SHA-3, R0und 2: Hardware Implementati0ns

Stefan Tillich

IAIK – Graz University of Technology
Stefan.Tillich@iaik.tugraz.at
www.iaik.tugraz.at

Overview

- SHA-3 competition, round 2
- The candidates from a HW perspective
- An attempt at a fair comparison of existing HW implementations
  - High-speed implementations
  - Low-area implementations

SHA-3 Competition

- Round 2: 50 down, 14 left
- Tweaks allowed till Sept. 15, 2009
- One year allocated for public comments
- NIST: “Hardware implementation will be addressed more thoroughly during the Round 2 analysis”
- HW implementations collected at ECRYPT2 “SHA-3 Zoo”

“Still Alive”

(c) Valve

- BLAKE
- Blue Midnight Wish
- CubeHash
- ECHO
- Fugue
- Grøstl
- Hamsi
- JH
- Keccak
- Luffa
- Shabal
- SHAvite-3
- SIMD
- Skein

The Candidates from a HW Perspective

- Concentrate on variants with 256-bit MD
- Neglect small details and fancy stuff, e.g. salting
- Main contributors to area
  - Storage
  - Compression function, message expansion function
- Functions on “critical path”
  - Dataflow within hash function

BLAKE(-32)

- Local wide-pipe design
- Storage
  - Message block: 512 bits
  - Chaining value: 256 bits
  - Inner state of compression function: 512 bits
- Compression based on G function
  - Addition mod 2^{32}, XOR, fixed rotation
  - 10 rounds à 8 G function invocations
  - Four G functions in parallel

Blue Midnight Wish(-256)

- Wide-pipe design
- Storage
  - Message block: 512 bits
  - Chaining value: 512 bits
  - Inner state (for result of f0): 512 bits
- Compression based on f_0, f_1, f_2 functions
  - Addition/subtraction mod 2^{32}, XOR, fixed shift/rotation
  - f_0: 16x 32-bit words in parallel
  - f_1: 16x 32-bit words in sequence
  - f_2: 16x 32-bit words in parallel
  - One invocation of f_0, f_1, f_2 per message block

CubeHash

- Uniform structure with high configurability
  - Message block: 8 – 1024 bits
- Storage
  - Chaining value: 1024 bit (32x 32-bit words)
- Compression based on simple rounds
  - Addition mod 2^{32}, XOR, fixed rotation, 32-bit word swap
  - Number of rounds depends on variant (1 to ∞)

ECHO(-256)

- Merkle-Damgård structure
- Message block: 1536 bits
- Storage
  - Chaining value: 512 bits
  - Inner state: 2048 bits (4 x 4 matrix of “AES states”)
- Compression based on AES transformations
  - 8 iterations of BIG.round:
    - BIG.SubWords: 16x 2 AES rounds in parallel
    - BIG.ShiftRows: Shifting of the matrix of “AES states”
    - BIG.MixColumns: 64x AES MixColumns in parallel

Fugue(-256)

- Based on Grindahl
  - Message block: 32 bits
- Storage
  - Internal state/Chaining value: 960 bits (4 x 30 bytes)
- Compression based on AES-like operations
  - 2 sub-rounds per message block:
    - Rotation & XOR of 32-bit words
    - 16x AES S-box in parallel
    - 1x Super-Mix: 16 x 16 matrix multiplication
  - Rather expensive output transformation

Grøstl(-256)

- Based on 2 parallel AES-like permutations
- Storage (each 8 x 8 bytes)
  - Message block: 512 bits
  - Chaining value: 512 bits
  - Intermediate state: 512 bits
- Compression with P & Q permutations
  - 10 rounds à 4 transformations:
    - AddRoundConstant
    - SubBytes: 64 AES S-boxes in parallel
    - ShiftBytes: Rotate 8-byte rows
    - MixBytes: 8 x 8 matrix multiplication

Hamsi(-256)

