Software

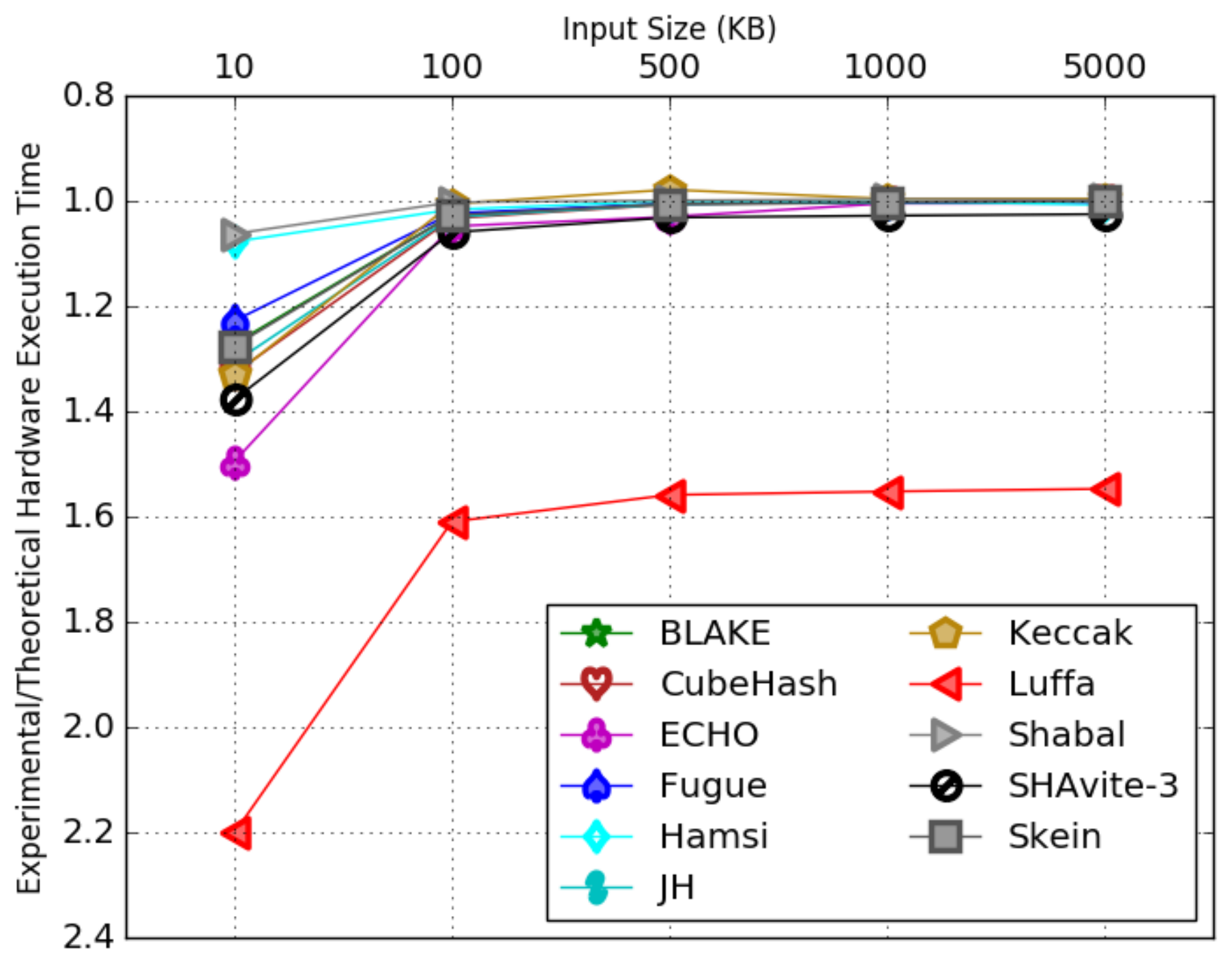
HW/SW speed up for three different input sizes in KB



