Comprehensive Comparison of Hardware Performance of Fourteen Round 2 SHA-3 Candidates with 512-bit Outputs Using Field Programmable Gate Arrays

Kris Gaj,
Ekawat Homsirikamol, and
Marcin Rogawski
George Mason University
U.S.A.

ATHENa – Automated Tool for Hardware EvaluatioN:

Toward Fair and Comprehensive Benchmarking of Cryptographic Algorithms using FPGAs

Kris Gaj, Jens-Peter Kaps,
Venkata Amirineni,
Marcin Rogawski,
Ekawat Homsirikamol
George Mason University
U.S.A.

Our Goals

- **Fair and comprehensive methodology** for evaluation of hardware performance in FPGAs (see our paper at CHES)

- **High-speed** fully autonomous implementations of

  all **14 SHA-3 candidates** & SHA-2

  256-bit & 512-bit variants

  optimized for the **maximum throughput to area ratio**

- **Open-source benchmarking tool** supporting optimization of tool options and efficient generation of results for multiple FPGA families

Our Methodology

- Design assumptions
    - Common and practical interface
    - No use of dedicated FPGA resources (Block RAMs, DSP units, etc.)
    - Padding in software (padding in hardware to be added soon)
    - No special modes of operation (salt, MAC, tree hashing)
- Language
    - VHDL
- Tools
    - standard FPGA vendor tools: Xilinx ISE & Quartus II
- Result generation for multiple FPGA families

Xilinx: Spartan 3, Virtex 4, Virtex 5

Altera: Cyclone II, Cyclone III, Stratix II, Stratix III

ATHENa – **A**utomated **T**ool for **H**ardware **E**valuatio**N**

http://cryptography.gmu.edu/athena

Benchmarking open-source tool, written in Perl, aimed at an

AUTOMATED generation of
OPTIMIZED results for
MULTIPLE hardware platforms

Currently under development at George Mason University.

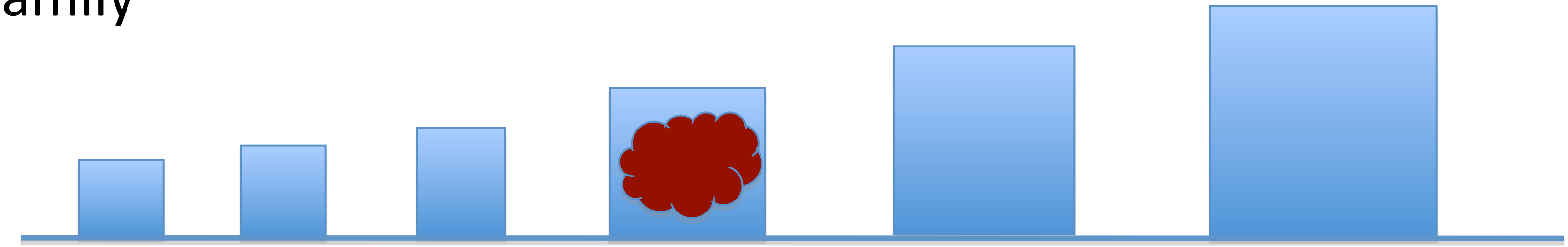
ATHENa Major Features (1)

- synthesis, implementation, and timing analysis in **batch mode**
- support for devices and tools of **multiple FPGA vendors**:

XILINX ALTERA

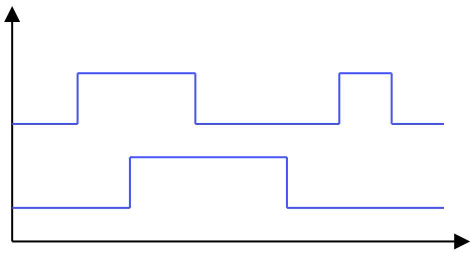
- generation of results for **multiple families** of FPGAs of a given vendor

- automated choice of a **best-matching device** within a given family



ATHENa Major Features (2)

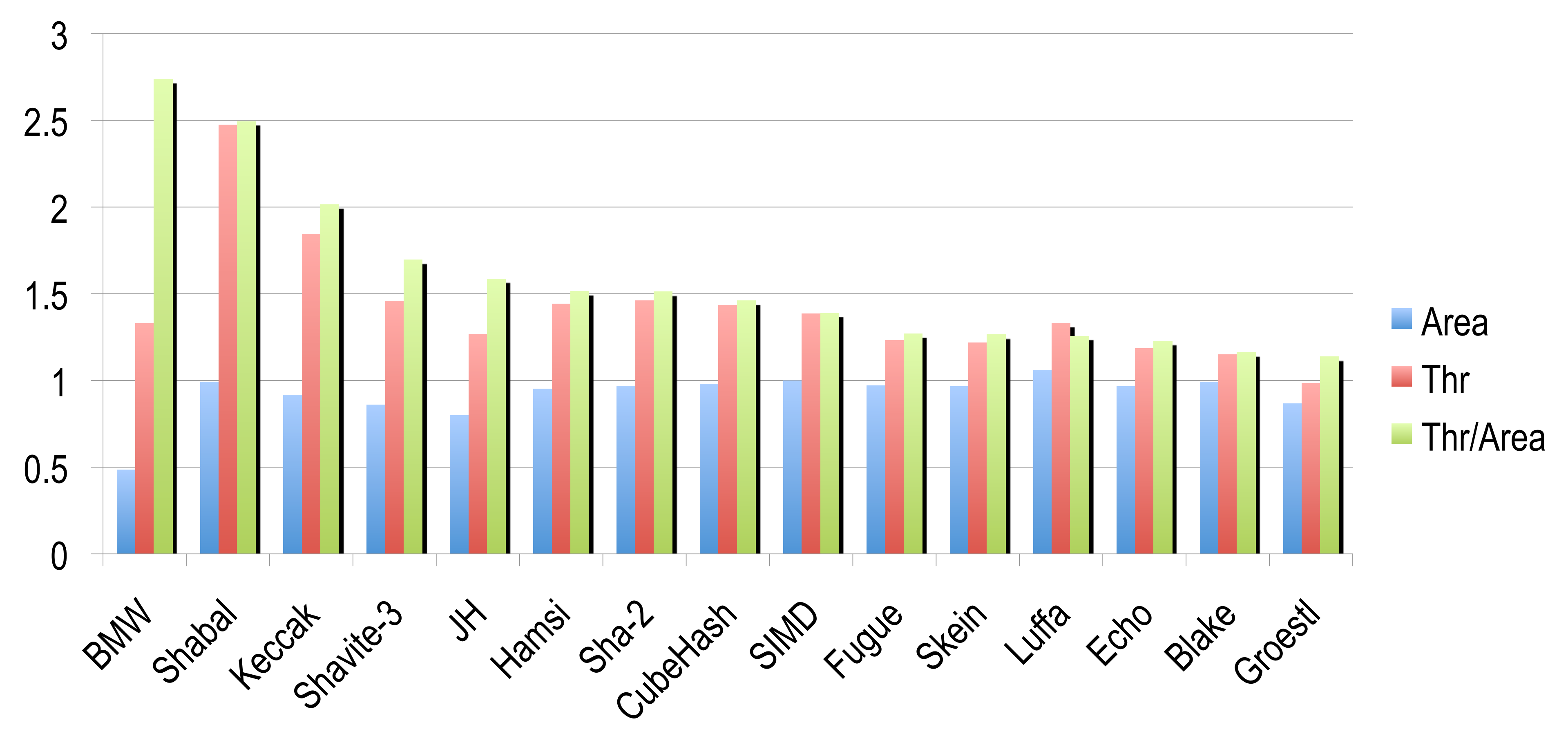
- **automated verification** of designs through simulation in batch mode



OR

- support for **multi-core processing**
- several **optimization strategies** aimed at finding
  - optimum options of tools
  - best target clock frequency
  - best starting point of placement
- automated **extraction and tabulation of results**

Relative Improvement of Results from Using ATHENa Virtex 5, 512-bit Variants of Hash Functions



Ratios of results obtained using ATHENa suggested options vs. default options of FPGA tools

Results

Performance Metrics

Primary

1. Throughput
   (single long message)

3. Throughput / Area

Secondary

2. Area

3. Hash Time for
   Short Messages
   (up to 1000 bits)

Correction Regarding Skein with a 256-bit Output

Variant of Skein used in our CHES paper & presentation:

(and other publications reported at the SHA-3 Zoo)

Skein-256-256

Variant recommended by the Authors of Skein:

