RTL Implementations and FPGA Benchmarking of Three Authenticated Ciphers Competing in CAESAR Round Two

William Diehl and Kris Gaj

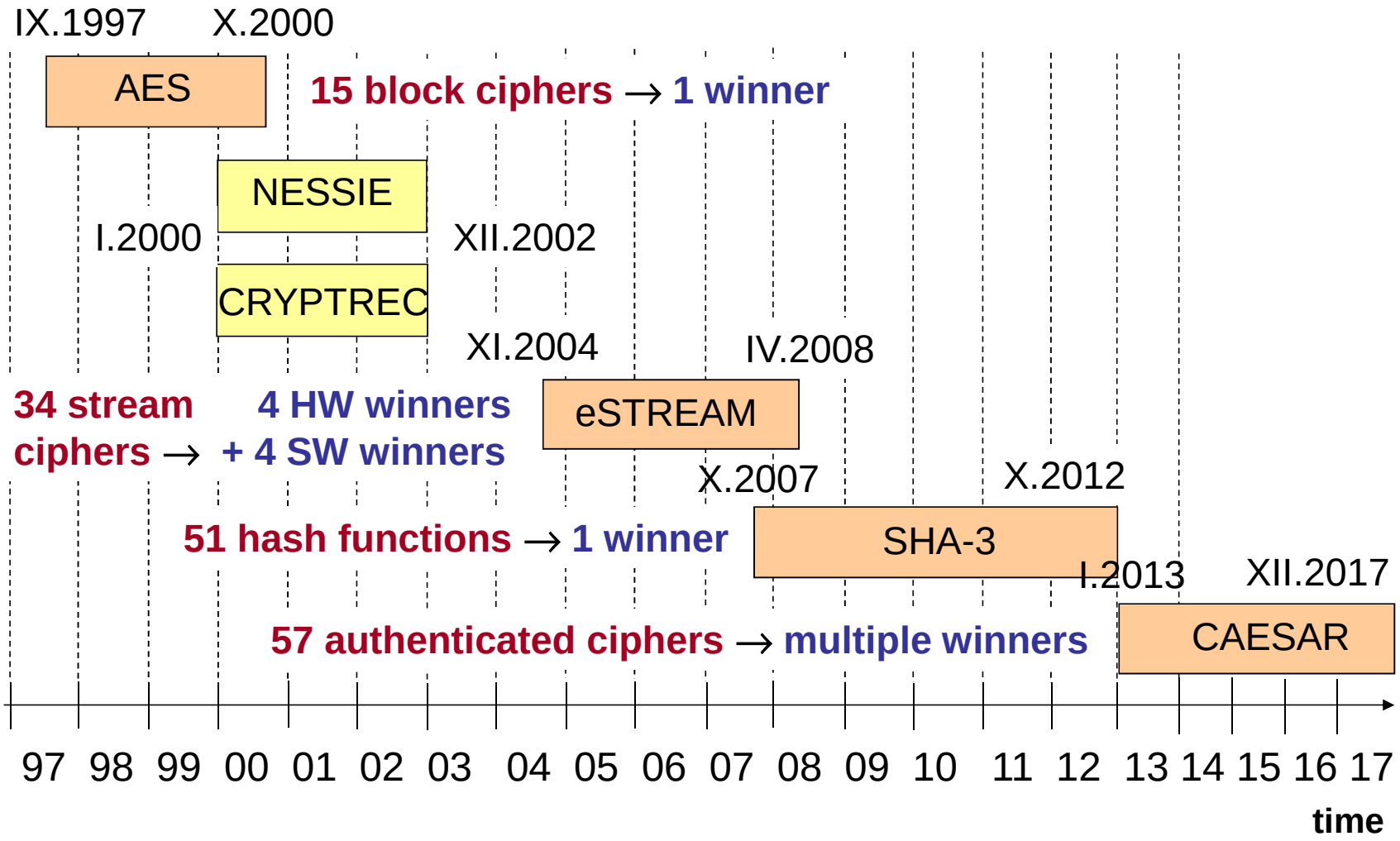
ECE Department, George Mason University, Fairfax, Virginia, USA
http://cryptography.gmu.edu

Based on work supported by the National Science Foundation under
Grant No. 1314540

Outline

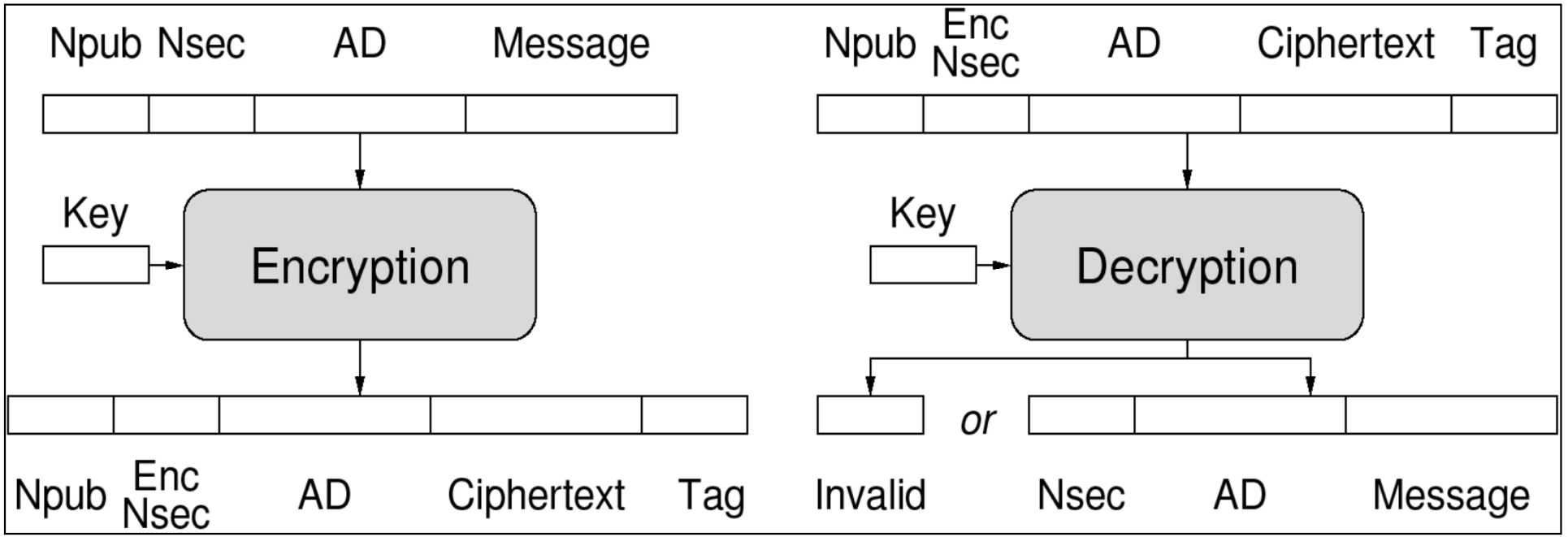
- CAESAR Competition and Authenticated Ciphers
- CAESAR Hardware API & Compliant Code Development
- Discussion of Designs
- Results
- Summary, Conclusions, and Lessons Learned