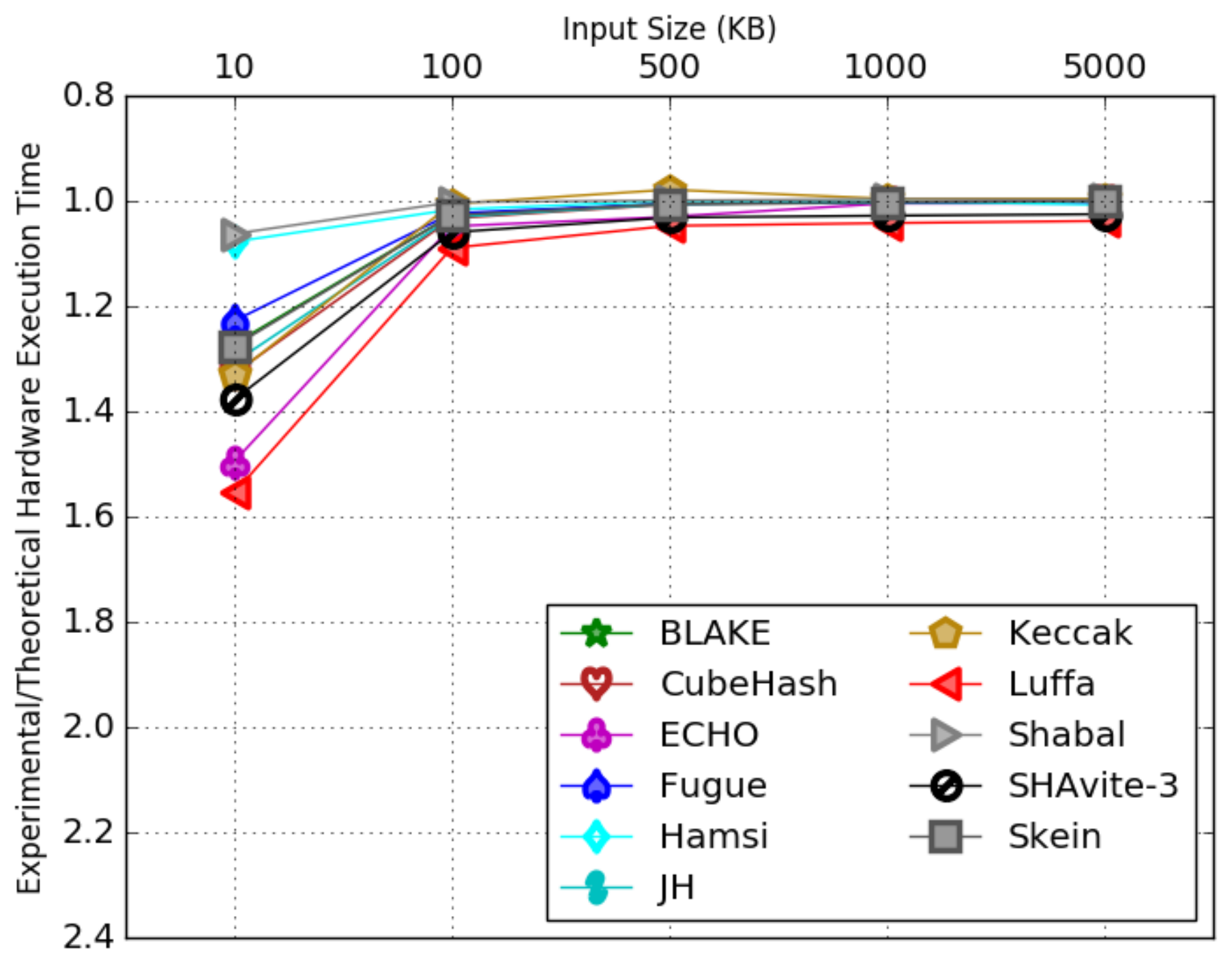
- Only algorithms with optimized software implementation and ARM architecture support are shown in this graph.

