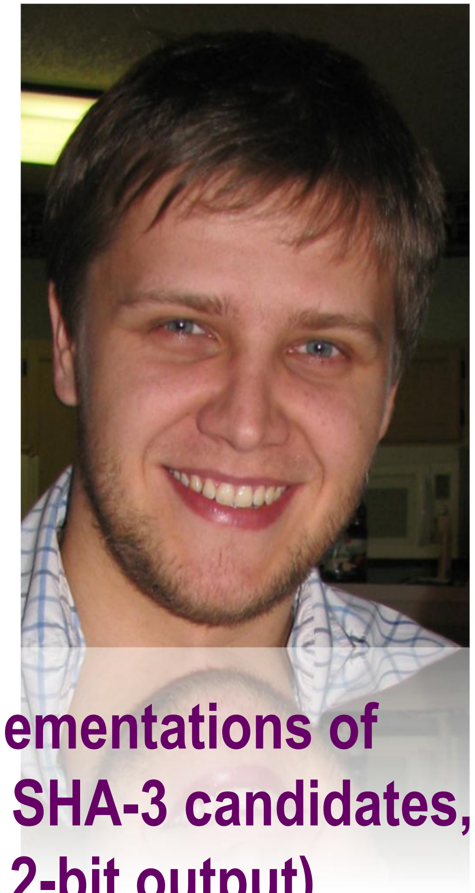
Investigating Design Space of Five Final SHA-3 Candidates in High-Performance FPGAs

<u>Kris Gaj,</u>
Ekawat Homsirikamol,
and Marcin Rogawski
George Mason University
U.S.A.

Co-Authors

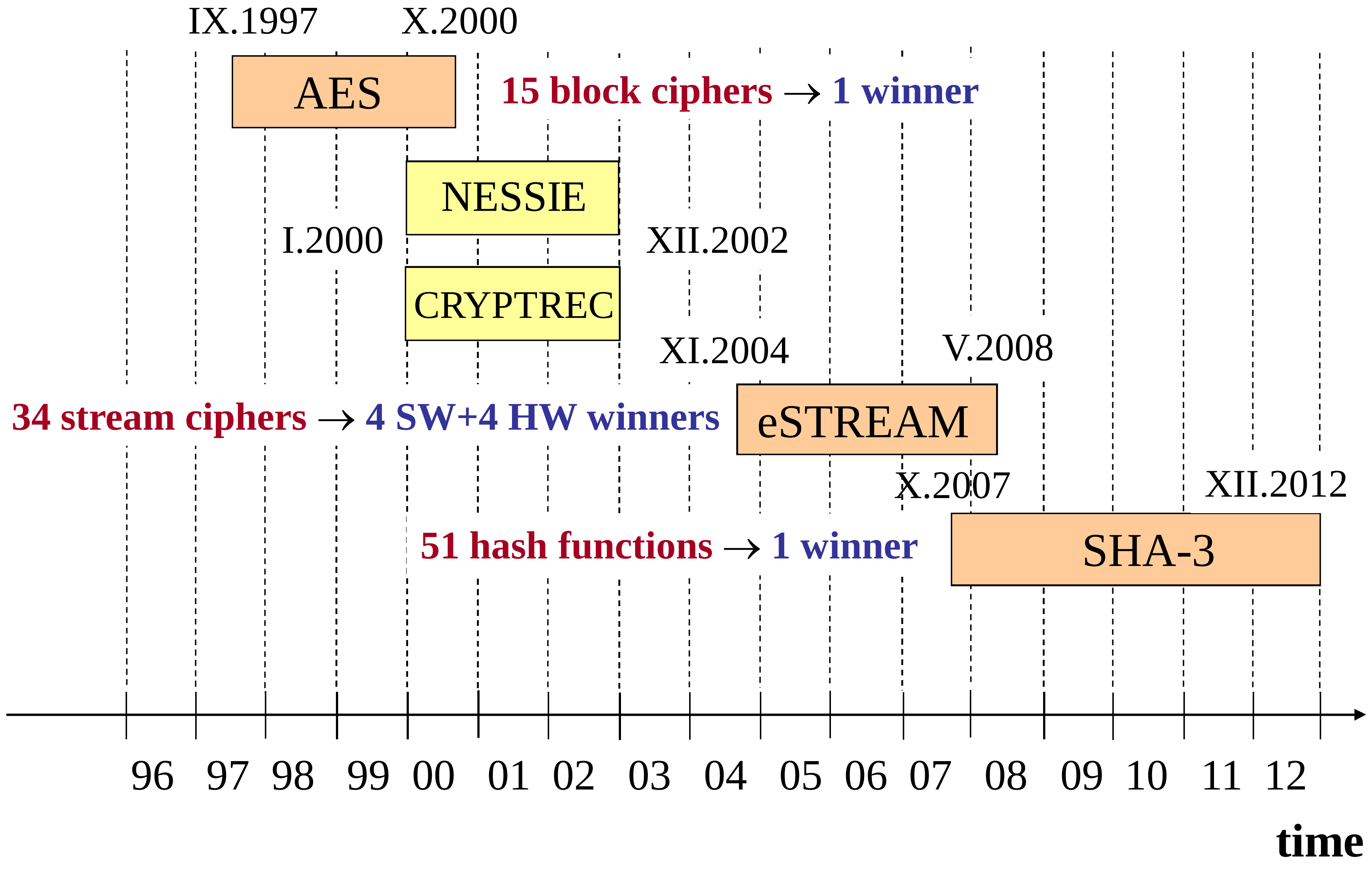
Ekawat Homsirikamol
a.k.a ñIceò

Marcin Rogawski



Developed optimized VHDL implementations of
14 Round 2 SHA-3 candidates, 5 Round 3 SHA-3 candidates, and SHA-2
in two variants each (256 & 512-bit output),
using several alternative architectures per each variant