Cryptographic Standard Contests



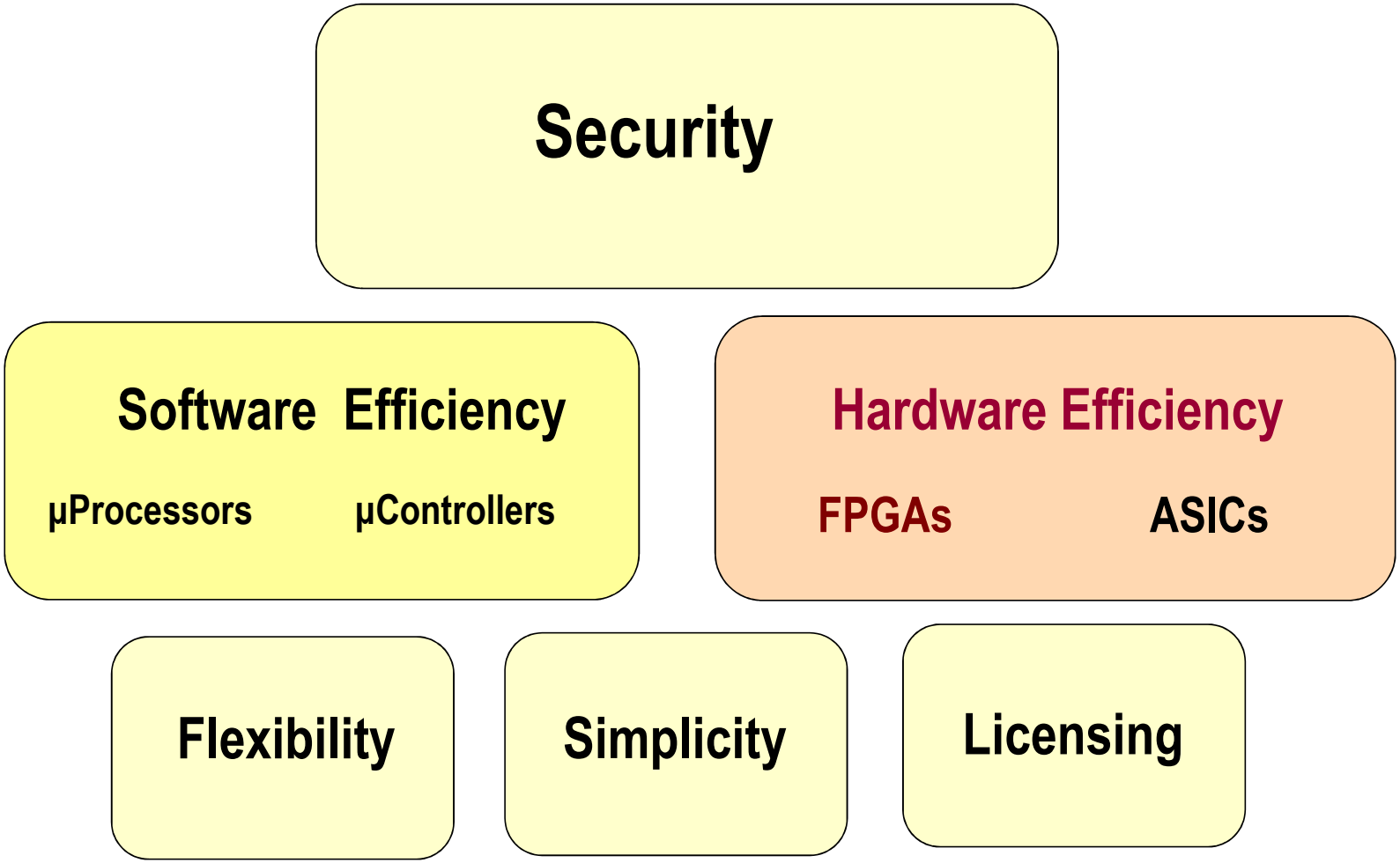
Authenticated Ciphers

Combine the functionality of **confidentiality**, **integrity**, and **authenticity**



Notation: Npub = Public Message Number; (Enc) Nsec = (Encrypted) Secret Message Number; AD = Associated Data

Evaluation Criteria



Motivation for Universal API

- Hardware API can have a high influence on Area and Throughput/Area ratio of all implementations
- Hardware API typically much more difficult to modify than Software API
- Without a comprehensive hardware API, the comparison highly unreliable and potentially unfair
- Designers can “play to strengths” and “hide weaknesses”

Conclusion: *Impossible to perform fair evaluation of hardware implementations without standardized interface and protocol*

CAESAR Hardware API

Specifies:

- Minimum compliance criteria
- Interface
- Communication protocol
- Timing characteristics

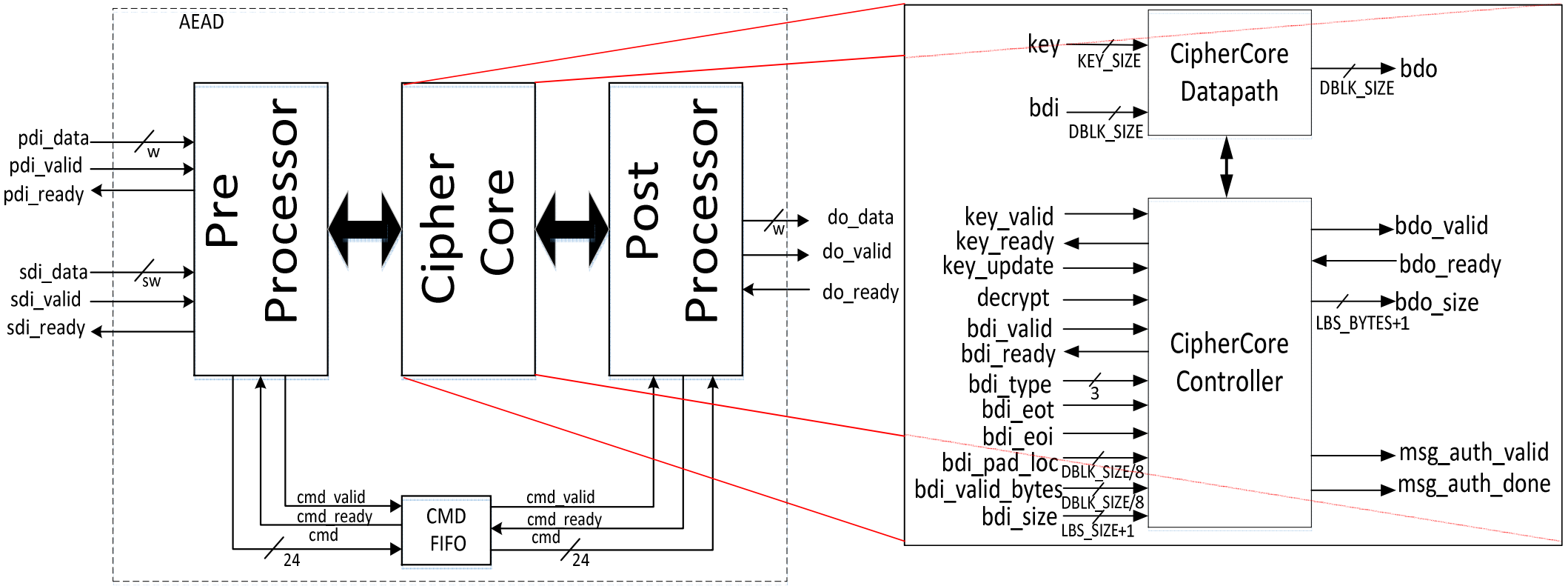
Assures:

- Compatibility
- Fairness

Timeline:

- Based on the GMU Hardware API presented at CryptArchi 2015, DIAC 2015, and ReConFig 2015
- Revised version posted on Feb. 15, 2016
- Officially approved by the CAESAR Committee on May 6, 2016

