From C to Hardware: Toward Using High-Level Synthesis for Hardware Benchmarking of Candidates in Cryptographic Contests

Kris Gaj
George Mason University

CERG @ GMU

http://cryptography.gmu.edu/

9 PhD students
8 MS students
co-advised by Kris Gaj & Jens-Peter Kaps

Major Focus of CERG

High-speed Implementations of Cryptographic Algorithms

Lightweight Implementations of Cryptographic Algorithms

Evaluation of Candidates for New Cryptographic Standards (secret-key, pairing-based, and post-quantum cryptography)

Implementation of Codebreaking Algorithms in Hardware

Side-Channel Attacks & Countermeasures

Primary Support for This Particular Project

Ekawat Homsirikamol
a.k.a “Ice”

Working on the PhD Thesis entitled “A New Approach to the Development of Cryptographic Standards Based on the Use of High-Level Synthesis Tools”

RTL codes developed by:
William Diehl,
Farnoud Farahmand,
Ahmed Ferozpuri,
Ekawat Homsirikamol, and
Marcin Rogawski

Outline

- Introduction & motivation
- Traditional vs. HLS-based development & benchmarking flow
- Case studies
    - SHA-3 contest
    - CAESAR contest
- Future work
- Conclusions

Cryptography is everywhere
We trust it because of standards

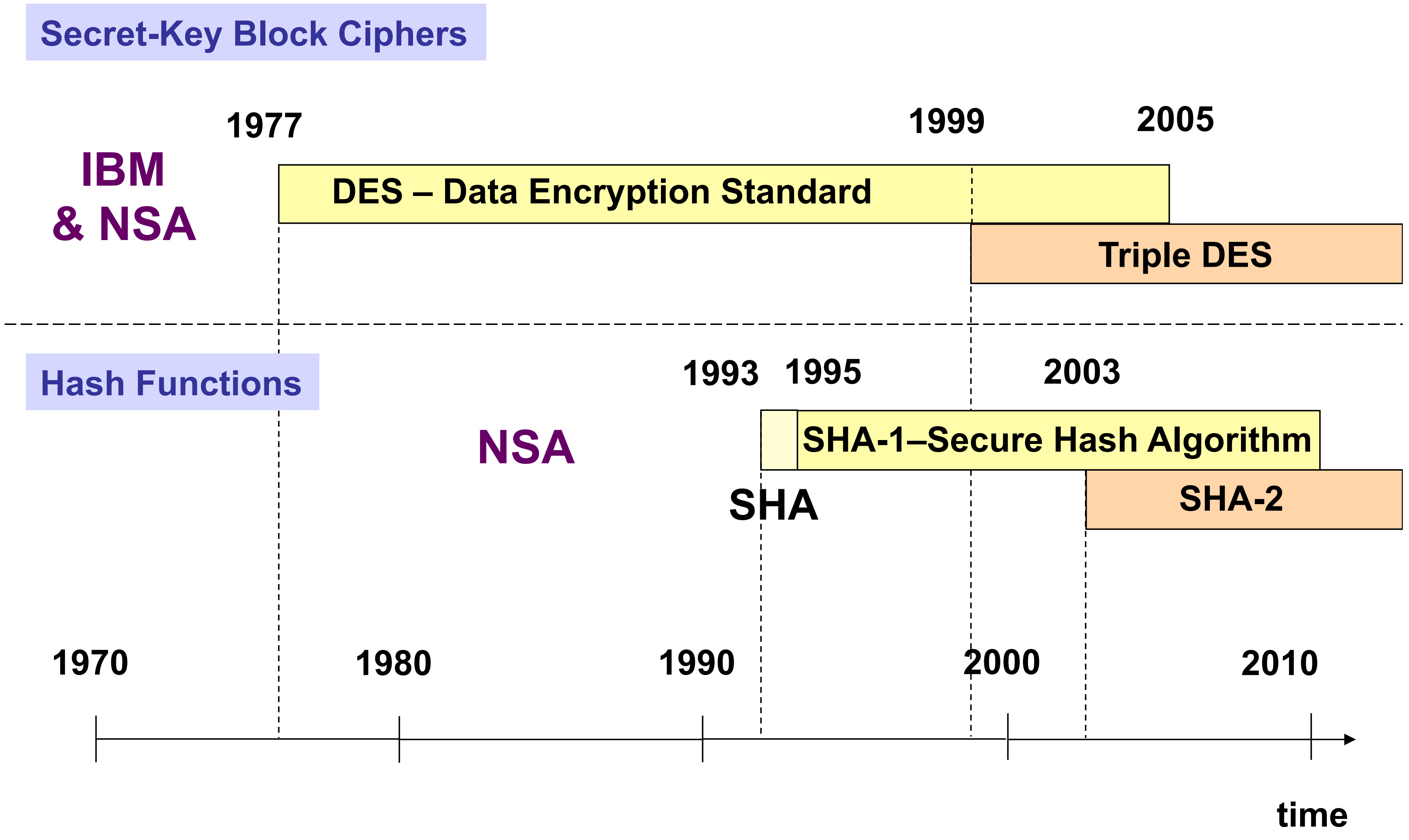
Buying a book on-line

Withdrawing cash from ATM

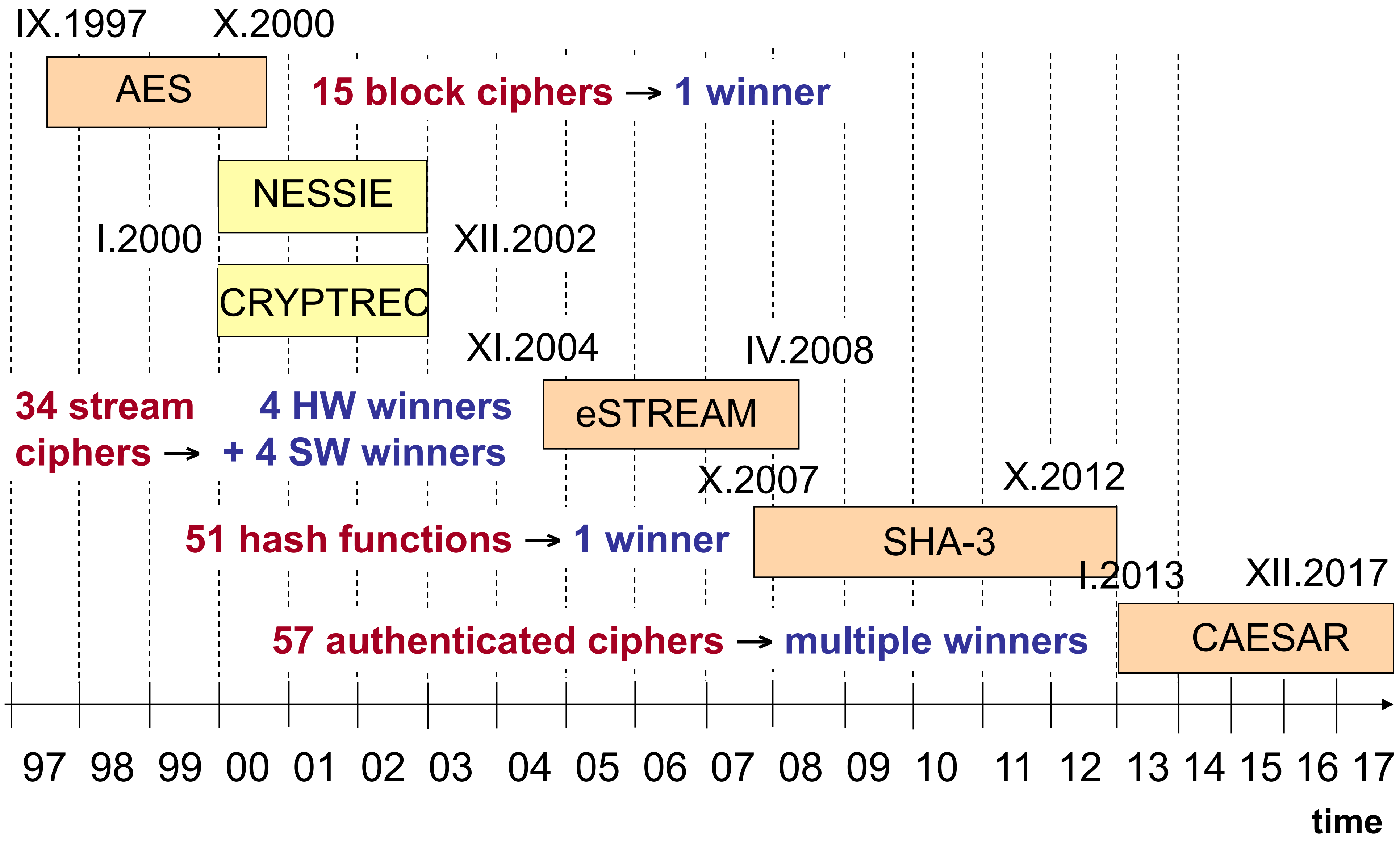
Teleconferencing
over Intranets

Backing up files
on remote server

Cryptographic Standards Before 1997



Cryptographic Standard Contests



Evaluation Criteria

Security

Software Efficiency

μProcessors μControllers

Hardware Efficiency

FPGAs ASICs

Flexibility

Simplicity

Licensing

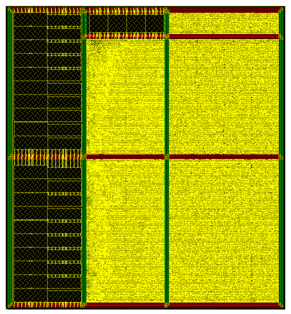
Benchmarking Tools

Tools for Benchmarking Implementations of Cryptography

Software

FPGAs

ASICs



SUPERCOP

D. Bernstein (UIC)
T. Lange (TUE)

2006-present

→

ATHENa

K. Gaj,
J. Kaps, et al.
(GMU)

2009-present

→

?

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

http://cryptography.gmu.edu/athena

Open-source benchmarking environment, written in Perl, aimed at
AUTOMATED generation of
OPTIMIZED results for
MULTIPLE hardware platforms.

FPL Community Award 2010

Why Athena?

"The Greek goddess Athena was frequently called upon to settle disputes between the gods or various mortals. Athena Goddess of Wisdom was known for her superb logic and intellect. Her decisions were usually well-considered, highly ethical, and seldom motivated by self-interest."

from "Athena, Greek Goddess of Wisdom and Craftsmanship"

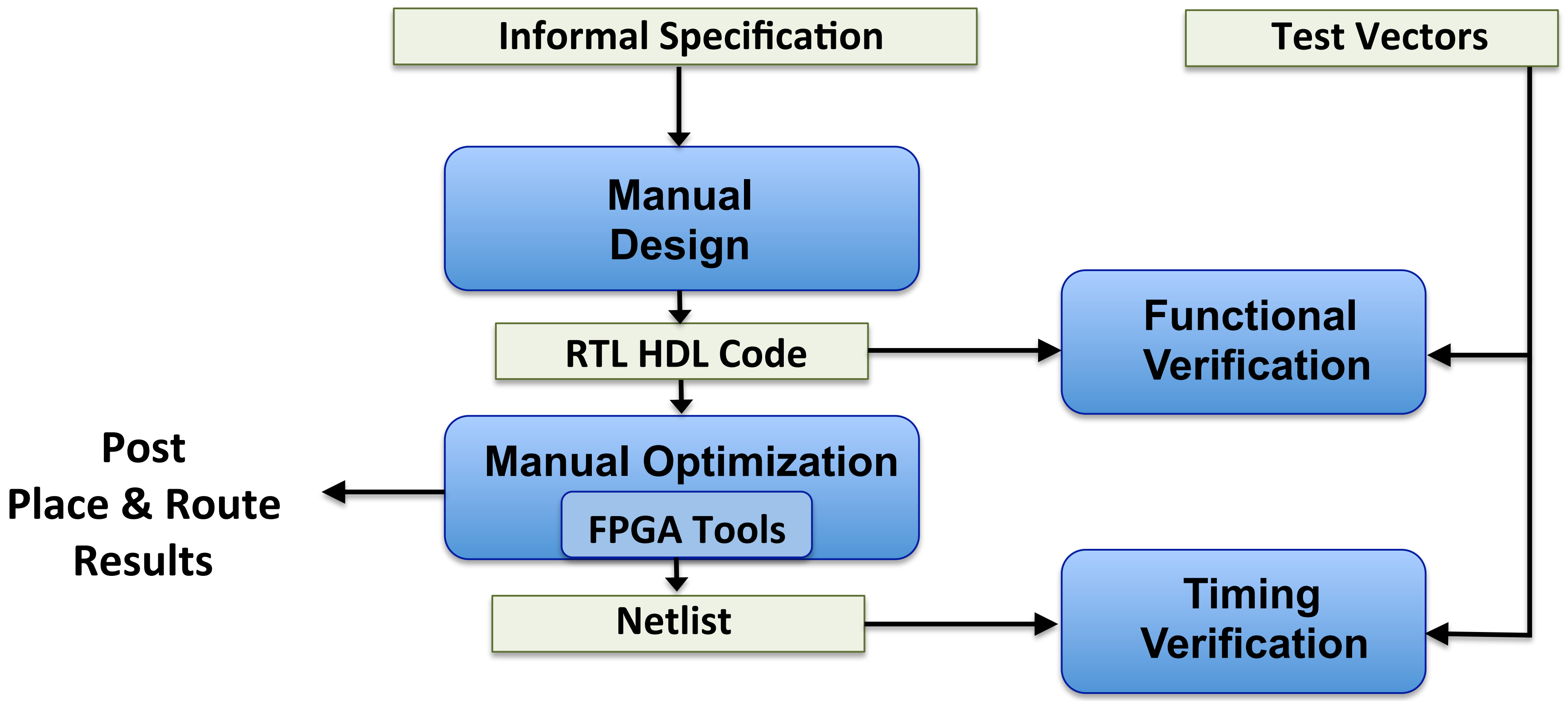
Generation of Results Facilitated by ATHENa

“working” with ATHENa…

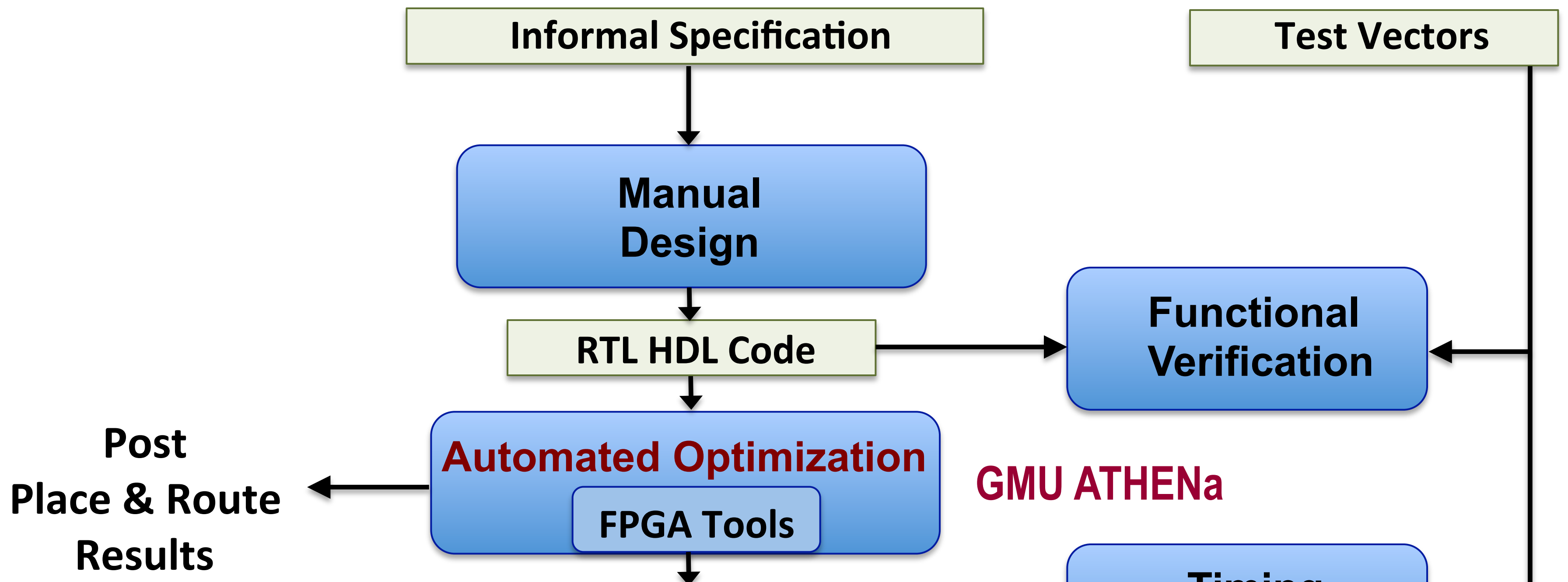
vs.