Cryptographic Standard Contests



SHA-3 Contest – Recent and Future Milestones

23 Aug 2010 – Second SHA-3 Candidate Conference, Santa Barbara, USA

9 Dec 2010 – Announcement of 5 algorithms qualified to Round 3

31 Jan 2011 – Acceptance of final tweaks for Round 3 Candidates

16 Feb 2011 – Publication of Round 2 report

22 Mar 2012 – Third SHA-3 Candidate Conference, Washington D.C.
or Gaithersburg, MD, USA

Summer 2012 – Announcement of a winner

Beginning of 2013 – Publication of the new FIPS standard

NIST Evaluation Criteria

Security

Software Efficiency

Hardware Efficiency

ASICs

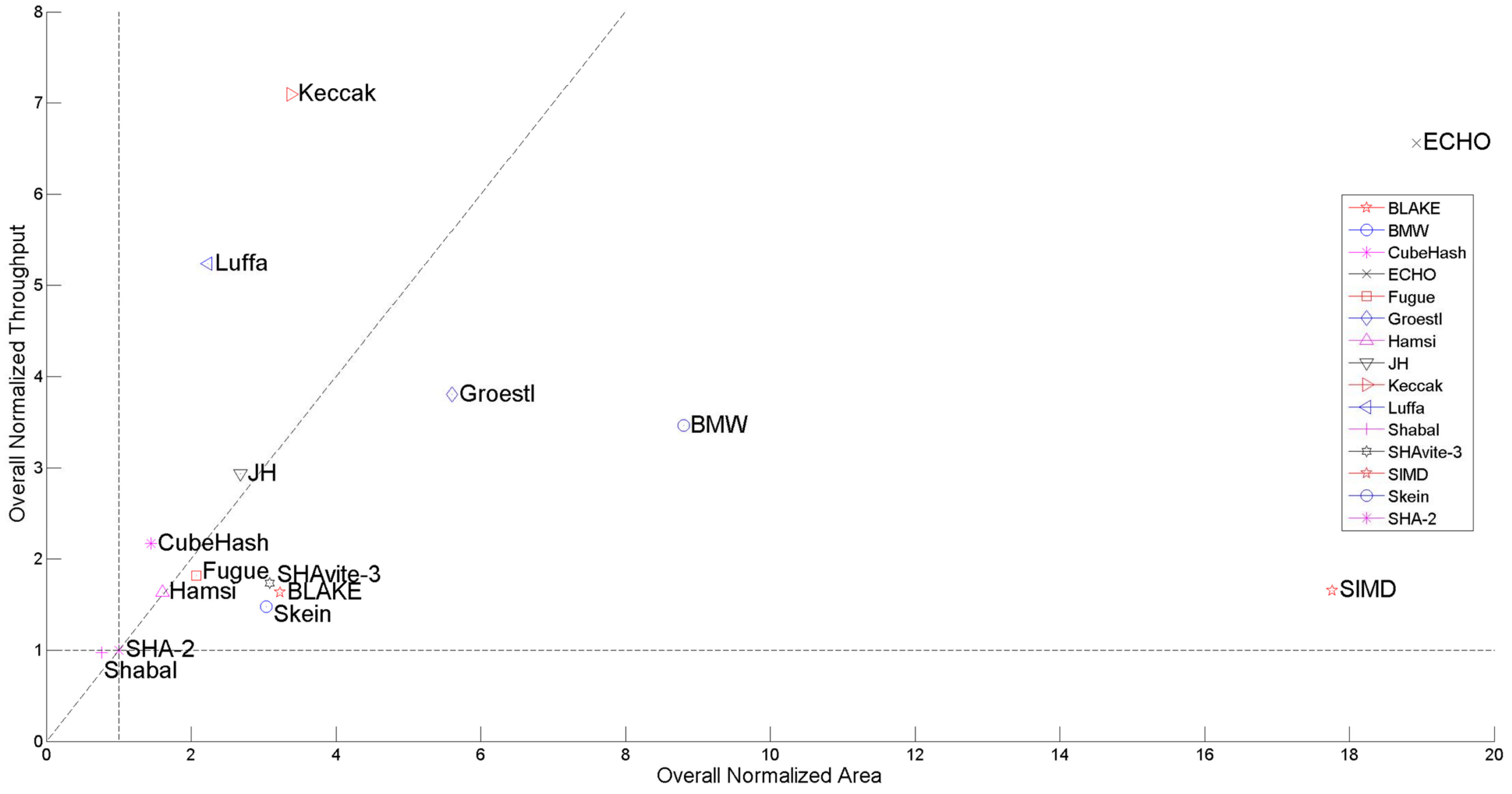
FPGAs

Flexibility

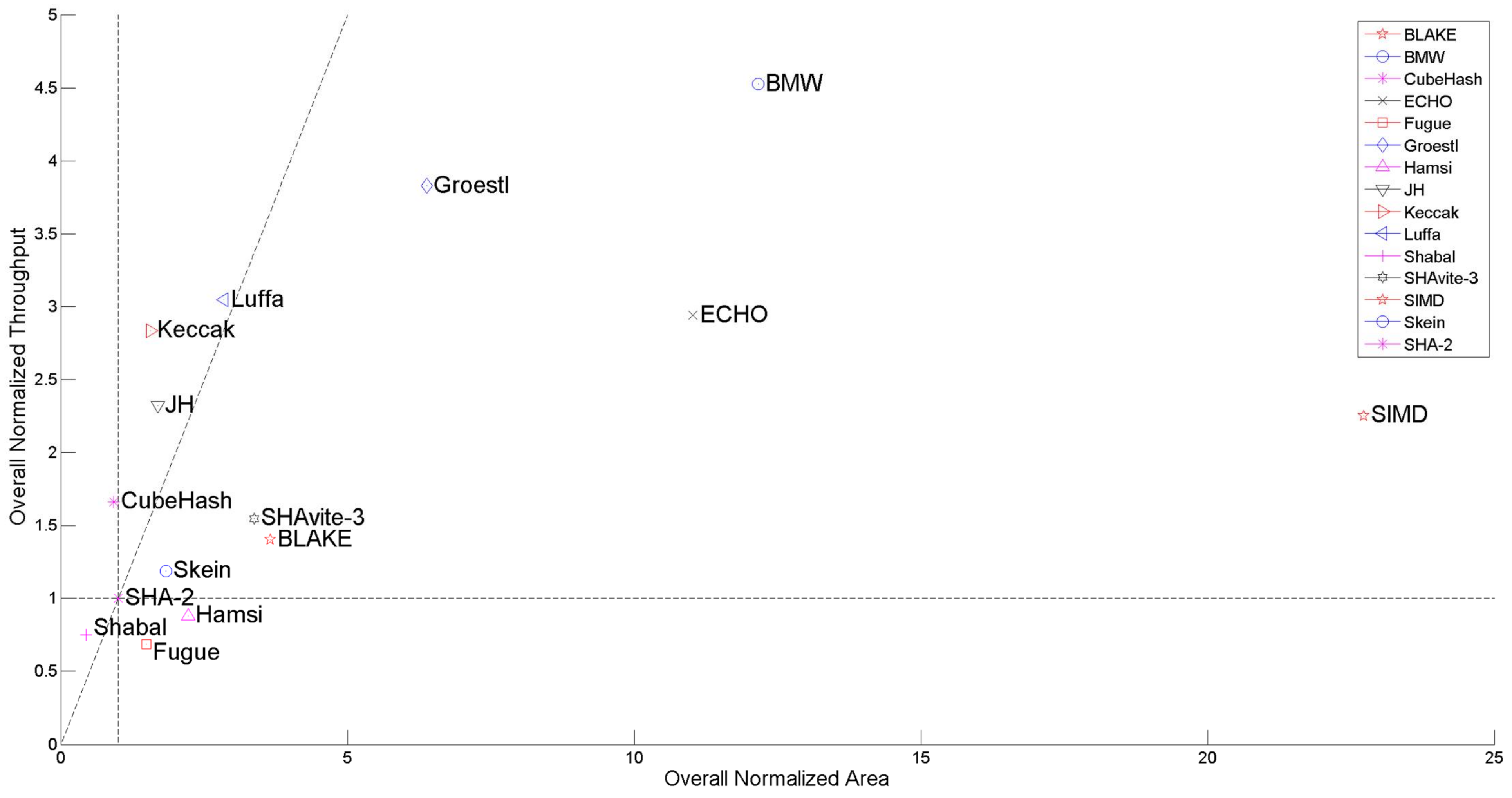
Simplicity

Licensing

Throughput vs. Area Normalized to Results for SHA-256 and Averaged over 11 FPGA Families – 256-bit variants



Throughput vs. Area Normalized to Results for SHA-512 and Averaged over 11 FPGA Families – 512-bit variants



SHA-3 Contest Finalists

Round 3 Evaluations

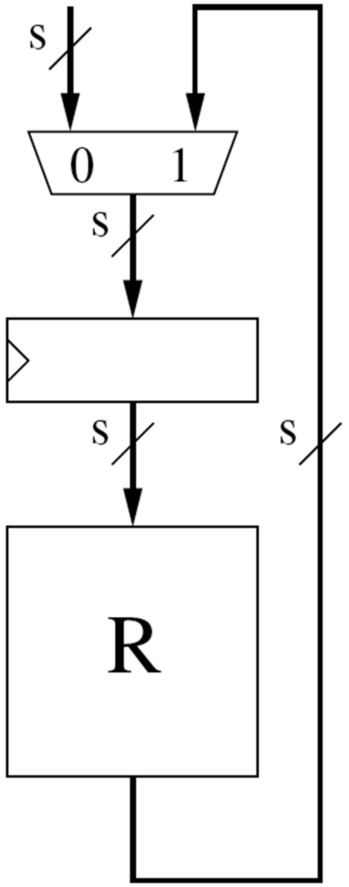
Our Goals for Round 3 of the SHA-3 Competition

- Analysis of multiple high-speed and medium-speed hardware architectures per each finalist, based on the known design techniques, such as
  - Folding
  - Unrolling
  - Pipelining

- Identifying the best architecture in terms of the throughput to area ratio (w/o using embedded resources)

- Analyzing the flexibility of all algorithms in terms of speed vs. area trade-offs

Starting Point: Basic Iterative Architecture



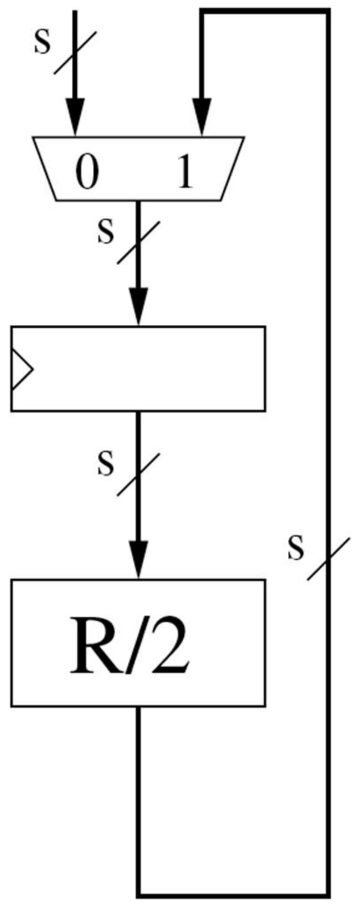
- datapath width = state size
- one clock cycle per one round/step

Block processing time = #R · T

#R = number of rounds/steps
T = clock period

Currently, most common architecture used to implement SHA-1, SHA-2, and many other hash functions.

Horizontal Folding - /2(h)



- datapath width = state size
- two clock cycles per one round/step

Block processing time = (2·#R) * T'

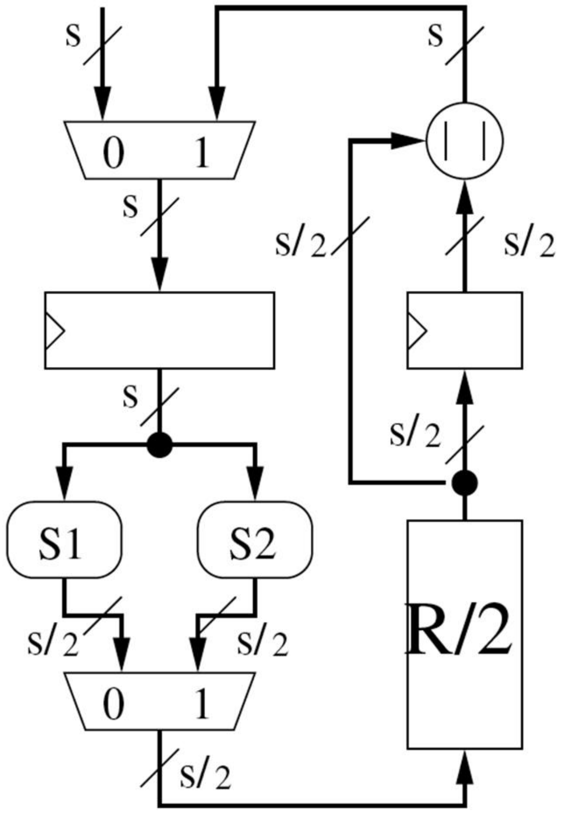
$T/2 < T' < T$

typically $T' \approx T/2$

Area/2 < Area' < Area

Typically Throughput/Area ratio increases

Vertical Folding - /2(v)