General Interface and Internal Architecture for High-Speed Implementations



Additional detail available at https://cryptography.gmu.edu/athena/

GMU Support for Designers of VHDL/Verilog Code

Implementer's Guide

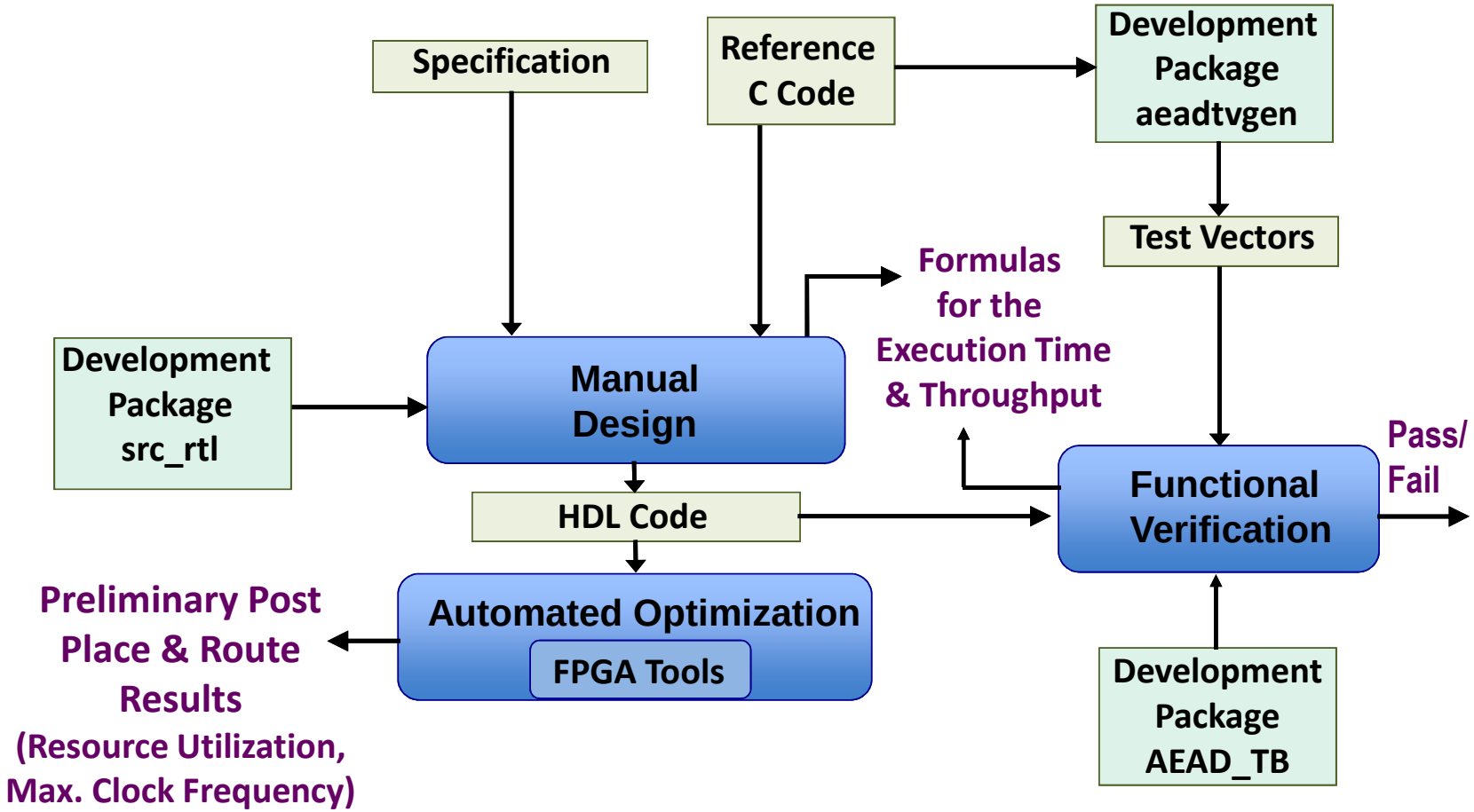
- v1.0 - May 12, 2016

Development Package

a. VHDL code of generic pre-processing and post- processing units for high-speed implementations (src_rtl)
b. Universal testbench (AEAD_TB)
c. Python app used to automatically generate test vectors (aeadtvgen)
d. Six reference high-speed implementations of Dummy authenticated ciphers

https://cryptography.gmu.edu/athena/index.php?id=download

The API Compliant Code Development



My Tasks

Ciphers

- SCREAM, POET, Minalpher
- OMD (Not presented)

Compliance

- Round Two published specification
- C Reference Code
- CAESAR HW API

Optimization Criteria

1. Throughput-to-area (TP/A) ratio
2. Throughput (Maximize frequency; minimize cycles/block)
3. Area (Minimize LUTs)

SCREAM

Side-Channel Resistant Authenticated Encryption with Masking

- Based on Liskov, Rivest, and Wagner's "Tweakable Block Cipher"
- Unique tweak for every block
- 128-bit key, state variable, tag
- Padding for Associated Data

SCREAM (cont'd)

Cryptographic primitive is Tweakable Block Cipher (TBC) (E_K)

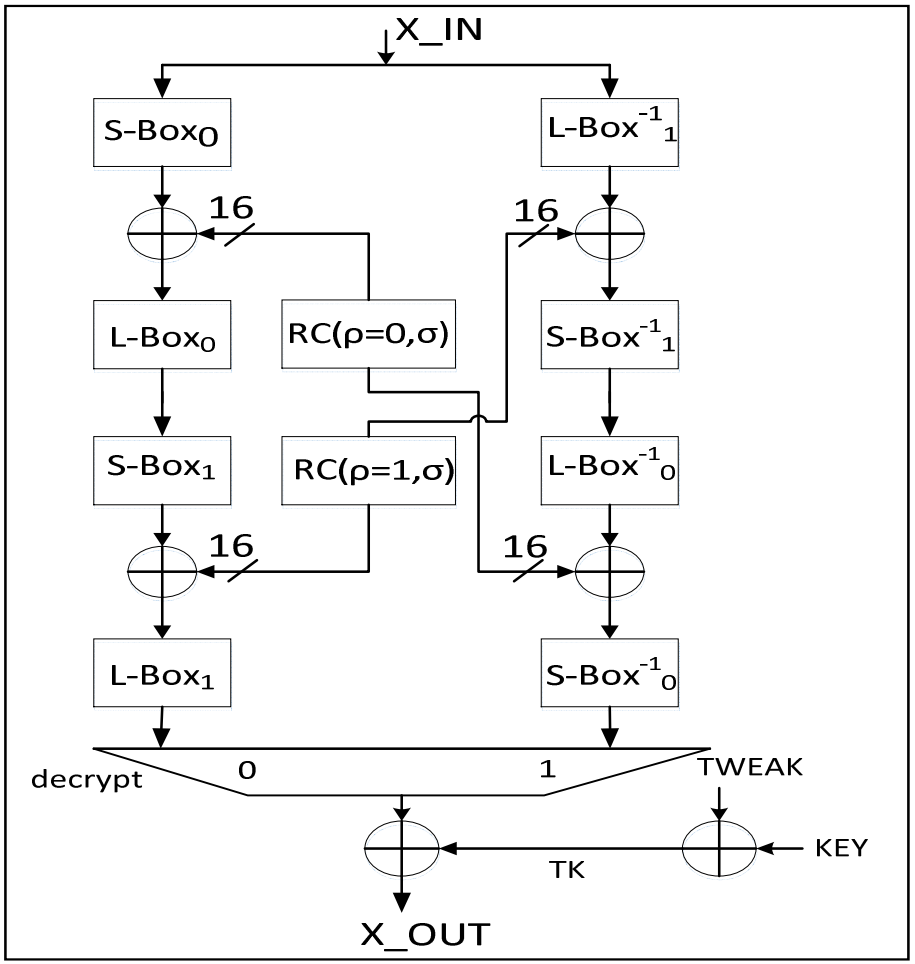
10 steps (σ) per block, 2 rounds (ρ) per step