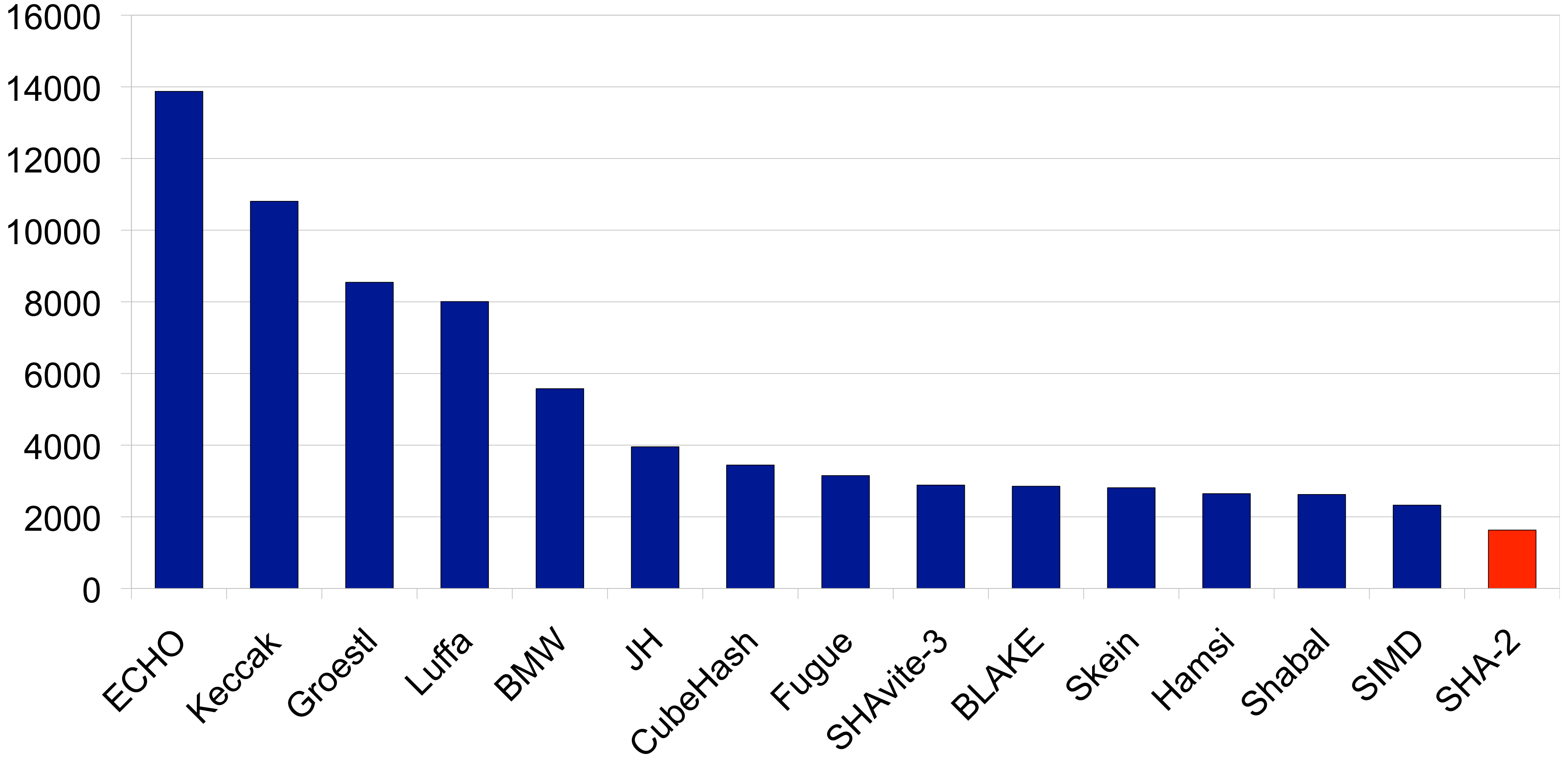
Skein-512-256

Proper values of the overall normalized parameters (vs. SHA-256):

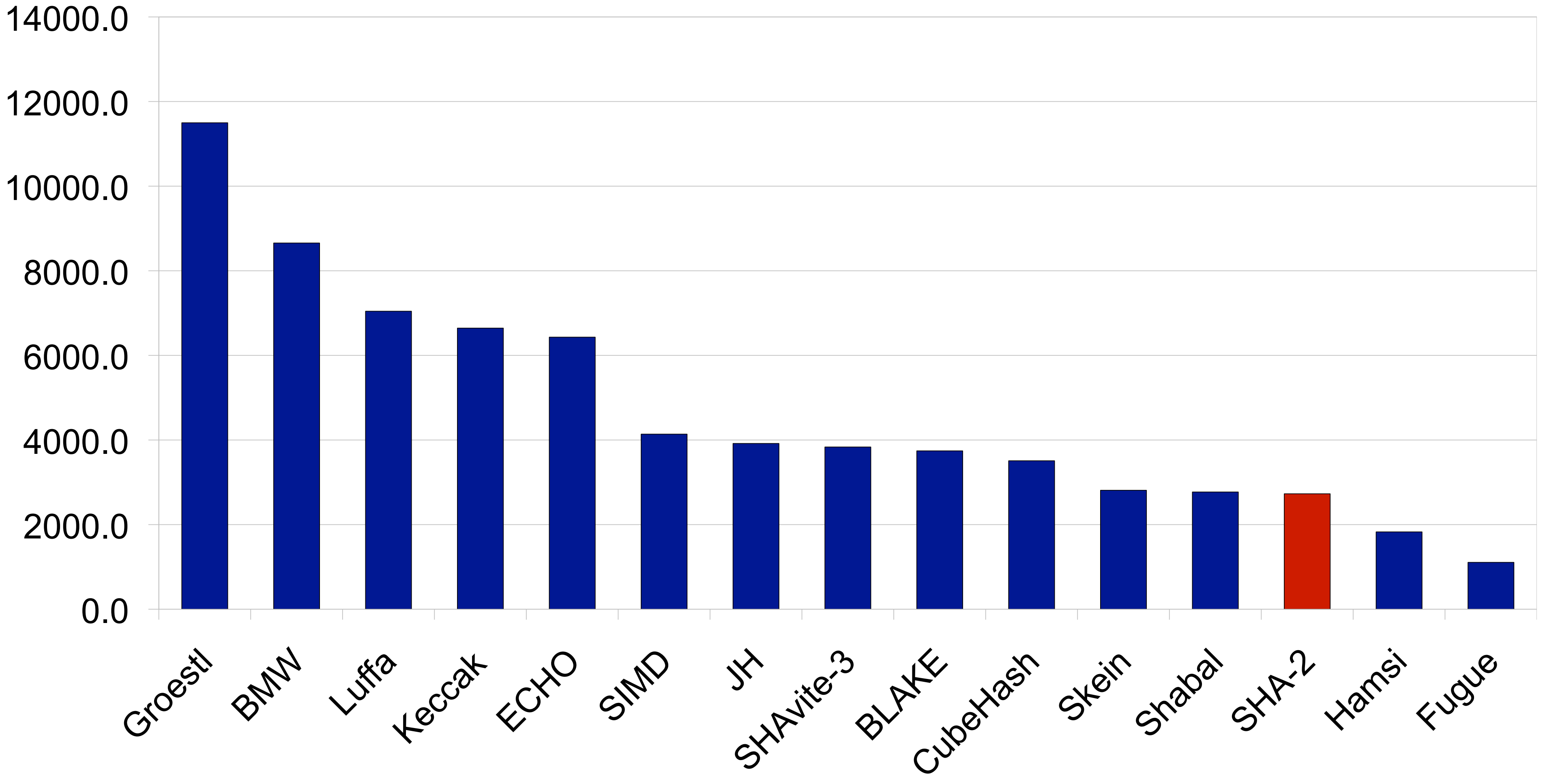
Throughput:	**0.79**	→	**1.50**	🟥	→	🟨
Area:	**3.41**	→	**4.09**	🟨	→	🟨
Throughput/Area:	**0.23**	→	**0.37**	🟥	→	🟨

We apologize for this mistake!!!

Throughput [Mbit/s]
Virtex 5, 256-bit variants of algorithms



Throughput [Mbit/s]
Virtex 5, 512-bit variants of algorithms



Normalization & Compression of Results

- **Absolute result**

  e.g., throughput in Mbits/s, area in CLB slices

- **Normalized result**

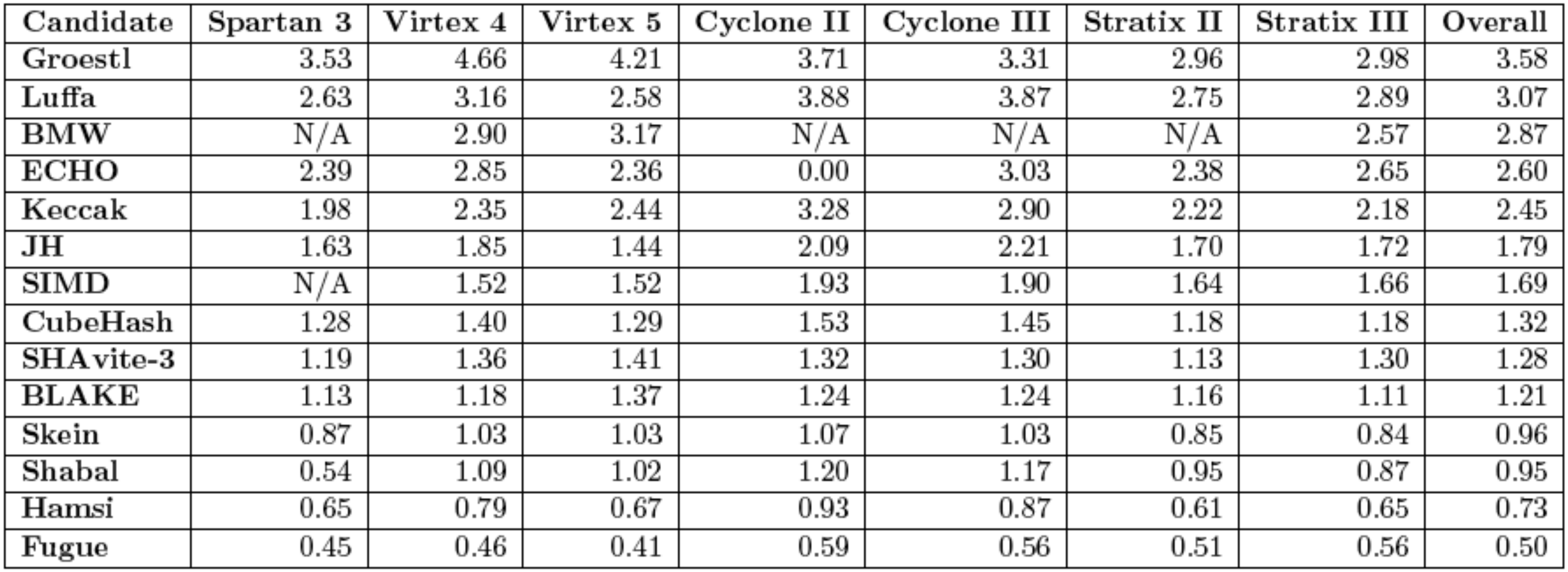
$$normalized\_result = \frac{result\_for\_SHA-3\_candidate}{result\_for\_SHA-2}$$