- "Concatenate-Permute-Truncate" design
- Message block: 32 bits
  - Expanded to 256 bits by linear code
- Storage
  - Chaining value: 256 bits
  - Internal state of permutation: 512 bits
- Non-linear permutation P / Pf
  - P applied 3 times per message block:
    - Addition of counter & constant
    - 128 (4 x 4) S-boxes in parallel
    - 4x linear transformation L in parallel (2^{128} -> 2^{128})
  - Output transformation: Pf applied 6 times on last block

JH(-256)

- Based on substitution-permutation network
- Storage
  - Message block: 512 bits
  - Internal state/Chaining value: 1024 bits
  - Round constant: 256 bits
- Compression based on round function R_8
  - R_8 applied 35 times on internal state & round constant
    - 256 & 64 (4 x 4) S-boxes in parallel
    - 128 & 32 linear transforms L over GF(2^4): 2^4 -> 2^4
    - Permutation of 256x & 64x 4-bit words

Keccak(-256)

- Sponge function construction
- Message block: 1024 bits
- Storage
  - Internal state: 1600 bits (25x 64-bit words)
- Round R, iterated 18 times per message block
  - XOR, AND, NOT
  - 25x bit permutation within words in parallel
  - Permutation of words
  - Addition of round constants

Luffa(-256)

- Based on round with message injection & permutation
- Message block: 256 bits
- Storage
  - Chaining value: 768 bits (3x 256-bit blocks)
- Message injection
  - Multiplication by constants in $GF(2^8)^{32}$ of 256-bit blocks (chaining value, message block)
  - Addition (XOR) of results
- Permutation: 3 Q_j (256-bit permutation) in parallel
  - 8 iterations of a step function
    - SubCrumb: 192 (4 x 4) S-boxes in parallel
    - MixWords: 12 linear 64-bit permutations (fixed rotations, XORs) in parallel
- Output transformation uses round with all-zero message

Shabal(-256)

- NLFSR-based compression function
- Storage
  - Message block: 512 bits
  - Chaining value: 1408 bits
  - Message block counter: 64 bits
- Compression function f_{Sh}
  - Addition/subtraction mod 2^{32}, XOR
  - 896-bit permutation P
    - 16 fixed rotations of 32-bit words in parallel
    - 3x 16 shifts of the NLFSR structure in sequence
      - Constant multiplication mod 2^{32}, fixed rotations of 32-bit words, XOR, AND, NOT
    - 3x 12 parallel additions mod 2^{32} in sequence
  - Output transformation: Invokes f_{Sh} 3 times

SHAvite-3($_{256}$)

- Based on HAIFA and AES
- Storage
  - Message block: 512 bits
  - Chaining value: 256 bits
  - Bit counter: 64 bits
- Compression function based on balanced Feistel network (E^{256} block cipher)
  - Function f: 3 AES rounds
  - 12 rounds
- Message expansion produces AES round keys
  - Non-linear step: 4 AES rounds, XOR with bit counter
  - Linear step: XOR previous round keys

SIMD(-256)

- FFT-based message expansion
- Storage
  - Message block: 512 bits
  - Chaining value: 512 bits
  - Intermediate state: 512 bits
- Message expansion 2^{512} -> 2^{4096} (3 layers)
  - Number-theoretic transform (FFT)
  - Inner code
  - Permutation
- Compression function
  - Uses block cipher with 4 parallel Feistel ladders (128 bits each)
    - Addition mod 2^{32}, step-dependent rotation, AND, OR, NOT
  - 32 iterations in total (128 bits of exp. message per step)
  - 4 extra steps at end

Skein(-x-256)

- Based on tweakable block cipher Threefish
- Storage
  - Message block: 256 / 512 / 1024 bits
  - Chaining value: 256 / 512 / 1024 bits
  - Intermediate state: 256 / 512 / 1024 bits
  - Expanded key & tweak: 512 / 768 / 1280 bits
- Compression / Output transf. with Threefish
  - 72 / 72 / 80 rounds
    - 2 / 4 / 8 additions mod 2^{64} in parallel
    - Round-dependent rotation, XOR, permutation of 64-bit words
    - Key expansion: 3 additions mod 2^{64} in parallel