- **Tweak updated once per step to form Tweak key (TK)**

Non-linear substitution layer composed of 8x8 "nearly involute" S-Boxes

16-bit Round Constant, RC(ρ,σ) applied each round

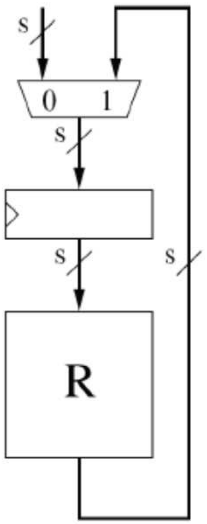
Linear permutation layer L-Box



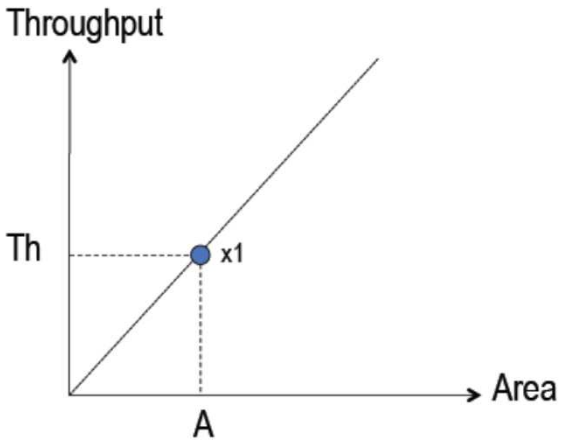
Bus widths are 128 bits unless indicated

Basic Iterative versus Unrolled Architectures

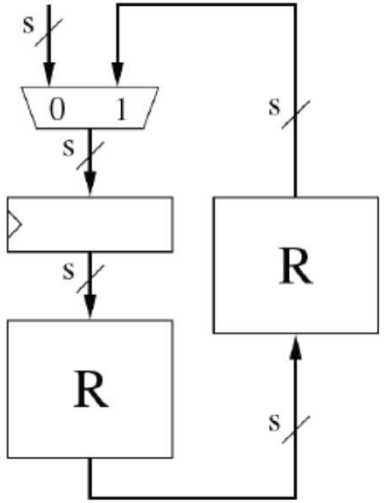
Basic Iterative



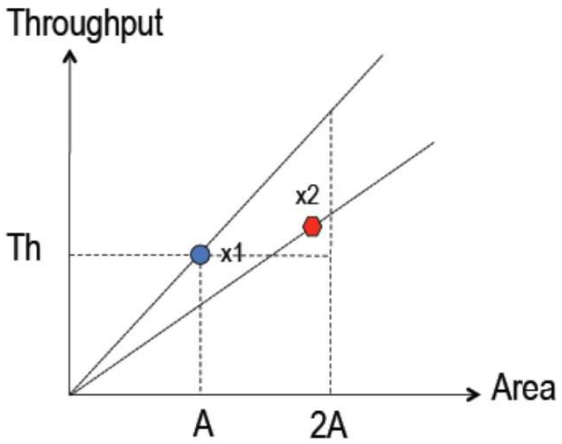
- datapath width = state size
- one clock cycle per one round/step



Unrolled x2



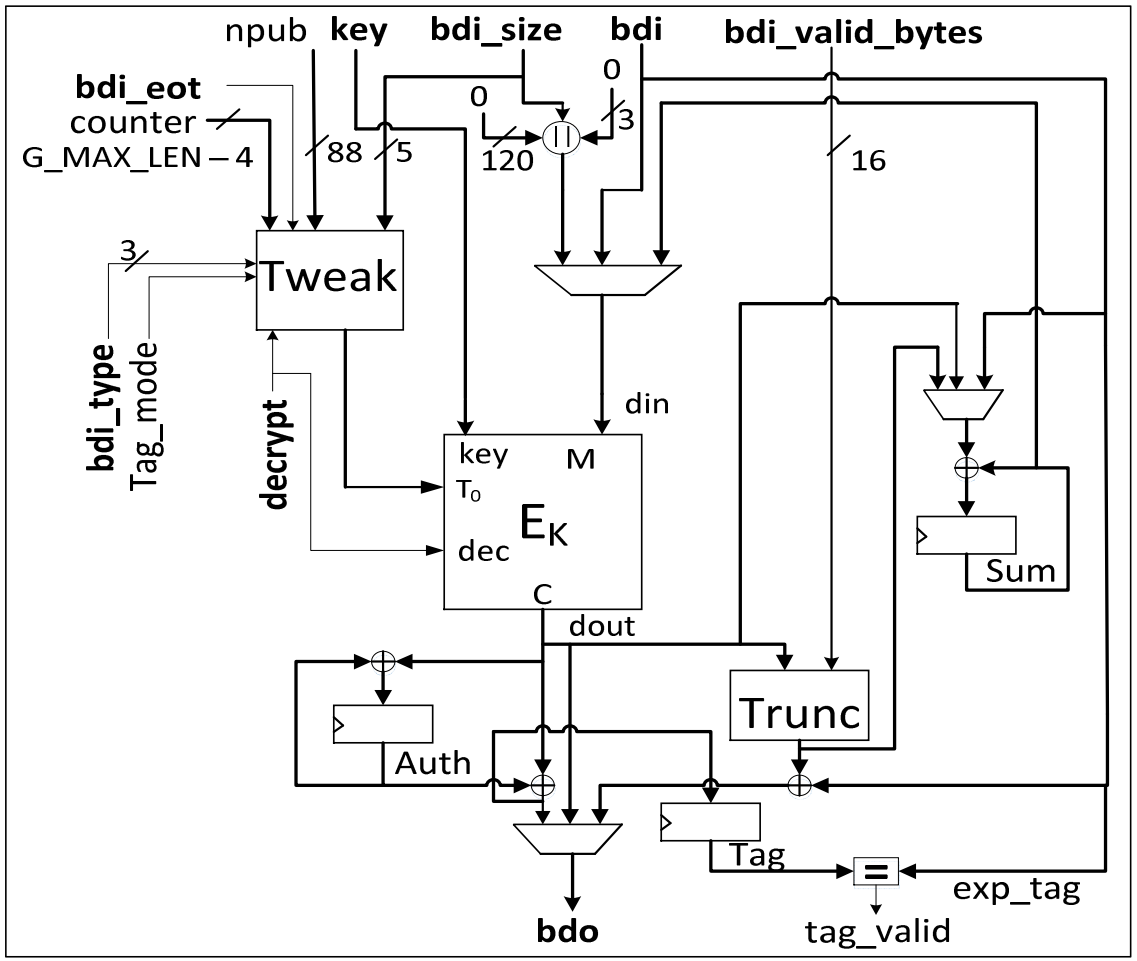
- datapath width = state size
- one clock cycle per two rounds



Typically TP/A ratio decreases

Source: "Throughput vs. Area trade-offs in High-speed Architectures of Five Round 3 SHA-3 Candidates Implemented Using Xilinx and Altera FPGAs," Homsirikamol, Rogawski, Gaj, 2011

SCREAM – CipherCore Datapath



Bus width of thick wires is 128 bits unless indicated. Bus width of thin wires is 1 bit.