- **Overall normalized result**

  Geometric mean of normalized results for

  all investigated FPGA families

Normalized Throughput
& Overall Normalized Throughput, 512-bit variants

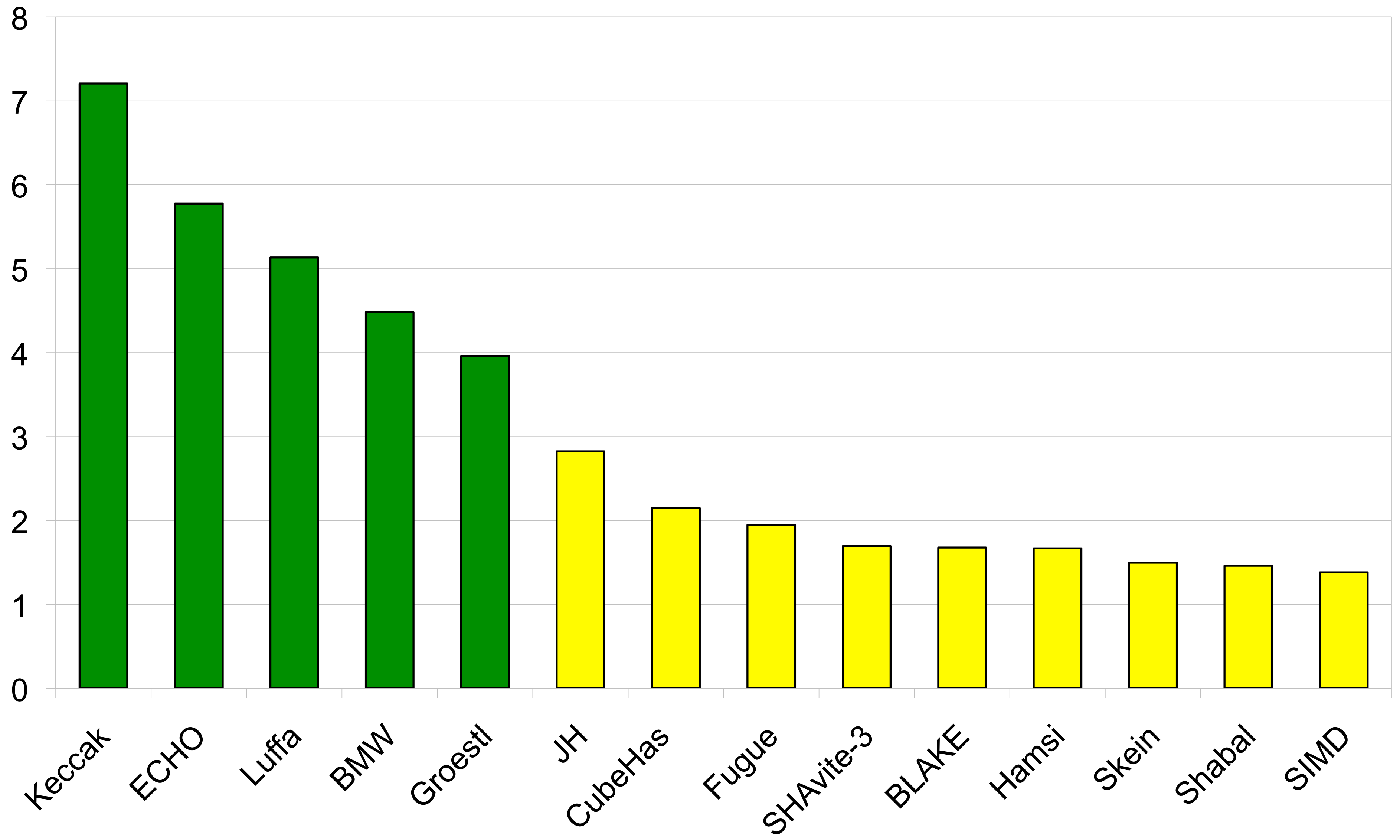


Candidate	Spartan 3	Virtex 4	Virtex 5	Cyclone II	Cyclone III	Stratix II	Stratix III	Overall
Groestl	3.53	4.66	4.21	3.71	3.31	2.96	2.98	3.58
Luffa	2.63	3.16	2.58	3.88	3.87	2.75	2.89	3.07
BMW	N/A	2.90	3.17	N/A	N/A	N/A	2.57	2.87
ECHO	2.39	2.85	2.36	0.00	3.03	2.38	2.65	2.60
Keccak	1.98	2.35	2.44	3.28	2.90	2.22	2.18	2.45
JH	1.63	1.85	1.44	2.09	2.21	1.70	1.72	1.79
SIMD	N/A	1.52	1.52	1.93	1.90	1.64	1.66	1.69
CubeHash	1.28	1.40	1.29	1.53	1.45	1.18	1.18	1.32
SHAvite-3	1.19	1.36	1.41	1.32	1.30	1.13	1.30	1.28
BLAKE	1.13	1.18	1.37	1.24	1.24	1.16	1.11	1.21
Skein	0.87	1.03	1.03	1.07	1.03	0.85	0.84	0.96
Shabal	0.54	1.09	1.02	1.20	1.17	0.95	0.87	0.95
Hamsi	0.65	0.79	0.67	0.93	0.87	0.61	0.65	0.73
Fugue	0.45	0.46	0.41	0.59	0.56	0.51	0.56	0.50

Overall = Geometric mean of normalized results for all investigated FPGA families

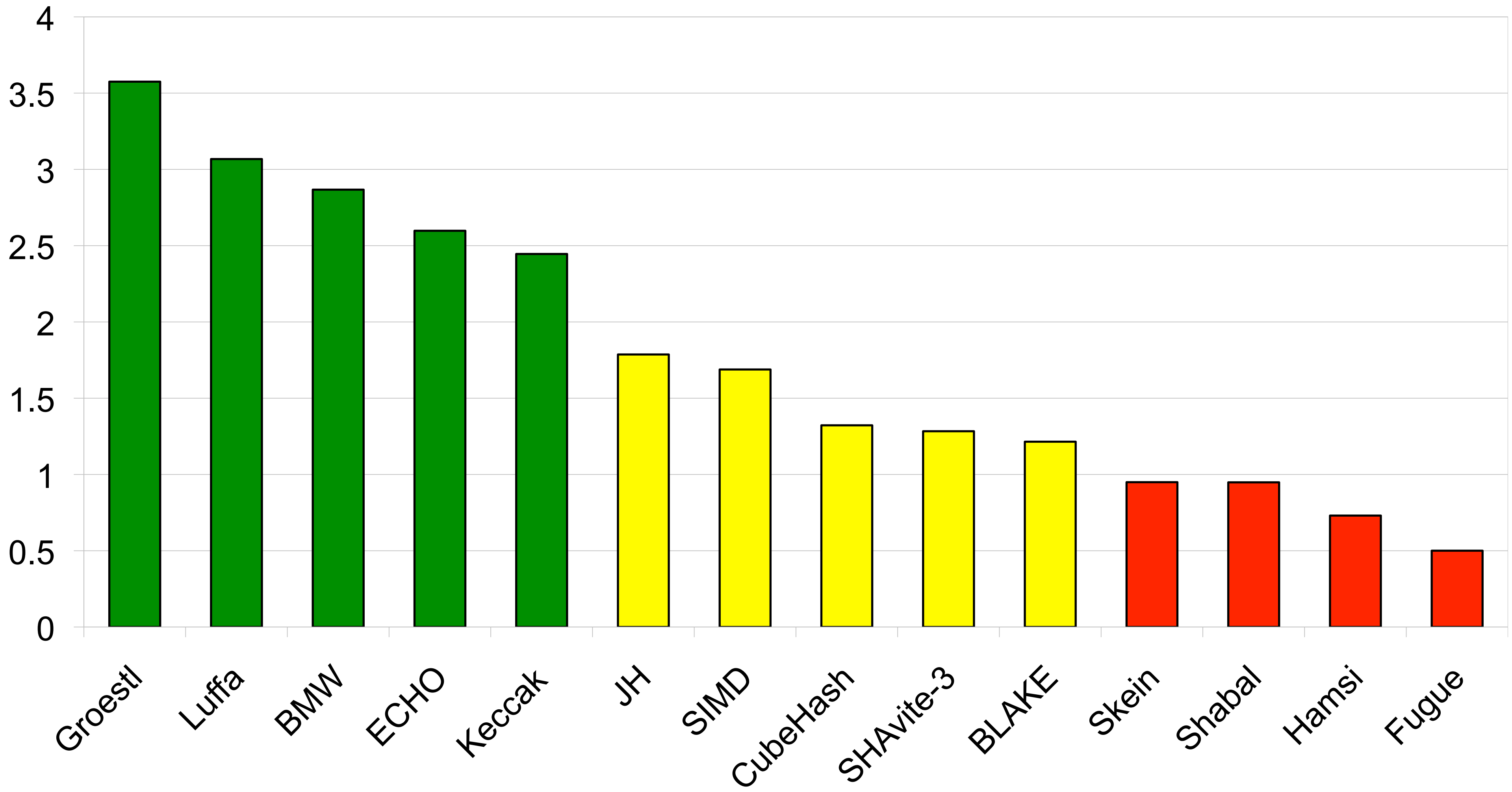
Overall Normalized Throughput: 256-bit variants of algorithms