old days…

Traditional Development and Benchmarking Flow

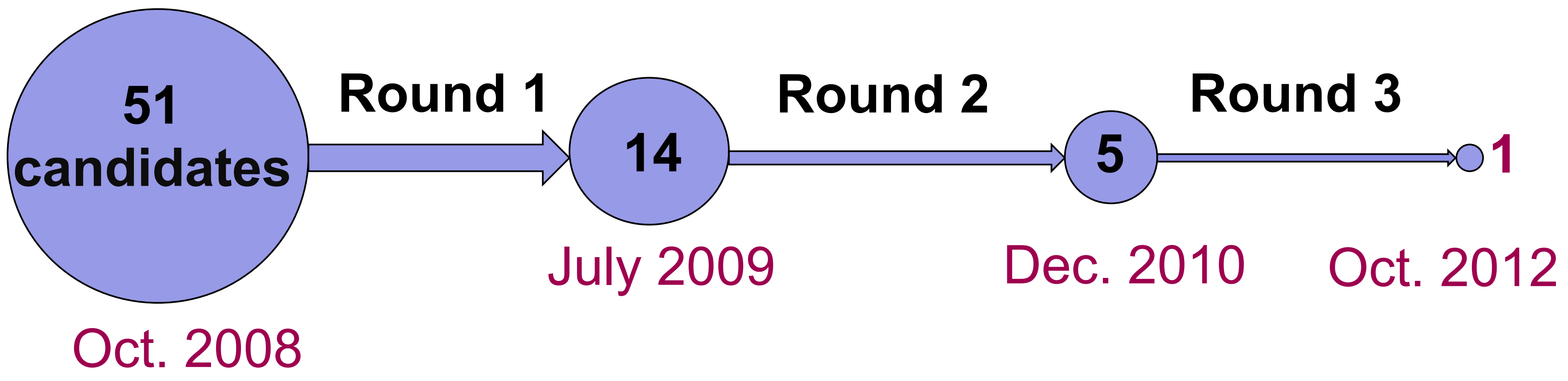


Extended Traditional Development and Benchmarking Flow



SHA-3 Contest 2007-2012

NIST SHA-3 Contest - Timeline



SHA-3 Round 2

Performance Metrics

Primary

1. Throughput

3. Throughput / Area

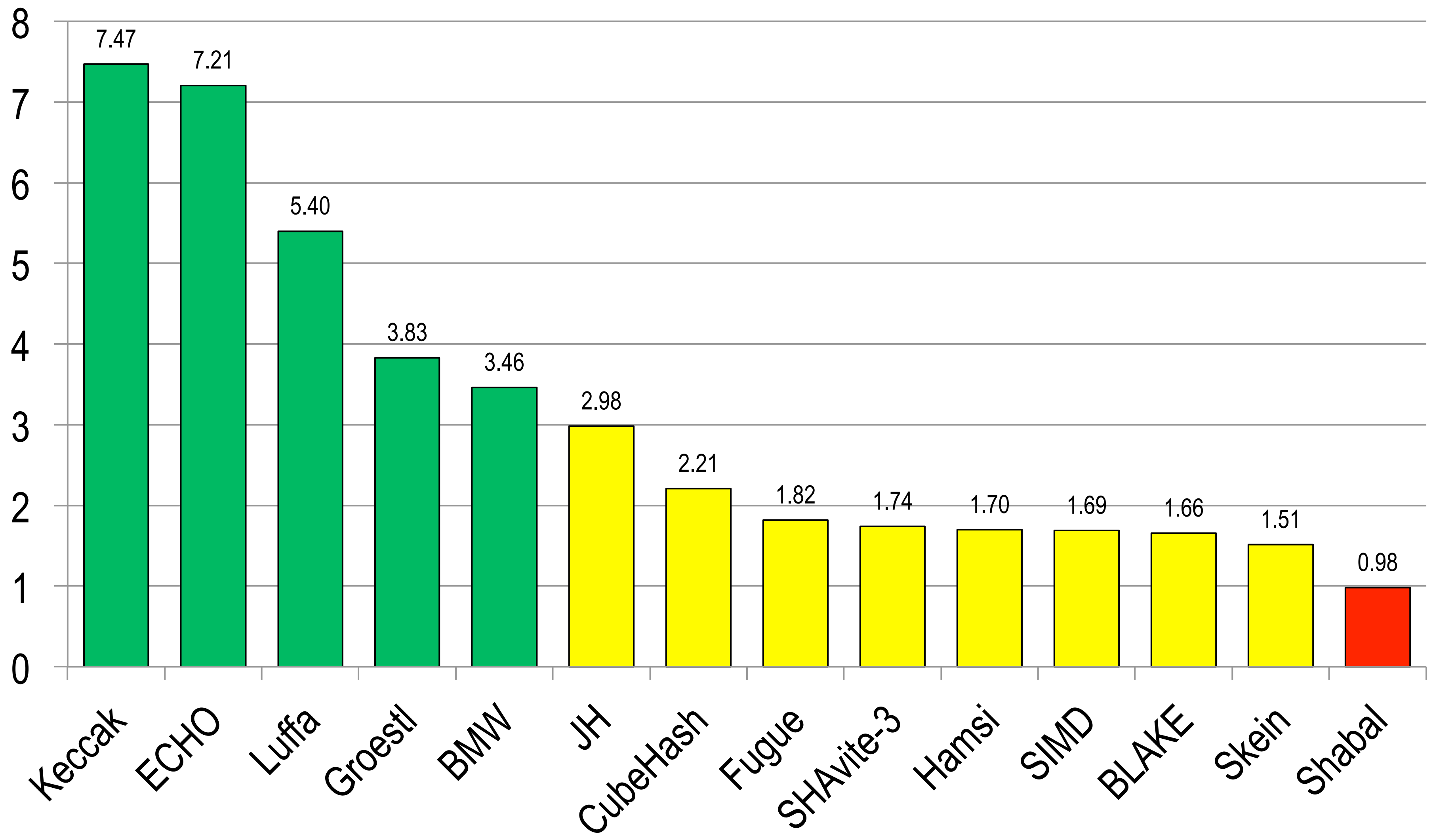
Secondary

2. Area

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

Overall Normalized Throughput: 256-bit variants of algorithms

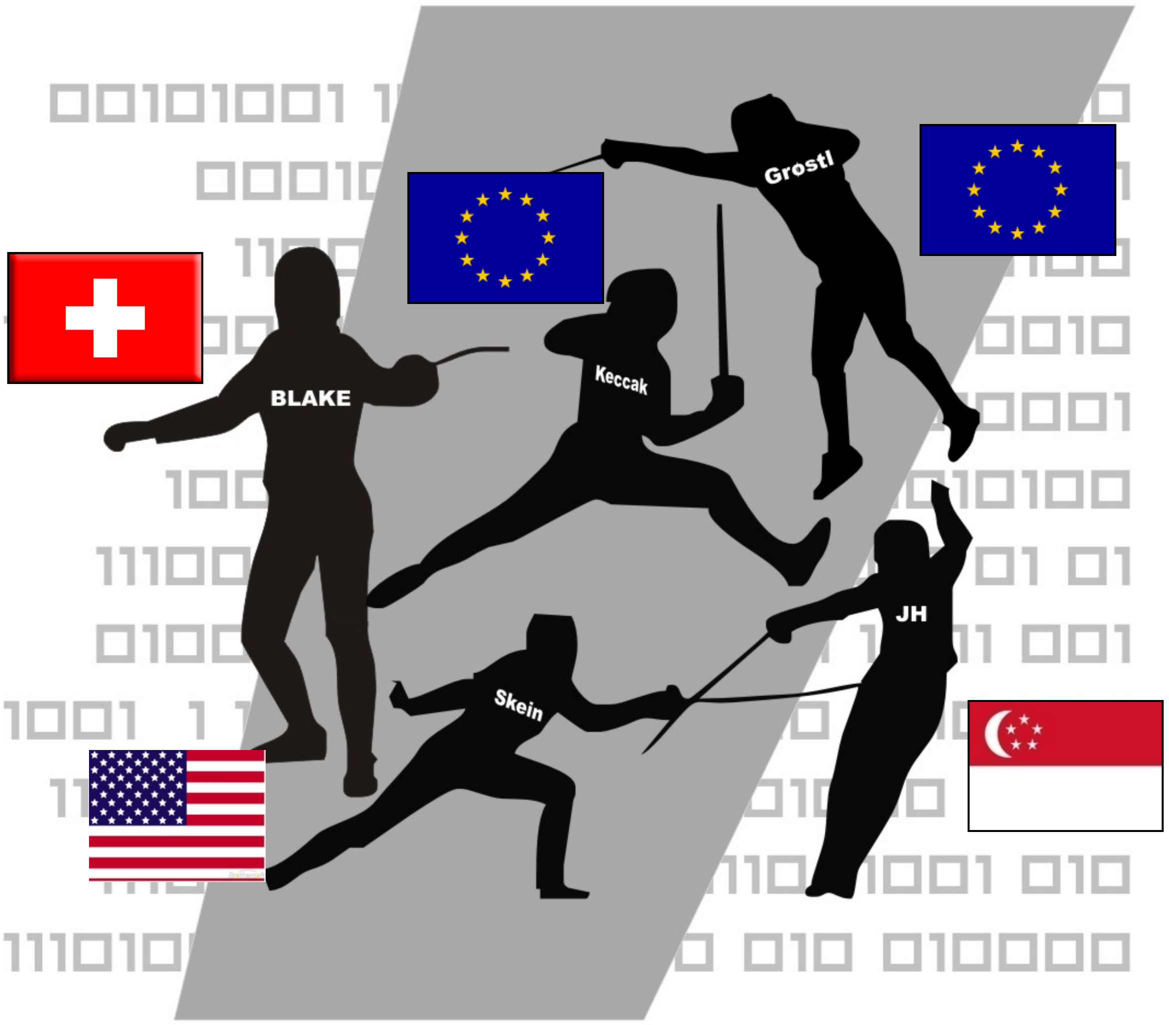
Normalized to SHA-256, Averaged over 10 FPGA families



	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	🟨	🟨	🟨	🟨	🟨	🟨	🟩	🟨

SHA-3 Round 3

SHA-3 Contest Finalists

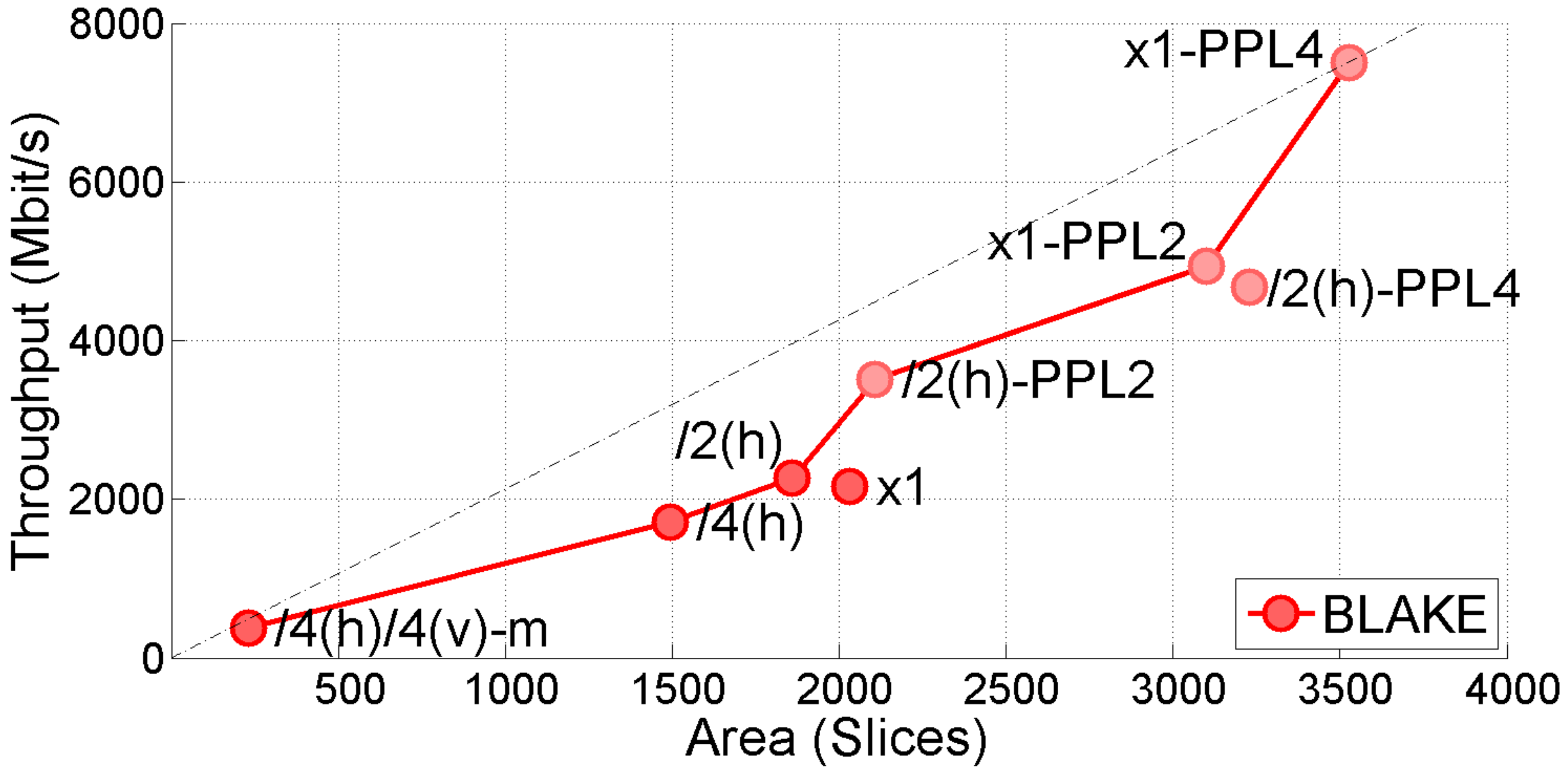


Benchmarking of the SHA-3 Finalists by CERG GMU

- 6 algorithms (BLAKE, Groestl, JH, Keccak, Skein, SHA-2)
- 2 variants (with a 256-bit and a 512-bit output)
- 7 to 12 different architectures per algorithm
- 4 modern FPGA families (Virtex 5, Virtex 6, Stratix III, Stratix IV)

Total: ~ 120 designs
~ 600+ results

BLAKE-256 in Virtex 5



x1 – basic iterative architecture

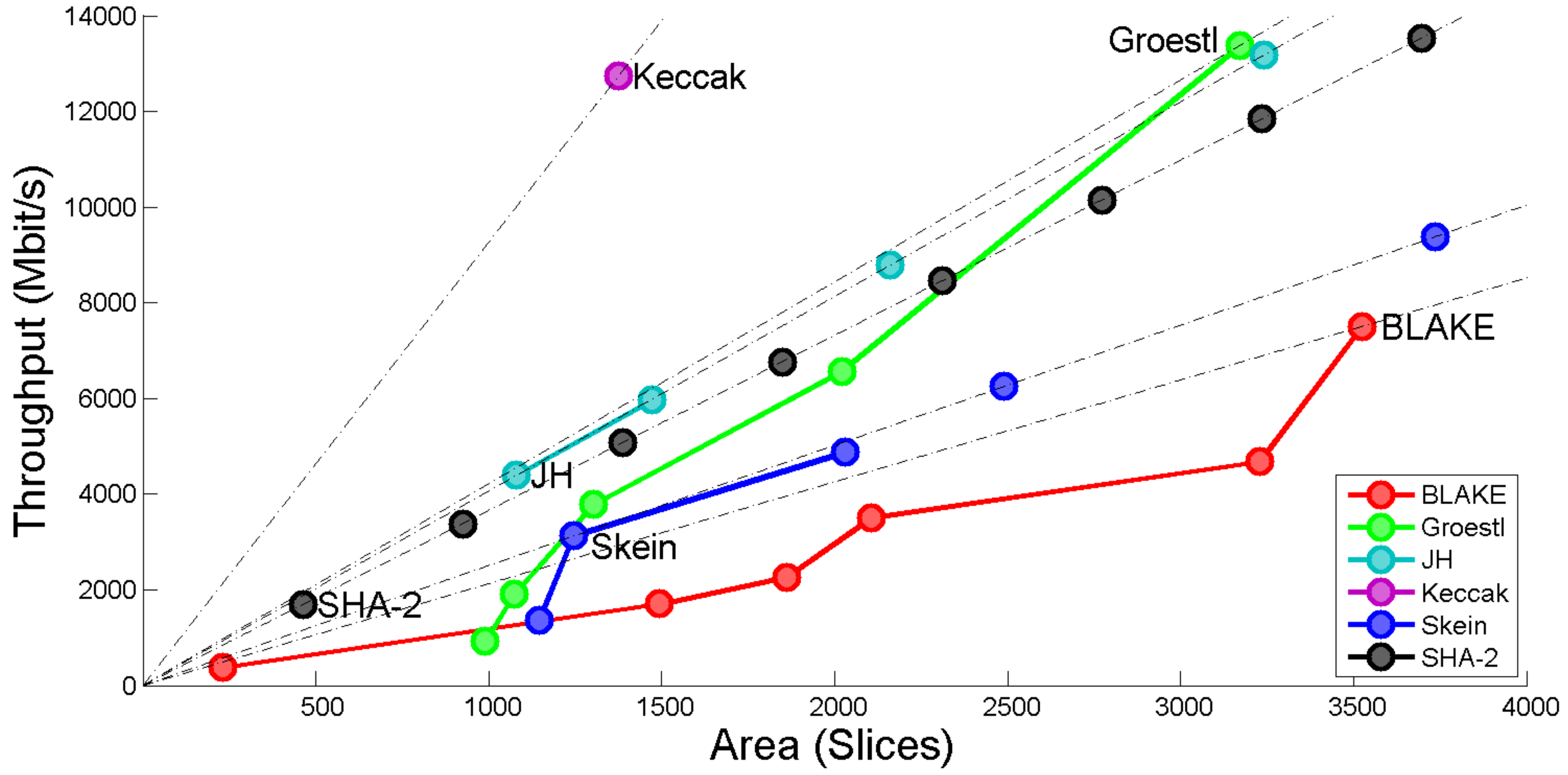
xk – unrolling by a factor of k

/k(h) – horizontal folding by a factor of k

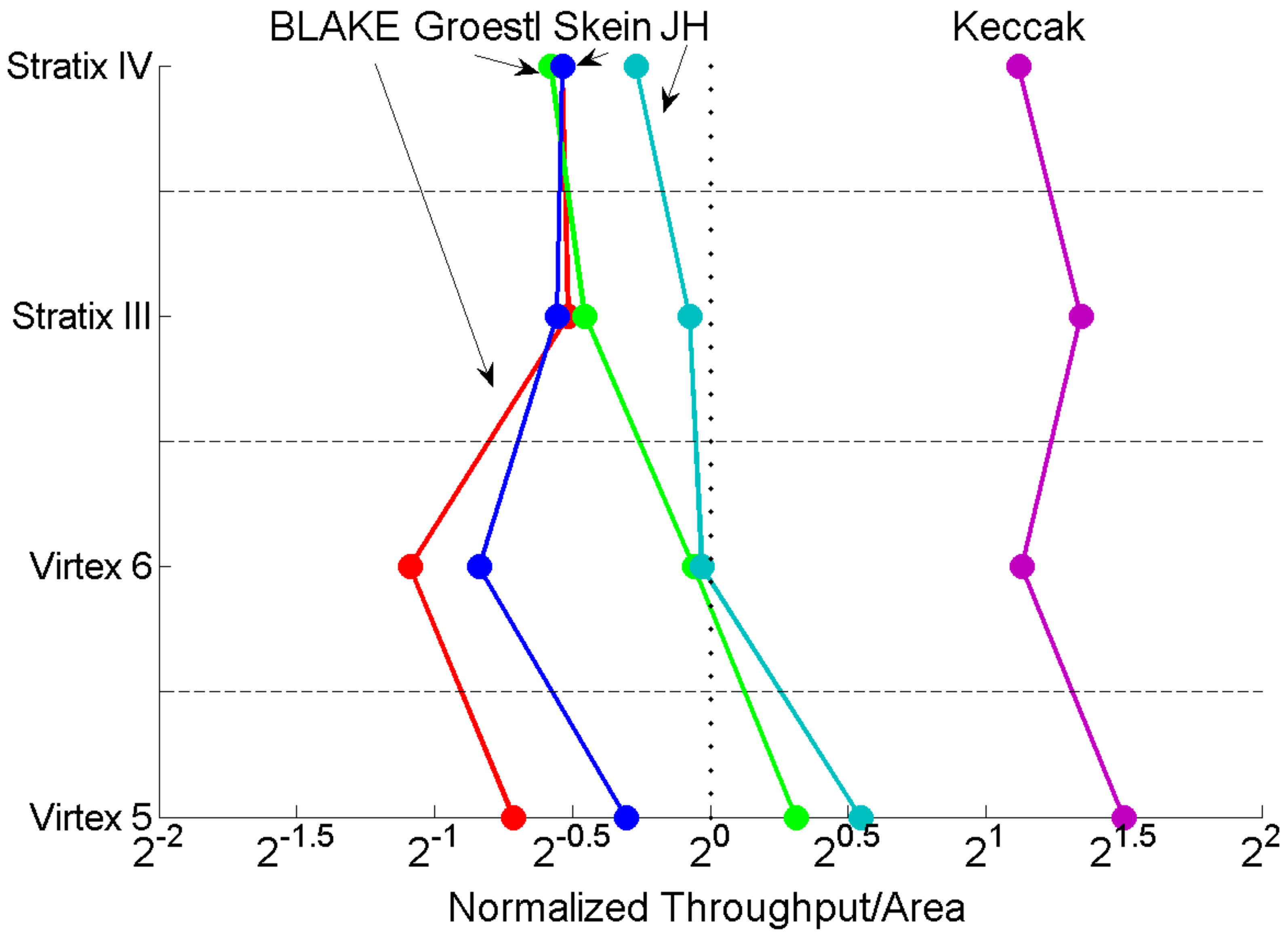
/k(v) – vertical folding by a factor of k

xk-PPLn – unrolling by a factor of k with n pipeline stages

256-bit variants in Virtex 5



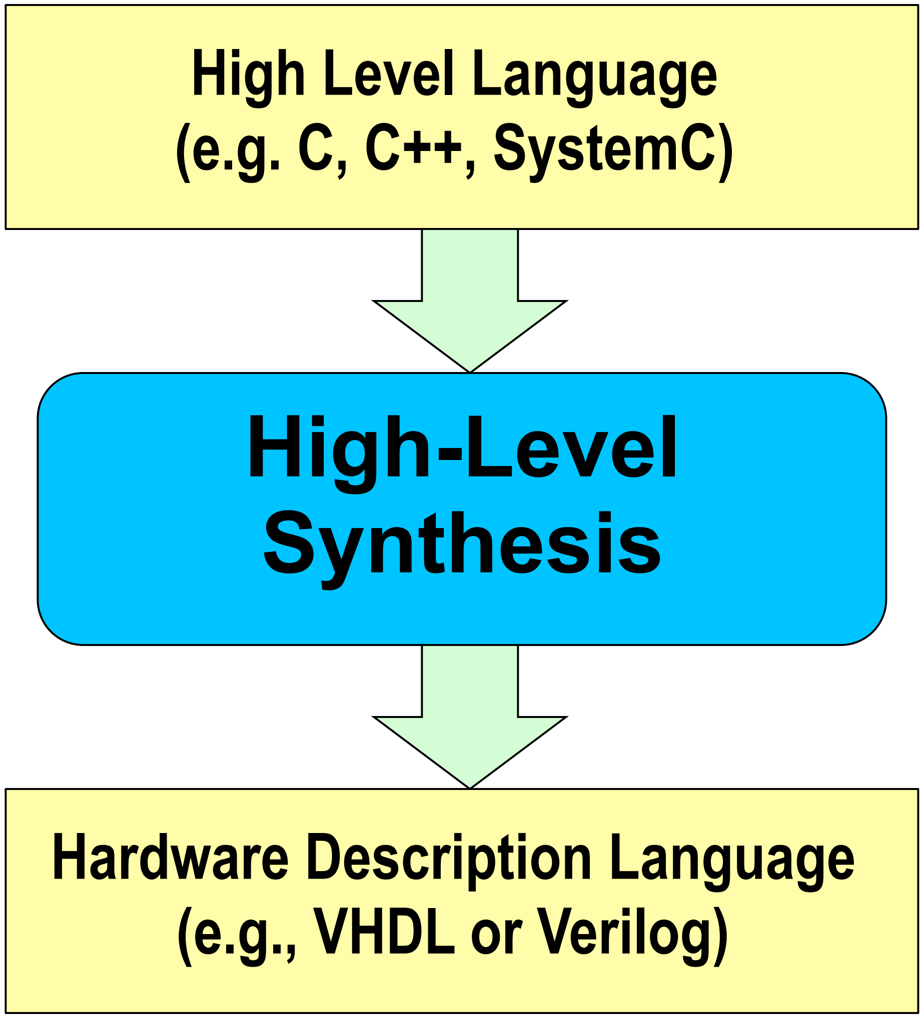
256-bit variants in 4 high-performance FPGA families



Remaining Difficulties of Hardware Benchmarking

- Large number of candidates
- Long time necessary to develop and verify
  RTL (Register-Transfer Level)
  Hardware Description Language (HDL) codes
- Multiple variants of algorithms
  (e.g., multiple key, nonce, and tag sizes)
- High-speed vs. lightweight algorithms
- Multiple hardware architectures
- Dependence on skills of designers

High-Level Synthesis (HLS)