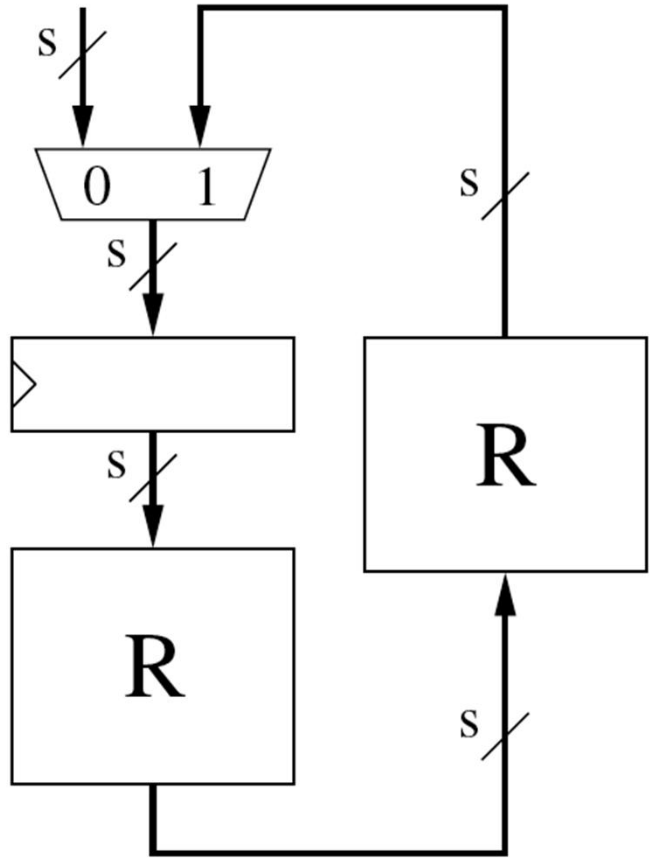


- datapath width = state size/2
- two clock cycles per one round/step

Block processing time = (2·#R) * T’

typically T’ ≈ T
Area/2 < Area' < Area

Unrolling - x2



- datapath width = state size
- one clock cycle per two rounds

Block processing time = (#R/2) * T'

$T < T' < 2 \cdot T$

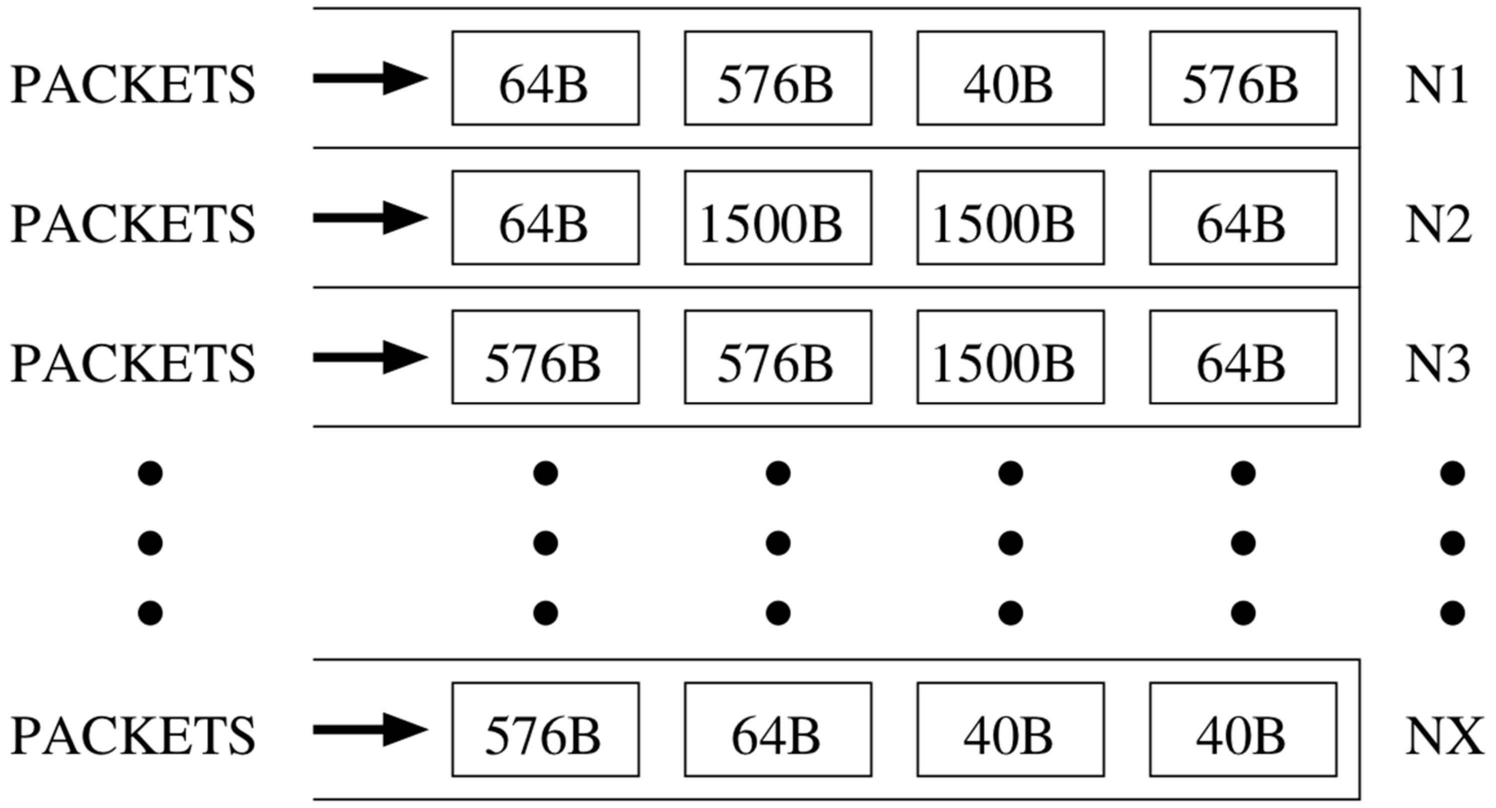
typically $T' \approx 2 \cdot T$

$Area/2 < Area' < 2 \cdot Area$

Typically $Area' \approx 2 \cdot Area$

Typically Throughput/Area ratio decreases

Multiple Packets Available for Parallel Processing

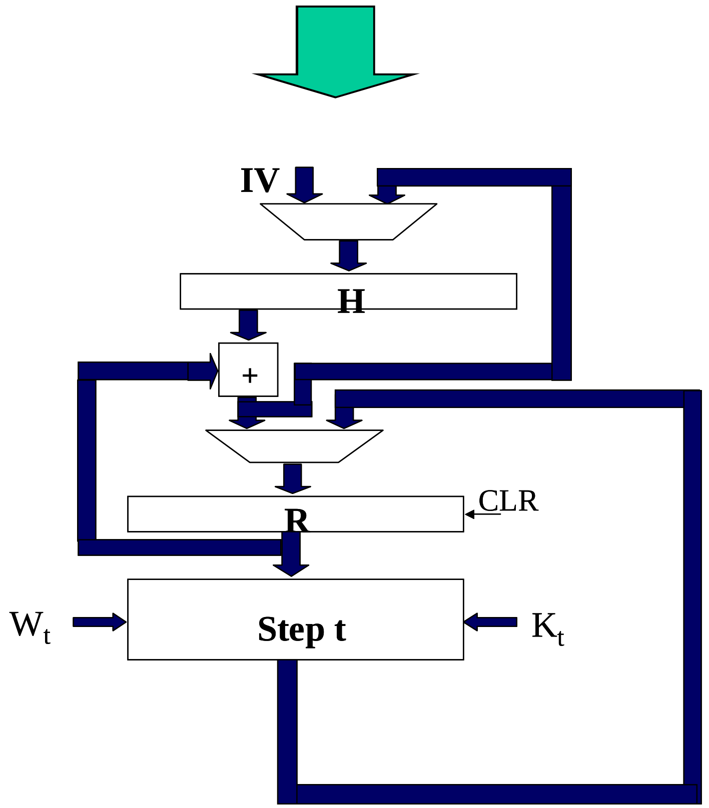


Typical sizes of packets: 40B – 1500B

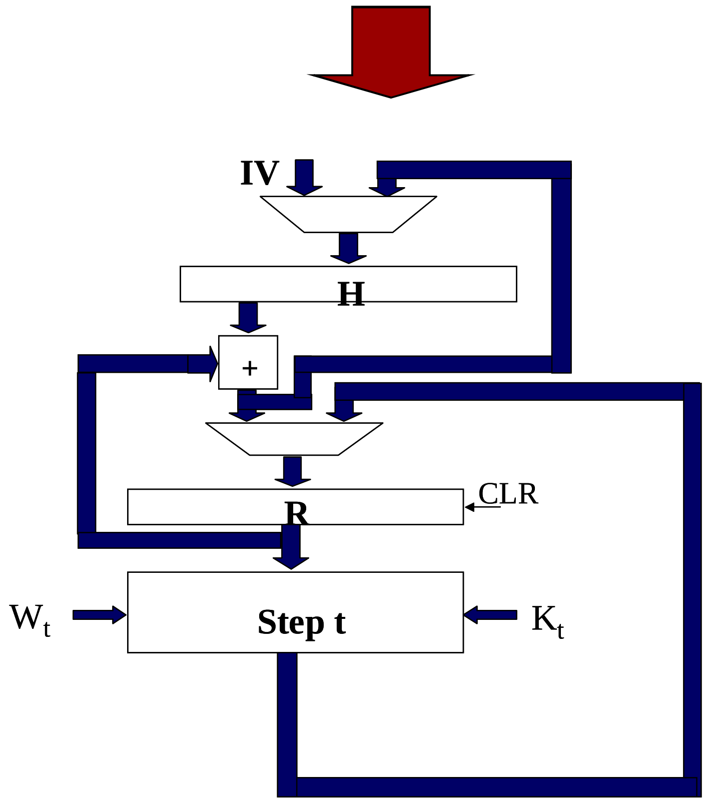
1500 B = Maximum Transmission Unit (MTU) for Ethernet v2