Interesting Features:

- Initial Tweak = f(npub, counter, type & length of block)
- Tag = f(Auth, E_K[Sum])
- Truncation of output of partial final plaintext and ciphertext blocks

Notation:

Auth = Authenticator
Sum = Checksum
Trunc = Truncation
T0 = Initial Tweak
E_K = Tweakable Block Cipher
npub = Public Message Number
exp_tag = expected tag

POET

- Pipelineable On-line Encryption with Authentication Tag
- “Cipher Agnostic” – uses any 128-bit block cipher and ε-AXU keyed hash function
- AES-128 used for block cipher, and AES-4 used for keyed hash
- 128-bit key, state variable, tag
- Padding for Associated Data

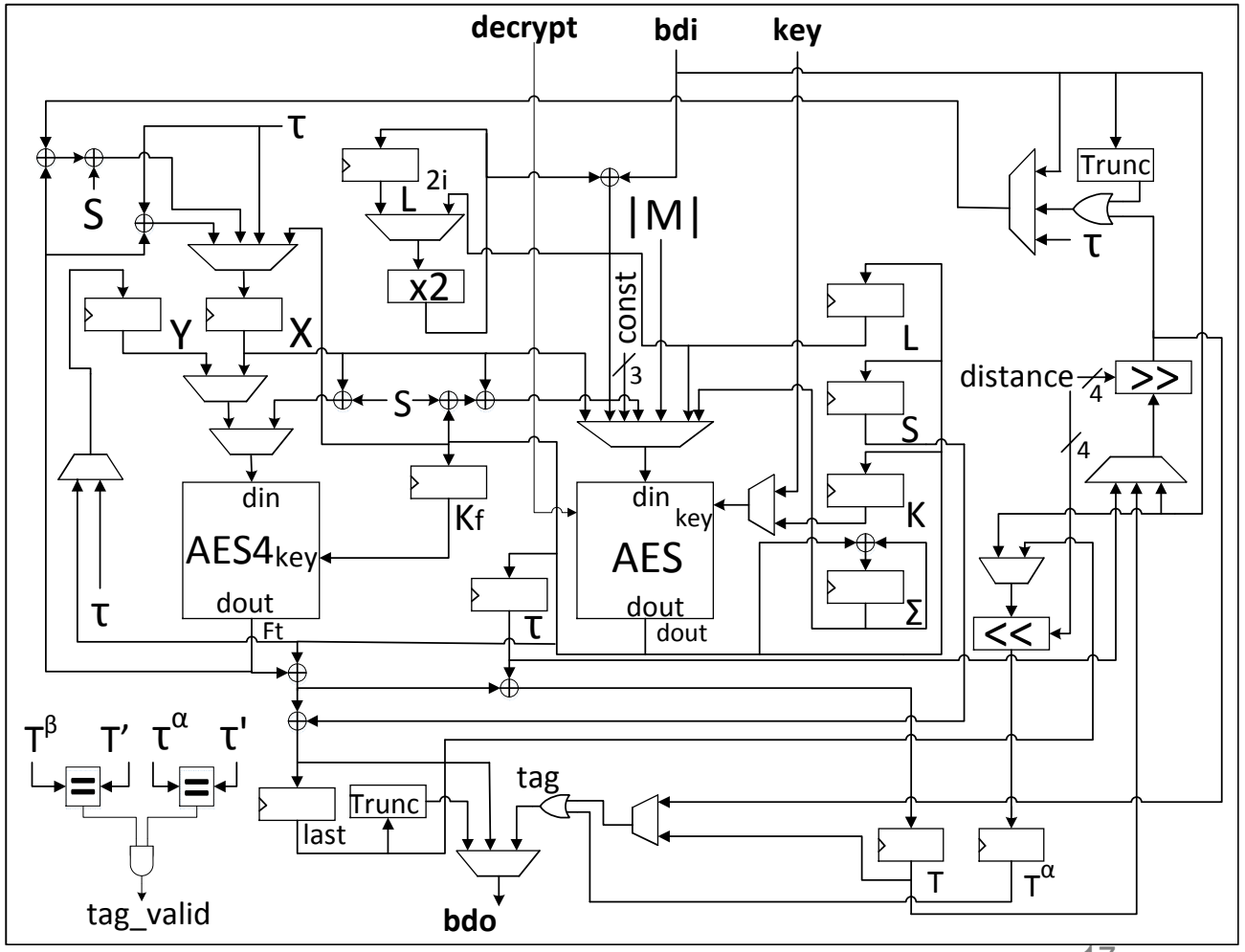
POET – CipherCore Datapath

Interesting Features:

- Requires three sub keys (L, K, Kf) and round key generation in AES
- L sub key multiplied by 2 in GF(2^{128}) during header processing
- Variable shifts for tag generation and verification

Notation:

x2 = GF(2^{128}) x2 field multiplier
τ = Authenticator
Σ = Associated Data cumulative sum
>> (<<) = Variable right (left) shift
|M| = length of message
$S = E_K(|M|)$



Bus width of thick wires is 128 bits unless indicated. Bus width of thin wires is 1 bit.

Minalpher

- Uses Tweakable Even-Mansour (TEM) with Minalpher-P primitive
- 2 TEM cores for encrypt/decrypt and tag generation in parallel
- 128-bit key, 256-bit state, 128-bit tag
- Each final plaintext block must have *10 padding even if full

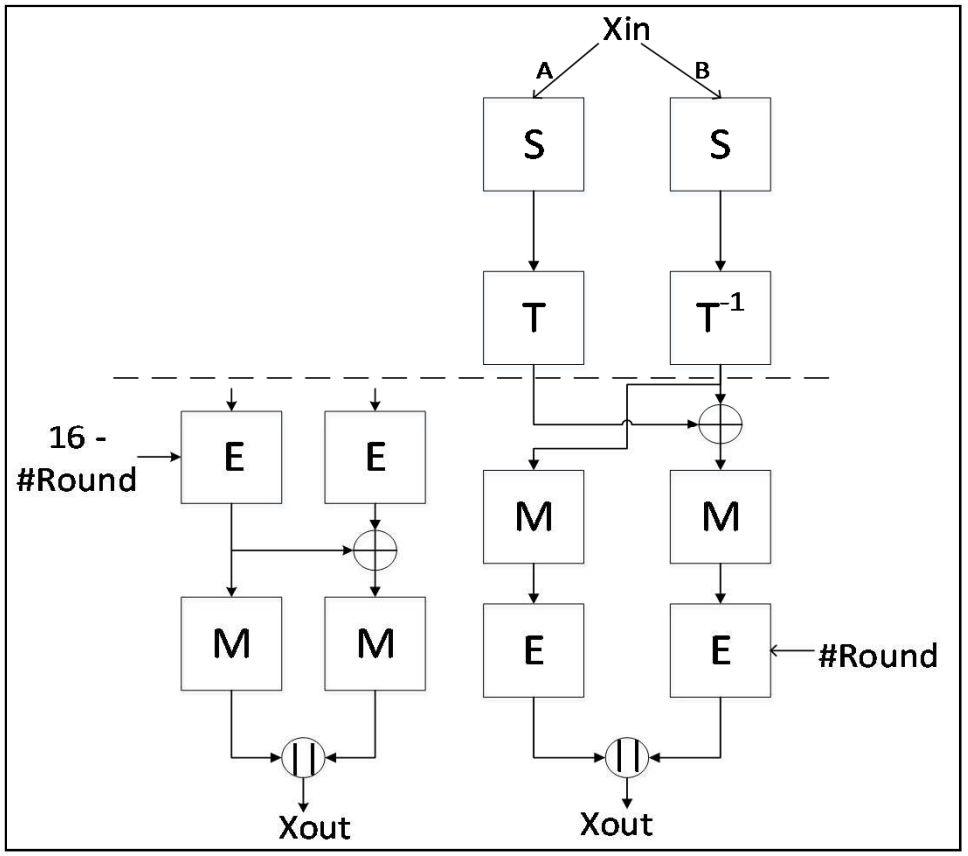
Minalpher (cont'd)