Normalized to SHA-256, Averaged over 7 FPGA families

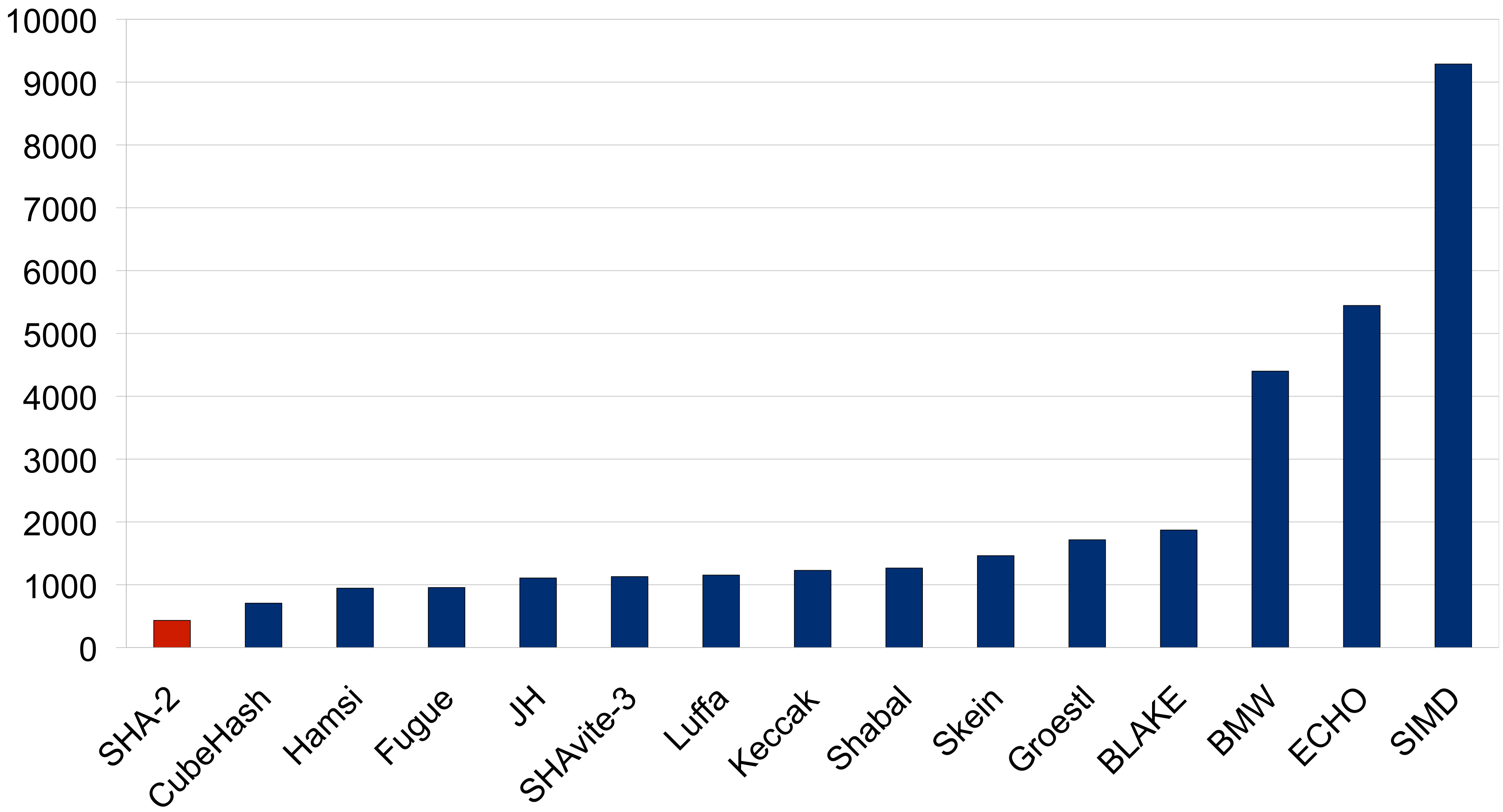


Overall Normalized Throughput: 512-bit variants of algorithms

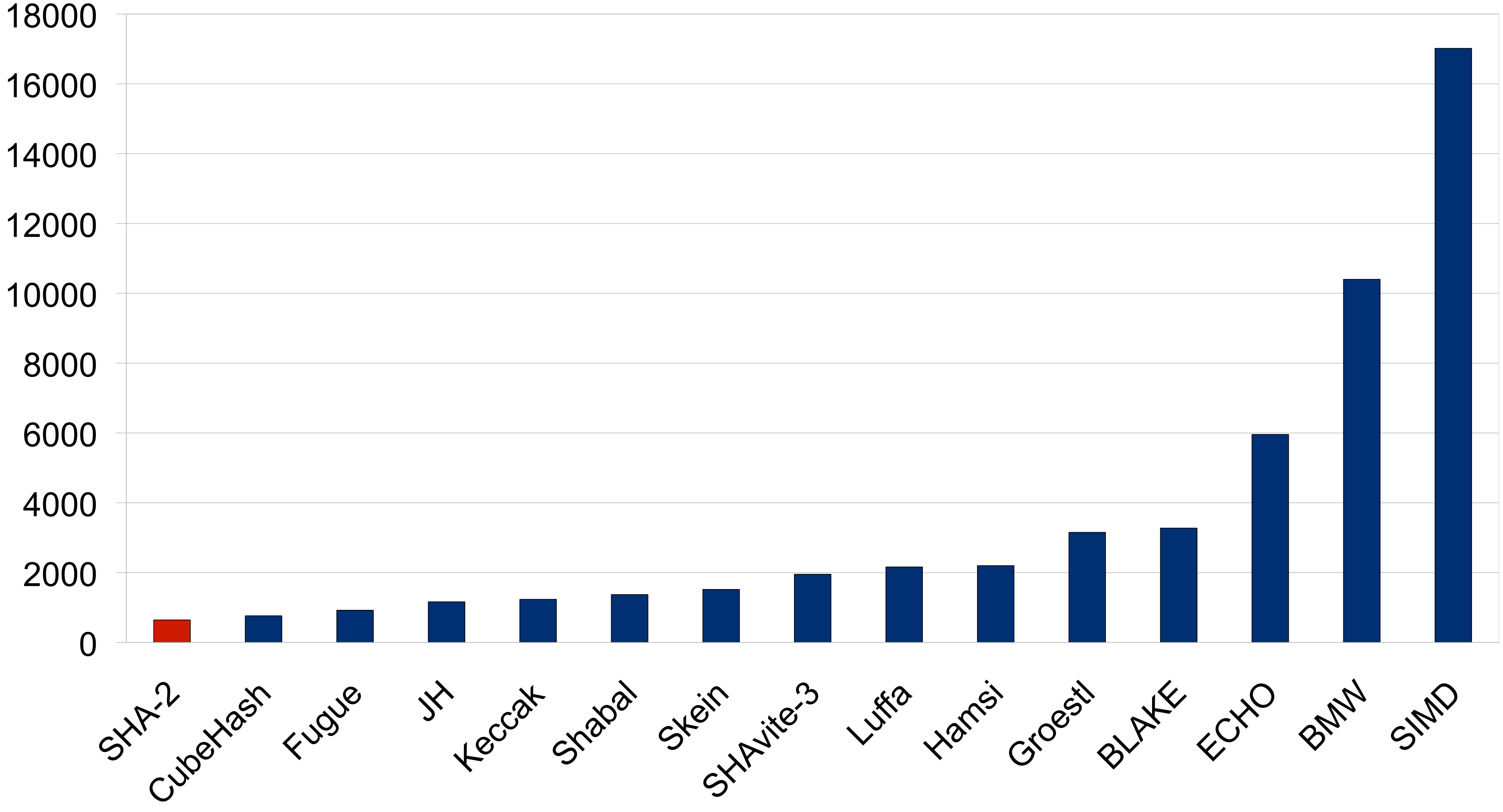
Normalized to SHA-512, Averaged over 7 FPGA families