Parallel Processing Using Multi-Unit Architecture – MU2

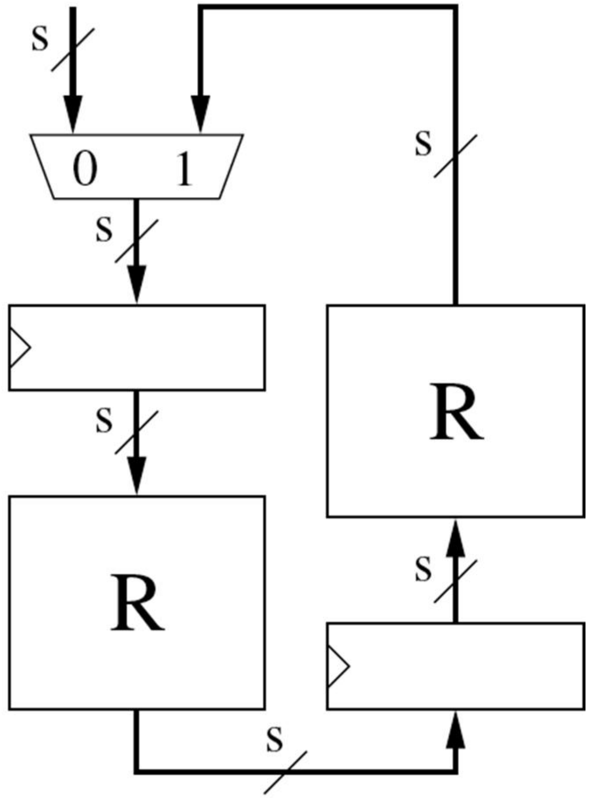
Packet 1

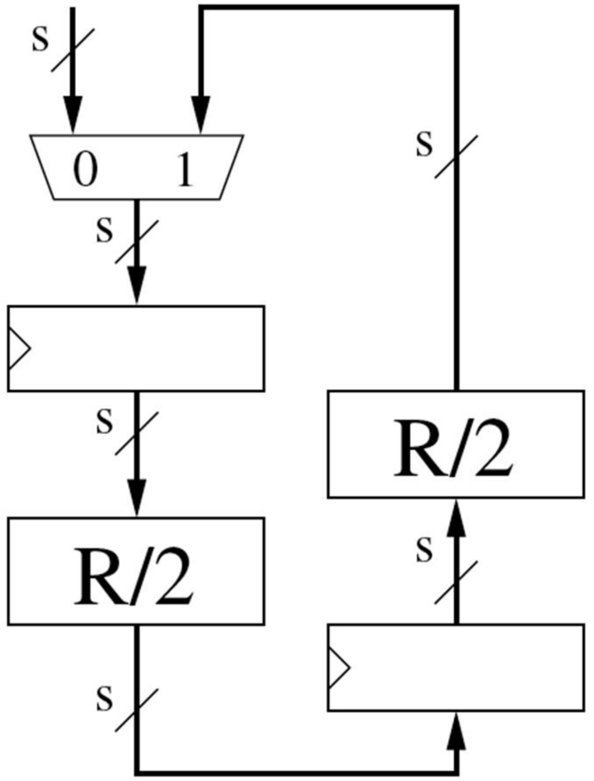


Packet 2



Pipelining - x2-PPL2, x1-PPL2





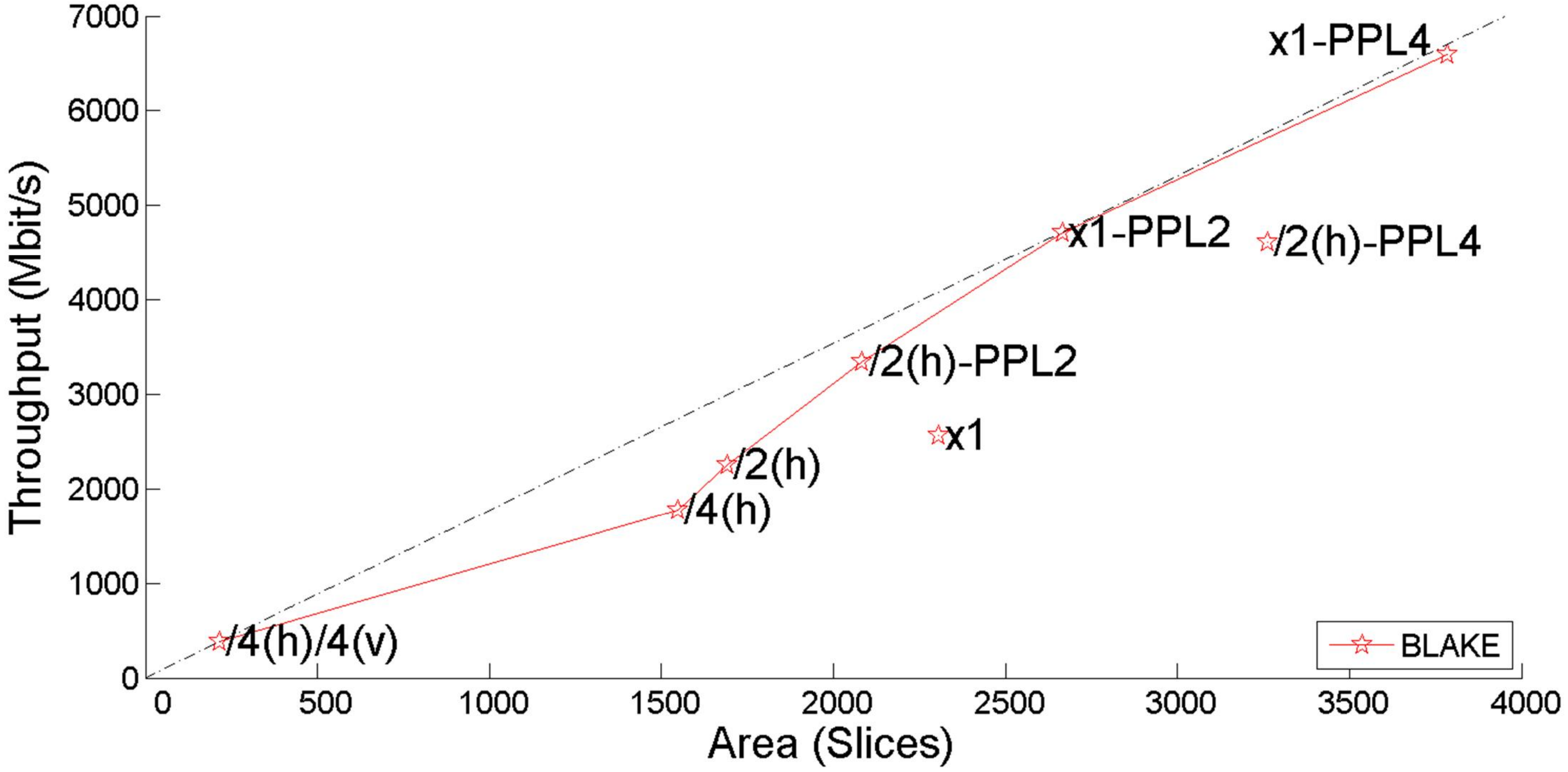
Comprehensive Evaluation

Åtwo major vendors: Altera and Xilinx (~90% of the market)

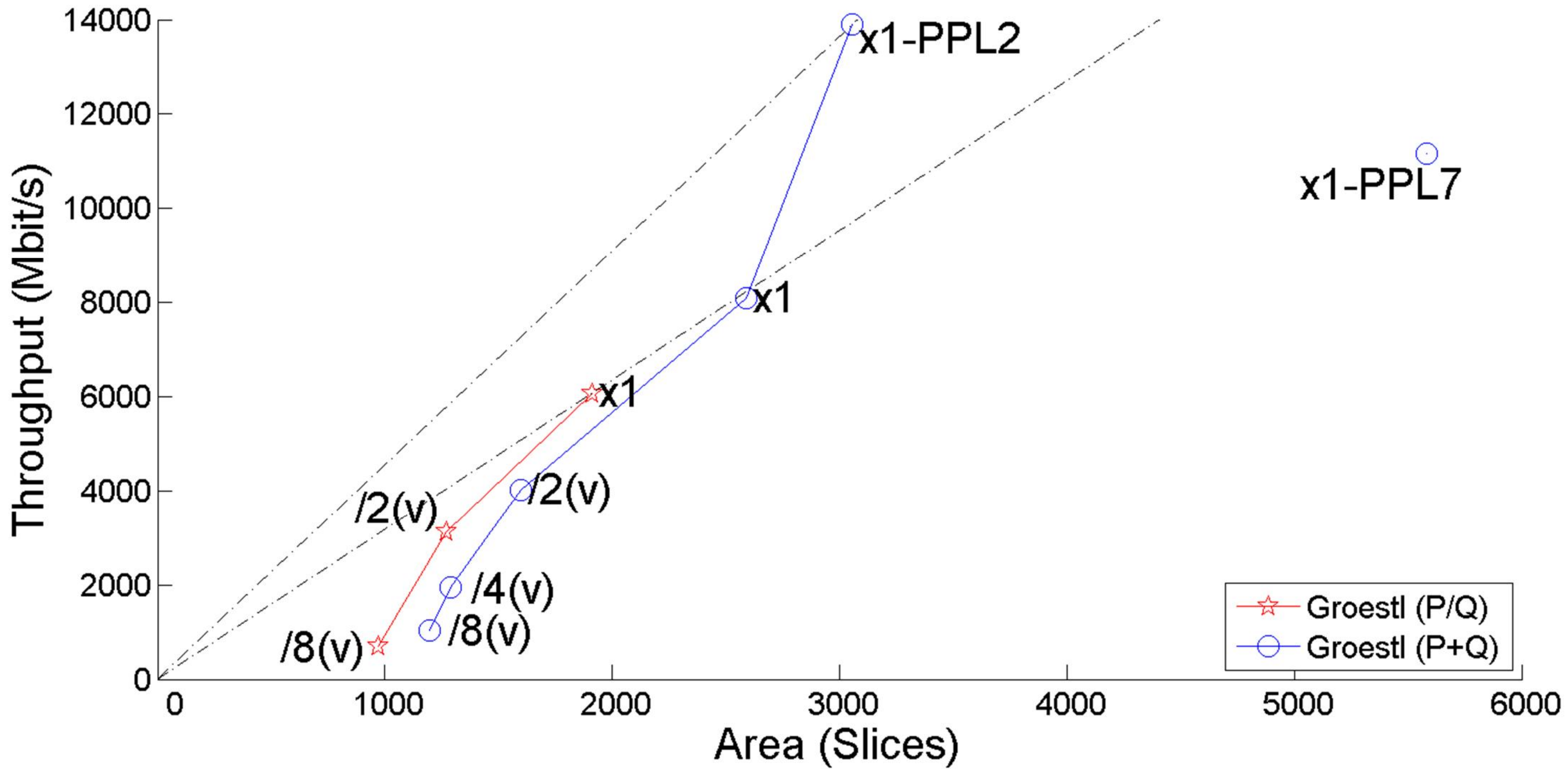
Åtwo most recent high-performance families