Each Minalpher-P consists of

- S (SubNibbles) 4x4 S-Boxes,
- T (ShuffleRows),
- E (Round Constant),
- M (MixColumns)
- Decryption has reversed 'E' and 'M'

17.5 rounds of Minalpher-P in one TEM

- Final "half round" is an extra S and T function



Bus widths of paths A and B are 128 bits.

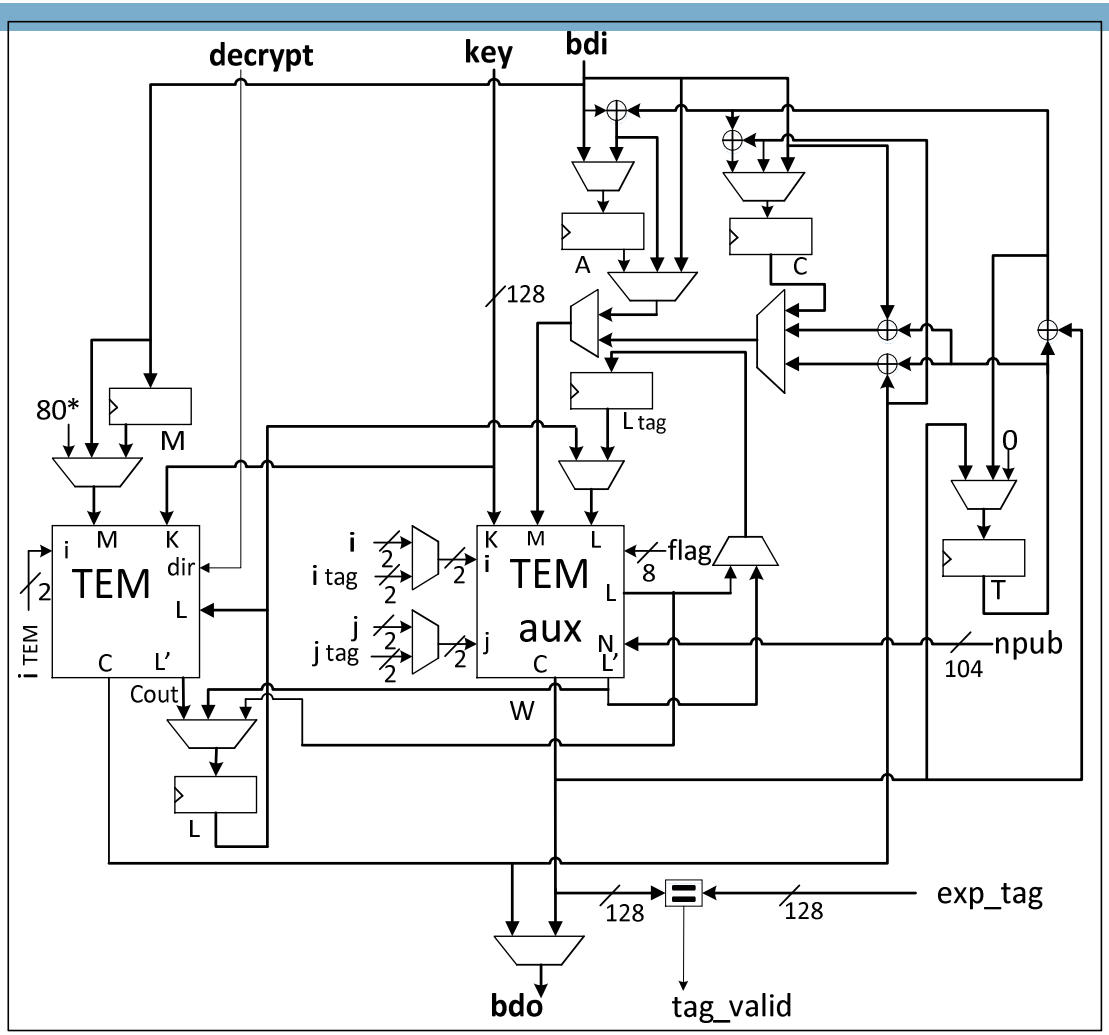
Minalpher – CipherCore Datapath

Interesting Features:

- Parallel encrypt/decrypt and tag generation using 2 TEM cores requires 19 cycles/block
- "Serial truncator" to remove final *10 during decryption

Notation:

TEM = Tweakable Even Mansour
A = Associated Data register
M = Plaintext/Ciphertext Register (TEM)
C = Plaintext/Ciphertext Register (TEM aux)
L = Block specific mask
T = Cumulative Tag generation
npub = Public Message Number
exp_tag = expected tag



Bus width of thick wires is 128 bits unless indicated. Bus width of thin wires is 1 bit.

FPGA Devices & Tools

FPGA Devices

- Xilinx Virtex-6: xc6vlx240tff1156-3

FPGA Tools:

Synthesis Tool: Xilinx XST 14.7

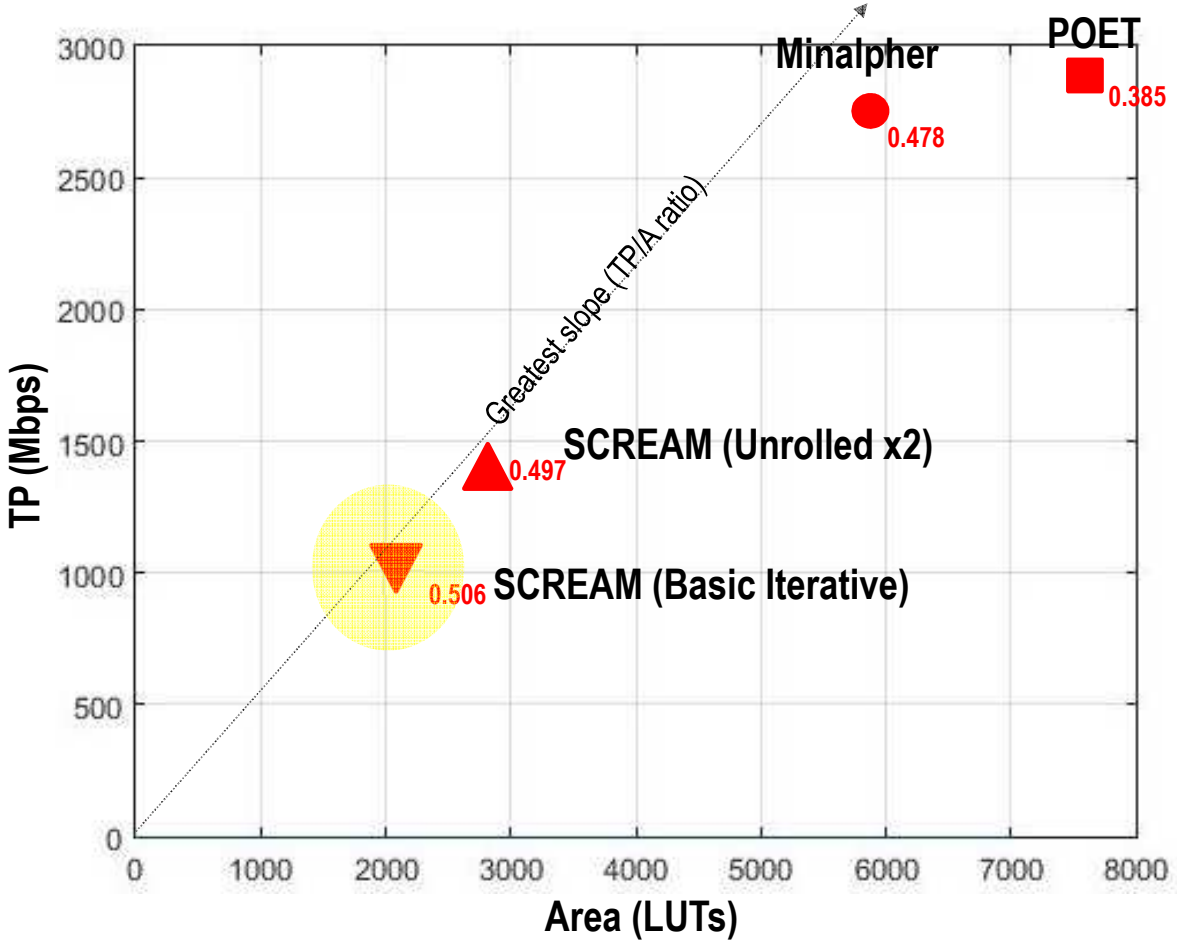
Implementation Tool: Xilinx ISE 14.7

Automated Optimization: ATHENa

Options of tools:

No embedded memories and no embedded DSP units allowed inside of AEAD

Results of GMU Implementations – Virtex 6



SCREAM has highest Throughput-to-Area (TP/A) Ratio

- Basic iterative (=1 round/clock cycle) higher TP/A than Unrolled x2

POET has high TP but large area

- Several AES cores required

Minalpher close to SCREAM in TP/A

Implementations by Other Groups

SCREAM (by Lubos Gaspar & Stephanie Kerckhof, CG UCL, INRIA)

- Full-block width custom interface
- No support for the CAESAR API Protocol

POET (by Amir Moradi, EmSec Rühr-Universität Bochum)

- Full-block width custom interface
- No support for the CAESAR API Protocol

Minalpher (by Takeshi Sugawara, Minalpher Team/Mitsubishi Electric)

- Fully compliant with the CAESAR API

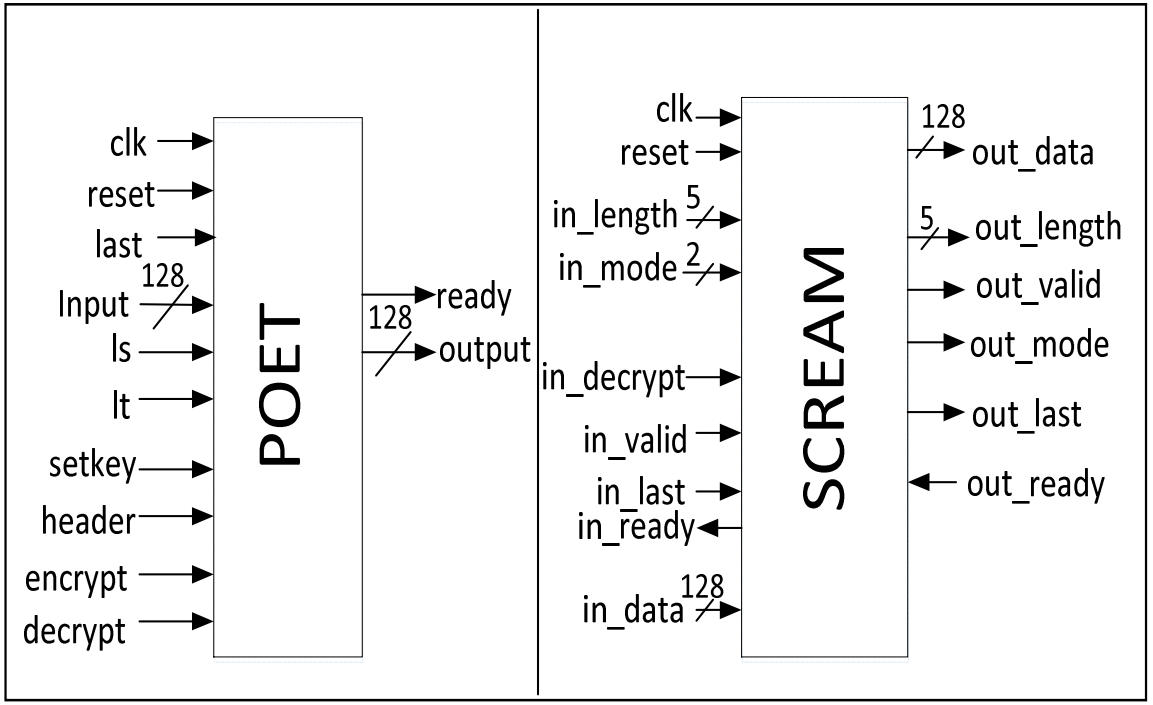
Differences in API Specifications

SCREAM

- Contains 5-bit “length” input and output fields (Allows for partial blocks)