Summary of Features

Candidate	Storage bits	„Throughput-relevant“ features
BLAKE-32	1,280	10 rounds: 2x G function in sequence; Add mod 2^{32}
BMW-256	1,536	Only one f_0, f_1, f_2 per message block; Add/sub mod 2^{32}
CubeHash	1,024	Number of rounds; Size of message block; Add mod 2^{32}
ECHO-256	2,560	8 iterations of BIG.round
Fugue-256	960	32-bit message block; 2 sub-rounds; AES S-box & Super-Mix
Grøstl-256	1,536	10 rounds; AES S-box & MixBytes
Hamsi-256	768	3x P permutation; (4 x 4) S-box; linear transform
JH-256	2,048	35 iterations of R8; (4 x 4) S-box; linear transform
Keccak-256	1,600	18x round R; Boolean functions
Luffa-256	768	8 iterations of step function; (4 x 4) S-box; linear permutation
Shabal-256	1,984	Add mod 2^{32}; 3x 16 shifts & 12 additions in sequence
SHAvite-3_{256}	832	12 rounds: 3x AES round; Message expansion: 4x AES round
SIMD-256	1,536	Number-theoretic transform (FFT) in message expansion
Skein-256-256	1,280	72 rounds; add mod 2^{64}
Skein-512-256	2,304	72 rounds; add mod 2^{64}
Skein-1024-256	4,352	80 rounds; add mod 2^{64}