	Altera		Xilinx	
Technology	Low-cost	High-performance	Low-cost	High-performance
90 nm	Cyclone II	Stratix II	Spartan 3	Virtex 4
65 nm	Cyclone III	Stratix III		Virtex 5
40-60 nm	Cyclone IV	Stratix IV	Spartan 6	Virtex 6

BLAKE-256 in Virtex 5



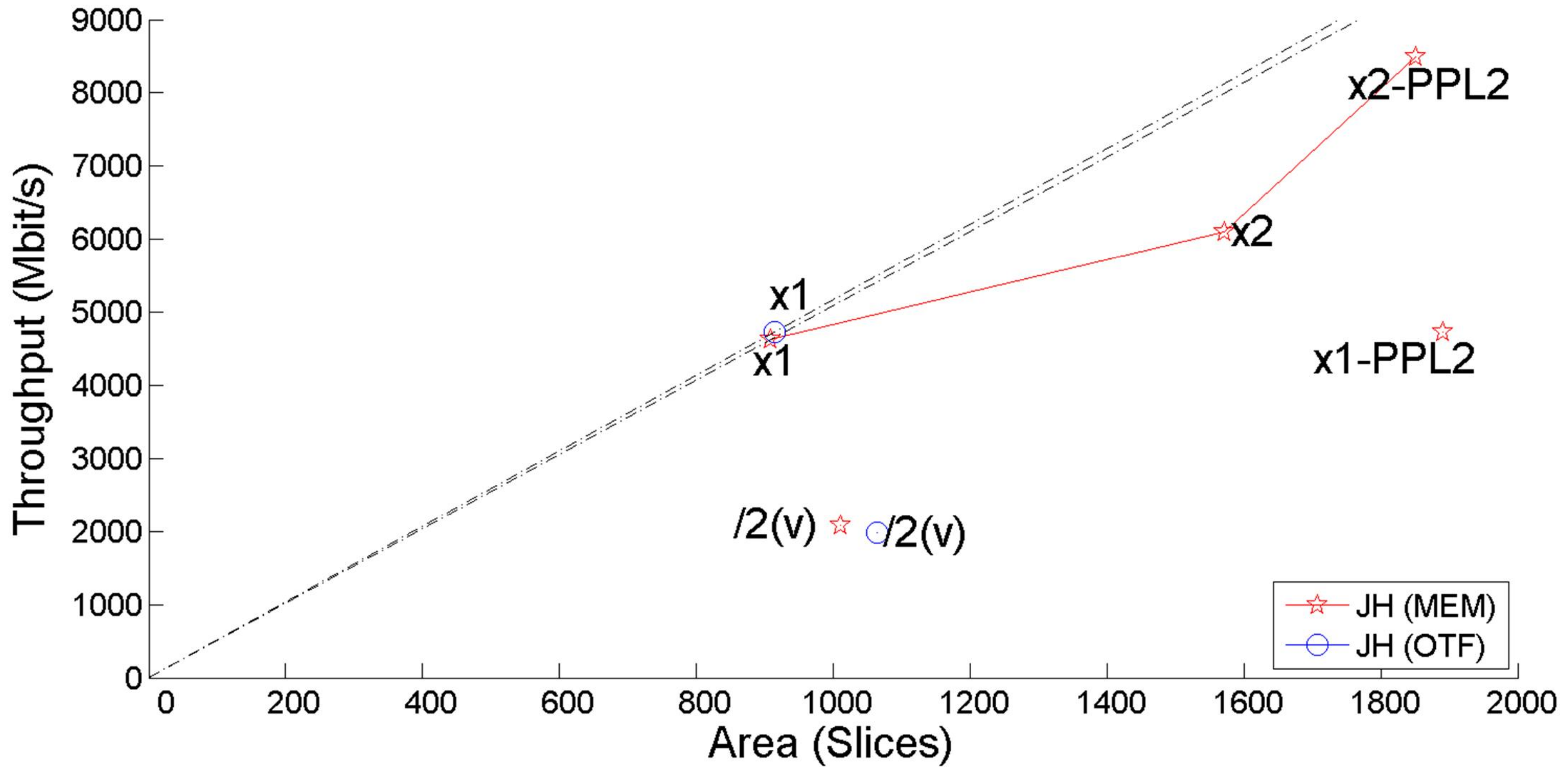
Groestl-256 in Virtex 5



Groestl P/Q – quasi-pipelined architecture; one unit shared between P and Q

Groestl P+Q – parallel architecture; two independent units for P and Q

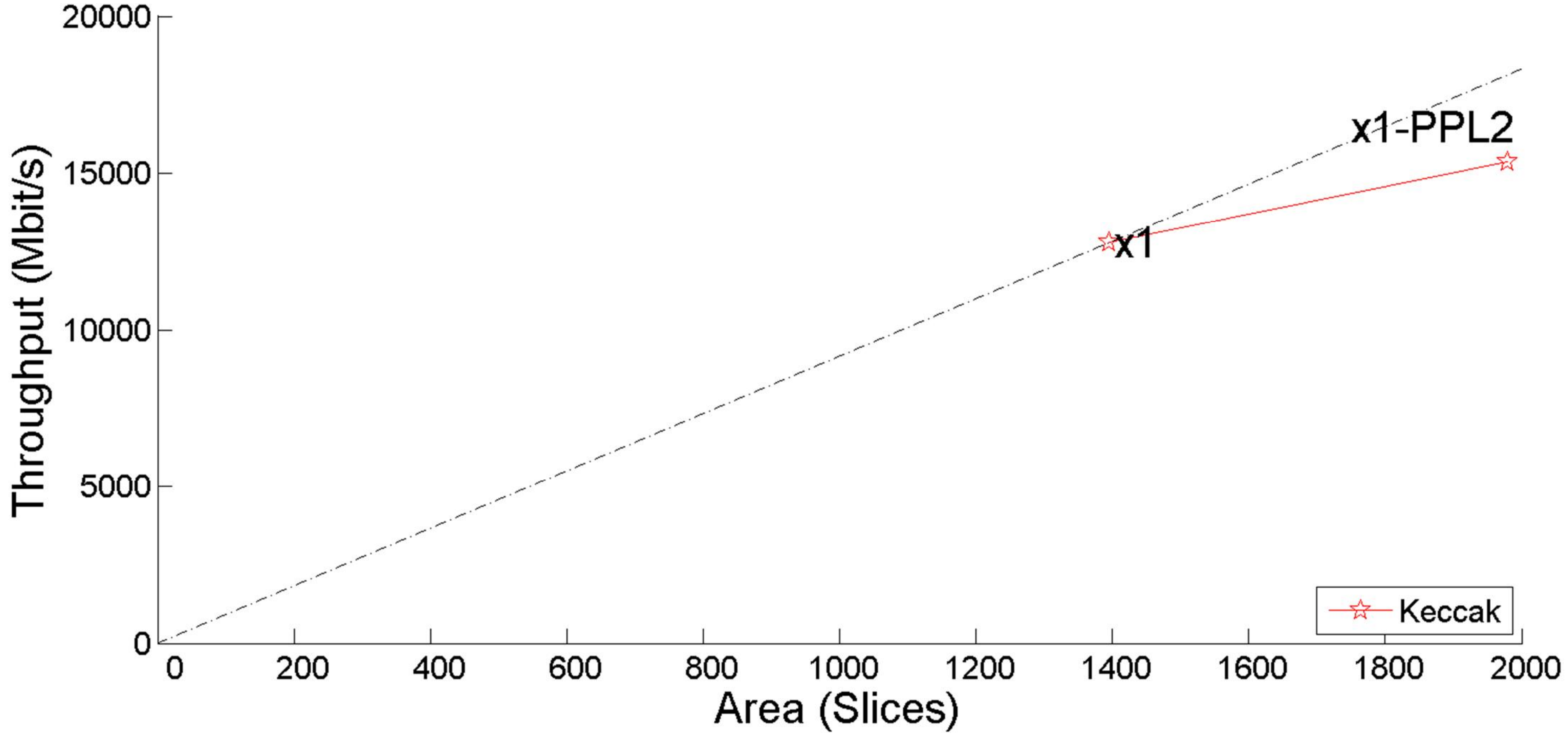
JH-256 in Virtex 5



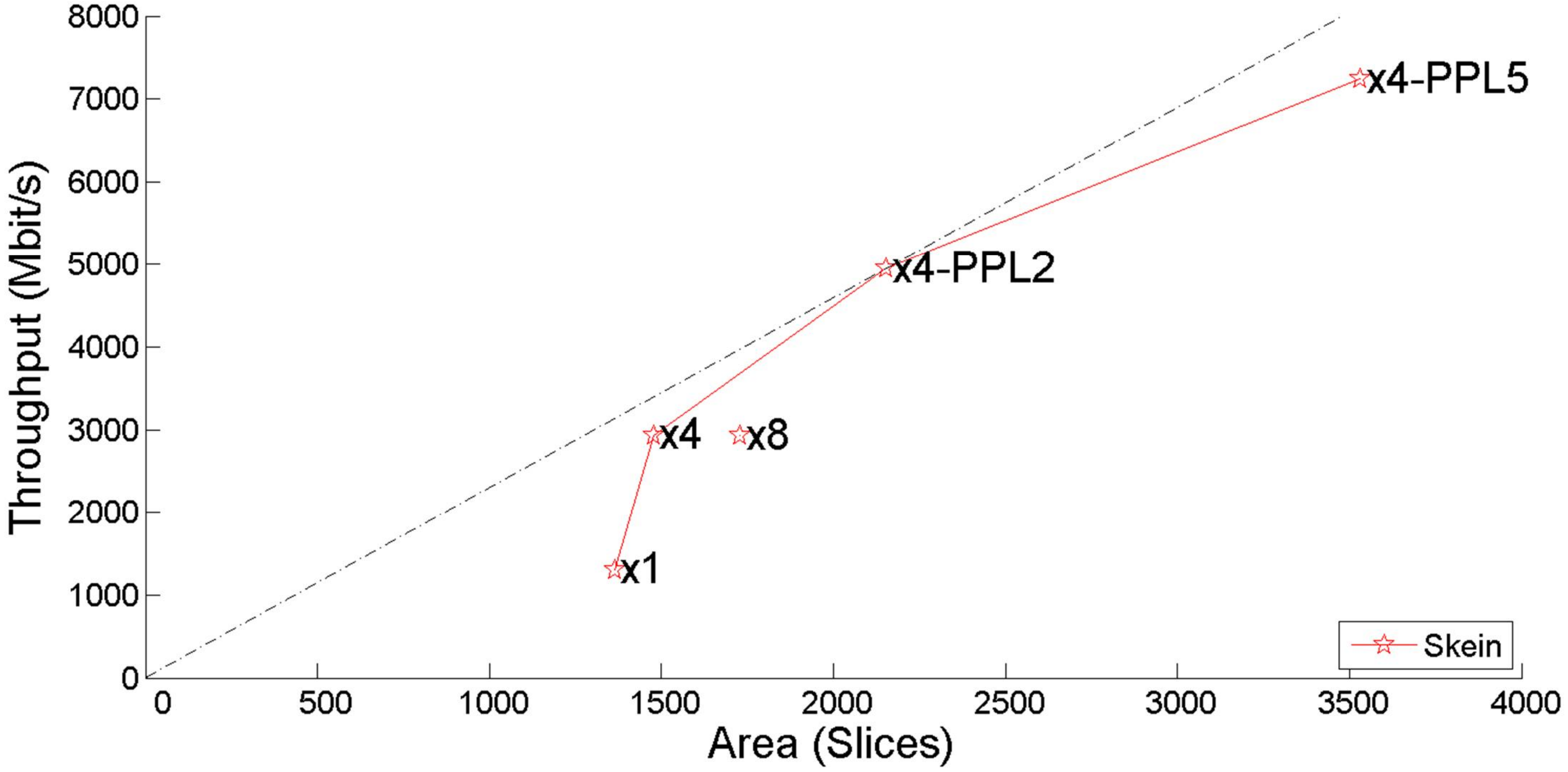
JH MEM – round constants stored in memory