Area [CLB slices]
Virtex 5, 256-bit variants of algorithms

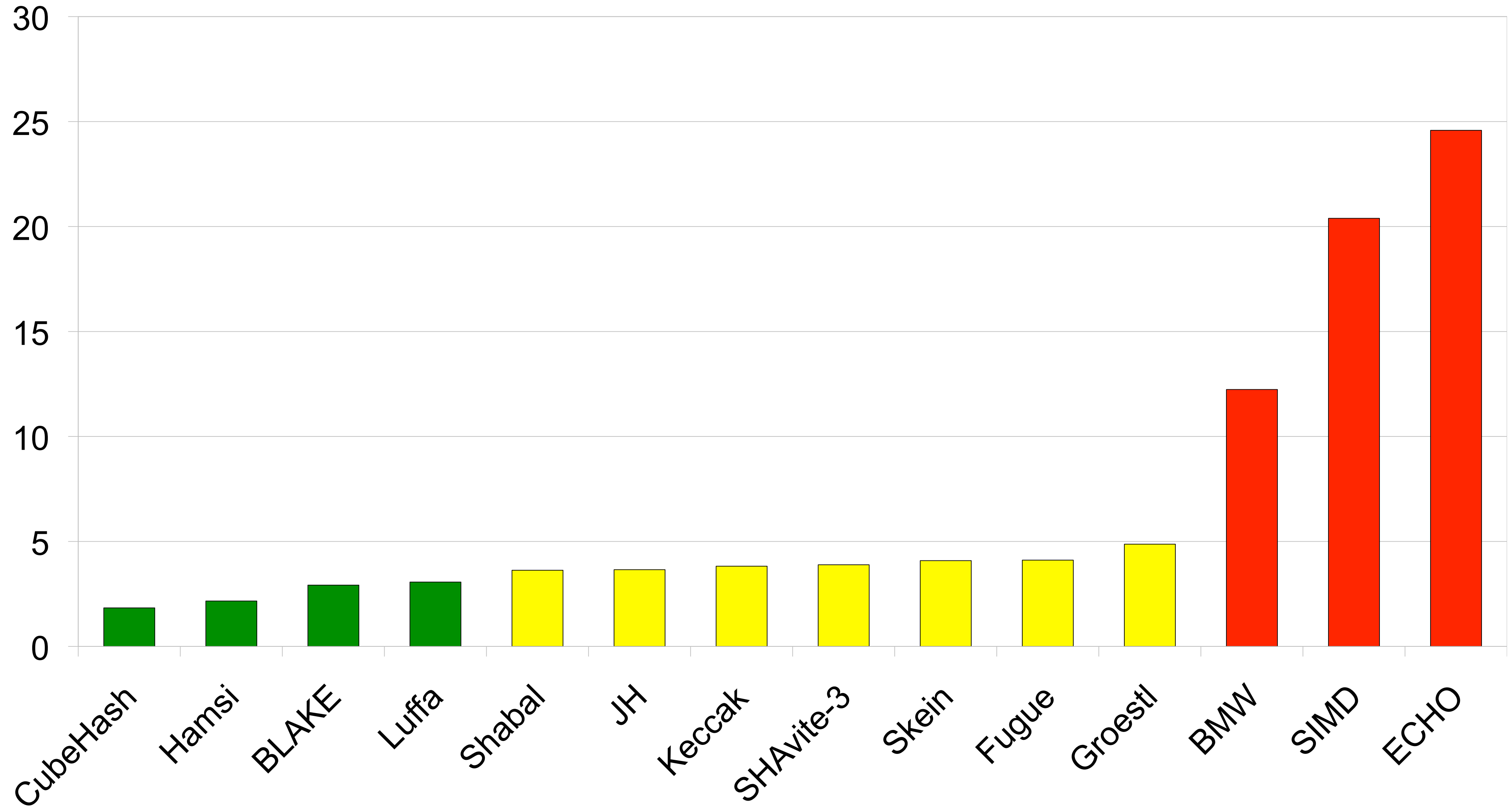


Area [CLB slices]
Virtex 5, 512-bit variants of algorithms



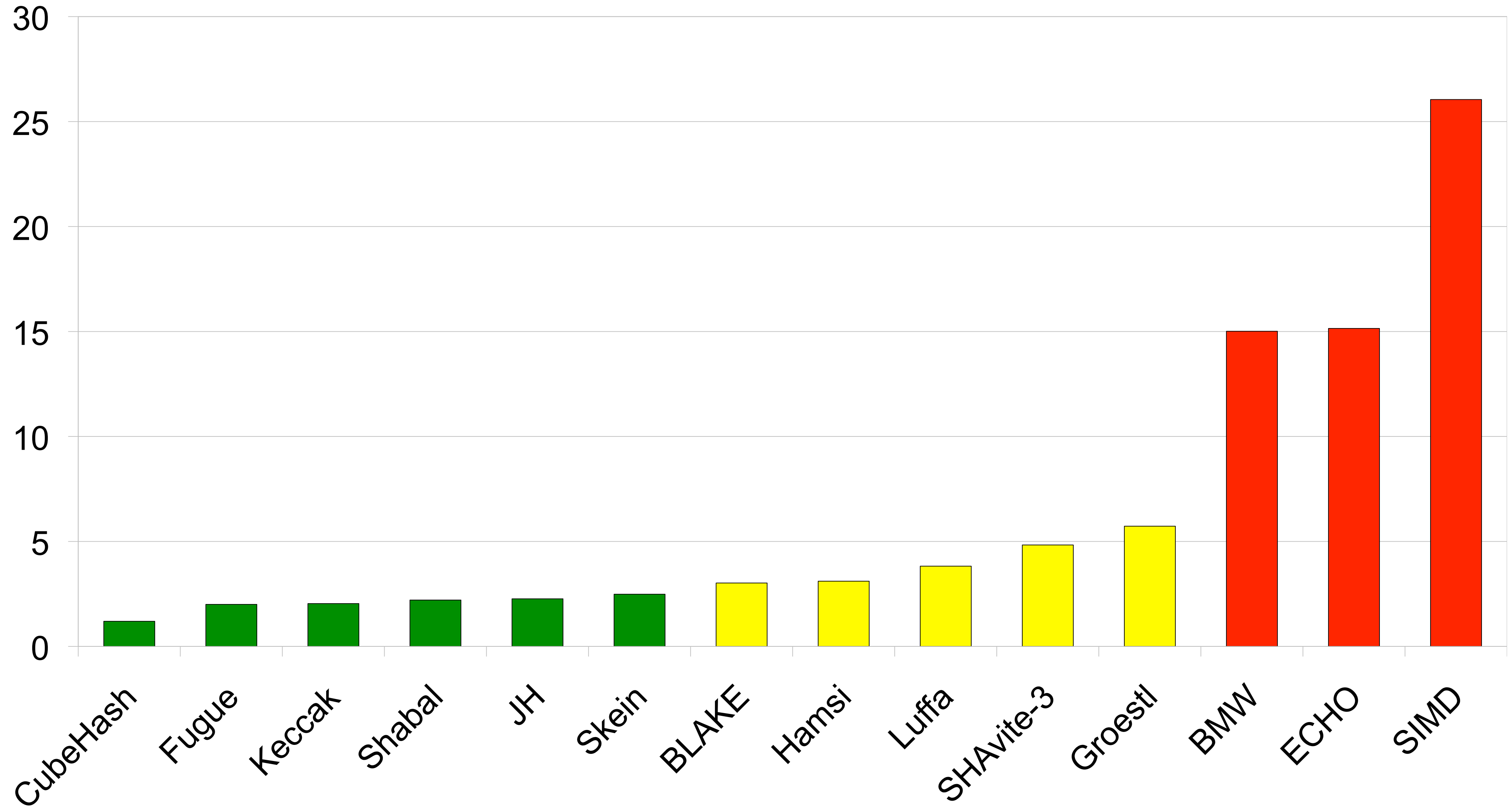
Overall Normalized Area: 256-bit variants of algorithms

Normalized to SHA-256, Averaged over 7 FPGA families



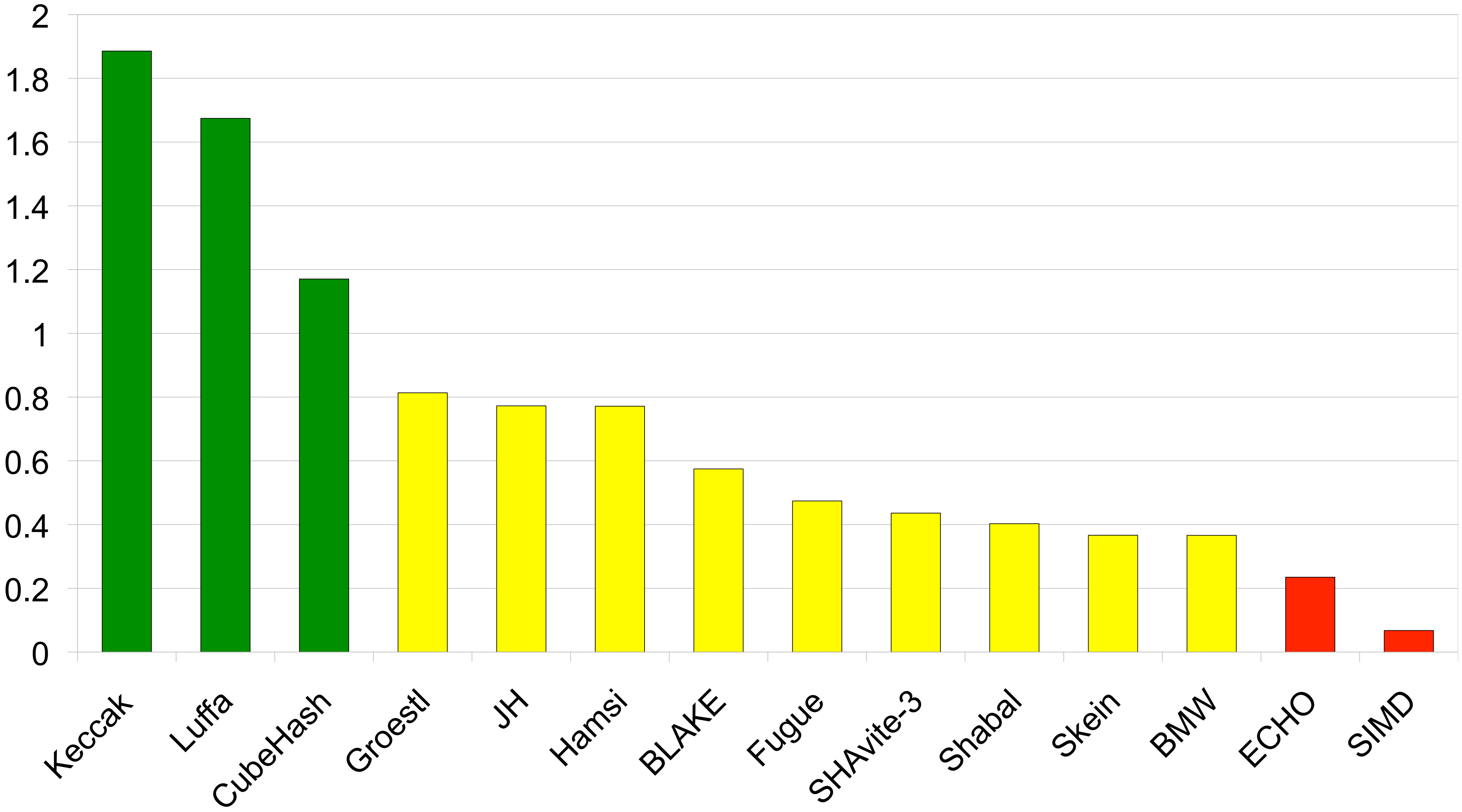
Overall Normalized Area: 512-bit variants of algorithms

Normalized to SHA-512, Averaged over 7 FPGA families



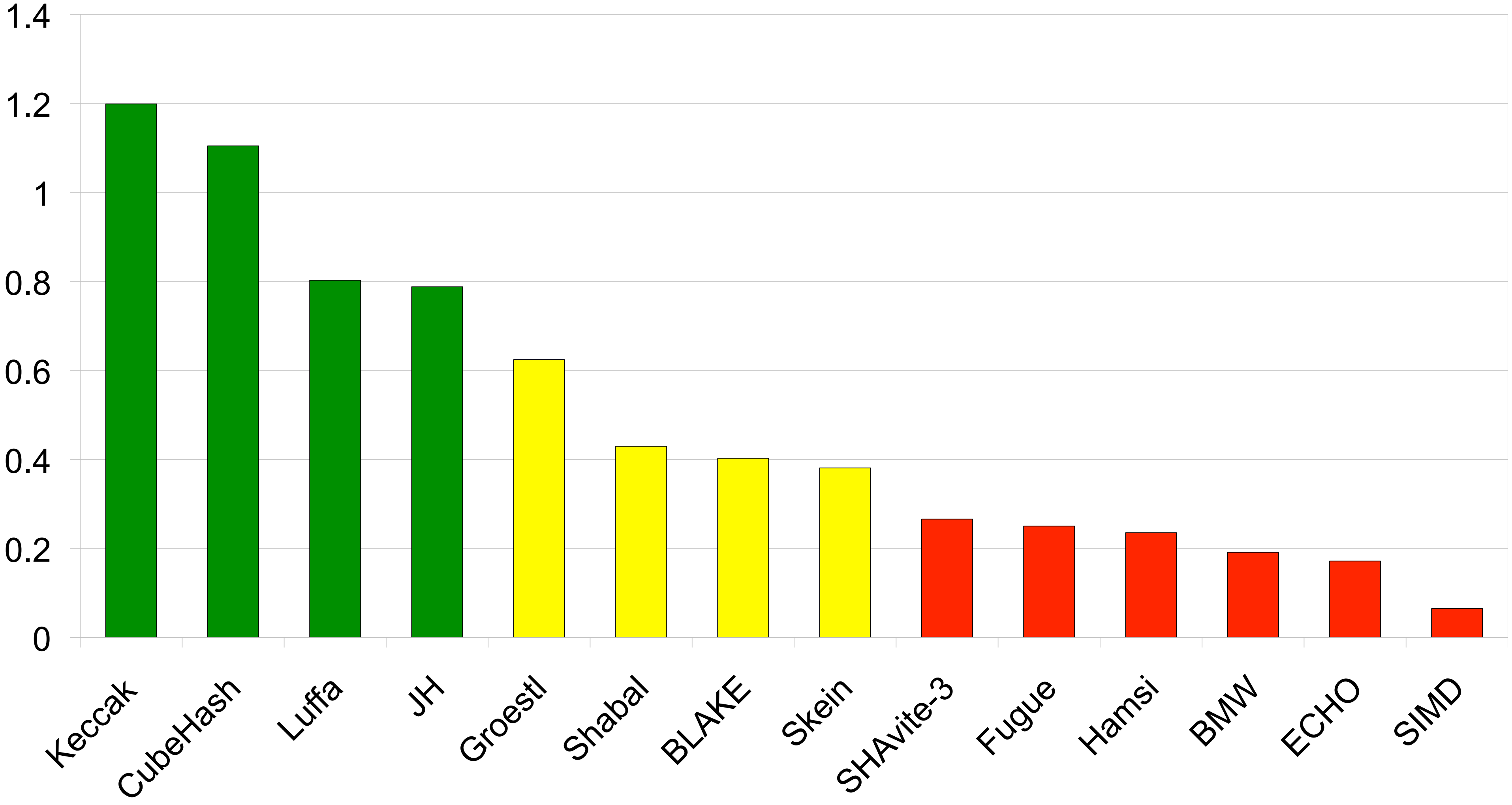
Overall Normalized Throughput/Area: 256-bit variants