RESULTS: Data Transaction Overhead

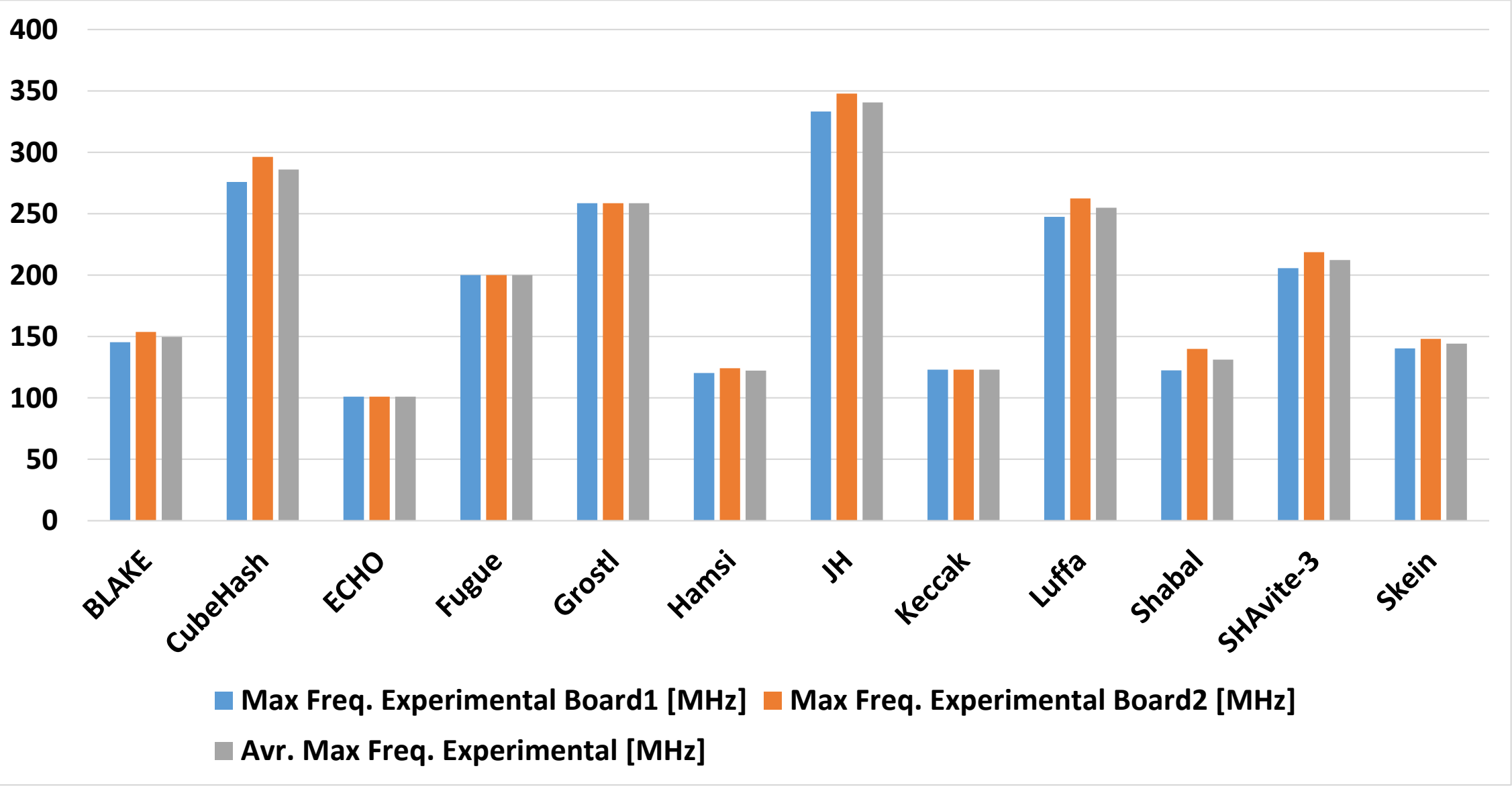
DMA core running at 100 MHz for all algorithms



DMA core running at 150 MHz in case of Luffa and 100 MHz for all other algorithms



RESULTS: Experiment on Two Different ZedBoards



CONCLUSIONS

- The testbed can be used to correctly measure performance of designs with the maximum throughput up to 64 bit · 150 MHz = 9.6 Gbit/s.
- For all the investigated hash functions, the overhead of the communication between PS and PL was below 5% for 100 kB messages and negligible for messages above 500 kB.
- All algorithms have also demonstrated significant speed up vs. their execution in software on the same chip, in spite of the substantial speed of the ARM core, operating at 667 MHz.
- Our experiments have also demonstrated that the maximum experimental clock frequency was always higher than the post-place and route frequency calculated by Vivado using static timing analysis.
- At the same time, somewhat unexpectedly, the spread of ratios experimental to post-place and route frequency is very large, ranging from 1 to 2. This fact can be explained by a different influence of parameter variations and operating conditions on the critical path of each hash core, due to a different physical location (placement) of these critical paths in the FPGA fabric

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1314540.