Short History of High-Level Synthesis

G. Martin & G. Smith “HLS: Past, Present, and Future,” IEEE D&ToC, 2009

Generation 1 (1980s-early 1990s): research period

Generation 2 (mid 1990s-early 2000s):

- Commercial tools from Synopsys, Cadence, Mentor Graphics, etc.
- Input languages: behavioral HDLs Target: ASIC

Outcome: Commercial failure

Generation 3 (from early 2000s):

- Domain oriented commercial tools: in particular for DSP
- Input languages: C, C++, C-like languages (Impulse C, Handel C, etc.), Matlab + Simulink, Bluespec
- Target: FPGA, ASIC, or both

Outcome: First success stories

Cinderella Story

AutoESL Design Technologies, Inc. **(25 employees)**

Flagship product:

AutoPilot, translating **C/C++/SystemC** to **VHDL or Verilog**

- **Acquired by the biggest FPGA company, Xilinx Inc., in 2011**
- **AutoPilot integrated into the primary Xilinx toolset, Vivado, as Vivado HLS, released in 2012**

"High-Level Synthesis for the Masses"

Our Hypotheses

- Ranking of candidate algorithms in cryptographic contests in terms of their performance in modern FPGAs & All-Programmable SoCs will remain the same independently whether the HDL implementations are *developed manually* or *generated automatically* using High-Level Synthesis tools

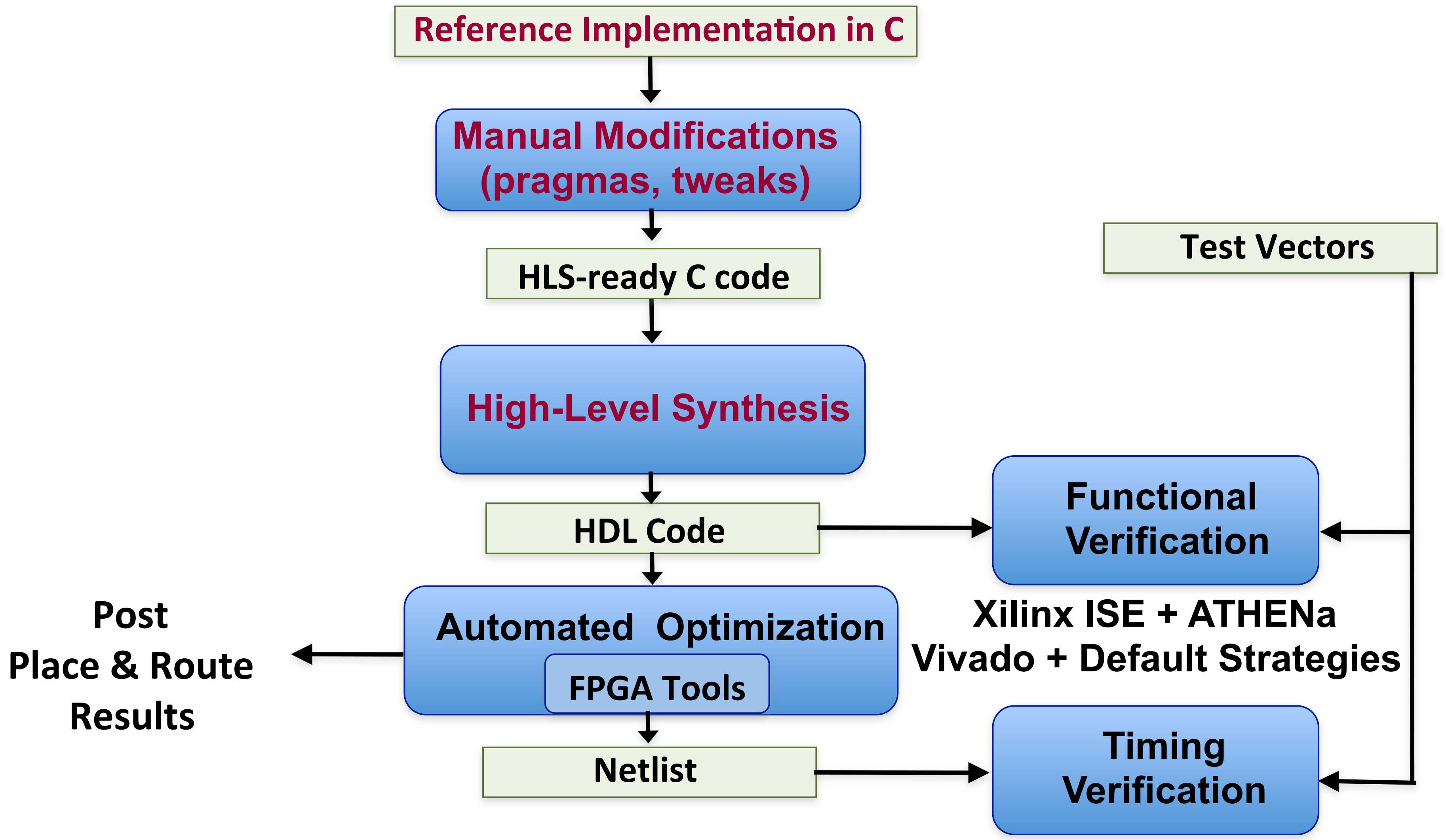
- The development time will be reduced by at least an order of magnitude

Potential Additional Benefits

Early feedback for designers of cryptographic algorithms

- Typical design process based only on security analysis and software benchmarking
- Lack of immediate feedback on hardware performance
- Common unpleasant surprises, e.g.,
  - Mars in the AES Contest
  - BMW, ECHO, and SIMD in the SHA-3 Contest

Proposed HLS-Based Development and Benchmarking Flow



Examples of Source Code Modifications

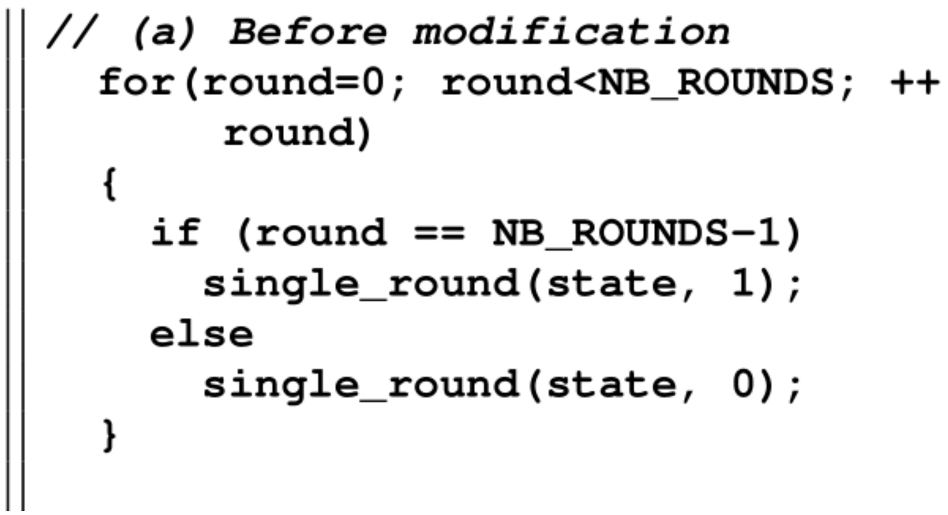
Unrolling of loops:

```
for (i = 0; i < 4; i ++)
#pragma HLS UNROLL
     for (j = 0; j < 4; j ++)
#pragma HLS UNROLL
         b[i][j] = s[i][j];
```

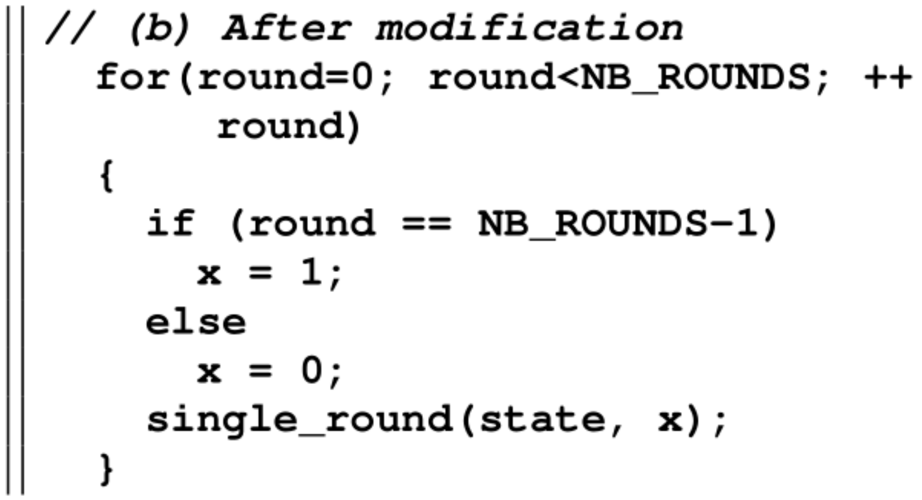
Flattening function's hierarchy:

```
void KeyUpdate (word8 k[4][4],
                word8 round)
{
 #pragma HLS INLINE
          ...
}
```

Function Reuse:



```
// (a) Before modification
  for(round=0; round<NB_ROUNDS; ++
       round)
  {
    if (round == NB_ROUNDS-1)
      single_round(state, 1);
    else
      single_round(state, 0);
  }
```



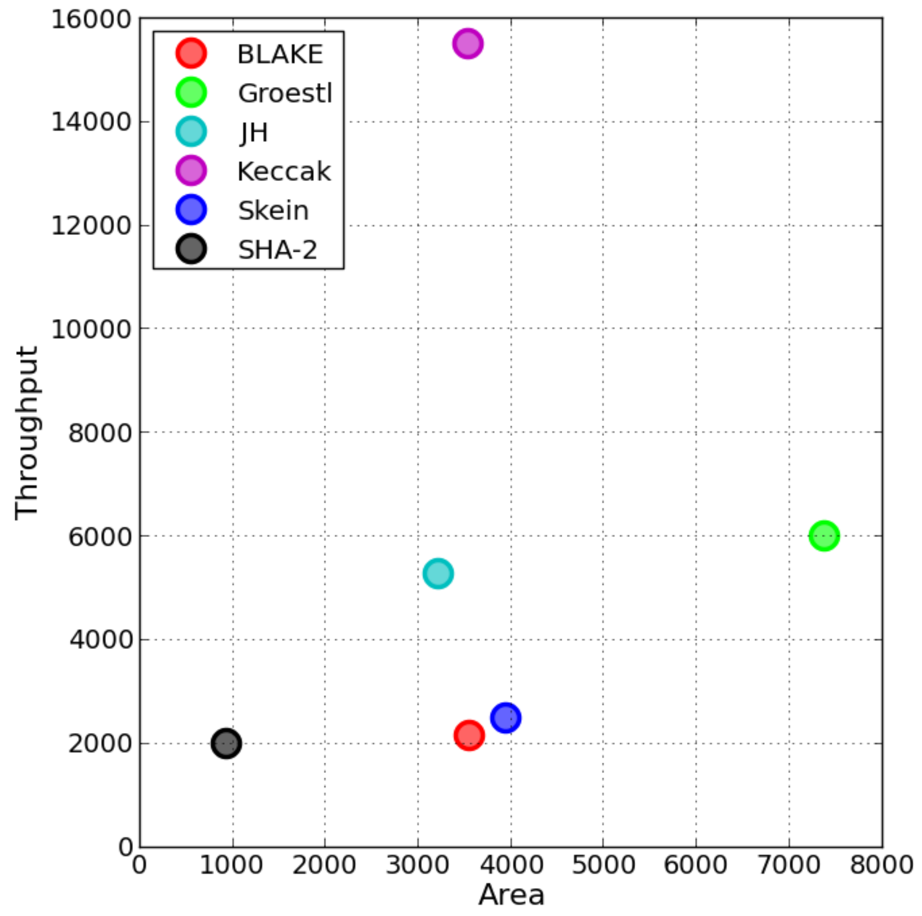
```
// (b) After modification
  for(round=0; round<NB_ROUNDS; ++
       round)
  {
    if (round == NB_ROUNDS-1)
      x = 1;
    else
      x = 0;
    single_round(state, x);
  }
```

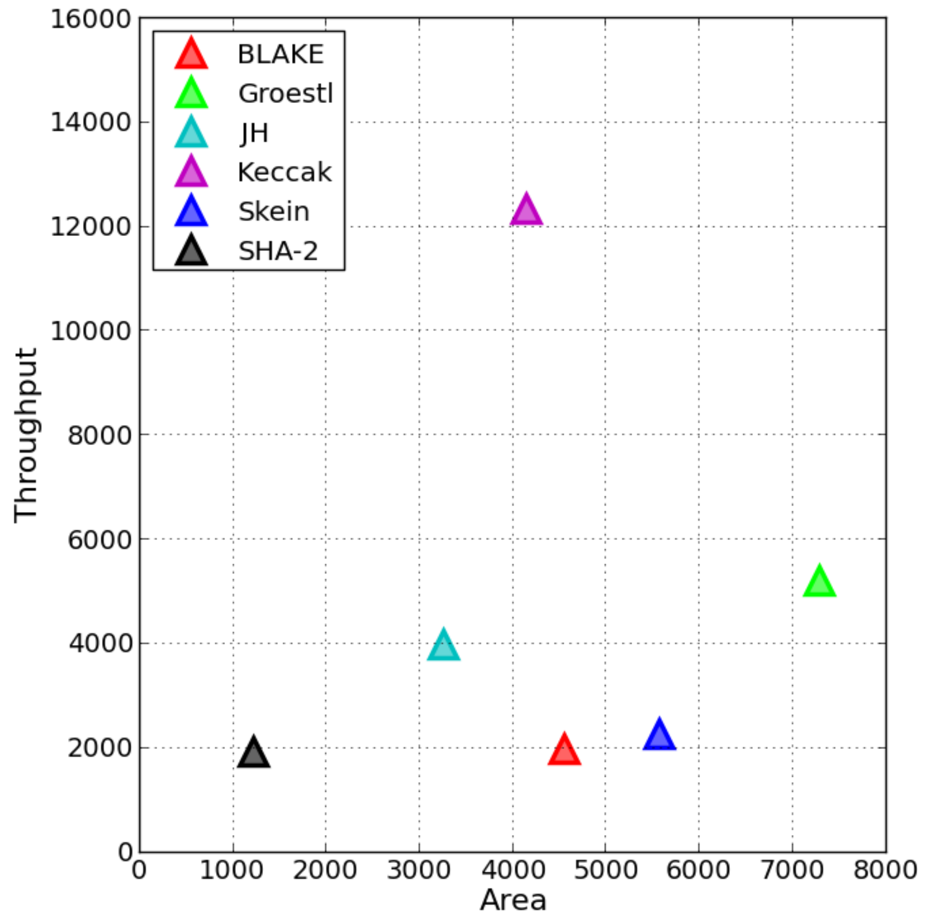
Our Test Case 1

- 5 final SHA-3 candidates + old standard SHA-2
- Most efficient sequential architectures

  (/2h for BLAKE, x4 for Skein, x1 for others)
- GMU VHDL codes developed during SHA-3 contest
- Reference software implementations in C included in the submission packages