Normalized to SHA-256, Averaged over 7 FPGA families

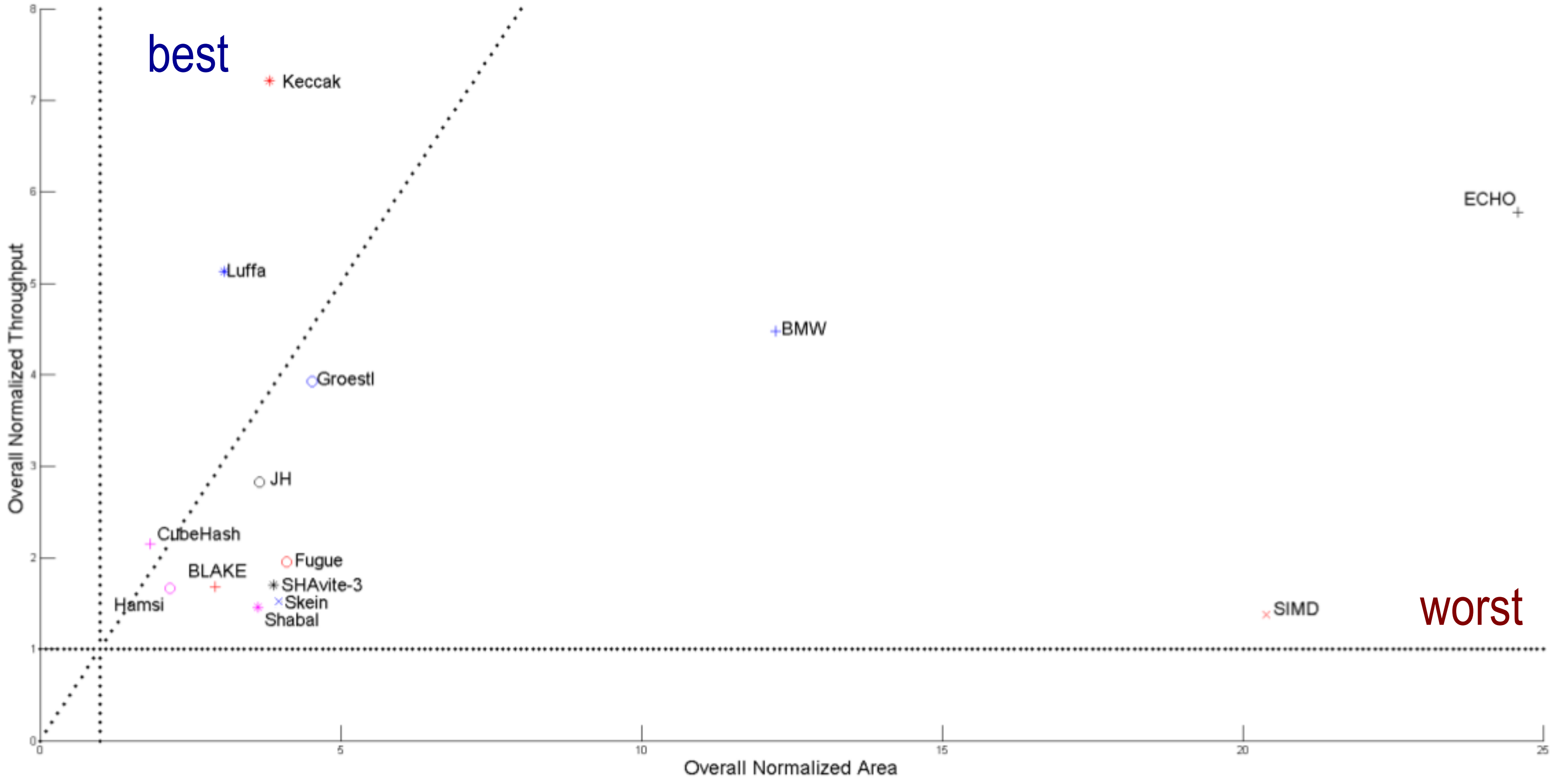


Overall Normalized Throughput/Area: 512-bit variants

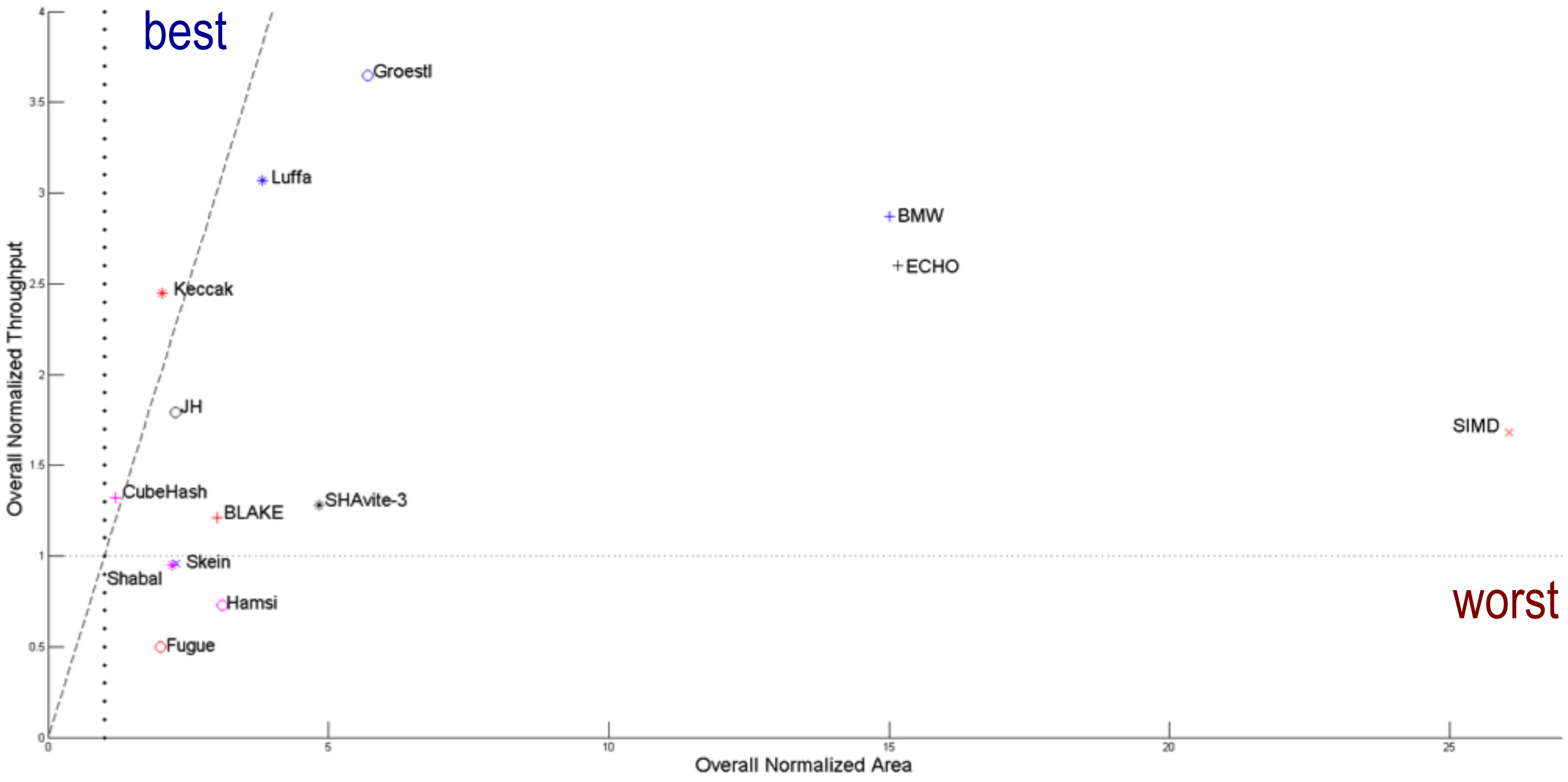
Normalized to SHA-512, Averaged over 7 FPGA families



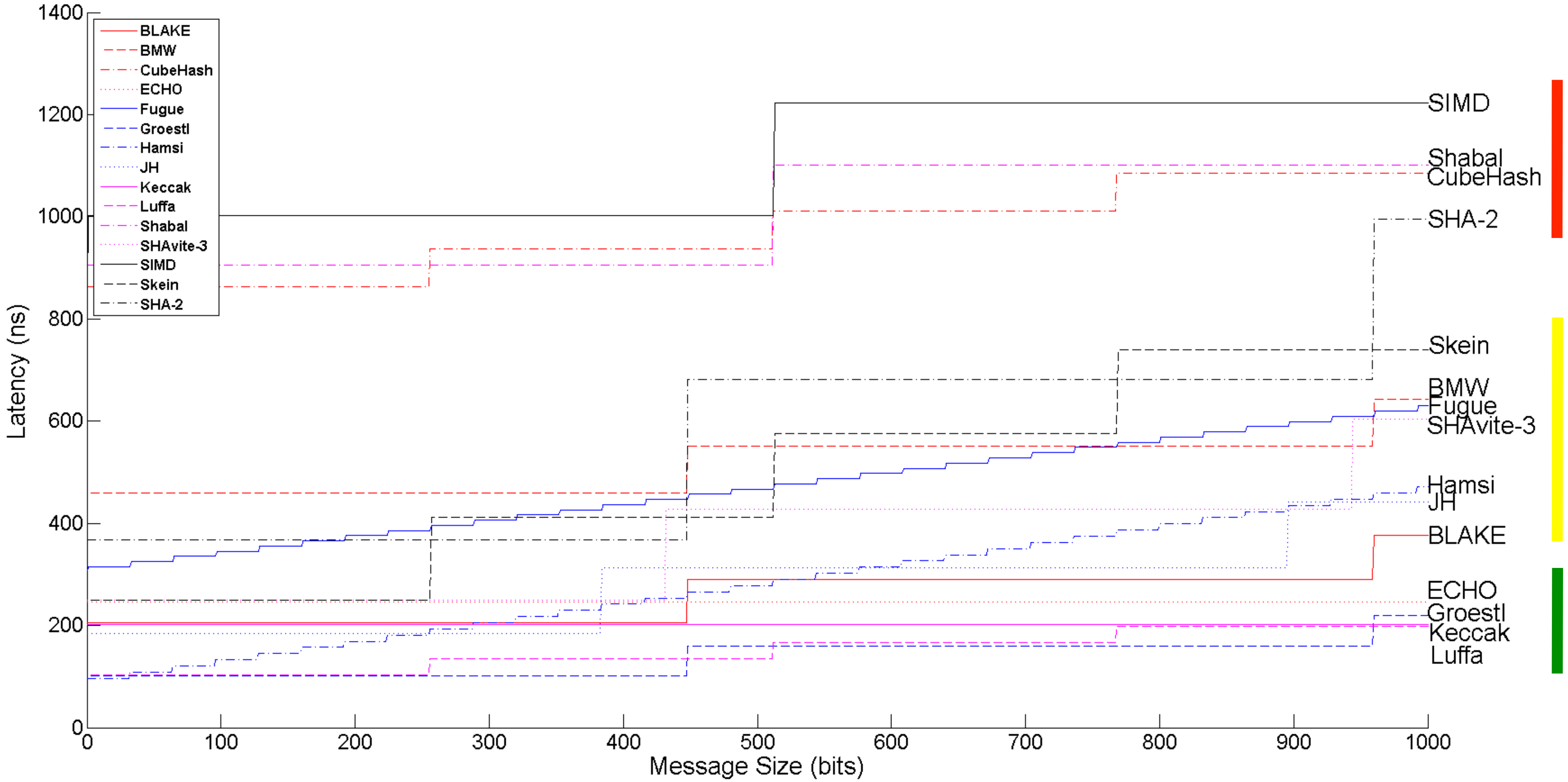
Throughput vs. Area Normalized to Results for SHA-256 and Averaged over 7 FPGA Families – 256-bit variants