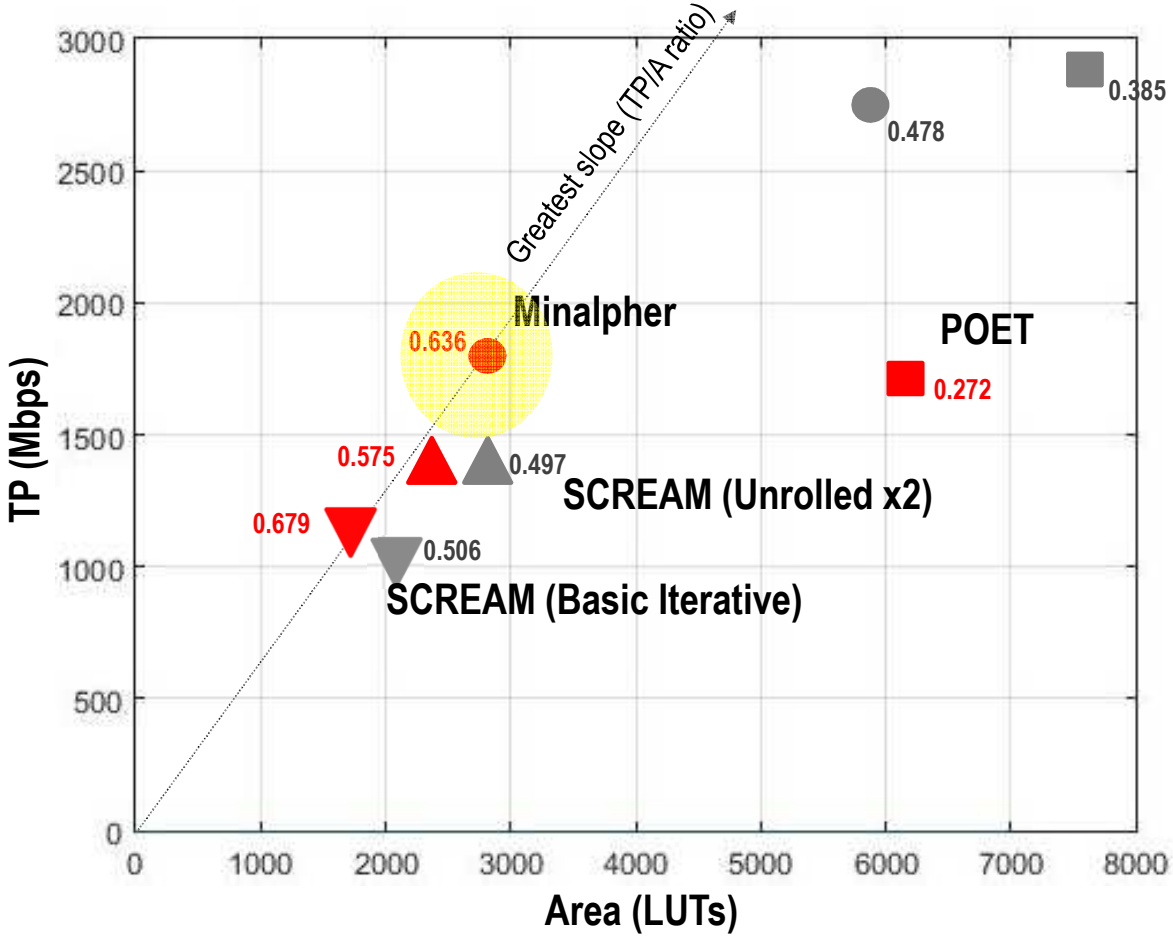
POET

- No input for length of final block (key parameter in tag generation and verification)
- Only 1 output port (Details of tag generation and verification left to higher protocol)



Derived from:: A. Moradi (2016) and S. Kerckhof, L. Gaspar (2015)

Comparison with non-GMU Implementations – Virtex 6



Minalpher has highest (TP/A) Ratio

- 1 TEM core versus 2 TEM cores in GMU design (39 cycles/block vs. 19 cycles/block)

SCREAM TP/As close to GMU

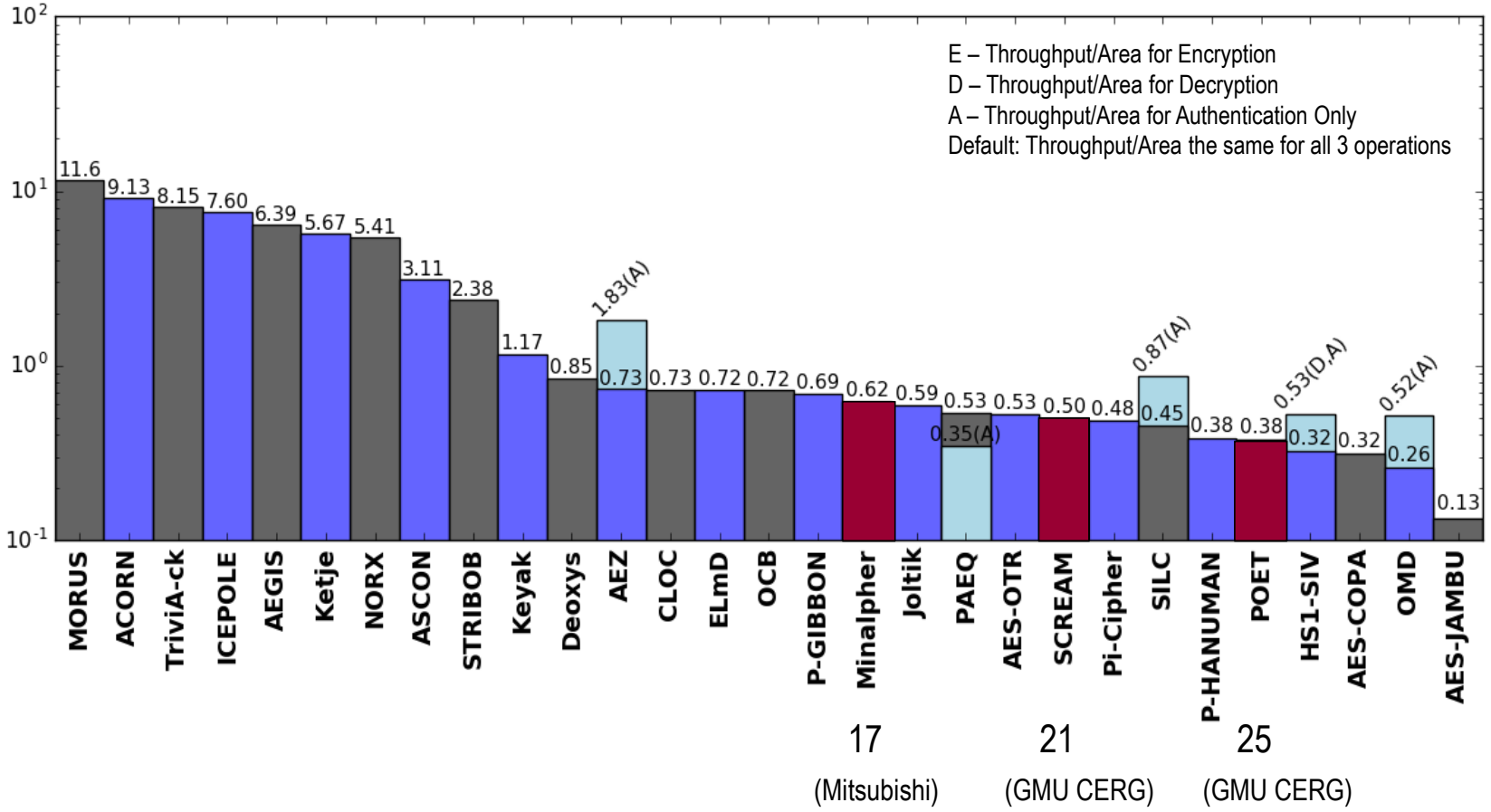
- Not compliant with CAESAR HW API

Divergent TP/A for POET

- Not compliant with CAESAR HW API
- Different choice of architecture

Comparison with all Round Two Candidates

Relative Throughput/Area in Virtex 6 vs. AES-GCM



Throughput/Area of AES-GCM = 1.020 (Mbit/s)/LUTs

Summary & Conclusions

Three functionally correct high-speed implementations of CAESAR Round Two Candidates using RTL design in CAESAR HW API

- Substantial part of GMU CERG benchmarking effort in support of CAESAR Round Two evaluations

SCREAM (w/basic iterative architecture) had highest TP/A in GMU CERG implementations

- Minalpher (Mitsubishi version) highest TP/A overall

Lessons Learned

Features of implemented ciphers negatively affecting their performance

- Variable Shifts and Truncations
- #Ciphertext blocks ≠ #Plaintext blocks

One Stop Website

https://cryptography.gmu.edu/athena/index.php?id=download
OR
https://cryptography.gmu.edu/athena
and click on CAESAR

- VHDL/Verilog Code of CAESAR Candidates: Summary I
- VHDL/Verilog Code of CAESAR Candidates: Summary II
- ATHENa Database of Results: Rankings View
- ATHENa Database of Results: Table View
- Benchmarking of Round 2 CAESAR Candidates in Hardware: Methodology, Designs & Results
- GMU Implementations of Authenticated Ciphers and Their Building Blocks
- CAESAR Hardware API v1.0

Questions?