Manual RTL vs. HLS-based Results: Altera Stratix III

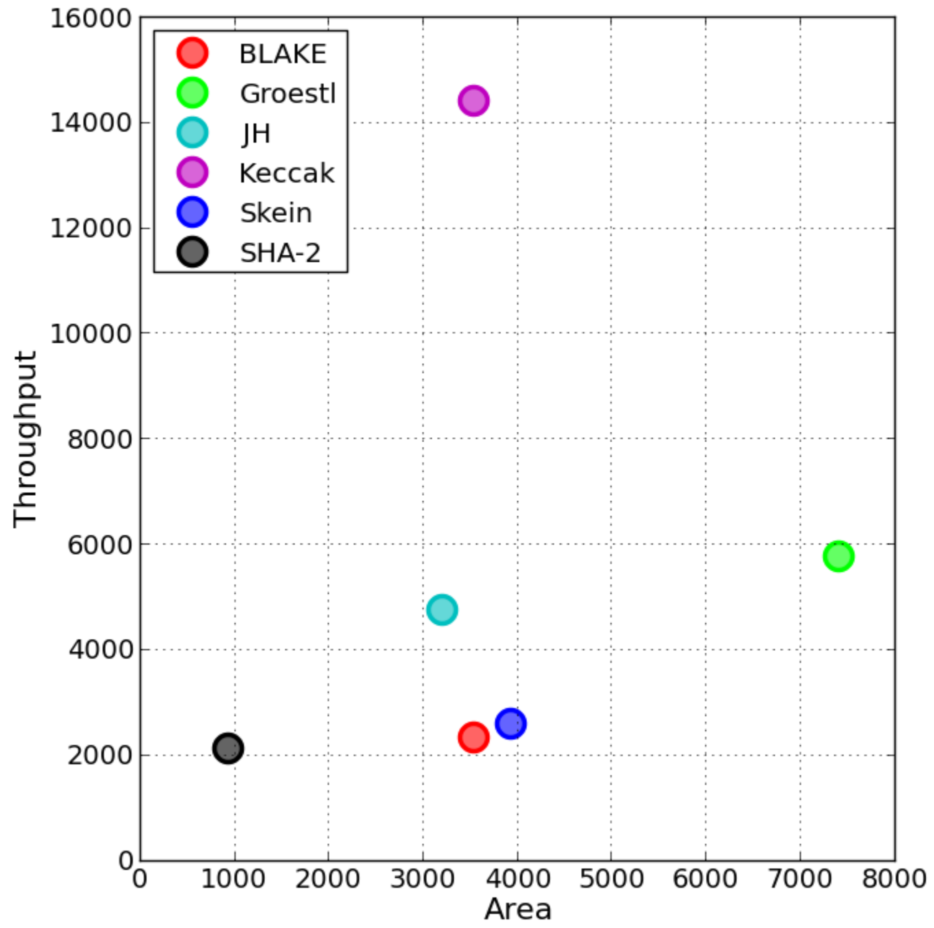


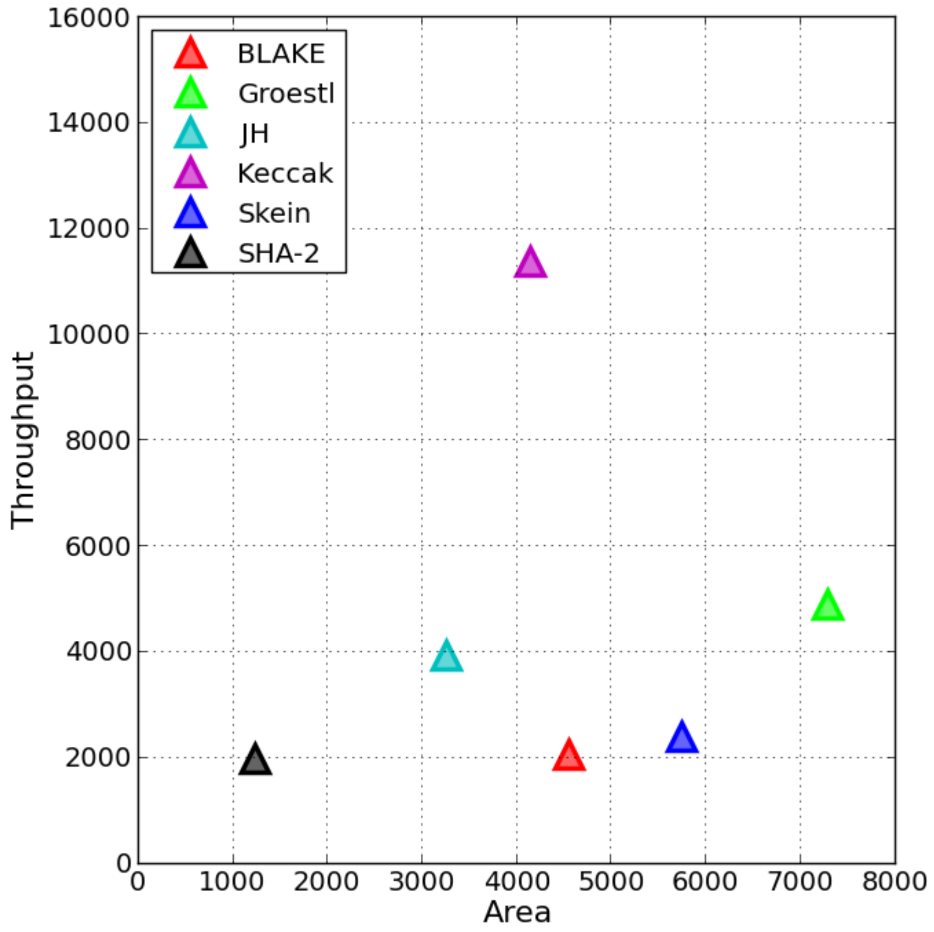


RTL

HLS

Manual RTL vs. HLS-based Results: Altera Stratix IV

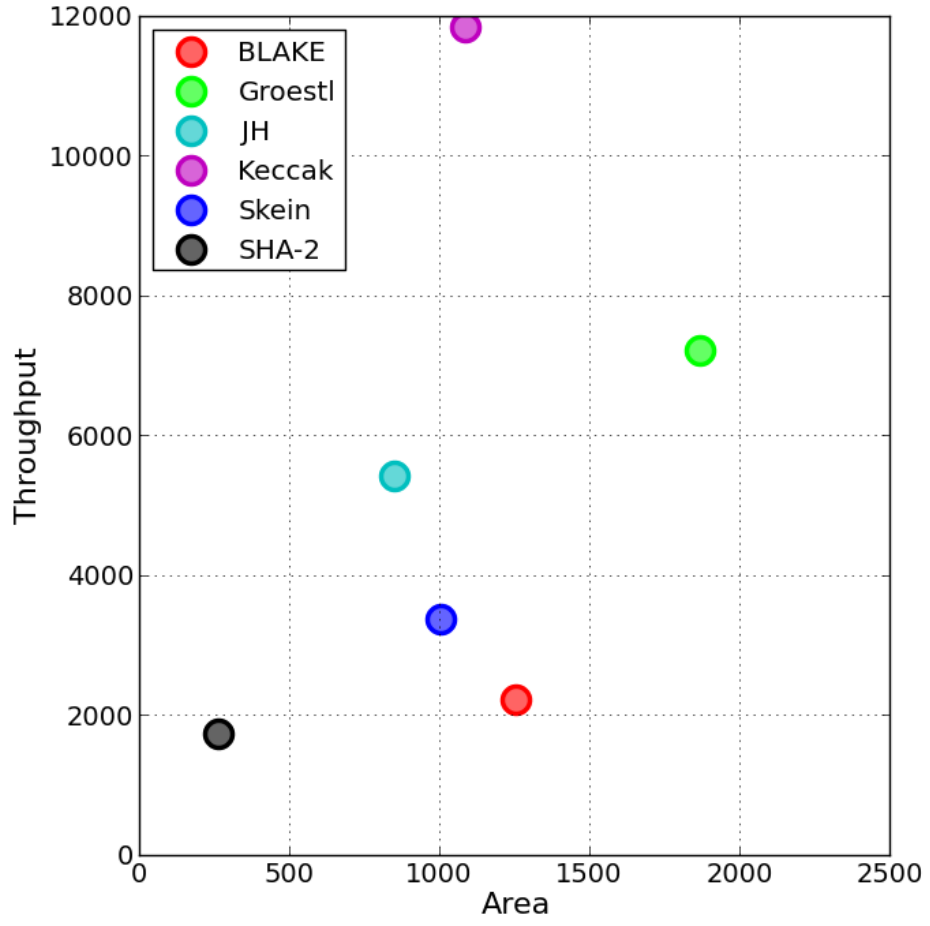


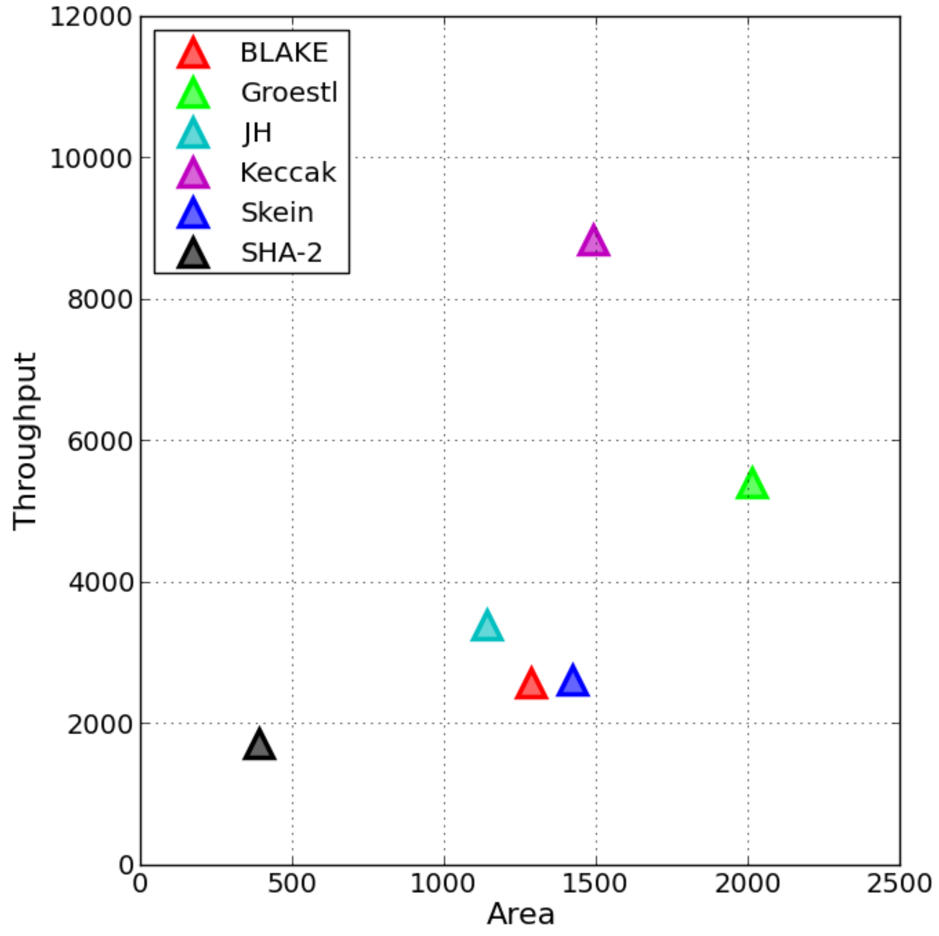


RTL

HLS

Lack of Correlation for Xilinx Virtex 6

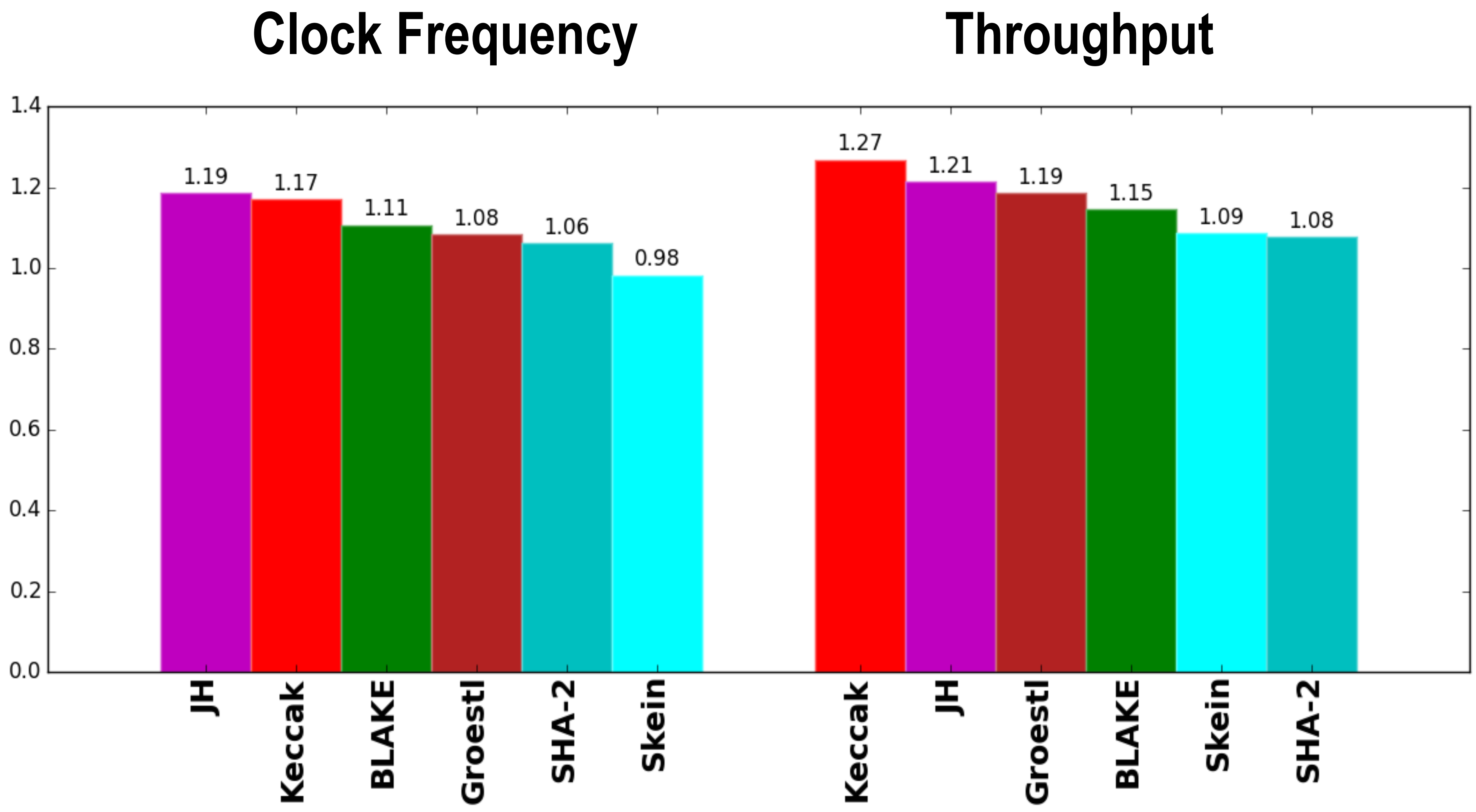




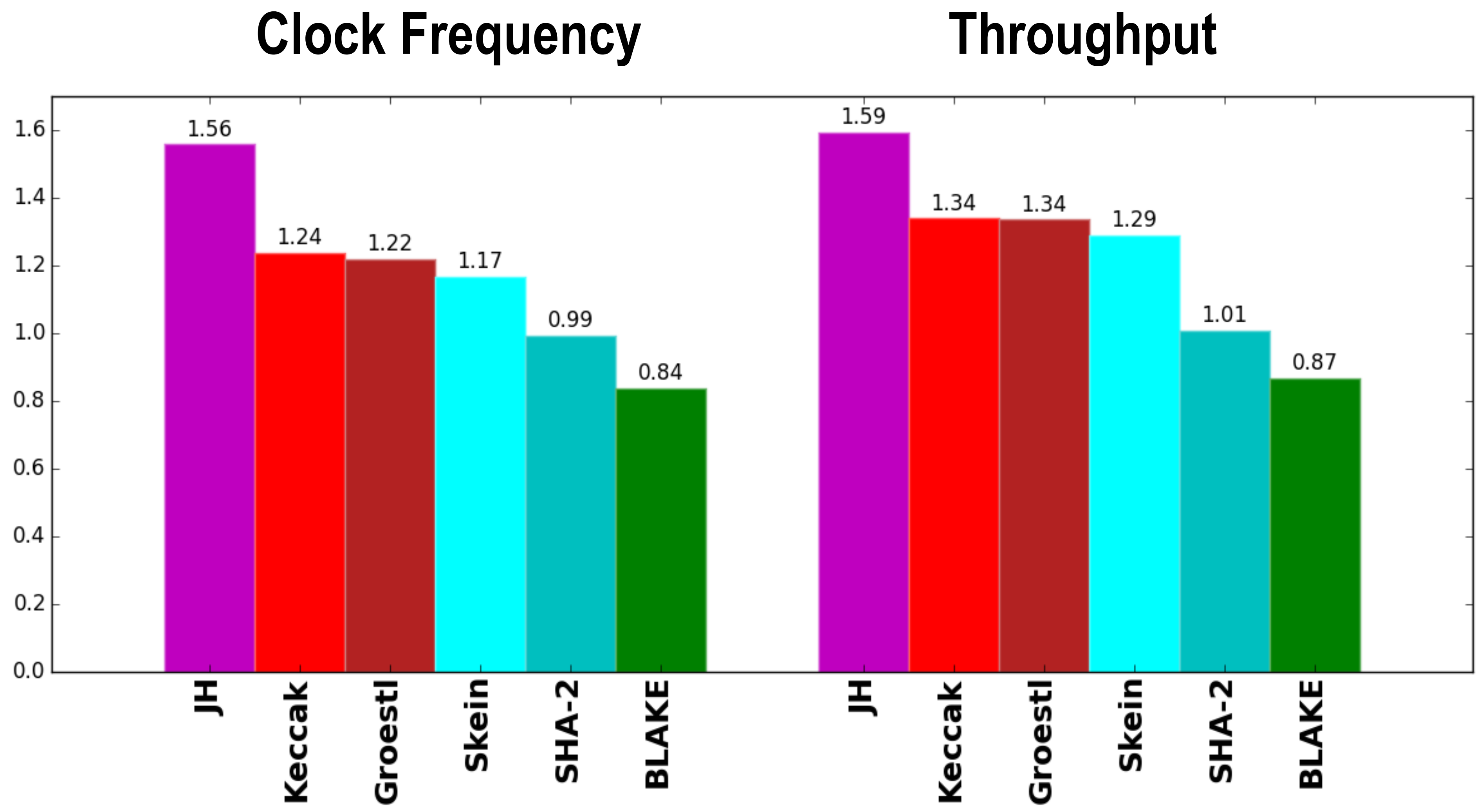
RTL

HLS

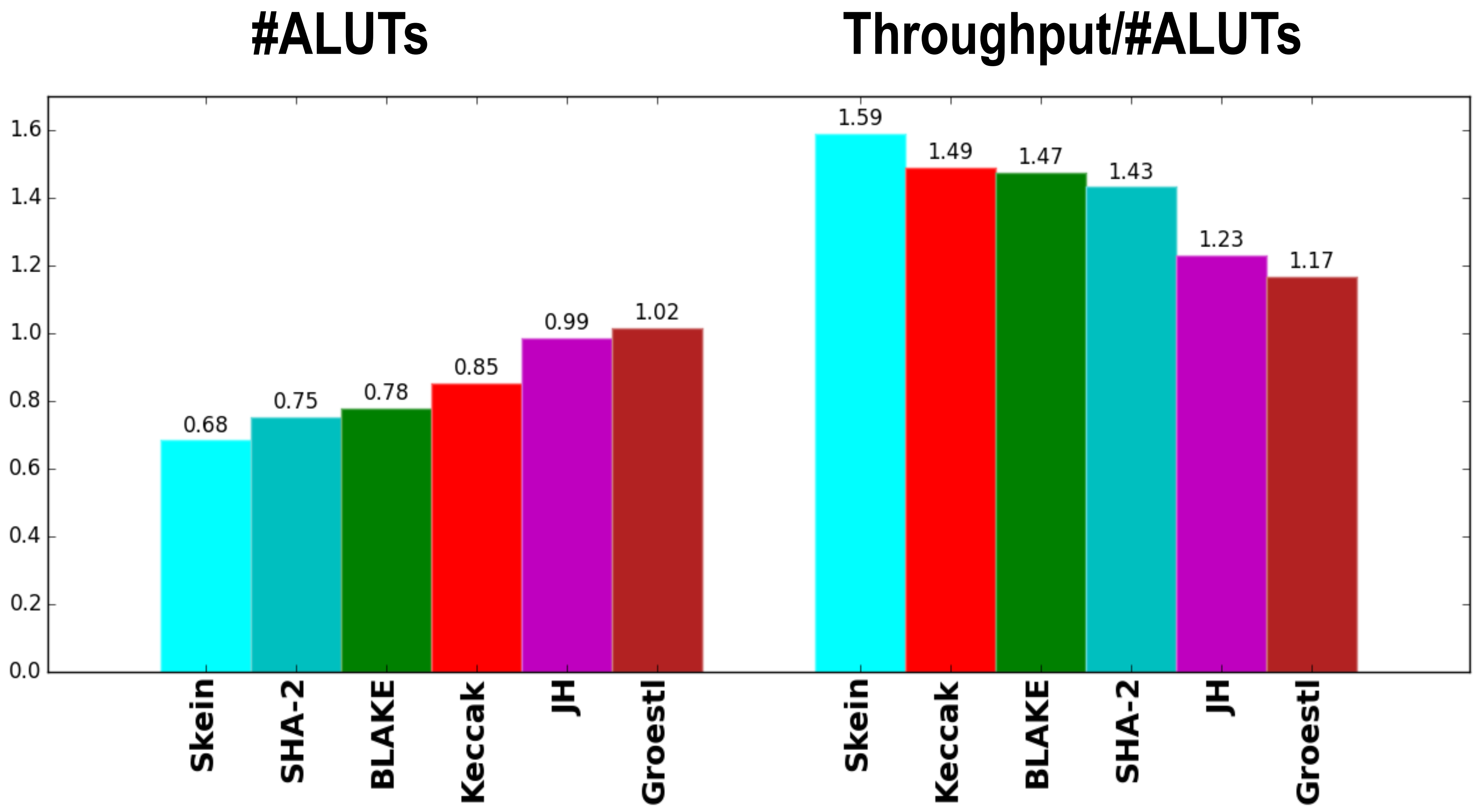
RTL vs. HLS Ratios in Altera Stratix IV



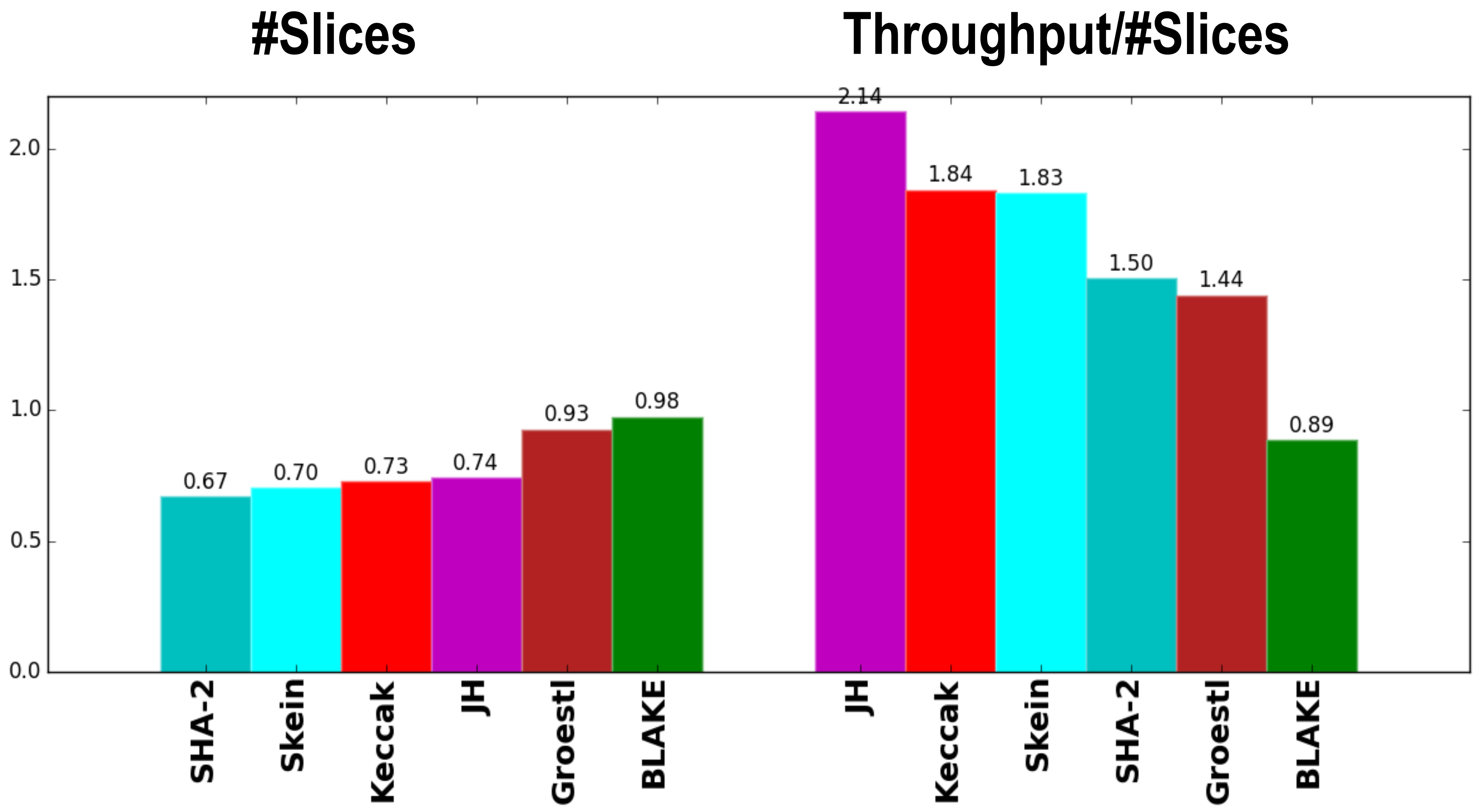
RTL vs. HLS Ratios in Xilinx Virtex 6



RTL vs. HLS Ratios in Altera Stratix IV



RTL vs. HLS Ratios in Xilinx Virtex 6



Hypothesis Check

Hypothesis I:

- Ranking of candidates in terms of throughput, area, and throughput/ area ratio will remain the same