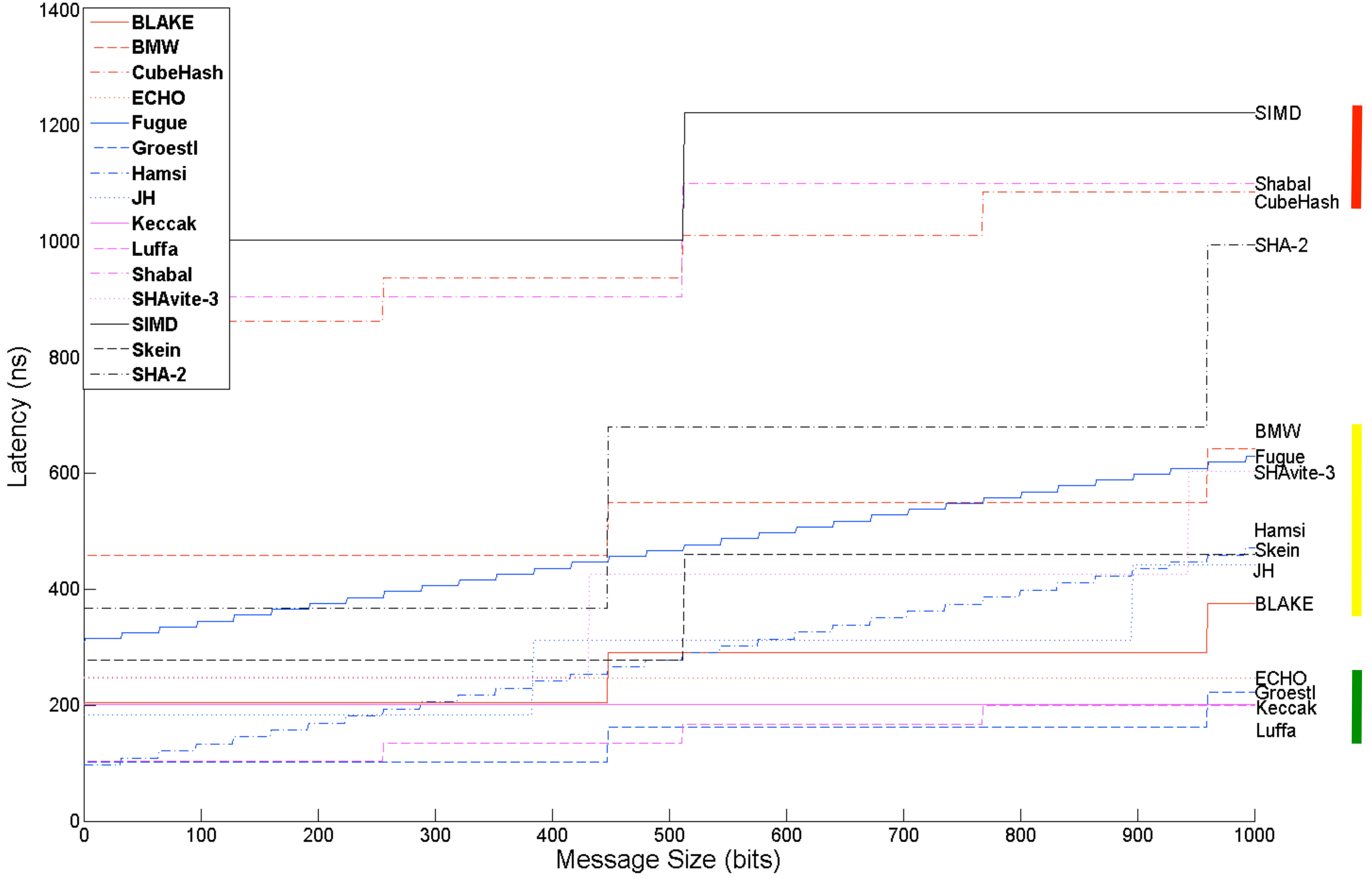
Throughput vs. Area Normalized to Results for SHA-512 and Averaged over 7 FPGA Families – 512-bit variants



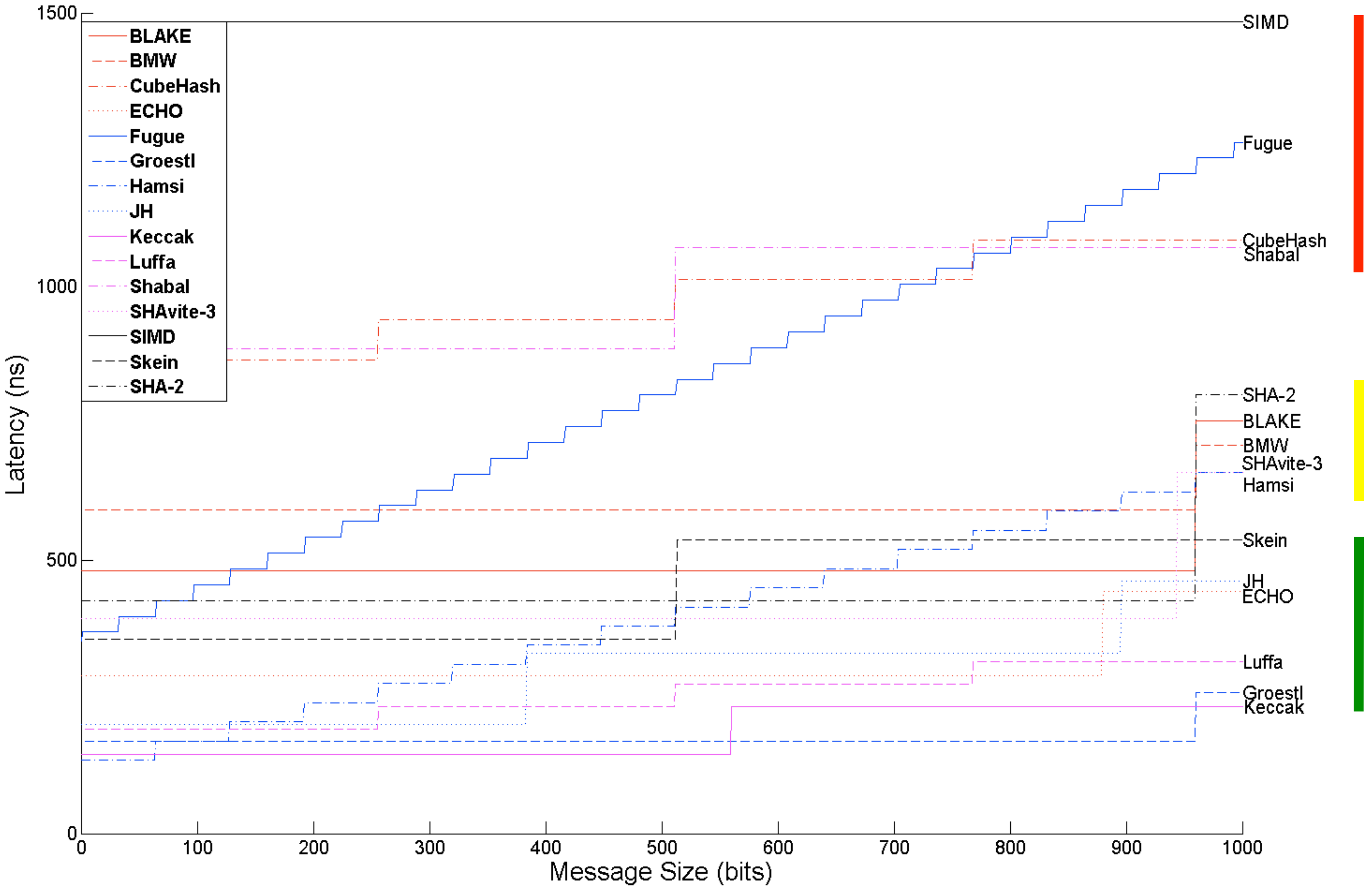
Execution Time for Short Messages up to 1000 bits
Virtex 5, 256-bit variants of algorithms



Execution Time for Short Messages up to 1000 bits
Virtex 5, 256-bit variants of algorithms



Execution Time for Short Messages up to 1000 bits
Virtex 5, 512-bit variants of algorithms



		256-bit variants				512-bit variants			
		Thr/Area	Thr	Area	Short msg.	Thr/Area	Thr	Area	Short msg.
	BLAKE	🟨	🟨	🟩	🟨	🟨	🟨	🟨	🟨
➡	**BMW**	🟨	🟩	🟥	🟨	🟥	🟩	🟥	🟨
	CubeHash	🟩	🟨	🟩	🟥	🟩	🟨	🟩	🟥
➡	**ECHO**	🟥	🟩	🟥	🟩	🟥	🟩	🟥	🟩
	Fugue	🟨	🟨	🟨	🟨	🟥	🟥	🟩	🟥
➡	**Groestl**	🟨	🟩	🟨	🟩	🟨	🟩	🟨	🟩
	Hamsi	🟨	🟨	🟩	🟨	🟥	🟥	🟨	🟨
	JH	🟨	🟨	🟨	🟨	🟩	🟨	🟩	🟩
➡	**Keccak**	🟩	🟩	🟨	🟩	🟩	🟩	🟩	🟩
➡	**Luffa**	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟩
	Shabal	🟨	🟨	🟨	🟥	🟨	🟥	🟩	🟥
	SHAvite-3	🟨	🟨	🟨	🟨	🟥	🟨	🟨	🟨
➡	**SIMD**	🟥	🟨	🟥	🟥	🟥	🟨	🟥	🟥
	Skein	🟨	🟨	🟨	🟨	🟨	🟥	🟩	🟩