JH OTF – round constants computed on-the-fly

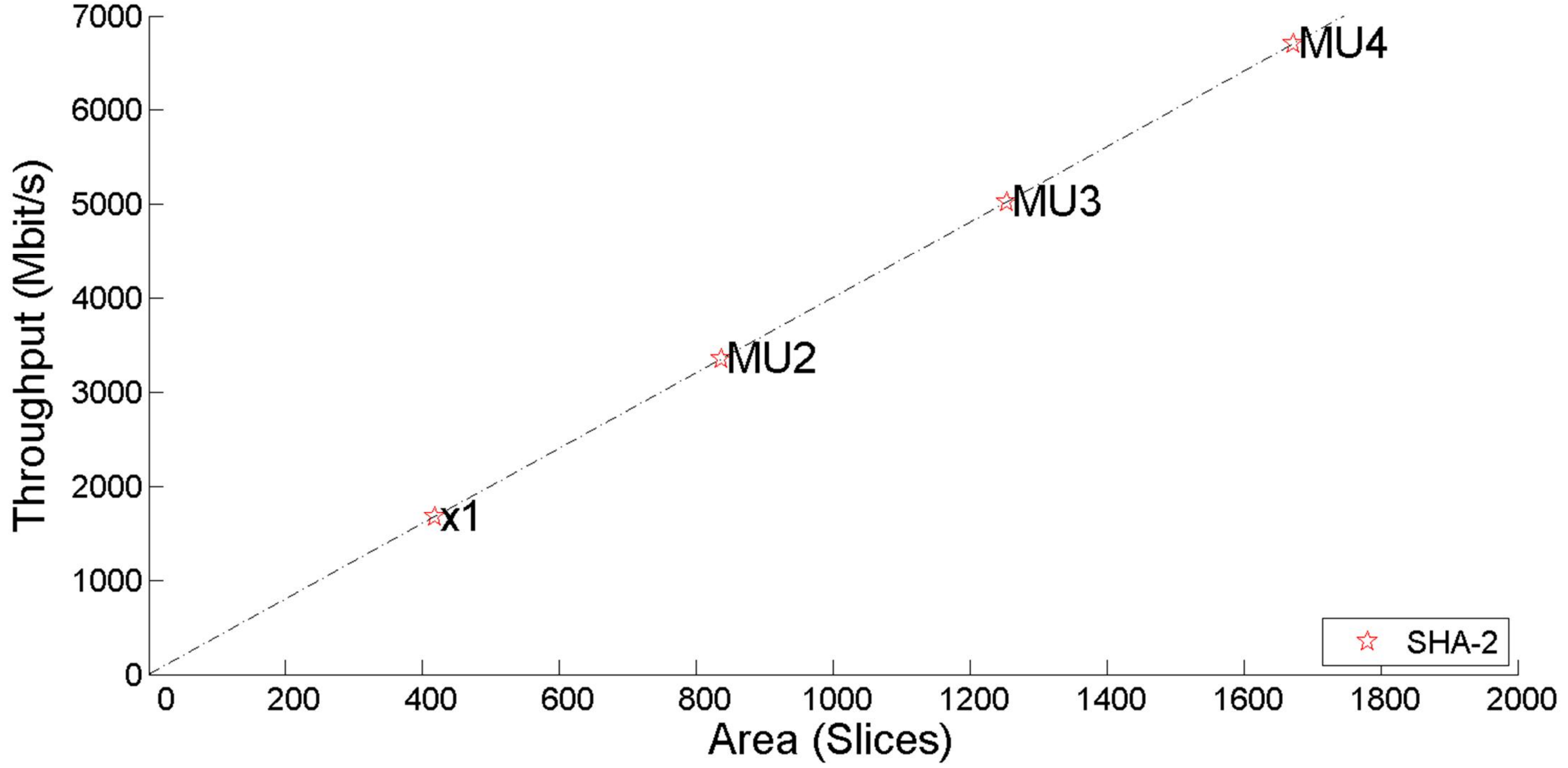
Keccak-256 in Virtex 5



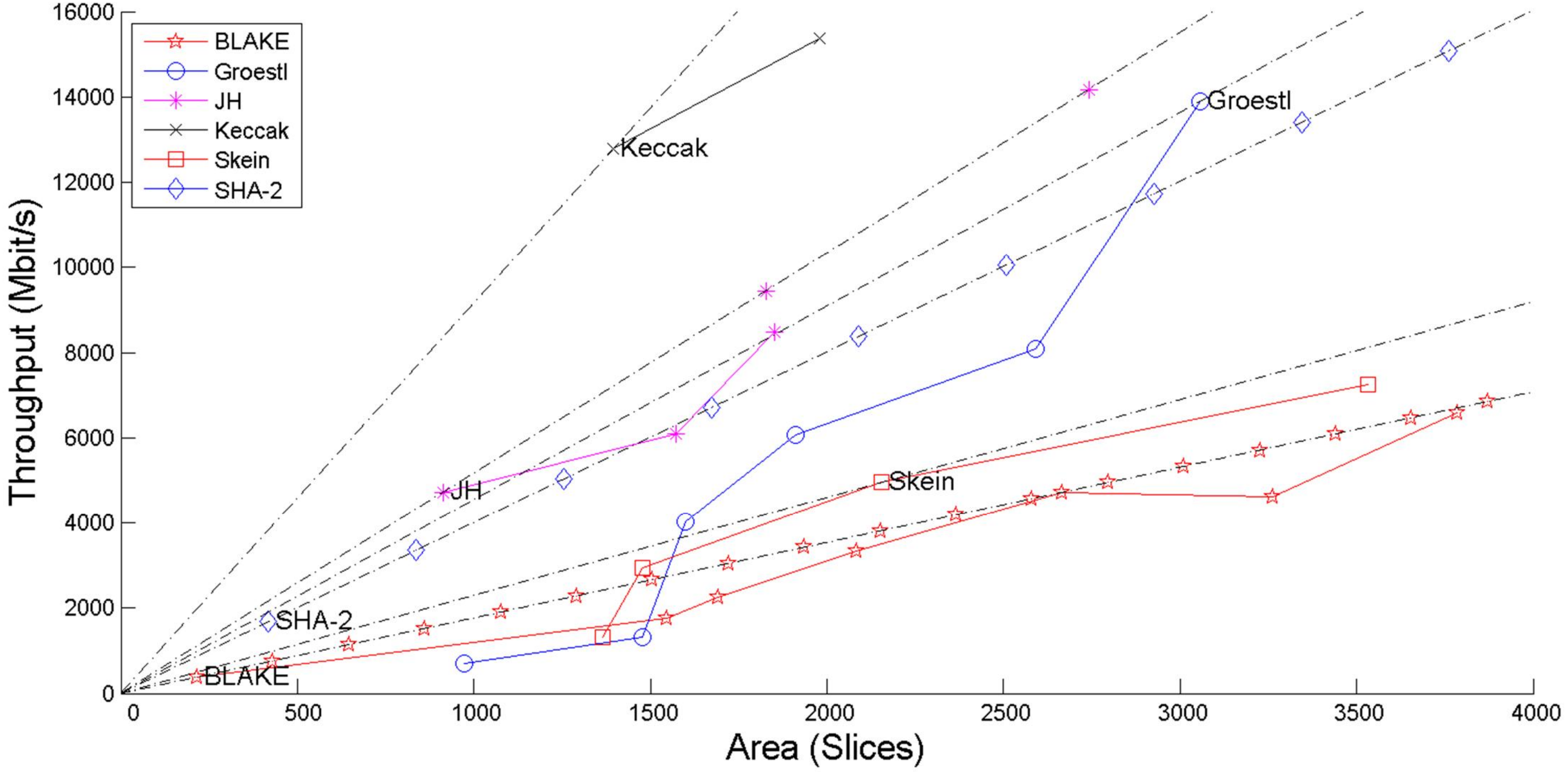
Skein-256 in Virtex 5



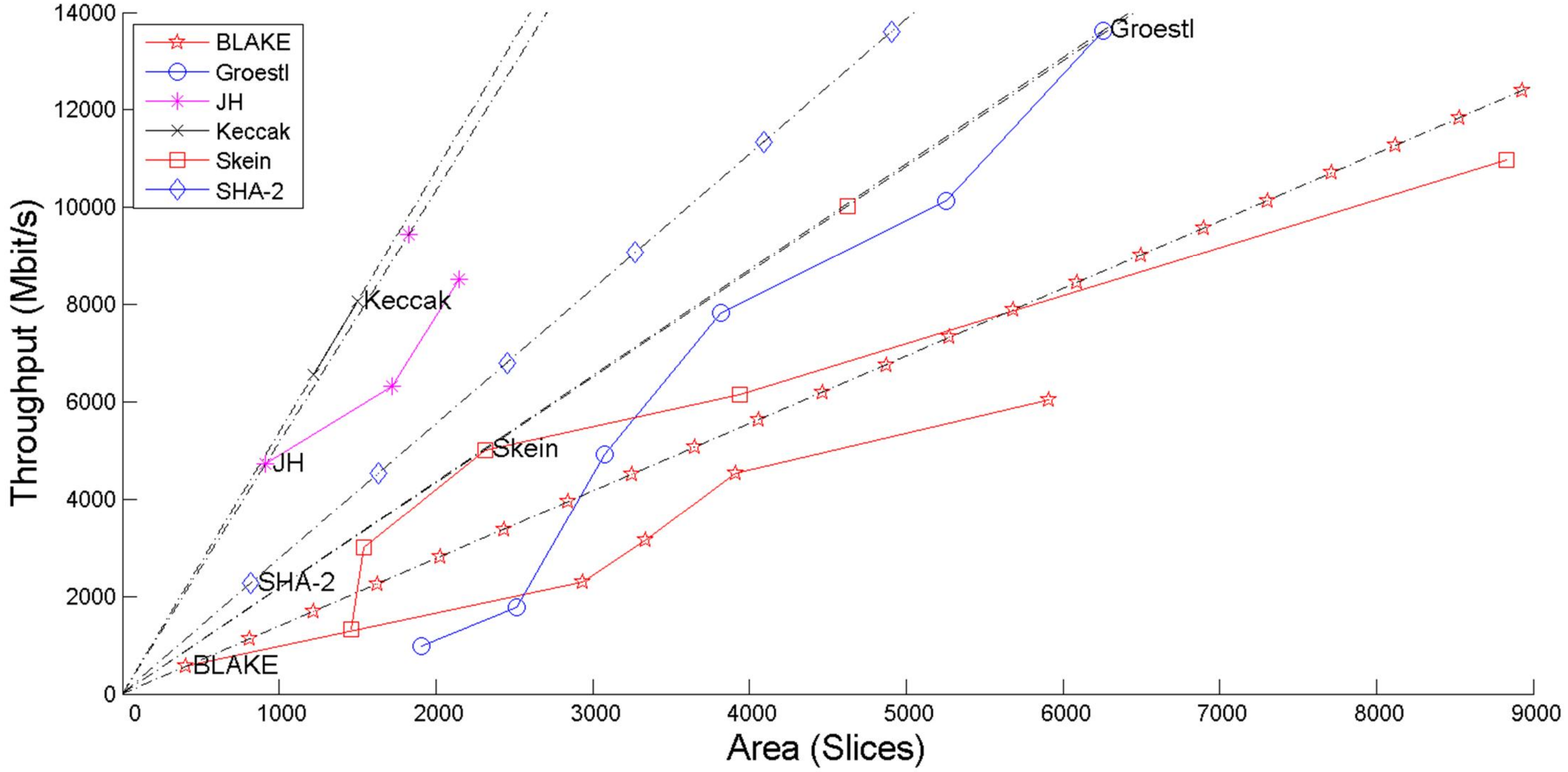
SHA-256 in Virtex 5



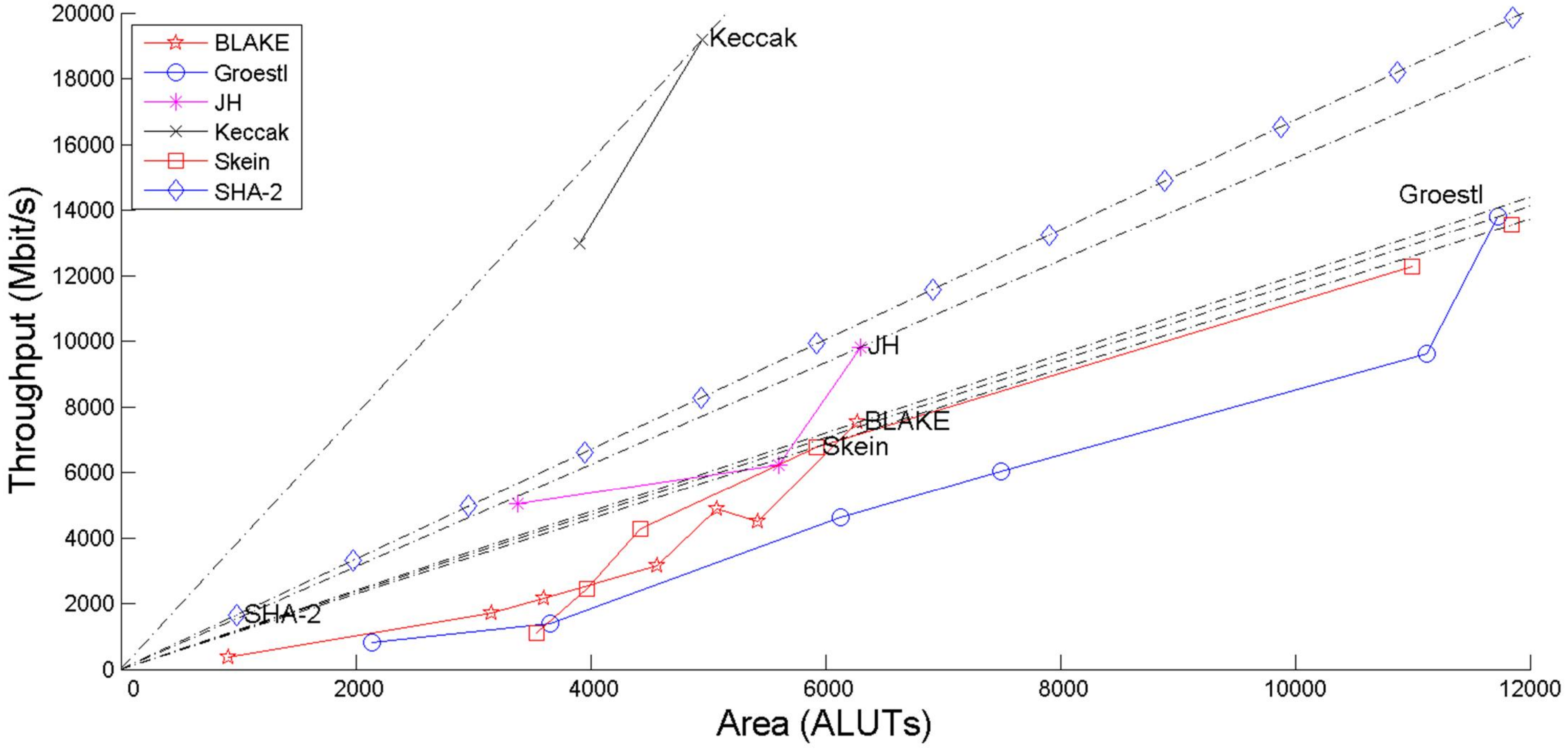
256-bit variants in Virtex 5



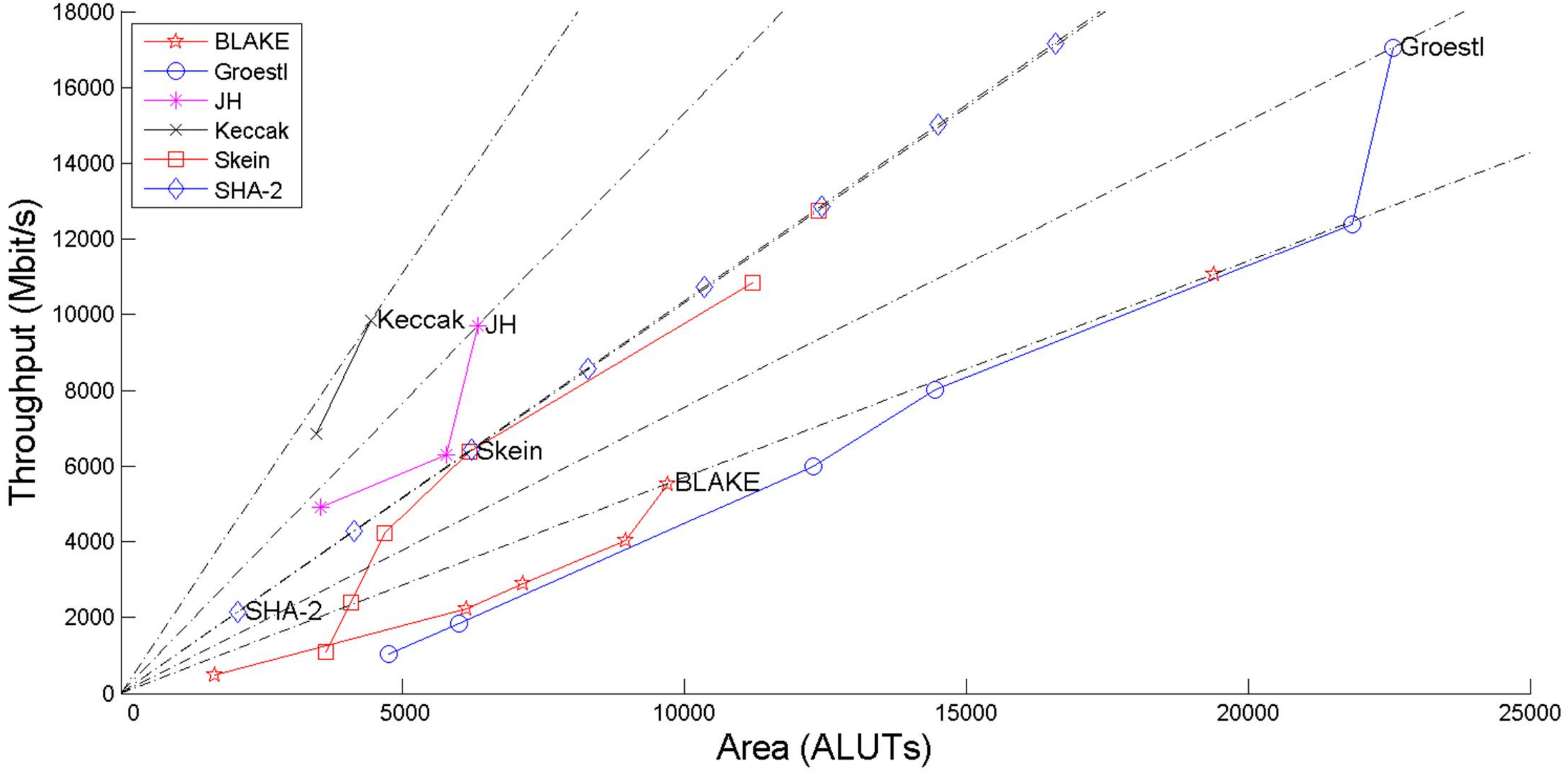
512-bit variants in Virtex 5



256-bit variants in Stratix III



512-bit variants in Stratix III



Summary

Keccak – consistently outperforms SHA-2, front runner
for high-speed implementations, but not suitable for folding

JH – performs better than SHA-2 most of the time,
not suitable for folding or inner-round pipelining

Groestl – better than SHA-2 for only one out of four FPGA families,
and only with relatively large area; suitable for vertical folding

Skein – the only candidate benefiting from unrolling;
easy to pipeline after unrolling

BLAKE – most flexible; can be folded horizontally and vertically,
can be effectively pipelined, however relatively slow
compared to other candidates.

Conclusions

- Using multiple architectures provides a more comprehensive view of the algorithms
- Algorithms differ substantially in terms of their flexibility and suitability for folding, unrolling, and pipelining