  TRUE for Altera Stratix III, Stratix IV

  FALSE for Xilinx Virtex 5, Virtex 6, and Virtex 7

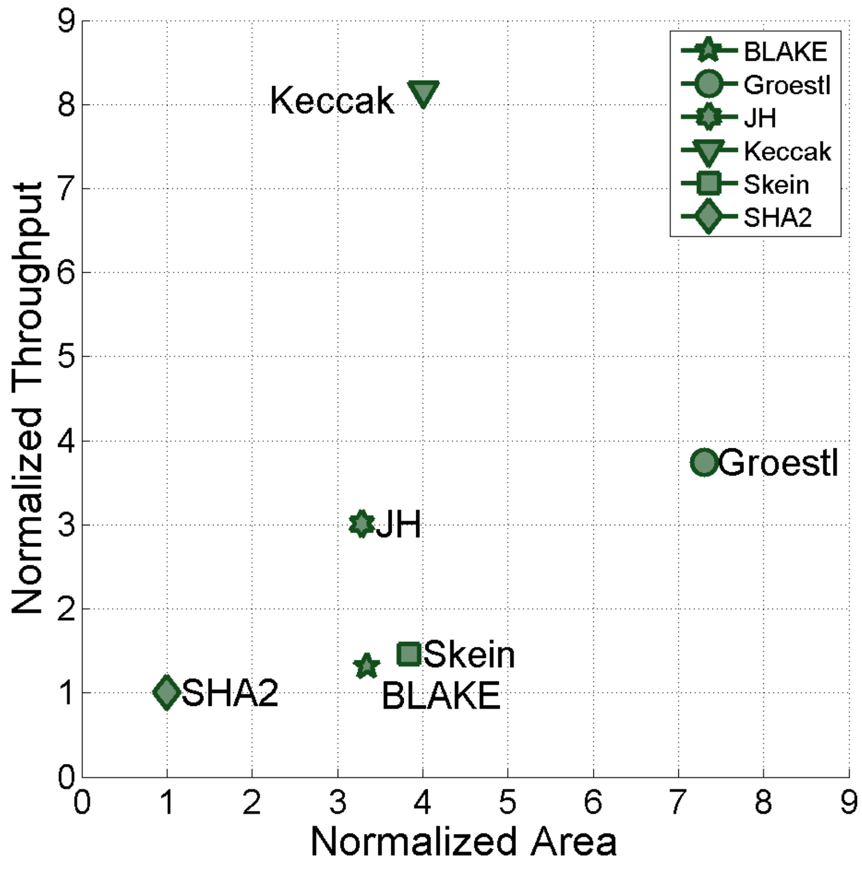
Hypothesis II:

- Performance ratios HDL/HLS similar across candidates

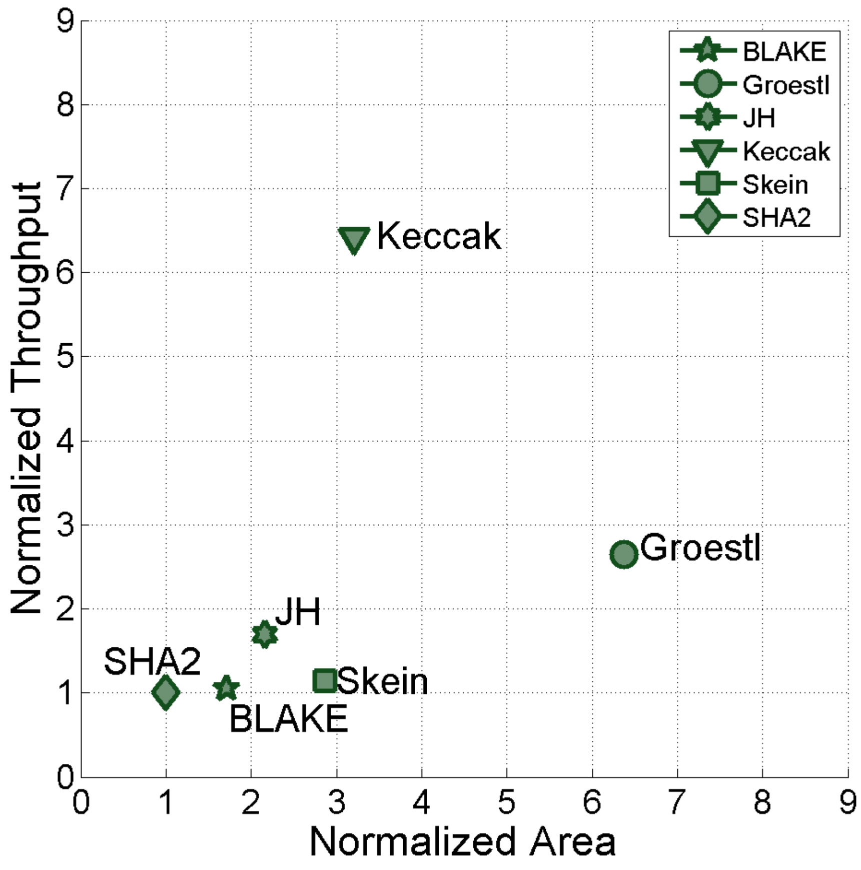
	Stratix III	Stratix IV
Frequency	0.99-1.30	0.98-1.19
Area	0.71-1.01	0.68-1.02
Throughput	1.10-1.33	1.08-1.27
Throughput/ Area	1.14-1.55	1.17-1.59

Correlation Between Altera FPGA Results and ASICs

Stratix III FPGA

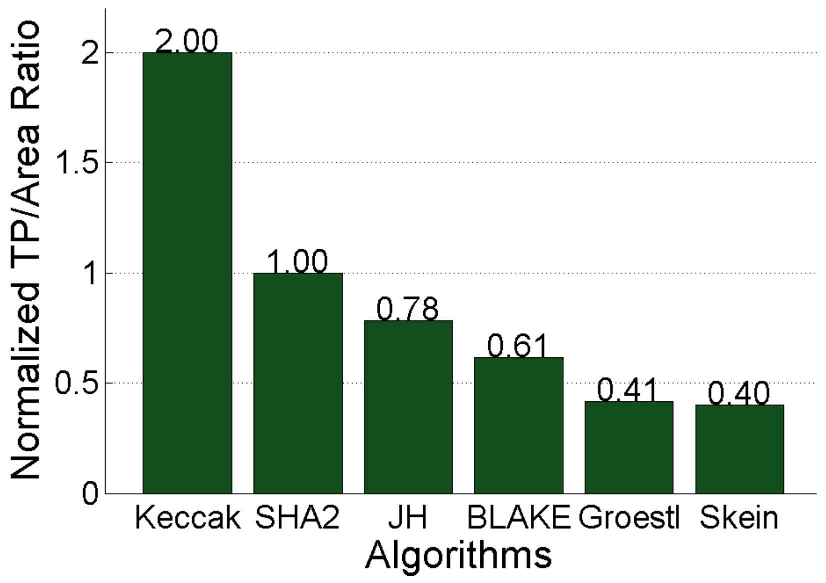


ASIC

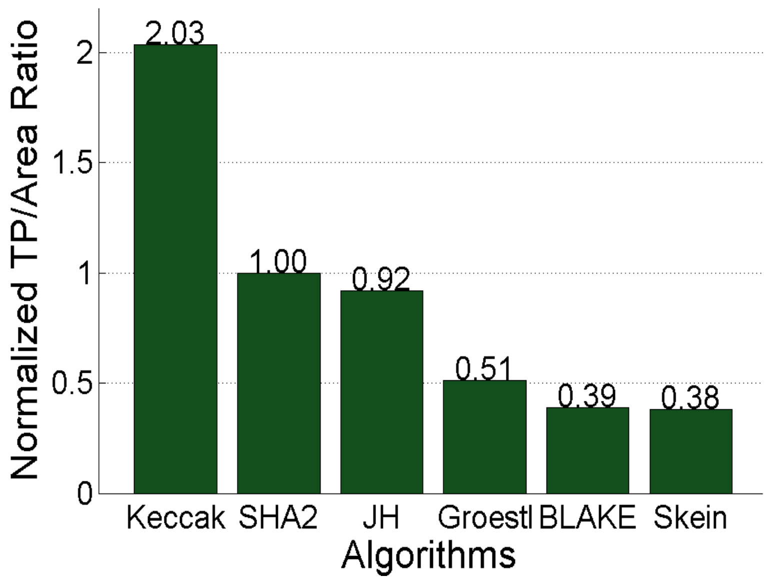


Correlation Between ASIC Results and FPGA Results

ASIC



Stratix III FPGA



CAESAR Contest 2013-2017

CAESAR Competition

Goal: Portfolio of new-generation authenticated ciphers

Period: March 2015 - December 2017 (tentative)

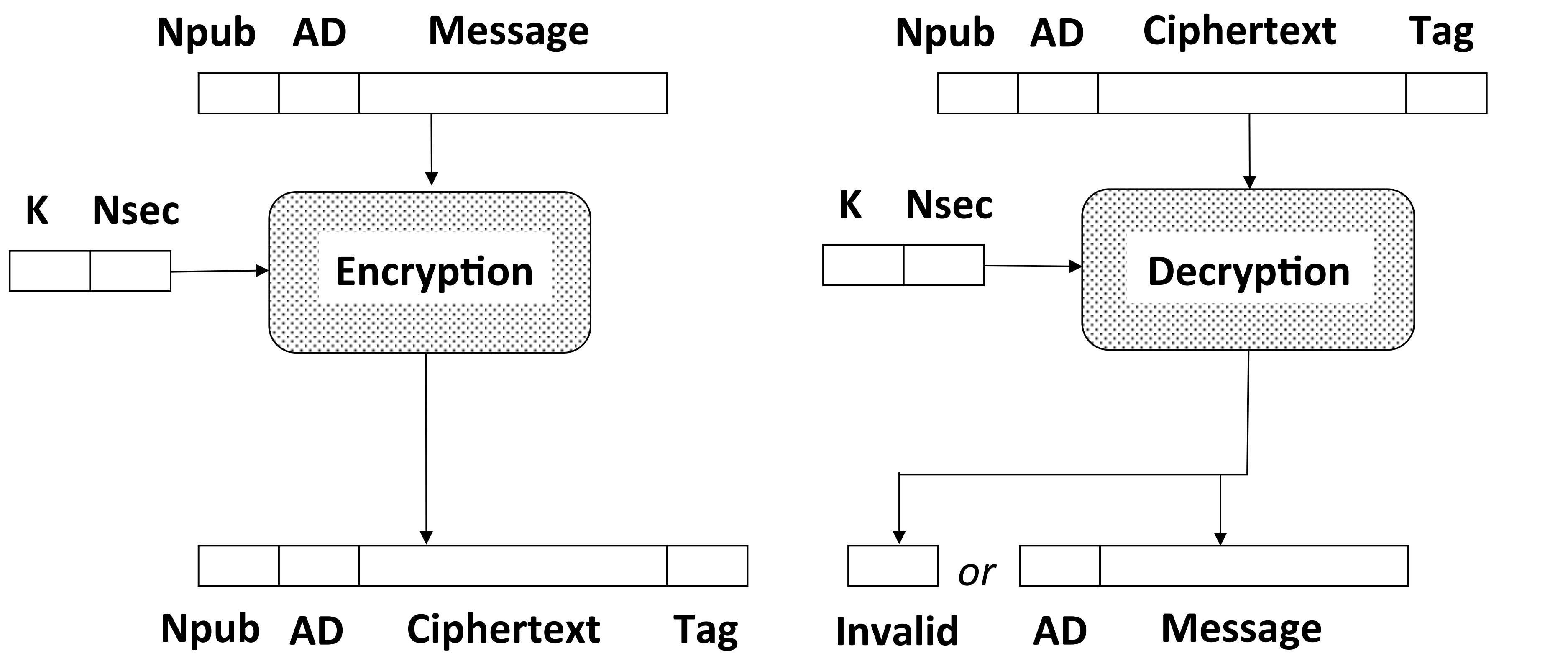
Organizer: An informal committee of leading cryptographic experts

Number of submitted candidates: 57

Upcoming milestones:

- Announcement of second-round candidates
- Round 2 tweaks
- VHDL/Verilog codes

Input and Output of an Authenticated Cipher



K - Secret key
Npub (Public Message Number), typically Nonce
Nsec (Secret Message Number) [supported by few algorithms]
AD – Associated Data

Our Test Case 2

- 8 Round 1 CAESAR candidates + current standard AES-GCM
- Basic iterative architecture
- GMU AEAD Hardware API
- Implementations developed in parallel using RTL and HLS methodology
- 2-3 RTL implementations per student, all HLS implementations developed by a single student (Ice)
- Starting point: Informal specifications and reference software implementations in C provided by the algorithm authors
- Post P&R results generated for
  - Xilinx Virtex 6 using Xilinx ISE + ATHENa, and
  - Virtex 7 and Zynq 7000 using Xilinx Vivado with 26 default option optimization strategies
- No use of BRAMs or DSP Units in AEAD Core

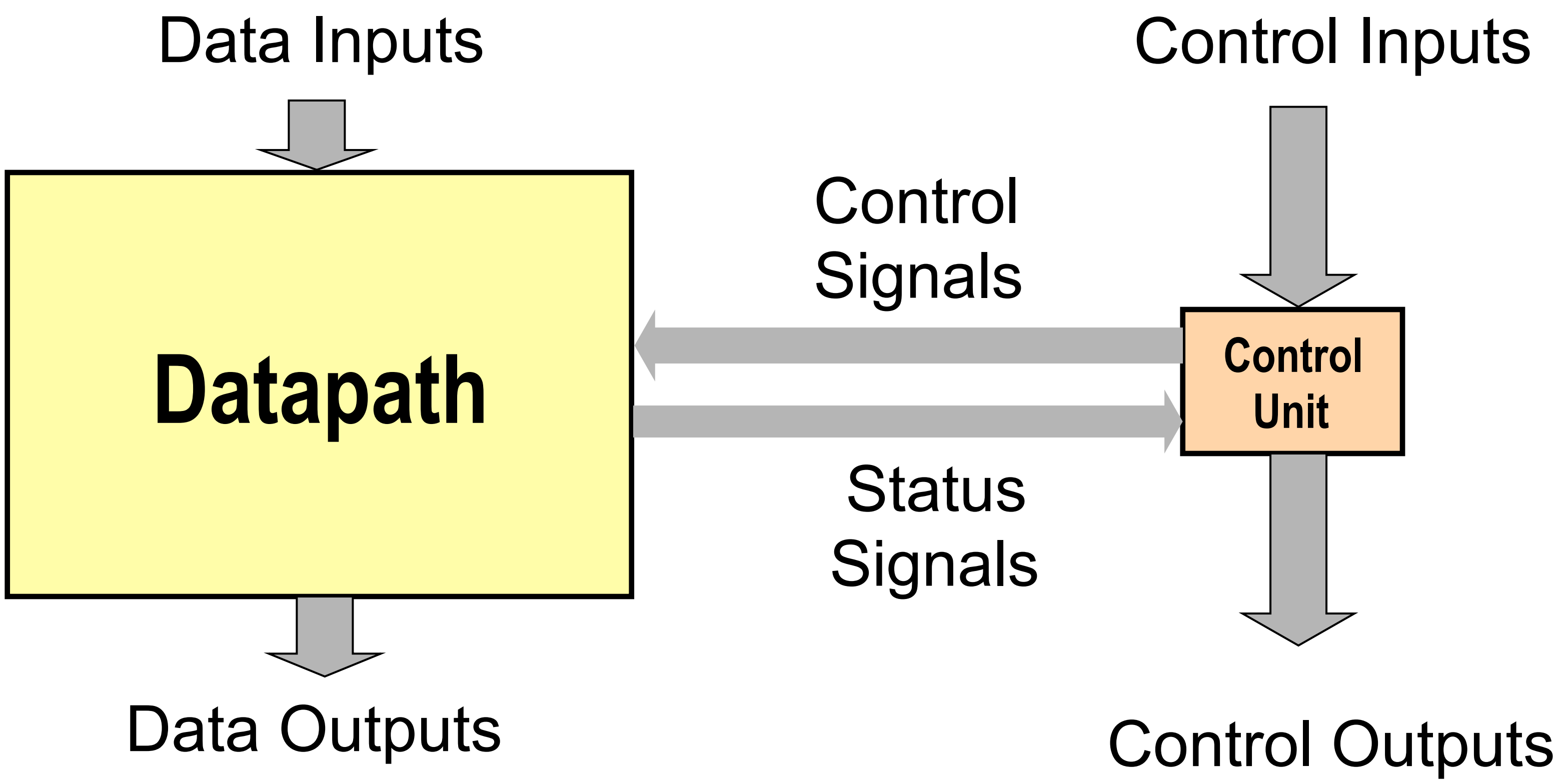
Parameters of Authenticated Ciphers

Algorithm	Key size	Nonce size	Tag size	Basic Primitive
		Block Cipher Based		
AES-COPA	128	128	128	AES
AES-GCM	128	96	128	AES
CLOC	128	96	128	AES
POET	128	128	128	AES
SCREAM	128	96	128	TLS
		Permutation Based		
ICEPOLE	128	128	128	Keccak-like
Keyak	128	128	128	Keccak-f
PRIMATEs-GIBBON	120	120	120	PRIMATE
PRIMATEs-HANUMAN	120	120	120	PRIMATE

Parameters of Ciphers & GMU Implementations

Algorithm	Word Size, w	Block Size, b	#Rounds	Cycles/Block RTL	Cycles/Block HLS
		Block-cipher Based			
AES-COPA	32	128	10	11	12
AES-GCM	32	128	10	11	12
CLOC	32	128	10	11	12
POET	32	128	10	11	12
SCREAM	32	128	10	11	12
		Permutation Based			
ICEPOLE	256	1024	6	6	8
Keyak	128	1344	12	12	14
PRIMATEs-GIBBON	40	40	6	7	8
PRIMATEs-HANUMAN	40	40	12	13	14

Datapath vs. Control Unit



Determines
- Area
- Clock Frequency

Determines
- Number of clock cycles

Encountered Problems

Control Unit suboptimal

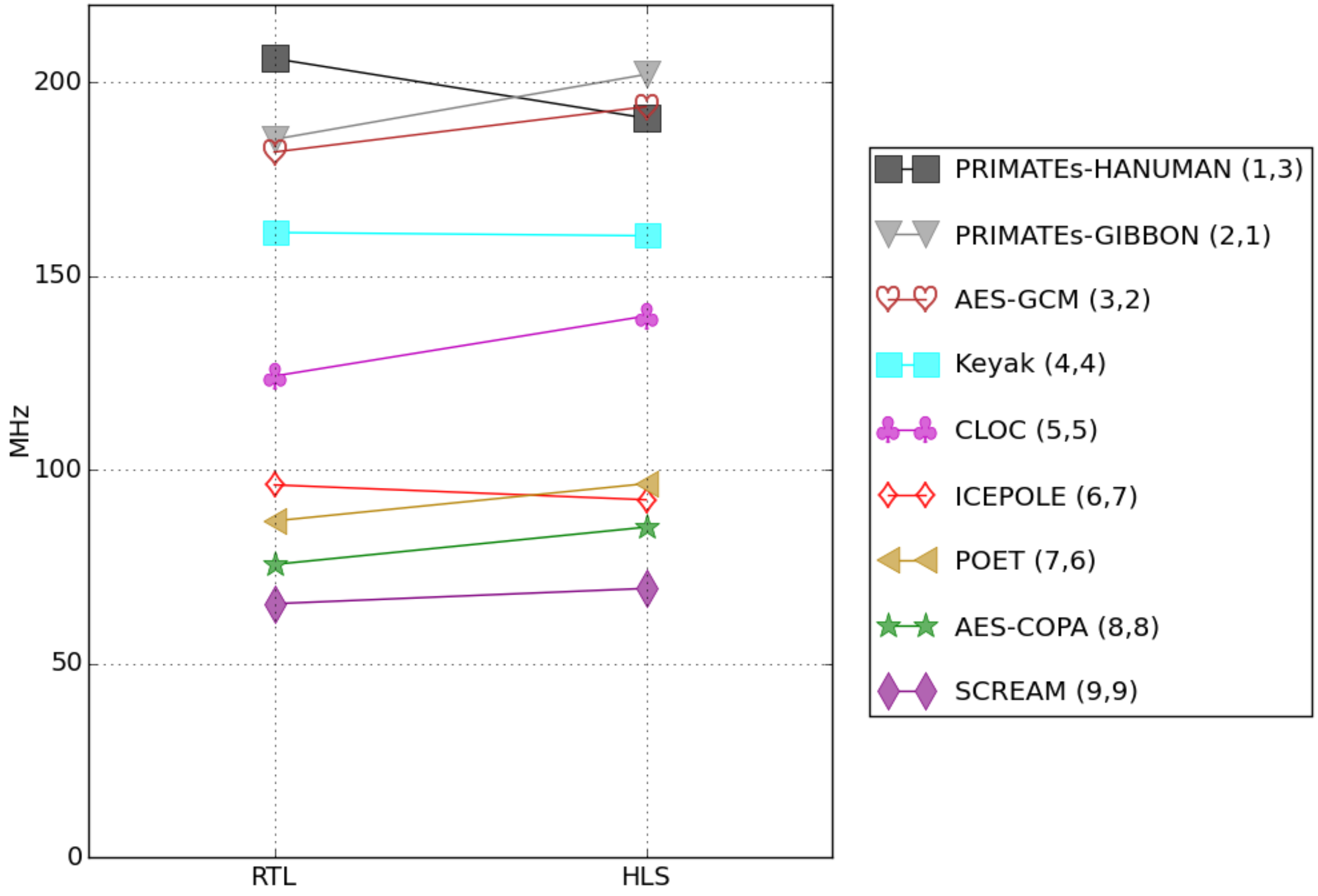
- Difficulty in inferring an overlap between completing the last round and reading the next input block
- One additional clock cycle used for initialization of the state at the beginning of each round
- The formulas for throughput:

HLS: Throughput = Block_size / ((#Rounds+2) * T_{CLK})

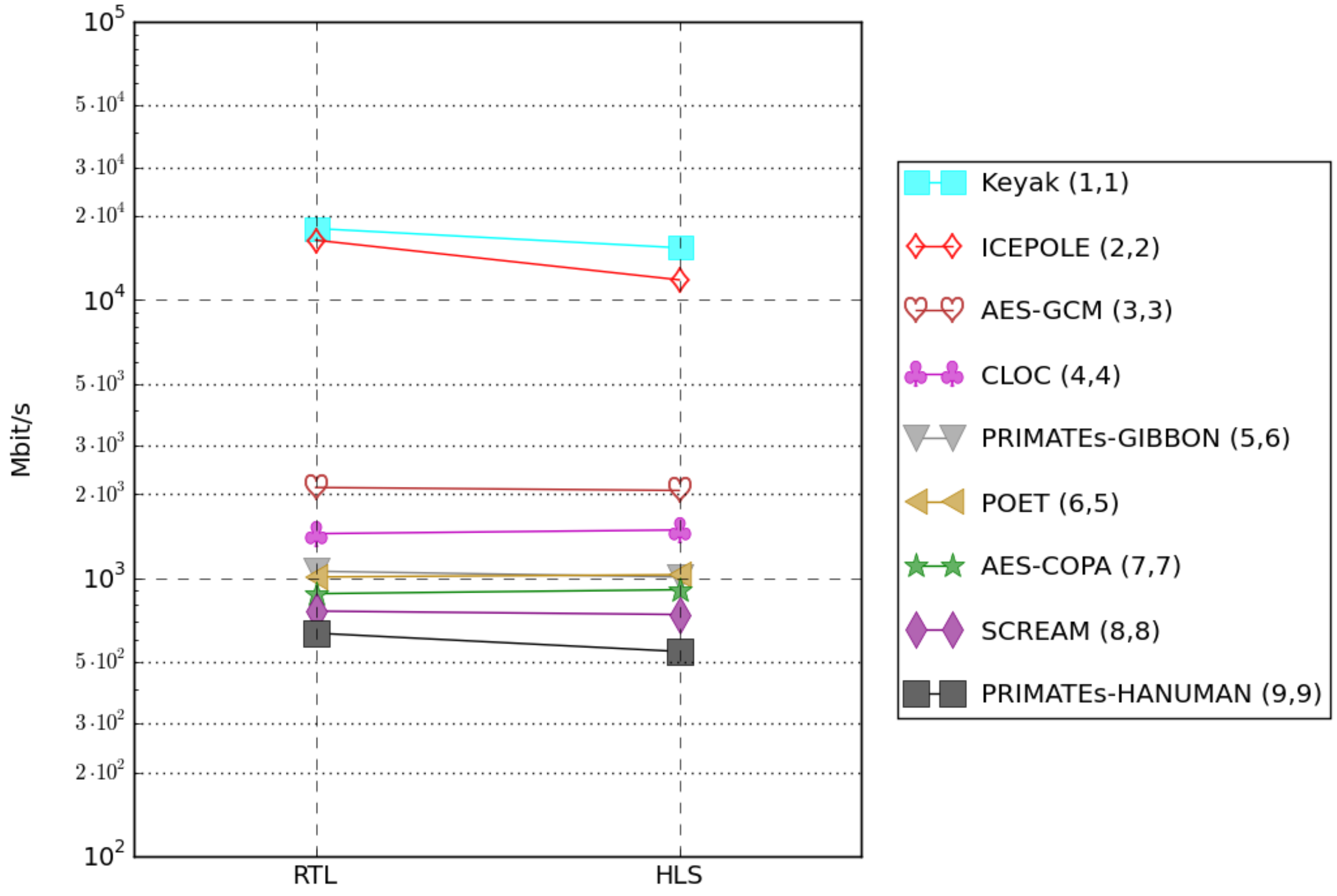
RTL: Throughput = Block_size / (#Rounds+C * T_{CLK})

C=0, 1 depending on the algorithm

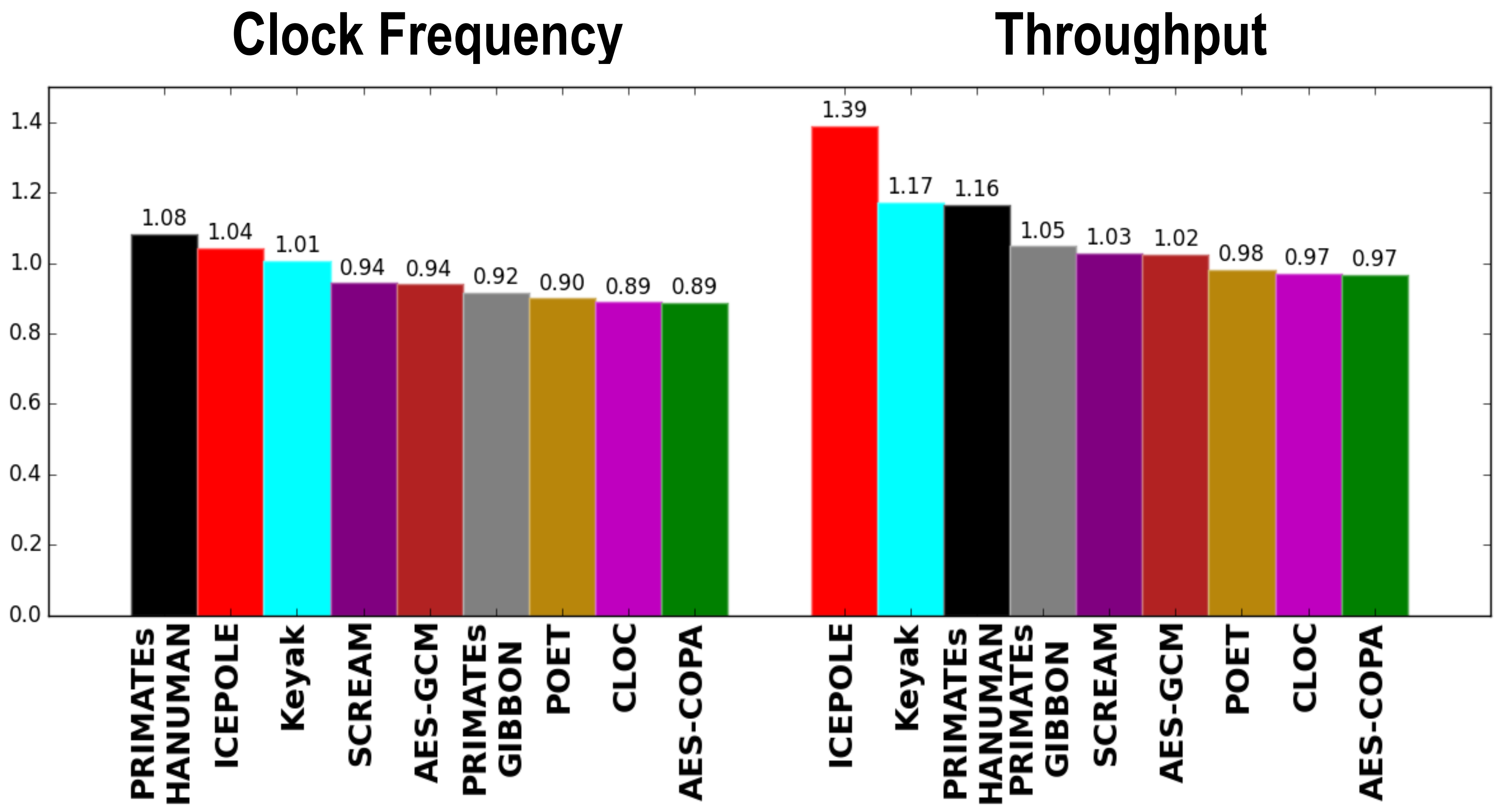
RTL vs. HLS Clock Frequency in Zynq 7000



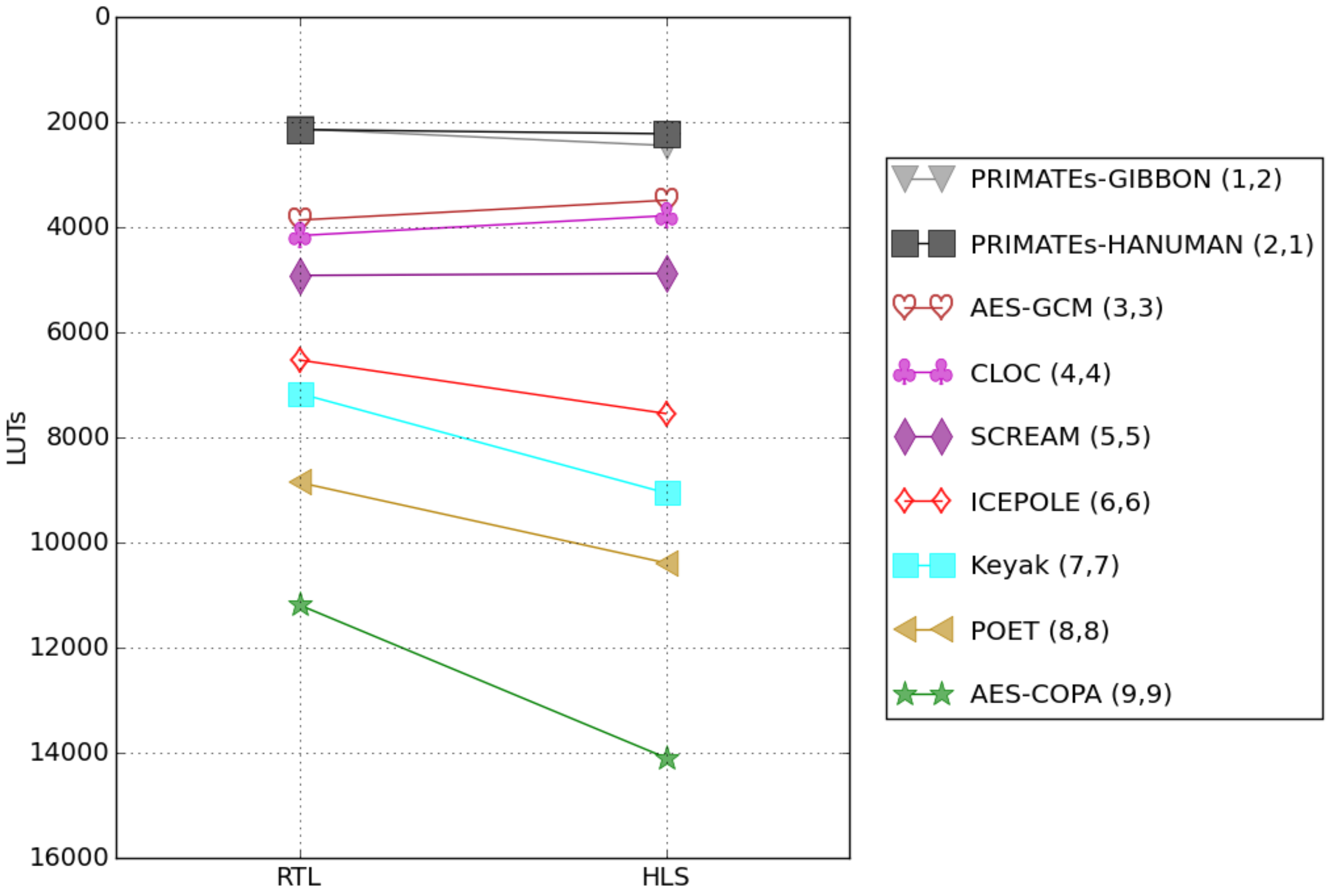
RTL vs. HLS Throughput in Zynq 7000



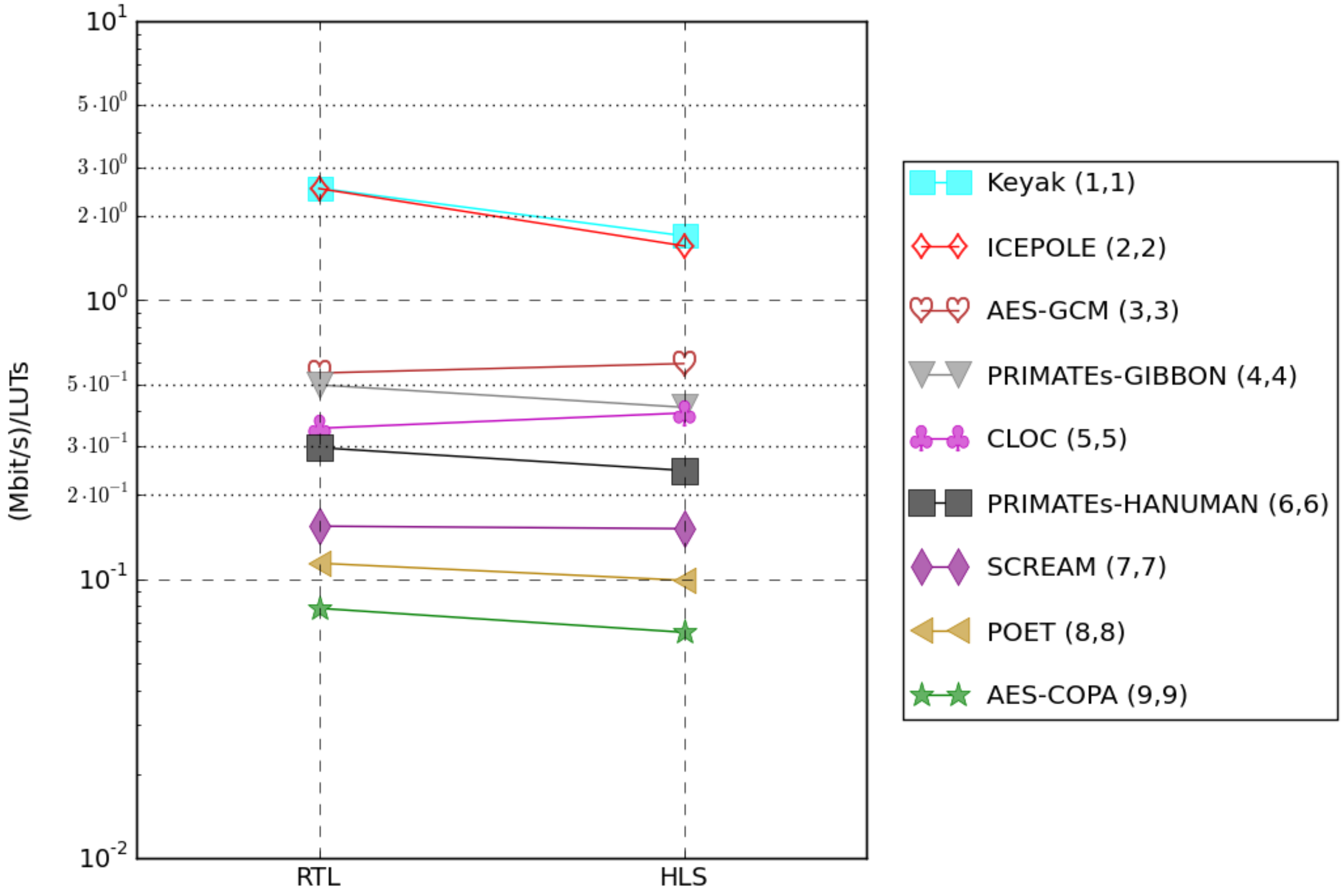
RTL vs. HLS Ratios in Zynq 7000



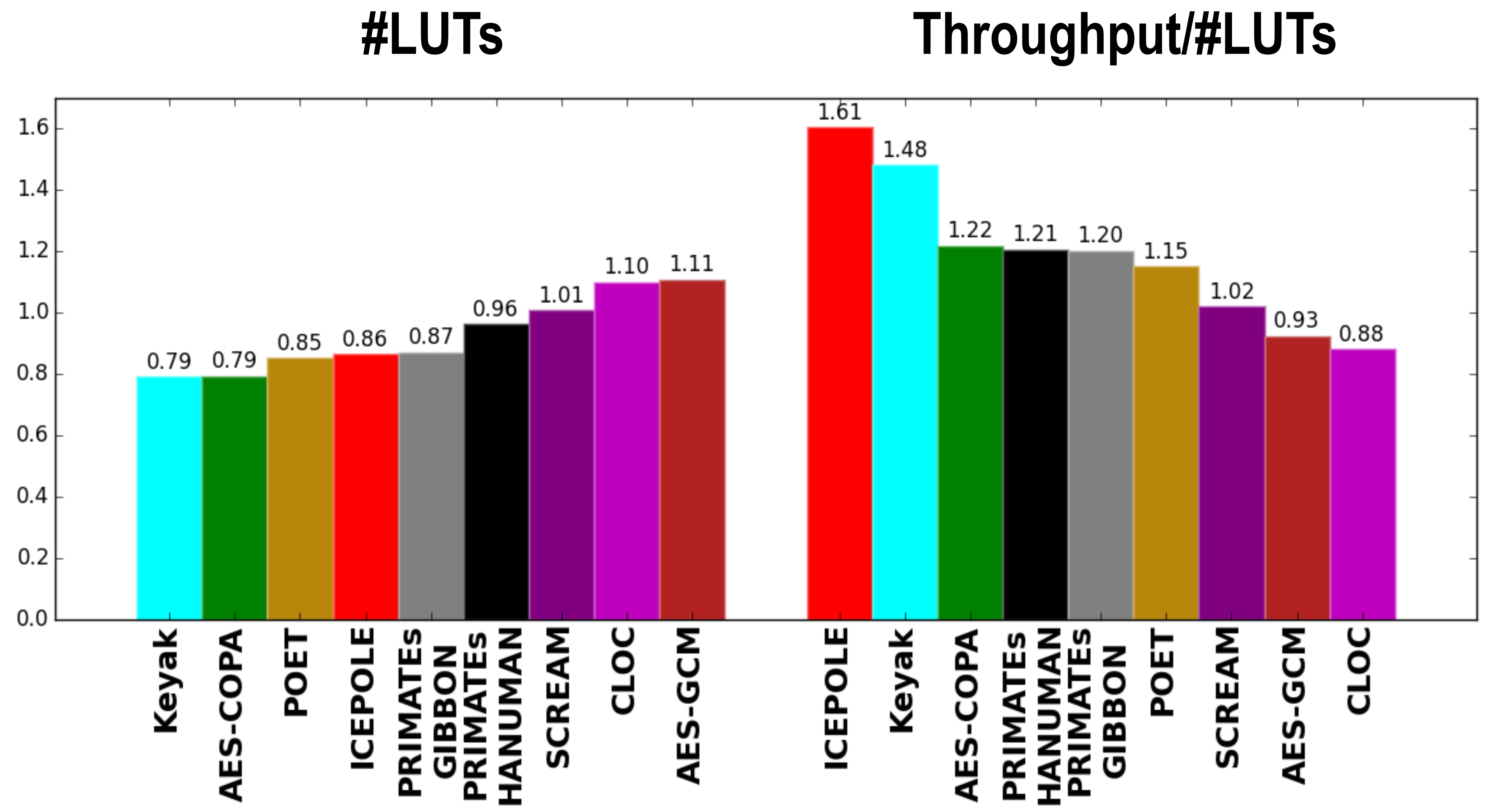
RTL vs. HLS #LUTs in Zynq 7000



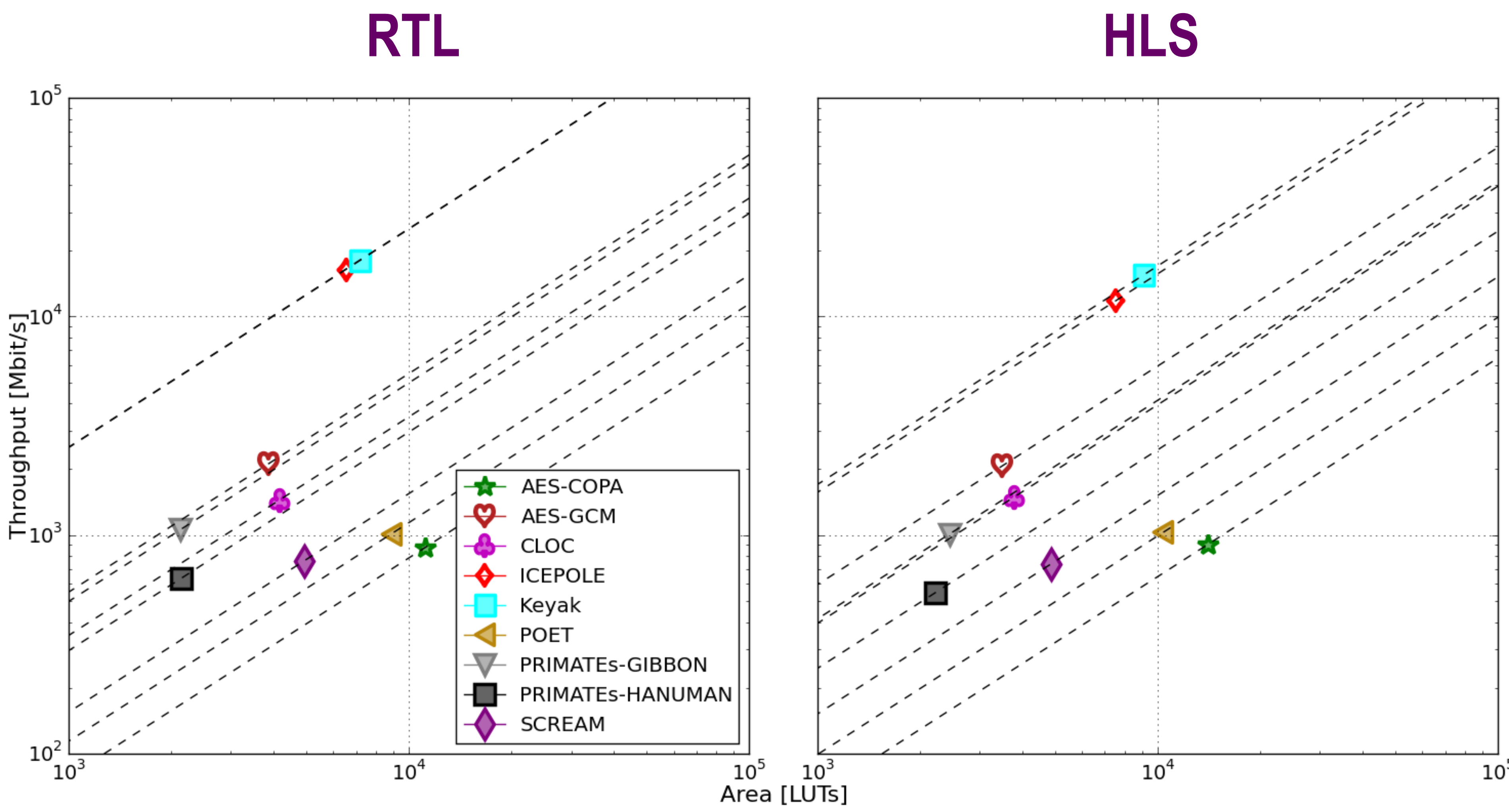
RTL vs. HLS Throughput/#LUTs in Zynq 7000