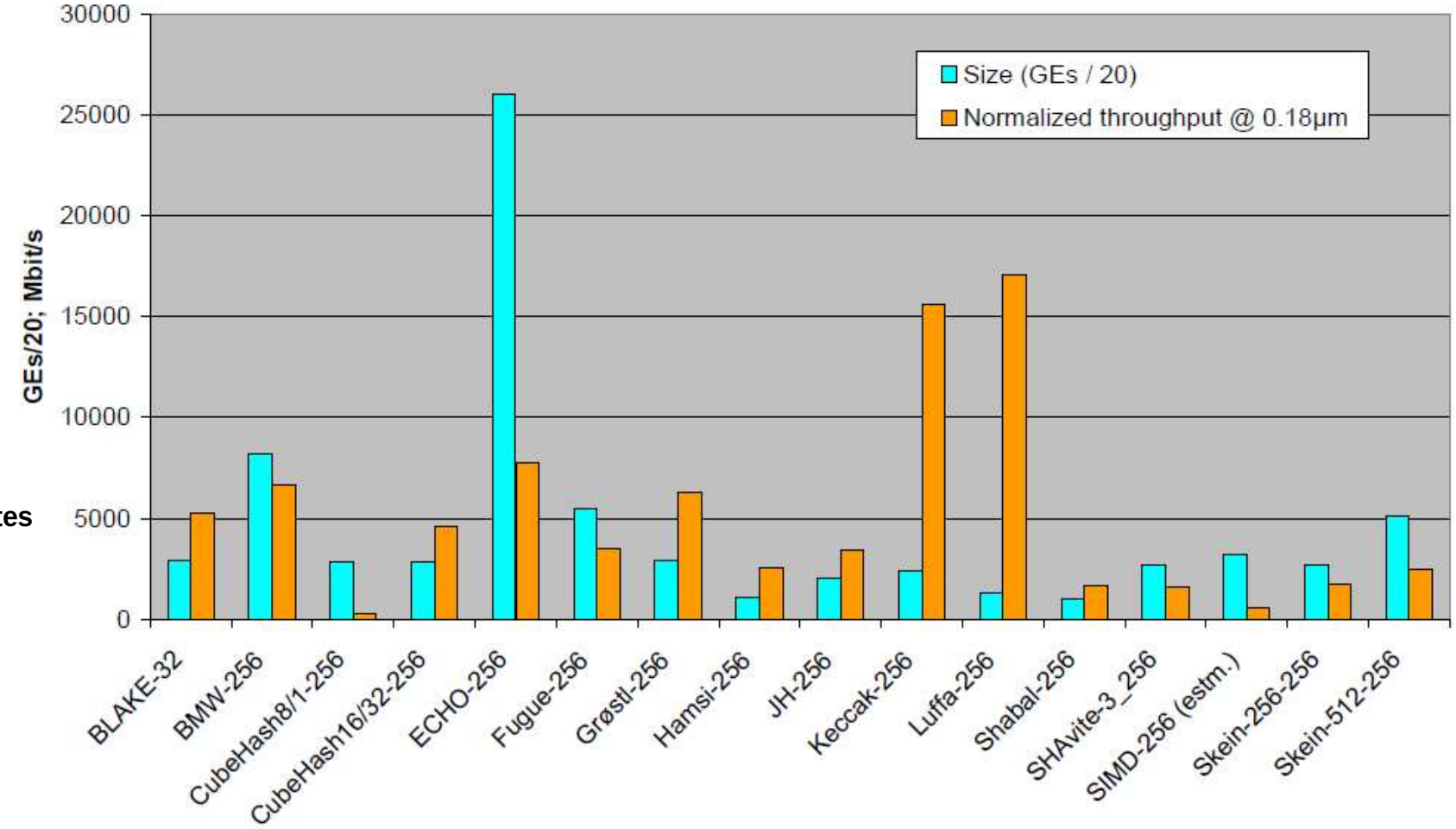
SHA-3, Round 2 Tweaks

Candidate	Round 2 tweaks (status as of Sept. 22, 22:30 CEST)
BLAKE	None
BMW	Changed inputs to f_0 and f_1; Additional invocation of compression function at end
CubeHash	2x rounds (r = 16); 32x size of message block (b = 32)
ECHO	None
Fugue	None
Grøstl	None
Hamsi	None (but additional variants specified)
JH	None
Keccak	Increased message block size (rate); increased number of rounds (18 -> 24)
Luffa	Modification of S-box; Changed order of SubCrumb inputs; Always one round in output transformation
Shabal	None
SHAvite-3	Inversion of some counter values
SIMD	Changed rotation constants and permutations for diffusion between parallel Feistels
Skein	Changed rotation constants

High-Speed HW Implementations

- Standard-cell implementations
  - Primary goal: High throughput
  - Secondary goal: Low area
- Direct comparison is problematic! Differences in
  - Implementation scope (full, ext. memory, core)
  - HW module interface
  - Process technology
  - Standard-cell library
  - Implementer's design experience & effort
  - Synthesis effort
  - ...
- Using very rough scaling to uniform minimal gate length (0.18 µm)