- Pipelined architectures the best in terms of the throughput to area ratio for 4 out of 5 candidates
- Two front-runners: Keccak, JH

Related Updates

GMU Source Codes

- First batch of GMU Source Codes made available at the ATHENa website at:

  http://cryprography.gmu.edu/athena

- Included in this release:
  - best non-pipelined architectures for each of the 14 Round 2 candidates and SHA-2
  - best non-pipelined architectures for each of the 5 Round 3 candidates
  - Each code supports two variants: with 256-bit and 512-bit output.

GMU Database of Results

- Available in the ATHENa database at

  http://cryptography.gmu.edu/athenadb

20 functions (14 Round 2 SHA-3 + 5 Round 3 SHA-3 + SHA-2)

x 2 variants

x 11 FPGA families = 440 combinations

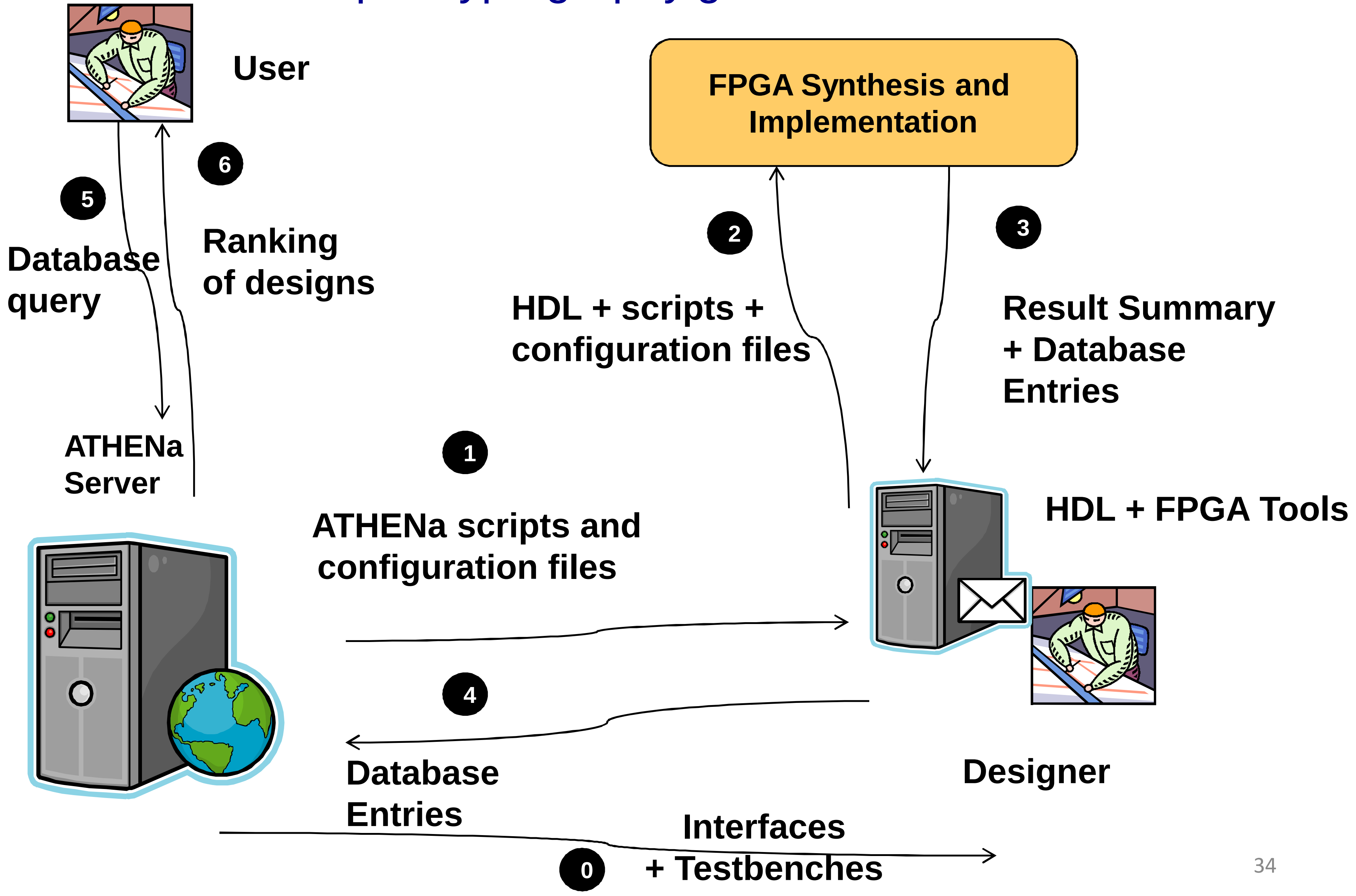
(440-not_fitting) = 423 optimized results

Support for easy replication of all results.

We invite other groups to submit results to our database

Invitation to Use ATHENa & ATHENa Database

http://cryptography.gmu.edu/athena


User
FPGA Synthesis and Implementation
6
5
Database query
Ranking of designs
2
3
HDL + scripts + configuration files
Result Summary + Database Entries
ATHENa Server
1
ATHENa scripts and configuration files
HDL + FPGA Tools
4
Database Entries
Designer
Interfaces + Testbenches
0

ATHENa – Progress since last CryptArchi