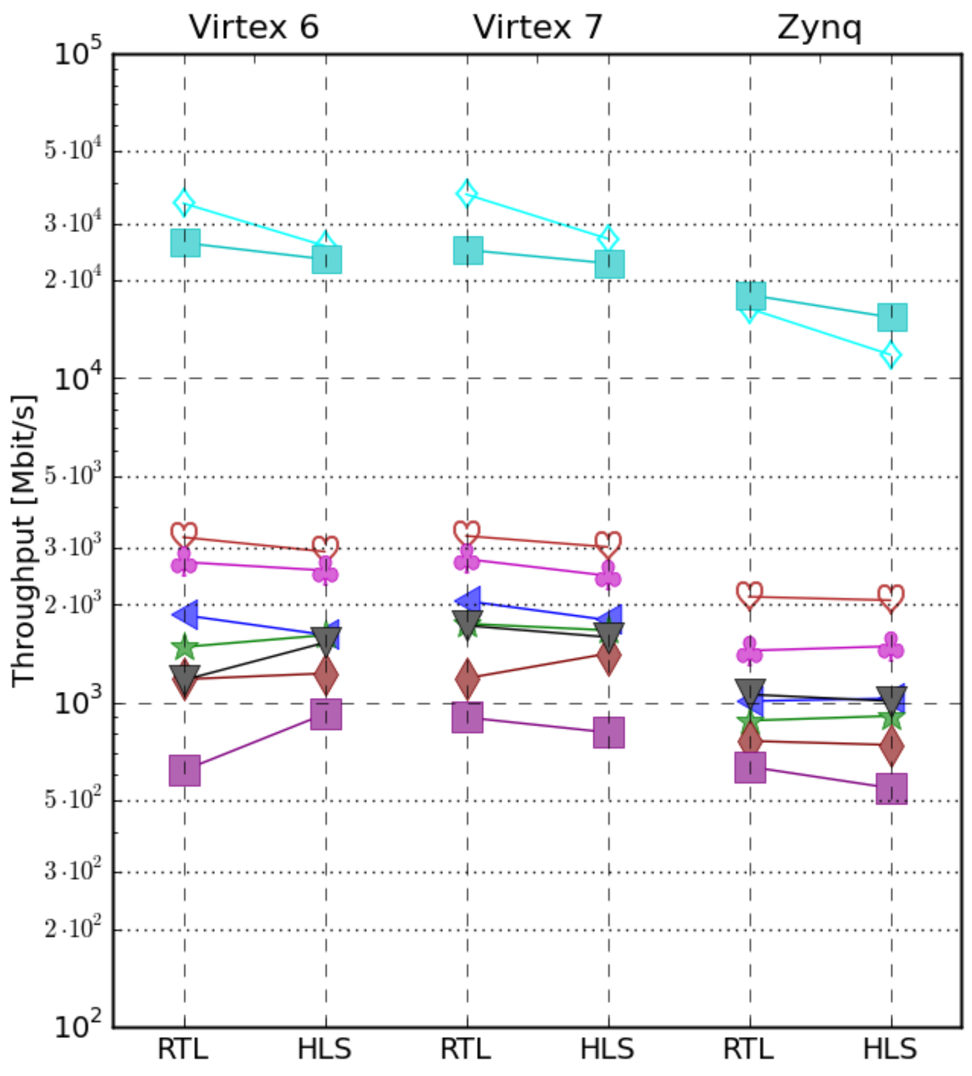
RTL vs. HLS Ratios in Zynq 7000



Throughput vs. LUTs in Zynq 7000

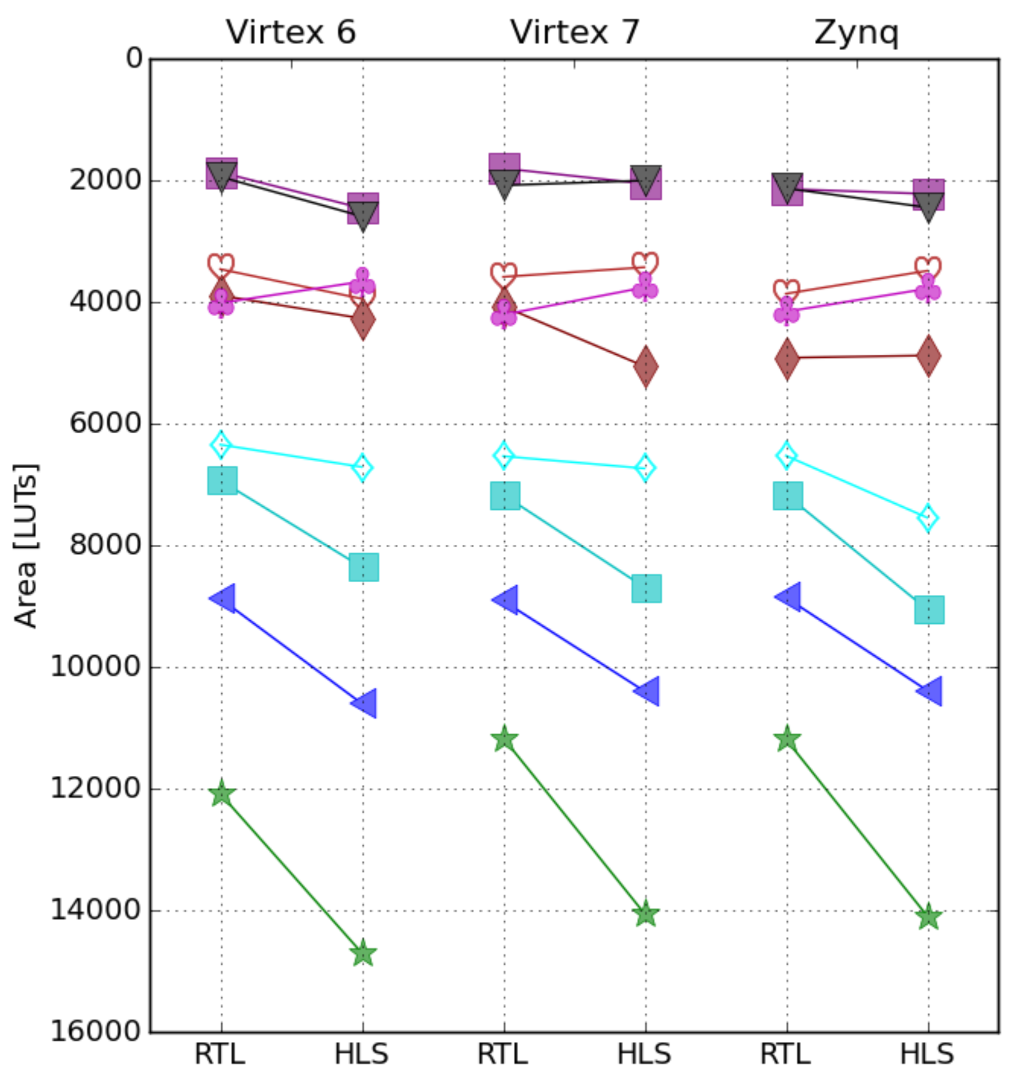


RTL vs. HLS Throughput



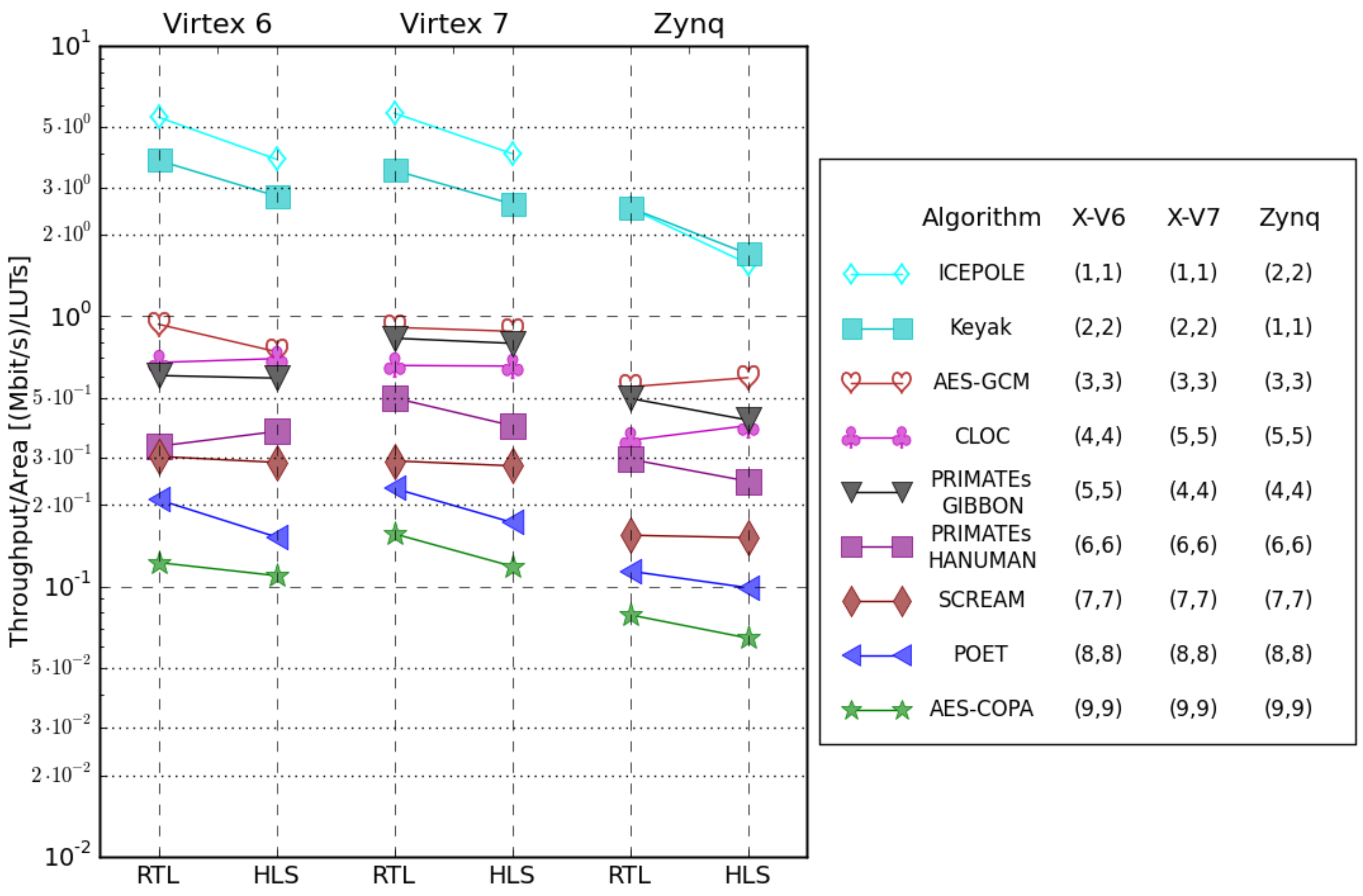
	Algorithm	X-V6	X-V7	Zynq
◇	ICEPOLE	(1,1)	(1,1)	(2,2)
■	Keyak	(2,2)	(2,2)	(1,1)
♡	AES-GCM	(3,3)	(3,3)	(3,3)
♣	CLOC	(4,4)	(4,4)	(4,4)
◀	POET	(5,5)	(5,5)	(6,5)
★	AES-COPA	(6,6)	(6,6)	(7,7)
♦	SCREAM	(7,8)	(8,8)	(8,8)
▼	PRIMATEs GIBBON	(8,7)	(7,7)	(5,6)
■	PRIMATEs HANUMAN	(9,9)	(9,9)	(9,9)

RTL vs. HLS #LUTs



	Algorithm	X-V6	X-V7	Zynq
■	PRIMATEs HANUMAN	(1,1)	(1,2)	(2,1)
▼	PRIMATEs GIBBON	(2,2)	(2,1)	(1,2)
♡	AES-GCM	(3,4)	(3,3)	(3,3)
♦	SCREAM	(4,5)	(4,5)	(5,5)
♣	CLOC	(5,3)	(5,4)	(4,4)
◇	ICEPOLE	(6,6)	(6,6)	(6,6)
■	Keyak	(7,7)	(7,7)	(7,7)
◀	POET	(8,8)	(8,8)	(8,8)
★	AES-COPA	(9,9)	(9,9)	(9,9)

RTL vs. HLS Throughput/#LUTs

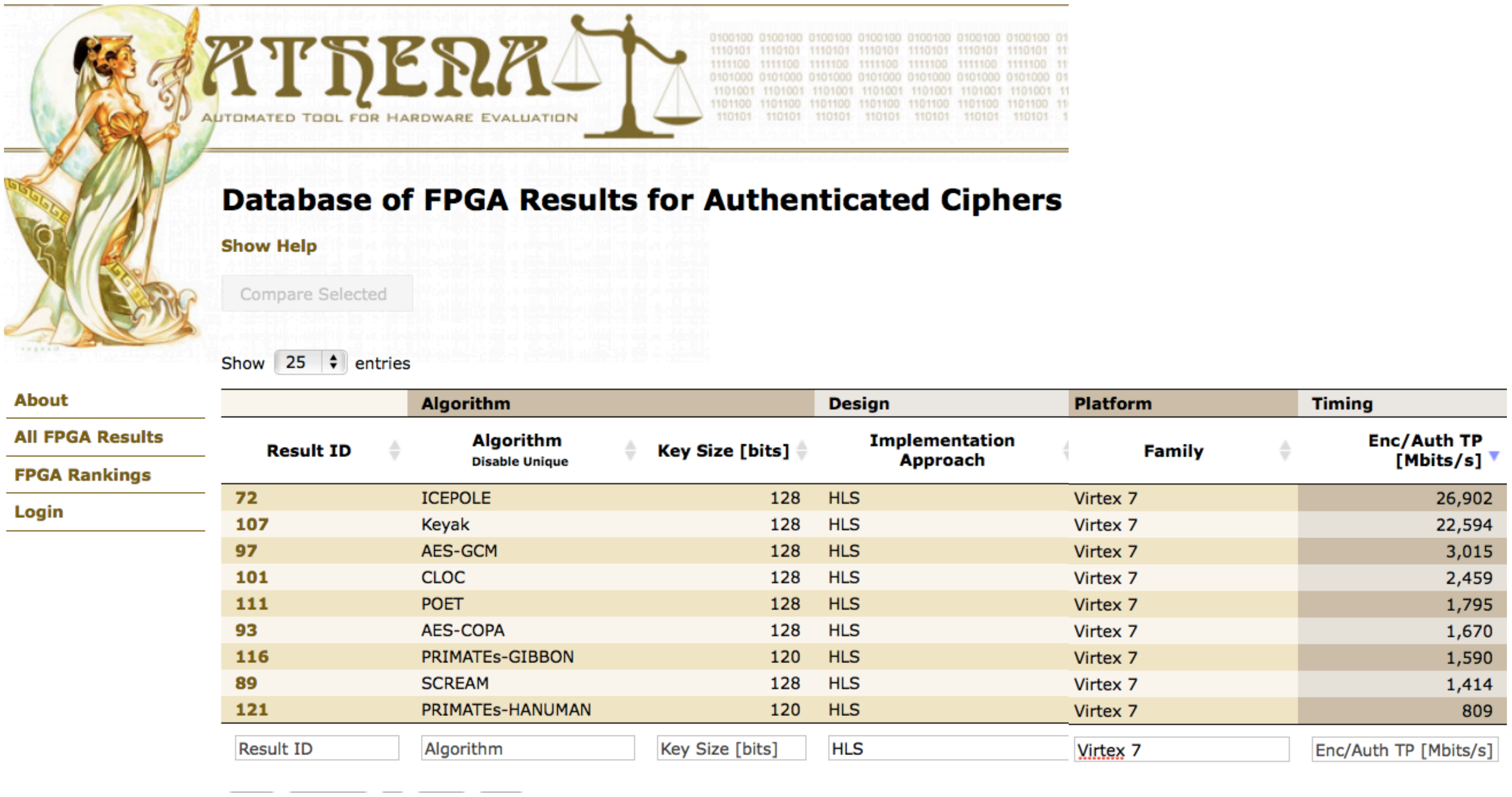


	Algorithm	X-V6	X-V7	Zynq
◇—◇	ICEPOLE	(1,1)	(1,1)	(2,2)
■—■	Keyak	(2,2)	(2,2)	(1,1)
♡—♡	AES-GCM	(3,3)	(3,3)	(3,3)
♣—♣	CLOC	(4,4)	(5,5)	(5,5)
▼—▼	PRIMATEs GIBBON	(5,5)	(4,4)	(4,4)
■—■	PRIMATEs HANUMAN	(6,6)	(6,6)	(6,6)
◆—◆	SCREAM	(7,7)	(7,7)	(7,7)
◀—◀	POET	(8,8)	(8,8)	(8,8)
★—★	AES-COPA	(9,9)	(9,9)	(9,9)

ATHENa Database of Results for Authenticated Ciphers

- Available at
  http://cryptography.gmu.edu/athena
- Developed by John Pham, a Master's-level student of Jens-Peter Kaps
- Results can be entered by designers themselves.
  If you would like to do that, please contact me regarding an account.
- The ATHENa Option Optimization Tool supports automatic generation of results suitable for uploading to the database

Ordered Listing with a Single-Best (Unique) Result per Each Algorithm



Database of FPGA Results for Authenticated Ciphers

	Algorithm		Design	Platform	Timing
Result ID	Algorithm Disable Unique	Key Size [bits]	Implementation Approach	Family	Enc/Auth TP [Mbits/s]
72	ICEPOLE	128	HLS	Virtex 7	26,902
107	Keyak	128	HLS	Virtex 7	22,594
97	AES-GCM	128	HLS	Virtex 7	3,015
101	CLOC	128	HLS	Virtex 7	2,459
111	POET	128	HLS	Virtex 7	1,795
93	AES-COPA	128	HLS	Virtex 7	1,670
116	PRIMATEs-GIBBON	120	HLS	Virtex 7	1,590
89	SCREAM	128	HLS	Virtex 7	1,414
121	PRIMATEs-HANUMAN	120	HLS	Virtex 7	809
Result ID	Algorithm	Key Size [bits]	HLS	Virtex 7	Enc/Auth TP [Mbits/s]

Details of Result ID 97

Algorithm

IV or Nonce Size [bits]:	96
Transformation Category:	Cryptographic
Transformation:	Authenticated Cipher
Group:	Standards
Algorithm:	AES-GCM
Tag Size [bits]:	128
Associated Data Support:	-
Key Size [bits]:	128
Secret Message Number:	-
Secret Message Number Size [bits]:	-
Message Block Size [bits]:	128
Other Parameters:	-
Specification:	**SP-800-38D.pdf**
Formula for Message Size After Padding:	-

Design

Design ID:	**21**
Impl Approach:	HLS
Hardware API:	GMU_AEAD_Core_API_v1
Primary Optimization Target:	Throughput/Area
Secondary Optimization Target:	-
Architecture Type:	Basic Iterative
Description Language:	VHDL
Use of Megafunctions or Primitives:	No
List of Megafunctions or Primitives:	-
Maximum Number of Streams Processed in Parallel:	1
Number of Clock Cycles per Message Block in a Long Message:	12
Datapath Width [bits]:	128
Padding:	Yes
Minimum Message Unit:	-
Input Bus Width [bits]:	32
Output Bus Width [bits]:	32