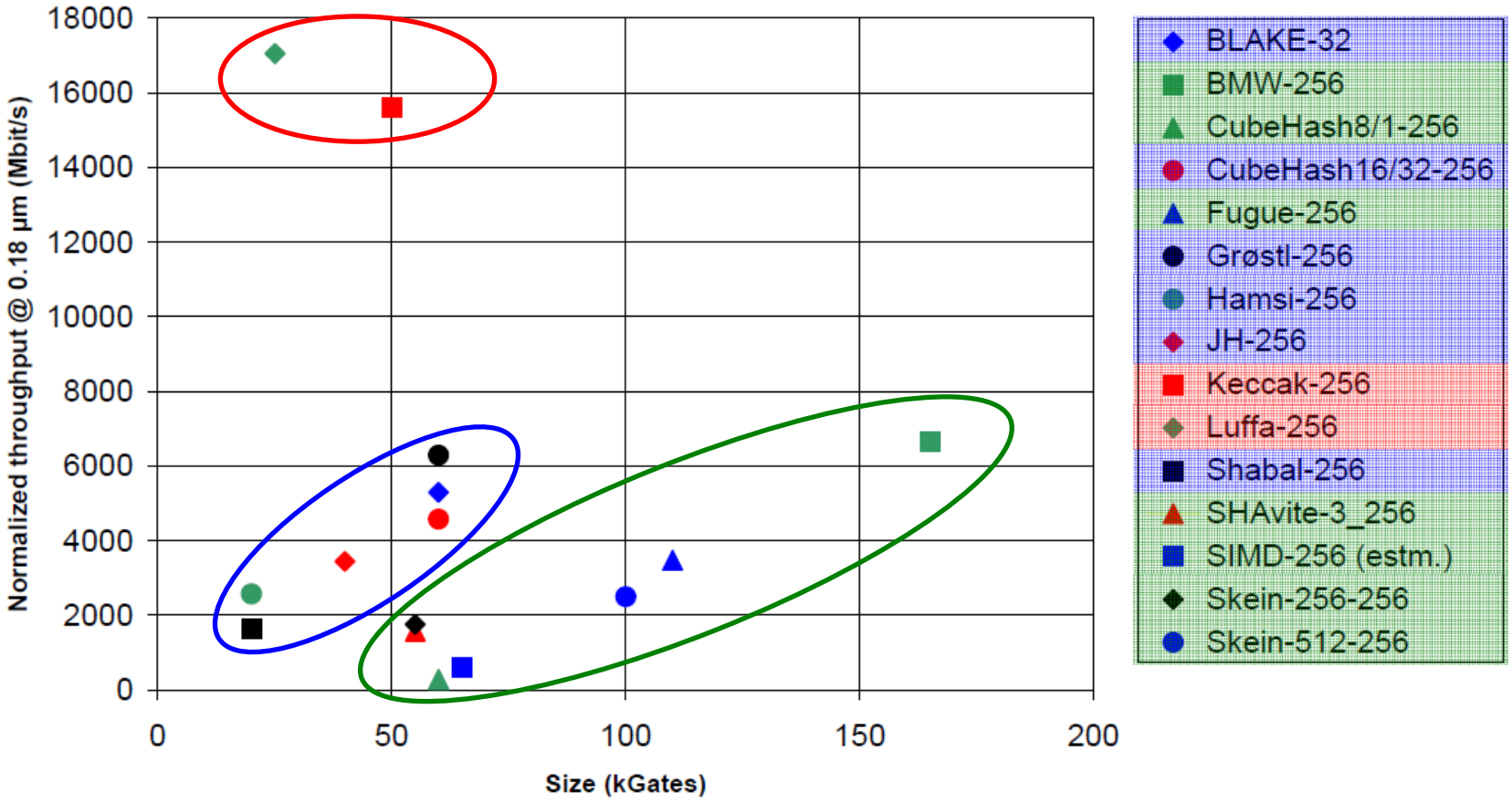
High-Speed HW Implementations



100 kGates

- Throughput scaled with square of the minimal gate length to 0.18 µm
- Results for CubeHash, JH, SHAvite-3, SIMD preliminary and pending publication
- BLAKE, BMW, and Shabal: Core functionality only