6 Sep 2010:	ATHENa 0.5.1
20 Nov 2010:	ATHENa 0.6
14 Dec 2010:	ATHENa 0.6.1
16 Jun 2011:	ATHENa 0.6.2

New in ATHENa 0.6

- support for Linux
- new comprehensive optimization strategy for Altera and Xilinx FPGAs: GMU_optimization_1
- possibility of iterating through multiple values of generics
- support for reducing the size of generated files (data trimming mode)
- support for using ATHENa together with Altera MegaWizard Plug-in Manager and Xilinx CORE Generator
- support for Verilog source files

New in ATHENa 0.6.2

- Capability to create replication files that can be used to regenerate optimized results without using ATHENa
- Extended support to generate database entries containing selected results to be uploaded to the ATHENa database.
- Support for AHDL
- Allow purely combinational circuits
- Added support for Quartus II 10.1 and Xilinx ISE 12.4 & 13.1
- Added support for new FPGA families

Coming soon!

New versions:

0.7: automated verification of designs through post-synthesis and timing simulation in batch mode

0.8: support for Actel FPGAs

0.9: additional heuristic optimization algorithms

1.0: accommodating comments received from users testing earlier versions.

Thank you!

Questions?

Questions?

CERG: http:/cryptography.gmu.edu

ATHENa: http:/cryptography.gmu.edu/athena