Summary of Results

- Throughput/Area & Throughput most crucial for high-speed implementations
- Area cannot be easily traded for Throughput

Best performers so far

1-2. Keccak & Luffa

3. Groestl

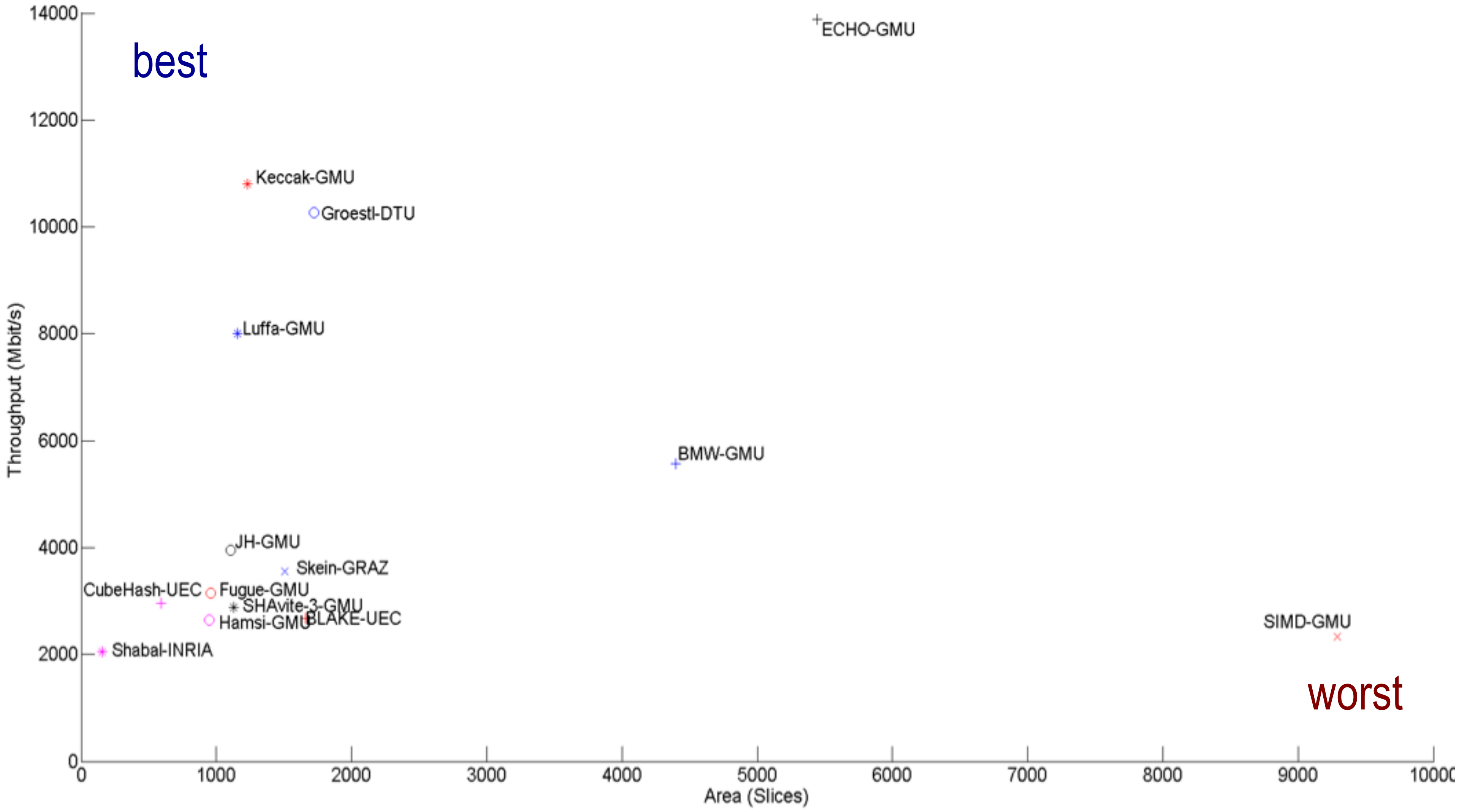
Worst performers so far:

14. SIMD

13. ECHO

12. BMW

Throughput vs. Area: Best reported results
Virtex 5, 256-bit variants of algorithms



Hints for Designers of Hash Functions

- Easy way to predict **approximately** the change in speed and area when moving from a 256-bit to a 512-bit variant

  in **high-speed** hardware implementations

$$\frac{Area(512)}{Area(256)} \approx \frac{Datapath\_width(512)}{Datapath\_width(256)} = \frac{State\_size(512)}{State\_size(256)}$$

$$\frac{Thr(512)}{Thr(256)} \approx \frac{\dfrac{Block\_size(512)}{Block\_size(256)}}{\dfrac{Round\_no(512)}{Round\_no(256)}} = \frac{Block\_size\_ratio}{Round\_no\_ratio}$$

512-bit variant vs. 256-bit variant – Predicted Behavior

Group 1: Area: ↔ Thr: ↔ Thr/Area: ↔

CubeHash, JH, Shabal, Skein

Group 2: Area: ↗ x2 Thr: ↗ x2 Thr/Area: ↔

BMW, SIMD

Group 3: Area: ↗ Thr: ↗ Thr/Area: ↘

BLAKE, Groestl, SHAvite-3, SHA-2

Group 4: Area: ↔ Thr: ↘ Thr/Area: ↘

ECHO, Keccak

Group 5: Area: ↗ Thr: ↔ Thr/Area: ↘

Hamsi, Luffa

Group 6: Area: ↗ Thr: ↘ Thr/Area: ↘

Fugue

More About our Designs & Tools

- CHES 2010 paper
  - **Methodology**
  - Results for 256-bit variants

FPL 2010 paper

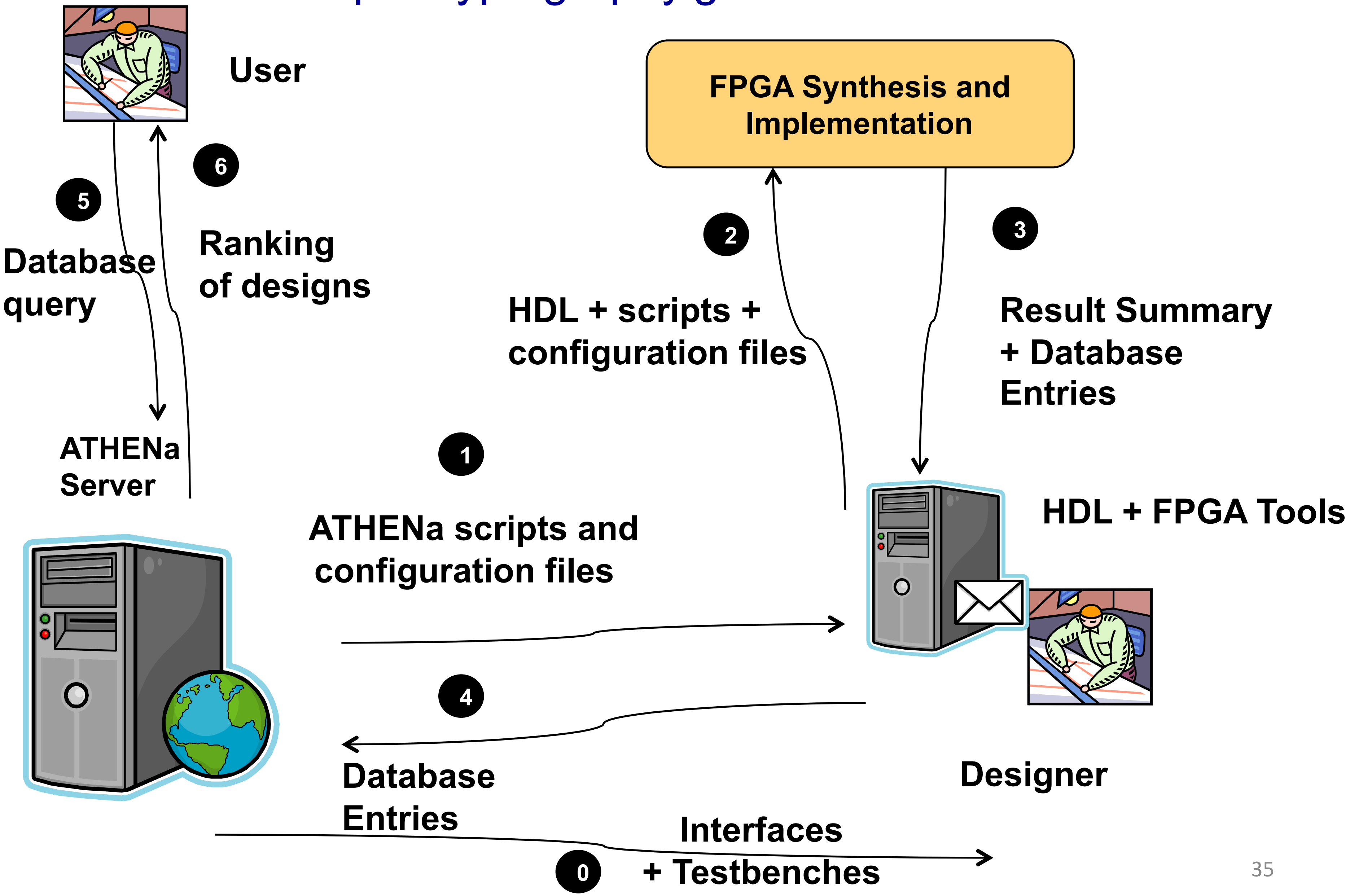
- ATHENa features
- Case studies

- Cryptology e-Print Archive
  - **Detailed hierarchical block diagrams**
  - **Corresponding formulas for execution time and throughput**

- ATHENa web site
  - **Most recent results**
  - Comparisons with results from other groups
  - **Optimum options of tools**

Invitation to Use ATHENa

http://cryptography.gmu.edu/athena



User

5

Database query

6

Ranking of designs

ATHENa Server

FPGA Synthesis and Implementation

2

HDL + scripts + configuration files

3

Result Summary + Database Entries

1

ATHENa scripts and configuration files

HDL + FPGA Tools

4

Database Entries

Designer

0

Interfaces + Testbenches

Thank you!

Questions?

Questions?

CERG: http:/cryptography.gmu.edu

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