High-Speed HW Implementations

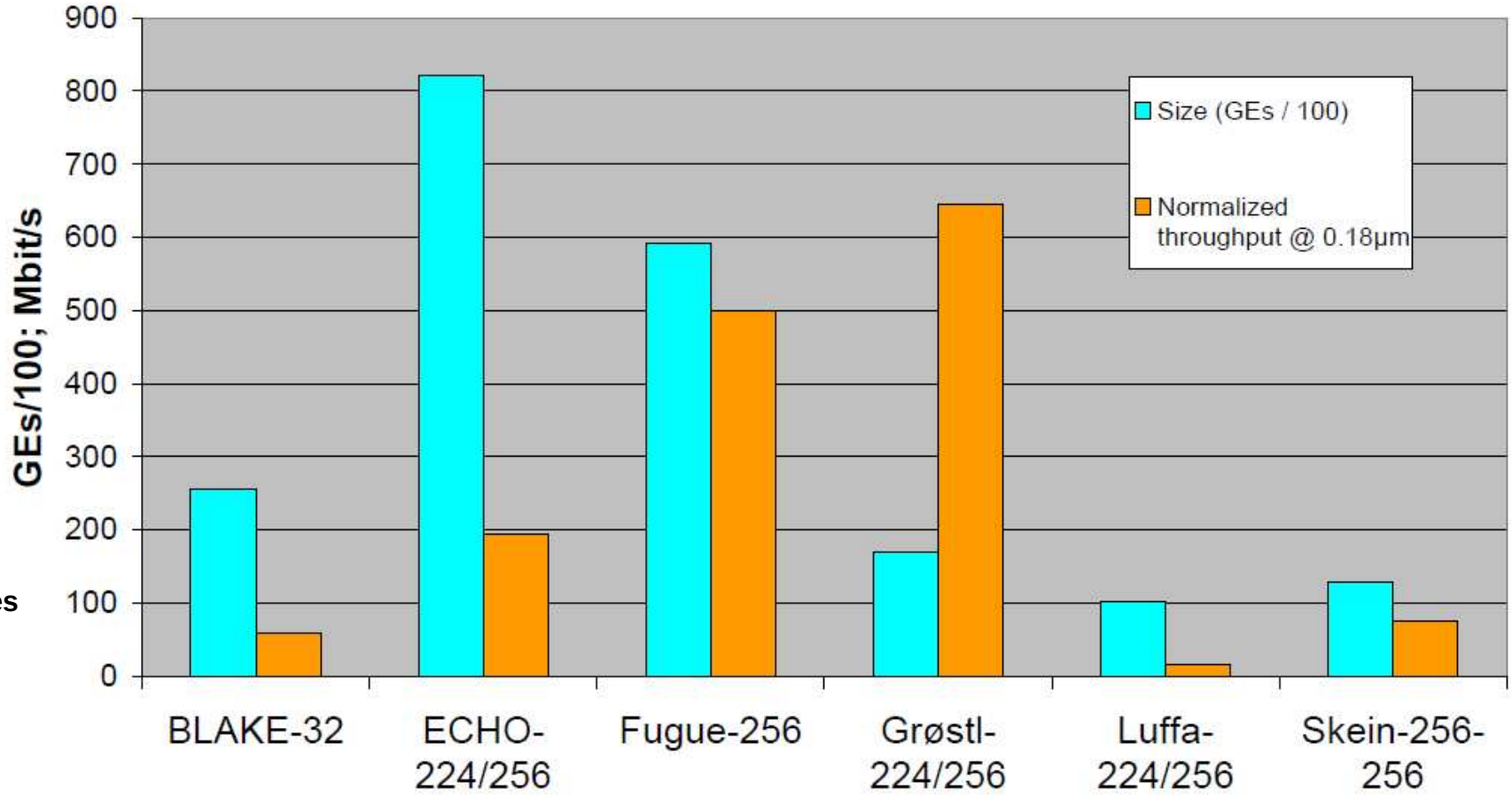


- Throughput scaled with square of the minimal gate length to 0.18 µm
- ECHO-256 omitted: ~ 7750 Mbit/s with 520 kGates
- Results for CubeHash, JH, SHAvite-3, SIMD preliminary and pending publication
- BLAKE, BMW, and Shabal: Core functionality only

Low-Area HW Implementations

- Primary target: Minimal area and/or power/energy consumption
- Raw throughput of lesser importance
- Comparison only of fully autonomous implementations
- Only 6 candidate implementations available so far
- To be taken with a grain of salt!
  - Some implementations not totally optimized towards low-area

Low-Area HW Implementations



10 kGates

- Throughput scaled with square of the minimal gate length to 0.18 µm
- Comparison includes only fully autonomous implementations

All the Figures ...

- ... on the pages of the ECRYPT2 “SHA-3 Zoo”: http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo
- All reported ASIC & FPGA implementations
- Frequent updates for the candidates

Thanks for your attention!

Questions?

(c) Valve

Shoot!