Details of Result ID 97

Platform

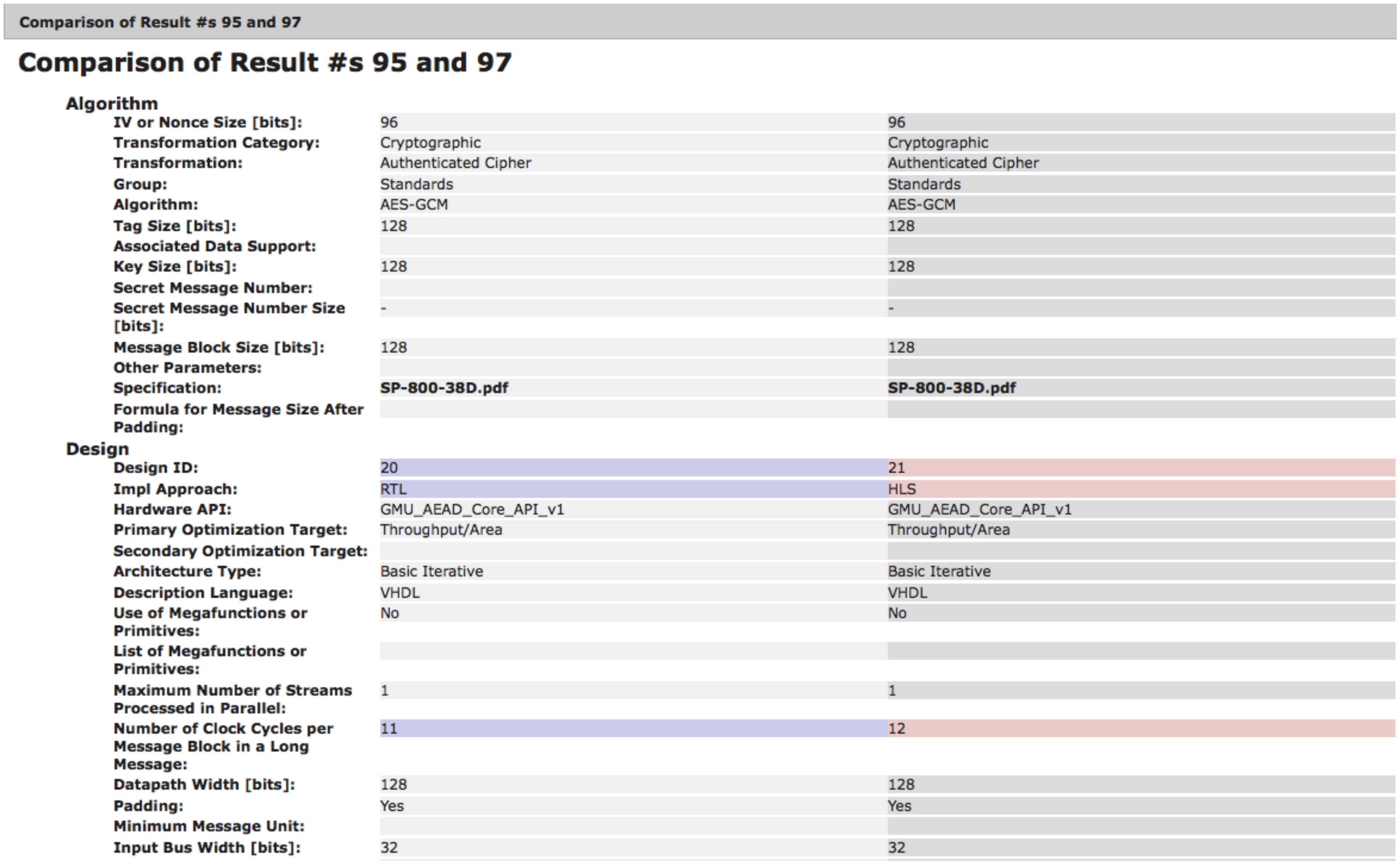
Device Vendor:	Xilinx
Family:	Virtex 7
Device:	xc7vx485tffg1761-2

Timing

Encryption/Authentication Throughput [Mbits/s]:	3,015
Decryption/Authentication Throughput [Mbits/s]:	3,015
Authentication-Only Throughput [Mbits/s]:	3,015
Synthesis Clock Frequency [MHz]:	-
Key Scheduling Time [ns]:	-
Requested Synthesis Clock Frequency [MHz]:	-
Requested Implementation Clock Frequency [MHz]:	-
Implementation Clock Frequency [MHz]:	282.650
(Encryption/Authentication Throughput)/LUT [(Mbits/s)/LUT]:	0.879
(Encryption/Authentication Throughput)/Slice [(Mbits/s)/Slice]:	2.728
(Decryption/Authentication Throughput)/LUT [(Mbits/s)/LUT]:	0.879
(Decryption/Authentication Throughput)/Slice [(Mbits/s)/Slice]:	2.728
(Auth-Only Throughput)/LUT [(Mbits/s)/LUT]:	0.879
(Auth-Only Throughput)/Slice [(Mbits/s)/Slice]:	2.728

Resource Utilization

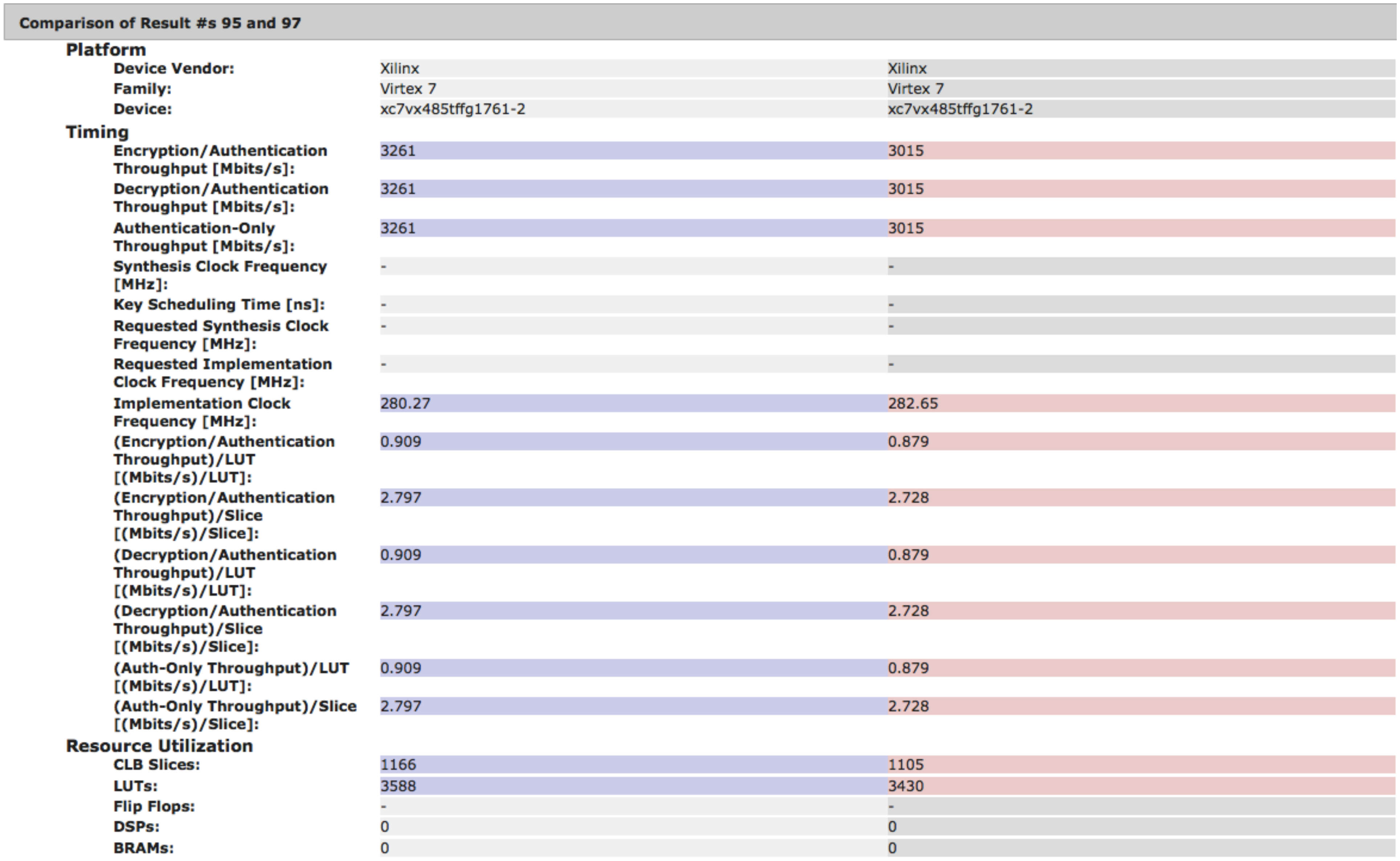
CLB Slices:	1,105
LUTs:	3,430
Flip Flops:	-
DSPs:	0
BRAMs:	0



Comparison of Result #s 95 and 97

Comparison of Result #s 95 and 97

Algorithm		
IV or Nonce Size [bits]:	96	96
Transformation Category:	Cryptographic	Cryptographic
Transformation:	Authenticated Cipher	Authenticated Cipher
Group:	Standards	Standards
Algorithm:	AES-GCM	AES-GCM
Tag Size [bits]:	128	128
Associated Data Support:		
Key Size [bits]:	128	128
Secret Message Number:		
Secret Message Number Size [bits]:	-	-
Message Block Size [bits]:	128	128
Other Parameters:		
Specification:	**SP-800-38D.pdf**	**SP-800-38D.pdf**
Formula for Message Size After Padding:		
Design		
Design ID:	20	21
Impl Approach:	RTL	HLS
Hardware API:	GMU_AEAD_Core_API_v1	GMU_AEAD_Core_API_v1
Primary Optimization Target:	Throughput/Area	Throughput/Area
Secondary Optimization Target:		
Architecture Type:	Basic Iterative	Basic Iterative
Description Language:	VHDL	VHDL
Use of Megafunctions or Primitives:	No	No
List of Megafunctions or Primitives:		
Maximum Number of Streams Processed in Parallel:	1	1
Number of Clock Cycles per Message Block in a Long Message:	11	12
Datapath Width [bits]:	128	128
Padding:	Yes	Yes
Minimum Message Unit:		
Input Bus Width [bits]:	32	32



Comparison of Result #s 95 and 97

Platform		
Device Vendor:	Xilinx	Xilinx
Family:	Virtex 7	Virtex 7
Device:	xc7vx485tffg1761-2	xc7vx485tffg1761-2
Timing		
Encryption/Authentication Throughput [Mbits/s]:	3261	3015
Decryption/Authentication Throughput [Mbits/s]:	3261	3015
Authentication-Only Throughput [Mbits/s]:	3261	3015
Synthesis Clock Frequency [MHz]:	-	-
Key Scheduling Time [ns]:	-	-
Requested Synthesis Clock Frequency [MHz]:	-	-
Requested Implementation Clock Frequency [MHz]:	-	-
Implementation Clock Frequency [MHz]:	280.27	282.65
(Encryption/Authentication Throughput)/LUT [(Mbits/s)/LUT]:	0.909	0.879
(Encryption/Authentication Throughput)/Slice [(Mbits/s)/Slice]:	2.797	2.728
(Decryption/Authentication Throughput)/LUT [(Mbits/s)/LUT]:	0.909	0.879
(Decryption/Authentication Throughput)/Slice [(Mbits/s)/Slice]:	2.797	2.728
(Auth-Only Throughput)/LUT [(Mbits/s)/LUT]:	0.909	0.879
(Auth-Only Throughput)/Slice [(Mbits/s)/Slice]:	2.797	2.728
Resource Utilization		
CLB Slices:	1166	1105
LUTs:	3588	3430
Flip Flops:	-	-
DSPs:	0	0
BRAMs:	0	0

Database of Results

Ranking View:

Supports the choice of

I. Hardware API (e.g., GMU_AEAD_Core_API_v1, GMU_AEAD_API_v1, GMU_CipherCore_API_v1)

II. Family (e.g., Virtex 6 (default), Virtex 7, Zynq 7000)

III. Operation (Authenticated Encryption (default), Authenticated Decryption, Authentication Only)

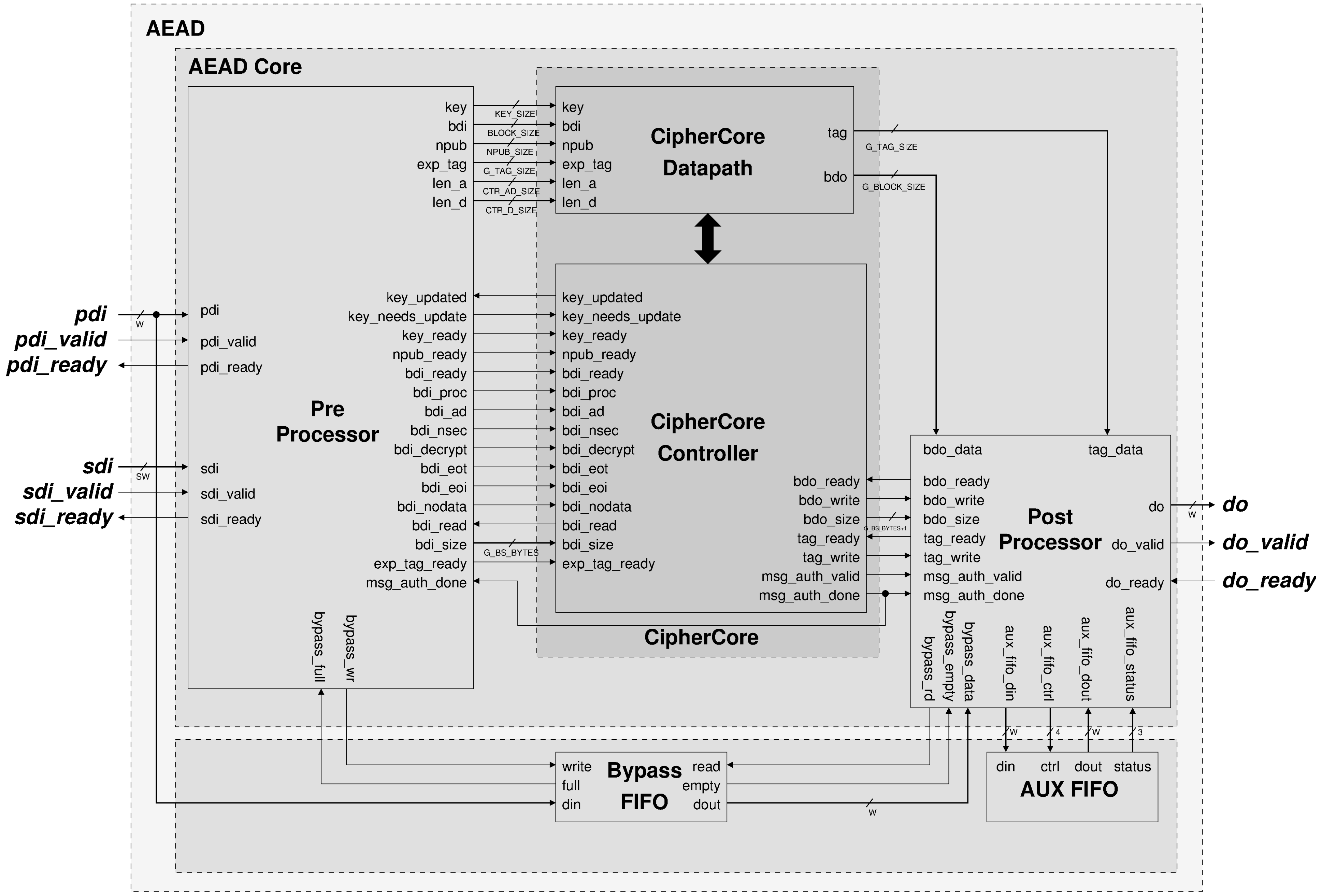
IV. Unit of Area (for Xilinx FPGAs: LUTs vs. Slices)

V. Ranking criteria (Throughput/Area (default), Throughput, Area)

Table View:

- more flexibility in terms of filtering, reviewing, ranking, searching for, and comparing results with one another

Uniform Hardware API



GMU Hardware API

- Complete Hardware API for authenticated ciphers developed, including
  - Interface
  - Communication Protocol
- Design with the GMU hardware API facilitated by
  - Detailed specification
  - Universal testbench and Automated Test Vector Generation
  - PreProcessor and PostProcessor Units for high-speed implementations
  - Universal wrappers for generating results
  - AES and Keccak-F Permutation source codes
  - Ease of recording and comparing results using ATHENa database
  - Full example of use in Zynq 7000 based on Xilinx AXI4 IPs
- GMU proposal open for discussion and possible improvements through
  - Better specification
  - Better implementation of supporting codes

Future Work

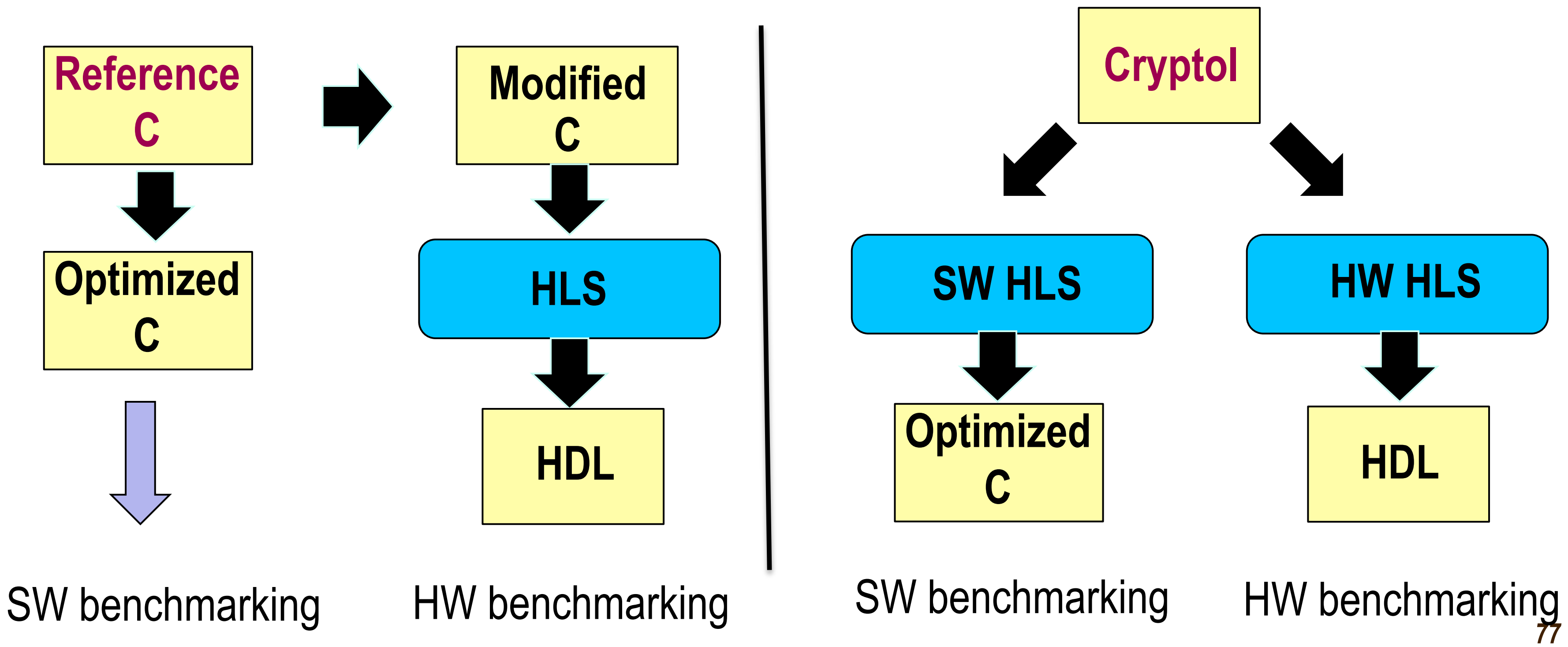
LegUp – Academic Tool for HLS

- Open-source HLS Tool
  - Developed at the University of Toronto
  - Faculty supervisors: Jason H. Anderson and Stephen Brown
  - FPL Community Award 2014
- High-Level Synthesis from C to Verilog
- Targets Altera FPGAs (extension to Xilinx relatively simple)
- Two flows
  - Pure Hardware
  - Hardware/Software Hybrid

    = Tiger MIPS + hardware accelerator(s) + Avalon bus + shared on-chip and off-chip memory

Cryptol – New Language for Cryptology

– Domain specific language for cryptology: Cryptol
  - High-level programming language similar to Haskell
  - Developed by Galois Inc. based in Portland, USA
– High-Level Synthesis from Cryptol to efficient Software and Hardware



Conclusions

- High-level synthesis offers a potential to facilitate hardware benchmarking during the design of cryptographic algorithms and at the early stages of cryptographic contests

- Our case studies demonstrated correct ranking for majority of candidates using all major performance metrics

- More research needed to overcome remaining difficulties
  - Suboptimal control unit
  - Wide range of RTL to HLS performance metric ratios
  - Efficient and reliable generation of HLS-ready C codes

Thank you!

Comments?

Questions?

Suggestions?

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

CERG: http://cryptography.gmu.edu